Chapter 1. Is There a Security Problem in Computing?

IDOL

1.1. What Does "Secure" Mean?

How we can protect are valuable assets? One of the safest place our money in bank in term of cash and gold jewelry. In old day most of the money in banks are kept as cash instead of check, is not easily traceable. Bank would protect our assets from robbery by keeping the asset in lockers and provides security by keeping guardsoutside. In olden days , communication and transportation was primitive enough , the legal administrative officer would informed the police about the robbery in the bank , the security personals would reach the site very late that time already the criminal escape from the bank . It would be very difficult to trace the criminal .That time the bank robbery is profitable business. So protecting our assets was very difficult and not always very effective.

But today,assets protection becomes very easier, with many effective technique against the criminals, by means of putting sophisticated alarm, CCTV place in the bank premises by tracking at activity of peoples inside and outside of the banks. The technique of the criminal investigation have become more effective in terms by taking the finger print , DNA , retinal pattern , iris recognition , voice of the person and person can be identify above the mentioned properties . The assets would be store much safer form for an example, Many banks now contains less cash than some retail stores because much of the bank's business is done though the check ,electronic transfers, credit cards , debit cards and so on. In Banks sites, large amount of hard cash and currency are stored in many layer of security levels: several physical layers, many complex locks are implement and multiple system party are required among the several peoples to allow to access the assets. There are significant improvements in transaction and communications mean that the police would reach the crime site in a minutes. Sophisticated alarm and CCTV would dispatch the alert to other officers in seconds about

IDOL

the suspect to watch for. From the criminal point of views, it very difficult for robbers to commit crime in the banks.

Protecting Valuables

This books is not dedicated how to protect the money or gold jewelry, it dedicated how to protect the computer resources. Form analysis of banks, we have learned some basic principle of protection. In other words, when we learn about the protecting the valuable information, we learn lot about the protecting the other valuable resources. The Table 1.1 give the difference between how people protect the computing resource and how bank protect the money.

Table 1-1. Protecting Money vs. Protecting Information.

Characteristic	Bank Protecting Money	People Protecting Information
Size and portability	Banks sites stores money are large , not all portable, Building are required to store, Guards are requires, vaults , Many levels of physical security to protect the money	Items storing valuable assets are small and portable. The physical device are required to protect the valuable information are so small, it contains of thousand rupees.
Ability to avoid physical contact	Difficult, when banks deals with physical money or currency, a criminal can demand the money and steal the money physical from the bank's premises.	Simple. When the information handle electronically, physical contact are not necessary. Instead banks handled the money electronically ,

IDOL

		transaction are done without any physical contact .Here money can be transferred through the computers, mobile , telephone and email
Value of assets	Very High.	Variable., It can be from high top low or vice versa, some sensitive information like medical history, taxpayment, education background are kept confidential , some of the information like troop movement , sales strategies are very sensitive information, still other information like phone number and address may be have no consequence and can access by other means

You can develop an understanding of the basic problems underlying computer security and the methods available to deal with them.

IDOL

In particular, we do the following:

- examine the risks of security in computing
- consider available countermeasures or controls
- stimulate thought about uncovered vulnerabilities
- identify areas where more work is needed

In this chapter, we will be examining different kinds of vulnerabilities computing systems are more prone to systems. What are reason for vulnerabilities for exploiting: the different kinds of attacks that are possible to happen in the system the kinds of people are involved or contribute in the security? Finally, we introduce how to prevent possible attacks on systems.

Characteristics of Computer Intrusion

Criminal can target to any part of a computing system to commit the crime. When we said about**computing system**, it's mean a collection of hardware, software, storage media, data, and people .the following resources are used by the organization for computing the task. Sometimes, the organization is least bother about the computing resource and consider the resources are not valuable to an outsider, they make mistake for not consider valuable assets. For an example, people consider most valuable property in a bank is the cash, gold, or silver in the bank's vault. But the people forget the most valuable is the customer informationkept in the bank's computer. The bank's customer's information is stored on paper, recorded on a storage devices like tape drives, hard disk or information may resident in memory, or transmitted over telephone lines or satellite links, and this information can be used many way for making a money illicitly. A competitor bank can use this information to steal the details of the customers or even to disrupt service and discredit the bank. An anonymous user or hacker could move money from one account to another bank account without the permission of owner. A group of impostor could contact large depositors and convince them to invest in fraudulent schemes. The variety of targets and attacks makes computer security very difficult

IDOL

Any system is consider more vulnerable than it has weakest point, for example a robber intention was to steal something from the house if a window gives him as easier access instead of two thick metal door for penetration . We can codify this idea as one of the principles of computer security.

The principles of computer system should be consider by security specialists by means different possible ways of penetration in to the system. Whenever the security parameters change, according to the policy of an organization the penetration analysis must repeatedly scan the vulnerability in the systems. Sometimes, the People underestimate the determination or creativity of attackers. It has to remember that computer security is a key roles for the defending team only its means , The attackers can (and will) use any technique for penetrating into the system .The security specialists has to think out of box , how to prevent the system form the attackers .

1.2 Attacks

When we test any computer system in the organization, the main jobs of ours to imagine that how the system could malfunction. Then ours responsibility to improve the system's design so the system can withstand any of the problems we have identified. In the same way, computer security specialist analyze a system from a security perspective, thinking about different ways in which the system's security can malfunction and diminish the value of its assets.

1.2.1 Vulnerabilities, Threats, Attacks, and Controls

A computer system has three separate valuable components: **hardware**, **software,** and **data**. Each valuable assets offers different values to different members of the community which are affected by system. The security analyst has to do different ways of brainstorming about the system or its information can leads to some kind of loss or harm to the system. For example, security analyst has to understand what can be data format or what kind of data contents should be protected in different way. Security analyst want the system secure such a way there should sure be data should not disclosed to unauthorized parties. He has to ensure that the data should be modified in illegitimate ways. At the same time, he

must ensure that legitimate users have access to the data. In this way, we can identify weaknesses in the system.

What is a vulnerability?

According to the Definition "the quality or state of being exposed to the possibility of being attacked or harmed, either physically or emotionally."

In the security term, a**vulnerability** is a weakness or flaw in the security system, for example, while in designing, procedure, or implementation of any system in term of application that might be exploited to cause loss or harm. For instance, a particular system may be vulnerable to people for accessing an unauthorizeddata, unauthorized people would manipulate the data because the system is failed to the verification of his / her identity and allow them to access unauthorized data.

What is threat?

According to the Definition "a statement of an intention to inflict pain, injury, damage, or other hostile action on someone in retribution for something done or not done".

A threat, in the context of computer security, refers to anything that has the potential to cause serious harm to a computer system. Let see the difference between a threat and a vulnerability, consider the illustration in Figure 1-1.

In the figure 1.1 , As a wall is holding a water its back ,so the water on the left side of the wall is a threat to the man stand on the right side of the wall, If the water could rise above the wall level, it could overflowing onto the man, or it could stay behind the wall, due to amount water pressure applied on the wall which cause the wall to be collapse. So it leads to threat of harm is for the man to get wet, get hurt, or be drowned. For now, the wall is intact, so the threat to the man is unrealized.

IDOL

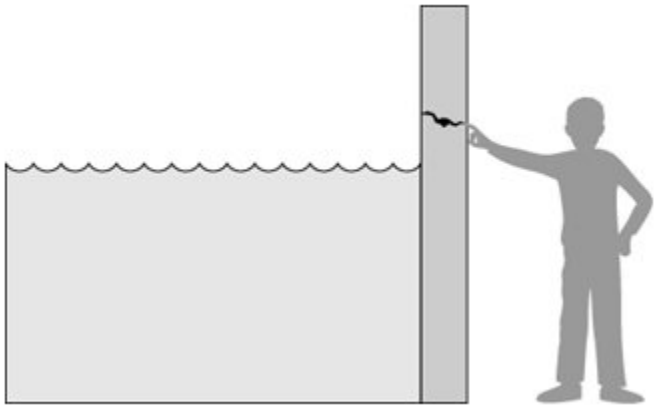


Figure 1.1. Threats, Controls, and Vulnerabilities.

However, we can see a small crack appear in the wall, it leads to a vulnerability that would threatens for security of man. If the water level rises beyond the level of the crack, it could exploit the vulnerability and cause a harm to man.

There are ample of threats to a computer system, including human- intervention and computer- intervention. We all are experience human errors while designing an application, flaws in hardware designing, and software failures. But there is an also natural disasters as threats, that it can bring a system down when the computer room is flooded or the data center collapses from an earthquake.

A human who cause an exploits can leads to a vulnerability perpetrates an **attack** on the system. An attacker might launched an attack from another system by sending an overwhelming set of messages to another system, it lead to virtually shutting down system's ability to function. Such attack known as denial-of-service attacks, which flood servers with more messages so the system will not handle the message so the system could function properly.

How do we solve the problems of vulnerability and threat? By means of control as a protective measure. A control is an action, device, procedure, or technique that removes or reduces a vulnerability in the system

In Figure 1.1, the man is placed his finger in the hole, it could control the threat of water being leaks until he cold finds permanent solution to the problem. In general, we can describe the relationship among threats, controls, and vulnerabilities in this way:

A threat is blocked by control of a vulnerability.

To invent the plan control, we must know as much as possible threat in the systems, we can view four way of threat in the system: interception, interruption, modification, and fabrication. Each threat could exploits vulnerabilities of the assets in computing systems; the threats are illustrated in Figure 1.2.

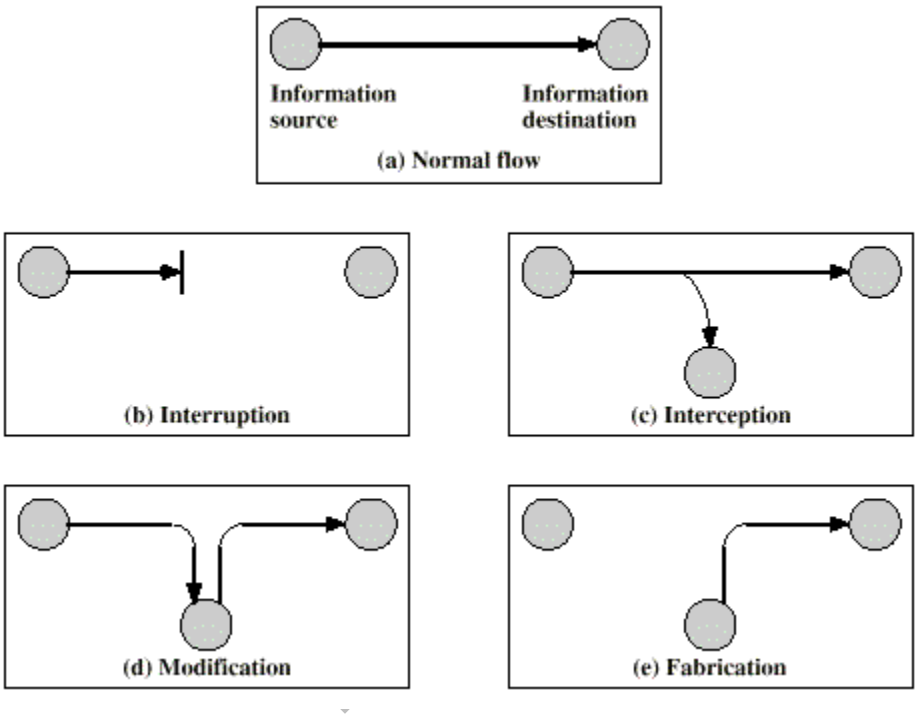


Figure 1.2 Types of System Security Threats.

a) Normal flow : An entity or person is send an information from source systems to destination system

b) An interruption, an asset of the system becomes lost, unavailable, or unusable. An example is malicious destruction of a hardware device, erasure of a program or data file, or malfunction of an operating system file manager so that it cannot find a particular disk file.
c) Interception: An interception means that some unauthorized party has gained access to an asset. The outside users can be a person, a program, or a computing system. Examples of this type of failure are illicit copying of program or data files, or wiretapping to obtain data in a network.
d) Modification: some unauthorized user not only access the unauthorized data, but also modified the data. For example hacker can modified the data value in the databases.it may alter the program computation part, it additionally perform the computation apart. The data can electronically modified, some hardware can also be modified.
e) Fabrication: An unauthorized party inserts counterfeit objects into the system. This is an attack on the authenticity. Examples include the insertion of spurious messages in a network or the addition of records to a file

Method, Opportunity, and Motive

A malicious attacker must have three things:

Method: the skills, knowledge, tools, and other things with which to be able to pull off the attack

Opportunity: the time and access to accomplish the attack

Motive: a reason to want to perform this attack against this system

1.3The Meaning of Computer Security

The main aim of computer security is to provide many ways to prevent the flaw, weakness from being exploited. In order to understand what precaution measure has to take at most it make the sense to say whether a system is "secure."

Security Goals

In our daily lives, we the word “security" in many ways. A "security system" key word came in the picture when we want to protect our house from intruders, if intruders tries to get in our house, we warning our neighbors or try to contact the police nearby in our locality. "Financial security" is the word say our involvement in different set of investments that are adequately funded; so we further hope our investments will grow in value over period of time so that we have enough money to survive later in life. Then we speak of children's "physical security," hoping that our children are safe from potential harm. Just as above terms is use for the specific domain, so we too does the phrase "computer security."

When we say about the term as “computer security”, it mean that we are addressing three important principle of any computer-related security i.e. **Confidentiality**, **integrity,** and **availability.**

- The term “Confidentiality” mean that to ensure about computer-related assets are accessed only by authorized users. That is, only those person or an entity should have access to something will have a right to access the things. The term "access," are not meant only reading but also for viewing, printing, or simply knowing whether a particular assets is exist or not. Confidentiality is sometimes called **secrecy** or **privacy**.
- The term “Integrity” means that particular assets can be modified or tampered only by authorized user only in authorized ways. The context regarding modification includes writing, changing, changing status, deleting, and creating of assets.
- The term “Availability” means that assets are accessible to only by an authorized users at particular times. In other words, if some person or system have an access to legitimatesystems or a set of objects, then access should not be prevented for the authorized users. For this reason, availability is sometimes known by its opposite, denial of service.

Security in computing addresses the three challenges. One of the challenges is to build a secure system to finding the right balance for achieving the goals, which often conflict. For example, it is very easy to protect a particular object's confidentiality in a secure system simply by not allowing everyone from reading that object. However, this system is unsecure, because it does not meet

thestandard requirement of availability for proper accessing the object. So there should be a balance between confidentiality and availability.

But balance is not meant at all ,In fact, these three characteristics can be independent from each other, it can be overlap (as shown in Figure 1.3), and can be mutually exclusive even . For example, we have seen the above example stating about strong protection of confidentiality which can severely restrict availability for the system. Let us we goes in depth of each of the three qualities.

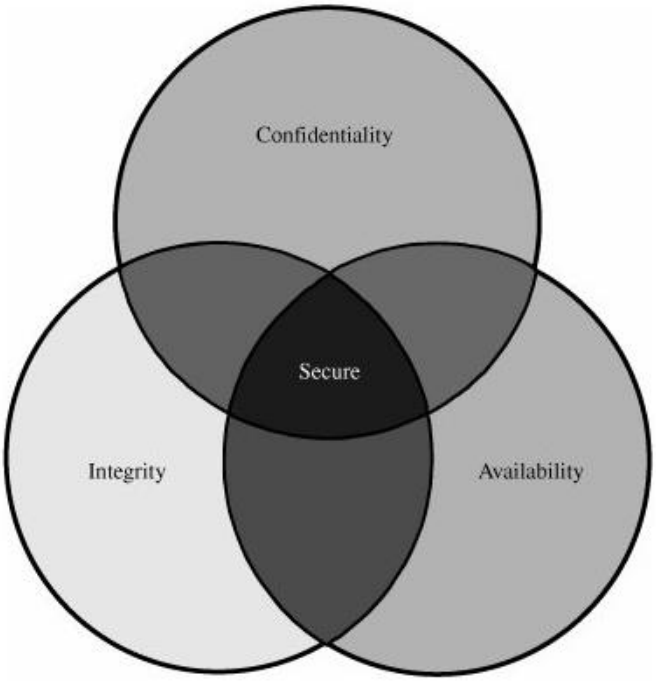


Figure 1.3. Relationship between Confidentiality, Integrity, and Availability.

Confidentiality

You may find the concept of confidentiality is to be straightforward: its say that only authorized people or systems can access protected data. However, have seen later in the chapter, that it is very difficult to ensure about the confidentiality. For example, it is very difficult to decide who is authorized person or system to determine which people or systems are authorized to access the current system? By "accessing" it difficult to determine whether an authorized user have access a single bit or the whole collection or pieces of data out of

context.It also difficult to determine can someone who is authorized to disclose those data to other parties?

Confidentiality is roughly equivalent to privacy. Measures undertaken to ensure confidentiality are designed to prevent sensitive information from reaching the wrong people, while making sure that the right people can in fact get it: Access must be restricted to those authorized to view the data in question. It is common, as well, for data to be categorized according to the amount and type of damage that could be done should it fall into unintended hands. More or less stringent measures can then be implemented according to those categories.

Integrity

The Integrity word say that to ensure that data should be correct or accurate and it should not be change during the transmission. One type of security attack could be interceptor intercept some important data and make changes to it before sending it on to the intended receiver.

Availability

Availability on of the important principle in security, it ensure that the information should be readily available to the authorized user all the times. Some types of security attack could deny access to the appropriate user For example, by breaking the web site for a particular search engine, a rival may become more popular.

Vulnerabilities

When we decide to test a system, we usually try to find out how the system can fail; we then find out the different ways in which the requirements, design, or code can leads to failure of the systems. In the same way, when we prepare to specify, design, code, or test a secure system, we try to find out different types of vulnerabilities that would prevent us from reaching one or more of our three security goals.

It could sometime find that it is easier to consider that vulnerabilities could apply to all three broad categories of system resources (hardware, software, and data), rather than to start with the security goals themselves. Figure 1.4 shows the types of vulnerabilities that could apply to the assets of hardware, software, and data.

These three assets and the connections among them are all potential security weak points. Let us look in turn at the vulnerabilities of each asset.

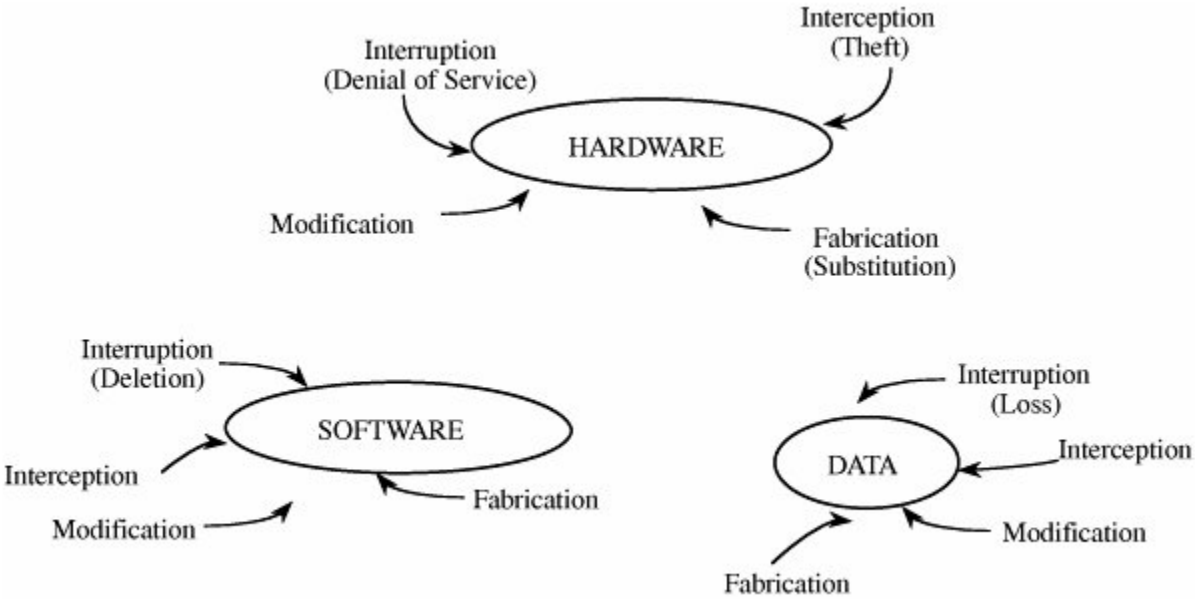


Figure 1.4. Vulnerabilities of Computing Systems.

Hardware Vulnerabilities

A hardware vulnerability is a flaw in a computer system that enables attacker attack the system hardware through remote or physical access.

The attack could be adding device, removing the devices, changing the devices, it could intercept the traffic to them, flooding the traffic until the functionality of the system become no longer.

There are other means of attack to the computer hardware physically. Computers have been drenched with water, burned, frozen, gassed, and electrocuted with power surges. People cloud spilled soft drinks, corn chips, ketchup, beer, and many other kinds of food on computing devices. Mice have chewed through cables. Particles of dust, and especially ash in cigarette smoke, have threatened precisely engineered moving parts. Computers have been kicked, slapped, bumped, jarred, and punched. Although such attacks might be intentional, most are not; this abuse might be considered "involuntary machine slaughter": accidental acts not intended to do serious damage to the hardware involved.

Another type of hardware vulnerability is an unexpected flaw in operation that allows attackers to gain control of a system by elevating privileges or executing code. These vulnerabilities can sometimes be exploited remotely, rather than requiring physical access.

Software Vulnerabilities

A software vulnerability can be seen as a flaw, weakness or even an error in the system that can be exploited by an attacker in order to alter the normal behavior of the system

A vulnerable software system can be exploited by attackers and the system could be compromised, the attacker might take control of the system to damage it, to launch new attacks or obtain some privileged information that he can use for his own benefit

A vulnerability in IIS, detailed in Microsoft Security Bulletin MS01-033, is one of the most exploited Windows vulnerabilities ever. A large number of network worms have been written over the years to exploit this vulnerability, including 'CodeRed'. CodeRed was first detected on July 17th 2001, and is believed to have infected over 300,000 targets. It disrupted a large number of businesses, and caused huge financial losses around the world. Although Microsoft issued a patch for the vulnerability along with the MS01-033 security bulletin, some versions of the CodeRed worm are still spreading throughout the Internet.

Software Deletion

Software is can easy to be delete. Each of us in our careers, accidentally erased a file or saved a bad copy of a program, destroying a good previous copy. So software's becomes high value to a commercial computing center, so accessing to software is usually carefully handled or controlled through a process called configuration management so that software cannot be deleted, destroyed, or replaced accidentally. Configuration management uses several techniques to ensure that each version of software or release retains its integrity. Whenever the new software is released the configuration management thoroughly tested to

verify that the improvements work correctly without degrading the functionality and performance of other functions and services, then the old version or release can be replaced with a newer version only .

Software Modification

Software is can become an exploitable vulnerable to system when there is modifications in the software which cause it to fail the systems or it cause to perform an unintended task. As, a software is become more susceptible to one errors, it is quite easy to modify to it. Even changing a bit or two in software can convert a working program into a failing one. It depend upon on which bit was changed, the program may crash when it begins or it may execute for some time before it falters.

Even more change in the software can leads to an error in the working of the software. In most of the time, program works well but fails in specialized circumstances. For instance, the software can be maliciously modified so that the system could failed when certain conditions are met or when a certain date or time is reached, such delay in the effect , such a program is called as a **logic bomb.** For example, an angry employee may modify a crucial program so that it accesses the system date and halts abruptly after August 1. The employee might quit on June l and plan to be at a new job miles away by August.

Other categories of software modification include

Trojan horse: a Trojan horse is a program that appears harmless, but is, in fact, malicious. Unexpected changes to computer settings and unusual activity, even when the computer should be idle, are strong indications that a Trojan is residing on a computer.

Virus: A computer virus is a malicious program that self-replicates by copying itself to another program. In other words, the computer virus spreads by itself into other executable code or documents. The purpose of creating a computer virus is to infect vulnerable systems, gain admin control and steal user sensitive data.

Trapdoor: A computer trapdoor, also known as a back door, provides a secret -- or at least undocumented -- method of gaining access to an application, operating system or online service.

Information leaks in a program: code that makes information accessible to unauthorized people or programs

Software theft

Software theft means the unauthorized or illegal copying, sharing or usage of copyright-protected software programs. Software theft may be carried out by individuals, groups or, in some cases, organizations who then distribute the unauthorized software copies to users.

Software theft is committed when someone performs any of the following:

Steals software media

Deliberately erases programs

Illegally copies or distributes a program

Registers or activates a software program illegally

Data Vulnerabilities

Security data, especially vulnerability data, have many concepts that translate nicely from the software quality realm. Vulnerabilities can be tracked in the same way as bugs, e.g., using modern issue tracking systems. Vulnerabilities manifest themselves as design flaws or coding mistakes in the system, much like bugs. However, the malicious nature of their use and the conceptual difference of preventing unintended functionality means that any analysis of vulnerabilities are subject to a variety of caveats.

When it comes to data security, a threat is any potential danger to information or systems. Threats could be an intruder network through a port on the firewall, a

process accessing data in a way that violates the security policy, a tornado wiping out a facility, or an employee making an unintentional mistake that could expose confidential information or destroy a file's integrity.

Data security suggests the second principle of computer security.

Principle of Adequate Protection: Computer items must be protected only until they lose their value. They must be protected to a degree consistent with their value.

Figure 1.5 illustrates how the three goals of security apply to data. In particular, confidentiality prevents unauthorized disclosure of a data item, integrity prevents unauthorized modification, and availability prevents denial of authorized access.

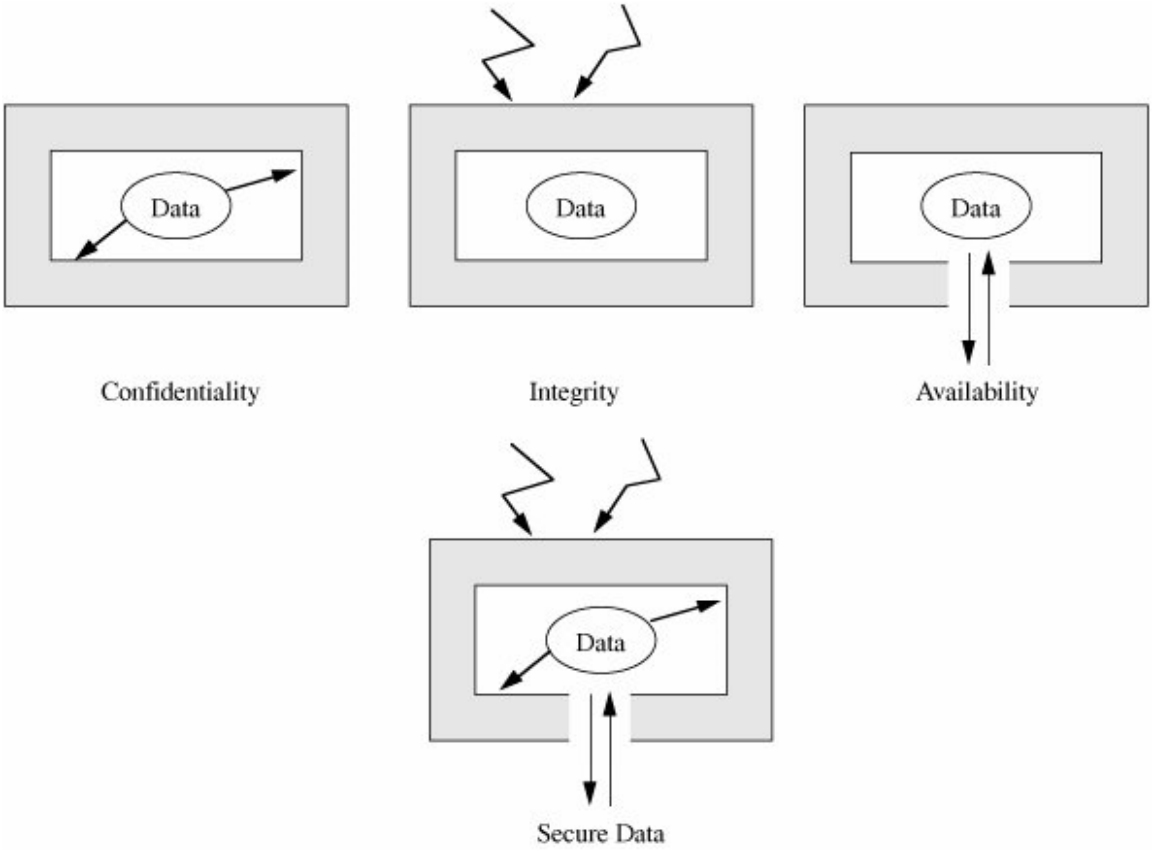


Figure 1.5. Security of Data.

Data confidentiality

Data confidentiality is about protecting data against unintentional, unlawful, or unauthorized access, disclosure, or theft.

Confidentiality has to do with the privacy of information, including authorizations to view, share, and use it. Information with low confidentiality concerns may be considered "public" or otherwise not threatening if exposed beyond its intended audience. Information with high confidentiality concerns is considered secret and must be kept confidential to prevent identity theft, compromise of accounts and systems, legal or reputational damage, and other severe consequences.

Examples of data with high confidentiality concerns include:

- Social Security numbers, which must remain confidential to prevent identity theft.
- Passwords, which must remain confidential to protect systems and accounts.

Consider the following when managing data confidentiality:

- To whom data can be disclosed
- Whether laws, regulations, or contracts require data to remain confidential
- Whether data may only be used or released under certain conditions
- Whether data is sensitive by nature and would have a negative impact if disclosed
- Whether data would be valuable to those who aren't permitted to have it (e.g., hackers)

Guidelines for data confidentiality

When managing data confidentiality, follow these guidelines:

Encrypt sensitive files.

Encryption is a process that renders data unreadable to anyone except those who have the appropriate password or key. By encrypting sensitive files (by using file passwords, for example), you can protect them from being read or used by those who are not entitled to do either.

Manage data access.

DOL

Controlling confidentiality is, in large part, about controlling who has access to data. Ensuring that access is only authorized and granted to those who have a "need to know" goes a long way in limiting unnecessary exposure. Users should also authenticate their access with strong passwords and, where practical, two-factor authentication. Periodically review access lists and promptly revoke access when it is no longer necessary.

Physically secure devices and paper documents.

Controlling access to data includes controlling access of all kinds, both digital and physical. Protect devices and paper documents from misuse or theft by storing them in locked areas. Never leave devices or sensitive documents unattended in public locations.

Securely dispose of data, devices, and paper records.

When data is no longer necessary for University-related purposes, it must be disposed of appropriately.

Sensitive data, such as Social Security numbers, must be securely erased to ensure that it cannot be recovered and misused.

Devices that were used for University-related purposes or that were otherwise used to store sensitive information should be destroyed or securely erased to ensure that their previous contents cannot be recovered and misused.

Paper documents containing sensitive information should be shredded rather than dumped into trash or recycling bins.

Manage data acquisition.

When collecting sensitive data, be conscious of how much data is actually needed and carefully consider privacy and confidentiality in the acquisition process. Avoid acquiring sensitive data unless absolutely necessary; one of the best ways to reduce confidentiality risk is to reduce the amount of sensitive data being collected in the first place.

Manage data utilization.

Confidentiality risk can be further reduced by using sensitive data only as approved and as necessary. Misusing sensitive data violates the privacy and confidentiality of that data and of the individuals or groups the data represents.

Manage devices.

Computer management is a broad topic that includes many essential security practices. By protecting devices, you can also protect the data they contain. Follow basic cybersecurity hygiene by using anti-virus software, routinely patching software, whitelisting applications, using device passcodes, suspending inactive sessions, enabling firewalls, and using whole-disk encryption.

Data integrity

Data integrity is the assurance that digital information is uncorrupted and can only be accessed or modified by those authorized to do so. Integrity involves maintaining the consistency, accuracy and trustworthiness of data over its entire lifecycle.

To maintain integrity, data must not be changed in transit and steps must be taken to ensure that data cannot be altered by an unauthorized person or program. Such measures include implementing user access controls and version control to prevent erroneous changes or accidental deletion by authorized users. Other measures include the use of checksums and cryptographic checksums to verify integrity. Network administration measures to ensure data integrity include documenting system administration procedures, parameters, and maintenance activities, and creating disaster recovery plans for occurrences such as power outages, server failure or security attacks. Should data become corrupted, backups or redundancies must be available to restore the affected data to its correct state.

Network

Network security is any activity designed to protect the usability and integrity of your network and data. It includes both hardware and software technologies. Effective network security manages access to the network. It targets a variety of threats and stops them from entering or spreading on your network.

Access control

Access control is a security technique that regulates who or what can view or use resources in a computing environment. It is a fundamental concept in security that minimizes risk to the business or organization.

There are two types of access control: physical and logical. Physical access control limits access to campuses, buildings, rooms, and physical IT assets. Logical access control limits connections to computer networks, system files, and data.

To secure a facility, organizations use electronic access control systems that rely on user credentials, access card readers, auditing and reports to track employee access to restricted business locations and proprietary areas, such as data centers. Some of these systems incorporate access control panels to restrict entry to rooms and buildings as well as alarms and lockdown capabilities to prevent unauthorized access or operations.

Access control systems perform identification authentication and authorization of users and entities by evaluating required login credentials that can include passwords, personal identification numbers (PINs), biometric scans, security tokens or other authentication factors. Multifactor authentication, which requires two or more authentication factors, is often an important part of the layered defense to protect access control systems.

1.4. Computer Criminals

Computer criminals are people who are caught and convicted of computer crimes such as breaking into computers or computer networks. Computer crime can be broadly defined as criminal activity involving information technology infrastructure, including illegal access (unauthorized access), illegal interception (by technical means of non-public transmissions of computer data to, from or within a computer system), data interference (unauthorized damaging, deletion, deterioration, alteration or suppression of computer data), systems interference (interfering with the functioning of a computer system by inputting, transmitting, damaging, deleting, deteriorating, altering or suppressing computer data), misuse of devices, forgery (or identity theft) and electronic fraud.

Amateur

People who are accidental accessed to unauthorized resources and execution of unauthorized operations. They don't harm to the regular users.The amateurs are the "normal" people who exploit the apparent security flaws to gain an advantage. So is a worker in an office, who can simply read the mail from other users

Crackers or Malicious Hackers

The crackers have more knowledge than the amateurs. They see often a challenge to break into a system and most of them have the attitude that there is no real victim. They use the World Wide Web, email, forums, etc. to get the newest information about insecure systems. "There is no common profile or motivation to attackers [crackers]."

Career Criminals

The career criminals are real experts who started commonly as computer professionals. They break into the systems to get some important data and sell them. This is often their main income

Terrorists

The link between computers and terrorism is quite evident. We see terrorists using computers in three ways:

Targets of attack: denial-of-service attacks and web site defacements are popular for any political organization because they attract attention to the cause and bring undesired negative attention to the target of the attack.

Propaganda vehicles: web sites, web logs, and e-mail lists are effective, fast, and inexpensive ways to get a message to many people.

Methods of attack: to launch offensive attacks requires use of computers.

1.5. Methods of Defense

IDOL

Computer crime is certain to continue. The goal of computer security is to institute controls that preserve secrecy, integrity, and availability. Sometimes these controls are able to prevent attacks; other less powerful methods can only detect a breach as or after it occurs.

How can we defend against a threat?

Prevent it: block the attack

Deter-it: make the attack harder or more expensive

Deflect it: make yourself less attractive to an attacker

Detect it: notice that attack is occurring (or has occurred)

Recover from it: mitigate the effects of the attack

Often, we'll want to do many things to defend against the same threat

“Defense in depth”

Example of defense

Threat: your car may get stolen

How to defend?

Prevent: is it possible to absolutely prevent?

Deter: Store your car in a secure parking facility

Deflect: Use “The Club”

Detect: Car alarms, LoJack

Recover: Insurance

Defense of computer systems

Remember we may want to protect any of our assets

IDOL

Hardware, software, data

Many ways to do this; for example:

Cryptography

Protecting data by making it unreadable to an attacker

Authenticating users with digital signatures

Authenticating transactions with cryptographic protocols

Ensuring the integrity of stored data

Aid customers' privacy by having their personal information automatically become unreadable after a certain length of time

Encryption

The most powerful tool in providing computer security is coding. By transforming data so that it is unintelligible to the outside observer, the value of an interception and the possibility of a modification or a fabrication are almost nullified.

Encryption provides secrecy for data. Additionally, encryption can be used to achieve integrity, since data that cannot be read generally also cannot be changed. Furthermore, encryption is important in protocols, which are agreed-upon sequences of actions to accomplish some task. Some protocols ensure the availability of resources. Thus, encryption is at the heart of methods for ensuring all three goals of computer security.

Encryption is an important tool in computer security, but one should not overrate its importance. Users must understand that encryption does not solve all computer security problems. Furthermore, if encryption is not used properly, it can have no effect on security or can, in fact, degrade the performance of the entire system. Thus, it is important to know the situations in which encryption is useful and to use it effectively

Software controls

IDOL

Programs themselves are the second link in computer security. Programs must be secure enough to exclude outside attack. They must also be developed and maintained so that one can be confident of the dependability of the programs.

Program controls include the following kinds of things:

. Development controls, which are standards under which a program is designed,coded,tested, and maintained

. Operating system controls, which are limitations enforced by the operating system to protect each user from all other users

. Internal program controls that enforce security restrictions, such as access limitations in a database management program

Software controls may use tools such as hardware components,encryption, or information gathering. Software controls generally affect users directly, and so they are often the first aspects of computer security that come to mind. Because they influence the way users interact with a computing system, software controls must be carefully designed. Ease of use and potency are often competing goals in the design of software controls

Hardware Controls

Numerous hardware devices have been invented to assist in computer security. These devices range from hardware implementations of encryption to locks limiting access to theft protection to devices to verify users' identities.

Policies

Some controls on computing systems are achieved through added hardware or software features, as described above. Other controls are matters of policy. In fact, some of the simplest controls, such as frequent changes of passwords, can be achieved at essentially no cost but with tremendous effect.

Legal and ethical controls are an important part of computer security. The law is slow to evolve, and technology involving computers has emerged suddenly. Although legal protection is necessary and desirable, it is not as dependable in this area as it would be in more well-understood and long-standing crimes.

The area of computer ethics is likewise unclear, not that computer people are unethical, but rather that society in general and the computing community, in particular, have not adopted formal standards of ethical behavior. Some organizations are attempting to devise codes of ethics for computer professionals. Although these are important, before codes of ethics become widely accepted and therefore effective, the computing community and the general public need to understand what kinds of behavior are inappropriate and why.

Physical Controls

Some of the easiest, most effective, and least expensive controls are physical controls. Physical controls include locks on doors, guards at entry points, backup copies of important software and data, and physical site planning that reduces the risk of natural disasters. Often the simple physical controls are overlooked while more sophisticated approaches are sought.

Effectiveness of Controls

Merely having controls do no good unless they are used properly. The next section contains a survey of some factors that affect the effectiveness of controls.

. **Awareness of Problem**

People using controls must be convinced of the need for security; people will willingly cooperate with security requirements only if they understand why security is appropriate in each specific situation. Many users, however, are unaware of the need for security, especially in situations in which a group has recently undertaken a computing task that was previously performed by a central computing department.

. **Likelihood of Use**

Of course, no control is effective unless it is used. The lock on a computer room door does no good if people block the door open. During World War II code clerks

IDOL

used outdated codes because they had already learned them and could encode messages rapidly. Unfortunately, the opposite side had already broken some of those codes and could decode those messages easily.

Principle of Effectiveness. Controls must be used to be effective. They must be efficient, easy to use, and appropriate.

This principle implies that computer security controls must be efficient enough, in terms of time, memory space, human activity, or other resources used, so that using the control does not seriously affect the task being protected. Controls should be selective so that they do not exclude legitimate accesses.

1.6 Review Question

1.Distinguish among vulnerability, hazard, and control.

2.Theft usually effects in some sort of harm. For instance, if a person steals your vehicle, you may undergo financial loss, trouble (by sacrificing your method of transport), and mental upset (due to the invasion of one's personal house and area). Record three forms of harm an organization might working experience from the fraud of computer apparatus.

3. List at the very least three forms of harm an organization could go through from electric espionage or unauthorized visiting of confidential corporation materials.

4.List at the very least three forms of damage an organization could suffer once the integrity of an application or company files is compromised.

5. Describe two types of vulnerabilities in cars for which automobile manufacturers have got instituted controls. Say to why you imagine these controls work, somewhat helpful, or ineffective.

6.One handle against accidental computer software deletion would be to save all older versions of an application. Needless to say, this control is usually prohibitively expensive with regards to the cost of safe-keeping. Suggest a less expensive control against unintentional software deletion. Can be your control efficient against all feasible causes of program deletion? Or even, what threats doesn't it cover?

IDOL

7. On an average multiuser computing technique (like a shared Unix program at a college or university or a business), who is able to modify the program code (software program) of this operatingsystem? Of a significant application program like a payroll program or perhaps a statistical analysis offer? Of an application developed and operated by a solo user? Who ought to be permitted to change each one of these examples of program code?

8. Suppose an application to print out paychecks secretly leaking a summary of names of staff earning greater than a certain amount every month. What controls could possibly be instituted to control the vulnerability of the leakage?

9.Some terms have already been created intentionally without explanation in this section. You ought to be in a position to deduce their meanings. What's an electric spy? What's an information dealer?

10 Preserving confidentiality, integrity, and option of data is really a restatement on the worry over interruption, interception, changes, and fabrication. Just how do the initial three concepts relate to the final four? That's, is the four equal to a number of from the three? Is among the three encompassed by a number of in the four?

11.Do you consider attempting to break into (that's, access or usage of) a processing program without authorization ought to be outlawed? Why or you will want to?

12. Describe a good example (apart from the one brought up in this section) of files whose confidentiality includes short timeliness, claim, each day or fewer. Describe a good example of information whose confidentiality includes timeliness greater than a year.

13. Can you currently apply any computer security and safety control measures? If that's the case, what? Against what disorders are you attempting to protect?

14. Describe a good example in which utter denial of assistance to an end user (that's, the user receives no response from your computer) is really a serious problem compared to that consumer. Describe another illustration where ten percent denial of support to a consumer (that's, the user's computation advances, but at a level ten percent slower than usual) is really a serious problem compared to that user. Could accessibility by unauthorized visitors to a computing technique create a ten percent denial of provider to the authentic users? How?

IDOL

15. Once you say that application is of top quality, what can you mean? So how exactly does security match your meaning of quality? For instance, can a credit card application be insecure but still be "good"?

16.Developers usually think of program quality with regards to faults and problems. Faults are troubles, such as for example loops that by no means terminate or misplaced commas in claims, that developers can easily see by considering the code. Problems are problems, like a system accident or the invocation of the incorrect function, which are visible to an individual. Hence, faults can are present in applications but never grow to be failures, as the ailments under which a problem becomes failing are never got to. How do computer software vulnerabilities match this program of faults and problems? Is every problem a vulnerability? Can be every vulnerability a problem?

17. Look at a program to show on your site your city's existing time and temps. Who should attack your plan? What forms of harm might they like to cause? What types of vulnerabilities might they exploit to lead to harm?

18.Look at a program which allows consumers to purchase products from the net. Who should attack this program? What forms of harm might they like to cause? What types of vulnerabilities might they exploit to result in harm?

19. Look at a program to simply accept and tabulate votes within an election. Who should attack this program? What forms of harm might they like to cause? What types of vulnerabilities might they exploit to lead to harm?

20.Look at a program which allows a surgeon in a single city to aid in functioning on an individual in another metropolis via a Web connection. Who should attack this program? What forms of harm might they like to cause? What types of vulnerabilities might they exploit to result in harm?

21.Reviews of computer safety measures failures appear usually in regular information. Cite a noted malfunction that exemplifies one (or even more) from the principles stated in this section: least complicated penetration, adequate defense, effectiveness, weakest link.

IDOL

1.7 References

1. Security in Computing, Fourth Edition By Charles P. Pfleeger - Pfleeger Consulting Group, Shari Lawrence Pfleeger - RAND Corporation Publisher: Prentice Hall

2. Cryptography and Network Security - Principles and Practice fifth edition Stallings William Publisher: Pearson

3. Cryptography And Network Security 3rd Edition behrouz a forouzan and debdeepmukhopadhyay 3/EPublisher: McGraw Hill Education

4. Cryptography and Network Security, 3e AtulKahatePublisher: McGraw Hill

IDOL

Chapter 2. Elementary Cryptography

2.0 Introduction of Cryptography

2.1. Terminology and Background

2.2. Substitution Ciphers

1.2.1 Vulnerabilities, Threats, Attacks, and Controls

2.3. Transpositions (Permutations)

2.4. Making "Good" Encryption Algorithms

2.5 The Data Encryption Standard

2.6. The AES Encryption Algorithm

2.7. Public Key Encryption

2.8. The Uses of Encryption

2.9 Review Question

2.8 References

IDOL

1.0 Introduction of Cryptography

Cryptography is process of converting ordinary plain text into unintelligible text and vice-versa. In cryptography, there are many strongest tools for controlling against many different kinds of security threats. The tools can convert data that cannot be read, modified, or fabricated easily.

Cryptography is based on higher mathematics, it requires in field of group and field theory, computational complexity, real analysis, probability and statistics. Perhaps it is not necessary to understand the underlying mathematics to be able to use cryptography.

In this chapter, we examining what encryption does and how it works. Introduction about the basic principles of encryption with two simple encryption methods: substitution and transposition. Next, we see the two encryption method can use to expand and improved to create stronger, more sophisticated protection. We will how an encryption can fail, due toweakness or flawed encryption process. We will analyze different techniques that can be used to break through the protective scheme and disclosed the original text. For the encryption of data we use three very popular algorithms i.e. DES, AES, and RSA in this day. We see much moredetails of these algorithms that can be used as building blocks with protocols and structures to perform other computing tasks, such as signing documents, detecting modification, and exchanging sensitive data.

2.1. Terminology and Background

Consider the following steps for sending messages from a **sender**, S, to a **recipient**, R. If Ssend the message to T and entrust to T, then who will delivers the message to R, T couldbecome the **transmission medium**. If an outsider,O, wants to access the message (in termsread, change, or even destroy the content of the message), Then we callO is an **interceptor** or **intruder**. If Swants to transmit the message at any time through theT, then message becomesvulnerable to exploitation, and O might try to access the content of the message in any of the following ways:

- The outsider will block the message, before reaching the message to R, hence it affect the availability of the message.
- The Intruder will intercept the message, by reading content of message or listening to the message, therefore its affects the confidentiality of the message.
- The intruder tries to modify the content of message, by seizing the message and or change the message in some other way, therebyaffecting the integrity of message.
- The intruder try to fabricate an authentic-looking message, arranging for it to be delivered as if it came from S, thereby also affecting the integrity of the message.

In previous chapter 1, we have seen there are four security failures cause by a message's vulnerabilities. So the encryption is a technique that can address all these problems. Encryption, probably the most fundamental building block of secure computing, is a means of maintaining secure data in an insecure environment. (It is not the only building block, however.) In this book, we study encryption as a security technique, and we see how it is used in protecting programs, databases, networks, and electronic communications.

Terminology

Encryption is the process in cryptography that converts the plain text in to unreadable text (encrypted format); **decryption** is the reverse process in cryptography, that covert the encrypted text back into original format. The terms **encode** and **decode** or **encipher** and **decipher** are used instead of encrypt and decrypt i.e. we say that the word encode, encrypt, or encipher the originalcontent of message to hide. The words decode, decrypt, or decipher it to reveal the content original message. A system used for the process of encryption and decryption is called a **cryptosystem**

The original content of a message is known as **plaintext**, whereas the encrypted text format is called **ciphertext**. This relationship is shown in Figure 2.1. For more convenient way for describing the relation, we denote a plaintext message as letter P as a it contains a sequence of individual characters P = <p_1, p_2, …, p_n>. Similarly, we denote a ciphertext is written as letterC = <c_1, c_2, …,c_m>. consider an example , the plaintext message written as "I want a code" can be denoted as the message of string <I, ,w,a,n,t , ,a , , c,o,de,>. It can be transformed into

ciphertext<c_1, c_2,..,c_{13}>, and the encryption algorithm tells us how the transformation is done.

Figure 2.1. Encryption.

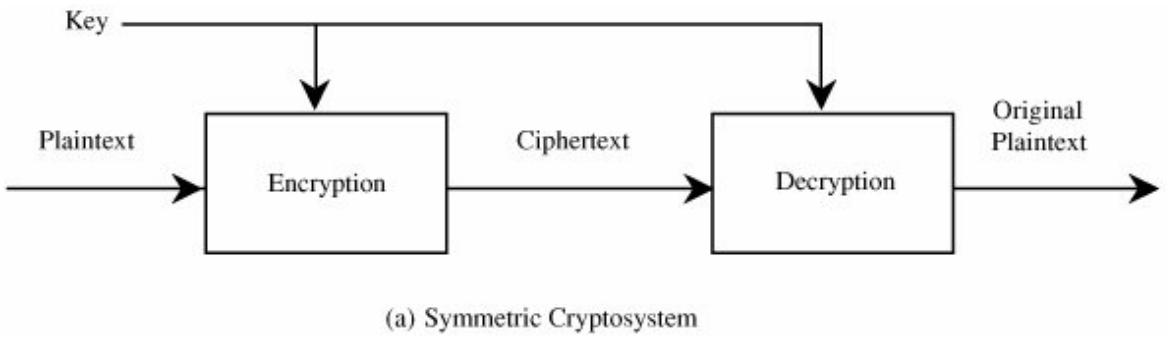
We describe the formal notation for transformations between plaintext and ciphertext. For example, we write C = E(P) and P = D(C), where C denote as ciphertext, E denote as the encryption rule, P denote as the plaintext, and D is denote for a decryption rule. We want a cryptosystem to be P = D (E (P)). In other words, we want to be able to convert the plaintext message tocipertext message in order to protect from an intruder, but we also retrieve the get the original message from the cipertext from which intended receiver able to read the message.

Encryption Algorithms

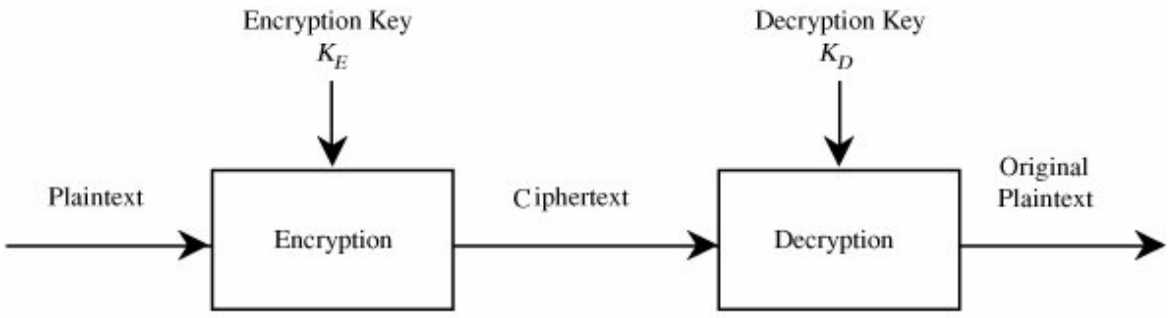
The cryptosystem defines sets of rule for how to encrypt the plaintext and how to decrypt the ciphertextmessage. The process of define a rules for encryption and decryption are called as **algorithms**, the main component in the algorithm is key, which denoted as K, from which we get resulted ciphertext from the plaintext, the algorithm, and the key value. As we note dependence C = E (K, P). Whereas E content is a set of encryption algorithms, and here we use the key K selects for one specific algorithm from the set.

The algorithm which use the same key for encryption and decryption keys, is called a **symmetric** encryption , the notation for the both the process with the key is P = D(K, E(K,P)).. The algorithms uses the different key for encryption and different decryption is called , **asymmetric** encryption ,Here a decryption key K_D, inverts the encryption of key K_E so that P = D(K_D, E(K_E,P)).

The difference between symmetric and asymmetric encryption is shown in Figure 2.2.



(a) Symmetric Cryptosystem



(b) Asymmetric Cryptosystem

An encryption scheme which does not require a key is called a **keyless cipher**.

The word **cryptography** means to written the hidden content by the practice of using encryption to conceal text. A **cryptanalyst is a person which** studies the process encryption and encrypted messages, in order to find the plain content from the hidden content

Both a cryptographer and a cryptanalyst may try an attempt to translate hidden or coded text back to its original form. Here the two terminology is different,i.e. a cryptographer will work on behalf of a legitimate user i.e. sender or receiver, whereas a cryptanalyst will work on behalf of an unauthorized interceptor.

Cryptology is the research in which we use to study of encryption and decryption process; which includes both cryptography and cryptanalysis.

Cryptanalysis

The main aim of cryptanalyst'sis to**break** an encryption process. That is, the cryptanalyst try to find and guess by using the original meaning of a ciphertext message. The intruder hope to determine which decryption algorithm will match the encrypting algorithm so the encoded messages will be broken to get plaintext. For instance, suppose two countries fighting with each other and second country will send the encoded message to own army headquarter ,the first country has intercepted encrypted messages of the second country. Cryptanalysts of the first country will decipher a particular message of the second country message so that the first country will anticipate the movements and resources of the second. If the first country have better decryption algorithm; then the first country can easily break the encryption of all messages sent by the second country.

Thus, a cryptanalyst can attempt to do any or all of six different things:

- It can break a single plain text message
- It can recognize patterns in encrypted messages, so cryptanalyst will able to break subsequent messages by a straightforward decryption algorithm
- Cryptanalyst will try to guess some meaningful information without having broken the encryption text by noticing an unusual frequency of communication or determine whether the communication was short or long for breaking the encrypted text.
- It will try to find the actual key applied in the algorithm, to break encrypted messages easily to get the readable message.
- It will try finding weaknesses in the implementation of algorithm that use of encrypting the plaintext.
- It will try to find general weaknesses in an encryption algorithm, without failing to intercept any plaintext message.

IDOL

Breakable Encryption

An encryption algorithm is known as breakable when a cryptanalyst can determine the algorithm and given an ample amount of time and space for breaking the encrypting algorithms.. However, theoretically an algorithm can be break may not impractical to try break an algorithm. Consider an example let a 25-character message can be expressed in just uppercase letters, so given cipher scheme may have 26^{25} (approximately 10^{35}) possible way for decryption , so the task is to select the right one out of the 26^{25}. If your computer could perform computation of message on the order of 10^{10} operations per second, finding this decipherment would require on the order of 10^{16} seconds, so it would roughly take a 10^{11} years for breaking the encrypted text. In this case, we theoretically we could create the solution of decipherment by determining the deciphering algorithm by examining all possibilities way. But still we have to ignore as infeasible with current technology.

Representing Characters

In computer system, we want to study different ways any character representation, whether it is written as ASCII characters, binary data, object code, or a control stream. In order who the character are encrypted consider an example, we start with an encryption of messages written in the standard 26-letter English alphabet, A through Z.

LETTER	A	B	C	D	E	F	G	H	I	J	K	L		
CODE	0	1	2	3	4	5	6	7	8	9	10	11		
LETTER	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
CODE	12	13	14	15	16	17	18	19	20	21	22	23	24	25

In above, the letter A is represented by a zero, B by a one, and so on. This representation allows performing an arithmetic operation on the "letters" of a message. Then we can also perform addition and subtraction on letters by adding and subtracting the corresponding code numbers. With the representation of Expression on letter such as A + 4 = E or L- 1 = K. Arithmetic operation can be performed in a alphabetic order where the alphabetic table order were circular. In other words, addition wraps around from one end of the table to the other so that Y + 4= C. Thus, every result of an arithmetic operation is between 0 and 25.

There are many types of encryption. We look two different forms of encryption: **substitutions**, where one letter is exchanged for another, and **transpositions**, where the order of the letters is rearranged. The main aims of two forms for studying with the different concept of encryption and decryption, we will learn some of the terminology and methods of cryptanalysis, and to study some of the weaknesses to which encryption is prone.

2.2. Substitution Ciphers

Substitution ciphers is a technique in which characters from the plaintext are simply substituted (replaced in a specific manner) with another set of characters, we get aresults in the form of ciphertext. This technique as called as a monoalphabetic cipher or **simple substitution.**

The Caesar Cipher

The Caesar Cipher, also called as a shift cipher, it is one of the oldest and simplest technique for encryption of a message. It is a type of substitution cipher technique, where each letter in the original message is replaced with a letter corresponding to a certain number of letters shifted up or down in the alphabet.

For an example, the Caesar Cipher encryption of a full message, using a left shift of 3.

Plaintext: THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG

Ciphertext: QEB NRFZH YOLTK CLU GRJMP LSBO QEB IXWV ALD

Advantages and Disadvantages of the Caesar Cipher

Advantages include:

- It is one of the simplest methods to use in cryptography and can provide less security to the information
- It only use a short key in the entire process
- One of the best methods to use if the system cannot use any complicated coding techniques requires few computing resources
- One of the easiest methods to use in cryptography and can provide minimum security to the information

IDOL

Disadvantages of using a Caesar cipher include:

- It has Simple structure usage
- It can only provide minimum security to the information
- Frequency of the letter pattern provides a big clue in deciphering the entire message

Cryptanalysis of the Caesar Cipher

Being arguably the simplest keyed cipher, the Caesar cipher can be broken in milliseconds using automated tools. Since there are only 25 possible keys (each possible shift of the alphabet), we just try decrypting the ciphertext using each key This form of solution is known as a 'brute force' solution, and is only possible for the very simplest of ciphers.

An Example

Our ciphertext is the following:

YMJHFJXFWHNUMJWNXTSJTKYMJJFWQNJXYPSTBSFSIXNRUQJXYHNUMJWX

To find out what the original was, we try decrypting it with each of the 25 possible keys, calculating the fitness for each trial decryption:

Key	plaintext
1	XLIGEIWEVGMTLIVMWSRISJXLIIEVPM…
2	WKHFDHVDUFLSKHULVRQHRIWKHHDUOL...
3	VJGECGUCTEKRJGTKUQPGQHVJGGCTNK...
4	UIFDBFTBSDJQIFSJTPOFPGUIFFBSMJ...
5	**THECAESARCIPHERISONEOFTHEEARLI...**
6	SGDBZDRZQBHOGDQHRNMDNESGDDZQKH...
7	RFCAYCQYPAGNFCPGQMLCMDRFCCYPJG…

IDOL

8	QEBZXBPXOZFMEBOFPLKBLCQEBBXOIF...
9	PDAYWAOWNYELDANEOKJAKBPDAAWNHE...
10	OCZXVZNVMXDKCZMDNJIZJAOCZZVMGD...
11	NBYWUYMULWCJBYLCMIHYIZNBYYULFC...
12	MAXVTXLTKVBIAXKBLHGXHYMAXXTKEB...
13	LZWUSWKSJUAHZWJAKGFWGXLZWWSJDA...
14	KYVTRVJRITZGYVIZJFEVFWKYVVRICZ...
15	JXUSQUIQHSYFXUHYIEDUEVJXUUQHBY...
16	IWTRPTHPGRXEWTGXHDCTDUIWTTPGAX...
17	HVSQOSGOFQWDVSFWGCBSCTHVSSOFZW...
18	GURPNRFNEPVCUREVFBARBSGURRNEYV…
19	FTQOMQEMDOUBTQDUEAZQARFTQQMDXU…
20	ESPNLPDLCNTASPCTDZYPZQESPPLCWT…
21	DROMKOCKBMSZROBSCYXOYPDROOKBVS…
22	CQNLJNBJALRYQNARBXWNXOCQNNJAUR...
23	BPMKIMAIZKQXPMZQAWVMWNBPMMIZTQ...
24	AOLJHLZHYJPWOLYPZVULVMAOLLHYSP...
25	ZNKIGKYGXIOVNKXOYUTKULZNKKGXRO...

IDOL

Cryptanalysis is the art of breaking codes and ciphers. The Caesar cipher is probably the easiest of all ciphers to break. Since the shift has to be a number between 1 and 25, (0 or 26 would result in an unchanged plaintext) we can

simply try each possibility and see which one results in a piece of readable text. If you happen to know what a piece of the ciphertext is, or you can guess a piece, then this will allow you to immediately find the key.

If this is not possible, a more systematic approach is to calculate the frequency distribution of the letters in the cipher text. This consists of counting how many times each letter appears. Natural English text has a very distinct distribution that can be used help crack codes. This distribution is as follows:

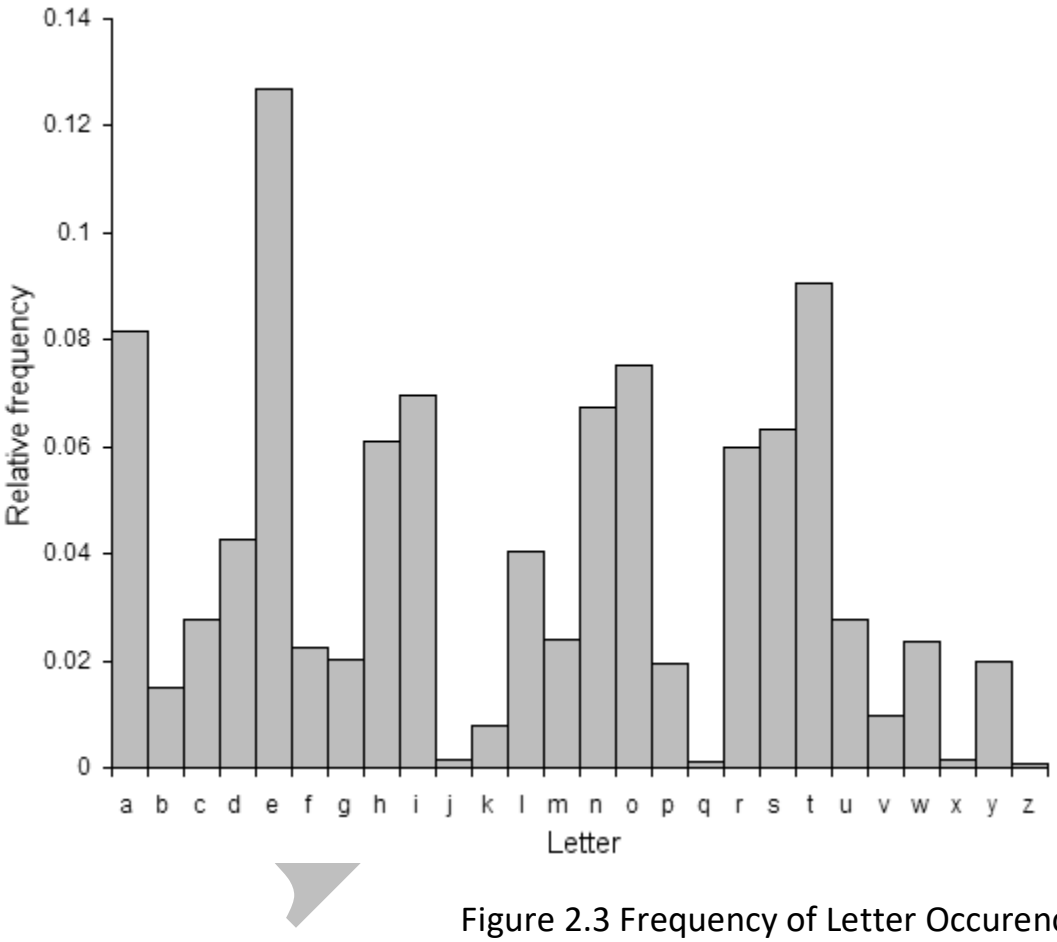


Figure 2.3 Frequency of Letter Occurence

This means that the letter **e** is the most common, and appears almost 13% of the time, whereas **z** appears far less than 1 percent of time. Application of the Caesar cipher does not change these letter frequencies, it merely shifts them along a bit (for a shift of 1, the most frequent ciphertext letter becomes **f**). A cryptanalyst just has to find the shift that causes the ciphertext frequencies to match up closely with the natural English frequencies, then decrypt the text

using that shift. This method can be used to easily break Caesar ciphers by hand.

One -Time Pad

The One-Time Pad, or OTP is an encryption technique in which each character of the plaintext is combined along with a character from a random **key stream.** Originally described in 1882 by banker Frank Miller (USA), it was re-invented in 1917 by Gilbert Vernam and Joseph Mauborgne. When applied correctly, the OTP provides a truely unbreakable cipher. It is named after the sheets of paper (pads) on which the key stream was usually printed. It also exists as *One Time Tape* (OTT).

Theory

The theory behind the OTP is that the encryption key has at least the same length as the actual message (i.e. the plaintext) and consists of truly random numbers. Each letter of the plaintext is 'added' to one element from the OTP using modulo-addition. This results in a cipher text that has no relation with the plaintext when the key is unknown. At the receiving end, the same OTP is used to retrieve the original plaintext.

For this to work, the following rules are mandatory:

- The OTP should consist of truly random characters (noise).
- The OTP (i.e. the key) should have the same length as the plaintext (or longer).
- Only two copies of the OTP should exist.
- The OTP should be used only once.
- Both copies of the OTP are destroyed immediately after use.

The Vernam Cipher

The Vernam cipher is a type of one-time pad devised by Gilbert Vernam for AT&T. The Vernam cipher is immune to most cryptanalytic attacks. The basic encryption

involves an arbitrarily long nonrepeating sequence of numbers that are combined with the plaintext.

Algorithm of Vernam cipher:

For a string of m numbers, a string of m random numbers is generated using akey r which is “large prime number”. Here the term “large” is in a sense that itshould have as many bits as the message to be transmitted has.

Encrypted output E (i)= (x (i) + k (i))%26

x(i) = Number at the ith position in input string

k(i) = Corresponding random number generated

Hence ‘m’ random numbers + ‘m’ meaningful numbers give rise to set ofm numbers which form the encrypted message.

Decrypted output D(i)= (x (i)- k(i))%26

Example of Vernam Cipher

- Here, we combine the key and the message using modular addition.
- The numerical values of corresponding message and key letters are added together, modulo 26.
- If key material begins with "XMCKL" and the message is "HELLO", then the coding would be

OTP Encryption Example

H	E	L	L	O	MESSAGE
7(H)	4(E)	11(L)	11(L)	14(O)	MESSAGE
+23(X)	12(M)	2(C)	10(K)	11(L)	KEY
=30	16	13	21	25	MESSAGEE +KEY
=4(E)	16(Q)	13(N)	21(V)	25(Z)	MESSAGEE +KEY (MOD 26)
E	Q	N	V	Z	➔ CIPHERTEXT

IDOL

If a number is larger than 25, then the remainder aftersubtraction of 26 is taken in modular arithmetic fashion .This simply means that if your computations "go past" Z,you start again at A.The ciphertext to be sent to Bob is thus "EQNVZ". Bobuses the matching key page and the same process, butin reverse, to obtain the plaintext. Here the key is *subtracted* from the ciphertext, againusing modular arithmetic.

OTP Decryption

E	Q	N	V	Z	CIPHERTEXT
4(E)	16(Q)	13(N)	21(V)	25(Z)	CIPHERTEXT
-23(X)	12(M)	2(C)	10(K)	11(L)	KEY
-19	4	11	11	14	CIPHERTEXT -KEY
7(H)	4(E)	11(L)	11(L)	14(O)	CIPHERTEXT -KEY (MOD 26)
H	E	L	L	O	➔ MESSAGE

NB: If a number is negative then 26 is added to make the number positive

OTP Cryptanalysis

• Suppose Eve intercepts Alice's ciphertext: "EQNVZ". If Eve had infinite computing power, she would quickly find that the key "XMCKL" would produce the plaintext "HELLO", but she would also find that the key "TQURI" would produce the plaintext "LATER"

E	Q	N	V	Z	CIPHERTEXT
4(E)	16(Q)	13(N)	21(V)	25(Z)	CIPHERTEXT
-19(T)	16(Q)	20(U)	17(R)	8(I)	POSSIBLE KEY
-15	0	-7	4	17	CIPHERTEXT -KEY

IDOL

11(L)	0(A)	17(T)	4(E)	17(R)	CIPHERTEXT -KEY (MOD 26)

It is possible to "decrypt" out of the ciphertext any message whatsoever with the same number of characters, simply by using a different key, and there is no information in the ciphertext which will allow Eve to choose among the various possible readings of the ciphertext Thus, OTP coined, the “Perfect Cipher”

Book Ciphers

A book cipher uses a large piece of text to encode a secret message. Without the key (the piece of text) it is very difficult to decrypt the secret message.

To implement a book cipher, each word in the secret message would be replaced with a number which represents the same word in the book. For example, if the word "attack" appeared in the book as word number 713, then "attack" would be replaced with this number. The result would be an encoded message that looked something like this.

713 23 245 45 124 1269 586 443 8 234

To decipher the message you simply count the number of words in the book and write down each one.

Vigenere Cipher

In a Caesar Cipher, each letter of the alphabet is shifted along some number of places; for example, in a Caesar cipher of shift 3, A would become D, B would become E and so on. The Vigenere cipher consists of using several Caesar ciphers in sequence with different shift values.

To encipher, a table of alphabets can be used, termed a tabula recta, Vigenère square, or Vigenère table. It consists of the alphabet written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet, corresponding to the 26 possible Caesar ciphers. At different points in the encryption process, the cipher uses a different alphabet from one of the rows. The alphabet used at each point depends on a repeating keyword.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Figure 2.4 Vigenère square or Vigenère Table

For example, suppose that the plaintext to be encrypted is:

ATTACKATDAWN

The person sending the message chooses a keyword and repeats it until it matches the length of the plaintext, for example, the keyword "LEMON":

LEMONLEMONLE

Each letter is encoded by finding the intersection in the grid between the plaintext letter and keyword letter. For example, the first letter of the plaintext, A, is enciphered using the alphabet in row L, which is the first letter of the key. This is done by looking at the letter in row L and column A of the Vigenere square, namely L. Similarly, for the second letter of the plaintext, the second letter of the key is used; the letter at row E and column T is X. The rest of the plaintext is enciphered in a similar fashion:

Plaintext: ATTACKATDAWN

Key: LEMONLEMONLE

Ciphertext: LXFOPVEFRNHR

Decryption is performed by finding the position of the ciphertext letter in a row of the table, and then taking the label of the column in which it appears as the

IDOL

plaintext. For example, in row L, the ciphertext L appears in column A, which taken as the first plaintext letter. The second letter is decrypted by looking up X in row E of the table; it appears in column T, which is taken as the plaintext letter.

Transposition Cipher

A transposition cipher is one which rearranges the order of the letters in the ciphertext (encoded text), according to some predetermined method, without making any substitutions. With transposition, the cryptography aims for diffusion, widely spreading the information from the message or the key across the ciphertext. Transpositions try to break established patterns. Because a transposition is a rearrangement of the symbols of a message, it is also known as a permutation.

Columnar Transpositions

It is another type of cipher where the order of the alphabets in the plaintext is rearranged to create the ciphertext. The actual plaintext alphabets are not replaced.

An example is a 'simple columnar transposition' cipher where the plaintext is written horizontally with a certain alphabet width. Then the ciphertext is read vertically as shown.

For example, the plaintext is "modern statue in white house" and the secret random key chosen is "five". We arrange this text horizontally in table with number of column equal to key value. The resulting text is shown below.

m	o	d	e	r
r	s	t	a	t
u	e	i	n	w
h	i	t	e	h
o	u	s	e	

The ciphertext is obtained by reading column vertically downward from first to last column. The ciphertext is 'mruhooseiudtitseaneertwh.

To decrypt, the receiver prepares similar table. The number of columns is equal to key number. The number of rows is obtained by dividing number of total ciphertext alphabets by key value and rounding of the quotient to next integer value.The receiver then writes the received ciphertext vertically down and from left to right column. To obtain the text, he reads horizontally left to right and from top to bottom row.

Monogram, Bigram and Trigram frequency counts

Introduction to Frequency Analysis

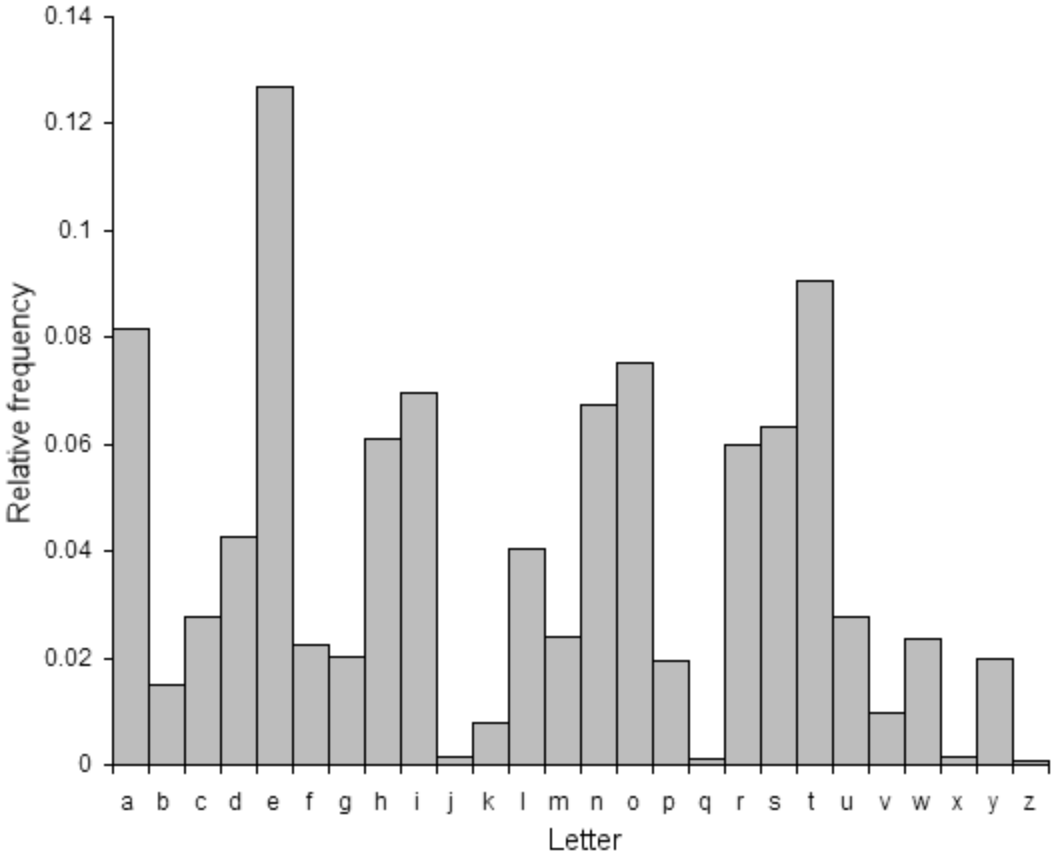
Frequency analysis is the practice of counting the number of occurrences of different ciphertext characters in the hope that the information can be used to break ciphers. Frequency analysis is not only for single characters, **it is also possible to measure the frequency of bigrams (also called digraphs), which is how often pairs of characters occur in text.Trigram frequency counts measure the occurrence of 3 letter combinations.**

When talking about bigram and trigram frequency counts, we will concentrate on text characterization as opposed to solving polygraphic ciphers e.g. playfair. The difference is that text characterizations depends on all possible 2 character combinations, since we wish to know about as many bigrams as we can (this means we allow the bigrams to overlap). When cracking playfair, we do not allow the bigrams to overlap.

Monogram Counts

Monogram frequency counts are most effective on substitution type ciphers such as the caesar cipher, substitution cipher, polybius square etc. It works because natural english text follows a very specific frequency distribution, which is not masked by substitution ciphers. The distribution looks like:

IDOL

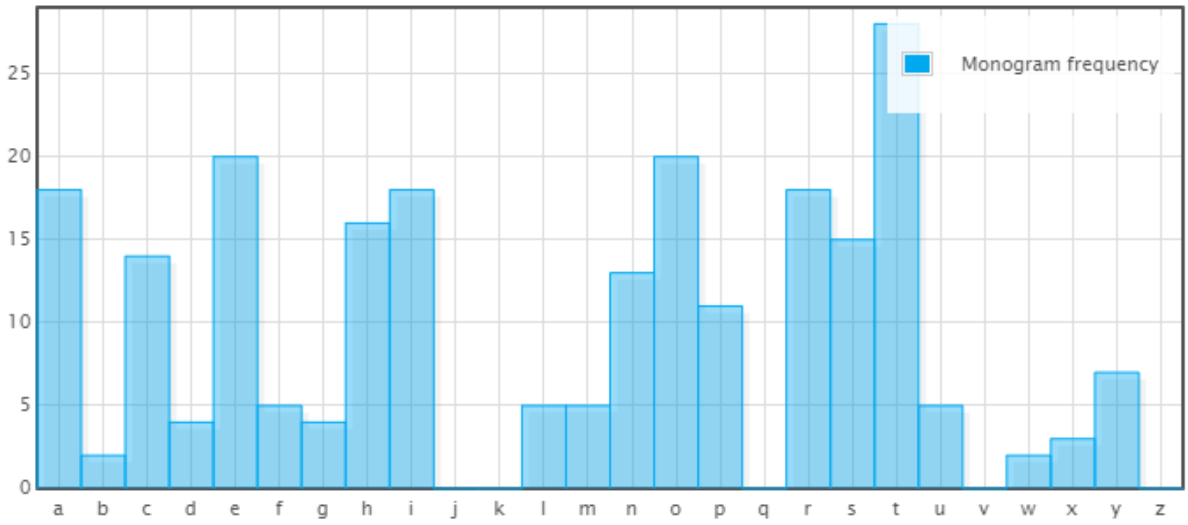


See an Example substitution cipher cryptanalysis on applications of frequency counts for solving substitution ciphers.

Consider a below text

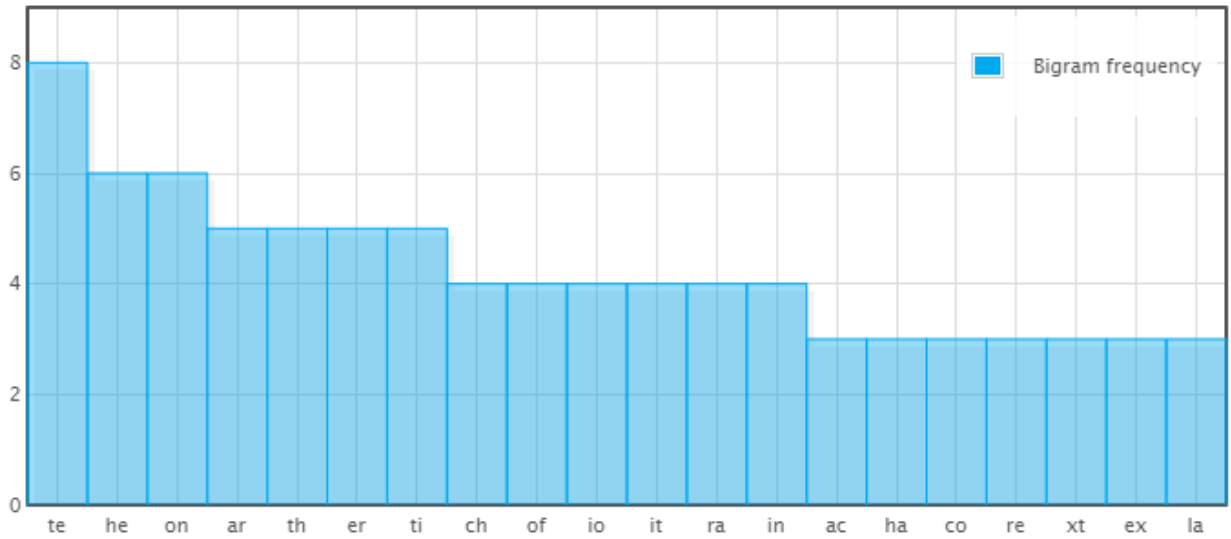
“In cryptography , a transposition cipher is a method of encryption by which the positions held by units of plaintext (which are commonly characters or groups of characters) are shifted according to a regular system, so that the ciphertext constitutes a permutation of the plaintext.”

In Monogram diagram, Occurrence of letters



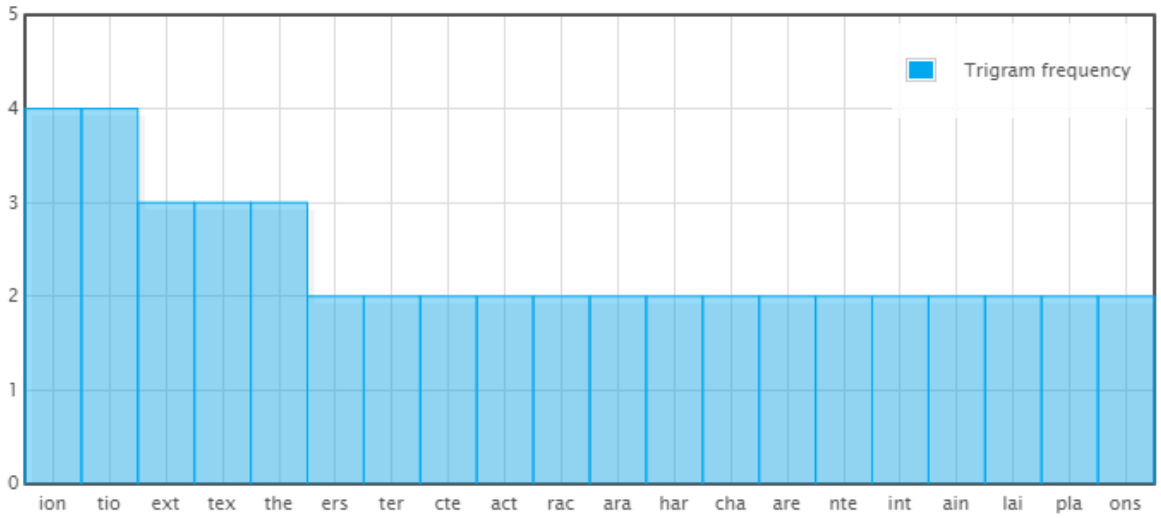
Bigram Counts

Bigram counts maintain the same principle as monogram counts, but instead of counting occurrences of single characters, bigram counts count the frequency of pairs of characters.



Trigram Counts

Just as bigram counts count the frequency of pairs of characters, trigram counts count the frequency of triple characters.



Cryptanalysis by Digram Analysis

Assume the block being compared is seven characters.The first comparison is c1 to c8, c2 to c9, …, c7 to c14. Then, we try a distance of eight characters, and so the window of comparison shifts and c1 is compared to c9, c2 to c10, and continuing.. For each window position, we ask two questions. First, do common digrams appear, and second, do most of the digrams look reasonable

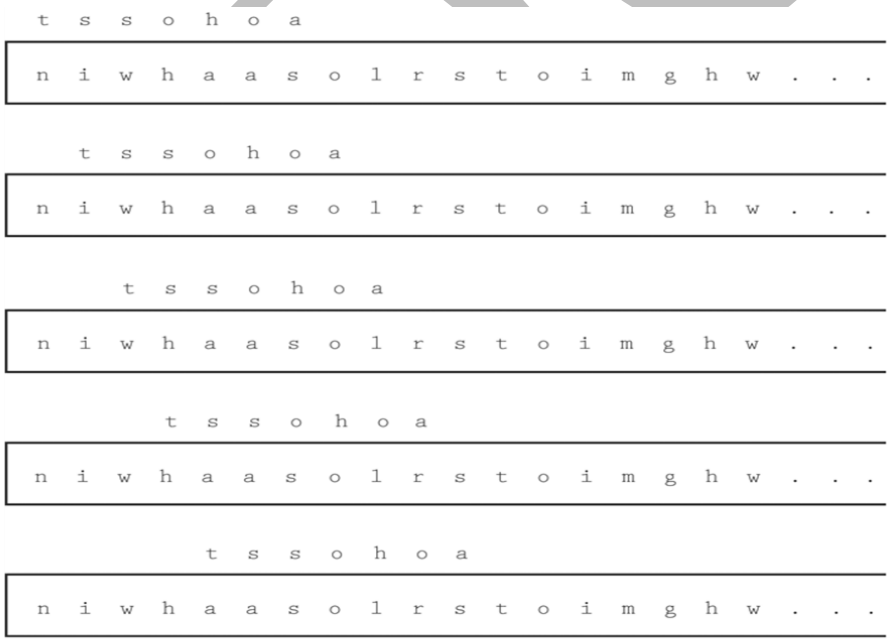


Figure 2-5 Moving Comparisons.

Combinations of Approaches

- Substitution and transposition can be considered as building blocks for encryption.
- A combination of two ciphers is called a **product cipher**.
- Product ciphers are typically performed one after another, as in $E_2(E_1(P,k_1), k_2)$

2.4. Making "Good" Encryption Algorithms

- What Makes a "Secure" Encryption Algorithm?
- What does it mean for a cipher to be "good"?
- The meaning of good depends on the intended use of the cipher
- A cipher to be used by military personnel in the field has different requirements from one to be used in a secure installation with substantial computer support
- In this section, we look more closely at the different characteristics of ciphers

Shannon's Characteristics of "Good" Ciphers

In 1949, Claude Shannon [SHA49] proposed several characteristics that identify a good cipher.

1. The amount of secrecy needed should determine the amount of labor appropriate for the encryption and decryption.

2. The set of keys and the enciphering algorithm should be free from complexity

3. The implementation of the process should be as simple as possible.

4. Errors in ciphering should not propagate and cause corruption of further information in the message.

5. The size of the enciphered text should be no larger than the text of the original message.

Properties of "Trustworthy" Encryption Systems

IDOL

Commercial users have several requirements that must be satisfied when they select an encryption algorithm. Thus, when we say that encryption is "commercial grade," or "trustworthy," we mean that it meets these constraints:

It is based on sound mathematics. Good cryptographic algorithms are not just invented; they are derived from solid principles.

It has been analyzed by competent experts and found to be sound. Even the best cryptographic experts can think of only so many possible attacks, and the developers may become too convinced of the strength of their own algorithm. Thus, a review by critical outside experts is essential.

It has stood the" ***test of time***." As a new algorithm gains popularity, people continue to review both its mathematical foundations and the way it builds on those foundations. Although a long period of successful use and analysis is not a guarantee of a good algorithm, the flaws in many algorithms are discovered relatively soon after their release.

Three algorithms are popular in the commercial world: DES (data encryption standard), RSA (Rivest Shamir Adelman, named after the inventors), and AES (advanced encryption standard). The DES and RSA algorithms (as well as others) meet our criteria for commercial-grade encryption; AES, which is rather new, meets the first two and is starting to achieve widespread adoption.

2.5 The Data Encryption Standard

In the late 1960s, IBM set up a research project in computer cryptography led by Horst Feistel. The project concluded in 1971 with the development of the LUCIFER algorithm. LUCIFER is a Feistel block cipher that operates on blocks of 64 bits, using a key size of 128 bits.

Because of the promising results produced by the LUCIFER project, IBM embarked on an effort, headed by Walter Tuchman and Carl Meyer, to develop a marketable commercial encryption product that ideally could be implemented on a single chip. It involved not only IBM researchers but also outside consultants and technical advice from NSA. The outcome of this effort was a refined version of LUCIFER that was more resistant to cryptanalysis but that had a reduced key size of 56 bits, to fit on a single chip.

In 1973, the National Bureau of Standards (NBS) issued a request for proposals for a national cipher standard. IBM submitted the modified LUCIFER. It was by far the best algorithm proposed and was adopted in 1977 as the Data Encryption Standard.

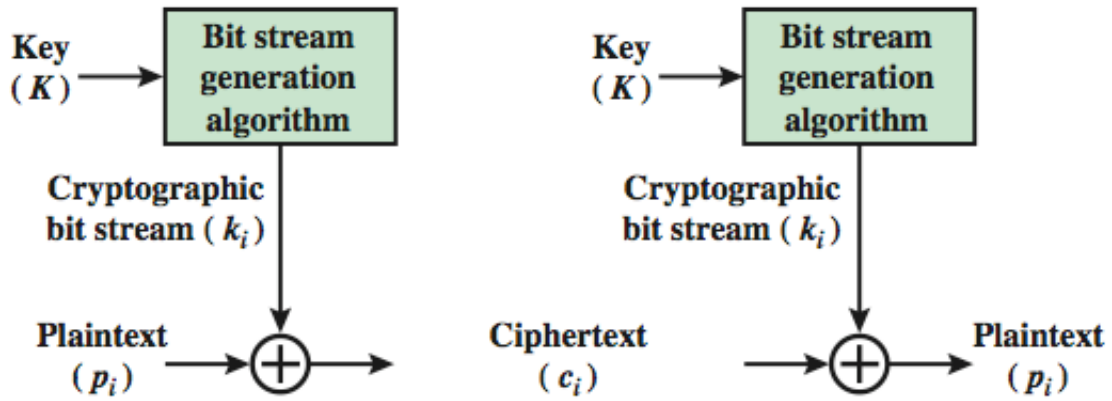
The Data Encryption Standard (DES) [NBS77], a system developed for the U.S. government, was intended for use by the general public.

Block vs Stream Ciphers

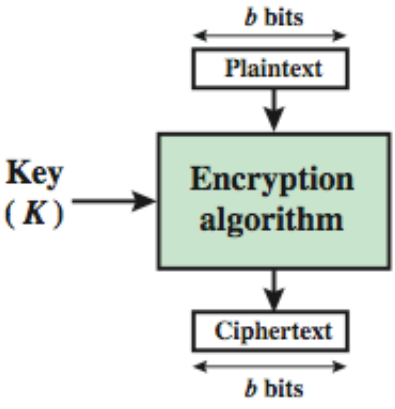
Block ciphers work a on block / word at a time, which is some number of bits. All of these bits have to be available before the block can be processed. Stream ciphers work on a bit or byte of the message at a time; hence process it as a "stream". Block ciphers are currently better analyzed, and seem to have a broader range of applications, hence focus on them.

IDOL

A block cipher is one in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length. Typically, a block size of 64 or 128 bits is used. As with a stream cipher, the two users share a symmetric encryption key A stream cipher is one that encrypts a digital data stream one bit or one byte at a time. In the ideal case, a one-time pad version of the Vernam cipher would be used in which the keystream (k) is as long as the plaintext bit stream (p).

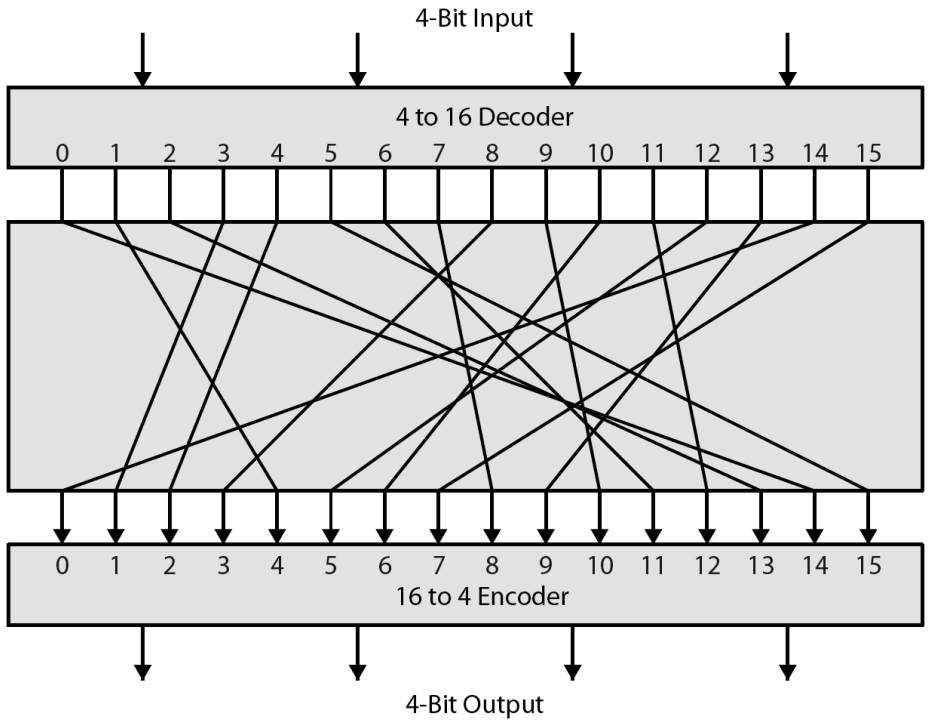


(a) Stream Cipher Using Algorithmic Bit Stream Generator



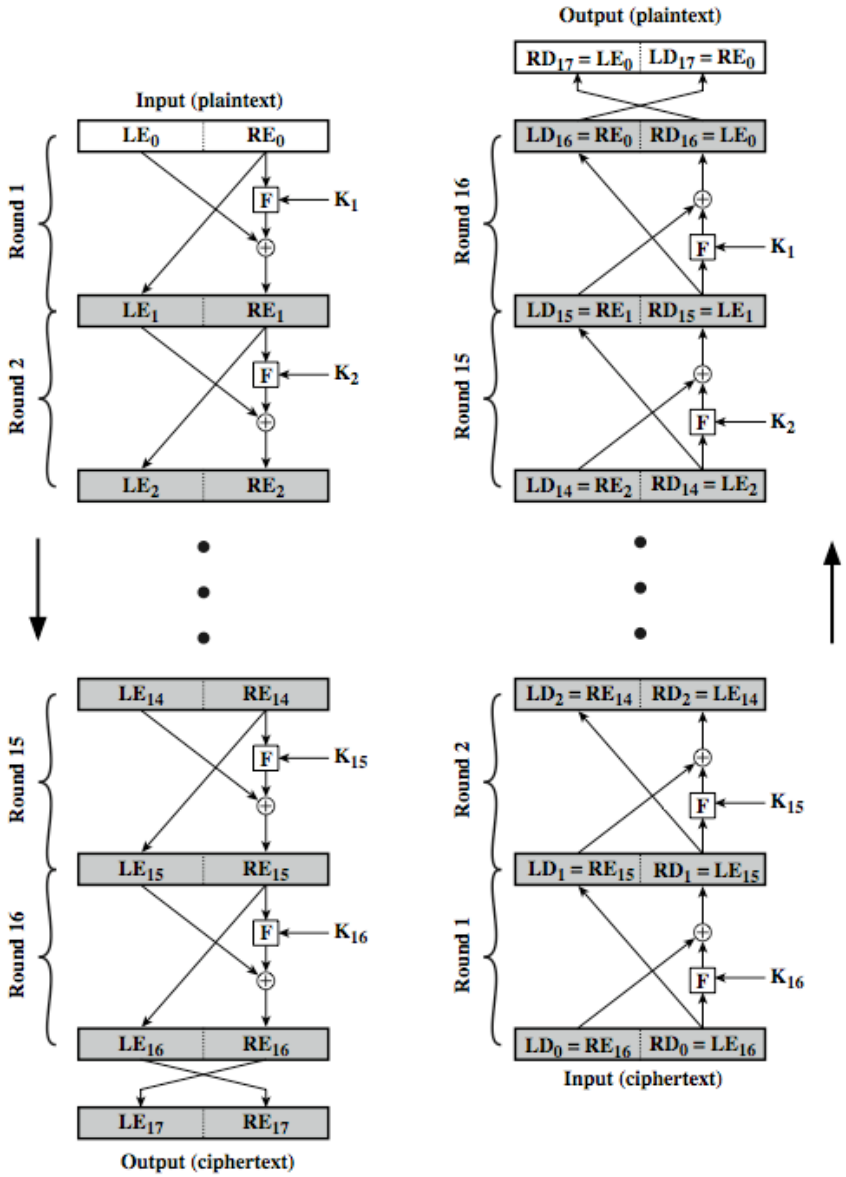
(b) Block Cipher

Most symmetric block encryption algorithms in current use are based on a structure referred to as a **Feistel block cipher**. A block cipher operates on a plaintext block of n bits to produce a ciphertext block of n bits. An arbitrary reversible substitution cipher for a large block size is not practical, however, from an implementation and performance point of view. In general, for an *n*-bit general substitution block cipher, the size of the key is *n x* 2*n*. For a 64-bit block, which is a desirable length to thwart statistical attacks, the key size is 64x 264 = 270 = 1021 bits. In considering these difficulties, Feistel points out that what is needed is an approximation to the ideal block cipher system for large n, built up out of components that are easily realizable.



Feistel refers to an *n*-bit general substitution as an ideal block cipher, because it allows for the maximum number of possible encryption mappings from the plaintext to ciphertext block. A 4-bit input produces one of 16 possible input states, which is mapped by the substitution cipher into a unique one of 16

possible output states, each of which is represented by 4 ciphertext bits. The encryption and decryption mappings can be defined by a tabulation, as shown in Figure. It illustrates a tiny 4-bit substitution to show that each possible input can be arbitrarily mapped to any output - which is why its complexity grows so rapidly.



The exact realization of a Feistel network depends on the choice of the following parameters and design features:

- block size - increasing size improves security, but slows cipher
- key size - increasing size improves security, makes exhaustive key searching harder, but may slow cipher
- number of rounds - increasing number improves security, but slows cipher
- subkey generation algorithm - greater complexity can make analysis harder, but slows cipher
- round function - greater complexity can make analysis harder, but slows cipher
- fast software en/decryption - more recent concern for practical use
- ease of analysis - for easier validation & testing of strength

Claude Shannon and Substitution-Permutation Ciphers

- Claude Shannon introduced idea of substitution-permutation (S-P) networks in 1949 paper
- form basis of modern block ciphers
- S-P nets are based on the two primitive cryptographic operations seen before:
  - *substitution* (S-box)
  - *permutation* (P-box)
- provide *confusion*&*diffusion* of message & key

IDOL

Confusion and Diffusion

- cipher needs to completely obscure statistical properties of original message
- a one-time pad does this
- more practically Shannon suggested combining S & P elements to obtain:
- **diffusion** – dissipates statistical structure of plaintext over bulk of ciphertext
- **confusion** – makes relationship between ciphertext and key as complex as possible

The terms **diffusion** and **confusion** were introduced by Claude Shannon to capture the two basic building blocks for any cryptographic system. Shannon's concern was to thwart cryptanalysis based on statistical analysis. Every block cipher involves a transformation of a block of plaintext into a block of ciphertext, where the transformation depends on the key. The mechanism of ***diffusion*** seeks to make the statistical relationship between the plaintext and ciphertext as complex as possible in order to thwart attempts to deduce the key. ***Confusion***seeks to make the relationship between the statistics of the ciphertext and the value of the encryption key as complex as possible, again to thwart attempts to discover the key.

So successful are diffusion and confusion in capturing the essence of the desired attributes of a block cipher that they have become the cornerstone of modern block cipher design.

IDOL

Overview of the DES Algorithm

The most widely used private key block cipher, is the Data Encryption Standard (DES). It was adopted in 1977 by the National Bureau of Standards as Federal Information Processing Standard 46 (FIPS PUB 46). DES encrypts data in 64-bit blocks using a 56-bit key. The DES enjoys widespread use. It has also been the subject of much controversy its security.

The DES algorithm is a careful and complex combination of two fundamental building blocks of encryption: substitution and transposition. The algorithm derives its strength from repeated application of these two techniques, one on top of the other, for a total of 16 cycles. The sheer complexity of tracing a single bit through 16 iterations of substitutions and transpositions has so far stopped researchers in the public from identifying more than a handful of general properties of the algorithm.

The algorithm begins by encrypting the plaintext as blocks of 64 bits. The key is 64 bits long, but in fact it can be any 56-bit number. (The extra 8 bits are often used as check digits and do not affect encryption in normal implementations.) The user can change the key at will any time there is uncertainty about the security of the old key.

The algorithm, leverages the two techniques Shannon identified to conceal information: confusion and diffusion. That is, the algorithm accomplishes two

things: ensuring that the output bits have no obvious relationship to the input bits and spreading the effect of one plaintext bit to other bits in the ciphertext. Substitution provides the confusion, and transposition provides the diffusion. In general, plaintext is affected by a series of cycles of a substitution then a permutation. The iterative substitutions and permutations are performed

The overall scheme for DES encryption is illustrated in Figure, which takes as input 64-bits of data and of key.

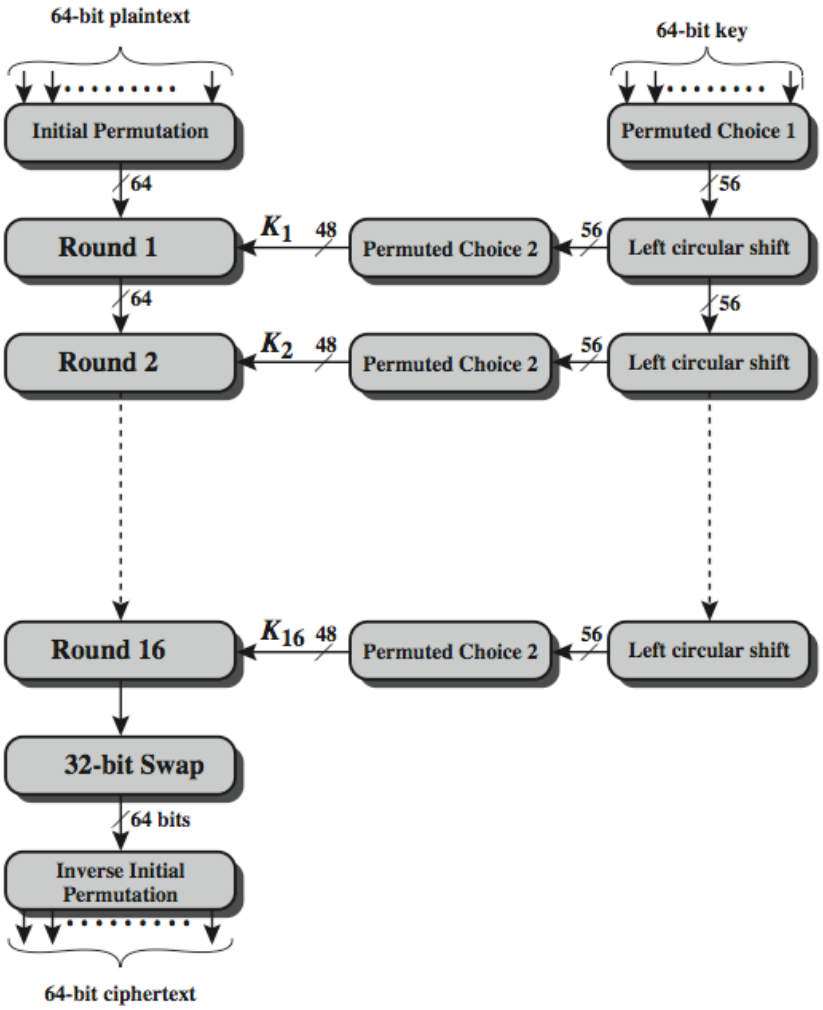
The left side shows the basic process for enciphering a 64-bit data block which consists of:

- an initial permutation (IP) which shuffles the 64-bit input block
- 16 rounds of a complex key dependent round function involving substitutions & permutations
- a final permutation, being the inverse of IP

The right side shows the handling of the 56-bit key and consists of:

- an initial permutation of the key (PC1) which selects 56-bits out of the 64-bits input, in two 28-bit halves
- 16 stages to generate the 48-bit subkeys using a left circular shift and a permutation of the two 28-bit halves

IDOL



Initial Permutation (IP)

- first step of the data computation
- IP reorders the input data bits
- even bits to LH half, odd bits to RH half
- quite regular in structure (easy in h/w)
- no cryptographic value

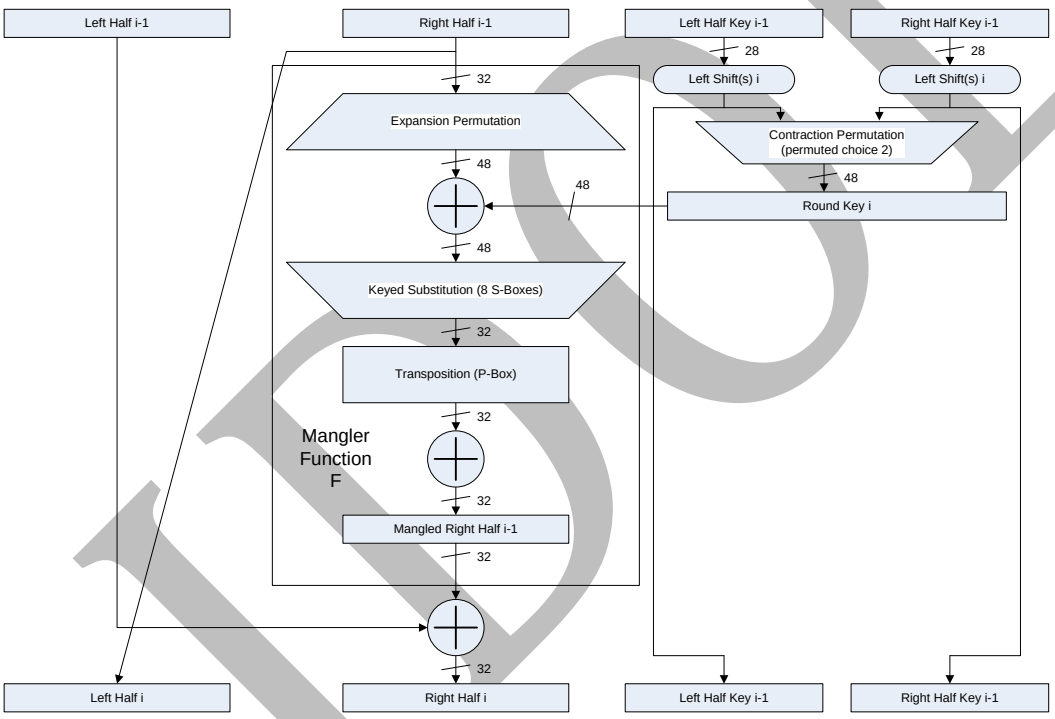
DES Round Structure

- uses two 32-bit L & R halves
- as for any Feistel cipher can describe as:

$Li = Ri–1$

$Ri = Li–1 \oplus F(Ri–1, Ki)$

- F takes 32-bit R half and 48-bit subkey:
  - expands R to 48-bits using perm E
  - adds to subkey using XOR
  - passes through 8 S-boxes to get 32-bit result
  - finally permutes using 32-bit perm P



DES Key Schedule

- forms subkeys used in each round
  - initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves
  - 16 stages consisting of:

- rotating each half separately either 1 or 2 places depending on the key rotation schedule K
- selecting 24-bits from each half & permuting them by PC2 for use in round function F

➢ note practical use issues in h/w vs s/w

DES Decryption

➢ decrypt must unwind steps of data computation

➢ with Feistel design, do encryption steps again using subkeys in reverse order (SK16 … SK1)

- IP undoes final FP step of encryption
- 1st round with SK16 undoes 16th encrypt round
- ….
- 16th round with SK1 undoes 1st encrypt round
- then final FP undoes initial encryption IP
- thus recovering original data value

Avalanche Effect

A desirable property of any encryption algorithm is that a small change in either the plaintext or the key should produce a significant change in the ciphertext. In particular, a change in one bit of the plaintext or one bit of the key should produce a change in many bits of the ciphertext. If the change were small, this might provide a way to reduce the size of the plaintext or key space to be searched. DES exhibits a strong avalanche effect

Strength of DES – Key Size

IDOL

- 56-bit keys have 256 = 7.2 x 1016 values
- brute force search looks hard
- recent advances have shown is possible
    - in 1997 on Internet in a few months
    - in 1998 on dedicated h/w (EFF) in a few days
    - in 1999 above combined in 22hrs!
- still must be able to recognize plaintext
- must now consider alternatives to DES

DES Design Criteria

- as reported by Coppersmith in [COPP94]
- 7 criteria for S-boxes provide for
    - non-linearity
    - resistance to differential cryptanalysis
    - good confusion
- 3 criteria for permutation P provide for
    - increased diffusion

Double DES

To address the discomfort, some researchers suggest using a double encryption for greater secrecy. The double encryption works in the following way. Take two keys, *k*1 and *k*2, and perform two encryptions, one on top of the other: *E*(*k*2, *E*(*k*1a*m*)). In theory, this approach should multiply the difficulty of breaking the encryption, just as two locks are harder to pick than one.

IDOL

Triple DES

The so-called triple DES procedure is $C = E(k3, E(k2, E(k1am)))$. That is, you encrypt with one key, decrypt with the second, and encrypt with a third. This process gives a strength equivalent to a 112-bit key (because the double DES attack defeats the strength of one of the three keys).

A minor variation of triple DES, which some people also confusingly call triple DES, is $C = E(k1, D(k2, E(k1am)))$. That is, you encrypt with one key, decrypt with the second, and encrypt with the *first* again.

2.6 The AES Encryption Algorithm

The AES Contest

In January 1997, NIST called for cryptographers to develop a new encryption system. As with the call for candidates from which DES was selected, NIST made several important restrictions. The algorithms had to be

- Unclassified
- publicly disclosed
- Available royalty-free for use worldwide
- Symmetric block cipher algorithms, for blocks of 128 bits
- Usable with key sizes of 128, 192, and 256 bits

In August 1998, fifteen algorithms were chosen from among those submitted; in August 1999, the field of candidates was narrowed to five finalists. The five then underwent extensive public and private scrutiny. The final selection was made on the basis not only of security but also of cost or efficiency of operation and ease of implementation in software. The winning algorithm, submitted by two Dutch cryptographers, was Rijndael.

IDOL

Origins

- clear a replacement for DES was needed
    - have theoretical attacks that can break it
    - have demonstrated exhaustive key search attacks
- can use Triple-DES – but slow, has small blocks
- US NIST issued call for ciphers in 1997
- 15 candidates accepted in Jun 98
- 5 were shortlisted in Aug-99
- Rijndael was selected as the AES in Oct-2000
- issued as FIPS PUB 197 standard in Nov-2001

The AES Cipher - Rijndael

- designed by Rijmen-Daemen in Belgium
- has 128/192/256 bit keys, 128 bit data
- an **iterative** rather than **feistel** cipher
    - processes data as block of 4 columns of 4 bytes
    - operates on entire data block in every round
- designed to be:
    - resistant against known attacks
    - speed and code compactness on many CPUs
    - design simplicity

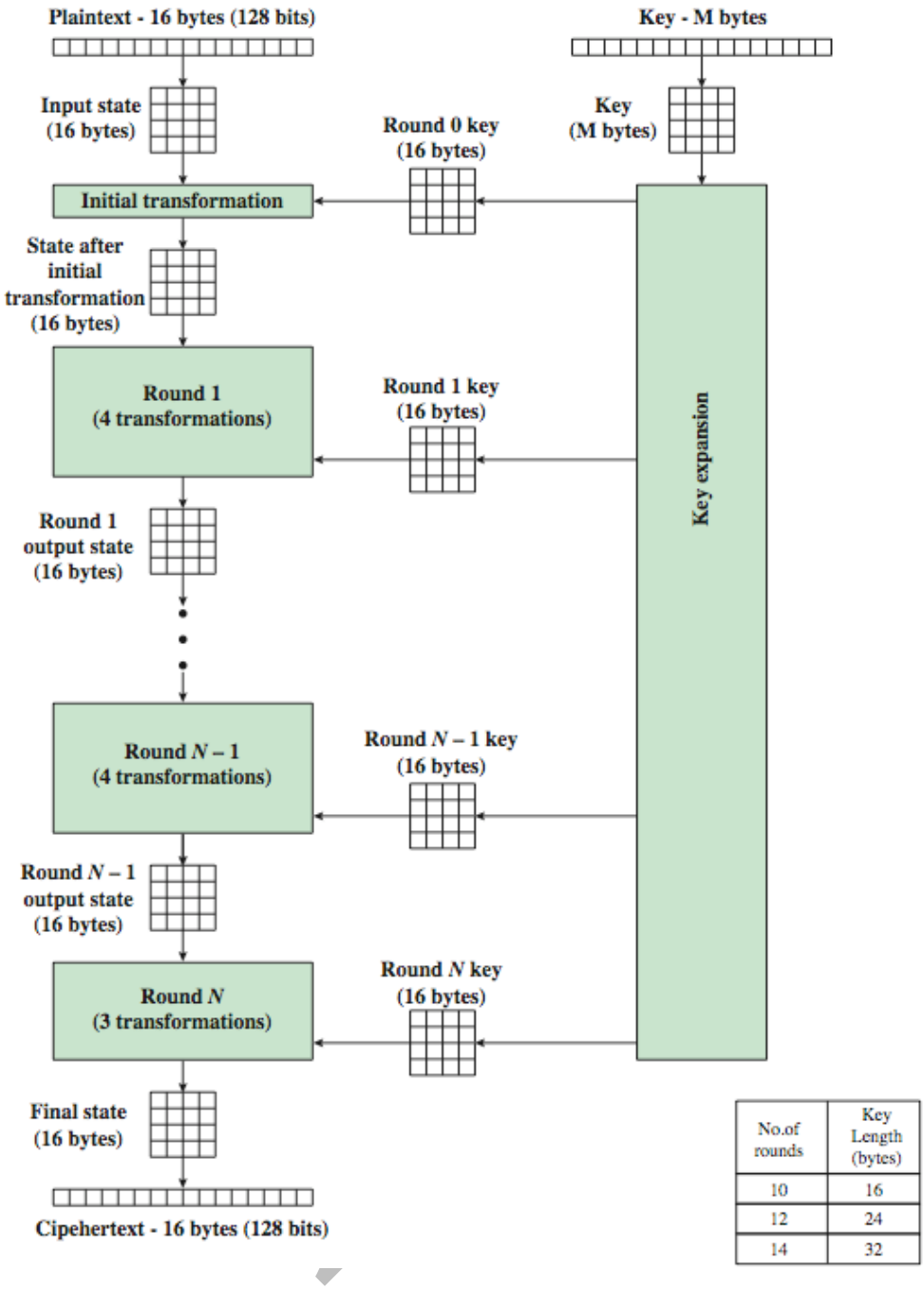
AES Structure

- data block of 4 columns of 4 bytes is state
- key is expanded to array of words

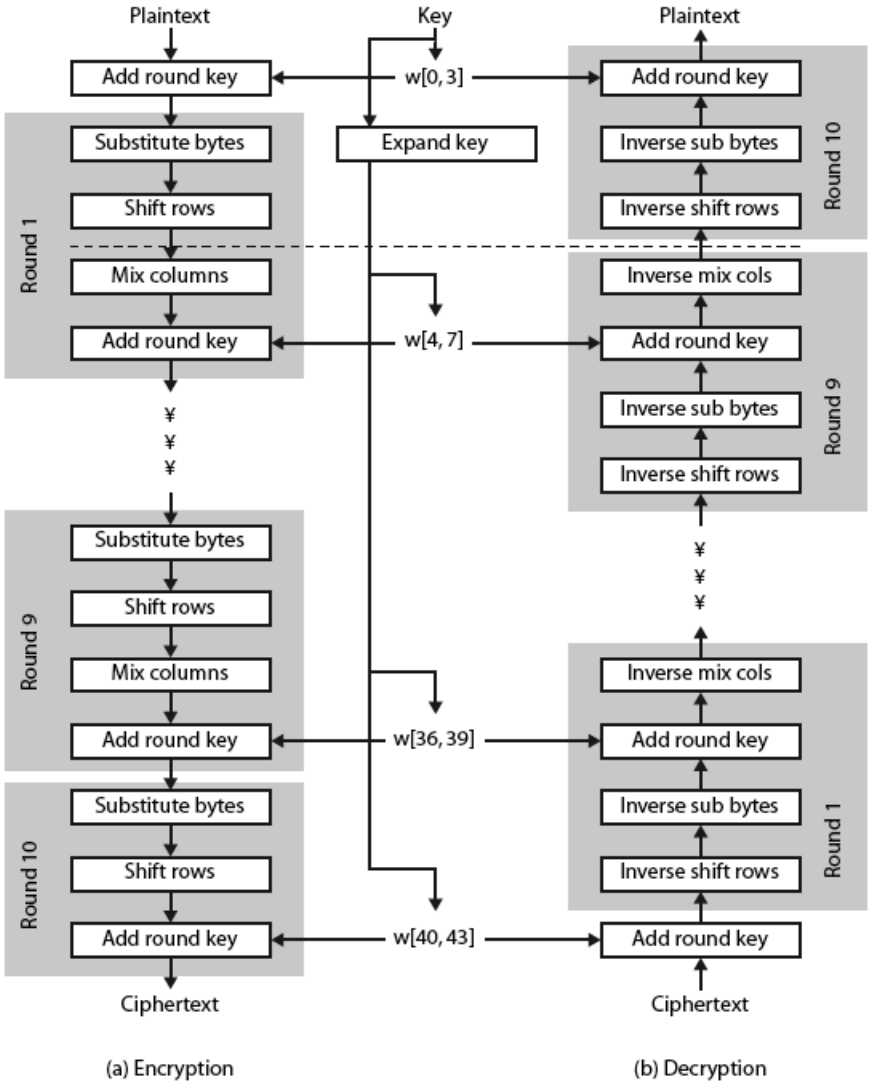
IDOL

- has 9/11/13 rounds in which state undergoes:
  - byte substitution (1 S-box used on every byte)
  - shift rows (permute bytes between groups/columns)
  - mix columns (subs using matrix multiply of groups)
  - add round key (XOR state with key material)
  - view as alternating XOR key & scramble data bytes
- initial XOR key material & incomplete last round
- with fast XOR & table lookup implementation

IDOL



No.of rounds	Key Length (bytes)
10	16
12	24
14	32



1. an **iterative** rather than **feistel** cipher
2. key expanded into array of 32-bit words
    1. four words form round key in each round
3. 4 different stages are used as shown
4. has a simple structure
5. only AddRoundKey uses key
6. AddRoundKey a form of Vernam cipher
7. each stage is easily reversible

8. decryption uses keys in reverse order
9. decryption does recover plaintext
10. final round has only 3 stages

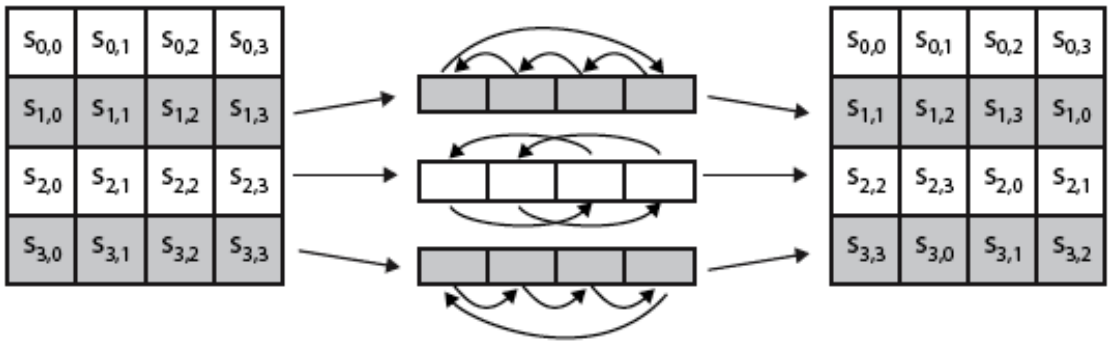
Substitute Bytes

- a simple substitution of each byte
- uses one table of 16x16 bytes containing a permutation of all 256 8-bit values
- each byte of state is replaced by byte indexed by row (left 4-bits) & column (right 4-bits)
    - eg. byte {95} is replaced by byte in row 9 column 5
    - which has value {2A}
- S-box constructed using defined transformation of values in GF(28)
- designed to be resistant to all known attacks

Shift Rows

- a circular byte shift in each each
    - 1st row is unchanged
    - 2nd row does 1 byte circular shift to left
    - 3rd row does 2 byte circular shift to left
    - 4th row does 3 byte circular shift to left
- decrypt inverts using shifts to right
- since state is processed by columns, this step permutes bytes between the columns

IDOL



87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

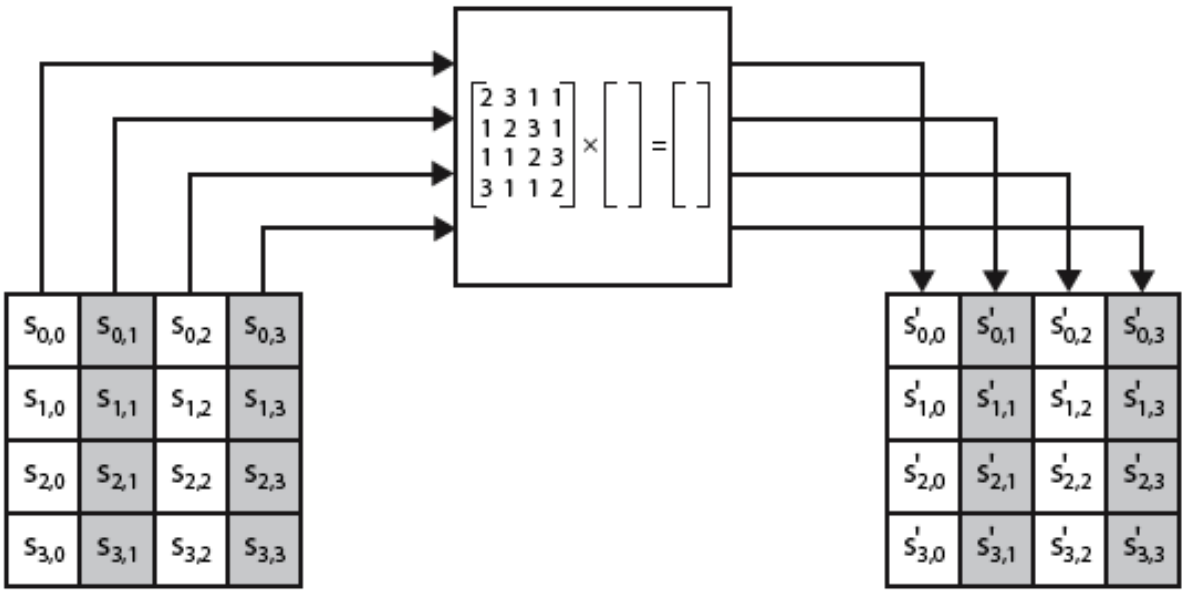
→

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

Mix Columns

- each column is processed separately
- each byte is replaced by a value dependent on all 4 bytes in the column
- effectively a matrix multiplication in GF(28) using prime poly m(x) =x8+x4+x3+x+1

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$



87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

→

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

$(\{02\} \bullet \{87\}) \oplus (\{03\} \bullet \{6E\}) \oplus \{46\} \oplus \{A6\} = \{47\}$

$\{87\} \oplus (\{02\} \bullet \{6E\}) \oplus (\{03\} \bullet \{46\}) \oplus \{A6\} = \{37\}$

$\{87\} \oplus \{6E\} \oplus (\{02\} \bullet \{46\}) \oplus (\{03\} \bullet \{A6\}) = \{94\}$

$(\{03\} \bullet \{87\}) \oplus \{6E\} \oplus \{46\} \oplus (\{02\} \bullet \{A6\}) = \{ED\}$

- can express each col as 4 equations
  - to derive each new byte in col
- decryption requires use of inverse matrix
  - with larger coefficients, hence a little harder

- have an alternate characterisation
    - each column a 4-term polynomial
    - with coefficients in GF(28)
    - and polynomials multiplied modulo (x4+1)
- coefficients based on linear code with maximal distance between code words

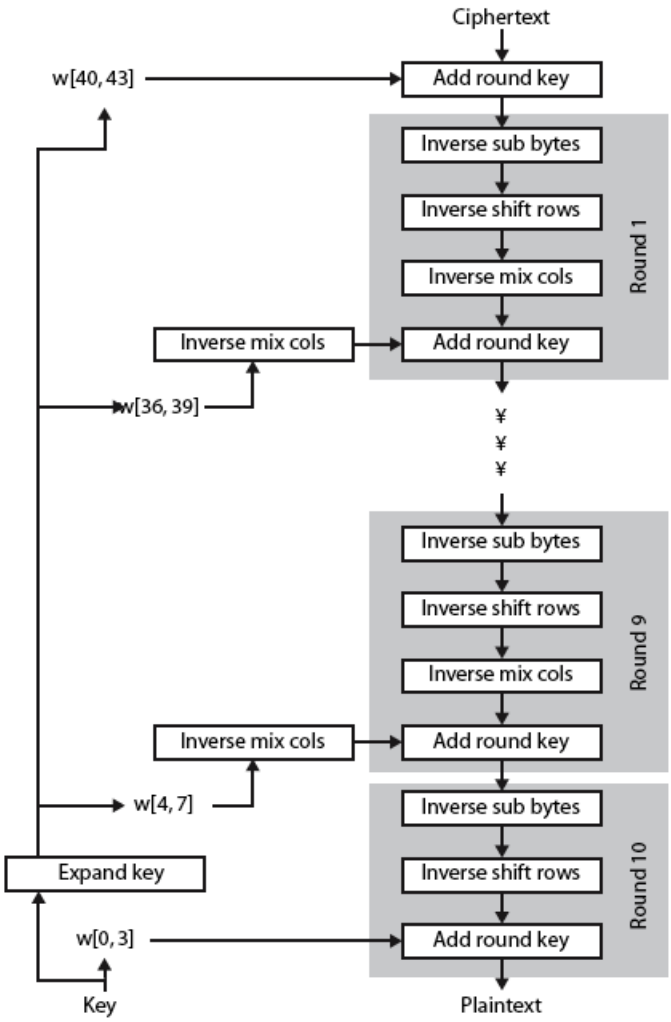
Add Round Key

- XOR state with 128-bits of the round key
- again processed by column (though effectively a series of byte operations)
- inverse for decryption identical
    - since XOR own inverse, with reversed keys
- designed to be as simple as possible
    - a form of Vernam cipher on expanded key
    - requires other stages for complexity / security

AES Decryption

- AES decryption is not identical to encryption since steps done in reverse
- but can define an equivalent inverse cipher with steps as for encryption
    - but using inverses of each step
    - with a different key schedule
- works since result is unchanged when
    - swap byte substitution & shift rows
    - swap mix columns & add (tweaked) round key

IDOL


Ciphertext
w[40, 43]
Add round key
Inverse sub bytes
Inverse shift rows
Inverse mix cols
Round 1
Inverse mix cols
Add round key
w[36, 39]
¥
¥
¥
Inverse sub bytes
Inverse shift rows
Inverse mix cols
Round 9
Inverse mix cols
Add round key
w[4, 7]
Inverse sub bytes
Inverse shift rows
Round 10
Expand key
w[0, 3]
Add round key
Key
Plaintext

Comparison of DES and AES

	DES	AES
Date	**1976**	**1999**
Block size	**64 bits**	**128 bits**
Key length	**56 bits (effective length)**	**128,192,256 bits(and possibly more)**
Encryption primitives	**Substitution, permutation**	**Substitution, shift, bit mixing**
Cryptographic primitives	**Confusion, diffusion**	**Confusion, diffusion**
Design	**Open**	**Open**
Design rationale	**Closed**	**Open**
Selection process	**Secret**	**Secret, but accepted open public comment**
Source	**IBM, enhanced by NSA**	**Independent Dutch cryptographers**

IDOL

Public Key Encryption

In 1976, Diffie and Hellman [DIF76] proposed a new kind of encryption system. With a public key[5] encryption system, each user has a key that does not have to be kept secret. Although counterintuitive, in fact the public nature of the key does not compromise the secrecy of the system. Instead, the basis for public key encryption is to allow the key to be divulged but to keep the decryption

technique secret. Public key cryptosystems accomplish this goal by using two keys: one to encrypt and the other to decrypt.

Asymmetric or public key encryption systems use two keys, a public key and a private key. Unfortunately, a few people call a symmetric or secret key system a “private key “system. This terminology is confusing. We do not use it in this book, but you should be aw are that you might encounter the terminology in other readings.

In a public key or asymmetric encryption system, each user has two keys: a public key and a private key. The user may publish the public key freely because each key does only half of the encryption and decryption process. The keys operate as inverses, meaning that one key undoes the encryption provided by the other key.

To see how, let k_{PRIV} be a user's private key, and let k_{PUB} be the corresponding public key. Then, encrypted plaintext using the public key is decrypted by application of the private key; we write the relationship as:$P = D(k_{PRIV}, E(k_{PUB}, P))$

That is, a user can decode with a private key what someone else has encrypted with the corresponding public key. Furthermore, with some public key encryption algorithms, including RSA, we have this relationship:

$P = D(k\text{PUB}, E(k\text{PRIV}, P))$

In other words, a user can encrypt a message with a private key, and the message can be revealed only with the corresponding public key. These two properties tell us that public and private keys can be applied in either order. In particular, the decryption function D can be applied to any argument so that we can decrypt before we encrypt. With conventional encryption, we seldom think of decrypting

IDOL

before encrypting. But the concept makes sense with public keys, where it simply means applying the private transformation first and then the public one.

We have noted that a major problem with symmetric encryption is the sheer number of keys a single user has to store and track. With public keys, only two keys are needed per user: one public and one private. Let us see what difference this makes in the number of keys needed. Suppose we have three users, B, C, and D, who must pass protected messages to user A as well as to each other. Since each distinct pair of users needs a key, each user would need three different keys; for instance, A would need a key for B, a key for C, and a key for D. But using public key encryption, each of B, C, and D can encrypt messages for A by using A's public key. If B has encrypted a message using A's public key, C cannot decrypt it, even if C knew it was encrypted with A's public key. Applying A's public key twice, for example, would not decrypt the message. (We assume, of course, that A's private key remains secret.) Thus, the number of keys needed in the public key system is relatively small.

Comparing Secret Key and Public Key Encryption:

	Secret Key (Symmetric)	**Public Key (Asymmetric)**
Number of keys	**1**	**2**
Protection of key	**Must be kept secret**	**One key must be kept secret; the other can be freely exposed**

IDOL

Best uses	**Cryptographic workhorse; secrecy and integrity of data single characters to blocks of data, messages, files**	**Key exchange, authentication**
Key	**Must be out-of-band**	**Public key can be used to distribute other keys**
Distribution Speed	**Fast**	**Slow; typically, 10,000 times slower than the secret key**

2.7. Public Key Encryption

So far, we've viewed encryption algorithms from the idea of view of earning the scrambling an easy task to do (so the sender can simply encrypt a note) as well as the decryption possible for the receiver, however, not to have an intruder. But this useful view of changing plaintext to ciphertext is an area of the picture. We should also study the function of tips in encryption. We've noted how beneficial keys could be in deterring an intruder, but we've assumed that the main element must remain key for it to work. In this area, we appear at methods to allow the primary to be the general population but still secure the meaning. We also concentrate on the RSA algorithm, an open key system that is clearly a preferred commercial-grade encryption approach.

In 1976, Diffie and Hellman suggested a new sort of encryption system. Using a public primary encryption technique, each user includes a key that will not need to be kept magic formula. Although counterintuitive, actually the public dynamics of the main element does not bargain the secrecy of the machine. Instead, the foundation for public essential encryption would be to allow the essential to come to be divulged but to help keep the decryption strategy secret. Public essential cryptosystems make this happen goal through the use of two tips: someone to encrypt and another to decrypt.

Asymmetric or general public key encryption methods use two tips, a public major and an exclusive key. Unfortunately, some individuals contact asymmetric or top secret key technique a "private crucial" program. This terminology can be confusing. We usually do not use it in this particular book, nevertheless, you must be aware that you may face the terminology in different readings.

Motivation

Why should producing the key people be attractive? With the standard symmetric key technique, each couple of users requires a separate main. But with general population key techniques, anyone utilizing a single public primary can send out a secret meaning to an end user, and the information remains adequately secured from being study by an interceptor. Why don't we investigate why that is so.

Recall that generally, an n-user technique demands n * (n - 1)/2 secrets, and each customer must track please remember a key for every other end user with which she or he wants to converse. As the amount of users grows, the amount of keys increases incredibly rapidly, as proven in Figure 2.10. Identifying and distributing these secrets is trouble. More serious is usually maintaining protection for the tips already sent out, because we can not expect customers to memorize numerous keys.

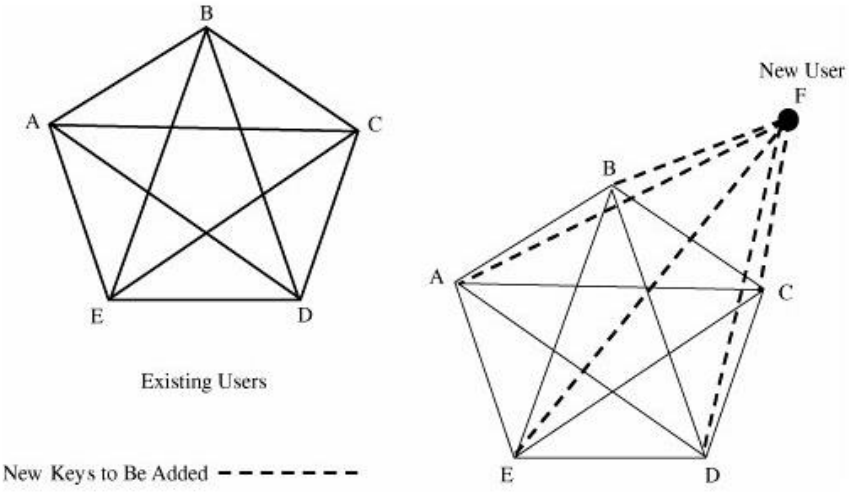


Figure 2.10. Key Proliferation.

Characteristics

We can decrease the problem of essential proliferation with a public key technique. In a common major or asymmetric encryption technique, each user possesses two secrets: a common key and an exclusive key. An individual may publish the general public key openly because each essential does only 1 / 2 of the encryption and decryption procedure. The keys run as inverses, and therefore one essential undoes the encryption supplied by the other key element.

To observe how, let k_{PRIV} be considered a user's private primary, and allow k_{PUB}function as corresponding public primary. After that, encrypted plaintext utilizing the public key can be decrypted by the program of the exclusive key; we publish the partnership as

$P = D(k_{PRIV}, E(k_{PUB}, P))$

That's, a consumer can decode with an exclusive key what another person has encrypted along with the corresponding public essential. On top of that, with some open main encryption algorithms, integrating RSA, we have this partnership:

$P = D(k_{PUB}, E(k_{PRIV}, P))$

Quite simply, a customer can encrypt a note with an exclusive key, as well as the message could be revealed only while using corresponding public primary. These two houses reveal that open public and private tips can be used in either purchase. Specifically, the decryption functionality D could be put on any argument in order that we are able to decrypt before we encrypt. With normal encryption, we hardly ever think about decrypting before encrypting. However, the concept is practical with public secrets, where it basically means using private change first and the general public one.

We have known that a significant problem with symmetric encryption may be the sheer amount of keys an individual user must store and observe. With public tips, only two tips are essential per person: one people and one exclusive. Let us find what variation this creates in the number of keys needed. Imagine we've three consumers, B, C, and D, who must move protected text messages to user A in addition to one another. Since each particular pair of customers' needs a major, each user would want three different tips; for example, A would want an integral for B, an integral for C, and an integral for D. But employing public key element

encryption, all of B, C, and D can encrypt information for A through the use of A's public primary. If B has got encrypted a note using A's open major, C cannot decrypt it, even though C knew it had been encrypted with A's general population key. Using A's public key element twice, for instance, wouldn't normally decrypt the information. (We assume, needless to say, that A's exclusive key remains hidden knowledge.) Thus, the amount of keys required in the general public key system is certainly relatively small.

The characteristics of the secret key and public key algorithms are compared in Table 2.5.

Table 2.5. Comparing Secret Key and Public Key Encryption.

	Secret Key (Symmetric)	**Public Key (Asymmetric)**
Number of keys	1	2
Protection of key	Must be kept secret	One key must be kept secret; the other can be freely exposed
Best uses	Cryptographic workhorse; secrecy and integrity of data single characters to blocks of data, messages, files	Key exchange, authentication
Key distribution	Must be out-of-band	Public key can be used to distribute other keys
Speed	Fast	Slow; typically, 10,000 times slower than secret key

RivestShamirAdelman Encryption

The **RivestShamirAdelman(RSA)** cryptosystem is really a public key method. Predicated on an underlying tough problem and called following its three inventors, this algorithm had been unveiled in 1978 also to date remains safe and

IDOL

sound. RSA has become the main topic of extensive cryptanalysis, no serious flaws have got yet been observed. Although the quantity of analysis is not any guarantee of any method's protection, our self-confidence in the technique grows after a while without the discovery of a flaw.

RSA depends on a location of mathematics referred to as number theory, where mathematicians study qualities of numbers such as for example their prime variables. The RSA encryption algorithm brings together results from quantity theory with the number of problems in figuring out the prime components of the confirmed number. As perform a number of the other algorithms we've examined, the RSA algorithm furthermore performs with arithmetic mod n.

The two secrets found in RSA, d, and e, are employed for decryption and encryption. They're actually compatible: Either could be chosen because the public major, but one possessing been chosen, another one should be kept, individual. For ease, we contact the encryption major e plus the decryption key element d. Also, due to the nature of the RSA algorithm, the tips can be used in either order:

$P = E(D(P)) = D(E(P))$

(You can think about E and D as two complementary capabilities, all of which "undoes" another.)

Any plaintext block P is usually encrypted as P^e mod n. As the exponentiation is conducted mod n, factoring P^e to discover the encrypted plaintext is certainly difficult. Nevertheless, the decrypting primary d is diligently chosen in order that $(P^e)^d$ mod n = P. So, the legitimate recipient who understands d easily computes $(P^e)^d$ mod n = P and recovers P and never have to factor P^e.

The encryption algorithm is dependant on the underlying issue of factoring good sized quantities. So far, no one has determined a shortcut or a simple and easy way to point large numbers within a finite set known as a discipline. In extremely technical but superb papers, Boneh critiques all the recognized cryptanalytic disorders on RSA and concludes that nothing is significant. As the factorization problem may be open for quite some time, most cryptographers think about this problem a good basis for any secure cryptosystem.

2.8. The Uses of Encryption

Encryption algorithms by itself are not the solution to everyone's encryption desires. Although encryption implements guarded communications channels, it is also used for different duties. Actually, incorporating symmetric and asymmetric encryption generally capitalizes on the very best top features of each.

Public essential algorithms are of help only for specific tasks because they're very sluggish. A public key element encryption may take 10,000 periods as long to execute like a symmetric encryption as the root modular exponentiation depends upon multiplication and division, which can be inherently slower compared to the bit functions (addition, exclusive Or perhaps, substitution, and shifting) which symmetric algorithms happen to be based. Because of this, symmetric encryption may be the cryptographers' "workhorse," and general public key encryption is usually reserved for particular, infrequent makes use of, where slow functioning is not an ongoing problem.

Let us appear more meticulously at four software of encryption: cryptographic hash features, key exchange, electronic digital signatures, and certificates.

Cryptographic Hash Functions

Encryption is mostly useful for secrecy; we normally encrypt something in order that its contentsor perhaps its existenceare unidentified to all or any but a privileged crowd. In some instances, however, integrity is really a more important issue than secrecy. For instance, in a doc retrieval system including legal records, it might be important to understand that the backup retrieved is strictly what was stashed. Likewise, in a very secure communications method, the necessity for the right transmission of information may override secrecy considerations. Let us take a look at how encryption supplies integrity.

In most data files, sun and rain or the different parts of the file aren't bound together at all. That's, each byte or tad or character can be independent of each various other ones in the record. This insufficient binding implies that changing one benefit influences the integrity in the file, but that certain change can simply go undetected.

What we wish to do can be somehow set a seal or protect around the document so that we are able to detect once the seal is broken and therefore understand that something is changed. This idea is comparable to the usage of polish seals on

words in medieval times; if the polish was damaged, the receiver would understand that someone had damaged the seal and browse the message inside. Just as, cryptography may be used to seal a record, encasing it in order that any change gets apparent. One method for delivering the seal would be to compute a cryptographic functionality, sometimes referred to as a hash or checksum or message digest with the file.

The hash purpose has special features. For example, some encryptions are determined by a function that's clear to see but tricky to compute. For a straightforward example, think about the cube feature, $y = x^3$. It really is not too difficult to compute x^3 yourself, with pencil and document, or having a calculator. However, the inverse function $\sqrt[3]{y}$ is a lot more challenging to compute. And the function $y = x^2$ does not have any inverse functionality since you can find two opportunities for $\pm\sqrt{y}$: +x and -x. Capabilities like these, which can be easier to compute than their inverses, are usually called one-way features.

A one-way function can be handy within an encryption algorithm. The event must be determined by all items of the file getting covered, so any shift to a good single little will adjust the checksum consequence. The checksum price is stored using the file. Then, every time the file is certainly accessed or employed, the checksum will be recomputed. When the computed checksum fits the stored benefit, chances are that the document is not changed.

A cryptographic function, like the DES or AES, is particularly appropriate for closing principles, since an outsider won't know the main element and thus will never be able to adjust the stored benefit to complement with data getting revised. For low-threat software, algorithms also simpler than DES or AES may be used. In stop encryption strategies, chaining method linking each stop to the prior block's price (and for that reason to all earlier blocks), for instance, by using a special OR to blend the encrypted earlier block with all the encryption of the existing one. A file's cryptographic checksum may be the last block with the chained encryption of an data file since that stop depends on all the blocks.

The most trusted cryptographic hash capabilities happen to be MD4, MD5 (where MD means Message Process), and SHA/SHS (Secure Hash Algorithm or Common). The MD4/5 algorithms had been developed by Ron Rivest and RSA Laboratories. MD5 can be an improved variant of MD4. Both condense a note of any sizing into

a 128-bit break down. SHA/SHS is comparable to both MD4 and MD5; it makes a 160-touch digest.

Wang et al. own announced cryptanalysis episodes on SHA, MD4, and MD5. For SHA, the assault can locate two plaintexts that create exactly the same hash process in roughly 263 steps, very good lacking the 280 methods that might be expected of your 160-little bit hash function, and incredibly simple for a reasonably well-financed attacker. Although this strike does not indicate SHA is pointless (the attacker must accumulate and analyze a lot of ciphertext examples), it can suggest usage of very long digests and prolonged tips. NIST possesses studied the episode carefully and advised countermeasures.

Key Exchange

Suppose you will need to deliver a protected subject matter to someone you don't know and would you not find out you. This example is more prevalent than you might think. For example, you might send your earnings tax go back to the government. You need the information to become protected, nevertheless, you do not actually know the one who is receiving the info. Similarly, you might use your online browser for connecting with a purchasing web site, trade personal (encrypted) e-mail, or request two hosts to determine a protected route. Each one of these situations depends upon having the ability to swap an encryption type in such a approach that no one else can intercept it. The issue of two earlier unknown gatherings exchanging cryptographic secrets is both difficult and important.

Public crucial cryptography might help. Since asymmetric secrets come in sets, one half of this pair could be exposed without reducing the other one half. To observe how, imagine S and R (our well-known sender and device) desire to derive a distributed symmetric key. Imagine likewise that S and R both contain public secrets for a standard encryption algorithm; contact these kPRIV-S, kPUB-S, kPRIV-R, and kPUB-R, for any private and general public tips for S and R, respectively. The easiest solution is usually for S to select any symmetric essential K, and deliver E(kPRIV-S,K) to R. After that, R needs S's public major, takes out the encryption, and obtains K. Alas, any eavesdropper who is able to get S's general public key may also obtain K.

Instead, permit S give E(kPUB-R, K) to R. Then simply, simply R can decrypt K. However, R does not have any guarantee that K originated from S.

But there's a useful alternative. The answer is made for S to deliver to R:

E(kPUB-R, E(kPRIV-S, K))

We can consider this exchange with regards to lockboxes and tips. If S really wants to send something covered to R (like a credit card variety or a group of medical data), then your exchange works something similar to this. S places the protected data inside a lockbox that may be opened simply with S's common key. After that, that lockbox is certainly put in the second lockbox that may be opened just with R's personal key. R may then use his exclusive main to open the external box (something just he can perform) and make use of S's public essential to open the interior box (showing that the deal originated from S). Quite simply, the standard protocol wraps the secured details in two deals: the initial unwrapped just with S's general public key, and the next unwrapped just with R's exclusive key. This process will be illustrated in Figure 2.11.

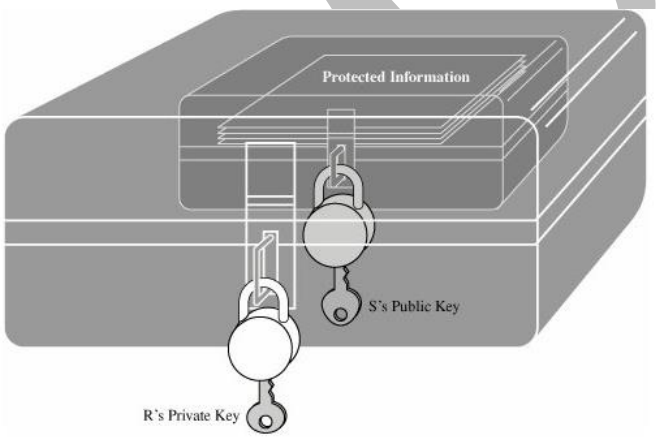


Figure 2.11. The Idea Behind Key Exchange.

Another approach not necessarily requiring pre-shared community keys may be the so-called DiffieHellman key exchange protocol. On this process, S and R employ some very simple arithmetic to switch a secret key. They acknowledge a field dimensions n as well as aintial number g; they are able to communicate these number in the apparent. Each believes up a top secret number, state, s and r. S directs to R g^s and R transmits to S g^r. After that S computes $(g^r)^s$ and R

computes $(g^s)^r$, which will be the very same, so $g^{rs} = g^{sr}$ turns into their shared secret keys.

Digital Signatures

Another typical problem parallels a standard human have to have: an purchase to transfer money from one particular person to another. Quite simply, you want to have the ability to send electronically the same as a computerized take a look at. We know how this transaction will be handled in the traditional, paper function:

A check is really a tangible item authorizing a monetary transaction.

The signature within the check out confirms authenticity because (presumably) just the respectable signer can create that signature.

Regarding an alleged forgery, an authorized can be known as in to appraise authenticity.

Once a test is cashed, it really is canceled such that it cannot be used again.

The paper look at isn't alterable. Or, virtually all types of alteration are often detected.

Transacting organization by check depends upon tangible objects in the prescribed web form. But tangible items do not can be found for dealings on computers. Thus, authorizing repayments by computer takes a different model. Why don't we consider the prerequisites of this type of situation, both through the standpoint of any bank and through the standpoint of your user

Suppose Sandy transmits her bank a note authorizing it to copy $100 to Tim. Sandy's bank or investment company must be in a position to verify and establish that the communication really originated from Sandy if she should soon after disavow mailing the message. The lender also really wants to understand that the message is certainly entirely Sandy's, that this is not altered on the way. On her component, Sandy really wants to ensure that her loan provider cannot forge many of these messages. Both functions desire to be sure the message will be new, not just a reuse of an previous message, and this it is not altered during transmitting. Using electronic alerts instead of papers complicates this technique.

IDOL

But we've ways to produce the process job. A digital signature bank is a standard protocol that produces exactly the same effect as a genuine signature: This is a mark that just the sender could make, but other folks can easily identify as from the sender. As being a real signature, an electronic signature can be used to confirm contract to a note.

Properties

A digital signature bank must connect with two primary circumstances:

It should be unforgeable. If individual P signs subject matter M with trademark S(P,M), it really is impossible for anybody else to create the match [M, S(P,M)].

It should be authentic. In case a person R obtains the match [M, S(P,M)] purportedly from P, R can be sure the signature is actually from P. Just P may have created this trademark, and the trademark is firmly mounted on M.

These two specifications, shown in Figure 2.12, will be the important hurdles in laptop transactions. Two even more properties, also attracted from parallels together with the paper-based environment, will be desirable for deals completed with aid from digital signatures:

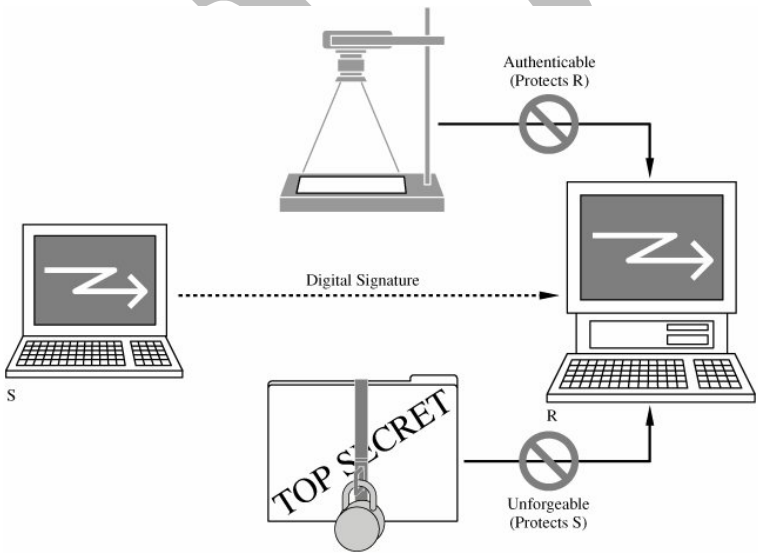


Figure 2.12. Requirements for a Digital Signature.

It isn't alterable. After becoming transmitted, M can't be evolved by S, R, or an interceptor.

It isn't reusable. A earlier message presented once more will be immediately recognized by R.

To observe how digital signatures job, we first gift a device that meets the initial two specifications. We adding to that treatment for satisfy the various other requirements.

Public important encryption systems are usually ideally suitable for electronic digital signatures. For very simple notation, why don't we assume that the general public main encryption for consumer U is seen through $E(M, K_U)$ and that the individual key change for U will be authored as $D(M,K_U)$. We are able to think about E because the privacy change (since sole U can decrypt it) and D because the authenticity change (since just U can develop it). Remember, nevertheless, that under some asymmetric algorithms such as for example RSA, D and E happen to be commutative, and each one can be put on any message. As a result,

$D(E(M,),) = M = E(D(M,),)$

If S needs to mail M to R, S makes use of the authenticity change to create $D(M, K_S)$. S in that case delivers $D(M, K_S)$ to R. R decodes the concept with the general public key change of S, processing $E(D(M,K_S), K_S) = M$. Since simply S can make a message which makes impression under $E(,K_S)$, the communication must obviously have result from S. This evaluation satisfies the authenticity need.

R helps you to save D(M,KS). If S should soon after allege that this message is really a forgery (not necessarily from S), R can merely present M and $D(M,K_S)$. Anyone can confirm that since $D(M,K_S)$ is definitely altered to M with the general public key change of Sbut simply S may have made $D(M,K_S)$in that case $D(M,K_S)$ should be from S. This check satisfies the unforgeable need.

There are additional approaches to utilizing digital signature bank; some make use of symmetric encryption, others employ asymmetric. The tactic shown below illustrates the way the protocol can deal with certain requirements for unforgeability and authenticity. To include secrecy, S can be applied E(M, K_R) as found in Figure 2.13.

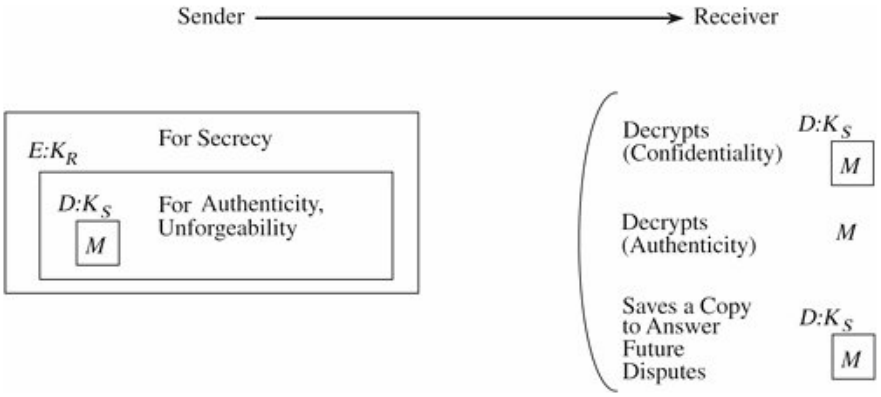


Figure 2.13. Use of Two Keys in Asymmetric Digital Signature.

Certificates

As individuals we establish have confidence in on a regular basis in our regular interactions with folks. We identify men and women we realize by realizing their voices, encounters, or handwriting. At some other times, we employ an affiliation to mention trust. For example, in case a stranger telephones us and we notice, "I represent the neighborhood federal..." or "I'm contacting behalf of the charity..." or "I'm calling in the university/hospital/police about your mom/father/son/daughter/brother/sister...," we might decide to believe the caller even though we have no idea her or him. With regards to the nature of the decision, we may opt to think the caller's affiliation or even to seek independent confirmation. For example, organic beef have the affiliation's range from calling directory and phone the party again. Or we might seek more information through the caller, such as for example "What color coat was she using?" or "Who's the president of one's corporation?" If we've a low amount of trust, we might even behave to exclude an outsider, such as "I'll mail the right to your charity instead of offer you my charge card number."

For each of the interactions, we've what we may call up a "trust threshold," a qualification to which we have been willing to trust an unidentified specific. This threshold prevails in commercial connections, also. When Acorn Creation Company delivers Big Steel Corporation an buy for 10,000 mattress sheets of steel, being shipped inside a week and covered within ten times, have faith in abounds. The purchase is printed with an Acorn form, agreed upon by someone defined as Helene Smudge, Acquiring Agent. Big Metal may begin organizing the steel perhaps before receiving cash from Acorn. Big Metallic may examine Acorn's credit history to choose whether to dispatch the purchase without payment initially. If dubious, Big Metal might phone Acorn and have to talk with Ms. Smudge within the purchasing section. But much more likely Big Steel will in actuality ship the products without figuring out who Ms. Smudge can be, whether she actually is actually the buying agent, whether she actually is authorized to invest in an order of this size, as well as whether the trademark is in fact hers. Oftentimes a transaction such as this develops by fax, in order that Big Steel will not even have a genuine signature on data file. In this case one, which happen daily, trust is dependant on overall look of authenticity (like a printed, signed type), outside details (like a credit file), and urgency (Acorn's demand that the metallic be shipped rapidly).

For electronic interaction to succeed, we should develop similar methods for two gatherings to establish have confidence in without having satisfied. A standard thread inside our personal and company interactions may be the ability to include a person or something attest to the lifetime and integrity of 1 or both get-togethers. The authorities, the Chamber of Business, or the higher Company Bureau vouches to the authenticity of your caller. Acorn indirectly vouches for the truth that Ms. Smudge is certainly its purchasing adviser by transferring the decision to her within the purchasing department. In a way, the telephone firm vouches to the authenticity of a celebration by list it within the directory. This idea of "vouching for" by way of a third party could be a basis for rely upon commercial adjustments where two events have no idea A large firm may have many divisions, each department may have various departments, each office may have different jobs, and each task may have more than a few task communities (with variations within the names, the amount of levels, and the amount of completeness of this hierarchy). The very best executive might not know by label or view every worker in the business, but an activity group leader understands all

associates of the duty group, the task leader has learned all task team leaders, etc. This hierarchy may become the foundation for trust through the entire organization.

A large company could have various divisions, each section may have some departments, each team may have different tasks, and each job may have more than a few task categories (with variations within the names, the amount of levels, and the amount of completeness from the hierarchy). The most notable executive might not know by brand or vision every worker in the business, but an activity group leader has learned all users of the duty group, the job leader is aware all task class leaders, and so forth. This hierarchy may become the foundation for trust through the entire organization.

To observe how, suppose two different people fulfill: Ann and Andrew. Andrew states he performs for exactly the same firm as Ann. Ann desires independent verification he does. She realizes that Costs and Betty are usually two task team leaders for exactly the same project (guided by Camilla); Ann performs for Invoice and Andrew for Betty. (The organizational associations are found in Figure 2.14.) These details offer Ann and Andrew a groundwork for trusting each other's id. The string of verification may be something similar to this:

Ann asks Charge who Andrew is definitely.

Charge either asks Betty if he is aware of her immediately or or even, asks Camilla.

Camilla asks Betty.

Betty replies that Andrew performs for her.

Camilla tells Charge.

Bill conveys to Ann.

IDOL

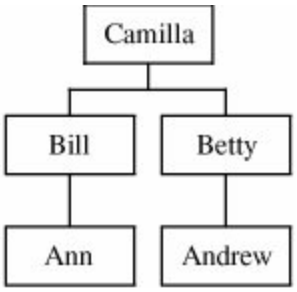


Figure 2.14. Organization in Hypothetical Company.

If Andrew is within a different activity group, it might be necessary to increase inside the organizational tree before a standard point is available.

We can work with a similar method for cryptographic key element exchange, as found in Figure 2-15. If Andrew and Ann desire to communicate, Andrew can provide his public essential to Betty, who goes by it to Camilla or right to Bill, who offers it to Ann. But this collection is not the way it could work in true to life. The key may possibly be along with a note saying it really is from Andrew, which range from some yellow document to an application 947 Affirmation of Identity. And when an application 947 can be used, subsequently Betty would also need to attach an application 632a Transmittal of Individuality, Camilla would connect another 632a, and Costs would attach your final one, as demonstrated in Body 2-15. This string of 632a varieties would say, essentially, "I'm Betty and I acquired this key along with the attached affirmation of identity individually from a particular person I know being Andrew," "I'm Camilla and I obtained this key plus the attached declaration of identity plus the linked transmittal of id personally from the person I understand to become Betty," etc. When Ann will get the main element, she can critique the string of data and conclude with affordable assurance that the main element really did result from Andrew. This process is a method of obtaining authenticated common tips, a binding of an integral, and a trusted identity.

IDOL

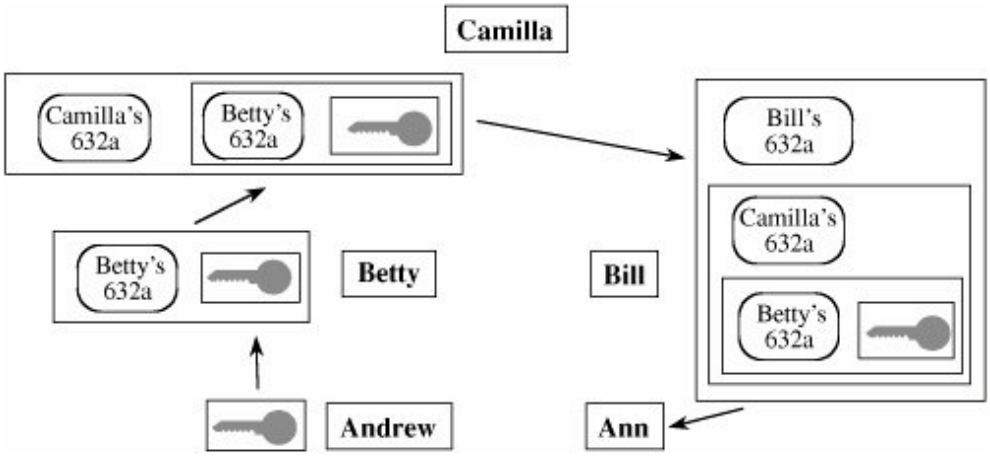


Figure 2.15. Andrew Passes a Key to Ann.

This model is effective within a business since there is always someone frequent to any two workers, even if both employees come in different divisions so the common person may be the president. The procedure bogs down, even so, if Ann, Invoice, Camilla, Betty, and Andrew all need to be obtainable whenever Ann and Andrew desire to speak. If Betty is usually away on a small business trip or Costs is off ill, the process falters. In addition, it can not work well in the event the president cannot find any meaningful job done because each day can be occupied with managing forms 632a.

To address the initial of these complications, Andrew can require his complete string of kinds 632a in the president right down to him. Andrew may then give a duplicate of this complete set in place to anyone in the business who would like his key. Rather than working from underneath up to common level, Andrew starts at the very top and derives his complete chain. He becomes these signatures any moment his superiors can be found, so they need not be accessible when he really wants to hand out his authenticated open public key.

The second issue is settled by reversing the procedure. Instead of beginning in the bottom (with task participants) and attempting to the top in the tree (the leader), we begin at the very top. Andrew thus includes a preauthenticated public main

for unlimited used in the future. Assume the expanded construction in our hypothetical company, demonstrating the president along with other levels, is really as illustrated in Figure 2.16.

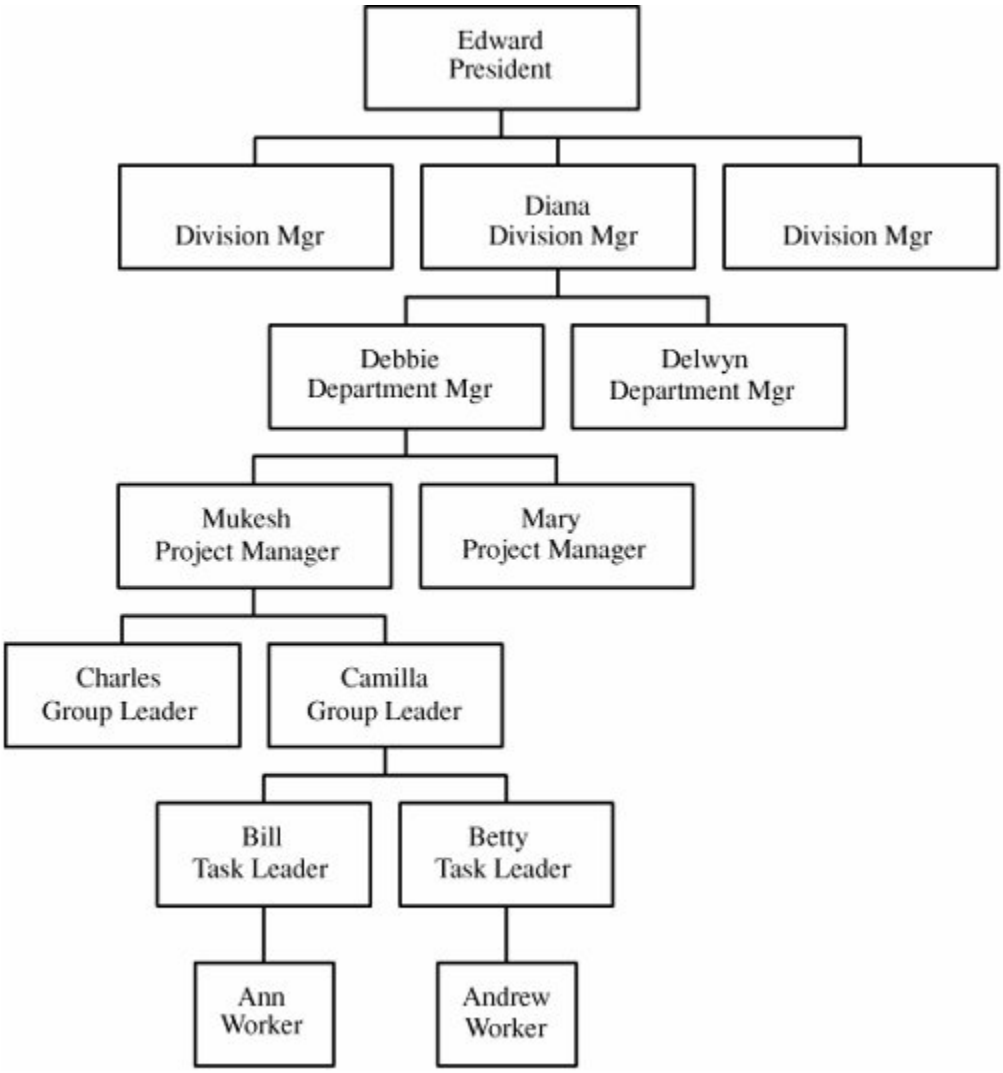


Figure 2.16. Expanded Corporate Structure.

The president makes a letter for every division manager stating "I'm Edward, the chief executive, I verify the identification of division director Diana, whom I understand professionally, and I believe in Diana to verify the identities of her subordinates." Each department manager does in the same way, duplicating the

president's notice with each notice the manager produces, etc. Andrew will get a packet of characters, from the us president down through his process group chief, each letter connected by name to another. If every staff in the business receives this type of packet, any two staff members who wish to exchange authenticated tips need only compare and contrast each other's packets; both packets could have at the very least Edward in keeping, perhaps various other high professionals, and sooner or later will deviate. Andrew and Ann, for instance, could assess their chains, decide that they had been exactly the same through Camilla, and track the bottom components. Andrew recognizes Alice's chain is certainly traditional through Camilla since it is indistinguishable to his string, and Ann has learned exactly the same. Each knows all of those other chain is precise because it uses an unbroken type of brands and signatures.

Certificates to Authenticate an Identity

This protocol can be represented easier electronically than in writing. With paper, it's important to protect against forgeries, to avoid section of one string from being changed and to make sure that the public key element in the bottom will the string. Electronically the whole lot can be carried out with electronic digital signatures and hash features. Kohnfelder appears to be the originator of the idea of using an digital certificate having a string of authenticators, that is extended in Merkle'sdocuments .

A public essential and user's identification are bound together with each other in a certificates, which is then simply signed by somebody referred to as a certificate expert, certifying the correctness on the binding. Inside our example, the business might setup a certificate design in the next way. First of all, Edward chooses a public key element pair, posts the general public component where everyone in the business can get it, and keeps the private aspect. Then, each section manager, such as for example Diana, results in her public major pair, puts the general public key in a note as well as her individuality, and moves the message firmly to Edward. Edward symptoms it by developing a hash worth of the meaning and encrypting the information along with the hash along with his private key element. By putting your signature on the concept, Edward affirms that the general public key (Diana's) as well as the identity (likewise Diana's) inside the message happen to be for exactly the same person. This communication is named Diana's certificate.

Most of Diana's department administrators create messages making use of their public tips, Diana signals and hashes each, and results them. She likewise appends to each a duplicate of the certification she obtained from Edward. In this manner, anyone can confirm a manager's license by you start with Edward's well-known general population primary, decrypting Diana's license to get her public main (and individuality), and employing Diana's public primary to decrypt the manager's qualification. Figure 2.17 demonstrates how certificates are manufactured for Diana and something of her professionals, Delwyn. This technique goes on down the hierarchy to Ann and Andrew. As demonstrated in Figure 2.18, Andrew's license is actually his particular person certificate coupled with all certificates for all those above him inside the line towards the president.

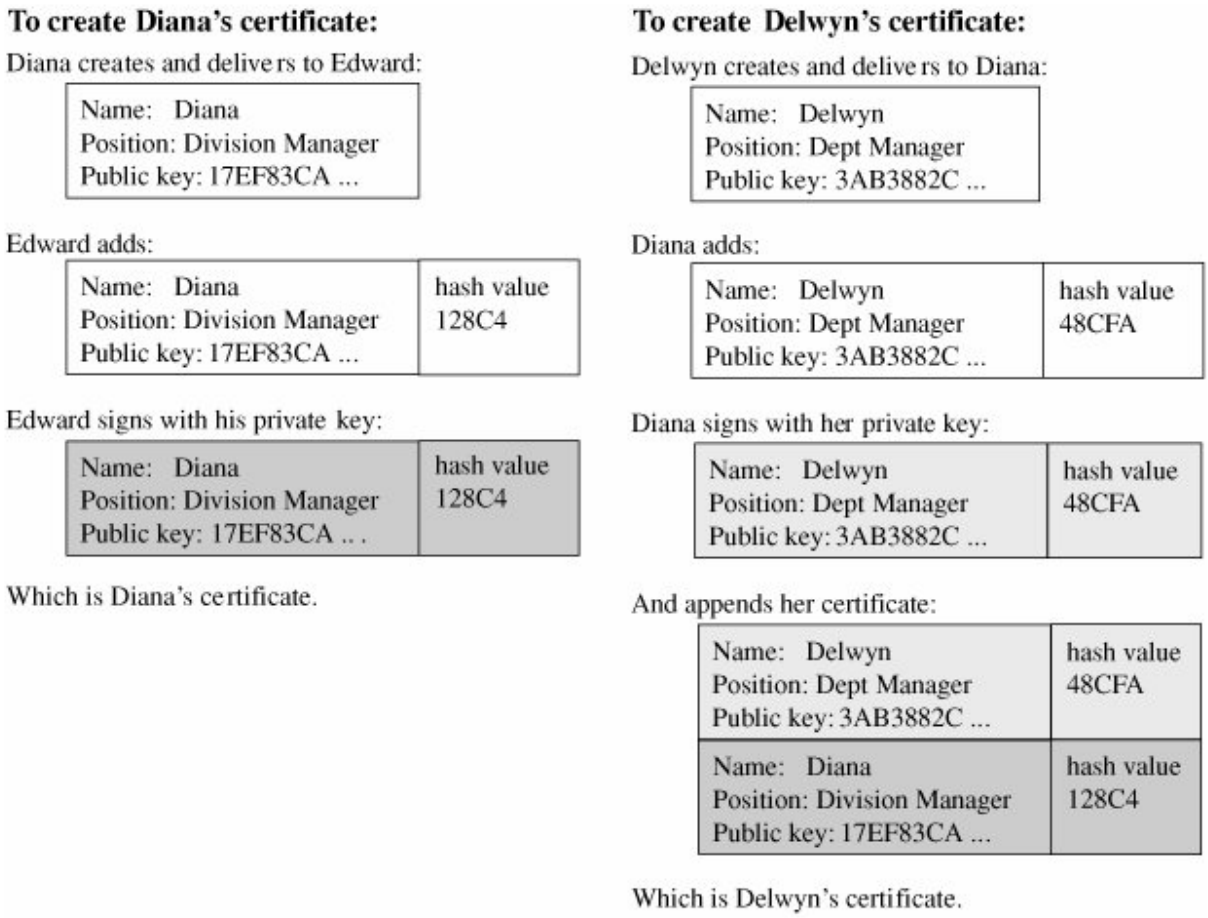


Figure 2.17. Signed Certificates.

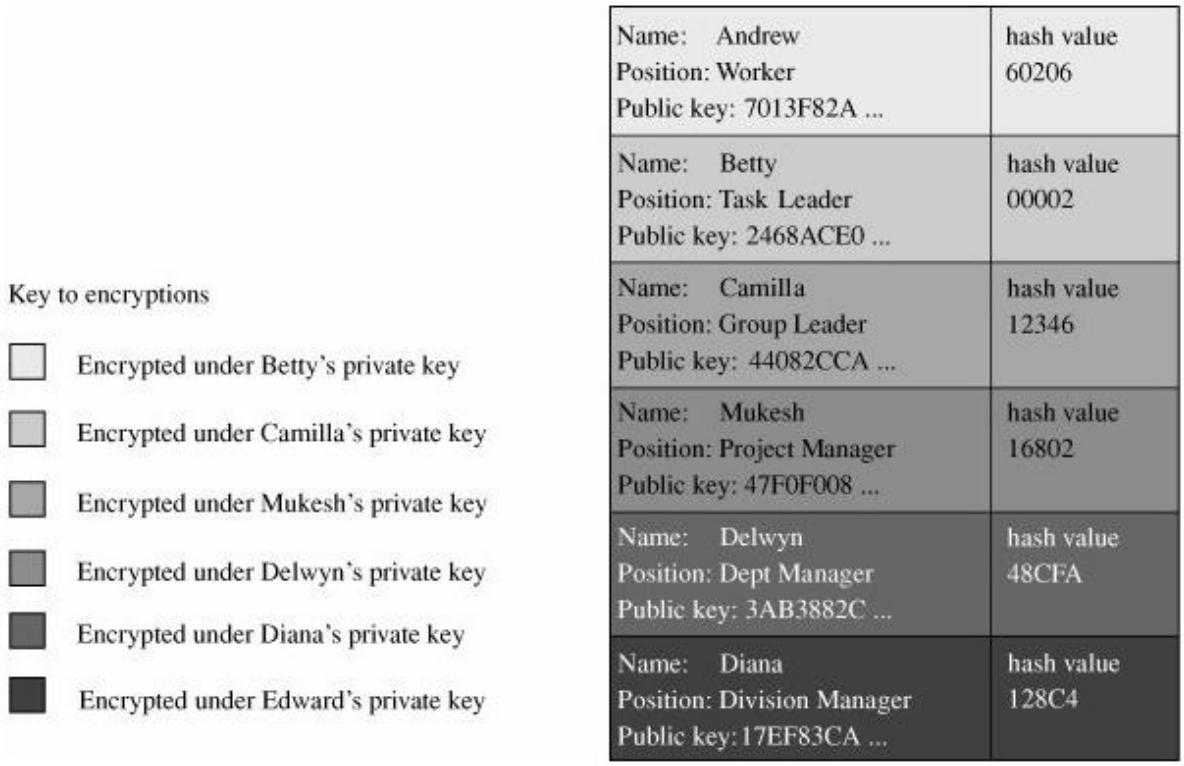


Figure 2.18. Chain of Certificates.

Trust With out a Single Hierarchy

In our illustrations, certificates were granted based on the managerial structure. Nonetheless it is not essential to have this type of structure or even to follow it to utilize certificate putting your signature on for authentication. Anyone who's considered acceptable being an authority can hint a certificate. For instance, if you wish to determine whether an individual received a qualification from a university or college, you would not necessarily contact the us president or chancellor but would rather go directly to the office of documents or the registrar. To check someone's employment, you may ask the workers workplace or the movie director of recruiting. And to look at if someone resides at a specific address, you may consult any office of public record information.

Sometimes, a person is chosen to verify the authenticity or validity of the document or particular person. For instance, a notary common attests for the validity of your (written) signature on the document. Some corporations have a stability officer to confirm that an staff has appropriate security and safety clearances to learn a report or attend a gathering. Many companies include a separate employees office for every web site or each flower location; the staff officer vouches for that employment status on the staff members at that web site. These officers or mind of office buildings could credibly hint certificates for folks under their purview. Organic hierarchies are present in contemporary society, and these identical hierarchies may be used to validate certificates.

The only trouble with a hierarchy may be the need for faith at the very top level. The complete string of authenticity is certainly safe because each document contains the key element that decrypts another certificate, aside from the top. Inside a company, it really is reasonable to believe the person at the very top. But if certificates happen to be to become trusted in electronic business, people should be able to alternate certificates firmly across companies, companies, and countries.

The Internet is really a large federation of systems for intercompany, interorganizational, and overseas (in addition to intracompany, intraorganizational, and intranational) connection. It isn't an integral part of any government, neither is it a privately possessed company. It really is governed by way of a board called the web Society. THE WEB Society has electricity simply because its people, the government authorities and businesses that together constitute the Internet, consent to interact. But there is really no "top" online. Different companies, such as for example C&W HKT, SecureNet, Verisign, Baltimore Technology, Deutsche Telecom, SocietaInterbancaria per l'Automatzione di Milano, Entrust, and Certiposte will be root certification government bodies, this means each is really a highest specialist that symptoms certificates. So, rather than one root and something top, there are lots of roots, largely organised around national limitations.

In this section, we introduced different approaches to key element distribution, which range from direct change to distribution by way of a central distribution

service to certified move forward distribution. We check out the notions of certificates and certificate government bodies in more detail in Section 7, where we discuss Community Essential Infrastructures. But no real matter what approach is taken up to key circulation, each provides its benefits and drawbacks. Points to bear in mind about any important distribution protocol are the following:

- What operational limitations are there? For instance, does the standard protocol require a continually available facility, like the key distribution middle?

- What trust prerequisites is there? Who and what entities should be trusted to do something properly?

- What's the safeguard against malfunction? Can an outsider impersonate the entities within the standard protocol and subvert protection? Can any get together of the standard protocol cheat without diagnosis?

- How efficient may be the protocol? A standard protocol requiring several actions to determine an encryption primary that'll be used often is a very important factor; it is very another to undergo several time-consuming actions for your one-time use.

- How easy may be the protocol to put into practice? Notice that complexness in computer execution may be not the same as manual use.

2.9 Review Question

1.What features would create an encryption definitely unbreakable? What qualities would produce an encryption impractical to break up?

2. Does indeed a substitution have to be a permutation on the plaintext icons? Why or you will want to?

3. Explain whythe product of two relatively simple ciphers, like a substitution and also a transposition, can perform a high amount of security.

4. How can you quickly test a bit of ciphertext to advise whether it had been likely the consequence of a simple substitution?

IDOL

5. How can you quickly test a bit of ciphertext to advise whether it had been likely the consequence of a transposition?

6. Suggest a way to obtain a very very long sequence of unstable numbers. Your resource must be a thing that both sender and device can readily obtain but that's not clear to outsiders and isn't transmitted straight from sender to receiver.

7. Provided the quickness of an ongoing ordinary personal computer (for residence or light workplace use), estimate the quantity of time essential to break a DES encryption by tests all 256 feasible keys. Create a similar estimate to get a 128-little bit AES key.

8. Record three applications when a stream cipher will be desirable. Are programs for stop ciphers more frequent? Why or you will want to? Why do you consider this is legitimate?

9. Are DES and AES stream or block ciphers?

2.7 References

1. Security in Computing, Fourth Edition By Charles P. Pfleeger - Pfleeger Consulting Group, Shari Lawrence Pfleeger - RAND Corporation Publisher: Prentice Hall

2. Cryptography and Network Security - Principles and Practice fifth edition Stallings William Publisher: Pearson

3. Cryptography And Network Security 3rd Edition behrouz a forouzan and debdeepmukhopadhyay 3/E Publisher: McGraw Hill Education

4. Cryptography and Network Security, 3e AtulKahate Publisher: McGraw Hill

IDOL

Chapter 3. Program Security

IDOL

3.1. Secure Programs

Consider what we indicate when we say an application is "secure." We all know that security implies some extent of trust that this program enforces anticipated confidentiality, integrity, and availability. As, how could we look at a software element or code portion and determine the security? This issue is, of course, similar to the challenge of assessing software level of quality in general. A good way to determine security or quality is to inquire most people to name the functions of software that lead to its overall security. However, we're likely to get several answers from differing people. This the difference occurs since the importance of the functions is determined by who will be analyzing the software. Such as, one person may determine that code is safe because it will take too long to break simply by the security controls. And somebody else may determine code is safe if it has operated for a length of time with zero apparent failures. Nevertheless, a then person might decide that any potential fault on meeting security requirements would make code insecure.
Early on work in computer security was according to the paradigm in "penetrate and patch," inwhich analysts looked for and repaired faults. Sometimes, a top-

quality "tiger staff" would be convened to check a system's security simply by attempting to make it fail. The check was considered as being an "evidence" of security; in the even, the system withstood any attacks, it was taken into consideration sound and secure,Unfortunately, sometimes the proof evolved into a counterexample, by which not simply one but several critical security problems had been uncovered. The problem finding, in turn, resulted in a rapid work to "patch" the machine to repair or regain the security. On the other hand, the patch efforts were largely useless, producing the system less safe and sound, rather than safer because they frequently introduced fresh faults.

There are around four reasons why.
1. The pressure to fix a specific problem urged a narrow concentrate on the fault by itself and not upon its context. Particularly, the analysts have taken notice of the immediate reason for the failure and not into the underlying style or requirements faults.
2. The fault often experienced nonobvious unwanted effects in places besides the immediate section of the fault.
3. Fixing one problem frequently caused a failure elsewhere, or the patch resolved the problem in just one place, not really in other appropriate places.
4. The fault could hardly be fixed properly since system functionality or overall performance would suffer on that basis.
The insufficiencies of penetrate-and-patch led experts to seek an easier way to be confident that code fulfills its security requirements. A good way to perform that is usually to compare the requirements with the behavior. That could be, to be aware of program security, we are able to examine programs to determine whether they work as their designers expected or users required. We call such unpredicted behavior an application security flaw; it can be inappropriate program behavior the effect of a program vulnerability.
Program protection flaws can easily derive from any type of software fault. That's, they cover
everything from uncertainty of plan requirements to the one-character error in coding or maybe typing. The flaws may result from problems within a code component or through the failure of different programs or program parts to interact compatibly through a distributed interface. The security defects can reveal a code that was deliberately designed or coded to be malicious or maybe

code that was just formulated in a sloppy or misguided way. Therefore, it makes sense to split program imperfections into two individual logical groups: inadvertent human errors versus malicious, deliberately induced flaws.

Types of Flaws

To aid our understanding of the problems and their prevention or correction, we can define categories that distinguish one sort of problem from another. For example, Landwehret al. present a taxonomy of program flaws, dividing them first into intentional and inadvertent flaws. They further divide intentional flaws into malicious and no maliciousones.

In the taxonomy, the inadvertent flaws fall into six categories:
- validation error (incomplete or inconsistent): permission checks
- domain error: managed access to data
- serialization and aliasing: program flow order
- inadequate identification and authentication: basis for authorization
- boundary condition violation: failure on the first or last case
- other exploitable logic errors

3.2. NON MALICIOUS PROGRAM ERRORS

Being a human being, programmers and various other developers make many blunders, the majority of which are unintentional and nonmalicious. Various such errors trigger program malfunctions but usually do not lead to more severe security vulnerabilities. Nevertheless, a couple of classes of errors possess plagued security and programmers professionals for many years, and there is hardly any reason to believe that they will go away. In this section, we look at three typical error types which have enabled many latest security breaches. We describe each type, why it really is relevant to protection, and specifically how it can be more prevented or mitigated.

Buffer Overflows

IDOL

A buffer overflow is the computing equivalent of trying to pour two liters of water into a one-liter pitcher: Some water is likely to spill out and make a tangle. And in computing, what mess these errors have made!

Definition

A buffer (or array or string) is a space in which data can be held. A buffer resides in memory. Because memory is finite, a buffer's capacity is finite. For this reason, in many programming languages the programmer must declare the buffer's maximum size so that the compiler can set aside that amount of space.

Let us look at an example to see how buffer overflows can happen. Suppose a C language program contains the declaration:

```
char sample[10];
```

The compiler sets aside 10 bytes to store this buffer, one byte for each of the ten elements of the array, sample[0] through sample[9]. Right now we execute the statement:

```
sample[10] = 'A';
```

The subscript beyond bounds (that could be, it generally does not fall between 0 and 9), therefore we have a problem. The nicest end result (from a security point of view) is made for the compiler to identify the problem and tag the mistake during compilation. Nevertheless, if the declaration were

```
test[i] = 'A';
```

we're able to not recognize the problem untilihad been set at the time of execution to a too-big subscript. It would be beneficial if, during execution, the program produced one message warning of thesubscript away of bounds. Sadly, in some dialects, buffer sizes do not need to be predefined, so you will not

IDOL

identify an out-of-bounds error. Moreover, the code had a need to check every subscript against its probable maximum value does take time and space at the time of execution, and the assets are put on catch an issue occurring relatively infrequently. Actually, if the compiler had been careful in examining the buffer declaration and make use of, this same issue will be caused due to pointers, that there is not any reasonable way to define an effective limit. Therefore, some compilers usually do not generate the code to evaluate for exceeding bounds. Why don't we examine this issue more closely? It is essential to notice that the potential overflow causes a significant problem only in most cases. The problem'soccurrence depends upon what is next to the array test. As an illustration, suppose each of the ten components of the array test are filled with the notice A and the incorrect reference uses the notice B, the following:

```
for (i=0; i<=9; i++)
test[i] = 'A';
test[10] = 'B'
```

All system and data components are in memory space during execution, posting space with the theoperating system, various other code, and resident routines. Therefore there is certainly four cases to review in deciding the place that the 'B' will go. In case the extra personality overflows into the user's data space, it simply overwrites a preexisting variable worth (or it might be created into an as-yet emptylocation), maybe affecting the program's result, but affecting no additional data or program.

IDOL

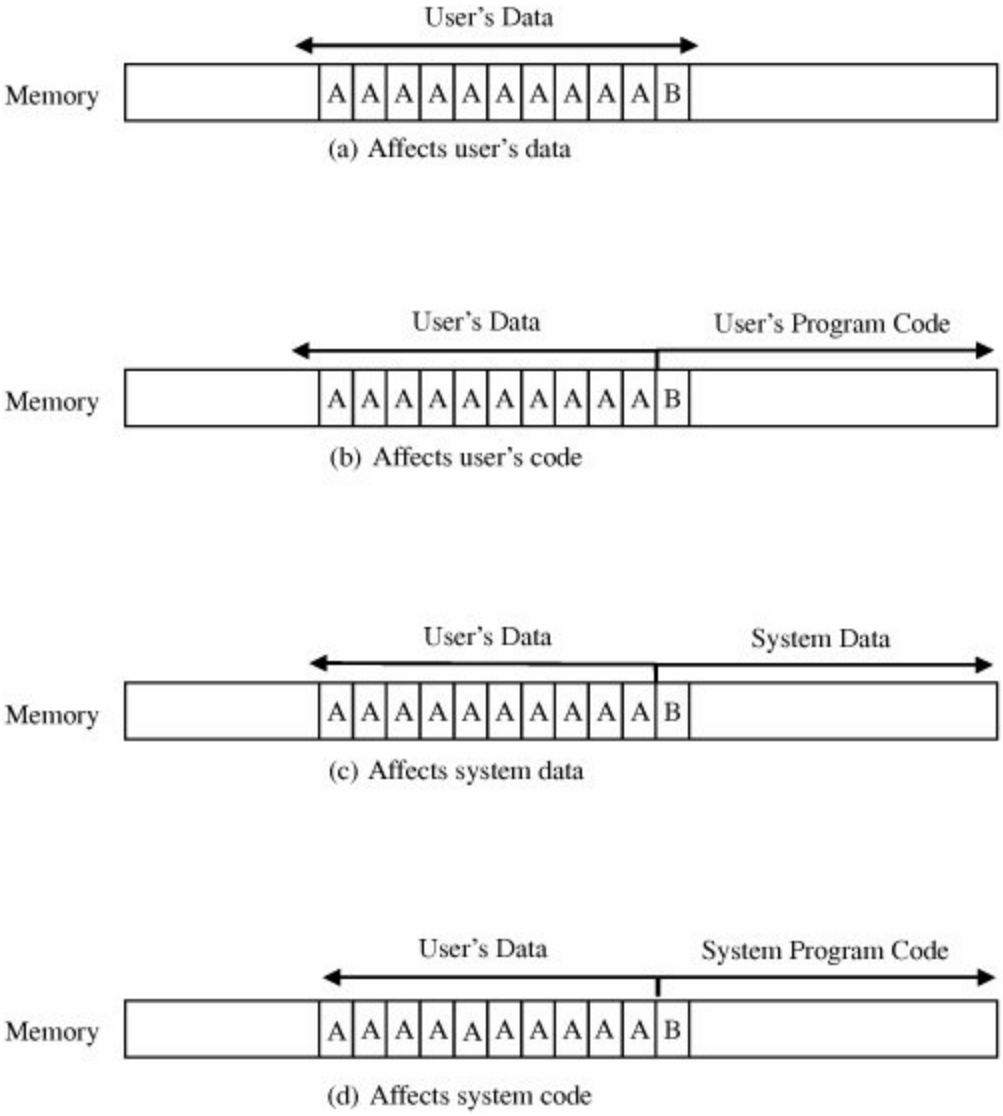


Figure 3.1. Places Where a Buffer Can Overflow.

3.3 VIRUS AND OTHER MALICIOUS CODE

On their own, programs are rarely security threats. The applications work on data, taking the action only once data and state changes result in it. Most of the function done using a program is unseen to users, so they may be not likely to understand any kind of malicious activity. For example, when was the previous time you did find a bit? Have you any idea regarding form a document is stored?

If you know a document resides somewhere on a disk, can you find it? Can you tell if a game Does the program do anything in addition to its expected interaction with you? Which files are modified by a word processor when you create a document? Most users cannot answer these questions. However, since computer data are not usually seen directly by users, malicious people can make programs serve as vehicles to access and change data and other programs. Let us look at the possible effects of malicious code and then examine in detail several kinds of programs that can be used for interception or modification of data.

Why Worry About Malicious Code?

the None of us likes the unexpected, particularly in the programs. Malicious code acts in unexpected ways, because of a malicious programmer's purpose. We think from the malicious code as lurking within our system: every or a few of a program that people are running or maybe a thenasty part of a different program that somehow links itself to a different (good) program.

.

Malicious Code Can Do Much (Harm)

Malicious code perform anything any kind of program can, for example, writing a note on a computer screen, halting a running program, producing a sound, or removing a stashed away document. Or malicious code performs nothing at all at this time; it can be placed to lie dormant, hiddenuntil some event leads to the code to behave. The trigger could be a period or particular date, an interval (for example, after thirty minutes), a meeting (for example, if a specified program is executed), a condition (to illustrate, when communication occurs over a modem), a count (one example is, the fifth time something takes place), some mixture of these types of, or a random scenario. In fact, malicious code perform different things every time, or nothing usually with something Malicious code runs beneath the user's authority. Hence, malicious code can affect everything the user can feel, and in a similar way. Users routinely have comprehensive control over their particular program code and documents; they are able to read, write, change, append, as well as delete them. And well, they should. But malicious code performs the same, devoid of the user's permission or even knowledge.

Malicious Code Has Been around a Long Time

The popular literature and press continue to highlight the effects of malicious code as if it were a relatively recent phenomenon. It is not. Cohen [COH84] is sometimes credited with the discovery of viruses, but in fact, Cohen gave a name to a phenomenon known long before.

For example, Thompson, in his 1984 Turing Award lecture, "Reflections on Trusting Trust", described code that can be passed by a compiler. In that lecture, he refers to an earlier Air Force document, the Multics security analysis. In fact, references to virus behavior go back at least to 1970. Ware's 1970 study (publicly released in 1979 and Anderson's planning study for the U.S. Air flow Force (to which Schell also refers) still accurately describe threats, vulnerabilities, and system security flaws, especially intentional ones. What is new about malicious code is the number of distinct instances and copies that have appeared. So malicious code is still around, and its effects are more pervasive. It is important for us to learn what it looks like and how it works, so that we can take steps to prevent it from doing damage or at least mediate its effects. How can malicious code take control of a system? How can it lodge in a system? How does malicious code spread? How can it be recognized? How can it be detected? How can it be stopped? How can it be prevented? We address these questions in the following sections.

Kinds of Malicious Code

Malicious code or a rogue program is the general name for unanticipated or undesired effects in programs or program parts, caused by an agent intent on the damage. This definition eliminates unintentional errors, although they can also have a serious negative effect. This definition also excludes coincidence, in which two benign programs combine for a negative dramatic on occasion. In general, malicious code can act with all the predictability of a two-year-old child: We know in general what two-year-olds do, we may even know what a specific two-year-

old often do in certain situations, but two-year-olds have an amazing capacity to do the unexpected effect.

The operator is the author of the program or the individual who causes its dispersion. By this definition, most defect found in software examinations, reviews, and testing don't qualify as malevolent code, since we consider them as unexpected. Be that as it may, remember as you peruse this section accidental issues can in certainty summon indistinguishable reactions from purposeful malignancy; a kind reason can, in any case, lead to a disastrous effect.

You are probably going to have been influenced by a virus at some time, either your computer was contaminated by one or on the grounds that you couldn't get to an accessan infected system.

A Virus is a program that can pass on the malicious code to different nonmalicious programs by adjusting them.

The word "Virus" was authored in light of the fact that the infected program acts like a biological virus: It contaminates other healthy file by appending itself to the program and either decimating it or existing together with it. Since infections are treacherous, we can't accept that a clean program yesterday is still perfect today. In addition, a good program can be changed to incorporate a duplicate file of the infection program, so the tainted good program itself starts to go about as an infection contaminating other programs. The infection generally spreads at a geometric rate, inevitably overwhelming a whole figuring computer system and spreading to all other associated computer system.

A Virus can be either transient or inhabitant. A transient Virus has a real existence that relies upon life of its host; the infection runs when its connected program executes and ends when its appended program closes. (Amid its execution, the transient virus may have spread its contamination to different program.) A resident virus finds itself in memory; at that point it can remain dynamic or be enacted as an independent program, even after its joined program closes.

A Trojan horse is harmful code that, notwithstanding its essential impact, has a second, nonobvious malignant effect for instance of a PC Trojan horse,

A logical bomb is a class of destructive code that "explodes" or goes off when a predefined condition happens. A period bomb is a logical bomb whose trigger is a period or date.

A trapdoor or back door is an element in a program by which somebody can get to the program other than by the self-evident, direct call, maybe with uncommon benefits. For example, an automated bank employee program may permit anybody entering the number 990099 on the keypad to process the log of everybody's exchanges at that machine. In this model, the trapdoor could be deliberate, for support purposes, or it could be an illegal route for the implementer to crash any record of wrongdoing.

A worm is a program that spreads duplicates of itself through a system. The essential distinction between a worm and a virus is that a worm spread through systems, and a virus can spread through any medium (yet for the most part, uses duplicated program or information documents). Moreover, the worm spreads duplicates of itself as an independent program, while the virus spreads duplicates of itself as a program that joins to or installs in different programs.

White et al. additionally characterize the rabbit as a virus or worm that self-duplicates without bound, with the goal of depleting some computing resource. For instance,a rabbit may make duplicates of itself and, store them on the hard disk, with an end goal to totally fill the hard disk.

The word "Virus" is frequently used to allude to any bit of malevolent code. Besides, at least two types of destructive code can be consolidated to deliver a third sort of issue. For example, a virus can be a period bomb if the viral code that is spreading will trigger an occasion after a timeframe has passed.

The sorts of malevolent code are condensed in Table 3-1

IDOL

TABLE 3-1 Types of Malicious Code

Code	Characteristics
Virus	Attaches itself to program and propagates copies of itself to other programs
Trojan horse	Contains unexpected, additional functionality
Logic bomb	Triggers action when the condition occurs
Time bomb	Triggers action when the specified time occurs
Trapdoor	Allows unauthorized access to functionality
Worm	Propagates copies of itself through a network
Rabbit	Replicates itself without limit to exhaust resources

Because "virus" is that the common name given to any or all types of malicious code and since fuzzylines exist between completely different forms of malicious code, we'll not be too restrictive within thefollowing discussion. we wish to appear at however malicious code spreads, however it's activated, and what result it will have. a pestilence may be a convenient term for mobile malicious code, and so inthe following sections, we have a tendency to use the term "virus" nearly completely. The points created applyalso to different types of malicious code.

How Viruses Attach

A printed duplicate of a virus does nothing and threatens no one. Even executable virus code sitting on a disk does nothing. What triggers a virus to start replicating? For a virus to do its malicious work and spread itself, it must be activated by being executed. Fortunately for virus writers but unfortunately for the rest of us, there are many ways to ensure that programs will be executed on a running computer.

For example, recall the SETUP program that you initiate on your computer. It may call dozens or hundreds of other programs, some on the distribution medium, some already residing on the computer, some in memory. If any one of these programs contains a virus, the virus code could be activated. Let us see how. Suppose the virus code were in a program on the distribution medium, such as a CD; when executed, the virus could install itself on a permanent storage medium

(typically, , a hard disk) and also in any and all executing programs in memory. Human intervention is necessary to begin the process; a human being puts the virus on the distribution medium, and perhaps another initiates the execution of the program to which the virus is certainly attached. (It will be possible for execution to happen without human intervention, though such as once execution is certainly triggered by using a date or the passing of some period of time.) After that, no human involvement is; the virus can propagate by itself.

A more common way of virus activation is as an attachment to an e-mail message. In this assault, the virus writer tries to convince the victim (the recipient of the e-mail message) to open the attachment. Once the viral attachment is definitely opened, the activated virus can do its work. Some modern e-mail handlers, in a get, to "help" the receiver (victim), automatically open attachments as soon as the receiver opens the body of the e-mail message. The virus can be executable code embedded in an executable attachment, but other types of files are equally dangerous. For example, objects such as graphics or photo images can contain code to be executed by an editor, so they can be transmission agents for viruses. In general, it is safer to force users to open documents on their own instead of automatically; it is a bad idea for programs to perform potentially security-relevant actions without a user's consent. However, ease-of-use often trumps security, so programs such as browsers, e-mail handlers, and viewers often "helpfully" open files without asking the user first.

Appended Viruses

A program virus attaches itself to the program; then, whenever this program is performing, the virus is triggered. This type of attachment is generally uncomplicated to program.

In the simplest circumstance, a virus inserts a replica of itself into the executable program document prior to the first executable instruction. After that, all of the virus instructions perform first; following the last virus instruction, control flows normally to what accustomed to end up being the first plan instruction. Such a scenario is proven in Figure 3.4.

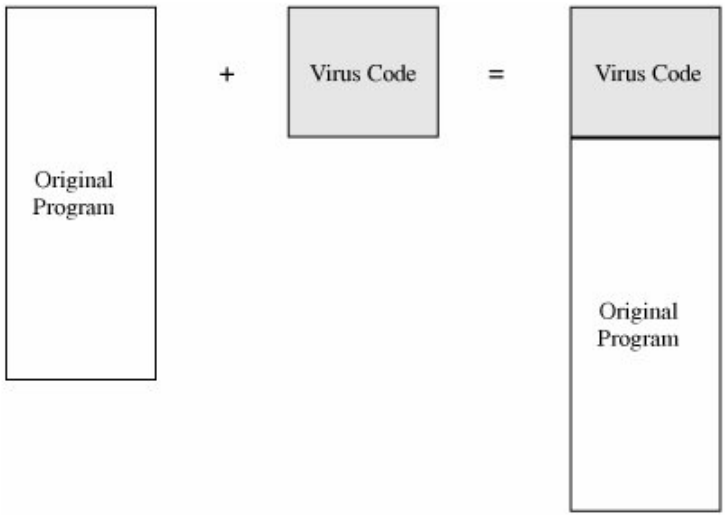


Figure 3.4. Virus Appended to a Program.

These types of attachment are simple and generally effective. The virus article writer does not know anything about this program to which the virus will certainly attach, and sometimes the attached program simply acts as a carrier pertaining to the virus. The virus functions its task after which transfers to the initial program. Typically, an individual is unaware of the consequence of the virus if the original program still does all that it used to. Most viruses attach in this manner.

An alternative to the attachment may be a virus that runs the first program however has management before and once its execution.For example, a virus writer may want to prevent detection of the virus. If the virus is stored on disk, its presence will be indicated by its file name or its size will affect the amount of space used on the disk. The virus writer can arrange for the virus to attach to the program that builds the list of files on the disk. If the virus regains control after the listing program generates the list, but before the list is displayed or printed, the virus could eliminate its entry from the list and falsify the number of spaces so that it does not appear not to exist. A surrounding virus is shown in Figure 3.5.

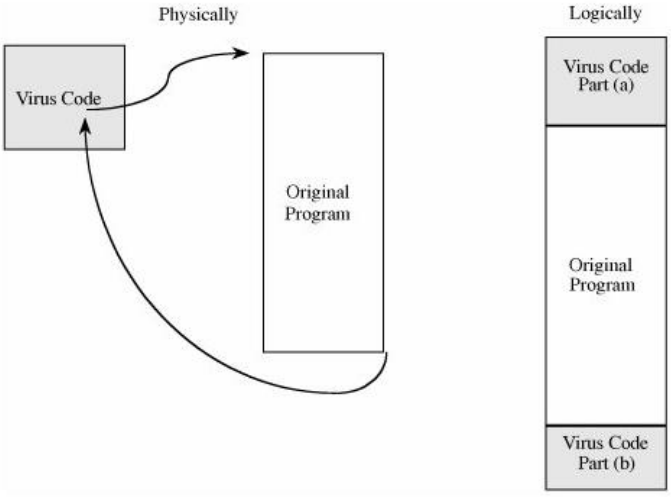


Figure 3-5. Virus Surrounding a Program.

A third circumstance happens when the infection replaces a portion of its target, incorporating itself into the original code of the target. Such a circumstance has appeared in Figure 3.6. Obviously, the infection author needs to know the accurate structure of the first program to realize where to embed which bits of a piece of code in the virus.

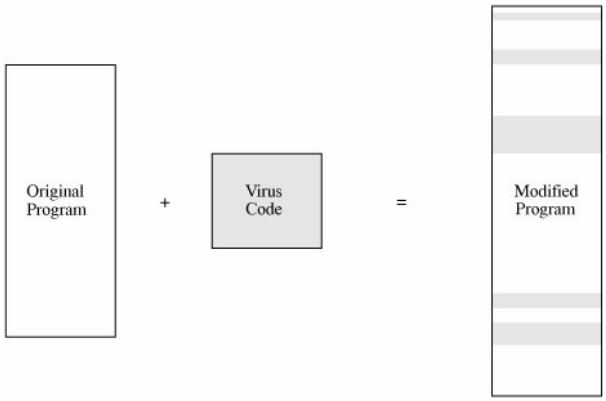


Figure 3.6. Virus Integrated into a Program.

At last, the virus can supplant the whole target, either mirroring the impact of the target or disregarding the normal impact of the target and performing just the Virus impact. For this situation, the client is destined to see the loss of the original program.

Document Viruses

presently, the most prominent virus type is the thing that we call the Document virus, which is executed inside a well-organized document, for example, written document, a database, a slide presentation, an image, or a spreadsheet. These documents are exceptionally organized records that contain the two data items (words or numbers) and command, (for example formulas, formatting controls, links). The commands are a piece of a rich programming language, including macros, variables and methods, file access to, and even systems calls. The author of a document virus utilizes any of the highlights of the programming language to perform malignant activities.

The standard client, as a rule, sees just the substance of the document (it's content or data), so the virus writer essentially incorporates the virus in the piece of the command of the document, as in the built-in program virus.

How Viruses Gain Control

The virus (V) must be summoned rather than the Target (T). Basically, the virus either needs to appear to be T, saying adequately "I am T" or the virus needs to drive T off the beaten path and become a substitute for T, saying successfully "Call me rather than T." An increasingly barefaced virus can essentially say "conjure me [you fool]."

The virus can accept that T's name by supplanting (or joining to) T's code in a document structure; this conjuring strategy is most fitting for the conventional program. The virus can overwrite T away (basically supplanting the duplicate of T away, for instance). On the other hand, the virus can change the pointers in the record table so the virus is situated rather than T at whatever point T is gotten to

IDOL

through the file system. These two cases have appeared in Figure 3.7.

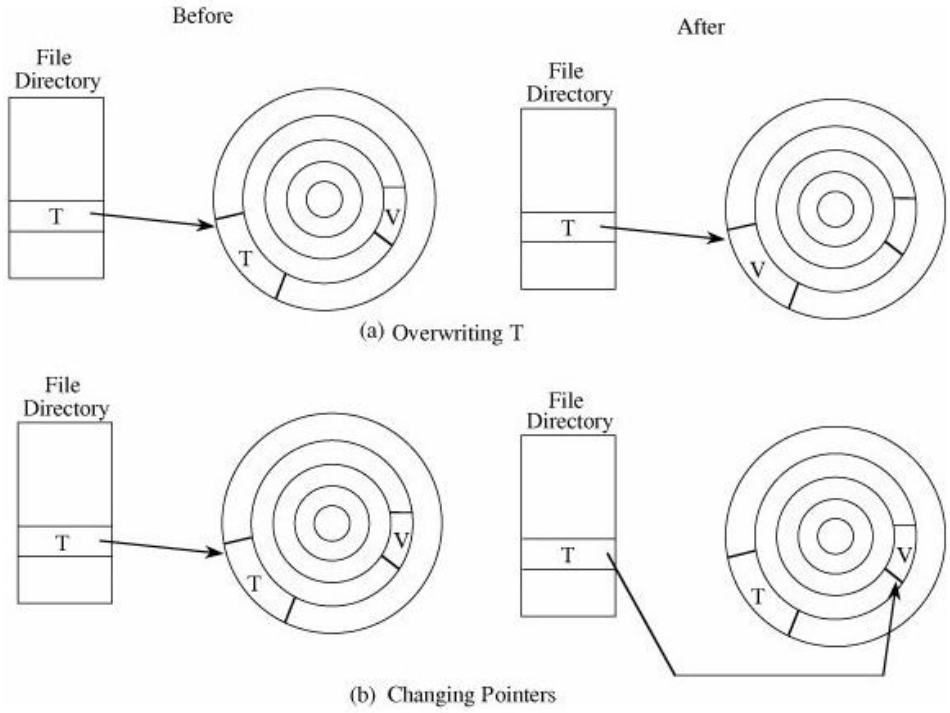


Figure 3.7. Virus Completely Replacing a Program.

The virus can supersede T by changing the succession that would have invoked T to now summon the Virus V; this summons can be utilized to supplant portions of the resident operating system by altering pointers to those inhabitant parts, for example, the table of handlers for various types of interrupts.

Homes for Viruses

The virus writer may discover these characteristics engaging in a virus:

It is difficult to identify.

It isn't effectively deleted or deactivated.

It spreads contamination broadly.

It can reinfect its home program or a different program.

It is anything but difficult to make.

It is a machine free and operating system independent.

Maybe a couple of viruses meet every one of these criteria. The virus writer looks over these goals when choosing what the Virus will do and where it will dwell.

Only a couple of years prior, the test for the virus writer was to compose code that would be executed more than once with the goal that the virus could duplicate. Presently, be that as it may, one execution is sufficient to guarantee across the board distribution. Numerous Viruses is transmitted by email, utilizing both of two courses. In the principal case, some virus writer creates another email message to all locations in the victim's individual's address book. These new messages contain a duplicate of the virus with the goal that it engenders generally. Frequently the message is a concise, effusive, nonspecific message that would urge the new recipient to open the connection from a companion (the principal recipient). For instance, the title or message body may peruse "I figured you may appreciate this image from our vacation." In the second case, the Virus writer can leave the tainted document for the unfortunate casualty to advance unwittingly. On the off chance that the virus's impact isn't quickly self-evident, the Victim may pass the tainted record accidentally to different unfortunate victims.

Give us a chance to look all the more carefully at the issue of viral residence.

One-Time Execution

Most of the virus today execute just once, spreading their infection and causing their impact in that one execution. A virus regularly lands as an email connection of a document virus. It is executed just by being opened.

Boot Sector Viruses

A unique instance of virus attachment, however once a genuinely prominent one, is the alleged boot sector virus. At the point when a PC is begun, control starts with firmware that figures out which equipment components are available, tests them, and exchanges control to an operating system. A given hardware platform can run a wide range of the operating system, so the operating system isn't coded in firmware however is rather invoke powerfully, maybe even by a client's decision, after the equipment test.

The operating system is programming stored on the hard disk. Code duplicates the operating system from hard disk to memory and exchanges control to it; this replicating is known as the bootstrap (frequently boot) load in light of the fact

that the operating system metaphorically maneuvers itself into memory by its bootstraps. The firmware does its control exchange by perusing a fixed number of bytes from a fixed area on the hard disk (called the boot sector) to a fixed location in memory and afterward hoping to that address (which will end up containing the first instruction of the bootstrap loader). The bootstrap loader at that point peruses into memory the remainder of the operating system from hard disk. To run an alternate operating system, the client just embeds a hard disk with the new operating system and a bootstrap loader. At the point when the client reboots from this new hard disk, the loader there gets and runs another operating system. This equivalent plan is utilized for PCs, workstations, and enormous centralized computers.

To take into consideration change, development, and vulnerability, hardware designers save a lot of room for the bootstrap loader. The boot area on a PC is somewhat under 512 bytes, however since the loader will be bigger than that, the equipment originators block "chaining," in which each block of the bootstrap is affixed to (contains the disk area of) the following blocks. This chaining permits enormous bootstraps yet additionally streamlines the establishment of a virus. The Virus writer just breaks the chain anytime, embeds a pointer to the virus code to be executed, and reconnects the chain after the infection has been introduced. This circumstance has appeared in Figure 3.8.

Understanding the Resident virus

Viruses are a colossal danger to anybody with an association with the web. These frightful programs commonly introduce and execute themselves without the unfortunate victim's knowledge. The effect of a viruses runs generally from hindering the exhibition of your PC to totally deleting the majority of your significant records. By and large, it will disperse itself to different machines you speak with, enabling it to impact on a whole network. Notwithstanding how extreme the result, a virus is something you don't need on your PC.

What is a memory Resident virus?

A memory Resident virus is a standout amongst the most widely recognized sorts of PC infection. It works by introducing malevolent code into the memory of your PC, tainting current program and any others you may introduce later on. So as to accomplish this, the Resident virus needs to discover a technique to dispense

memory for itself, which means it must discover some place to stow away. Also, it must build up a procedure that initiates the inhabitant code to start tainting different records.

A Resident virus may utilize various procedures to spread its a disease. A standout amongst the most disregarded strategies includes the TSR (Terminate-Stay-Resident) intrude on capacity. While this technique is the most effortless to summon disease, it is likewise effectively distinguished by an infection scanner. An increasingly wanted system includes the control of MBCs (memory control blocks). Finally, a virus needs to append itself to explicit hinders so as to dispatch the occupant code. For example, if a virus is modified to enact each time a program is run, it must be snared to interfere with capacities assigned for stacking and executing that specific application.

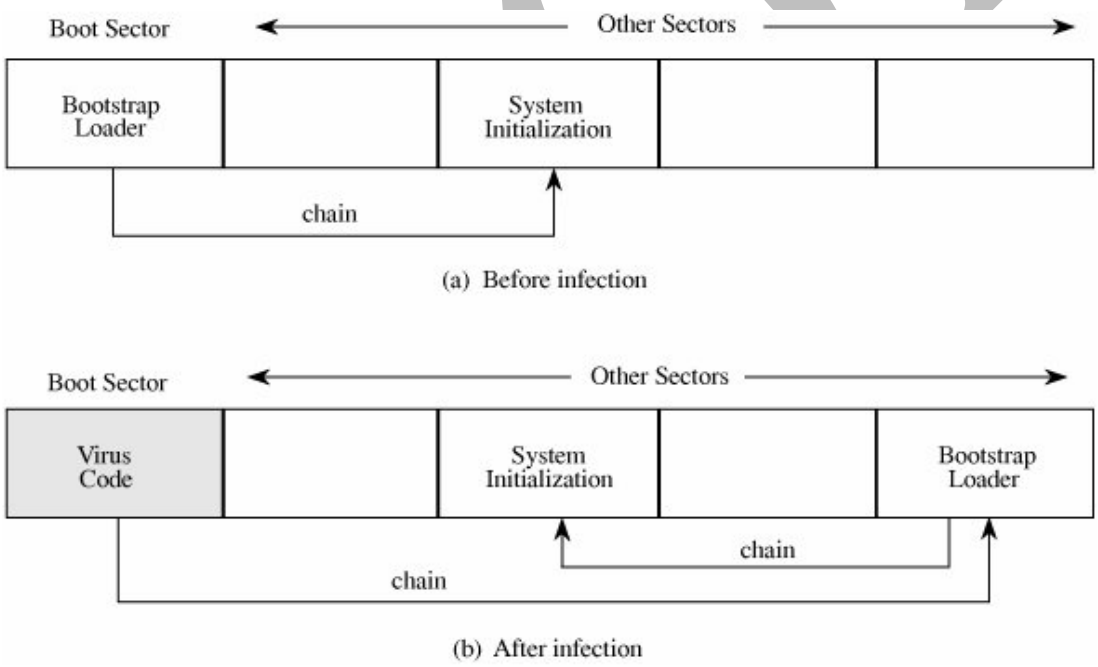


Figure 3-8. Boot Sector Virus Relocating Code.

Other Homes for Viruses

A virus that does not relocate to one of these comfortable foundations needs to fight more for itself. In any case, saying this doesn't imply that that the virus will go destitute.

One famous home for a virus is an application program. Numerous applications, for example, word processors and spreadsheets, have a “macro " include, by which a client can record a progression of commands and rehash them with one invocation. Such program additionally gives a "startup macro" that is executed each time the application is executed. A virus writer can make a virus large scale that adds itself to the startup orders for the application. It likewise then implants a duplicate of itself in documents with the goal that the infection spreads to anybody getting at least one of those records.

Libraries are additionally great spots for vindictive code to dwell. Since libraries are utilized by numerous projects, the code in them will have a wide impact. Furthermore, libraries are frequently shared among clients and transmitted starting with one client then onto the next, a training that spreads the contamination. At long last, executing code in a library can pass on the viral infection to other transmission media. Compilers, loaders, linkers, runtime screens, runtime debuggers, and even virus control projects are a great possibility for facilitating infections since they are generally shared.

Virus Signatures

A virus signature is the unique finger impression of a virus. It is a lot of interesting information, or bits of code, that enable it to be distinguished. Antivirus programming utilizes a virus signature to discover a virus in PC file systems, permitting to distinguish, isolate, and evacuate the virus. In the antivirus programming, the virus signature is alluded to as a definition record or DAT document.

The different virus may have a similar virus signature, which permits the antivirus program to identify numerous virus when searching for a solitary virus signature. Due to this sharing of a similar virus signature between different virus, the antivirus program can in some cases recognize a virus that isn't known yet. New infections have a virus signature that is not utilized by a different virus, however

new "strains" of realized virus some of the time utilize a similar virus signature as prior strains.

Antivirus software performs visit virus signature, or definition, updates. These updates are vital for the product to identify and expel a new virus. The new virus is being made and discharged practically day by day, which powers antivirus software to need continuous updates.

```
/default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
%u9090%u6858%ucbd3
%u7801%u9090%u6858%ucdb3%u7801%u9090%u6858
%ucbd3%u7801%u9090
%u9090%u8190%u00c3%u0003%ub00%u531b%u53ff
%u0078%u0000%u00=a
HTTP/1.0
```

Storage Patterns

Most viruses connect to programs that are put away on media, for example, disk. The connected virus piece is invariant, so the beginning of the virus code turns into a recognizable signature. The connected piece is constantly situated in a similar position with respect to its joined document. For instance, the virus may dependably be toward the starting, 400 bytes from the top, or at the base of the contaminated file. In all probability, the virus will be toward the start of the record in light of the fact that the virus writer needs to acquire control of execution before the true code of the infected program is in control. In the most straightforward case, the virus code sits at the highest point of the program, and the whole virus performs its vindictive responsibility before the ordinary code is infected. In different cases, the virus infection comprises of just a bunch of guidelines that point or hop to other, progressively nitty gritty directions somewhere else. For instance, the infection code may comprise of condition testing and a hop or call to a different infection module. In either case, the code to which control is exchanged will likewise have a conspicuous example. Both of these circumstances have appeared in Figure 3.9.

IDOL

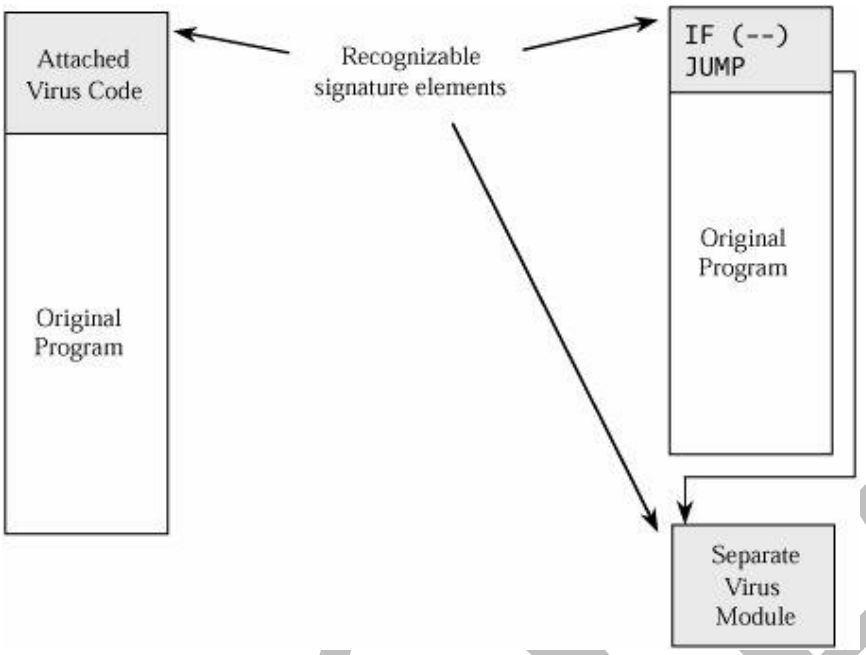


Figure 3.9. Recognizable Patterns in Viruses.

A virus may join itself to a document, in which case the record's size increases. Or on the other hand, the virus may destroy all or part of the basic program, where case the program's size does not change but rather the program's working will be debilitated. The virus writer needs to pick one of these distinguishable impacts.

The virus scanner can utilize a code or checksum to identify changes to a document. It can likewise search for suspicious examples, for example, a JUMP instruction as the initial instruction of a system program (in the event that the virus has situated itself at the base of the record yet is to be executed first, as in Figure 3.9).

Execution Patterns

A virus copywriter may want a virus to complete several things from the same time, such as spread virus, avoid recognition, and cause harm. These types of goals are shown within Table 3.2, along using ways each goal can easily be addressed. Unfortunately, numerous of these behaviors are usually perfectly normal and may otherwise go undetected. Regarding instance, one goal will be modifying the file type; many normal programs create files, delete files, and even

write to storage mass media. Thus, no key indicators point to the occurrence of a virus.

Table 3-2. Virus Effects and Causes.

Virus Effect	How It Is Caused
Attach to an executable program	• Modify file directory • Write to the executable program file
Attach to data or control file	• Modify directory • Rewrite data • Append to data • Append data to self
Remain in memory	• Intercept interrupt by modifying interrupt handler address table • Load self in the nontransient memory area
Infect disks	• Intercept interrupt • Intercept operating system call (to format disk, for example) • Modify system file • Modify the ordinary executable program
Conceal self	• Intercept system calls that would reveal the self and falsify the result • Classify self as "hidden" file
Spread infection	• Infect boot sector • Infect systems program • Infect ordinary program • Infect data ordinary program reads to control its execution

IDOL

Prevent deactivation	• Activate before deactivating program and block deactivation • Store copy to reinfect after deactivation

Transmission Patterns

A virus works well only if it offers some way of transmission by one location to a different. Because we have already viewed, viruses can travel throughout the boot process by simply attaching to an executable file or perhaps traveling within data. The travel itself arises during the execution of the already infected program. Considering that a virus can implement any instructions a course could, virus travel is not really restricted to any single method or execution pattern. With regard to instance, a virus may arrive on the disk or even from a network, travel during its host's execution to a hard disk drive start sector, reemerge the next occasion the particular host computer is booted, and remain in recollection to infect other drives as they are reached.

Polymorphic Viruses

virus signature may be typically the most reliable means for some sort of virus scanner to acquire a virus. If a new particular virus always commences with the string 47F0F00E08 (in hexadecimal) and contains line 00113FFF located at term 12, it is improbable that other programs or perhaps data files will include these exact characteristics. Regarding longer signatures, the likelihood of a correct fit increases.

If the disease scanner will always seem for those strings, next the clever virus copywriter can cause something some other than those strings in order to be in those opportunities. Many instructions cause simply no effect, such as including 0 to a quantity, comparing many to on its own, or jumping to the particular next instruction. These recommendations, sometimes called no-ops, may be sprinkled into an item of code to pose any pattern. For illustration, the virus could include two alternative but comparable beginning words; after becoming installed, the virus may choose one with the couple of words for its first word. Then, a

computer virus scanner would have in order to look for both styles. A virus which could alter its appearance is referred to as some sort of polymorphic virus. (Poly implies "many" and morph means that "form. ")

A two-form polymorphic virus may be taken care of easily as two self-employed viruses. Therefore, herpes article writer intent on preventing the diagnosis of the virus will need either a large or perhaps a limitless number of varieties so that the amount of possible forms is as well large for a disease scanner to find. Simply sneaking in a random number or perhaps string with a fixed location in the executable edition of a virus is not really sufficient, because the trademark with the virus is merely the constant code removing from the total the random part. A new polymorphic virus needs to aimlessly reposition all parts regarding itself and randomly modify all fixed data. As a result, instead of containing typically the fixed (and therefore searchable) string "HA! INFECTED BY SIMPLY A VIRUS, " a new polymorphic virus has in order to change even that style sometimes.

Trivially, assume some sort of virus writer has a hundred bytes of code plus 50 bytes of data. To make two malware instances different, the article writer might distribute the initial version as 100 octets of code followed by simply all 50 bytes regarding data. A second type could possibly be 99 bytes associated with the code, a jump teaching, 50 bytes of info, and the last octet of code. Other types are 98 code octet jumping to the second option, 97 and three, and etc .. Just by moving items around, the virus copywriter can create enough diverse appearances to fool easy virus scanners. As soon as the reader-writers became aware of these kinds of tricks, on the other hand, they refined their personal definitions.

A simple selection of polymorphic virus utilizes encryption under various takes some time to make the saved kind of the virus various. These are sometimes known as encrypting viruses. This kind of computer virus must contain three distinctive parts: a decryption key element, the (encrypted) object computer code of the virus, plus the (unencrypted) object code with the decryption routine. For these kinds of viruses, the decryption program itself, or a phone to a decryption collection routine, must be inside the clear so that will become the signature.

To stay away from detection, it's not almost all copy of a polymorphic virus has to change from every other backup. If the virus modifications occasionally, not every backup will match a person of every other duplicate.

The Source of Viruses

Considering that a virus can get rather small, its signal could be "hidden" inside additional larger and more complex programs. 200 lines involving a virus might be divided into one hundred bouts of two lines regarding code and a leap each; these one 100 packets may be easily concealed inside a compiler, the database manager, data administrator, or some other big utility.

Virus discovery may be aided by a treatment to find out if two applications are equivalent. However, assumptive brings about computing are extremely discouraging with regards to the difficulty of the equivalence trouble. The general question "Are these two programs equal? " is undecidable (although that question can become answered for several specific twos of programs). Even overlooking the general undecidability trouble, two modules may generate subtly different results that will may or may not be safety-relevant. One may boost your speed, or typically the first may use some sort of brief file for work area whereas the other performs just about all its computations in recollection. These differences could get benign, or they may be a new marker of disease. Therefore, we are not likely to formulate a screening system which could separate infected quests from uninfected ones.

Even though the general is bitter, the particular is not really. When we know that a new particular virus may invade a computing system, all of us can check for this and detect it when it is there. Having found herpes simplex virus, however, we are kept with the work of cleaning the system of the computer. Taking away the virus in a new running system requires staying able to detect in addition to eliminate its instances quicker than it can be distributed.

Prevention of Virus Infection

The handiest manner to prevent the infection of a virus isn't always to acquire the executable code from an infected source. This philosophy was clean to comply with because it changed into easy to tell if a file becomes executable or not. For

example, on PCs, the a .Exe extension became a clear signal that the file was executable. However, as we've got cited, brand new files are greater complex, and an apparently nonexecutable report can also have a few executable codes buried deep within it. For example, a word processor may additionally have commands within the record report; as we cited earlier, those instructions, called macros, make it easy for the user to do complex or repetitive matters. But they're absolutely executable code embedded in the context of the record. Similarly, spreadsheets, presentation slides, other workplaces- or commercial enterprise-associated documents, or even media documents can include code or scripts that may be accomplished in various ways and thereby harbor viruses. And, as we've seen, the packages that run or use those files can also attempt to be beneficial by way of robotically invoking the executable code, whether you need it run or no longer! Against the principles of exact protection, email handlers can be set to automatically open (without appearing access manipulate) attachments or embedded code for the recipient, so your electronic mail message may have animated bears dancing across the pinnacle.

Another approach virus writers have used is a bit-acknowledged feature within the Microsoft document design. Although a file with a. Document extension is anticipated to be a Word report, in reality, the genuine file type is hidden indiscipline on the begin of the record. This convenience ostensibly allows a person who inadvertently names a Word document with a.PPT(Power-Point) or another extension. In a few cases, the working gadget will try to open the associated application but, if that fails, the device will switch to the application of the hidden record type. So, the virus writer creates an executable file, names it with an inappropriate extension, and sends it to the victim, describing it's far as an image or a necessary code upload-in or something else appropriate. The unwitting recipient opens the document and, without proceeding to, executes the malicious code.

More recently, the executable code has been hidden in files containing massive information units, including images or study-only files. These bits of viral code aren't easily detected by using virus scanners and surely no longer by means of the human eye. For instance, a document containing a photograph can be quite granular; if each sixteenth bit is part of a command string that may be performed, then the virus may be very hard to hit upon.

Because you cannot usually understand which assets are inflamed, you must anticipate that any out of doors source is infected. Fortunately, when you are receiving a code from an out of doors source; alas, it isn't viable to cut off all contact with the outdoor world.

In their interesting paper evaluating laptop virus transmission with human disorder transmission,.examine that individuals' efforts to preserve their computers unfastened from viruses lead to communities which might be typically loose from viruses because contributors of the network have little (electronic) touch with the out of doors international. In this situation, transmission is contained not because of restrained contact but due to restricted touch out of doors the community. Governments, for an army or diplomatic secrets and techniques, regularly run disconnected community groups. The trick seems to be in deciding on one's community prudently. However, as use of the Internet and the World Wide Web will increase, such separation is almost not possible to keep.

Nevertheless, there are several strategies for constructing a reasonably safe network for digital contacts, such as the following:

Use the best industrial software program received from dependable, properly-set up companies. There is always a threat that you may get hold of an epidemic from a huge producer with a call all of us would apprehend. However, such companies have good sized reputations that could be critically damaged by means of even one terrible incident, in order that they go to a few diplomae of trouble to maintain their merchandise virus-loose and to patch any trouble-causing code proper away. Similarly, software program distribution groups could be careful about the products they deal with.

Test all new software program on a remote pc. If you must use software from a questionable supply, check the software first on a computer that isn't related to a network and contains no sensitive or important facts. Run the software and search for unexpected behavior, even easy behavior which includes unexplained figures on the display. Test the pc with a copy of an up to date virus scanner created earlier than the suspect software is run. Only if the program passes these assessments should you put in it on a less isolated device?

Open attachments handiest while you realize them to be safe. What constitutes "safe" is up to you, as you have got in all likelihood already found out on this

chapter. Certainly, an attachment from an unknown source is of questionable safety. You can also mistrust an attachment from a known source but with a weird message.

Make a recoverable gadget image and shop it safely. If your gadget does end up inflamed, this smooth version will permit you to reboot securely as it overwrites the corrupted device documents with smooth copies. For this motive, you need to hold the photograph write-covered at some stage in the reboot. Prepare this photo now, earlier than contamination; after contamination, it's miles too late. For protection, prepare a further copy of the secure boot photograph.

Make and keep backup copies of executable machine files. This way, in the event of deadly disease contamination, you could take away inflamed documents and reinstall from the easy backup copies (stored in a secure, offline location, of direction). Also, make and keep backups of crucial data files that would incorporate infectable code; such documents consist of word-processor documents, spreadsheets, slide shows, snapshots, sound documents, and databases. Keep those backups on cheaper media, including CDs or DVDs so that you can keep antique backups for a long time. In case you locate contamination, you want to be able to begin from a smooth backup that is one taken earlier than the infection.

Use virus detectors (often known as virus scanners) often and update them every day. Many of the available virus detectors can each discover and get rid of the infection from viruses. Several scanners are higher than one because one might also stumble on the viruses that others miss. Because scanners search for virus signatures, they are constantly being revised as new viruses are observed. New virus signature documents or new variations of scanners are allotted regularly; often, you may request automatic downloads from the seller's net website. Keep your detector's signature report up to date.

Truths and Misconceptions About Viruses

Because viruses often have a dramatic impact on the laptop-using network, they're regularly highlighted within the press, mainly in the business section. However, there's a good deal incorrect information in stream approximately viruses. Let us observe some of the famous claims approximately them.

Viruses can infect the simplest Microsoft Windows structures. False. Among students and office employees, PCs walking Windows are popular computers, and there may be greater people writing software program (and viruses) for them than for every other form of processor. Thus, the PC is maximum often the goal whilst a person makes a decision to put in writing an epidemic. However, the standards of virus attachment and contamination follow similarly to different processors, including Macintosh computer systems, Unix and Linux workstations, and mainframe computers. Cell telephones and PDAs are now also virus targets. In fact, no writeable stored-software laptop is resistant to a feasible virus assault. As we cited in Chapter 1, this case method that each one gadget containing computer code, together with vehicles, airplanes, microwave ovens, radios, televisions, vote casting machines, and radiation therapy machines have the capacity for being inflamed by a plague.

Viruses can regulate "hidden" or "read-handiest" files. True. We may additionally try to guard files by using the use of two running gadget mechanisms. First, we are able to make a file a hidden file in order that a person or software list all documents on a storage device will no longer see the file's call. Second, we will practice a study-only safety to the file in order that the person can't exchange the document's contents. However, each of those protections is applied by means of software program, and virus software can override the local software program's protection. Moreover, software program safety is layered, with the operating machine supplying the most primary protection. If a secure running machine obtains manipulate before a virus contaminator has performed, the operating device can save you infection as long as it blocks the assaults the virus will make.

Viruses can seem handiest in facts files, or only in Word files, or most effective in applications. False. What are the data? What is an executable record? The distinction between those two standards isn't always continually clean, because a data document can manage how software executes and even purpose software to execute. Sometimes a statistics file lists steps to be taken with the aid of the program that reads the statistics, and these steps can encompass executing an application. For example, a few programs contain a configuration report whose statistics are exactly such steps. Similarly, phrase-processing record documents might also comprise startup commands to execute whilst the record is opened; those startup commands can include malicious code. Although strictly speaking, a

virus can spark off and unfold handiest whilst a program executes, in truth, facts documents are acted on by way of applications. Clever virus writers were capable of making statistics manipulate documents that motive programs to do many things, inclusive of pass along copies of the virus to different information files.

Viruses unfold simplest on disks or only thru e-mail. False. File-sharing is often executed as one consumer presents a replica of a report to any other consumer by using writing the file on a portable disk. However, any approach of digital record switch will work. A document can be located in a network's library or posted on a bulletin board. It can be attached to an e-mail message or made to be had for download from an internet website online. Any mechanism for sharing files of packages, data, documents, and so forth can be used to transfer a virulent disease.

Viruses can not continue to be in reminiscence after an entire strength off/energy on reboot. True, however . . . If a deadly disease is resident in reminiscence, the virus is misplaced when the reminiscence loses strength. That is, computer memory (RAM) is volatile, so all contents are deleted while strength is misplaced. However, viruses are written to disk definitely can remain through a reboot cycle. Thus, you may acquire a plague infection, the virus may be written to disk (or to network storage), you could flip the gadget off and return on, and the virus may be reactivated for the duration of the reboot. Boot area viruses benefit manipulate when a device reboots (whether it's for a hardware or software reboot), so a boot zone virus may also continue to be via a reboot cycle because it activates right away when a reboot has completed.

Some very low-level hardware settings (for instance, the dimensions of a disk mounted) are retained in reminiscence known as "nonvolatile RAM," however those places are not without delay accessible via applications and are written only by packages run from read-handiest memory (ROM) for the duration of hardware initialization. Thus, they are tremendously immune to virus attack.

Viruses can't infect hardware. True. Viruses can infect only matters they could regulate; reminiscence, executable files, and data are the primary objectives. If hardware carries writeable garage (so-referred to as firmware) that may be accessed below software manipulate, that garage is situation to virus attack. There have been a few times of firmware viruses. Because a deadly disease can

control hardware this is a problem to program manage, it would seem as if a hardware device has been inflamed through a deadly disease, but it is definitely the software program riding the hardware that has been infected. Viruses can also exercise hardware in any manner an application can. Thus, for instance, an epidemic should reason a disk to loop incessantly, transferring to the innermost track then the outermost and returned once more to the innermost.

Viruses can be malevolent, benign, or benevolent. True. Not all viruses are bad. For example, an endemic may locate uninfected packages, compress them so that they occupy much less reminiscence, and insert a copy of a habitual that decompresses this system when its execution starts. At the equal time, the virus is spreading the compression feature to different applications. This virus could drastically lessen the amount of storage required for stored packages, probably through as much as 50 percent. However, the compression might be performed on the request of the virus, no longer on the request, or maybe know-how, of this system owner.

First example of malicious code: the brain virus

One of the first viruses is also one of the most studied. The so-called Brain virus got its name because it changes the label of any disc that attacks the word "BRAIN". This particular virus, which is believed to have originated in Pakistan, attacks PCs running an old Microsoft operating system. There have been numerous variants; Due to the number of variants, people believe that the source code of the virus was released to the clandestine virus community.

What does

The brain, like all viruses, seeks to transmit its infection. This virus first sits in the upper memory and then executes a system call to reset the upper memory linked under itself so that it is not affected while it is running. Catch the interrupt number 19 (disk read) by resetting the interrupt address table to point to it and then set the address for interrupt number 6 (not used) to the previous address of interrupt 19. From this way, antivirus screens detect calls on the disk. , handling of anyone who reads the start sector (returning the original start content that was moved to one of the bad sectors); other disk calls go to the normal disk read controller, through interrupt 6.

The brain virus seems to have no more effect than transmitting the infection, as if it were an experiment or a proof of concept. However, virus variants either erase the disks or destroy the file allocation table (the table that shows what files are on a storage medium).

How it spreads

The brain virus is positioned in the boot sector and in six other sectors of the disk. One of the six sectors will contain the original start code, moved there from the original boot sector, while two others contain the remaining code of the virus. The remaining three sectors contain a duplicate of the others. The virus marks these six sectors as "defective" so that the operating system does not try to use them. (With low-level calls, you can force the disk drive to read what the operating system has marked as bad sectors). The virus allows the startup process to continue.

Once established in memory, the virus intercepts disk read requests for the disk drive under attack. With each reading, the virus reads the disk boot sector and inspects the fifth and sixth bytes for the hexadecimal value 1234 (your signature). If it finds that value, it concludes that the disk is infected; If not, it infects the disk as described in the previous paragraph.

What was learned

This virus uses some of the standard tricks of viruses, such as concealment in the boot sector and interception and interrupt detection. The virus is almost a prototype for later efforts. In fact, many other virus writers seem to have modeled their work on this basic virus. Therefore, it could be said that it was a useful learning tool for the virus writers community.

Unfortunately, their infection did not raise public awareness of the viruses, apart from a certain amount of fear and misunderstanding. The subsequent viruses, such as the Lehigh virus that spread through the computers of Lehigh University, the nVIR viruses that emerged from the prototype code published on the bulletin boards and the Scores virus that was first found at NASA in Washington DC circulated more widely and with greater effect. Fortunately, most viruses seen to date have a modest effect, such as displaying a message or making a sound. That

is, however, a matter of luck, since the writers who could put together the simplest viruses obviously had all the talent and knowledge to make viruses much more malevolent. There is no general cure for viruses. Virus scanners are effective against the known viruses of today and the general patterns of infection, but they can not counteract the variant of tomorrow. The only safe prevention is the complete isolation of external contamination, which is not feasible; in fact, you may even get a virus from software applications that you buy from reputable provider

Instance: The Internet Worm

In the evening of a couple of November 1988, an earthworm was released to the particular Internet, leading to serious problems for the system. Not only were numerous systems infected, and also whenever word of the difficulty spread, many more uninfected systems severed their community connections to prevent by themselves from getting infected. Spafford and his team in Purdue University andEichen plus Rochlis at M. I actually. T.studied the worm thoroughly, and Orman did an exciting retrospective analysis 15 yrs following your incident.

Note: This incident is usually normally known as "worm, " although it gives most of the features of viruses.

The criminal was Robert T. Morris, Jr., a graduate pupil at Cornell University which created and released typically the worm. Having been guilty in 1990 of breaking the 1986 Computer Scam and Abuse Act, part 1030 of U. S i9000. Code Title 18. He or she received a fine involving $10, 000, a three-year suspended jail sentence, and even was required to carry out 400 hours of local community service.

What It Do

Judging from its computer code, Morris programmed the Net worm to achieve three major objectives:

Determine where that could spread to.

Pass on its infection.

Remain undocumented and undiscoverable.

What Influence It Had

The worm's primary effect was reference exhaustion. Its source program code indicated that the earthworm was supposed to check out whether a target number was already infected; if you do, the worm would work out so that either typically the existing infection or the particular new infector would eliminate. However, because of a new supposed flaw in typically the code, innovative copies performed not terminate. Therefore, a great infected machine soon grew to be burdened with many reports of the worm, just about all busily attempting to pass on the infection. Thus, the particular primary observable effect has been serious degradation in overall performance of affected machines.

A new second-order effect was typically the disconnection of many devices from the Internet. Method administrators tried to serious their connection with typically the Internet, either because their very own machines were already afflicted and the system directors wanted to keep typically the worm's processes from seeking for sites that in order to spread or because their very own machines were not but infected and the staff members wished to avoid having all of them become so.

The disconnection led to a third-order effect: isolation and lack of ability to perform necessary function. Disconnected systems could certainly not contact other systems in order to carry on the standard research, collaboration, business, or perhaps information exchange users predicted. System administrators on shut off systems could not employ the network to change data with their counterparts with other installations, so reputation and containment or healing information was unavailable.

Typically the worm caused an believed 6, 000 installations in order to shut down or detach from the Internet. Altogether, several thousand systems had been disconnected for several days and nights, and several hundred regarding these systems were shut down to users for some sort of day or more when they were disconnected. Estimations of the cost associated with the damage range through $100, 000 to $97 million.

How It Performed

The worm exploited a number of known flaws and construction failures of Berkeley edition 4 of the Unix operating system. It accomplishedor had code that made an appearance to attempt to accomplishits about three objectives.

Determine where in order to spread. The worm experienced three techniques for tracking down potential machines to victimize. It first tried to be able to find user accounts to be able to invade on the focus on machine. In parallel, the particular worm attempted to take advantage of a bug within the ring finger program and then to be able to utilize a trapdoor inside the sendmail mail handler. All three of these types of security flaws were nicely known within the general Unix community.

The initial security drawback was a joint consumer and system error, inside which the worm attempted guessing passwords and prevailed because it found one. The particular Unix password file is definitely trapped in encrypted form, nevertheless the ciphertext in the particular file is readable by simply anyone. (This visibility is usually the system error.) The worm encrypted different popular passwords and as opposed their ciphertext to the particular ciphertext of the stashed password file. The earthworm tried the account label, the owner's name, in addition to a short list involving 432 common passwords (such as "guest, " "password, " "help, " "coffee, " "coke, " "aaa"). If none of these kinds of succeeded, the worm employed the dictionary file saved on the system regarding use by application transliteration checkers. (Choosing a well-known password is the consumer error.) When that got a match, typically the worm could log throughout to the corresponding accounts by presenting the plaintext password. Then, as a great user, the worm can try to find other machines to be able to which the person could attain access. (See the write-up by Robert T. Morris, Sr. and Ken Thompson [MOR79] upon selection of good security passwords, published a decade ahead of the worm, and typically the section in Chapter 5 on passwords people pick.)

The 2nd flaw concerned fingerd, this program that runs continuously as a solution to other computers' desires for facts about method users. The safety drawback involved causing the insight buffer to overflow, pouring into the return handle stack. Thus, when typically the finger call terminated, fingerd executed instructions that got been pushed there a good additional part of typically the

buffer overflow, evoking the particular worm to be attached to a web-based layer.

The third flaw engaged a trapdoor in the particular sendmail program. Ordinarily, this specific program runs in the particular background, awaiting signals by others wanting to give mail to the technique. When it receives like a signal, sendmail becomes a destination address, which in turn it verifies, and next begins a dialog to be able to receive the message. On the other hand, when utilizing debugging function, the worm causes sendmail to receive and perform a command string instead than the destination tackle.

Spread infection. Having located a suitable target equipment, the worm would work with one of these about three techniques to send a bootstrap loader to the concentrate on machine. This loader comprised of 99 lines associated with C code to get put together and executed for the goal machine. The bootstrap termesconseillés would then fetch typically the rest of the earthworm from the sending web host machine. An element involving good computer securityorstealthwas included in the exchange involving the host and the particular target. When the target's bootstrap requested the relaxation of the worm, typically the worm supplied an just one time password back to the particular host. Without this pass word, the host would instantly break the connection to be able to the target, presumably inside an effort to make sure against "rogue" bootstraps (ones a real administrator may well develop to try to be able to obtain a copy in the rest of the earthworm for subsequent analysis).

Stay undiscovered and undiscoverable. Typically the worm visited considerable plans to prevent its breakthrough discovery once established on the sponsor. For instance, if the transmission error occurred when the remaining portion of the worm seemed to be being fetched, the termesconseillés zeroed and then wiped all code already transported and then exited.

Simply because soon as the earthworm received its full computer code, it brought the program code into memory, encrypted that, and deleted the unique copies from disk. Therefore, no traces were still left on disk, and perhaps a memory dump would certainly not readily expose the particular worm's code. The earthworm periodically changed thier brand and process identifier to ensure that no single name might increase a large quantity of computing time.

The thing that was Learned

The Internet earthworm sent a shock say through the Internet local community, which at that moment was largely populated simply by academics and researchers. The particular affected sites closed several of the loopholes used by the worm in addition to usually tightened security. Several users changed passwords. A couple of researchers, Farmer and Spafford , developed some sort of program for system managers to check for a few associated with the same flaws typically the worm exploited. However, protection analysts checking for web site vulnerabilities across the Web find that most of the same exact security flaws remain in existence nowadays. A new attack on the net would not succeed in the same scale while the Internet worm, nonetheless it could still cause considerable inconvenience to many.

Typically the Internet worm was not cancerous in that just extended to other systems nevertheless did not destroy any kind of portion of them. It gathered sensitive data, such like account passwords, but that did not retain these people. While acting as a good user, the worm can have deleted or overwritten files, distributed them anywhere else, or encrypted them in addition to held them for ransom. The next worm might not be so harmless.

The worm's effects stirred several people to motion. One positive outcome by this experience was growth of an infrastructure with regard to reporting and correcting malevolent and nonmalicious code imperfections. The Internet worm took place around the same period that Cliff Stoll reported the problems in tracking a good electronic intruder (and their subsequent difficulty in obtaining one to deal along with the case). The laptop or computer community realized it required to organize. The resulting Personal computer Emergency Response Team (CERT) at Carnegie Mellon University or college was created; it plus similar response centers about the world have executed a great job of gathering and disseminating information in malicious code attacks plus their countermeasures. System directors now exchange home elevators issues and solutions. Security arrives from informed protection in addition to action, not from lack of edcuation and inaction.

3.4. Targeted Malicious Code

To date, we have looked with anonymous code written to be able to affect users and equipment indiscriminately. Another class involving malicious code is created for a particular technique, for a particular software, and for a certain objective. Many of the computer virus writers' techniques apply, nevertheless additionally, there are some new kinds. Bradbury [BRA06] looks at the switch over time in aims and skills of harmful code authors.

Trapdoors

A new trapdoor is an unrecorded access point to a new module. Developers insert trapdoors during code development, probably to test the component, to provide "hooks" by simply which to get in touch future adjustments or enhancements, or to be able to allow access in the event the component should fail down the road. Throughout addition to these reliable uses, trapdoors can permit programmer access in order to a program once it's placed in production.

Cases of Trapdoors

Because calculating systems are complex setups, programmers usually develop in addition to test systems in a new methodical, organized, modular method, taking advantage of typically the way the product is constructed of modules or pieces. Often, programmers first test out each small component regarding the machine separate from typically the other components, in the step called unit assessment, to ensure that the particular component works correctly simply by itself. Then, developers test out components together during the use testing, to view how that they function as they deliver messages and data from a single to the other. Quite than paste each of the parts together in a "big bang" approach, the testers group logical clusters associated with some components, and each and every cluster is tested inside a way that permits testers to control plus understand what will make some sort of component or its user interface fail.

To be able to test a component itself, the developer or specialist cannot use the bordering routines that prepare suggestions or work with end result. Instead, it is almost always necessary in order to write "stubs" and "drivers, " simple routines in order to inject data in and even extract results from typically the component being tested. Because testing continues, these slip and drivers are removed

because they are substituted by the actual pieces whose functions they simulate. For example, the a couple of modules MODA and MODB in Figure 3.10 happen to be being tested with the particular driver MAIN plus the slip SORT, OUTPUT, and NEWLINE.

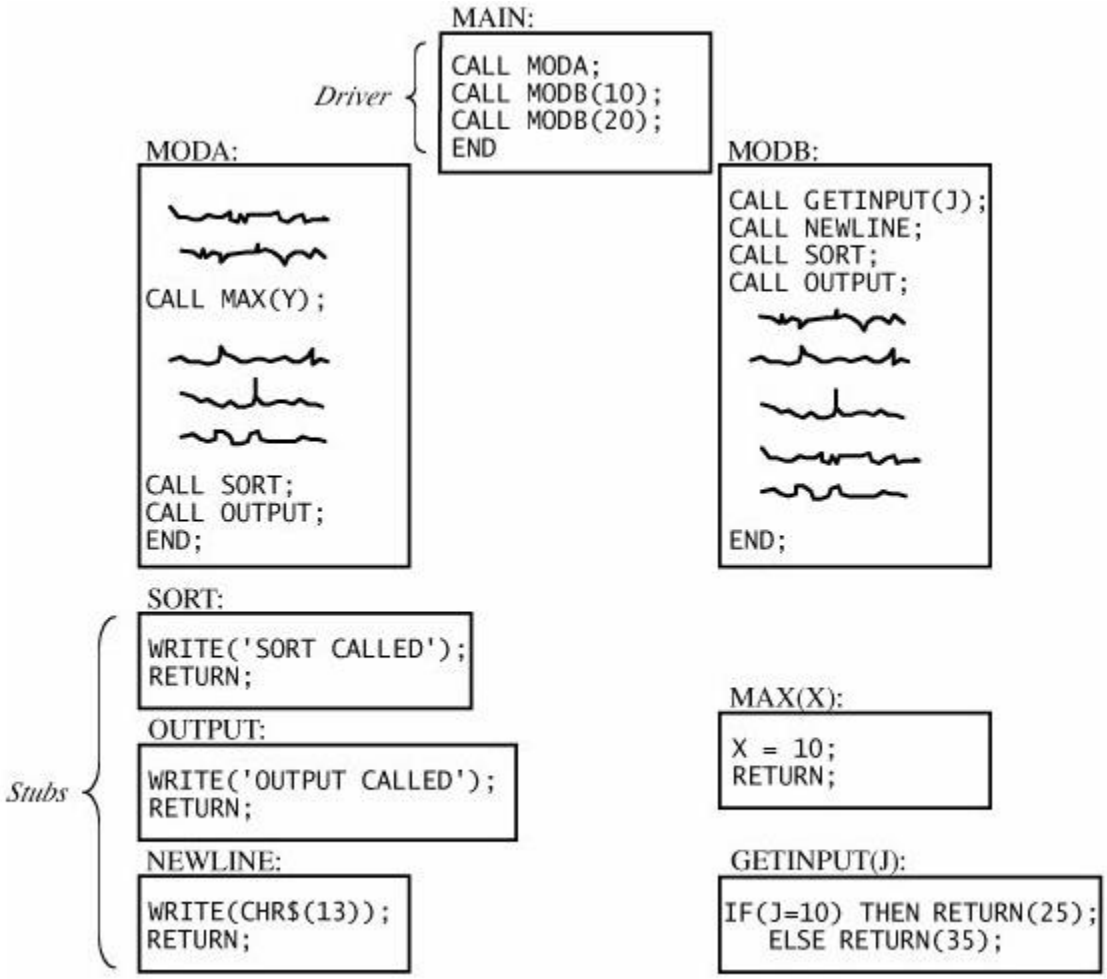


Figure 3.10. Stubs and Drivers.

During both unit in addition to integration testing, faults happen to be usually discovered in elements. Sometimes, when the resource of a problem is not really obvious, the developers put in debugging code in shady modules; the debugging program code makes visible what is definitely going on as being the elements execute and interact. As a result, the extra code may well force components to

exhibit the intermediate results associated with a computation, to print out the number of every step of the method as it is carried out, or to perform more computations to check the particular validity of previous parts.

To control stubs or even invoke debugging code, the particular programmer embeds special handle sequences in the component's design, specifically for assistance testing. For example, a factor in a text format system might be developed to recognize commands like as. PAGE,. TITLE, plus. SKIP. During testing, typically the programmer may have invoked the debugging code, making use of a command with the series of parameters with the form var = worth. This command allows typically the programmer to modify typically the values of internal software variables during execution, possibly to try corrections to this specific component or supply ideals passed to components this particular one calls.

Command insert is a recognized tests practice. However, if kept in place after the screening, the excess commands can turn out to be a problem. These are unrecorded control sequences that create side effects and could be used as trapdoors. In fact, the world wide web earthworm spread its infection simply by using just such the debugging trapdoor in the electronic mail program.

Inadequate error checking can be another resource of trapdoors. A very good developer will design the system to ensure that any information value is checked ahead of it is used; the particular checking involves making certain typically the data type is right and also ensuring that typically the value is within appropriate bounds. But in a few poorly designed systems, unwanted input may not end up being caught and can end up being transferred for use within unanticipated ways. For illustration, a component's code might check for one associated with three expected sequences; obtaining none of the few, it should recognize a good error. Suppose the creator uses a CASE declaration to look for every one of the three choices. A careless programmer may possibly allow a failure only to fall through the CIRCUMSTANCE without having to become flagged as a problem. The finger flaw taken advantage of with the Morris worm happens exactly that way: The C library I/O regular fails to check no matter if characters are left within the input buffer prior to returning a pointer to some supposed next character.

Equipment processor design provides one other common example of this kind of security downside. Here, it often takes place that not all probable binary opcode

values include matching machine instructions. Typically the undefined opcodes sometimes carry out peculiar instructions, either mainly because of an intent to be able to test the processor design and style or because of the oversight by the processor chip designer. Undefined opcodes are usually the hardware counterpart involving poor error checking intended for software.

Just like viruses, trapdoors are not always awful. They can be extremely useful to find security faults. Auditors sometimes request trapdoors in production programs in order to insert fictitious but well-known transactions to the system. In that case, the auditors trace the particular flow of those transactions by way of the system. Nevertheless , trapdoors must be documented, use of them should be highly controlled, and they should be designed and employed with full understanding involving the actual consequences.

Reasons for Trapdoors

Developers usually get rid of trapdoors during program growth, once their intended performance is spent. However, trapdoors can persist in manufacturing programs because the builders

miss to remove these people

intentionally leave them within the program for testing

purposely leave them in the particular program for maintenance regarding the finished program, or even

intentionally leave them within the program as a nanny ways of access to the particular component after it will become an accepted part associated with a production system

Typically the first case is a good unintentional security blunder, the particular next two are significant exposures of the anatomy's security, and the 4th is the very very first step of an downright attack. You should remember that will the fault is just not using the trapdoor itself, that can be an useful technique intended for program testing, correction, in addition to maintenance. Rather, the wrong doing is with the machine growth process, which will not make sure that the trapdoor is definitely "closed" when it will be no longer needed. That may be, the trapdoor

IDOL

becomes a new vulnerability if no one particular notices it or functions to prevent or handle its use in prone situations.

In general, trapdoors really are a vulnerability when that they expose the program to customization during execution. They can easily be exploited by the particular original developers or utilized by anyone that finds out the trapdoor by chance or perhaps through exhaustive trials. Some sort of system is not protected when someone believes that will no person else would discover the hole.

Salami Attack

We noted in chapter 1 a trigger known while a salami attack. This particular method gets its brand from the way peculiar bits of meat and even fat are fused within a sausage or salami. Just as, a salami attack integrates bits of seemingly of no concern data to yield strong results. For example, courses often disregard small sums of money in their particular computations, as when right now there are fractional pennies like interest or tax is definitely calculated. Such programs might be subject to a new salami attack, because typically the a small amount usually are shaved from each calculation and accumulated elsewheresuch while in the programmer's banking account! The shaved amount is really small that an personal case is unlikely to be able to be noticed, and typically the accumulation can be completed in order that the books still equilibrium overall. However, accumulated portions can also add upward to a tidy quantity, supporting a programmer's earlier retirement or new automobile. It is usually the resulting expenses, not the shaved sums, that provides the attention associated with the authorities.

Types of Salami Attacks

The classic experience of a salami harm involves interest computation. Presume your bank pays six. 5% interest on your current account. The eye is definitely declared on an yearly basis but is determined monthly. If, after typically the first month, your traditional bank balance is $102. 87, the lender can calculate typically the interest inside the following method. For a month together with 31 days, we split the interest rate simply by 365 to get the particular daily rate, and and then multiply it by 23 to get the curiosity for your month. Thus, typically the total interest for 31st days is 31/365*0. 065*102. 87 = $0. 5495726. Since banks deal simply in full cents, a new typical practice is to be able to round down if some

sort of residue is no a lot more than half a penny, and gather if the residue is a break up cent or more. On the other hand, people check their appeal computation closely, and much less still would complain concerning getting the amount $0. 5495 rounded down to $0. 54, as opposed to up to be able to $0. 55. Most courses that perform computations in currency recognize that as a result of rounding, a sum associated with individual computations may become a few cents diverse from the computation placed on the sum of the particular balances.

How it shifts these fractional cents? Typically the pc security folk story is told of the programmer who collected typically the fractional cents and awarded them to just one bank account: hers! The interest system merely had to stability total interest paid in order to interest due on typically the total in the balances associated with the individual accounts. Auditors will probably not see the activity in one particular specific account. In periods with many accounts, the particular roundoff error can become substantial, along with the programmer's consideration pockets this roundoff.

Although salami attacks can internet more and be much more interesting. For example of this, as opposed to shaving fractional mere cents, the programmer may get a few cents by each account, again if, perhaps that no individual has got the desire or understanding in order to recompute the amount typically the bank reports. Most individuals finding a result several cents different from of which of the lender would take the bank's figure, that attributed the difference to an error throughout arithmetic or even a misunderstanding involving the conditions under which often interest is credited. Or even a program might guideline them with a 20 dollars fee for a specific service, while the business standard is $15. In the event that unchecked, the excess $5 may be credited for an account regarding the programmer's choice. Typically the amounts shaved aren't always small: One attacker had been able to make withdrawals of $10, 000 or even more against accounts of which had shown little latest activity; presumably, the opponent hoped the owners had been ignoring their accounts.

The reason why Salami Attacks Persist

Personal computer computations are notoriously susceptible to small errors involving rolling and truncation, especially any time vast quantities are to be put together with small ones. Quite than document the specific errors, it is less

difficult for programmers and customers to accept a few problems as natural and inescapable. To reconcile accounts, the particular programmer includes a blunder a static correction in computations. Inadequate auditing of these corrections is 1 reason why the salami attack may be disregarded.

Usually, the origin code associated with a system is also big or complex in order to be audited for salami attacks unless there is usually a reason to suspect one particular. Size and time happen to be definitely in the area of the malicious designer.

Rootkits and the Sony XCP

A later variant on the virus style may be the rootkit. A rootkit is really a piece of harmful code which goes in order to great lengths to not end up being discovered or, if uncovered and removed, to improve itself whenever possible. Subject rootkit refers to the particular code's try to operate since root, the superprivileged consumer of a Unix technique.

A typical rootkit may interfere with the standard interaction betweena consumer and the os while follows. Whenever the end user executes a command of which would demonstrate rootkit's occurrence, for example, by position files or processes inside memory, the rootkit intercepts the call and filter the result returned in order to the user so of which the rootkit does not necessarily appear. For example, in the event that a directory contains 6 files, one of which can be the rootkit, the rootkit will pass the listing command to the working system, intercept the end result, delete the listing intended for itself, and display towards the user only the several other files. The rootkit will likely adjust such items as file size counts to conceal itself. Realize that the rootkit needs in order to intercept this data in between the result and typically the presentation interface (the plan that formats results with regard to the user to see).

Ah, two can enjoy that game. Suppose an individual suspect code is modifying your file display plan. Then you write a new program that displays data, then examines the storage and file system straight to enumerate files, and examines these two results. A new rootkit revealer is simply such a program.

Some sort of computer security expert known as Mark Russinovich developed a new rootkit revealer, which he or she ran on one regarding his systems. He had been surprised to locate a rootkit. On further exploration, he determined the rootkit had been installed whenever he loaded and performed a music CD within the computer. Felten plus Halderman extensively examined this rootkit, named XCP (short intended for extended copy protection).

Just what XCP Will

The XCP rootkit prevents a consumer from copying a songs CD while allowing the particular CD to get played while music. To get this done, it involves its own special really good music player that will is allowed to have fun with the CD. But XCP disrupts any other gain access to to the protected songs CD by garbling the particular result any other course of action would obtain in seeking to read from typically the CD.

The rootkit features to install itself if the CD is first placed inside the PC's drive. To be able to do this, XCP is dependent on a "helpful" function of Windows: With "autorun" Windows looks for a new file having a specific label, and if it finds out that, it opens and even executes the file without having the user's involvement. (The file name can become configured in Windows, though it is autorun. exe by default.) An individual can disable the autorun feature.

XCP has to hide through the user so of which the user cannot only remember to remove this. So the rootkit will as we just referred to: It blocks display regarding any program whose title begins with sys (which is how it will be named). Unfortunately for Fiat, this feature concealed not necessarily just XCP but virtually any program beginning with sys from any source, harmful or not. So virtually any virus writer could hide a virus just simply by naming it sysvirus-1, intended for example.

Sony did several things wrong: First, even as we just observed, it sent out code that inadvertently frees an unsuspecting user's method to possible infection simply by other writers of destructive code. Second, Sony sets up that code without the particular user's knowledge, much much less consent, and it utilizes strategies to prevent the particular code's removal.

Patching typically the Penetration

The storyline of XCP became very public within November 2005 when Russinovich described what he located and several news solutions picked up the account. Confronted with serious negative marketing, Sony decided to launching an uninstaller for the particular XCP rootkit. Remember, on the other hand, from the start associated with this chapter why "penetrate and patch" was left behind as a security approach? The pressure for the quick repair sometimes directed to shortsighted solutions of which addressed instant situation in addition to not the underlying lead to: Fixing one problem usually caused a failure anywhere else.

Sony's uninstaller by itself opened serious security slots. It was presented as being a web page that saved and executed the deletion. However the programmers did certainly not check what code they will were executing, hence the website page would run any kind of code from any resource, not just the planned uninstaller. And worse, typically the downloading code remained sometimes after uninstalling XCP, which means that the vulnerability remained. (In fact, Sony applied two different rootkits coming from two different sources and even, remarkably, the uninstallers with regard to both rootkits had this particular same vulnerability.)

The number of computers were infected with this rootkit? Nobody knows without a doubt. Kaminsky found 500, 000 referrals in DNS tables towards the site the rootkit connections, but some of individuals DNS entries could help accesses by hundreds or perhaps thousands of computers. The number of users of computers where the rootkit was mounted are aware of this? Again nobody knows, or does anybody know exactly how many of those installation may not yet have already been removed.

Effect of Privilege Escalation

Some sort of malicious code writer wants a privilege escalation. Producing, installing, or modifying a method file is difficult, however it is easier to load some sort of file to the user's area. In this example, typically the malicious code writer simply has to create a new small shell program, title it Sys3, store that anywhere (even in some sort of temporary directory), reset the particular search path, and employ a program (Live Update). Each of these behavior is common for nonmalicious downloaded code.

The consequence of operating this attack would be that the harmful version of Sys3 gets control in privileged function, and from that stage it could replace

operating method files, download and set up new code, modify method tables, and inflict almost some other harm. Having manage once with higher opportunity, the malicious code can easily set a flag in order to receive elevated privileges within the future.

Interface Misunderstandings

The name for this particular attack is borrowed by Elias Levy . An interface illusion is usually a spoofing attack within which all or element of an online page will be false. The thing of typically the attacker is to persuade the user to perform something inappropriate, for example, to enter personal bank information about a site that will is not the bank's, to click yes on the button that actually implies no, or simply to be able to scroll the screen to be able to activate a celebration that leads to malicious software being set up on the victim's device. Levy's excellent article offers other excellent examples.

The thing is that every dot associated with the screen is addressable. So if a real user interface can paint dot seventeen red, so can some sort of malicious interface. Considering the fact that, the malicious interface can show phony address bars, scroll night clubs that are not browse bars, and even some sort of display that looks the same to the real point, because it is similar in all ways typically the attacker wants it to be able to be.

Nothing here is definitely new, of course. Individuals diligently save copies regarding e-mail messages as evidence that they received many of these a message when, in fact a simple text manager will produce any authentic-looking message you want. Program pranksters love to send facetious messages to unsuspecting consumers, warning that this computer will be annoyed. All of these types of derive from the identical point: There is little or nothing unique, no trusted way assured to be the private and authentic conversation channel directly to typically the user.

Keystroke Signing

Keep in mind the movies in which often a detective would criminal a note pad upon a desk, hold upward to the light, plus browse the faint impression associated with a message that got been written and after that torn off that protect? We have a computer counterpart associated with that tactic, too.

1st, recognize that there basically a direct path in between an important you press in your keyboard and typically the program (let's say the word processor) that deals with that keystroke. When an individual press A, it triggers a switch that creates a sign that is usually received by a gadget driver, converted and assessed and passed along, till finally your word processor chip receives the A; there exists still more conversion, examination, and transmission until typically the A appears on your current screen. Many programs get close to in this chain. In several points in the process an individual could change a plan in order that A would look on the screen whenever you pressed W should you wanted.

If all plans work as intended, that they receive and send character types efficiently and discard just about every character as soon while it is sent in addition to another arrives. A malevolent program called a keystroke logger retains a surreptitious copy of all important factors pressed. Most keystrokes usually are uninteresting, but we may well want to protect typically the privacy of identification amounts, authentication strings, and enjoy notes.

A keystroke logger can be independent (retaining a log of each key pressed) or that can be tied to be able to some program, retaining files only if a specific program (such as being a consumer banking application) runs.

Man-in-the-Middle Attacks

A keystroke logger is a special contact form of the greater general man-in-the-middle attack. You can find two editions of this attack: all of us cover the application variety here and then grow around the concept in Phase 7 on networks.

A new man-in-the-middle attack is a single in which a malevolent program interjects itself in between two other programs, commonly between an user's insight and an application's effect. One example of a new man-in-the-middle attack could get a program that managed between your word cpu along with the file system, consequently that each time an individual thought you were conserving your file, the mid program prevented that, or even scrambled your text or even encrypted your file. Just what ransom would you become willing to pay to be able to get back the document on which you acquired been employed by the final week?

Timing Assaults

Personal computers are fast, and they also function far faster than people can follow. However since we all know, typically the time it takes a pc to perform an activity depends on the level the task: Creating 30 database records takes around two times as well as creating 10. And so, in theory at the very least, whenever we could solution computer time precisely, and even we could control various other things being done found in the computer, we can infer the size regarding the computer's input. Inside of most situations size is certainly relatively uninteresting towards the assailant. But in cryptography, however, smallest bit of data could be significant.

Brumley and even Boneh investigated a program that will does RSA encryption intended for web sites. The creators try to derive the real key by successive guesses involving accelerating value as options for your key. Although typically the details of the harm are beyond the opportunity of this book, the concept is to use a strategy inside the optimization of RSA encryption. Grossly oversimplified, security with numbers less compared to the key take consecutively, sequentially longer amounts of period as the numbers acquire nearer to the important, but then the time period to encrypt drops dramatically once the key price is passed. Brute pressure guessing is prohibitive throughout time. But the experts show that you avoid have to try almost all values. You infer the real key a few bits with a time from your remaining (most significant bit). Thus you might try 00xxx, 01xxx, 10xxx, and 11xxx, noticing that the time period to compute rises by 00xxx to 01xxx, increases from 01xxx to 10xxx, and falls between 10xxx and 11xxx. This informs you the key price is between 10xxx in addition to 11xxx. The attack functions with a lot longer keys (on the order of multitude of bits) as well as the authors employ about a million choices for the xxx part. Still, this technique enables the authors to infer the real key a bit with time, all in line with the amount of time the particular encryption takes. The experts performed their experiments upon a network, avoid accurate local timing instruments, in addition to still, they were capable to deduce keys.

IDOL

Cryptography is the primary place in which speed in addition to size is information which will not be revealed. Although you must be mindful that malicious code is capable of doing similar attacks undetected.

Concealed Channels: Programs That Drip Information

So far, many of us have looked over malicious computer code that performs unwelcome steps. Next, we turn to be able to programs that communicate info to people who should never receive it. The conversation travels unnoticed, accompanying various other, perfectly proper, communications. The overall name for these remarkable paths of communication is usually covert channels. The principle of a covert route comes from a document by Lampson; Millen presents a good taxonomy of covert channels.

Presume several students is organizing for an exam regarding which each question features four choices (a, w, c, d); one college student in the group, Sophie, understands the material completely and he or the girl agrees to help typically the others. States she will certainly reveal the answers to be able to the questions, in buy, by coughing once regarding answer "a, " sighing for an answer "b, inch and etc .. Sophie uses the communications channel that outsiders may not notice; the girl communications are concealed the open channel. This interaction is a human illustration of a covert station.

We start by conveying how a programmer can make covert channels. The strike is more complex compared to one by a solitary programmer accessing an info source. A programmer that has direct access in order to data can usually only read the data and even write it to a new data file or print it. In case, however, the programmer is usually one step removed through the data, for example, exterior the organization owning typically the data the programmer must physique how to get with the data. One way will be to supply a bona fide program having a pre-installed Trojan horse; after the horses are enabled, it locates and transmits the info. Even so, it would be also bold to generate a new report labeled "Send this kind of report to Jane Johnson in Camden, Maine"; the particular programmer has to organize to extract the information more surreptitiously. Covert programs are ways of extracting info clandestinely.

Figure 3.11 programs a "service program" made up of a Trojan horse of which tries to copy details from a legitimate consumer (who is allowed usage of the

information) to the "spy" (who ought certainly not to be permitted to access the particular information). The consumer may not necessarily know that a Trojan malware horse is running and could not be in accord to leak information to be able to the spy

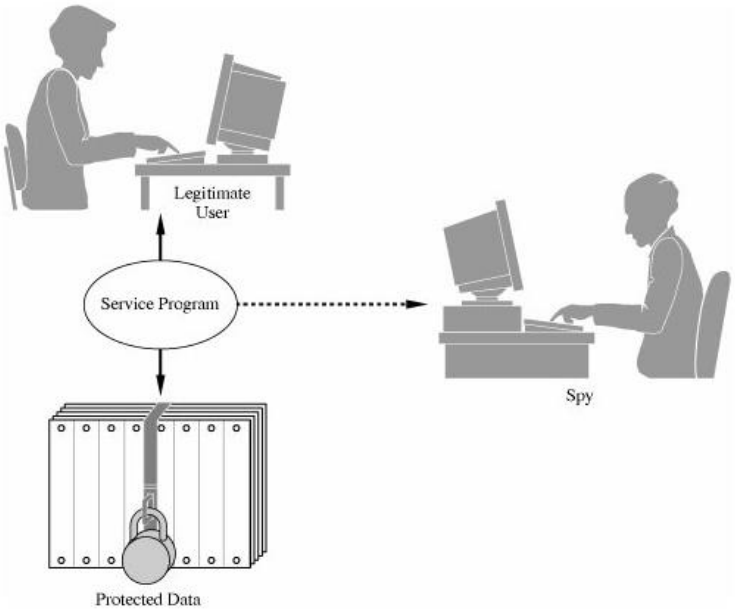


Figure 3.11. Covert Channel Leaking Information.

Covert Channel Overview

A developer should not have gain access to sensitive data of which a program processes following your program has been set into operation. For instance, a programmer for any standard bank has no need in order to access the names or even balances in depositors' balances. Programmers to get a securities company to have no need to be aware of what buy and market orders exist for typically the clients. During program assessment, access to the actual data may be sensible, but not following your software has been accepted intended for regular use.

Still, the programmer might be in a position to profit from the reassurance that a customer is on the subject of to sell a lot of some sort of particular stock or that the large new account only been opened. Sometimes the programmer may want to be able to develop a program of which secretly communicates a few of the info on which it functions. In this case, typically the programmer will be the "spy,

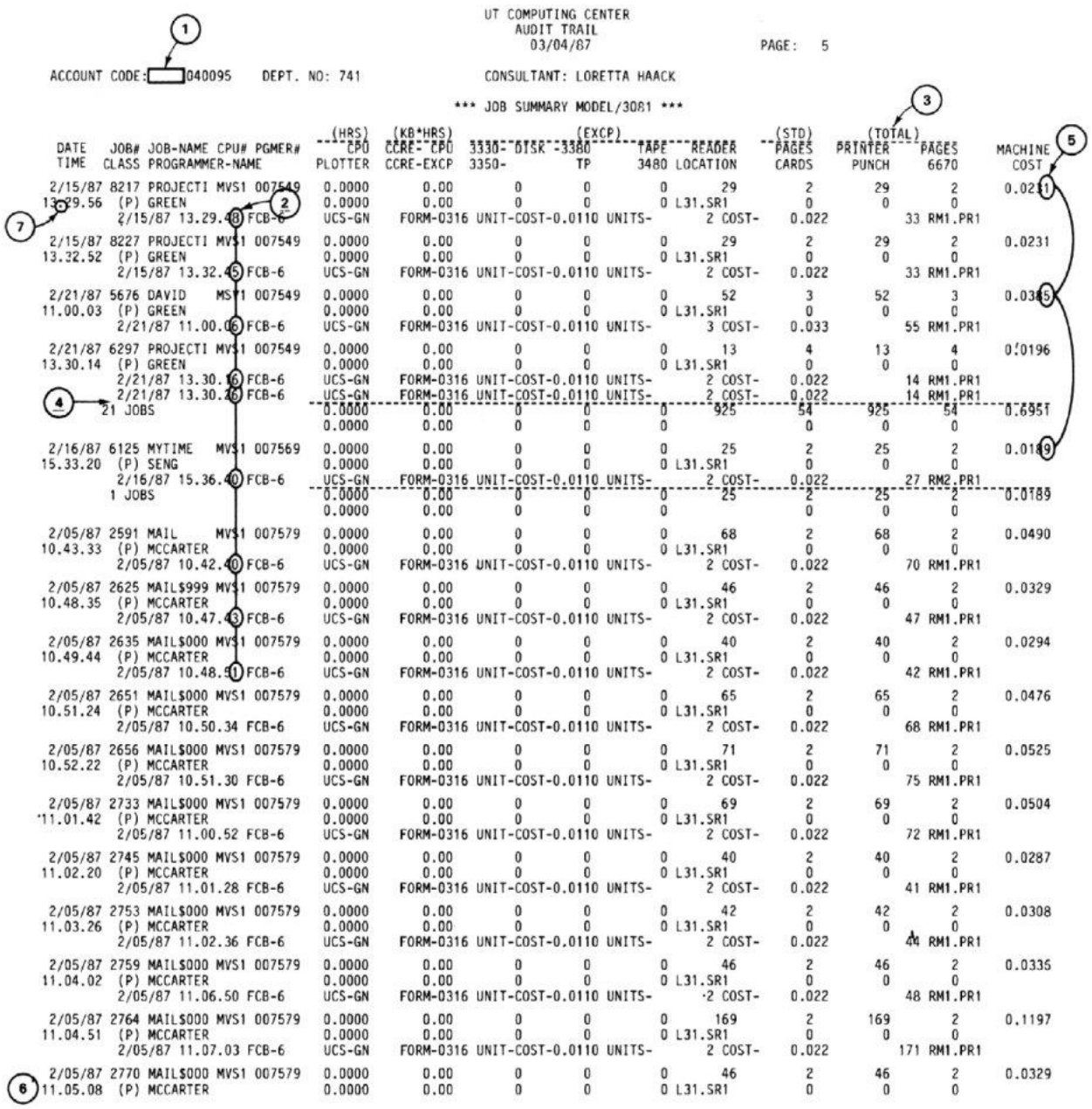
inches and the "user" will be whoever ultimately runs typically the program written by typically the programmer.

How to Create Covert Channels

A developer can always find techniques to communicate data ideals covertly. Running a plan that produces a particular output report or exhibits a worth may become too obvious. For occasion, in some installations, the printed report might from time to time be scanned by safety measures staff before it is usually delivered to its designed recipient.

If printing typically the data values themselves is actually obvious, the programmer may encode the data beliefs within innocuous report simply by varying the format regarding the output, changing the particular lengths of lines, or perhaps printing delete word making certain values. For occasion, changing the word "TOTAL" to "TOTALS" in a new heading will not be seen, but this creates some sort of 1-bit covert channel. Typically the absence or presence associated with the S conveys one particular bit of information. Number values can be put in insignificant positions regarding output fields, and the particular number of lines each page can be transformed. Types of these subtle programmes are shown in Figure 3.12.

IDOL



UT COMPUTING CENTER
AUDIT TRAIL
03/04/87 PAGE: 5

ACCOUNT CODE: 040095 DEPT. NO: 741 CONSULTANT: LORETTA HAACK

*** JOB SUMMARY MODEL/3081 ***

DATE TIME	JOB# CLASS	JOB-NAME PROGRAMMER-NAME	CPU#	PGMER#	(HRS) CPU PLOTTER	(KB*HRS) CORE-CPU CORE-EXCP	(EXCP) 3330-DISK 3350-	3380 TP	TAPE 3480	READER LOCATION	(STD) PAGES CARDS	(TOTAL) PRINTER PUNCH	PAGES 6670	MACHINE COST
2/15/87 13.29.56	8217 (P)	PROJECTI MVS1 GREEN	007549		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	29 L31.SR1	2 0	29 0	2 0	0.0231
	2/15/87 13.29.48	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		33 RM1.PR1	
2/15/87 13.32.52	8227 (P)	PROJECTI MVS1 GREEN	007549		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	29 L31.SR1	2 0	29 0	2 0	0.0231
	2/15/87 13.32.45	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		33 RM1.PR1	
2/21/87 11.00.03	5676 (P)	DAVID MVS1 GREEN	007549		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	52 L31.SR1	3 0	52 0	3 0	0.0385
	2/21/87 11.00.06	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	3 COST-	0.033		55 RM1.PR1	
2/21/87 13.30.14	6297 (P)	PROJECTI MVS1 GREEN	007549		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	13 L31.SR1	4 0	13 0	4 0	0.0196
	2/21/87 13.30.16	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		14 RM1.PR1	
	2/21/87 13.30.16	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		14 RM1.PR1	
	21 JOBS				0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	925	54 0	925 0	54 0	0.6951
2/16/87 15.33.20	6125 (P)	MYTIME MVS1 SENG	007569		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	25 L31.SR1	2 0	25 0	2 0	0.0189
	2/16/87 15.36.40	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		27 RM2.PR1	
	1 JOBS				0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	25	2 0	25 0	2 0	0.0189
2/05/87 10.43.33	2591 (P)	MAIL MVS1 MCCARTER	007579		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	68 L31.SR1	2 0	68 0	2 0	0.0490
	2/05/87 10.42.40	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		70 RM1.PR1	
2/05/87 10.48.35	2625 (P)	MAIL$999 MVS1 MCCARTER	007579		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	46 L31.SR1	2 0	46 0	2 0	0.0329
	2/05/87 10.47.43	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		47 RM1.PR1	
2/05/87 10.49.44	2635 (P)	MAIL$000 MVS1 MCCARTER	007579		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	40 L31.SR1	2 0	40 0	2 0	0.0294
	2/05/87 10.48.51	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		42 RM1.PR1	
2/05/87 10.51.24	2651 (P)	MAIL$000 MVS1 MCCARTER	007579		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	65 L31.SR1	2 0	65 0	2 0	0.0476
	2/05/87 10.50.34	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		68 RM1.PR1	
2/05/87 10.52.22	2656 (P)	MAIL$000 MVS1 MCCARTER	007579		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	71 L31.SR1	2 0	71 0	2 0	0.0525
	2/05/87 10.51.30	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		75 RM1.PR1	
2/05/87 11.01.42	2733 (P)	MAIL$000 MVS1 MCCARTER	007579		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	69 L31.SR1	2 0	69 0	2 0	0.0504
	2/05/87 11.00.52	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		72 RM1.PR1	
2/05/87 11.02.20	2745 (P)	MAIL$000 MVS1 MCCARTER	007579		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	40 L31.SR1	2 0	40 0	2 0	0.0287
	2/05/87 11.01.28	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		41 RM1.PR1	
2/05/87 11.03.26	2753 (P)	MAIL$000 MVS1 MCCARTER	007579		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	42 L31.SR1	2 0	42 0	2 0	0.0308
	2/05/87 11.02.36	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		44 RM1.PR1	
2/05/87 11.04.02	2759 (P)	MAIL$000 MVS1 MCCARTER	007579		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	46 L31.SR1	2 0	46 0	2 0	0.0335
	2/05/87 11.06.50	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		48 RM1.PR1	
2/05/87 11.04.51	2764 (P)	MAIL$000 MVS1 MCCARTER	007579		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	169 L31.SR1	2 0	169 0	2 0	0.1197
	2/05/87 11.07.03	FCB-6			UCS-GN	FORM-0316	UNIT-COST-0.0110		UNITS-	2 COST-	0.022		171 RM1.PR1	
2/05/87 11.05.08	2770 (P)	MAIL$000 MVS1 MCCARTER	007579		0.0000 0.0000	0.00 0.00	0 0	0 0	0 0	46 L31.SR1	2 0	46 0	2 0	0.0329

① Number of spaces after :
② Last digit in field that would not be checked
③ Presence or absence of word (TOTAL) in header line
④ No space after last line of subtotal
⑤ Last digit in insignificant field
⑥ Number of lines per page
⑦ Use of . instead of :

Figure 3.12. Covert Channels.

Storage Channels

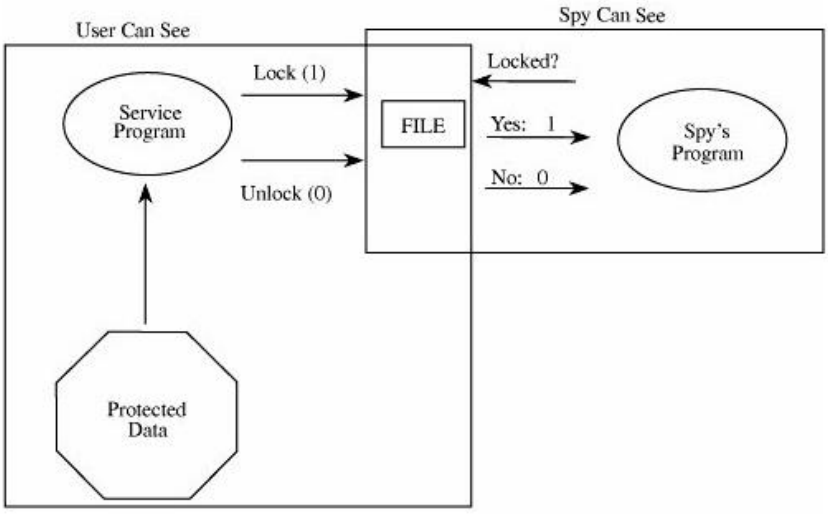
Many covert channels are named storage channels because these people pass information when using the occurrence or absence of items in storage.

A basic sort of a hidden channel will be the file fasten channel. In multiuser devices, files could be "locked" in order to prevent two people through writing to the similar file at the exact same time (which could dodgy the file, if anyone writes over some involving the particular other wrote). The particular main system or database managing system allows only 1 program to write into a file at an occasion by blocking, delaying, or even rejecting write requests coming from other programs. A concealed channel can signal a single bit of information simply by whether or not some sort of file is locked.

Bear in mind that the service plan has a Trojan horse composed by the spy yet run by the unsuspecting customer. As shown in Figure 3-13, the service software reads confidential data (to which the spy must not have access) and indicators the data one little bit at any given time by locking or perhaps not locking some data file (any file, the items of which are human judgments and not even modified). The service program plus the spy need a new common timing source, damaged into intervals. To sign a 1, the power program locks the data file for the interval; with regard to a 0, your locking mechanism. Later in the period, the spy tries in order to lock the file on its own. When the spy program is unable to lock the file, that knows the service plan must have locked the particular file, and thus typically the spy program concludes typically the service program is signaling a one; if the secret agent program can lock the particular file, it knows the particular service program is signaling a 0.

IDOL

Figure 3.13. File Lock Covert Channel



The similar approach can end up being used with disk storage area quotas or other sources. With disk storage, the particular service program signals an one by creating an massive file, so large that will it consumes most regarding the available disk room. The spy program after tries to create some sort of large file. If this succeeds, the spy software infers that the power program did not produce a large file, plus so the service plan is signaling a zero; otherwise, the spy software infers an one. Similarly typically the existence of a data file or other resource regarding a particular name can easily be used to sign. Observe that the spy does indeed not need usage of a new file itself; the only existence of the record is adequate to sign. The spy can identify the presence of some sort of file it cannot study by trying to make a file of the particular same name; if typically the request to create is definitely rejected, the spy decides that the utility plan has such a data file.

To signal more compared to one bit, the services program and the criminal program signal one tad in each time period. Figure 3.14 shows the service program signaling typically the string 100 by toggling the existence of a new file.

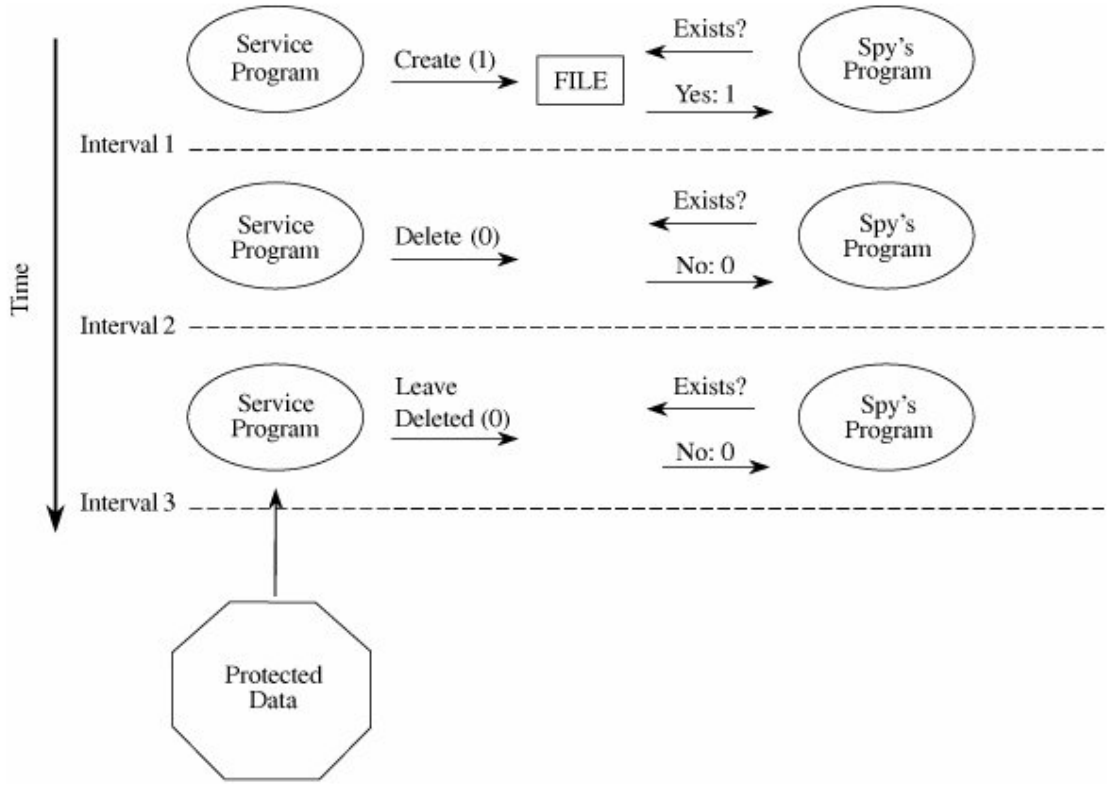


Figure 3.14. File Existence Channel Used to Signal 100.

Inside our final example of this, a storage channel utilizes a server of special identifiers. Recall that many bakeries, banks, as well as other professional establishments have a device to distribute numbered seat tickets so that customers could be served within the purchase in which they appeared. Some computing systems offer a similar server of distinctive identifiers, usually numbers, utilized to name temporary files, in order to tag and track emails, or to record auditable events. Different processes could request the next special identifier from your server. Yet two cooperating processes could use the server in order to send a signal: Typically the spy process observes regardless of whether the numbers it will get are sequential or regardless of whether a number is lacking. A missing number indicates that the service plan also requested a range, thereby signaling 1.

Inside all of these good examples, the service program plus the spy need gain access to to a shared useful resource (such as a data file, or even knowledge associated with the existence of some sort of file) and a contributed sense of

time. Because shown, shared resources happen to be common in multiuser conditions, where the resource could possibly be as seemingly innocuous since whether a file is available, a tool is no cost, or space remains upon the disk. An origin regarding shared time is additionally usually available, since many courses need access to typically the current system time to be able to set timers, to document the time at which in turn events occur, or in order to synchronize activities. Karger and even Wray offer a real-life illustration of a covert route in the movement of the disk's arm and next describe ways to control the potential information seepage from this channel.

Moving data one bit each time must seem awfully gradual. But computers operate from such speeds that your minuscule rate of one bit per millisecond (1/1000 second) would never become noticed but could effortlessly be handled by 2 processes. In which rate regarding 1000 bits per 2nd (which is unrealistically conservative), this entire book can be leaked in concerning two days. Increasing the pace by an order regarding magnitude or two, which often is still quite conventional, reduces the transfer moment to minutes.

Time Channels

Other covert stations, called timing channels, go away information by using typically the speed where things take place. Actually, timing channels happen to be shared resource channels throughout which the shared reference is time.

A support program uses a time channel to communicate by simply using or not utilizing an assigned amount of work time. In the basic case, a multiprogrammed program with two user functions divides time into obstructions and allocates blocks associated with processing alternately to 1 process and the some other. A process is presented processing time, but in the event that the process is awaiting another event to arise and contains no processing to be able to do, it rejects typically the offer. The service procedure either uses its stop (to signal a 1) or rejects its stop (to signal a 0). Such a situation is usually shown in Figure 3.15, first with the support process and the spy's process alternating, and after that with the service procedure communicating the string info to the spy's procedure. In the second element of the example, the particular service program wants to be able to signal 0 in the particular third time block. That will do this through the use of just enough time in order to determine it wants to be able to send a 0 plus then

pause. The criminal process then receives handle for the remainder involving the time block.

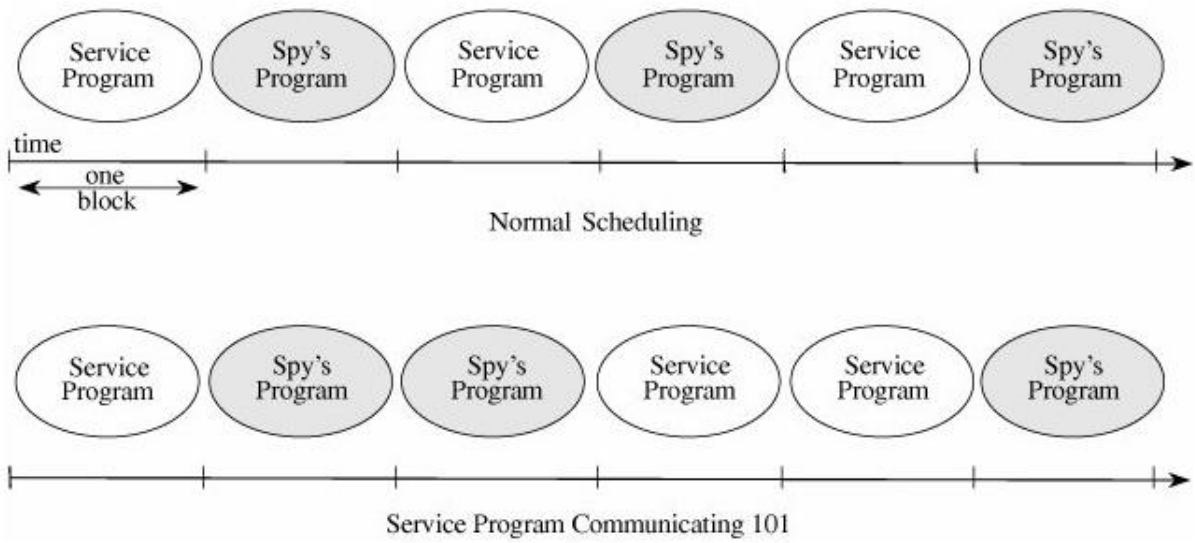


Figure 3.15. Covert Timing Channel.

Thus far, all examples have got involved just the assistance process and the spy's process. In fact, multiuser computing systems typically have got more than just a couple of active processes. The just complications added by even more processes are that typically the two cooperating processes should adjust their timings plus deal with the feasible interference from others. Intended for instance, with the exclusive identifier channel, other operations will also request verifications. If on average d other processes will obtain m identifiers each, then your service program will obtain over n*m identifiers regarding a 1 and not any identifiers for a zero. The gap dominates the particular result of all various other processes. Also, the services process plus the spy's method can use sophisticated code techniques to compress their particular communication and detect and deal with transmission errors caused simply by the consequence of other unrelated techniques.

Identifying Potential Covert Channels

In this particular description of hidden channels, ordinary things, many of these as the existence associated with a file or period used for a calculation, happens to be

the medium via which a covert station communicates. Covert channels happen to be not easy to discover because media are therefore numerous and often utilized. Two relatively old strategies remain the standards regarding locating potential covert stations. One works by examining the time of the system, and the various other works at the resource code level.

Shared Resource Matrix

Since the foundation of a covert funnel is a shared source, the search for possible covert channels involves locating all shared resources plus determining which processes may write to and examine through the resources. The approach was introduced by Kemmerer. Although time-consuming, the technique can become automated.

To make use of this method, you construct a matrix of resources (rows) and even processes that can gain access to them (columns). The matrix entries are R with regard to "can read (or observe) the resource" and Meters for "can set (or modify, create, delete) the particular resource. " For an example of this, the file lock funnel has got the matrix shown inside Table 3.3.

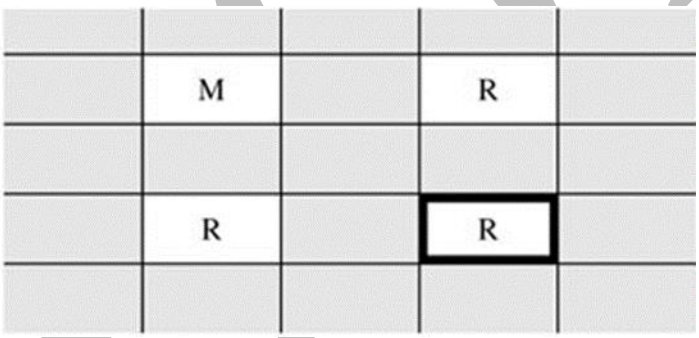
	Service Process	**Spy's Process**
Locked	R, M	R, M
Confidential data	R	

Table 3.3. Shared Resource Matrix

You then look for two columns and two rows having the following pattern:

IDOL

This particular pattern identifies two solutions and two processes in a way that the second process will be not in order to read through the second resource. Nevertheless, the first process can easily pass the information in order to the second by looking at the second useful resource and signaling the info by way of the first resource. As a result, this pattern implies the actual information flow as displayed here.



Next, you full the shared resource matrix by adding these meant information flows and studying the matrix for undesired flows. Thus, you can easily tell the spy's method can read the secret data by using the covert channel through the particular file lock, as displayed in Table 3.4.

	Service Process	**Spy's Process**
Locked	**R, M**	**R, M**
Confidential data	**R**	**R**

Table 3.4. Complete Information Flow Matrix.

Data Flow Approach

Denning derived a new technique for flow research from a program's format. Conveniently, this analysis could be automated within a compiler to ensure that information flow possibilities could be detected while some sort of program is under enhancement.

Using this method, we could recognize nonobvious flows details between statements in the program. For example, all of us know that the affirmation B: =A, which designates the value of A new to the variable M, obviously supports an information flow from A in order to B. This type involving flow is called a good "explicit flow. " Likewise, the pair of claims B: =A; C: =B indicates an information movement from A to Chemical (by way of B). The conditional statement IN THE EVENT THAT D=1 THEN B: =A has two flows: coming from a to B because involving the assignment, but furthermore from D to M, because the value associated with B can change when and only when the price of D is one This second flow is known as an "implicit flow. inches

The statement B: =fcn(args) supports an information circulation in the function fcn to be able to B. At a succinct, pithy level, we could say that will there is a probable flow from the fights args to B. Even so, we could more carefully analyze the function to be able to determine whether the function's value depended on most of its arguments in addition to whether any global ideals, not part of typically the argument list, affecting the particular function's value. These details flows can be followed from the bottom upwards: At the bottom right now there must be functions that will call no other attributes, and can analyze all of them after which use those benefits to analyze the features that call them. Simply by looking at the fundamental functions first, we may say definitively whether at this time there is a potential info flow from each disagreement to the function's end result and whether there are usually any flows from international variables. Table 3-5 databases several samples of syntactic

Statement	**Flow**

B:=A	from A to B
IF C=1 THEN B:=A	from A to B; from C to B
FOR K:=1 to N DO stmts END	from K to stmts
WHILE K>0 DO stmts END	from K to stmts
CASE (exp) val1: stmts	from exp to stmts
B:=fcn(args)	from fcn to B
OPEN FILE f	none
READ (f, X)	from file f to X
WRITE (f, X)	from X to file f

format analysis area of compilation. This particular analysis may also be performed upon the higher-level design standards.

Covert Channel Conclusions

Covert Channel represent a true danger to secrecy in data systems. A covert station attack is fairly superior, but the basic strategy is just not beyond the abilities of even a general programmer. Considering that the subverted system can be practically virtually any user service, such while a printer utility, growing the compromise can always be as easy as growing a virus or any kind of another kind of Trojan malware horse. And up in order to date experience has displayed how readily viruses might be planted.

Capacity in addition to speed does not issue; our estimate of a multitude of bits per second is usually unrealistically low, but perhaps at that rate a great deal information leaks swiftly. Using modern hardware architectures, selected covert channels inherent within the hardware design possess capacities of millions associated with bits per second. And even the attack does not really require

significant finance. As a result, the attack could end up being very effective in most circumstances involving highly sensitive information.

For these reasons, protection researchers have worked faithfully to develop tips for concluding covert channels. The drawing a line under results have been annoying; in ordinarily open surroundings, there is essentially little control of the agitation, destabilization of a utility system, nor is there a good effective way of verification such programs for concealed channels. And other within a few very large security systems, systems are unable to control the flow associated with information from a hidden channel. The hardware-based programmes may not be closed, given the particular underlying hardware architecture.

3.5. Controls Against Program Threats

The particular style we have simply described is simply not pretty. Presently there are many ways an application can fail and a lot of methods to turn the fundamental faults into security problems. It is needless to say much better to focus on elimination than cure; how carry out we use controls in the course of software developmentthe specifying, building, writing, and testing regarding the programto find plus eliminate the sorts associated with exposures we certainly have discussed? Typically the discipline society engineering details this question more internationally, devising methods to ensure typically the quality of software. Within this book, we provide a good overview of several approaches that can prove beneficial in finding and correcting security flaws

In this section performing at three types associated with controls: developmental, operating method, and administrative. We go over each in return.

Developmental Controls

Many controls can end up being applied during software enhancement to ferret out in addition to fix problems. So allow us begin by searching at the size regarding development itself, to discover what tasks are engaged in specifying, designing, making, and testing software.

The Nature of Software Development

IDOL

Software development is usually considered solo effort; the programmer sits with a new specification or design in addition to grinds out line following line of code. But also in fact, software development is really a collaborative effort, involving individuals with different skill sets that combine their expertise to make a working product. Development needs people who can

designate the machine, by capturing the particular requirements and building a new model of how typically the system should work by the users' point involving view

design the method, by proposing an answer to the problem referred to by the requirements plus creating a model involving the solution

implement the machine, by using the style as a blueprint with regard to building a working option test the system, in order to ensure that it fits the requirements and accessories the solution as named for within the design

evaluation the system at numerous stages, to make positive that the final products are generally consistent with the requirements and design models

record the device, so that customers can be trained plus supported

manage the machine, to be able to estimate what resources can be needed for advancement also to track when the particular system is going to be done

preserve the system, tracking difficulties found, changes needed, in addition to changes made, and analyzing their effects on total quality and efficiency

One particular person could do each one of these things. But more usually than not knowing, some sort of team of developers performs together to execute these duties. Sometimes a team associate does more than 1 activity; a tester could take part in the requirements review, for example, or an implementer can certainly write documentation. Each staff is different, and staff dynamics play a significant role in the team's success.

Keep in head the kinds of superior attacks described in the particular previous section. Balfanzreminds us all that we must design and style systems that are equally secure and usable, advocating these points:

You still cannot retrofit usable security.

Resources aren't solutions.

Mind the particular upper layers.

Keep your buyers satisfied.

Think locally; work locally.

We can analyze product and process to view how both contribute to be able to quality and in specific to security as being an element of quality. Let people start out with the product, to be able to get a sense teaching how we recognize superior quality secure software.

Modularity, Encapsulation, and also the exact product information Hiding

Program code usually includes a long shelf-life and is enhanced more than time as needs alter and faults are found out and stuck. For this kind of reason, a key basic principle society engineering is to be able to create a design or perhaps code in small, plus self-contained units, called parts or modules; when a strategy is written this way, many of us declare it is flip. Modularity offers advantages with regard to program development generally in addition to security in particular.

In case a component is isolated through the effects of other pieces, it is easier to be able to trace a problem for the fault that caused that and to limit the particular damage the fault reasons. It is also much easier to maintain the system, considering that becomes an isolated part never affect other elements. Plus its easier to discover where vulnerabilities may lay if the component will be isolated. We call this kind of isolation encapsulation.

Information hiding is another characteristic involving modular software. When info is hidden, each part hides its precise rendering or some other style and design decision from the some others. Thus, each time a change is certainly needed, the overall style can remain intact whilst only the necessary modifications are created to particular components.

Allow us take a look at these attributes in more detail.

Modularity

Modularization is the method of dividing a process into subtasks. This section is done on some sort of logical or functional base. Each component performs a new separate, independent part associated with the task. Modularity is usually depicted in Figure 3.16. The goal is in order to have each component fulfill four conditions:

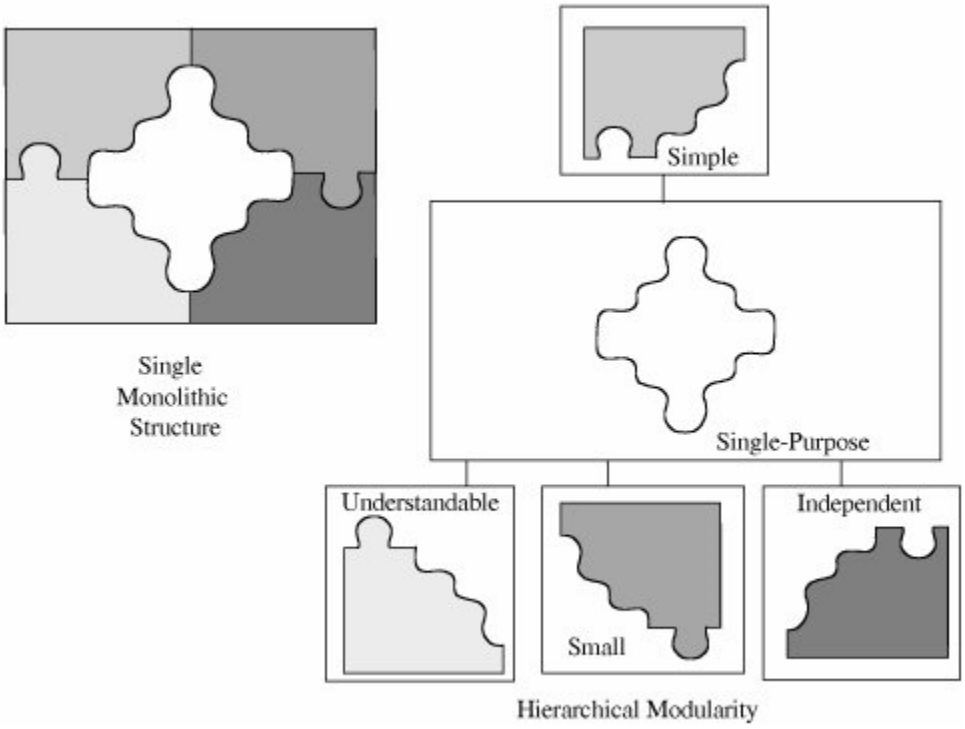


Figure 3.16. Modularity

single-purpose: functions one function

small: is composed of some information intended for which a human could readily grasp both framework and content

simple: benefits a low degree involving complexity in order that a man can readily be well known with purpoeand composition with the module independent: works a job isolated through other modules

Other element characteristics, such as possessing a single input and one output or using the limited set of encoding constructs, indicate modularity. By a security standpoint, modularity should improve the chance that an implementation is usually correct.

Particularly, smallness is definitely an important quality that will help security analysts understand exactly what each component does. Of which is, in good software program, design and program devices should be only since large as required to execute their required functions. Presently there are several advantages in order to having small, and 3rd party components.

Maintenance. In case a part implements a single operate, it can be substituted easily which has a revised 1 if necessary. The newest element may be needed because of to a change throughout requirements, hardware, or atmosphere. Sometimes the replacement is definitely an enhancement, using the smaller, faster, more appropriate, or else better module. Typically the interfaces between this element plus the remainder of typically the design or code are usually few and well defined, so the associated along with the replacement are noticeable.

Understandability. A system consisting of many small elements is usually simpler to know than one large, unstructured block of code.

Recycling. Components developed for starters aim can often be used again consist of systems. Recycle of correct, existing design and style or code components can easily significantly lower the difficulty associated with implementation and testing.

Correctness. A failure could be rapidly traced to its trigger if the components conduct only one task every.

Testing. A single aspect with well-defined inputs, results, and function could be examined exhaustively by itself, without having concern for its outcomes on other modules (other compared to expected function and even output, of course).

Safety analysts must be capable to understand each element as an independent device and be assured associated with its limited effect in other components.

A flip component usually has large cohesion and low joining. By cohesion, we result in that all the components of an element have a new logical and functional purpose for being there; each factor of the part is associated with the component's single purpose. A remarkably cohesive component contains a substantial degree of focus upon the reason; a low level of cohesion signifies that

the particular component's contents is surely an not related jumble of actions, frequently put together because involving time-dependencies or convenience.

Joining appertains to the education with which an aspect is determined by other components throughout the system. Thus, reduced or loose coupling is definitely better than high or perhaps tight coupling because the particular loosely coupled components are really free from unwitting disturbance from other components. This specific difference in coupling is usually shown in Figure 3.17.

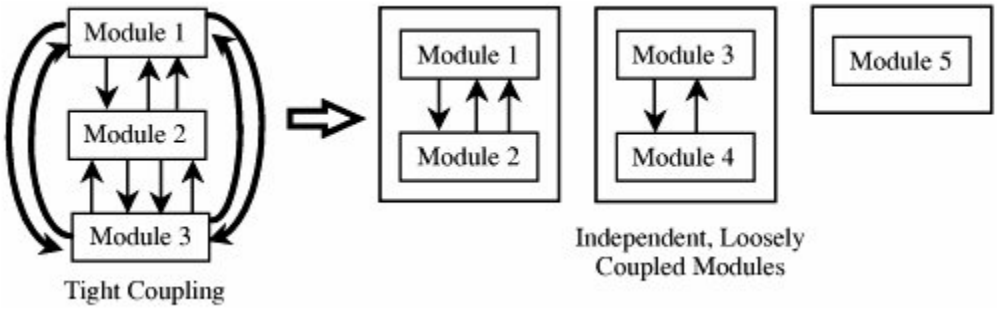


Figure 3.17. Coupling.

Encapsulation

Encapsulation hides the component's implementation details, although it does not automatically mean complete isolation. A lot of components must share details with other components, generally with good reason. Nevertheless , this sharing is thoroughly documented so that some sort of component is affected just in known ways simply by others inside the system. Discussing is minimized so of which the fewest interfaces potential are used. Limited terme lower the number of concealed channels that could be constructed.

A good encapsulated component's protective border can be translucent or perhaps transparent, as needed. Berardpaperwork that encapsulation could be the "technique for packaging the info [inside a component] in such an approach as to hide just what should be hidden make visible what is designed to be visible. very well.

Information Hiding

Developers who else work where modularization is definitely stressed can be certain that other components may have limited effect upon the ones they

create. Thus, we can believe of a factor as the kind of black field, with certain well-defined plugs and outputs and some sort of well-defined function. Other components' designers do not want to know how the particular module completes its perform; it really is enough to get assured that the aspect performs its task in certain correct manner.

This concealment is the information concealing, depicted in Figure 3.18. Information hiding is appealing because developers cannot very easily and maliciously alter typically the components of others when they do not recognize how the components does job.

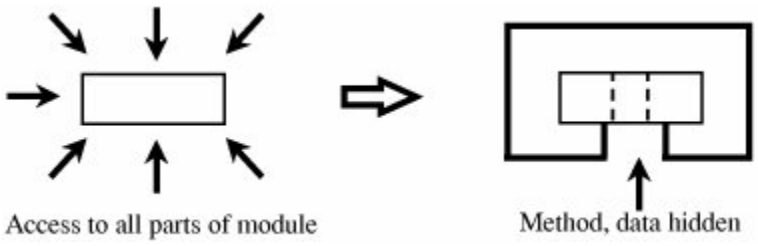


Figure 3.18. Information Hiding.

These kinds of three characteristicsmodularity, encapsulation, and even also the precise item information hidingare fundamental guidelines society engineering. They will be also good security methods because they lead in order to modules that can get understood, analyzed, and relied on.

Mutual Suspicion

Programs are usually not always trustworthy. In spite of an operating system to be able to enforce access limitations, that may be impossible or perhaps infeasible to bound typically the access privileges of a great untested program effectively. Within this case, an individual Circumstance is legitimately worried concerning a new program S. Yet , program P might be invoked by one more program, Q. There is usually no way for Queen that P is right or proper, any even more than an user perceives that of P.

Consequently, we use the principle of mutual suspicion to be able to describe the relationship in between two programs. Mutually shady programs operate as when other routines in the particular system were malicious or even incorrect. A calling plan cannot trust its referred to as subprocedures to be right, and a called

subprocedure cannot trust its phone program to be appropriate. Each protects its program data so the other provides only limited access. Regarding example, a procedure to be able to sort the entries in the list cannot be trustworthy never to modify those components, while that procedure are not able to trust its caller in order to provide any list from all or to offer you the number associated with elements predicted.

Confinement

Confinement is a technique applied by an operating program on a suspected software. A confined program will be strictly limited in exactly what system resources it can certainly access. If the program will be not trustworthy, the information that can access are firmly limited. Strong confinement can be helpful in limiting the particular spread of viruses. Due to the fact a virus spreads simply by means of transitivity and even shared data, every one of the information and programs in a solitary compartment of a limited program can affect just the data and plans in the same area. Therefore, the virus can certainly spread only to items for the reason that compartment; it are unable to get outside the area

Genetic Diversity

At the local electronics shop an individual can buy a blend printerscannercopierfax machine. It arrives at a good cost (compared to costs in the four separate components) since there is considerable overlap in operation among those four. This is compact, and an individual need only install 1 thing on your program, not four. But in the event that any part of this fails, you lose a new lot of capabilities almost all at once.

Related in order to the argument for modularity and information hiding and even reuse or interchangeability regarding software components, some folks recommend genetic diversity: that is risky having a lot of components of something take place from one source, i have heard it said.

Geer at al. wrote a written report examining the monoculture regarding computing dominated by one particular manufacturer: Microsoft today, APPLE yesterday, unknown tomorrow. These people are at the similar in agriculture where the entire crop is susceptible to a single virus. Malicious code from the particular Morris worm to typically the Code Red virus has been especially

harmful just because an important proportion of the earth's computers ran versions involving the identical operating

Tight integration involving products is a comparable concern. The Windows running system is tightly connected to Internet Explorer, any office Suite, and the View e-mail handler. A weeknesses in a of these could also affect the some others. Because of the small integration, fixing a weakness in one will surely have a great impact on the other people, whereas a vulnerability inside another vendor's browser, with regard to example, can affect Expression only to the level they communicate through a new well-defined interface.

Peer Critiques

We turn next in order to the process of growing software. Certain practices in addition to techniques can assist all of us in finding real and even potential security flaws (as well as other faults) and fixing them prior to we turn the device more than to the users. If recommend several major tactics for building what that they call "solid software":

expert reviews

hazard analysis

assessment

good design and style

prediction

permanent analysis

configuration management

evaluation of errors

Here, all of us look at each training briefly, and we explain its relevance to safety measures controls. We begin using peer reviews.

You have got probably been doing many type of review for since many years as a person have been writing signal: desk-checking your work or even asking a colleague in order to look more than a routine to be able to ferret out any troubles. Today, an application assessment is associated with a number of formal

IDOL

process steps to be able to make it more efficient, and we review just about any artifact of the enhancement process, not just signal. But the essence associated with a review remains identical: sharing a product together with colleagues able to remark about its correctness. Generally there are careful distinctions amongst three types of expert reviews:

Review: The creature is presented informally to some team of reviewers; typically the goal is consensus plus buy-in before development earnings further.

Walk-through: The creature is presented to typically the team by its originator, who leads and handles the topic. Here, schooling is the goal, plus the focus is on learning about a solitary document.

Inspection: This a lot more formal process is some sort of detailed analysis when the creature is checked against a new prepared list of problems. The creator does not necessarily lead the discussion, in addition to the fault identification plus correction are often manipulated by statistical measurements.

An intelligent engineer who finds some sort of fault can deal using it in at minimum three ways:

by understanding how, when, and exactly why errors occur

by using activity to prevent mistakes

by simply scrutinizing products to get the instances and results of errors that have been skipped

Expert reviews address this difficulty directly. Unfortunately, many agencies give only lip assistance to peer review, plus reviews continue to be not portion of mainstream software anatomist activities.

But you will find persuasive reasons to do testimonials. An overwhelming amount involving evidence shows that several types of peer overview in software engineering may be extraordinarily effective. Regarding example, early studies with Hewlett-Packard in the nineteen eighties revealed that those designers performing peer review in their projects enjoyed a new significant advantage over all those relying only on standard dynamic testing techniques, regardless of whether black box or white colored box. Figure 3.19 even comes close to the fault discovery charge (that is, faults found out per hour) among white-box testing,

black-box testing, home inspections, and software execution. It's clear that inspections learned far more faults throughout the same period regarding time than other choices. This end result is especially compelling for big, secure systems, where live life running for fault finding is probably not an option

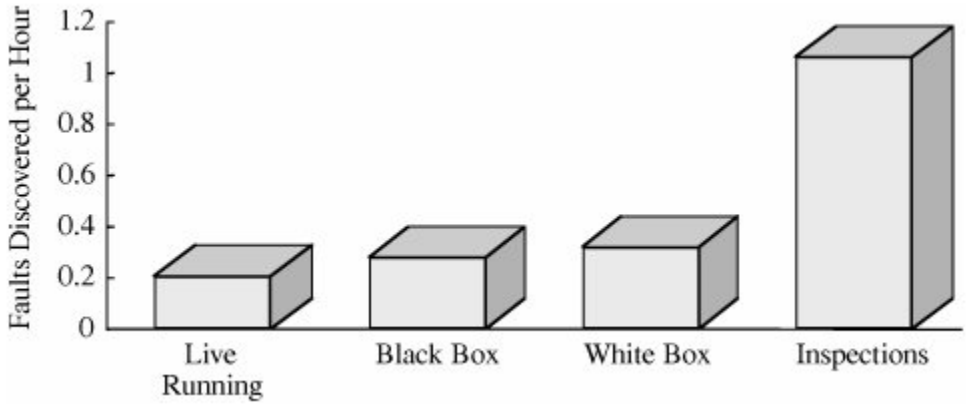


Figure 3.19. Fault Discovery Rate Reported at Hewlett-Packard.

Scientists and practitioners have consistently shown potency involving reviews. For instance, Smith described the info in his significant repository of project details to paint an image of how reviews in addition to inspections find faults within accordance with other finding activities. Because products change so wildly by dimensions, Table 3-6 presents typically the fault discovery rates comparative to the number associated with 1000s of lines of signal inside the delivered product.

Discovery Activity	**Faults Found (Per Thousand Lines of Code)**
Requirements review	2.5
Design review	5
Code inspection	10
Integration test	3
Acceptance test	2

Hazard Analysis

Hazard analysis is definitely a pair of systematic techniques supposed to expose potentially unsafe system states. In specific, it can help people expose security concerns and even then identify prevention or even mitigation ways of address these people. That is, hazard examination ferrets out likely will cause of problems so that will we could then apply the appropriate way of stopping the problem or treatment its likely consequences. As a result, it usually involves establishing hazard lists, as okay as procedures for discovering "what if" scenarios in order to trigger consideration of nonobvious hazards. The sources regarding problems can be hiding in any artifacts associated with the development or servicing process, not merely in the particular code, so a danger analysis must be wide-ranging in its domain involving investigation; in other words and phrases, hazard analysis is really a technique issue, not just a new code issue. Similarly, generally there are many types of issues, ranging from incorrect program code to unclear consequences of the particular action. A very good hazard analysis takes almost all of them into mind.

Though hazard analysis is mostly fine practice on any job, it is required inside some regulated and essential application domains, and this can be invaluable with regard to locating security flaws. This is never too early on to be thinking concerning the types of hazards; typically the analysis must start when a person first start thinking concerning creating a new program or when someone offers a significant upgrade to be able to an existing system. Risk analysis should continue over the system life cycle; you have to identify potential hazards that may be introduced during system design and style, installation, operation, and preservation.

Many different techniques support the particular identification and management associated with potential hazards. Among typically the most effective are risk and operability studies (HAZOP), failure modes and results analysis (FMEA), and wrong doing tree analysis (FTA). HAZOP is a structured research technique originally developed for that process control and substance plant industries. Over the particular last several years it offers been adapted to uncover potential hazards in safety-critical software systems. FMEA will be a bottom-up technique utilized at the system element level. A team pinpoints each component's possible flaws or

fault modes; the particular team then determines may trigger the fault and even exactly what systemwide results each fault might have got. By keeping system implications in mind, the group often finds possible method failures which are not really made visible by some other analytical means. FTA matches FMEA. It is a new top-down technique that commences with a postulated harmful system malfunction. Then, typically the FTA team works in reverse to identify the probable precursors to the accident. By tracing back coming from a specific hazardous failure, the team can identify unexpected contributors to incidents, and can then search for opportunities to mitigate typically the risks.

These techniques is definitely clearly great for finding in addition to preventing security breaches. We all decide which strategy is usually most appropriate by knowing how much we find out about causes and results. For example, Table 3.7 suggests that when all of us know the cause in addition to effect of a provided problem, we can enhance the description of precisely how the system should react. This clearer picture will assist requirements analysts understand just how any problem is associated to other requirements. This also helps designers recognize exactly what the technique should do helping testers know how to check to verify that typically the system is behaving appropriately. If we can explain a known effect using unknown cause, we make use of deductive techniques such while fault tree analysis in order to help us understand the particular likely causes of the particular unwelcome behavior. Conversely, we might know the cause involving a problem but is not recognize all the effects; in this article, we use inductive strategies such as failure ways and effects analysis in order to help us trace through cause to all or any possible results. For example, suppose all of us know that a subsystem is unprotected and may lead to securities failing, but we do not necessarily understand how that failure can impact the rest involving the system. We can easily use FMEA to produce a list of achievable effects and then assess the tradeoffs between more protection and possible troubles. Finally, to get problems in relation to which organic beef not but be aware, we will perform an exploratory evaluation, for example, a hazard and operability study.

IDOL

	Known Cause	Unknown Cause
Known effect	Description of system behavior	Deductive analysis, including fault tree analysis
Unknown effect	Inductive analysis, including failure modes and effects analysis studies	Exploratory analysis, including hazard and operability

Table 3-7. Perspectives for Hazard Analysis

IDOL

Testing

Testing is a method activity that homes found in on product quality: building the product failure no cost or failure tolerant. Every software problem (especially any time it relates to security) has the potential not really only to make software are unsuccessful but also for detrimentally affecting a business or perhaps a life. Thomas Little, head of NASA's research of the Mars lander failure, noted that "One of the things many of us kept in mind in the course of the course of each of our review is that inside of the conduct of place missions, you get just one strike, not three. Even when thousands of functions will be performed flawlessly, just 1 mistake may be catastrophic in order to a mission" . This same sentiment applies for security: The malfunction of one control presents a vulnerability that is definitely not ameliorated by any kind of number of functioning adjustments. Testers improve software top quality by finding as several faults as you possibly can and by simply writing up their studies carefully so that programmers can locate the leads to and repair the troubles if possible.

Tend not to dismiss a point from Thompson : Security assessment is hard. Side outcomes, dependencies, unpredictable users, in addition to flawed implementation bases (languages, compilers, infrastructure) all bring about to this difficulty. Although the essential complication along with security testing is of which we cannot look in just the behavior the program gets correct; we also have to be able to look for the 100s of ways the plan might go wrong.

Screening usually involves several levels. First, each program element is tested on the own, isolated from the particular other components in typically the system. Such testing acknowledged as module testing, aspect testing, or unit examining, verifies that the element functions properly with typically the sorts of input expected by a study of typically the component's design. **Unit testing** is done in some sort of controlled environment whenever achievable so that the check team can feed the predetermined set of files for the component being examined and observe what end result actions and data will be produced. In addition, the particular test team checks typically the internal data structures, reasoning, and boundary conditions for that input and output info.

When collections of elements have been put through product testing, the next stage is making certain the barrière among the components are usually defined and handled effectively. Indeed, interface mismatch may be a significant safety vulnerability. **Integration testing** is definitely the procedure for verifying that will the system components communicate as described in the particular system and program design and style specifications.

Once we will be sure that information is definitely passed among components inside accordance with the style, we test the program to ensure that that has the required functionality. The **function test** evaluates the particular system to determine regardless of whether the functions described simply by the requirements specification will be actually performed by the particular integrated system. The end result is a functioning program.

The function test comes anywhere close to the system being constructed with the functions described within the developers' requirements specification. And then, a **performance test** comes anywhere close the system with typically the remainder of the software in addition to hardware requirements. It is definitely

during the function and gratification tests that security demands are examined, and the particular testers confirm that typically the system is as protected as it is needed to be.

Once the functionality test is complete, builders are certain that the device functions according to their particular comprehension of the system explanation. The next step is definitely conferring with the buyer to make certain that will the device works according in order to customer expectations. Developers sign up for the customer to accomplish a good **acceptance test** when the technique is checked against typically the customer's requirements description. On completing acceptance testing, typically the accepted system is set up in the environment inside which it will end up being used. One final **installation test**is set you back again make sure that the particular system still functions since it should. However, safety requirements often state that will a process should not perform something

The goal of device and integration testing is usually to ensure that the particular code implemented the design and style properly; that is certainly, that the particular programmers have written codes to do what the particular designers intended. System assessment includes a very different goal: to ensure that the machine does what the consumer wants to carry out. Regression testing, an element of system testing, is certainly particularly important for safety measures purposes. After a transform is made to boost the system or fix a challenge, **regression testing** ensures of which all remaining functions happen to be still working and that will performance has not recently been degraded by the alter.

Each of the sorts of tests listed here can easily be performed from 2 perspectives: black box and even clear box (sometimes named white box). **Black-box testing** treats a system or perhaps its components as dark-colored boxes; testers cannot "see inside" the system, thus they apply particular plugs and verify that they will get the expected result. **Clear-box testing** allows presence. Here, testers can look at the design and signal directly, generating test situations using the code's actual design. Thus, clear-box testing has learned that component X employs CASE statements and may look for instances when the input causes control to drop through to a sudden line. Black-box testing needs to rely on read more regarding

the required inputs in addition to outputs because the real code is just not available regarding scrutiny.

The particular combo of techniques correct for testing a provided system depends on typically the system's size, application website, quantity of risk, and several other factors. But comprehending the effectiveness of each and every technique helps us recognize what is correct intended for each particular system. With regard to instance, Olsen describes the advancement at Contel IPC regarding a system containing 184, 000 lines of signal. He tracked faults uncovered during various activities, plus found differences:

17. 3 or more percent of the flaws were found during assessments with the system design

19. 1% during component design and style examination

15. 1 % during code inspection

30. 4 percent during the usage testing

16. 6 per cent during system and regression test

Only 0.1 percent of the errors were revealed after the particular system was placed inside the field. Thus, Olsen's work shows the value of using different methods to uncover different varieties of faults during enhancement; it is not adequate to count on a solitary method for catching just about all problems.

Who does the particular testing? From the security point of view, **independent testing** is very desirable; it may stop a developer from trying to hide something in a new routine or keep some sort of subsystem from controlling typically the tests that is applied in order to it. Thus, independent screening increases the likelihood a test will expose typically the result of an invisible feature.

An example regarding a testing is exclusive to computer security: **penetration testing**. Within this form involving testing, testers specifically attempt to make software fall short. That is, instead regarding testing to find out that application does do what this is expected to (as could be the goal in the particular other types of screening we just listed), the particular testers try to notice if the software truly does what it is certainly not intended to, which is definitely to fail or, especially,

fail to enforce safety. Because penetration testing normally pertains to full systems, not really individual applications,

Good Design

We saw earlier found in this chapter that modularity, information hiding, and encapsulation are characteristics of very good design. Several design-related procedure activities are particularly interesting building secure software:

by using a philosophy of fault threshold

having a consistent coverage for handling failures

recording the style rationale and background

using design patterns

All of us describe each of these types of activities in turn.

Developers should try to assume faults and handle all of them in manners that lessen disruption and maximize security and security. Ideally, we wish our system to get fault free. But throughout reality, we must think about the system will fall short, and that we make sure that will unexpected failure will not take the system down, ruin data, or destroy living. For example, rather compared to waiting for the device in order to fail (called passive wrong doing detection), we might produce the device so that that reacts in an satisfactory way to a failure's occurrence. Active fault recognition could possibly be practiced by, regarding instance, adopting a viewpoint of mutual suspicion. Rather of assuming that information passed from other devices or components are proper, we are able to always check that will the data are in bounds and of the correct type or format. We are able to also use redundancy, contrasting the outcome of two or even more processes to determine that will they agree, before we all use their result within a task.

If improving a fault is also risky, inconvenient, or costly, we can choose as an alternative to train fault tolerance: separating destruction caused by the particular fault and minimizing dysfunction to users. Although mistake tolerance is just not always consideration of as a safety technique, it supports the particular

IDOL

idea, discussed in Part 8, our security plan allows us to elect to mitigate the effects regarding a security problem as an alternative of preventing it. Regarding instance, rather than mount expensive security controls, all of us may choose to acknowledge the risk that essential data may be dangerous. If actually a safety measures fault destroys important information, we may decide to be able to isolate the damaged info set and automatically go back to some backup data set in place to ensure that users can carry on to perform system features.

More generally, we are able to design and style or code defensively, simply as we drive defensively, by constructing a regular policy for handling disappointments. Typically, failures include

faltering to realise a service

providing the particular wrong service or info

corrupting information

We can easily build into the design and style a particular way regarding handling each problem, choosing from one of about three ways:

Retrying: restoring the machine to its previous point out and performing the services again, using a diverse technique

Correcting: restoring the particular system to its earlier state, correcting some technique characteristic, and performing typically the service again, using the particular same strategy

Reporting: repairing the system to it is previous state, reporting the condition to an error-handling part, but not providing the services once more

This consistency involving design helps us look at for security vulnerabilities; many of us look for instances of which are different from typically the standard approach.

Design rationales and history tell individuals the reasons the technique is made one way rather of another. Such data helps us since the technique evolves, so we can easily integrate the design associated with our security functions with no compromising the integrity involving the system's overall style.

IDOL

Moreover, the design historical past enables us to appearance for patterns, noting exactly what designs work best through which situations. For example, we all can reuse patterns of which have been successful throughout preventing buffer overflows, inside ensuring data integrity, or even in implementing user username and password checks.

Prediction

Among the many varieties of prediction we carry out during software development, many of us try to predict the hazards involved in building in addition to using the program. As many of us see in depth throughout Chapter 8, we should postulate which unwelcome activities might occur and in that case make plans to stop all of them or at least offset their effects. Risk conjecture and management are specifically important for security, in which we are always coping with unwanted events that possess negative consequences. Our intutions help us decide which often controls to make use of and exactly how many. For example, anytime we think the danger of a particular protection breach is small, organic beef not want to spend a large amount regarding money, time, or energy in installing sophisticated settings. Or we may employ the likely risk influence to justify using various controls at once, a strategy called "defense in level. inch

Static Analysis

Just before a process is up and working, we could examine its style and code to find and repair security faults. We noted earlier that will the peer review procedure involves this kind regarding scrutiny. But static evaluation is somewhat more than peer evaluation, in fact it is usually performed just before peer review. We will use tools and strategies to examine you will involving design and code to be able to see if the features warn us of achievable faults lurking within. Regarding example, a large range of amounts of nesting may well indicate how the design or perhaps code is not easy to go through and understand, so that it is simple for a malicious creator to bury dangerous computer code deep within the method.

For this end, we could examine several aspects regarding the design and signal:

control flow structure

files flow structure

data composition

The control flow could be the sequence in which guidelines are executed, including iterations and loops. This element of design or signal can also show just how often a particular coaching or routine is carried out.

Data flow follows the particular trail of an info item since it will be accessed and modified by system. Many times, purchases put on data are structure, and that we use data stream measures to show people how then when each files item is written, study, and changed.

The files structure is the method by which the data are structured, in addition to the particular system itself. For illustration, in the event the data are set up as lists, stacks, or even queues, the algorithms intended for manipulating them are very likely to be well realized and well defined.

There are various approaches to static research, especially because there will be so many ways to be able to create and document some sort of design or program. Computerized tools are available to be able to generate not only amounts (such as depth associated with nesting or cyclomatic number) but also graphical depictions of control flow, information relationships, and the range of paths from series of code to one more. These aids can support us observe how the flaw in one section of a system can influence other parts.

Configuration Administration

When we develop application, it is important to be able to know who is producing which changes to just what then when:

corrective changes: preserving control over the system's everyday features

adaptive changes: sustaining control of system changes

perfective changes: perfecting current acceptable features

preventive adjustments: preventing system performance by degrading to unacceptable degrees

We want a point associated with control over the computer software changes so that 1 change will not inadvertently unnecessary the effect of the prior change. And we desire to control what is usually a proliferation of various versions and

releases. Intended for instance, a product may well run using several different websites or in numerous different conditions, necessitating different code to be able to support the same operation. Configuration management is the particular process by which many of us control changes during advancement and maintenance, and it also presents several advantages in protection. In particular, configuration managing scrutinizes new and transformed code to ensure, amongst other things, that safety flaws have not recently been inserted, intentionally or inadvertently.

Four activities are included in configuration management:

construction identification

configuration control and even change management

configuration auditing

status accounting

Configuration identity sets up baselines that all other code is going to be compared after shifts are made. That may be, we all build and document a great inventory of all elements that comprise the program. The inventory includes certainly not only the code a person and your colleagues may well have created, but furthermore database management systems, thirdparty software, libraries, test instances, documents, and more. After that, we "freeze" the base and carefully control exactly what happens to it. Any time a change is suggested and made, it will be described when it comes to how the particular baseline changes.

Configuration handle and configuration management guarantee we can coordinate split, related versions. For instance, there might be closely related types of the system to implement on 16-bit and 32-bit processors. Three ways to be able to control the changes usually are separate files, deltas, and even conditional compilation. If we all use separate files, we all have different files intended for each release or variation. For example, we may possibly build an encryption program in two configurations: 1 that utilizes a short key element length, to comply using the law in selected countries, and another that will run on the long key. And then, version 1 may always be composed of components A2 through Ak and B1, while version 2 will be A1 through Ak in addition to B2, where

IDOL

B1 and even B2 do key duration. That is, the types are the same besides for the separate key element processing files.

Alternatively, all of us can designate a certain version as the key version of the system and even then define other types in terms of just what is different. The variation file, called a delta, contains editing commands to explain the ways to change the main version straight into the variation.

Lastly, we can do conditional compilation, whereby a solitary code component addresses just about all versions, counting on the compiler to determine which transactions to apply to which often versions. This approach appears appealing for security software because every one of the code seems in one place. Nevertheless, if the variations are incredibly complex, the code is quite difficult to read in addition to understand.

Once a construction management strategy is selected and applied, the program ought to be audited regularly. The configuration audit confirms how the baseline is complete and even accurate, that changes happen to be recorded, that recorded adjustments are made, and of which the actual software (that is, the software because used in the field) is reflected accurately inside the documents. Audits happen to be usually done by impartial parties taking one involving two approaches: reviewing just about every entry inside the baseline in addition to comparing it with typically the software in use or perhaps sampling from a bigger set just to verify compliance. For systems together with strict security constraints, the particular first approach is more suitable, but the second method may be more useful.

Finally, status accounting documents advice about the elements: where they originated in (for instance, purchased, reused, or even written from scratch), typically the current version, the switch history, and pending transformation requests.

All four pieces of activities are executed by way of a configuration and transformation control board, or CCB. The CCB contains reps from all organizations along with a vested fascination with typically the system, perhaps including clients, users, and developers. The particular board reviews all offered changes and approves adjustments based on need, design and style integrity, future plans with regard to the software, cost, and even more. The developers

putting into action and testing the modification work with a system librarian to control plus update relevant documents in addition to components; they also publish detailed documentation about typically the changes and test effects.

Configuration management offers 2 advantages to those associated with us with security worries: protecting against unintentional risks and guarding against harmful ones. Both goals are usually addressed when the construction management processes protect the particular integrity of programs plus documentation. Because changes take place only after explicit endorsement from the configuration management expert, all changes are in addition carefully evaluated for part effects. With configuration supervision, previous versions of courses are archived, so a new developer can retract a new faulty change when this is necessary.

Malicious adjustment is made very tough with a strong overview and configuration management method set up. In fact, , poor configuration control offers resulted in no less than one method failure; that sidebar furthermore confirms the principle regarding easiest penetration from Phase 1. Once an examined program is accepted with regard to inclusion in a program, the developer cannot go in to make smaller, and subtle changes, many of these as inserting trapdoors. Typically the developer has access to be able to the running production plan only through the CCB, whose members are conscious of such security removes.

Standards of Program Development

No software development firm worth its salt enables its developers to create code without notice in any kind of manner. The good software program development practices described previously in this chapter have got all been validated by simply many years of training. Although none is Brooks's mythical "silver bullet" that will guarantees program correctness, good quality, or security, they most add demonstrably to typically the strength of programs. Therefore, organizations prudently establish specifications for how programs are usually developed. Even advocates associated with agile methods, which provide developers a peculiar degree regarding flexibility and autonomy, inspire goal-directed behavior based upon earlier experience and past good results. Standards and guidelines can easily capture wisdom from past projects and boost the probability that the resulting

IDOL

method will be correct. Throughout addition, you want to ensure that will the systems we construct are reasonably simple to preserve and are compatible along with the systems with which in turn they interact.

We could exercise some degree associated with administrative control over application development by considering various kinds of standards or perhaps guidelines:

standards of style, including using specified style tools, languages, or techniques, using design diversity, plus devising strategies for mistake handling and fault threshold

standards of documentation, terminology, and coding style, which include the layout of code for the page, choices of titles of variables, and work with of recognized program set ups

standards of programming, like mandatory peer reviews, intermittent code audits for correctness, and compliance with criteria

standards of testing, this sort of as using program confirmation techniques, archiving test benefits for future reference, making use of independent testers, evaluating analyze thoroughness, and encouraging test out diversity

standards of setup management, to control entry to and changes involving stable or completed system units

Standardization improves the particular conditions under which just about all developers work by building a common framework in order that no one developer is usually indispensable. It also enables carryover from a single project in order to another; lessons learned about previous projects available regarding use by all within the next project. Standards, in addition, assist in maintenance, given that the maintenance team can easily find required information inside a well-organized program. Yet, we must take treatment that the standards perform not unnecessarily constrain the particular developers.

Firms concerned regarding security and committed in order to follow software development specifications often perform security audits. In a security review, an independent security analysis team arrives unannounced to check on each project's compliance together with standards and guidelines. These people

reviews requirements, designs, records, test data and ideas, and code. Knowing of which documents are routinely looked at, a developer is less likely to set suspicious code inside a component in typically the first place.

Process Standards

You have a couple of friends. Sonya is really well organized, she maintains lists of things you can do, the girl always knows how to find some sort of tool or who provides a specific book, and every thing is completed before it is usually needed. Dorrie, on the particular other hand, is a new mess. She can by no means find her algebra reserve, her desk has consequently many piles of paperwork you cannot see the particular top, and he or perhaps she seems to package with everything as being a problems because she is likely to overlook things until the last second. Who would you select to organize and operate a major social functionality, a new product release, or even a multiple-author paper? Many people would pick Sonya, concluding that her corporation skills are very essential. There is no promise that Sonya would carry out a better job as compared to Dorrie, but you may possibly assume the chances happen to be better with Sonya.

All of us know that software growth is difficult in portion since it has inherently individuals aspects that are quite difficult to judge beforehand. Still, we may consider that software built within an orderly manner provides a better potential for becoming good or secure.

The application Engineering Institute developed typically the ability Maturity Model (CMM) to evaluate organizations, not necessarily products. The International Criteria Organization (ISO) developed method standard ISO 9001, that is somewhat identical to the CMM. Lastly the U. S. Country wide Security Agency (NSA) designed the System Security Design CMM observe. Almost all of these are procedure models, in that they will examine how an firm does something, not precisely what it does. Thus, these people judge consistency, and a lot of folks extend consistency to top quality. For views on that will subject, see Bollinger and even McGowan and Curtis. El Emam has also viewed at the reliability associated with measuring a procedure.

Right now go back to typically the original descriptions of Sonya and Dorrie. Who might make the better creator? That question is complicated because many of people have friends like Dorrie who are fabulous coders, but we may furthermore

know great programmers that resemble Sonya. And several successful teams have each. Order, structure, and regularity can lead to good software jobs, nonetheless it is not sure to be able to be the only approach to go.

3.6 Review Question

1. Suppose you are a customs inspector. You are responsible for checking suitcases for secret compartments in which bulky items such as jewelry might be hidden. Describe the procedure you would follow to check for these compartments.

2. Your boss hands you a microprocessor and its technical reference manual. You are asked to check for undocumented features of the processor. Because of the number of possibilities, you cannot test every operation code with every combination of operands. Outline the strategy you would use to identify and characterize unpublicized operations.

3. Your boss hands you a computer program and its technical reference manual. You are asked to check for undocumented features of the program. How is this activity similar to the task of the previous exercises? How does it differ? Which is the most feasible? Why?

4. Could a computer program be used to automate testing for trapdoors? That is, could you design a computer program that, given the source or object version of another program and a suitable description, would reply Yes or No to show whether the program had any trapdoors? Explain your answer.

5. A program is written to compute the sum of the integers from 1 to 10. The programmer, well trained in reusability and maintainability, writes the program so that it computes the sum of the numbers from k to n. However, a team of security specialists scrutinizes the code. The team certifies that this program properly sets k to 1 and n to 10; therefore, the program is certified as being properly restricted in that it always operates on precisely the range 1 to 10. List different ways that this program can be sabotaged so that during execution it computes a different sum, such as 3 to 20.

DOI

6. One means of limiting the effect of an untrusted program is confinement: controlling what processes have access to the untrusted program and what access the program has to other processes and data. Explain how confinement would apply to the earlier example of the program that computes the sum of the integers 1 to 10.

7. List three controls that could be applied to detect or prevent salami attacks.

8. The distinction between a covert storage channel and a covert timing channel is not clear-cut. Every timing channel can be transformed into an equivalent storage channel. Explain how this transformation could be done.

9. List the limitations on the amount of information leaked per second through a covert channel in a multiaccess computing system.

10. An electronic mail system could be used to leak information. First, explain how the leakage could occur. Then, identify controls that could be applied to detect or prevent the leakage.

11. Modularity can have a negative as well as a positive effect. A program that is overmodularized performs its operations in very small modules, so a reader has trouble acquiring an overall perspective on what the system is trying to do. That is, although it may be easy to determine what individual modules do and what small groups of modules do, it is not easy to understand what they do in their entirety as a system. Suggest an approach that can be used during program development to provide this perspective.

12. You are given a program that purportedly manages a list of items through hash coding. The program is supposed to return the location of an item if the item is present or to return the location where the item should be inserted if the item is not in the list. Accompanying the program is a manual describing parameters such as the expected format of items in the table, the table size, and the specific calling sequence. You have only the object code of this program, not the source code. List the cases you would apply to test the correctness of the program's function.

IDOL

13. You are writing a procedure to add a node to a doubly linked list. The system on which this procedure is to be run is subject to periodic hardware failures. The list your program is to maintain is of great importance. Your program must ensure the integrity of the list, even if the machine fails in the middle of executing your procedure. Supply the individual statements you would use in your procedure to update the list. (Your list should be fewer than a dozen statements long.) Explain the effect of a machine failure after each instruction. Describe how you would revise this procedure so that it would restore the integrity of the basic list after a machine failure.

14. Explain how information in an access log could be used to identify the true identity of an impostor who has acquired unauthorized access to a computing system. Describe several different pieces of information in the log that could be combined to identify the impostor.

15. Several proposals have been made for a processor that could decrypt encrypted data and machine instructions and then execute the instructions on the data. The processor would then encrypt the results. How would such a processor be useful? What are the design requirements for such a processor?

1.7 References

1. Security in Computing, Fourth Edition By Charles P. Pfleeger - Pfleeger Consulting Group, Shari Lawrence Pfleeger - RAND Corporation Publisher: Prentice Hall

2. Cryptography and Network Security - Principles and Practice fifth edition Stallings William Publisher: Pearson

3. Cryptography And Network Security 3rd Edition behrouz a forouzan and debdeepmukhopadhyay 3/E Publisher: McGraw Hill Education

4. Cryptography and Network Security, 3e AtulKahate Publisher: McGraw Hill

IDOL

Chapter 4. Protection in General-Purpose Operating Systems

IDOL

4.0 Protection in General-Purpose Operating Systems

An OperatingSystem provides a couple of goals: managing shared access and implementing an interface to allow that access. Underneath those goals will be assist actions, incorporating identification and authentication, naming, filing objects, scheduling, communication among the processes, and reclaiming and reusing objects. Operating system characteristics consists of

access control
identity and credential management
information flow
audit as well as integrity protection

each one of these activities offers security ramifications. Operating systems vary from basic types assisting an individual job at any given time .this kind of operating system may operate a personal digital assistant to complex multiuser, multitasking systems, and, obviously, security factors help to increase when operating systems become more and more complicated.

We begin by studying the contributions of operating systems have made to positively user security. An operating system supports multiprogramming that is, the concurrent use of a system by more than one user, so operating system designers have developed ways to protect one user's computation from inadvertent or malicious interference by another user. Among all those features provided for this goal are memory protection, file protection, general control of usage of objects as well as user authentication. This chapter studies finally, the controls that provide these types of four features. We have oriented this discussion to the user: How do the controls safeguard users, and how can end users incorporate all those controls?

IDOL

4.1. Protected Objects and Methods of Protection

We begin by reviewing the history of protection in operating systems. This background helps us understand what kinds of things operating systems can protect and what methods are for sale to protecting them

A Bit of History

Previously, there were no os's: Users entered all their applications directly into the device in binary by way of switches. Most of the time, program entry was made by physical manipulation of the toggle switch; in various other instances, the entry was worked with a more complicated electronic method, by using an input device, for instance, a keyboard. Because each individual had exclusive usage of the computing program, users were needed to arrange blocks of time intended for running the device. These users were accountable for loading their personal libraries of support exercise routines assemblers, compilers, distributed subprogramsand "clearing up" after making use of by removing any susceptible code or data.

The first os's were simple utilities, known as executives, designed to help individual programmers and also to smooth the transition from user to another. Early executives supplied linkers and loaders intended for relocation, comfortable access to compilers and assemblers, and then automated loading of subprograms coming from libraries. The executives dealt with the tedious facets of programmer support, concentrating on one programmer during execution.

Operating systems had taken on a much wider role (and a distinct identity) as the concept of multiprogramming was first implemented. Seeing that two users can interleave use of the resources of an individual computing system, researchers designed concepts, for example, scheduling, posting, and parallel make use of. Multiprogrammed operating systems also called monitors, oversaw every program's execution. Monitors had taken an active role, while executives had been passive. That may be, an executive remained in the history, waiting for being called into provider by their requesting user. Yet a monitor definitely asserted

control of the processing system and provided resources towards the user only once the request was according to general good utilization of the system. Likewise, the executive waited to get a request and provided provider on demand; the monitor has taken care of control over almost all resources, allowing or denying most computing and loaning assets to end users as they required them.

Multiprogramming brought another change to computing. Each time one person was utilizing a system, the just force to be guarded against was the end user himself or herself. A person making a mistake could have experienced silly, still, one user cannot impact the computation of any kind of another user negatively. However, multiple users presented more difficulty and risk considerably. User A could properly become upset if Consumer B's applications or data developed a negative effect on A's program's functionality. Therefore, safeguarding one user's applications and data from various other users' programs are becoming an important issue in multiprogrammedos's.

4.1. Protected Objects and Methods of Protection

We begin by reviewing the history of protection in operating systems. This background helps us understand what kinds of things operating systems can protect and what methods are available for protecting them.

A Bit of History

Once upon a time, there were no operating systems: Users entered their programs directly into the machine in binary by means of switches. In many cases, program entry was done by physical manipulation of a toggle switch; in other cases, the entry was performed with a more complex electronic method, by means of an input device such as a keyboard. Because each user had exclusive use of the computing system, users were required to schedule blocks of time for running the machine. These users were responsible for loading their own libraries of support routine assembles, compilers, shared subprograms and "cleaning up" after use by removing any sensitive code or data.

The first operating systems were simple utilities, called **executives**, designed to assist individual programmers and to smooth the transition from one user to another. The early executives provided linkers and loaders for relocation, easy access to compilers and assemblers, and automatic loading of subprograms from libraries. The executives handled the tedious aspects of programmer support, focusing on a single programmer during execution.

Operating systems took on a much broader role (and a different name) as the notion of multiprogramming was implemented. Realizing that two users could interleave access to the resources of a single computing system, researchers developed concepts such as scheduling, sharing, and parallel use. **Multiprogrammed operating systems**, also known as **monitors**, oversaw each program's execution. Monitors took an active role, whereas executives were passive. That is, an executive stayed in the background, waiting to be called into service by a requesting user. But a monitor actively asserted control of the computing system and gave resources to the user only when the request was consistent with general good use of the system. Similarly, the executive waited for a request and provided service on demand; the monitor maintained control over all resources, permitting or denying all computing and loaning resources to users as they needed them.

Multiprogramming brought another important change to computing. When a single person was using a system, the only force to be protected against was the user himself or herself. A user making an error may have felt foolish, but one user could not adversely affect the computation of any other user. However, multiple users introduced more complexity and risk. User A might rightly be angry if User B's programs or data had a negative effect on A's program's execution. Thus, protecting one user's programs and data from other users' programs became an important issue in multiprogrammed operating systems.

Protected Objects

- In fact, the surge of multiprogramming led to many areas of a computing system needed protection:
- memory space
- sharable I/O equipment, for instance, disk drives

- reusable I/O equipment serially, for instance, computer printers and tape drives
- sharable subprocedures and applications
- networks
- sharable data

Because it presumed task designed for handled posting, the operating system had a need to safeguard such objects. In the next sections, we look at some of the mechanisms with which os's have forced these varieties of objects' protection. Various operating-system safeguard mechanisms have already been maintained hardware

Security Methods of Operating Systems

The foundation of protection is generally separation: staying one user's objects distinguish from all other end users.Rushby and as well, Randell mentioned that parting within an operating-system can take place in numerous ways:

· **physical separation,** wherein unique processes make use of several physical objects, such as separate printers meant for output needing several examples of protection

· **temporal separation**, wherein processes going through several security criteria will be performed at different times

· **logical separation**, wherein users work beneath the impression the fact that no additional processes are available, because when an operating-system constraint a program's accesses to the program is not able to gain access to objects outside their authorized domains

·**cryptographic separation**, wherein processes hide their data and computations in such a way that they can be unintelligible to help you outside processes

Obviously, mixtures of two or more of such kinds of separation can also be possible.The types of separation are outlined approximately in increasing order of difficulty toimplement, and, for the first three, in lowering order of the security supplied. On the other hand, the first two methods are extremely rigid and may

result in poor resource usage. As a result, we would like to transfer the duty of protection to the operating system to permit concurrent execution of processes having to cope with varied security requirements.

Nevertheless, separation is merely half the answer. We would like to different end users and their objects, but we also want to manage to offer to share for some of these objects. For example, two users with different protection, levels might want to employ precisely the same search algorithm or function call.
All of us wants users to be able to share the features and algorithms without diminishing their unique specific security needs. An operating-system support parting and sharing in various techniques, providing safeguard any type or form of time in many levels.

· **Do not protect.**Os's without security work when delicate methods are receiving operate in different situations.
· **Isolate.** When an operating-system gives isolation, several procedures operating aren't aware of the existence of one another simultaneously. Every procedure possesses its address space, files, and other objects also. The operating system has to restrict as a result every process for some reason the objects of various other processes will be obscured totally.
· **Share all or talk about almost nothing**. Due to this sort of security, who owns an object promises this to be personal or public. An open public object exists to everyone users, while a private object exists with their owner simply.
· **Share through access restriction.** Because of protection by gain access to constraint simply, the operating system inspections the allowability of each user's potential use of an object. That's, access control usually is applied for a specific eliminate consumer ıncluding a specific object. Prospect lists of suitable actions guide the operating-system in deciding in the event that the specified user will need to have use of a particular object. In a few sense, the operating system offers a shield among items and users, ensuring authorized has usage of happening exclusively.

IDOL

· **Share by features.** An extension of little access sharing, this kind of safeguard enables powerful developing of sharing privileges designed for objects. The known degree of sharing depends upon this owner or the topic may be, within the framework of the computation, or within the thing itself.

· **Limit usage of an object.** This kind of protection limitations not merely the access an object but also the utilize made of that object after it can often be accessed immediately. For example, a user may be given permission to be to see an extremely sensitive record, however, never to print a duplicate of it. More strongly even, a user could be given permission to end up being entry to data inside a database to assist you to get statistical summaries (for instance, average income at a particular grade level), however, never to ascertain particular data values (salaries of people).

4.2. Memory and Address Protection

The obvious issue in multiprogramming is generally protecting against one program from having an effect on the data and programs within the memory space of other users. However, protection could be constructed into the hardware mechanisms which usually restrain effective utilization of memory space, therefore solid protection could be offered at effectively no extra cost.

Fence

The easiest type of memory space protection was presented in single-user operating systems to avoid a defective user program from doing damage to the area of the resident section of the operating system. As the name indicates, a fence can be described as a strategy to restrict users to one side of the boundary. In a single implementation, the fence must have been a predetermined memory space address, allowing the operating system to reside in on a single side as well as the user to remain on the other. An illustration of this case is shown in Figure 4.1. However, this type of implementation was extremely limited just because a predetermined magnitude of space was in fact usually available to the operating system, regardless of it had been required or not. If lower than the predetermined space was needed, the extra space was misused. On the other

hand, if the operating system required extra space, it might not really expand beyond the fence boundary.

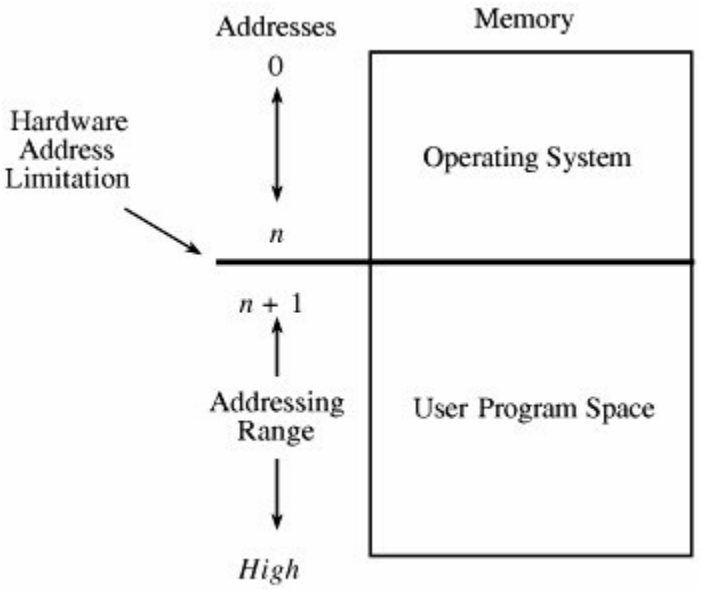


Figure 4.1. Fixed Fence

An additional implementation utilized a hardwar register, known as a fence register, including the address of the end of the operating system. Contrary to a fixed fence, from this scheme, the positioning of the fence could possibly be modified. Every time a user system produced an address intended for data alteration, the address was indeed immediately compared to the fence address. If the address was more than the fence address (that could be, within the user area), the instruction was performed; if it was, in fact, lower than the fence address (that could be, inside operating system place), the fault condition grew up. The utilization of fence registers is demonstrated in Figure 4.2.

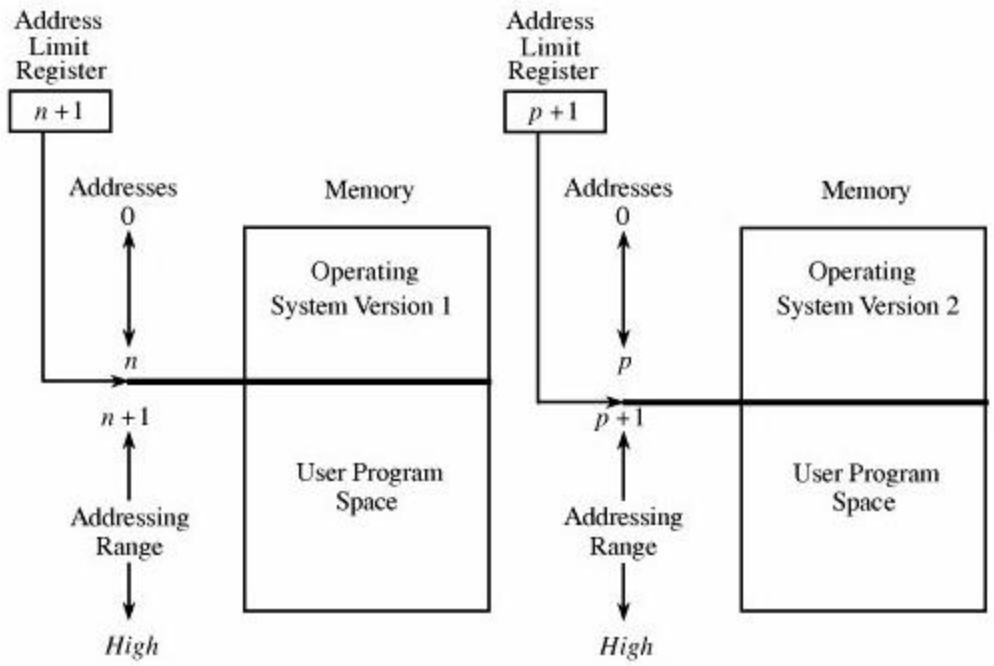


Figure 4.2. Variable Fence Register.

A fence register shields just in a single direction. To put it differently, an operating system could be safeguarded from an individual user, however the fence are not able to shield a single user from an additional user. In the same way, an user are not able to determine specific regions of this program as invulnerable (such as code in the program itself or maybe a read-only data area).

Relocation

In the event the operating system could be believed for being of the predetermined size, programmers may create their particular code believing that the program will begin at a constant address. This kind of feature of the operating system makes it simple to look for the address of any kind of object inside the program. On the other hand, it also makes the idea effectively difficult to improve the starting address if, as an illustration,

a fresh version of the operating system is usually larger or maybe smaller than the actual. Generally, if the size of the operating system can be permitted to modify, afterward programs should be developed in a fashion that will not be based upon positioning at any particular area in memory space.

Relocation is a procedure for having a program developed just as if it initiated at address 0 and then converting all addresses which will show the actual particular address from which the program is positioned in memory. In many cases, this kind of work simply requires putting in a consistent relocation factor with each address in the program. That is certainly, the relocation factor is the beginning address of the memory space allocated for the program.

Ideally, the fence register can be utilized from this situation to provide you an essential excessive advantage: The fence register could be a hardware relocation device. The details of the fence register will be placed into every program address. This course of action equally relocates the address and assurances that nobody can gain access to an area less than the fence address. (Addresses will be considered as unsigned integers, therefore conjoining the value in the fence register to every number is normally bound to build a result at or maybe over a fence address.) Special instructions could be added intended for a couple of occasions when a program legally expects to gain access to an area from the operating system.

Base/Bounds Registers

IDOL

A significant advantage of an operating program because of fence registers is a capacity to relocate; this sort of feature is vital in a multiuser environment particularly. Because of many customers, none of them can certainly understand in advance in which a program is going to be loaded designed for execution. The relocation register resolves nevertheless, the nagging problem giving a base or starting address. All the addresses within a scheduled plan will be offsets out of this base address. A variable fence register is normally termed as a base register.A variable fence register is normally described as a base register.

Fence registers provide a lower bound (a beginning address) however, no upper one. A higher bound can be helpful in understanding how very much space is normally allocated furthermore to examining for overflows into "forbidden" areas. To get over this sort of problems,To, another register is added, as proven in Figure 4-3. The next register, referred to as bounds register, is definitely a higher address limit, very much the same that the fence or base sign-up is actually a lower address limit. Every plan address will end up being over a bottom address because the contents of the bottom register will be put into the address; every address is additionally examined to ensure that it's beneath the bounds address truly. This way, a program's addresses will end up being properly limited to the region between the base and also the bounds registers.

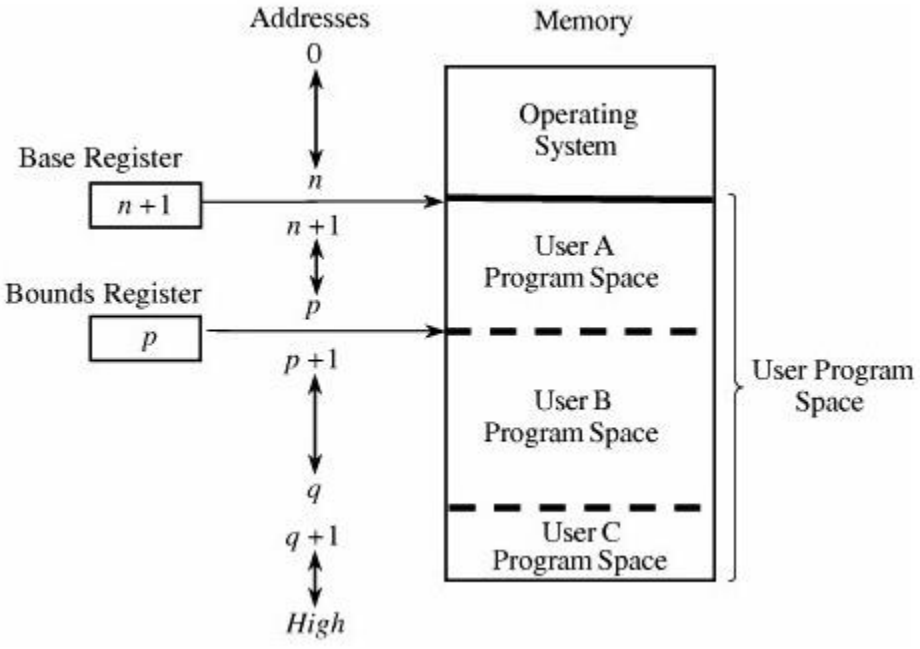


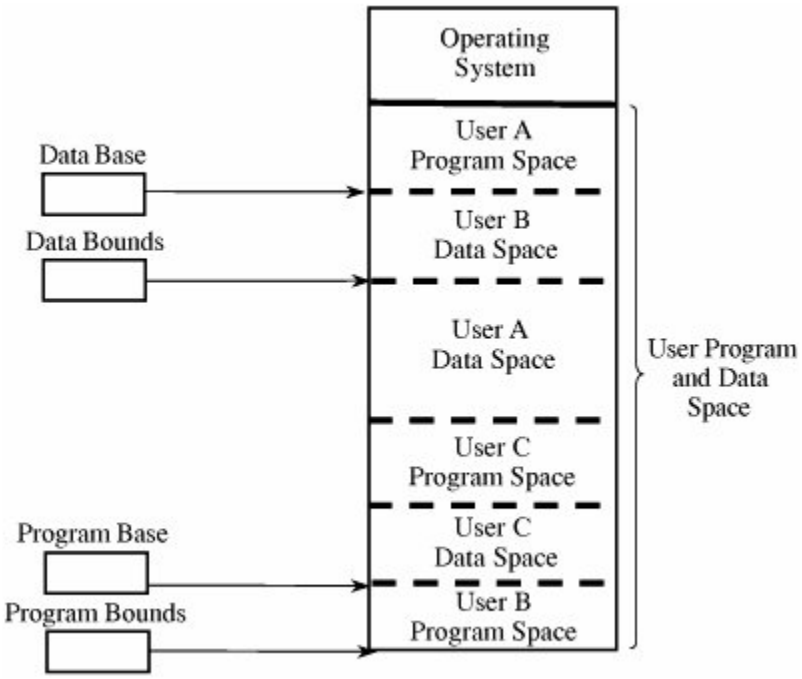
Figure 4-3. Pair of Base/Bounds Registers.

This method shields a program's addresses from changes simply by another end user. Once execution adjustments from a single user's program to another's, the operating system need to replace the contents of the base and bounds registers to reveal the actual address space for the user. This kind of change is usually section of the basic planning,This kind of change is section of the basic planning usually, known as context switch, the fact that operating system needs to execute once shifting control from one user to another.

Using a couple of base/bounds registers, users are usually effectively guarded right from outside users,users are effectively guarded right from outside users usually, or perhaps, even more properly, outside users will be guarded right from errors in different various user's program. Incorrect addresses within a user's address space could impact that may program since the base/bounds looking at assurances just that every address is usually within the user's address space. To illustrate, users mistake may happen each time a subscript beyond the range or maybe an undefined variable produces an address reference inside the user's space although, however, within the executable instructions of the user's program. In this way, a user may unintentionally store data on the top of instructions. This kind of error allows a user unintentionally damage an application, yet (luckily) only the user's own program.

We are able to resolve this kind of overwriting issue by utilizing an additional set of base/bounds registers, one particular intended for the instructions (code) from the program another for the data space. In that case, only instruction fetches (instructions for being executed) will be relocated as well as, examined along with the first register pair, and after that data accesses (operands of instructions) will be relocated and examined along with the second register pair. The usage of two

pairs of base/bounds registers is shown in Figure 4- 4. Even though two pairs of registers will not protect against all program errors, they will limit the effects of data-manipulating instructions towards the data space. The pairs of registers provide one more crucial advantage: the capability to divided a program into two pieces which can be relocated individually.



Tagged Architecture

An additional issue with using base/bounds registers to get relocation or protection is normally their particular contiguous characteristics. Every group of registers limitations has usage of to a successive collection of addresses. A compiler or loader can merely piece together an application to ensure that all code areas will end up being adjacent and all data areas will end up being adjacent.

However, in some instances, you might want to shield a couple of data values however, not virtually all. To illustrate, a workers record may need shielding this field designed for income however, not office get in touch with and location number. Furthermore, a programmer may want to ensure the ethics of particular data ideals by letting them finish up being

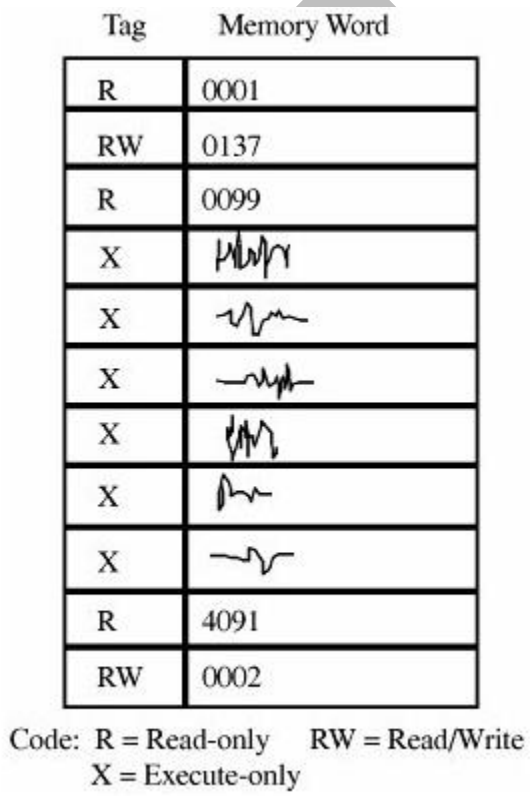
made if this program is normally initialized yet forbidding this program from changing them afterward. This types of program shields from errors in the programmer's very own code. A programmer could also make use of a distributed subprogram from the common library. We are able to address a few of these nagging problems by making usage of good style, both in the operating system and in the various other programs keeping operate. Recall that in Chapter 3 most of us learned great design features, for instance, details modularity and hiding in plan design. These varieties of characteristics determine that one program component have to tell another component the real minimal capability of data needed for both of them to execute their job.

Further, operating-system-specific style features shall help, too. Bottom/bounds registers generate an all-or-nothing scenario designed for sharing: Whether plan makes all its data open to become utilized and changed or it forbids access all. Whether there were a than band of registers designed for distributed data, all data will together have to be placed. A procedure could hardly share data items a, B, and C with one element, A, C, and D with another, and A, B, and D with a third. In order to to achieve the kind of sharing we desire is to maneuver every ideal band of data values for some contiguous space. However, this type or types of treatment might not be appropriate if the data items had been large records, arrays, or structures.

An alternate is tagged architecture, by which each and every word of machine storage offers several excessive bits to identify the access privileges compared to that word. These varieties of entryway bits could possibly be arranged by just privileged (operating-system) instructions. The bits will be tested each right time an instruction has usage of that location.

To illustrate, as proven in Figure 4.5, one storage area could be guarded seeing that execute-just (to illustrate, the thing code of guidelines), although another is normally guarded designed for fetch-only (one of these

is, read) data gain access to, and another attainable for adjustments (to illustrate, write). This way, two adjacent places might have different access privileges. Second of all, by extra tag bits, different classes of data (numeric, character, pointer or address, and undefined) could possibly be separated, and data areas could be covered for privileged (operating-system) access only.



Tag	Memory Word
R	0001
RW	0137
R	0099
X	
X	
X	
X	
X	
X	
R	4091
RW	0002

Code: R = Read-only RW = Read/Write
X = Execute-only

Figure 4.5. Example of Tagged Architecture.

This type or sort of safety technique has been applied to a few devices, though the level of tag bits have already been somewhat little also. The Burroughs B6500-7500 program used three tag bits to split data phrases (three types), descriptors (ideas), and control phrases (stack ideas and addressing control phrases). The IBM System/38 utilized a tag to control both access and integrity.

An alternative utilized one tag that positioned on a combined group of successive locations, such as for example 128 or 256 bytes. Because of one tag for the block of addresses, additional expense for applying tags was not as high much like the main one tag per area. The Intel I960 prolonged architecture processor chip used a tagged architecture using a little on each memory phrase which often designated the word such as a "capability," rather than as a typical location for guidelines or data. A capability managed the use of a variable-sized memory segment or block. This kind of many feasible tag ideals backed storage sections that ranged in proportions from 64 to 4 billion bytes, having a promising 2256 different security domains.

Compatibility condition of code proven problems together with the acceptance of a tagged architecture. A tagged architecture might not be as useful as more modern methods, as we shortly see. Some of the important pc vendors continue being working with operating systems which have been designed and applied in the past designed for architectures of this period. Certainly, most producers are locked towards an extra standard memory architecture because of this of large option of elements ıncluding a choose to preserve compatibility among os's and machine families. A tagged architecture may need important changes to consider all the operating system code, a necessity which can be really costly. But since the price of memory is constantly on the fall, the implementation of a tagged architecture turns into even more possible

Segmentation

We present a few additional methods to protection, every of which could be applied on the top of a standard machine framework, recommending a much better possibility of approval. Even though these types of methods are actually traditional by simply computing's criteria these were built somewhere between 1965 and 1975 they've been applied on various devices since that time. Secondly, they provide essential strengths in dealing with, with memory protection as being a wonderful reward.

The most important of those couple methods, segmentation, entails the easy idea of separating a program in to distinguish parts. Each part includes a logical unity, demonstrating a association of all of code or data values. By way of example, a segment could be the code of a single method, the data associated with an array,

or the number of all local data values utilised by a particular component. Segmentation was created as being a feasible way to create the result of the counterpart of the unbounded number of base/bounds registers. To put it differently, segmentation enables a program to be split up into various parts needing diverse access privileges.

Each area incorporates a particular name. A code or data items inside section is tended to as the pair <name, offset>, where name is the name of the fragment containing the information thing and balance is its area inside the portion (that is, its separation from the beginning of the fragment).

Intelligently, the software engineer pictures a program as a long gathering of fragments. Sections can be independently migrated, enabling any portion to be put in any accessible memory areas. The connection between a coherent portion and its actual memory position is appeared in Figure 4.6.

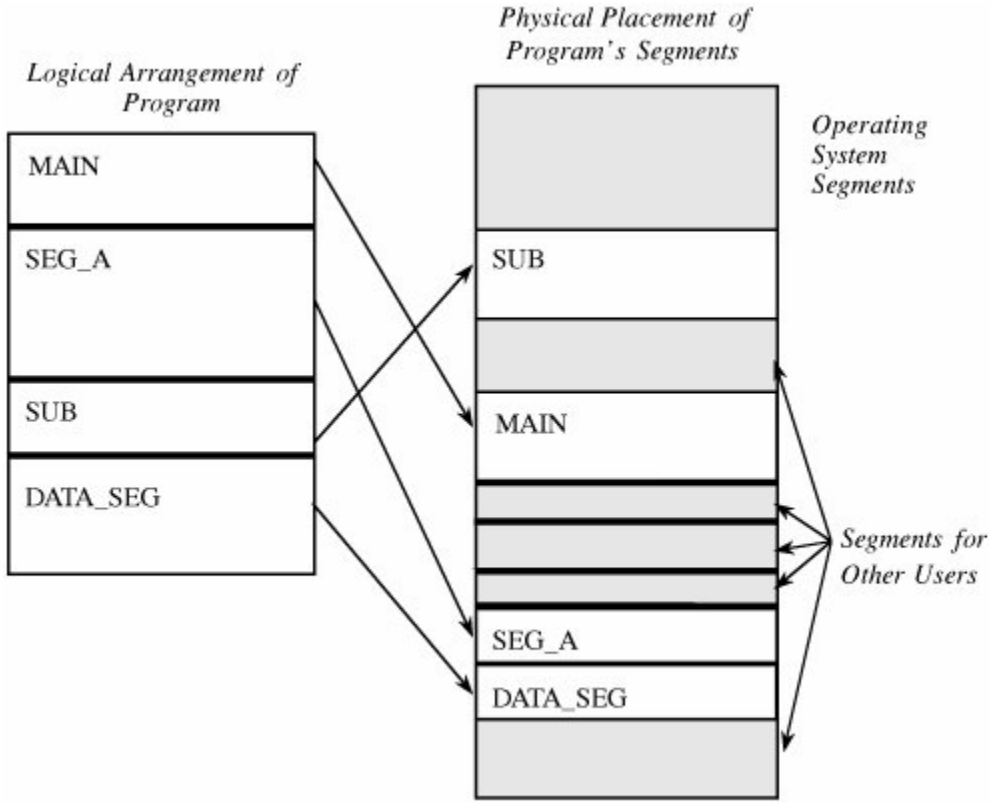


Figure 4.6. Logical and Physical Representation of Segments.

Consequently, a User's program does not realize what genuine memory tends to its employment. It has no way and no need to decide the accurate location related with a specific <name, offset>. The <name, offset> pair is satisfactory to get to any data or instruction to which a program ought to approach.

This concealing up of addresses has three positive aspects of working for the operating system.

the operating system can put any location at any area or move any location to any area, even after the program starts to execute. Since it converts all address references by a section address table, the operating system needs possibly update the location in that one table when a portion is moved.

A location can be expelled from primary memory (and put away on a helper gadget) on the off chance that it isn't being utilized presently.

Each location reference goes through the operating system, so there is a chance to check everyone for security.

Because of this last function, a process can get access to a phase simplest if that process seems in that process's phase translation table. The operating system controls which software have entries for a specific phase in their section deal with tables. This manage presents robust safety of segments from getting entry to with the aid of unpermitted tactics. For example, program A may have got right of entry to segments BLUE and GREEN of person X but now not too different segments of that user or of some other user. In a truthful way, we will permit a person to have distinctive protection instructions for one of a kind segments of an application. For instance, one phase might be read-best statistics, a second might be execute-best code, and a 3rd might be writeable data. In a state of affairs like this one, segmentation can approximate the intention of separate safety of various pieces of an application, as outlined in the preceding section on tagged architecture.

Segmentation offers those safety blessings:

Each address with reference is checked for safety.

Many different instructions of data objects may be assigned one of a kind degrees of protection.

Two or extra persons can be shared get right of entry to to a segment, with probably different access rights.

A person can not generate an address with or access to an unpermitted segment.

Paging

One opportunity to segmentation is paging. The program is divided into equal-sized portions referred to as pages, and memory is split into equal-sized gadgets known as page frames. (For implementation reasons, the page size is typically selected to be a strength of among 512 and 4096 bytes.) As with segmentation, every address in a paging scheme is a two-element item, this time which include <page, offset>.

Each cope with is again translated with the aid of a system similar to that of segmentation: The operating machine keeps a desk of consumer page numbers and their proper addresses in reminiscence. The page component of each <page, offset> reference is transformed to a web page body deal with with the aid of desk research; the offset portion is added to the web page body deal with to provide the real memory address of the object known as <page, offset>. This process is illustrated in Figure 4.8.

IDOL

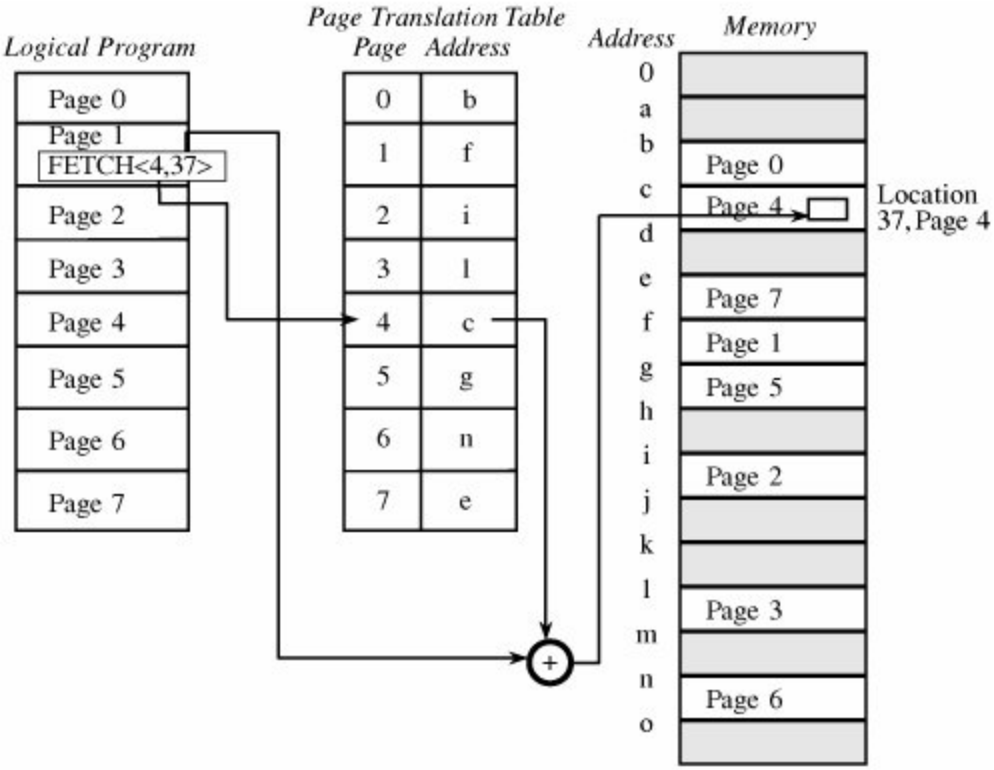


Figure 4-8. Page Address Translation.

IDC

4.3. Control of Access to General Objects

4.3. Control of Access to General Objects

Protecting memory is a selected case of the more trendy problem of protective objects. As multiprogramming has evolved, the numbers and sorts of objects shared have additionally expanded. Here are some examples of the types of items for which safety is appropriate:

memory

a file or data set on an auxiliary storage device

an executing software in memory

a directory of files

a hardware device

a data structure, which include a stack

a table of the operating system

commands, especially privileged instructions

passwords and the person authentication mechanism

the protection mechanism itself

The memory protection mechanism may be pretty simple due to the fact each memory access is guaranteed to undergo specific points in the hardware. With extra trendy objects, the number of factors of getting right of access to may be large, a central authority thru which all accesses pass may be missing, and the kind of get admission to may not clearly be restricted to read, write, or execute.

Furthermore, all accesses to memory occur thru an application, so we can discuss with the program or the programmer as the having access to the agent. In this chapter, we use phrases just like the user or the subject in describing get access to entry to general objects. This user or subject could be a person who uses a

IDOL

computing device, a programmer, a software, some other object, or something else that seeks to apply an object.

There are several complementary desires in defensive objects.

Check every access. We may additionally need to revoke a consumer's privilege to get entry to an object. If we have formerly legal the user to get permission to access the object, we do no longer always intend that the user has to hold indefinite get right of access to to the object. In fact, in a few situations, we may additionally want to save you, in addition, get the right of access to immediately when we revoke authorization. For this reason, each gets entry to via a consumer to an object ought to be checked.

Enforce least privilege. The principle of least privilege states that a topic needs to have access to the smallest wide variety of objects important to carry out a few jobs. Even if greater information could be vain or harmless if the user has been to have got access to objects, the users must not have that extra get right of access to objects. For instance, an application needs to not have access to absolutely the memory address to which a page number address is translated, even though this system could not use that address in any effective manner. Not permitting access to needless objects guards in opposition to safety weaknesses if part of the protection mechanism ought to fail.

Verify acceptable usage. The ability to get entry to is a yes-or-no decision. But it is equally essential to check that the interest to be activity on an object is appropriate. For example, a data structure that includes a stack has particularly suitable operations, consisting of push, pop, clear, and so forth. We can also need no longer only to control who or what has access to a stack however additionally to be guaranteed that the accesses finished are valid stack accesses.

In the next section, we don't forget protection mechanisms suitable for standard objects of unspecified sorts of, such as the varieties of objects listed above. To make the explanations easier to understand, we occasionally use an example of a specific object, along with a document. Note, but, that a preferred mechanism can be used to protect any of the types of object indexed.

Directory

One simple way to shield an object is to use a mechanism that works like a file directory. Imagine we're trying to shield field (the set of objects) from users of a computing system (the set of objects). Every file has a completely unique owner who possesses "control" get admission to rights (along with the rights to claim who has what gets entry to the system) and to revoke access to any individual user at any time. Each user has a file directory, which lists all of the files to which that user has get right of entry to the system.

Clearly, no user can be allowed to write within the file directory because that might be a manner to forge get entry to file. Therefore, the operating system must hold all files directories, under command from the proprietors of files. The obvious rights access to files are commonly study, write, and execute acquainted on many shared systems. Furthermore, some other right access, the owner, is possessed by using the owner, allowing that person to provide and revoke access rights. Figure 4.10 shows an example of a file directory.

IDOL

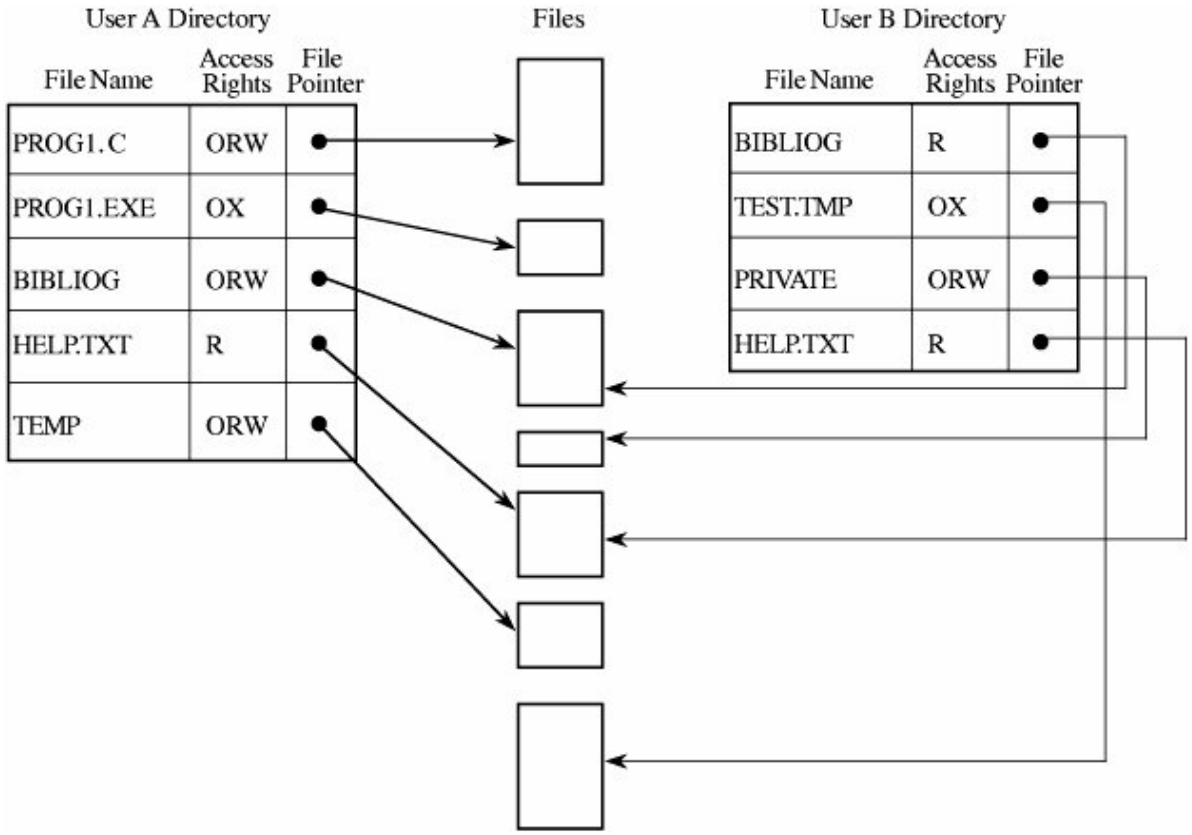


Figure 4.10. Directory Access.

This method is straightforward to enforce as it uses one list per person, naming all of the objects that users are authorized to get an entry into the system. However, several difficulties can arise. First, the list will become too big if many shared objects, along with libraries of subprograms or a not unusual table of users, are accessible to all users. The directory of every user should have one entry for each such shared object, despite the fact that the person has no intention of accessing the object. Deletion must be pondered in all directories.

A second difficulty is revocation to getting entry to. If proprietor A offers exceeded to person B the proper to study document F, an access for F is made inside the directory for B. This granting of access implies a stage of consider among A and B. If A later questions that trust, A can also want to revoke the get right of entry to proper of B. The working device can reply without difficulty to the

single request to delete the proper of B to get admission to F due to the fact that motion entails deleting one access from a specific listing. But if A wants to take away the rights of absolutely everyone to get admission to F, the operating system ought to search every man or woman directory for the entry F, a pastime that can be time-consuming on a massive machine. For instance, big timesharing systems or networks of smaller systems can effortlessly have 5,000 to 10,000 active debts. Moreover, B may also have passed the access right for F to another user, so A may not recognize that F's get entry to exists and should become revoked. This trouble is mainly serious in a network.

One-Third difficulty involves pseudonyms. Owners A and B would have two different data files called F, and they might both ought to allow access simply by S. Obviously, the directory for S could not contain two entries underneath the exact name for different data files. Therefore, S needs to be capable to uniquely determine the F for A (or B). One procedure is to are the first owner's designation as though it were section of the file name, with a notation such as A:F (or B:F).

Suppose, however, that S has trouble remembering file contents from the name F. Another approach is to allow S to name F with any name unique to the directory of S. Then, F from A could be called Q to S. As shown in Figure 4.11, S may have forgotten that Q is usually F from A, and so S requests access again from A for F. But by now A may have more trust in S, so A transfers F with greater rights than before. This action opens up the possibility that one subject, S, may have two distinct sets of access rights to F, one under the name Q and one under the name F. In this way, allowing pseudonyms leads to multiple permissions that are not necessarily consistent. Thus, the directory approach is most likely too simple for most object protection situations.

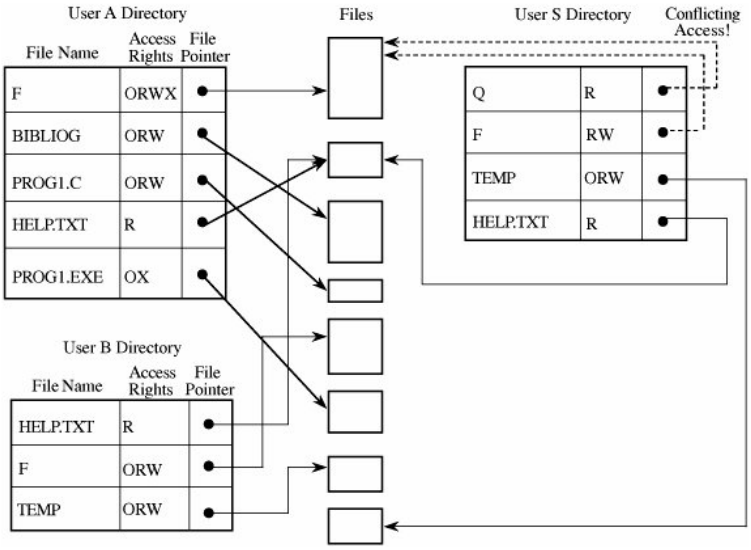


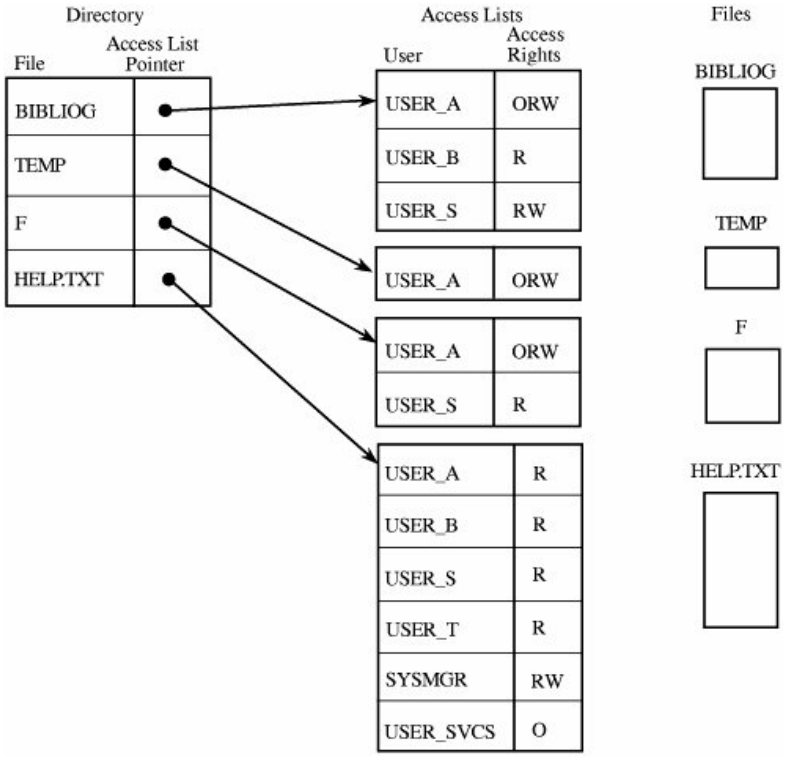
Figure 4.11. Alternative Access Paths.

Access Control List

An elective portrayal is the entrance control list. There is one such list for each instance or object, and the list demonstrates all subjects who ought to approach the instance or object and what their access rights are. This methodology varies from the directory list on the grounds that there is one access control list for every object; a directory is made for each subject. In spite of the fact that this distinction appears to be little, there are some huge focal points.

To perceive how, think about subjects A and S, both of whom approach object F. The operating system will keep up only one access list for F, demonstrating the access rights for A and S, as appeared in Figure 4.12. The access control list can incorporate general default sections for any user. Along these lines, unique clients can have unequivocal rights, and every other user can have a default set of rights. With this association, an open document or program can be shared by every conceivable user of the systems without the requirement for a passage for the object in the individual directory of the individual users.

IDOL



Access Control Matrix

We could think of the directory site like a listing of things accessible by the single theme, and the access checklist as a table determining subjects that can gain access to just one object. The information in these two illustrations are equivalent, the differentiation being the ease associated with use in given circumstances.

As an alternative, we are able to use an access management matrix, a table inside which each row signifies a subject, each steering column represents an object, and entry is the collection of access rights with regard to that subject to that will object. An example rendering of your access control matrix is shown in Stand 4.1. In general, typically the access control matrix is usually sparse (meaning that just about all cells are empty): Just about all subjects do not possess access rights to just about all objects. The access matrix can be represented since a list of triples,

getting the form <subject, object, rights>. Searching a large range of these triples is usually inefficient enough that this particular implementation is seldom applied.

	BIBLIOG	TEMP	F	HELP.TXT	C_COMP	LINKER	SYS_CLOCK	PRINTER
USER A	ORW	ORW	ORW	R	X	X	R	W
USER B	R	-	-	R	X	X	R	W
USER S	RW	-	R	R	X	X	R	W
USER T	-	-	-	R	X	X	R	W
SYS_MGR	-	-	-	RW	OX	OX	ORW	O
USER_SVCS	-	-	-	O	X	X	R	W

Table 4.1. Access Control Matrix.

Until now, we have examined security schemes in which typically the operating system must maintain program all the safety objects and rights. Yet other approaches put a few of the burden around the user. For example, the user may be expected to have a plane ticket or pass that allows access, much like some sort of ticket or identification credit card that should not be duplicated. Extra formally, we say that will a capability is definitely an unforgeable token that gives the particular possessor certain rights to an object. The Multics in addition to Hydra systems used capabilities intended for access control. Theoretically, a new subject can create innovative objects and can stipulate the operations allowed upon those objects. For illustration, users can create items, such as files, info segments, or subprocesses, and even can also specify typically the acceptable forms of operations, many of these as read, write, and even execute. But an customer can also create entirely new objects, such since new data structures, in addition to can define types regarding accesses previously unknown to be able to the system.

A functionality is a ticket providing permission to a controlled by include a certain type involving usage of an object. Intended for the capability to offer you solid protection, the admission must be unforgeable. 1 way to help it become unforgeable is to not provide the ticket directly to the particular user. Instead, the

functioning system holds all seat tickets on behalf of the particular users. The operating technique returns to the customer a pointer to the os data structure, which in turn also links to typically the user. A capability can easily be created only simply by a specific request by your user to typically the operating system. Each functionality also identifies the permitted accesses.

Alternatively, capabilities may be encrypted under the key available only in order to the access control device. If the encrypted capacity contains the identity coming from the rightful owner, customer A cannot copy typically the capability and provide it in order to user B.

One achievable access directly to a great object is transfer or even propagate. A subject getting this right can go copies of capabilities to be able to other subjects. In change, each of these abilities also offers a list regarding permitted types of has access to, one of which may also be transfer. Within this instance, process A new can pass a duplicate of a power to B, that can then pass a new copy to C. N can prevent further submission of the capability (and therefore prevent further scattering in the access right) by simply omitting the transfer ideal from the rights handed in the capacity to G. B might still go certain access rights to be able to C, but not the particular justification to propagate gain access to rights to subjects.

Since a process executes, this operates in a domain name or local name room. The domain is the particular collection of objects to be able to which the process offers access. A domain with regard to an user in a presented time might include a few programs, files, data sectors, and I/O devices many of these as a printer and even a terminal. An illustration of a domain is definitely shown in Figure 4.13.

IDOL

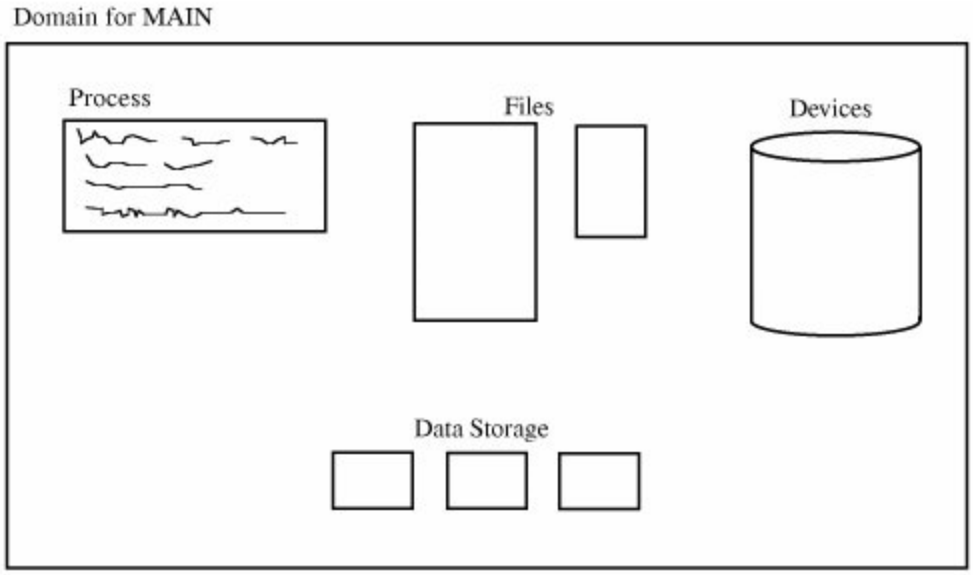


Figure 4-13. Process Execution Domain

As execution proceeds, the procedure may call a subprocedure, passing a portion of the object to which it approaches as contentions to the subprocedure. The domain of the subprocedure isn't really equivalent to that of its calling system; truth be told, a calling technique may pass just a portion of its articles to the subprocedure, and the subprocedure may approach rights to different items not available to the calling strategy. The guest may likewise pass just a portion of its entrance rights for the items it goes to the subprocedure. For instance, a method may go to a subprocedure the privilege to peruse however not alter specific information esteem.

Since every capacity distinguishes a solitary item in a domain , the accumulation of abilities characterizes the area. At the point when a procedure calls a subprocedure and passes certain items to the subprocedure, the operating shapes a heap of the considerable number of abilities of the present system. The working framework at that point makes new abilities for the subprocedure, as appeared in Figure 4.14.

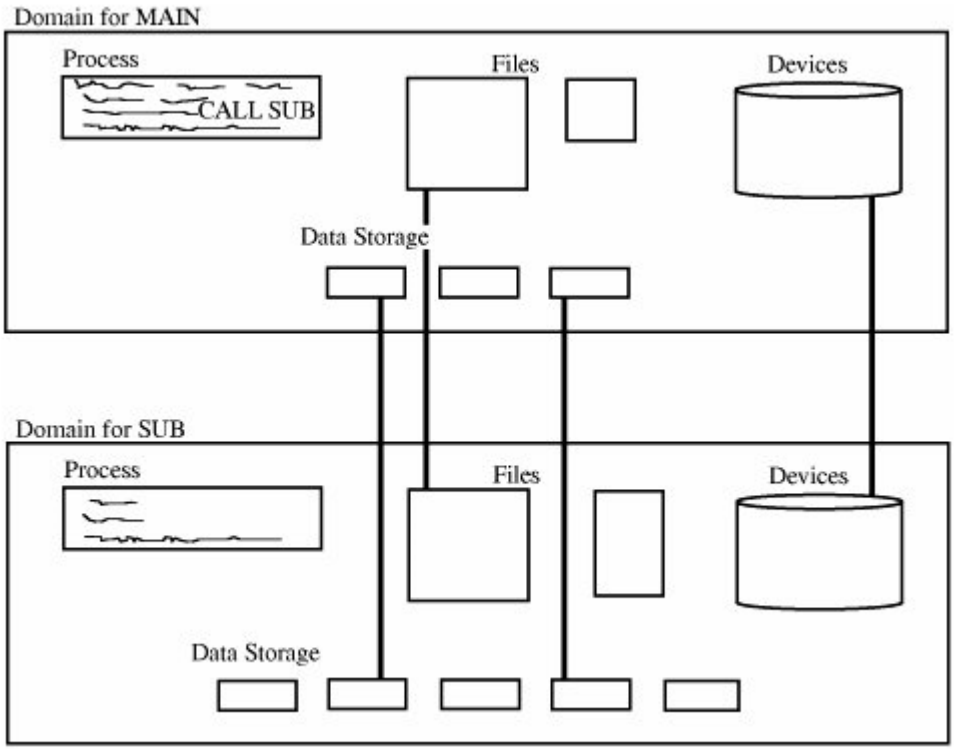


Figure 4.14. Passing Objects to a Subject.

Operationally, capabilities are an uncomplicated method to keep track associated with the access rights regarding subjects to objects in the course of execution. The capabilities happen to be backed up by some sort of more comprehensive table, many of these as an access management matrix or an entry control list. Each moment a process seeks in order to utilize a new item, the operating-system examines the particular master listing of objects and even subjects to ascertain whether typically the object is accessible. If you do, the operating system produces a capability for of which object.

Capabilities should be kept in memory inaccessible in order to normalcy users. One method of accomplishing this really is to be able to store capabilities in sections not pointed at with the user's segment table in order to enclose them in guarded memory as from a couple of base/bounds registers. Another strategy is to use some sort of tagged architecture machine in order to identify capabilities as set ups requiring protection.

During delivery, only the capabilities regarding objects that have already been accessed with the current method are kept readily offered. This restriction improves typically the speed with which entry to a subject could be checked. This technique is basically the a single used in Multics.

Capabilities can be suspended. For the issuing theme revokes a capability, zero further access beneath the terminated capability should be authorized. A capability table can easily contain pointers to typically the active capabilities spawned underneath it so that typically the operating system can find what access rights must be deleted if a capacity is revoked. A related problem is deleting abilities for users which are simply no longer active.

Kerberos

Essential research on capabilities placed the groundwork for future production use in methods for instance Kerberos. Kerberos implements both authentication and access authorization simply by means of capabilities, known as tickets, secured with symmetrical cryptography. Microsoft has established much of its gain access to control in NT+ in Kerberos.

Kerberos requires 2 systems, called the authentication server (AS) and the particular ticket-granting server (TGS), which usually are both section of the important distribution center (KDC). Some sort of user presents an authenticating credential (such as the password) to the authentication server and receives a new ticket showing that typically the user has passed authentication. Obviously, the ticket should be encrypted to prevent the particular user from modifying or even forging one claiming to be able to be a different customer, and the ticket should contain some provision to be able to prevent one user coming from acquiring another user's solution to impersonate that customer.

Now let us suppose that an user, Paul, would like to access a source R (for example, the file, printer, or system port). Joe sends the particular TGS his authenticated admission and a request to work with R. Assuming Joe is usually allowed access, the TGS returns to Joe a couple of tickets: One shows May well

that his access in order to R has been certified, and the second is usually for Joe to existing to R in buy to access R.

Kerberos implements single sign-on; that will is, an user symptoms on once and inside the future all typically the user's (allowable) actions happen to be authorized without the customer needing to sign about again. When an consumer wants usage of a reference in a different domain name, say on a various system or in some sort of different environment or maybe a various company or institution, mainly because long as authorization protection under the law happen to be established between the particular two domains, the customer's access occurs without typically the user's signing on to another system.

Kerberos accomplishes their local and remote authentication and authorization with a great approach to shared top secret encryption keys. In truth, each user's password is usually used as an security key. (That trick likewise means that passwords will be never exposed, reducing the particular risk from interception.)

Procedure-Oriented Access Control

One target of access control is definitely restricting not just which often subjects have access to be able to an object, but likewise the actual can carry out to that object. Go through versus write access may be controlled rather readily simply by most operating systems, although more complex control will be not so easy to attain.

By procedure-oriented protection, all of us imply the existence regarding a procedure that handles access to objects (for example, by performing its very own user authentication to improve the standard protection provided simply by the basic operating system). In essence, the process forms a capsule all-around the object, permitting just certain specified accesses.

Treatments can ensure that has access to an object end up being made via a trusted user interface. For example, neither consumers nor general main system regimens might be allowed instant access to the stand of valid users. Rather, the only accesses authorized might be through about three procedures: one to put an user, one to be able to delete an user, in addition to one to check regardless of

whether a particular name matches to a valid end user. These procedures, especially put and delete, could employ their particular checks to help to make sure that calls for them are legitimate.

Procedure-oriented protection tools the principle of data hiding because the method of implementing a theme are known only to be able to the object's control method. Naturally, this degree associated with protection includes a penalty regarding inefficiency. With procedure-oriented security, there can be not any simple, fast access, fixed up object is regularly used.

Our survey regarding access control mechanisms offers intentionally progressed from very simple to complex. Historically, while the mechanisms have offered greater flexibility, they include succeeded in doing therefore with a price associated with increased overhead. For instance, implementing capabilities that wants to be checked upon each access is challenging than implementing a basic index structure that is inspected only on a subject's first access to a great object. This complexity is definitely apparent both for the customer and to the implementer. The user is conscious of additional protection characteristics, but the naive customer may be frustrated or perhaps intimidated at having in order to select protection options along with little knowledge of their performance. The implementation complexity turns into apparent in slow reply to users. The complete amount between simplicity and even functionality is a continuous battle in security.

Role-Based Access Control

We possess not yet distinguished between kinds of users, yet we wish some users (such as administrators) to experience significant privileges, and all of us want others (such like regular users or guests) to have lower benefits. In companies and academic institutions, this can obtain complicated for the normal user becomes an manager or a baker techniques to the candlestick makers' group. Role-based access handle allows us to connect privileges with groups, these kinds of as all administrators could accomplish this or candlestick makers are forbidden in order to do this. Administering protection is easier if all of us can control access by simply job demands, not by simply person. Access control retains up with

anspecific who changes responsibilities, and even the system administrator will not have to pick the appropriate access handle settings for somebody. For even more details on the detailed aspects of role-based access handle.

4.4. File Protection Mechanisms

Up to now, we have examined strategies to protecting a basic object, no matter typically the object's nature or variety. Sometimes protection schemes will be particular to the sort. To see the way they function, we focus within this part on file protection. Typically the examples we present will be only representative; they carry out not cover all probable means of file defense on the market.

Basic Forms of Protection

We mentioned earlier that all multiuser operating systems must give some minimal protection to be able to keep one user coming from maliciously or inadvertently being able to access or modifying the data files of another. As typically the quantity of users has cultivated, so even offers the difficulty of these protection strategies.

AllNone Protection

Inside the primary IBM OS systems, data were by default general public. Any user could study, modify, or delete some sort of file owned by any kind of other user. Instead involving software- or hardware-based security, the main protection involved have confidence in along with ignorance. System developers supposed that users can be trusted not in order to read or modify others' files because the customers would expect the identical respect from others. Lack of edcuation helped this case, because a good user could access the file only by brand; presumably users knew typically the names only of these files to which that they had legitimate access.

Nevertheless, it was acknowledged of which particular system files have been sensitive and that typically the system administrator could guard them with an username and password. A typical user could workout this feature, but account details were viewed as many valuable for protecting running system files. Two

sagesse guided password use. Occasionally, passwords controlled all has access to (read, write, or delete), giving the system manager complete control of most files. But quite frequently passwords controled only compose and delete accesses given that these two actions influenced others. In either situation, the password mechanism needed a system operator's treatment each time access to be able to the file began.

Even so, this all-or-none protection is usually unacceptable for a number of reasons.

Lack of trust : The presumption of trustworthy users will be not necessarily justified. Regarding systems with few customers who all know every other, mutual respect may possibly suffice; but also in large methods where its not most user knows every various other user, there is zero basis for trust.

Too coarse: Even if a great user identifies an established of trustworthy users, generally there is no convenient solution to allow access only in order to them.

Rise of sharing : This protection scheme is usually more suitable for the batch environment, through which consumers have little opportunity to socialize with other users and even in which users conduct their thinking and checking out when not reaching typically the system. However, on shared-use systems, users interact together with others and programs symbolizing other classes of customers.

Complexity: Because (human) owner intervention is required intended for this file protection, running system performance is degraded. For this reason, this specific type of file defense is discouraged by calculating centers for all nevertheless the most sensitive files sets.

File listings: With regard to accounting purposes and in order to help users remember intended for what files they are usually responsible, various system resources can make a record of all files. As a result, users are not always ignorant of what documents reside on the program. Interactive users may try out to browse through virtually any unprotected files.

Group Protection

As the all-or-nothing approach has numerous drawbacks, researchers sought a better way to protect documents. They focused on discovering groups of users that had some common romantic relationship. In the typical Unix+ setup, the planet is divided into 3 classes: the user, some sort of trusted working group connected with the user, and even the remaining portion of the users. Intended for simplicity we are able to call these types of classes user, group, plus world. Windows NT+ makes use of groups for instance Administrators, Electric power Users, Users, and Friends. (NT+ administrators can furthermore create other groups.)

All authorized users happen to be separated into groups. The group may consist regarding several members working upon a common project, the department, a class, or perhaps a single user. The schedule for group membership is usually must share. The party members incorporate some common curiosity and therefore are presumed to have files in order to share with another party members. In this strategy, no user belongs in order to multiple group. (Otherwise, the member owned by groupings A and B can pass along an Some sort of file to another N group member.)

Whenever creating a file, the user defines access protection under the law to the file intended for the user, for various other members of the similar group, and for just about all other users generally speaking. Usually, the choices for gain access to rights are a restricted set, such as update, readexecute, read, writecreatedelete. For a particular record, an user might state read-only access to the particular general world, read plus update access to the particular girls, and all privileges to the user. This particular approach would be appropriate for a paper getting developed by a bunch, wherein the different members by the crew might improve sections being written inside the group. The papers itself should be readily available for people outside the class to examine but not really change.

A key good thing about the group protection strategy is its ease involving implementation. A person is recognized simply by two identifiers (usually numbers): an user ID in addition to a group ID. These types of identifiers are kept in the particular file directory entry regarding each file and happen to be obtained

by the working system when an customer logs in. Therefore, the particular operating system can quickly check if the recommended access to a data file is requested from somebody whose group ID complements the group ID with regard to the file to become seen.

Although this protection plan overcomes some of the particular shortcomings of the all-or-nothing scheme, it introduces several new difficulties from the particular own.

Group affiliation: An individual user cannot belong in order to two groups. Suppose Mary is owned by one group using Ann and to some sort of second group with Costs. If Tom indicates that will a file is to be able to be readable by typically the group, to which group(s) does this permission recommend? Suppose a file associated with Ann's is readable simply by the group; does Expenses have access to that? These ambiguities are almost all simply resolved by proclaiming that every user is supposed to be to exactly one party. (This restriction does not necessarily mean that most users fit to the same team.)

Multiple personalities: To be able to overcome the one-person one-group restriction, certain people may possibly obtain multiple accounts, enabling them, in effect, in order to be multiple users. This specific hole inside the protection method leads to new issues because a single individual can be only 1 user each time. To discover how problems arise, imagine Tom obtains two balances, thereby becoming Tom1 within a group with Ann and Tom2 in the group with Bill. Tom1 is not really in the identical group as Tom2, thus any files, programs, or even aids developed under typically the Tom1 account could be offered to Tom2 only when they are offered to the whole world. Multiple personalities guide to a proliferation associated with accounts, redundant files, restricted protection for files involving general interest, and hassle to users.

All groupings: To avoid multiple personas, the device administrator may choose that Tom should include access to all his / her files any time he or she is active. This solution sets the responsibility on Mary to regulate with whom he or she shares what things. Intended for example, he might get in Group1 with Ann and Group2 with Invoice. He creates a Group1 file to share using Ann. But if they are

active in Group2 next time he is logged throughout, he still sees typically the Group1 file and may possibly not understand that it will be not accessible to Costs, too.

Limited sharing: Data can be shared simply within groups or together with the world. Users would like to be able in order to identify sharing partners intended for a file on the per-file basis; for illustration, sharing one file using ten people and an additional file with twenty other folks.

Individual Permissions

In revenge of their drawbacks, the particular file protection schemes many of us have described are comparatively simple and straightforward. Typically the simplicity of implementing these people suggests other easy-to-manage strategies which provide finer degrees associated with security while associating agreement with a single data file.

Persistent Permission

From the other situations you are familiar along with persistent permissions. The common implementation of this scheme utilizes a name (you declare a dinner reservation beneath the name of Sanders), a symbol (you demonstrate your driver's license or perhaps library card), or the secret (you say a new secret word or offer the club handshake). In the same way, in computing you will be allowed access because they are upon the access list, offering a token or admission, or giving a pass word. User access permissions could be required for any kind of access or only with regard to modifications (write access).

Most these approaches present apparent difficulties in revocation: Using someone off one record is easy, but it really is considerably more complicated to find almost all lists authorizing someone in addition to remove him or the girl. Reclaiming a symbol or even password is more difficult.

Temporary Acquired Permission

Unix+ techniques provide an interesting authorization scheme based on a new three-level usergroupworld hierarchy. The particular Unix designers added a new

permission called set userid (suid). Issue protection is definitely set for a data file to be executed, typically the protection level is of which of the file's user, not the executor. To be able to see how it performs, suppose Tom owns a new file and allows Ann to execute it using suid. When Ann completes the file, she features the protection rights associated with Tom, not of himself.

This peculiar-sounding permission features an useful application. This permits an user to determine data files to which often access is allowed just through specified procedures.

Intended for example, suppose you need to build a computerized dating services that manipulates a databases of folks available on certain nights. Sue might get interested in a particular date for Saturday, but the lady may have already refused a new request from Jeff, declaring she had other programs. Sue instructs the assistance to not reveal to Barry that she's available. In order to use the service, Drag into court, Jeff, and others should be able to read typically the file and write in order to it (at least indirectly) to determine who will be available or to write-up their availability. But in case Jeff can read the particular file directly, he might discover that Sue has humiliated. Consequently, your dating assistance must force Sue in addition to Jeff (and all others) to access this document only through an entry program that would monitor your data Jeff obtains. Yet if the file gain access to is limited to go through and write by a person as the owner, File suit and Jeff will never ever be able to get into data into it.

Typically the solution is the Unix SUID protection. You generate the database file, offering only you access agreement. You additionally write the plan that is to reach the particular database, and save that with the SUID security. Then, when Jeff completes your program, he quickly acquires your access agreement, but only during performance of the program. Shaun never has direct gain access to to the file mainly because your program will carry out the exact file access. If Jeff exits from the program, he regains their own access rights in addition to loses yours. Thus, your own program can access typically the file, but the software must display to Shaun only the data Barry is allowed to observe.

This mechanism is easy for system functions of which general users should get able to perform simply within a prescribed way. Regarding example, the particular program should be able to be able to modify the file associated with users' passwords, but personal users must be able to change their very own own passwords at any time these people wish. With the SUID feature, a password modification program can be possessed with the system, which can therefore have full gain access to to the system pass word table. The program to be able to change passwords also offers SUID protection to ensure that if a normal user completes it, the program could modify the password record in a carefully limited way on behalf involving the person.

Per-Object and Per-User Protection

The primary restriction of these protection strategies is the ability to be able to create meaningful groups regarding related users who ought to have similar entry to connected objects. The access handle lists or access manage matrices described earlier give very flexible protection. Their own disadvantage is for the person who wants to enable access to many customers and to many various data sets; such a great user must still stipulate each data set in order to be accessed by every user. As a fresh user is added, of which user's special access privileges must be specified simply by all appropriate users.

4.5. User Authentication

An os bases much of their protection on knowing that an user of the particular system is. In real life situations, people commonly question for identification from men and women they do not realize: A bank employee may well ask for a driver's license before cashing typically the, library employees may demand some identification before asking out books, and settlement officials ask for given as evidence of personality. In-person identification is often simpler than remote identification. Regarding instance, some universities never report grades over the particular telephone because the workplace workers do not automatically know the students dialling. Nevertheless , a professor that recognizes the voice involving a certain student can easily release that student's marks. Over time, organizations and even software has developed indicates of authentication, using

papers, voice recognition, fingerprint plus retina matching, and additional trusted means of identity.

In computing, the options are more limited along with the possibilities less secure. Any individual can attempt to sign in into a computing program. Unlike the professor which recognizes a student's words, the computer cannot identify electrical signals in one individual as being any totally different from those of anyone different. Thus, most computing authentication systems has to be based about some knowledge shared simply by the computing method and the user.

Authentication mechanisms use any involving three qualities to verify an user's identity.

Some thing the user knows. Accounts, PIN numbers, passphrases, some sort of secret handshake, and mom's maiden name are samples of what an user might know.

Something the customer has. Identity badges, actual physical keys, a driver's certificate, or an uniform usually are common examples of issues people have that help make them recognizable.

Something the particular user is. These authenticators, called biometrics, depend on some sort of physical characteristic of the particular user, such as a new fingerprint, the pattern associated with a person's voice, or even a face (picture). These authentication methods are old (we recognize friends in man or woman by way of some sort of faces or on some sort of telephone by their voices) but are just beginning to be used found in computer authentications. See Sidebar 4-3 for the glimpse with some of the guaranteeing approaches.

Two or a lot more forms could be combined with regard to more solid authentication; intended for example, a bank credit card along with a PIN combine some thing the consumer has with anything the person knows

Passwords as Authenticators

The nearly all common authentication mechanism with regard to user to operating technique is a password, a new "word" known to pc and user. Although

IDOL

pass word protection seems to present a relatively secure method, human practice sometimes degrades its quality. With this segment we consider passwords, requirements for selecting them, in addition to ways of using these people for authentication. We consider by noting other authentication techniques through studying troubles in the authentication method, notably Trojan horses masking as the computer authentication process.

Use of Security passwords

Passwords are mutually agreed-upon code words, assumed to be able to be known only in order to the user and the particular system. In some instances an user chooses account details; in other cases the device assigns them. The span and format of the particular password also vary coming from one system to a new.

Also though they are commonly used, passwords suffer coming from some difficulties of usage:

Loss. Depending on how the particular passwords are implemented, this is possible that not any one will be ready to replace a dropped or forgotten password. The particular operators or system directors can certainly intervene in addition to unprotect or assign a specific password, but often they can determine what password the user has chosen; when the user loses the particular password, home must become assigned.

Use. Supplying some sort of password for each entry to a file can become inconvenient and time taking in.

Disclosure. If a pass word is disclosed to a great unauthorized individual, the data file becomes immediately accessible. In the event that the user then alters the password to reprotect the file, all typically the other legitimate users has to be informed of the fresh password because their aged password will fail.

Revocation.. To revoke one wearer's access right to the file, someone must transform the password, thereby evoking the same problems since disclosure.

IDOL

The use regarding passwords is fairly easy. A user enters a few part of identification, such since a name or a great assigned user ID; this kind of identification can be accessible to the public or quick to guess because that does not provide the particular real security of the particular system. The machine then needs a password from typically the user. If the username and password matches that on use for the user, the particular user is authenticated in addition to allowed access to the device. If the password complement fails, the system asks for the password again, within case the user mistyped.

Additional Authentication Information

Besides the name and password, we are able to use other information obtainable to authenticate users. Assume Adams works in typically the accounting department through the switch between 8: 00 a new. m. and 5: 00 p. m., Monday by way of Friday. Any legitimate entry attempt by Adams have to be made during all those times, through a workstation within the accounting department workplaces. By limiting Adams in order to logging in under individuals conditions, the machine protects towards two problems:

Someone through outside might try in order to impersonate Adams. This test would be thwarted simply by either the time regarding access or the interface through which the gain access to was attempted.

Adams may well attempt to access the program from home or about a weekend, planning in order to use resources prohibited or even to do something of which would be too high-risk with other people about.

Limiting users to specific workstations or certain instances of access can result in complications (as when a great user legitimately should operate overtime, a person offers to get into the system although out of town over a business trip, or the particular workstation fails). Nevertheless, some companies use these types of authentication techniques because typically the added security they offer outweighs inconveniences.

IDOL

Using further authentication information is named multifactor authentication. Two types of authentication (which is, obviously, acknowledged as two-factor authentication) vs. one, assuming of study course that this two forms are usually strong. But as typically the number of forms rises, so also does typically the inconvenience. (For example, consider about getting through the security checkpoint at a great airport.) Each authentication factor requires the method and its administrators in order to manage more security info.

Attacks on Passwords

Exactly how secure are passwords them selves? Passwords are somewhat restricted as protection devices as a result of relatively small number regarding bits of information these people contain.

Here are a few ways you might end up being able to determine a great user's password, in reducing order of difficulty.

Consider all possible passwords.

Attempt frequently used passwords.

Try out passwords likely for typically the user.

Hunt for the program list of passwords.

Request the user.

Loose-Lipped Systems

So far the particular process seems secure, although in fact it includes a few vulnerabilities. To see exactly why, consider the actions associated with a would-be intruder. Authentication is based on the particular actual <name, password> pair A complete incomer is presumed to realize nothing of the method. Suppose the intruder endeavors to access a method in the following way. (In the following cases, the system messages are generally in uppercase, and typically the user's responses are usually in lowercase.)

IDOL

WELCOME TO TYPICALLY THE XYZ COMPUTING SYSTEMS

ENTER IN USER NAME: adams

ILL USER NAMEUNKNOWN USER

GET INTO USER NAME:

We thought that the intruder recognized nothing of the program, but and not having to do a lot, the intruder found out there that adams is not really typically the name of an approved user. The intruder may try other common titles, first names, and very likely generic names like technique or operator to create a new list of authorized customers.

An alternative solution arrangement of the particular login sequence is proven below.

THANKS FOR VISITING THE XYZ COMPUTING DEVICES

ENTER CONSUMER NAME: adams

ENTER PASS WORD: john

INVALID ACCESS

ENTER IN USER NAME:

This method notifies an user associated with a failure only right after accepting both the customer name and the security password. The failure message ought to not indicate unique typically the user name or pass word that is unacceptable. Inside this way, the burglar does not know which usually failed.

These examples furthermore gave a clue about which computing system is definitely being accessed. The legitimate outsider does not have right in order to know that, and legit insiders already know just what system they may have accessed. Inside the example below, the particular user is given not any information until the technique is assured in the personality of the user.

GET INTO USER NAME: adams

IDOL

GET INTO PASSWORD: john

INVALID ENTRY

ENTER USER NAME: adams

ENTER PASSWORD: johnq

HERE YOU ARE AT THE XYZ COMPUTING TECHNIQUES

Exhaustive Attack

In a good exhaustive or brute power attack, the attacker attempts all possible passwords, normally in certain automated fashion. Regarding course, the quantity of possible security passwords depends on the execution of the particular processing system. For example, in case passwords are words containing of the 26 figures AZ and can turn out to be of any length by 1 to 8 heroes, there are 261 security passwords of 1 character, 262 passwords of 2 figures, and 268 passwords involving 8 characters. Therefore, typically the system in general has 261 + 262 +... & 268 = 269 -- 1 5 * 1012 or five million feasible passwords. The number involving seems intractable enough. When we were to work with a computer to produce and even try each password in a rate of looking at one password per nanosecond, it could take on the particular order of 150 decades to test all account details. But if we might accelerate the search in order to one password per microsecond, the work factor falls to about 8 days. This amount of moment is reasonable if the particular reward is large. With regard to instance, an intruder may well try to break the particular password on a document of bank card numbers or perhaps bank account information.

Although the break-in time may be made more tractable in several ways. Searching for some sort of single particular password will not necessarily require almost all passwords to be tried out; an intruder needs to be able to try only until typically the correct password is determined. If the group of most possible passwords were equally distributed, an intruder is likely to need to try simply half the password place: the expected number involving searches to find virtually any particular password. However, a good intruder can also work with to advantage

the point that account details are not evenly sent out. As a password has in order to be remembered, people have a tendency to pick simple accounts. This feature reduces the dimensions of the password space.

Possible Passwords

Think of the word.

Could be the word a person thought of long? Is definitely it uncommon? Is that challenging to spell or in order to pronounce? The response to almost all three of these issues is probably no.

Penetrators looking for passwords realize these kinds of very human characteristics and even rely on them to their edge. Therefore, penetrators try strategies that are prone to prospect to rapid success. In case people prefer short account details to long ones, typically the penetrator will plan in order to try all passwords although to try them throughout order by length. Right now there are only 261 and up. 262 + 263=18, 278 passwords of length a few or less. At the particular assumed rate of 1 password per millisecond, almost all of these passwords could be checked in 18. 278 seconds, hardly a concern having a computer. Even growing the tries to 5 or 5 characters increases the count only to be able to 475 seconds (about 6 minutes) or 12, 356 seconds (about 3. five hours), respectively

15	0.50%	were a single(!) ASCII character
72	2%	were two ASCII characters
464	14%	were three ASCII characters
477	14%	were four alphabetic letters
706	21%	were five alphabetic letters, all the same case
605	18%	were six lowercase alphabetic letters
492	15%	were words in dictionaries or lists of names
2831	86%	total of all above categories

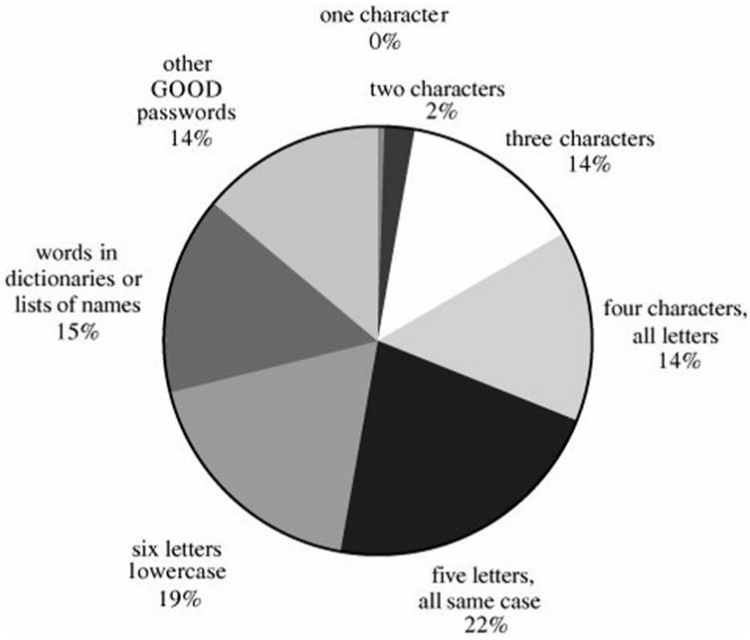


Figure 4.15. Users' Password Choices.

Lest you dismiss these results as dated (they were claimed in 1979), Klein frequent the experiment in 1990 and Spafford in 1992. Each gathered roughly 15,000 passwords. Klein claimed that 2.7 per-cent of this passwords have been guessed in mere quarter-hour of machine moment and 21 percentage were guessed within a week! Spafford located the average password length was initially 6.8 heroes, and 28.9 percent consisted of simply lowercase alphabetic heroes. Observe that both these analyses were done after the Web worm (defined in Section 3) succeeded, partly by breaking poor passwords.

Even in 2002, the British online bank Egg found users nonetheless choosing poor passwords. A full 50 pct of passwords for his or her online banking service were family customers' labels: 23 percentage children's names, 19 percent a husband or wife or companion, and 9 percent their own. Alas, pets arrived in at only 8

percent, while stars and soccer (soccer) stars tied at 9 percent each. And in 1998, Knight and Hartley reported that about 35 percent of passwords are usually deduced from syllables and initials from the account owner's name.

Two friends we know have told us their passwords once we aided them administer their methods, and their passwords would both have been among the first we would contain guessed. But, you claim, these are amateurs unacquainted with the security risk of a weakened password. At a recently available meeting, a security and safety expert associated this experience: He thought he had picked a solid security password, so he asked a category of learners to request him a few questions and offer some guesses concerning his password. He was surprised that they questioned only a few questions before they had deduced the password. And this was a security specialist.

Several news posts have claimed which the four most typical passwords happen to be "God," "sex," "love,"and "money" (the order among those is unspecified). The possibly apocryphal set of prevalent passwords at geodsoft.com/howto/security password/common.htm appears at other places on the Internet. Or start to see the default password checklist at www.phenoelit.de/dpl/dpl.html. Whether these are seriously passwords we have no idea. Nevertheless, it warrants a peek because similar lists are bound to be built into some hackers' equipment.

Several network internet sites posting dictionaries of phrases, technology fiction characters, spots, mythological names, Chinese language words, Yiddish phrases, and other specialised lists. All these lists are uploaded to help webpage administrators identify customers who have chosen weak passwords, but the same dictionaries can also be used by attackers of sites that do not need like attentive administrators. The COPS ,Break , and SATAN utilities allow an administrator to scan something for weak passwords. But these very same utilities, or various other homemade ones, let attackers to accomplish the same. Nowadays Internet sites present so-called password recuperation software program as freeware or shareware for under $20. (These are password-cracking courses.)

People think they could be clever by picking a simple password and replacing certain characters, such as for example 0 (zero) for letter O, 1 (one) for letter I or L, 3 (three) for letter E or @ (at) for letter A. But consumers aren't the only real individuals who could come up with these substitutions. Knight and Hartley record, in order, 12 tips an attacker might attempt to be able to determine a security password. These steps are in increasing amount of difficulty (number of guesses), so they indicate the quantity of work to that your attacker must go to derive a password. Listed below are their password speculating steps:

-. no password

-. exactly like an individual ID

-. is, or is derived from, the user's name

-. common word list (for example, "password," "hidden knowledge," "non-public") plus common names and designs (for instance, "asdfg," "aaaaaa")

-. short university dictionary

-. complete English term list

-. common non-English terminology dictionaries

-. short college or university dictionary with capitalizations (PaSsWorD) and substitutions (0 for O, and so forth)

-. complete English with capitalizations and substitutions

-. popular non-English dictionaries with capitalization and substitutions

-. brute power, lowercase alphabetic characters

-. brute force, complete character set

IDOL

Although the final step will always succeed, the ways right away preceding it are so frustrating that they can deter all however the devoted attacker for whom period isn't a limiting component.

Plaintext System Security password List

To validate passwords, the machine must have a way of comparing entries with actual passwords. Rather than trying to speculate a user's password, an attacker may as an alternative target the system password document. Why think when with one table you can ascertain all passwords with complete accuracy?

On some systems, the password listing is a data file, organized essentially being a two-column stand of person IDs and equivalent passwords. This information is certainly also obvious to leave out in the wild. Various security methods are used to conceal this stand from those that should not view it.

You might defend the desk with strong admittance controls, limiting usage of the operating-system. But perhaps this tightening up of control is looser than it should be, because don't assume all operating system component needs or deserves usage of this table. For instance, the operating-system scheduler, accounting exercises, or storage supervisor have no need to know the table's material. Unfortunately, in a few systems, there are n+1 known customers: n regular users along with the operating-system. The operating system isn't partitioned, consequently all its modules have access to all privileged info. This monolithic view of the operating-system implies that a end user who exploits a flaw in a single portion of the operating system has access to all the system's deepest secrets and techniques. A better strategy is to control table usage of the modules that require access: the user authentication module and the parts connected with installing new consumers, for example.

f the stand is stashed in plain eyesight, an intruder can merely dump memory with a convenient time and energy to access it. Cautious timing may allow a user to dump the contents of all of recollection and, by exhaustive lookup, find worth that appear to be the password stand.

IDOL

System backups could also be used to obtain the password table. In order to recover from system errors, system administrators regularly back up the file room onto some auxiliary channel for safe storage space. In the unlikely event of a problem, the file system can be reloaded from the backup, using a loss simply of changes made since the last backup. Backups normally contain only record contents, without protection mechanism to control file gain access to. (Physical protection and access settings for the backups themselves are usually depended on to provide security for the articles of backup press.) If a regular customer can access the backups, perhaps ones from weeks, months, or years ago, the password tables stored inside them may include entries which are still valid.

Finally, the password file is a copy of an file saved on disk. Anyone with access to the drive or anyone who can overcome file gain access to restrictions can buy the password file.

Encrypted Security password File

There is an easy way to foil an intruder seeking passwords in simple perception: encrypt them. Regularly, the password record is disguised .from view with conventional encryption or one-way ciphers.

With normal encryption, either the complete password table will be encrypted or simply the password column. Whenever a user's password is certainly received, the stored password is usually decrypted, and both are compared.

Even with encryption, there is still a slight visibility because for an instantaneous the user's security password comes in plaintext in key memory. That's, the password can be acquired to anyone who could obtain access to all of memory.

A safer approach utilizes one-way encryption. The password table's entries happen to be encrypted by way of a one-way encryption and then stored. Once the user gets into a password, additionally it is encrypted and compared with the table. If both values are similar, the authentication succeeds. Of course, the encryption must be so that it is improbable that two passwords would encrypt to exactly the same ciphertext, but this feature is true for most risk-free encryption algorithms.

With one-way encryption, the password file can be stored in ordinary view. For example, the password stand for that Unix operating system can be read by any user unless special accessibility controls have already been installed. As the contents will be encrypted, backup copies of the security password table are no more a problem.

There's always the possibility that two people might choose the same password, so creating two similar entries in the password file. Despite the fact that the entries are usually encrypted, each user will understand the plaintext equal. For example, if Costs and Kathy both select their passwords on April 1, they might choose APRILFOOL as a password. Bill might read the password record and observe that the encrypted type of his security password is equivalent to Kathy's.

Unix+ circumvents this vulnerability by using a password extension, named the salt. The salt is really a 12-bit number shaped from the machine time and the procedure identifier. Hence, the salt is likely to be unique for every user, and it can be stored in plaintext within the password data file. The salt is certainly concatenated to Bill's password (pw) when he selects it; E(pw+saltB) is stored for Costs, and his salt value can be kept. When Kathy chooses her password, the salt differs because the time or the process number is different. Call this different one saltK. On her behalf, E(pw+saltK) and saltK will be placed. When either individual tries to log in, the machine fetches the correct salt in the password desk and includes that while using password before accomplishing the encryption. The encrypted versions of (pw+sodium) are very different for both of these users. When Costs looks down the security password list, the encrypted edition of his security password will not take a look at all like Kathy's.

Storing the password file in the disguised contact form relieves much of the pressure to safeguarded it. Better still is to limit access to operations that legitimately will need access. In this manner, the password data file is secured to an even commensurate while using protection provided by the security password itself. Someone who has broken the control buttons of the file system has access to data, not only passwords, which is a serious menace. But if an attacker

effectively penetrates the external security covering, the attacker nevertheless must see through the encryption of the password file to access the useful data in it.

Indiscreet Users

Speculating passwords and bursting encryption can be tedious or challenging. But there is a simple way to obtain a security password: Get it directly from an individual! People frequently tape a security password aside of a terminal or write it over a card just inside the top workplace drawer. Users are afraid they will forget their passwords, or they can not be bothered attempting to remember them. It really is particularly tempting to write the passwords down when customers have several addresses.

Users sharing function or data can also be tempted to talk about passwords. If an individual needs a record, it is better to say "my security password is x; obtain the file yourself" than to arrange to share the file. This example is a result of user laziness, nonetheless it may be caused or exacerbated by way of a system that makes sharing inconvenient.

Within an admittedly unscientific poll performed by Verisign, two-thirds of individuals approached on the road volunteered to reveal their password to get a coupon best for a cup of coffee, and 79 percentage admitted they applied the same password for several system or site.

Password Selection Criteria

On the RSA Security Seminar in 2006, Bill Gates, head of Microsoft, explained his perspective of a global where passwords would be obsolete, having long gone the way in the dinosaur. In their place innovative multifactor authentication technology would offer much larger security than passwords ever could. But that is Bill Gates' watch of the future; despite generations of articles or blog posts about their weakness, passwords are usually with us nonetheless and will be for quite a while.

IDOL

So what can we conclude about passwords? They must be hard to imagine and difficult to find out exhaustively. But the degree of issues should be correct to the safety measures needs of the problem. To these edges, we present different guidelines for password selection:

Use characters apart from only AZ. If passwords happen to be chosen from letters AZ, you can find only 26 alternatives for each figure. Adding digits expands the number of options to 36. Employing both uppercase and lowercase letters plus digits expands the number of possible figures to 62. Although this change seems small, the result is large when someone is testing a full space of most possible combos of characters. It requires about 100 hours to check all 6-letter words selected from letters of 1 case only, but it takes about 2 years to test all 6-mark passwords from top- and lowercase letters and digits. Although 100 hours is reasonable, 2 years is oppressive enough to make this attack far less attractive.

Choose long passwords. The combinatorial explosion of passwords begins at length four or five 5. Choosing much longer passwords helps it be less likely a password will undoubtedly be uncovered. Understand that a brute power penetration can quit as soon as the password is found. Some penetrators will try the easy casesknown words and quick passwordsand move ahead to another concentrate on if those episodes fail.

Avoid actual titles or text. Theoretically, there are 26^6 or about 300 million 6-letter "words", but you can find only about 150,000 text in an excellent collegiate dictionary, overlooking length. By choosing one of the 99.95 per-cent nonwords, you power the attacker to use a longer brute drive search instead of the abbreviated dictionary look for.

Choose an unlikely password. Password option is a dual bind. To keep in mind the password easily, you want one which has special significance to you. On the other hand, you don't wish someone else in order to imagine this special meaning. One easy-to-remember password will be 2Brn2B. That improbable looking jumble is really a simple transformation of "for being or not to be." The first letters of phrases from a music, a few characters from different thoughts of an exclusive phrase, or perhaps a memorable basketball score are types of

realistic passwords. But don't be too noticeable. Password-cracking tools in addition test replacements of 0 (zero) for o or O (notice "oh") and 1 (one) for l (notice "ell") or $ for S (letter "ess"). Consequently I10veu has already been in the research file.

Change the password regularly. Even though there is absolutely no reason to think that the password has been jeopardized, change is advised. A penetrator may split a password system by obtaining an old list or working exhaustively on an encrypted list.

Don't create it down. (Note: This time-honored tips is relevant only when physical security is a serious risk. People who have accounts on many different machines and machines, not to mention bank and bank card PINs, could have trouble remembering all of the access codes. Setting up all codes exactly the same or employing insecure but easy-to-remember passwords may be more dangerous than composing passwords on the reasonably well guarded list.)

Don't tell anyone else. The easiest attack is **social engineering**, where the attacker contacts the system's administrator or a customer to elicit the password for some reason. For instance, the attacker may phone a user, case to end up being "system supervision," and have the user to verify the user's security password. Under no circumstances should you ever give out your private security password; genuine administrators can circumvent your password if need be, and others are merely trying to deceive you.

To help users select good passwords, some techniques present meaningless but pronounceable passwords. For instance, the VAX VMS technique randomly creates five passwords from which the user selects one. They're pronounceable, so the user can duplicate and memorize them. However, the user may misremember a security password because of getting interchanged syllables or characters of your meaningless string. (The noise "bliptab" is no easier misremembered than "blaptib" or "blabtip.")

Yan et al. performed experiments to find out whether consumers could keep in mind passwords or passphrases far better. First, they found that users are terrible

at remembering arbitrary passwords. And directions to users about the importance of choosing good passwords possessed little effect. However when they asked users to select their very own password based on some mnemonic word they selected themselves, the consumers selected passwords that were harder to guess than normal (not predicated on a saying) passwords.

Other systems motivate users to change their passwords frequently. The regularity of password change is usually a system parameter, which may be changed for your characteristics of confirmed installation. Suppose the frequency is defined at 1 month. Some systems begin to warn the user after 25 times that the password is about to expire. Others hold out until 1 month and inform the user that the security password has expired. Some methods nag without end, whereas other devices take off a user's access if a security password has expired. Still others force the user immediately in to the password change power on the initial login after 1 month.

Grampp and Morris claim that reminder process is not necessarily good. Choosing passwords isn't difficult, but under great pressure a user may choose any password, merely to fulfill the system's need for a fresh one. In addition, if this is the only moment a password could be changed, a negative password choice cannot be changed before next scheduled moment.

One-Time Passwords

A one-time password will be one that adjustments every time it is used. Instead of assigning a static expression to a customer, the machine assigns a static numerical function. The system provides an debate to the function, and the user computes and comes back the function value. Such systems are also called challengeresponse techniques because the technique presents a challenge to an individual and judges the authenticity of an individual with the user's response. Here are some simple examples of one-time password features; these functions are overly simplified to make the explanation easier. Highly complex functions may be used in place of these simple kinds for coordinator authentication in a network.

f(x) = x + 1. With this particular function, the machine prompts using a benefit for x, and an individual enters the worthiness x + 1. The forms of mathematical functions used are limited only by the power of the user to compute the reaction efficiently. Other similar options happen to be $f(x) = 3x^2 - 9x + 2$, $f(x) = p_x$, where px is the xth prime variety, or f(x) = d * h, where d may be the time and h may be the hour of the current time. (Alas, several users cannot execute simple arithmetic within their heads.)

f(x) = r(x). For this function, the receiver uses the argument because the seed for your random amount generator (available to both the recipient and coordinator). An individual replies with the value of the initial random number developed. A variant of this scheme utilizes x as a number of random numbers to generate. The receiver produces x random figures and sends the xth of these to the variety.

$f(a_1a_2a_3a_4a_5a_6) = a_3a_1a_1a_4$. With this particular function, the system provides a character string, that your user must transform in some predetermined manner. Again, many different character operations can be used.

f(E(x)) = E(D(E(x)) + 1). In this particular function, the laptop directs an encrypted price, E(x). An individual must decrypt the value, perform some numerical performance, and encrypt the result to come back it to the machine. Clearly, for individual employ, the encryption performance must be a thing that can be done easily by hand, unlike the good encryption algorithms. For machine-to-machine authentication, on the other hand, an encryption algorithm such as for example DES or AES is appropriate.

One-time passwords are very very important to authentication because an intercepted security password is useless since it cannot be reused. However, their effectiveness is limited with the complexness of algorithms people should be expected to keep in mind. A password-generating machine can implement more technical functions. Several types are readily available at reasonable prices. They are quite effective at countering the risk of transmitting passwords in plaintext across a network.

The Authentication Process

Authentication usually works as described formerly. However, users once in a while mistype their passwords. A end user who receives a note of INCORRECT LOGIN will thoroughly retype the login and access the system. A good user who's a terrible typist can log in successfully in several tries.

Some authentication procedures are intentionally sluggish. A legitimate individual won't complain when the login process can take 5 or 10 a few moments. To some penetrator who's striving an exhaustive search or a dictionary search, however, 5 or 10 moments per trial can make this school of attack usually infeasible.

A person whose login makes an attempt continually fail may not be an authorized person. Systems commonly detach a user following a few failed logins, forcing an individual to reestablish a connection with the machine. (This step will slow down a penetrator who is trying to permeate the machine by telephone. Aft In more secure installations, ending penetrators is even more significant than tolerating consumers' mistakes. For example, some technique administrators assume that legitimate customers can form their passwords appropriately within three tries. After three successive security password failures, the account for that user is certainly disabled in support of the safety administrator can reenable it. This action identifies accounts that may be the prospective of attacks by penetrators.

Fixing Flaws inside the Authentication Process

Password authentication assumes that anyone who is aware of a password may be the individual to whom the password belongs. As we have seen, passwords can be guessed, deduced, or inferred. Some people hand out their passwords for the asking. Some other passwords have already been obtained simply by someone observing a end user typing inside the password. The password can be considered as a preliminary or first-level piece of evidence, but skeptics will need more convincing proof.

There are several ways to provide a second level of protection, incorporating another round of passwords or perhaps achallengeresponse interchange.

ChallengeResponse Systems

As we contain just noticed, the login is usually period invariant. Except when passwords happen to be evolved, each login appears like every other. A more sophisticated login requires a user Identification and password, accompanied by a challengeresponse interchange. In such an interchange, the system prompts the user for an answer which will be different each time an individual logs in. For instance, the machine might exhibit a four-digit variety, and the user would have to correctly enter a function like the sum or product or service on the digits. Each customer is assigned a different challenge work to compute. Because there are many possible challenge functions, a penetrator who catches the user Identification and password cannot always infer the proper function.

A physical device similar to a calculator may be used to implement a far more complicated response purpose. The user gets into the challenge number, and these devices computes and displays the response for the user to enter order to log in. (er a small number of problems, the penetrator must reconnect, which takes a couple of seconds.)

Impersonation of Login

In the systems we have defined, the proof will be one-sided. The machine demands certain recognition of the user, but the consumer is supposed to trust the system. However, a programmer can easily write a program that displays the standard prompts for person ID and security password, captures the pair entered, retailers the pair inside a file, exhibits SYSTEM Problem; DISCONNECTED, and exits. This assault is a type of Trojan horse. The perpetrator pieces it up, departs the terminal unattended, and waits for an innocent victim to try a login. The naive sufferer may not perhaps suspect a security breach has occurred.

To foil this sort of attack, the user should be sure the road to the system is reinitialized each and every time the system is used. On some systems, turning the terminal on / off again or pressing the BREAK major generates a clear signal for the computer to prevent any running method with the terminal. (Microsoft selected <CTRLALTDELETE> because the way to the safe authorization

mechanism because of this.) Don't assume all computer recognizes power-off or Separate being an interruption of the current method, though. And processing systems tend to be accessed through systems, so physical reinitialization is impossible.

Alternatively, the user can be suspicious of the processing system, in the same way the system is suspicious of the user. The user won't enter confidential files (like a security password) until persuaded that the processing system is legitimate. Needless to say, the laptop acknowledges the user only after passing the authentication process. A computing method can display some information acknowledged only by an individual and the system. For example, the system might read the user's label and reply "YOUR Final LOGIN Had been 10 APRIL AT 09:47." An individual can verify the date and period are accurate before stepping into a secret password. If higher security is desired, the system can mail an encrypted timestamp. An individual decrypts this and discovers that enough time is current. The user then replies with an encrypted timestamp and password, to convince the machine that a harmful intruder has not intercepted a security password from some prior login.

Biometrics: Authentication Not really Using Passwords

Some sophisticated authentication devices are actually available. The unit consist of handprint detectors, speech recognizers, and identifiers of patterns in the retina. Authentication with such devices uses unforgeable physical features to authenticate customers. The cost is constantly on the fall as these devices are implemented by major market segments; the devices are useful in very high security situations. In this particular section we consider a several approaches available.

Biometrics are natural authenticators, predicated on some physical feature of our body. The set of biometric authentication systems is still developing. Now there will be devices to recognize the following biometrics: fingerprints, hand geometry (shape and size of fingers), retina and iris (parts of the eye), tone, handwriting, blood vessels in the finger, and encounter. Authentication with biometrics features benefits over passwords because a biometric can't be lost, stolen,

forgotten, lent, or forged and is definitely available, always accessible, so to speak.

Id versus Authentication

Two concepts are easily confused: id and authentication. Biometrics have become reliable for authentication but significantly less reputable for authentication. The reason is mathematical. All biometric viewers work in two phases: First of all, a consumer registers with the reader, where time a feature of an individual (for example, the geometry in the hand) is taken and reduced to a template or style. During registration, an individual may be asked to present the hand many times so that the registration software program can adjust for variations, such as for example how the hands is positioned. Next, the user in the future looks for authentication from the system, during which time the machine remeasures the hand and compares the brand new measurements together with the stored template. If the new measurement is nearby enough to the template, the system accepts the authentication; in any other case, the machine rejects it. Every template is therefore a routine of some amount of measurements.

Unless every design template is unique, that's, no two people have exactly the same measured hand geometry, the system cannot uniquely determine subjects. However, so long as it is improbable that an imposter could have exactly the same biometric template as the real user, the system can authenticate. The distinction is between a system that talks about a side geometry and says "this is Captain Hook" (recognition) versus a man who claims "I, Captain Hook, provide my palm to show who I'm" and the machine confirms "this side fits Captain Hook's template" (authentication). Biometric authentication can be feasible today; biometric identification is basically still a study topic.

Problems with Biometrics

There are many problems with biometrics:

Biometrics are fairly new, and some people get their employ intrusive. Palm geometry and deal with recognition (which can be done from a camera over the room) are usually scarcely invasive, but people have real worries about peering

into a laser beam or sticking a hand into a slot. for some examples of persons resisting biometrics.)

Biometric recognition gadgets are expensive, although as the devices are more popular, their costs go down. Even now, outfitting every user's workstation with a reader can be expensive for a big company with many employees.

All biometric visitors apply sampling and set up a threshold for when a match is near enough to accept. The device has to sample the biometric, measure often hundreds of tips, and compare and contrast that set of measurements having a template. There's usual variability if, for example, your face is tilted, you press one side of the finger more than another, or your speech is suffering from an infection. Variant reduces accuracy.

Biometrics can become a single point of failure. Consider a retail application in which a biometric recognition is certainly associated with a payment design: As one user puts it, "If my credit card fails to enroll, I can usually pull out another card, but if my fingerprint is not recognized, I've only that certain hand." Forgetting a password is a user's fault; failing biometric authentication is not.

Although equipment is improving, there are still incorrect readings. We content label a "false good" or "false accept" a studying that is recognized when it should be rejected (that's, the authenticator will not match) as well as a "false adverse" or "false reject" one which rejects when it should accept. Often, reducing a false good rate increases fake negatives, and vice versa. The results for a incorrect negative are usually less than for any false positive, so an acceptable technique may have a false positive charge of 0.001 per-cent but a incorrect negative rate of just one 1 percent.

The speed of which a recognition must be done limits accuracy and reliability. We might ideally like to acquire several readings and merge the outcomes or measure the closest match. But authentication is performed to allow a user to accomplish something: Authentication isn't the end objective but a gate maintaining an individual from the goal. An individual understandably really wants to see through the gate and becomes frustrated and annoyed if authentication takes too long.

Although we prefer to think of biometrics as exceptional parts of an individual, forgeries are feasible. The most renowned example was an synthetic fingerprint produced by research workers in Japan . Although tricky and unusual, forgery will undoubtedly be an issue whenever the praise for a bogus positive is high enough.

Sometimes overlooked inside the authentication discussion is that credibility is a two-sided problem: The system needs guarantee that an individual is authentic, however the user desires that same guarantee about the method. This second concern has led to a new school of computer scams called phishing, where an unsuspecting customer submits sensitive info to a malicious program impersonating a trustworthy one. Common goals of phishing disorders are banks along with other financial institutions because fraudsters use the sensitive info they get from customers for taking customers' cash from the real institutions.

Authentication is vital for an operating system because accurate individual identification is the key to specific access rights. Just about all operating systems and computing program administrators have applied reasonable but stringent security steps to lock out illegal users before they can access system methods. often an inappropriate mechanism is pressured into use as an authentication device.

IDOL

4.6 Review Question

1. Give an example of the usage of physical parting for security in a computing environment.

2. Give an example of the usage of temporal separation for security inside a computing environment.

3. Give an example of an thing whose security stage may transform during execution.

4. Respond to the allegation "A great operating system needs no protection for its executable program code (in memory) because that program code is a duplicate of code maintained on disk."

5. Explain what sort of fence register can be used for relocating a user's program.

6. Can any number of concurrent processes get protected in one another by just one couple of platform/bounds registers?

7. The talk of foundation/bounds registers means that program code can be execute-only and this data areas are usually read-write-only. Will be this ever false? Explain your solution.

8. A design employing tag parts presupposes that adjacent storage area locations hold dissimilar points: a type of code, a bit of data, a type of code, two bits of data, and so forth. Most programs do not look like that. How can tag bits turn out to be appropriate in a situation in which courses have the extra conventional set up of program code and data?

9. What are some other levels of safety that users should apply to program code or data, in addition to the common read, write, and execute agreement?

10. If two customers share access to a segment, they must do so by exactly the same name. Must their defense rights into it be the similar? Why or why not?

11. An issue with either segmented or paged target translation is timing. Assume a user wants to read some files from an input device into memory space. For

proficiency during data transport, often the actual memory address at which the data should be placed is furnished to a I/O device. The real address is passed in order that time-consuming target translation does not have to be done during a extremely fast data move. What security difficulties does this approach bring?

12. A directory is also an subject to which entry should be handled. Exactly why is it not appropriate to allow users to change their own directories?

13. Why should the directory of one user not end up being generally accessible (for read-only gain access to) to other users?

14. Describe each of the following four kinds of access control mechanisms in terms of (a) ease of determining authorized accessibility during execution, (b) ease of adding access for a new subject, (c) simple deleting access by way of a subject, and (d) simple creating a fresh thing to which all themes by default have got access.

per-subject access command list (that's, one list for every subject tells all of the items to which that subject has admittance)

per-object access management list (that's, one list for every object tells all the subjects who have access to that thing)

access control matrix

capability

15. Assume a per-subject entry control list is used. Deleting an subject in such a system can be inconvenient because all modifications must be made to the control listings of all subject matter who did get access to the object. Recommend an alternative, less expensive means of controlling deletion.

16. File gain access to control relates mainly for the secrecy aspect of security. What's the relationship between an accessibility control matrix plus the integrity of the items to which entry is being operated?

IDOL

17. One characteristic of an capability-based protection method is the potential of one method to move a copy of your capability to another process. Describe a situation in which one process can transfer a capacity to another.

18. Describe a system by which an operating system can enforce limited transfer of functions. That is, procedure A might send a capability to method B, but A wants to stop B from transferring the capability to any other processes.

Your design should include a explanation of the actions to be carried out by A and B, as well as the activities conducted by and the information maintained because of the operating system.

19. Listing two disadvantages of using real separation in a computing system. Record two drawbacks of making use of temporal separation in the computing system.

20. Explain why asynchronous I/O activity is a difficulty with many memory space protection schemes, like basic/bounds and paging. Suggest a solution to the issue.

21. Suggest an efficient scheme for sustaining a per-user defense scheme. That is, the system keeps one website directory per user, and that directory lists all the objects to that your user is authorized access. Your design and style should address the needs of a system with 1000 customers, of whom only 20 are lively at any time. Each user has an regular of 200 permitted objects; you can find 50,000 full objects in the system.

4.7 References

1. Security in Computing, Fourth Edition By Charles P. Pfleeger - Pfleeger Consulting Group, Shari Lawrence Pfleeger - RAND Corporation Publisher: Prentice Hall

2. Cryptography and Network Security - Principles and Practice fifth edition Stallings William Publisher: Pearson

IDOL

3. Cryptography And Network Security 3rd Edition behrouz a forouzan and debdeepmukhopadhyay 3/E Publisher: McGraw Hill Education

4. Cryptography and Network Security, 3e AtulKahate Publisher: McGraw Hill

IDOL

Chapter 5. Designing Trusted Operating Systems

IDOL

5.0 Introduction

In this particular chapter

What makes a great operating system "secure"? Or perhaps "trustworthy"?

How are respected systems designed, and which in turn of those design rules carry over naturally in order to other program development responsibilities?

How do we produce "assurance" of the correctness of any trusted operating program?

Operating systems will be the excellent providers of security inside of computing systems. They assistance many programming capabilities, enable multiprogramming and sharing regarding resources, and enforce limitations on program and consumer behavior. Because they have got such power, operating devices will also be targets for assault, because breaking through the particular defenses of your operating technique gives access to the particular secrets of computing devices.

We all say that a running system is trusted in case we have confidence which it provides these four providers consistently and effectively. Within this chapter, we take the particular designer's perspective, viewing a new trusted operating-system in words of the design and even function of components of which provide security services. The initial four sections of this particular chapter correspond to typically the four major underpinnings regarding a trusted os:

Plan. Every system could be referred to by its requirements: assertions of what the method should do and exactly how it should take action. The operating system's security needs are a set involving well-defined, consistent, and implementable rules that have recently been clearly and unambiguously stated. If the operating program is implemented to satisfy these requirements, it fulfills the user's expectations. In order to ensure that the demands are clear, consistent, plus effective, the operating method usually follows an explained security policy: a collection of rules that formulate what is to end up being secured and why. We all begin this chapter by simply studying several security plans for trusted operating devices.

Model. To create a trusted operating system, the particular designers should be confident that will the proposed system may meet its requirements whilst

protecting appropriate objects and even relationships. They usually start off by constructing a unit of the environment to become secured. The model is truly a representation of the coverage the operating system will certainly enforce. Designers compare typically the model together with the system specifications to make sure of which the general system functions will be not compromised or degraded by the security demands. Then, they study various ways of enforcing of which security. In the 2nd section of this chapter, all of us consider several different designs for os security.

Design. After having selected some sort of security model, designers select a means to put into action it. Thus, the style involves both what the particular trusted operating system is usually (that is, its designed functionality) and how that is to be created (its implementation). Another main section of this phase addresses choices to end up being made during development regarding a trusted operatingsystem.

Trust.Because the operating program plays a central position in enforcing security, many of us seek some basis for believing that that will meet our anticipation. Our trust in the device is rooted in two factors: features (the os has got all the necessary operation needed to enforce typically the expected security policy) plus assurance (the operating program has been implemented throughout such a way that will we have confidence it will eventually enforce the security coverage correctly and effectively). Inside the fourth part associated with this chapter, we check out what makes a certain design or implementation deserving of trust.

The part ends which includes examples of genuine trusted systems. Several of these kinds of systems have been published, and more are underneath development. In some instances, the secure systems had been originally suitable for security; inside of others, security features have been added to existing working systems. Our examples demonstrate that both approaches may make a secure running system.

5.1. What Is a Trusted System?

Before we commence to examine a reliable operating system at length, let us seem more carefully in the terminology involved with understanding and explaining trust. What would it not take for all of us to take into account something secure? The term secure displays a dichotomy: Something is definitely either protected or not safe and sound. If secure, it will withstand all disorders, nowadays, tomorrow, and a hundred years from now. And when we declare that it is safe and sound, you either take our assertion (and purchase and utilize it) or reject it (and frequently do

not utilize it or utilize it but usually do not trust it). So how exactly does security change from top quality? If we declare that something is fine, you are significantly less thinking about our claims and much more interested in a target appraisal of if the thing fits your efficiency and functionality necessities. From this point of view, security is one element of goodness or top quality; you may elect to balance security and safety with other attributes (such as for example speed or ease of use) to choose a system that's best, given the options you might have. In particular, the machine you develop or select could be pretty good, though it may possibly not be as safe and sound as you desire it being.

We claim that software will be trusted application if we realize that the program code has become rigorously designed and analyzed, supplying us purpose to believe that the program code does what it really is expected to perform and nothing extra. Typically, trusted program code could be a foundation which other, untrusted, program code runs. That's, the untrusted system's high quality depends, partly, on the trustworthy code; the reliable program code establishes the baseline for security and safety of the entire system. Specifically, an operating system can be respected software if you find a base for trusting it correctly regulates the accesses of elements or systems manage from it. For instance, the operating system might be likely to limit consumers' accesses to particular files.

To believe any software, we bottom our confidence in rigorous examination and testing, searching for certain key features:

Functional correctness. This program does what it really is supposed to, also it works correctly.

Enforcement of integrity. Even though presented erroneous instructions or instructions from unauthorized consumers, the program sustains the correctness of the info with which they have contact.

Limited opportunity: This program is permitted to access secure info, but the admittance is reduced and neither the accessibility rights nor the info are transferred along to some other untrusted applications or back again to an untrusted caller.

Appropriate confidence degree. The program has become examined and ranked at a qualification of trust befitting the type of data and surroundings in which it really is to be utilized.

Trusted software is frequently used as a safe method for general users to gain access to sensitive data. Trustworthy programs are accustomed to performing confined (risk-free) procedures for customers without permitting users to possess immediate access to sensitive files.

Security professionals would rather speak of respected instead of safe and sound operating systems. A reliable system connotes one which meets the designed security requirements, can be of high adequate good quality and justifies the user's self-confidence in that good quality. That is, faith is perceived because of the system's recipient or user, certainly not by its programmer, designer, or maker. As an end user, may very well not have the ability to evaluate that confidence directly. You might trust the look, a professional examination, or the view of an appreciated colleague. However, in the end, it really is your duty to sanction the amount of trust you need.

You should realize that there may be degrees of putting your trust in; unlike security, have faith in isn't a dichotomy. For instance, you trust particular friends with profound secrets, nevertheless, you trust others and then provide you with the period. Trust is really a characteristic that usually grows as time passes, relative to evidence and expertise. For instance, lenders increase their rely upon borrowers because the borrowers repay loan products as expected; debtors with good have confidence in (credit score) details can borrow much larger amounts. Finally, confidence is earned, not necessarily stated or conferred. The comparability in Table

Secure	**Trusted**

5.1.

highlights a few of these distinction

Table 5.1. Qualities of Security and Trustedness

Either-or: Something either is or is not secure.	Graded: There are degrees of "trustworthiness."
Property of presenter	Property of receiver
Asserted based on product characteristics	Judged based on evidence and analysis
Absolute: not qualified as to how used, where, when, or by whom	Relative: viewed in context of use
A goal	A characteristic

The adjective trusted appears often in this section, as in trustworthy process (an activity that can influence system security, or perhaps a process whose inappropriate or harmful execution is with the capacity of violating system security and safety policy), trusted merchandise (an evaluated and authorized product), trusted computer software (the program portion of something that may be relied upon to enforce protection policy), trusted processing base (the group of all protection systems within a processing system, including equipment, firmware, and computer software, that collectively enforce a unified stability policy over something or method), or respected system (something that employs enough hardware and computer software integrity measures to permit its make use of for processing hypersensitive details). These definitions will be paraphrased from. Popular to these definitions will be the concepts of

enforcement of security policy

sufficiency of actions and mechanisms

evaluation

In studying respected os's, we examine tightly why is trustworthy.

To know an operating system keeps the security and safety we expect, we should have the ability to state its protection policy. A safety policy is really a statement in the security we assume the machine to enforce. An operating system (or any piece of a reliable system) could be trusted only with regards to its security insurance policy; that is, towards the security needs the machine is likely to satisfy.

5.2. Security Policies

We start off our research of security insurance policy by examining armed service security policy since it has been the foundation of much respected operating system advancement and is rather easy to express precisely. Next, we proceed to security insurance policies that commercial organizations might adopt.

Military Security and Safety Policy

The military security plan is dependant on protecting classified facts or data. Each little bit of information is rated at a specific sensitivity level, such as for example unclassified, restricted, confidential, secret, or ” top secret.". The rates or degrees form a hierarchy, plus they reflect a growing order of level of sensitivity, as revealed in Figure 5.1. That's, the info at a confirmed level is extra sensitive compared to the information in the particular level below it and not as much delicate than in the particular level above it. For instance, restricted information is certainly more hypersensitive than unclassified but significantly less sensitive than private. We are able to denote the level of sensitivity of an object O by rank $_{O}$. In the others of this section, we believe these five level of sensitivity levels.

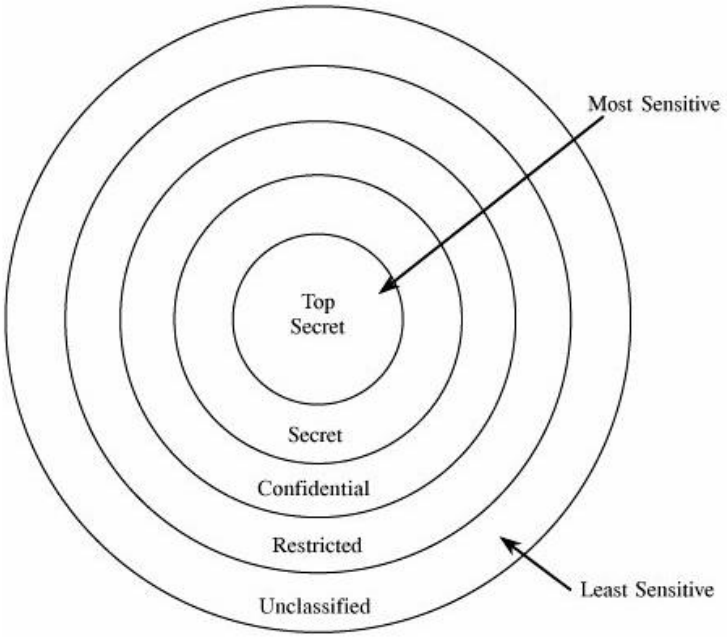


Figure 5.1. Hierarchy of Sensitivities.

Information access is bound with the **need-to-know** concept: Usage of sensitive data is usually allowed and then subjects who need to find out those data to execute their careers. Each little bit of classified information could be associated with a number of projects, named compartments, describing the topic matter of the info. For instance, the alpha task may use magic formula information, simply because may the beta task, but employees on alpha don't need access to the info on beta. Quite simply, both projects work with secret details, but each is fixed to only the trick information necessary for its particular task. In this manner, compartments support enforce need-to-know constraints so that persons obtain access and the information that's highly relevant to their work. A compartment can include information of them costing only one sensitivity levels, or it could cover info at several levels of sensitivity levels. The partnership between compartments and the level of sensitivity levels is proven in Figure 5.2.

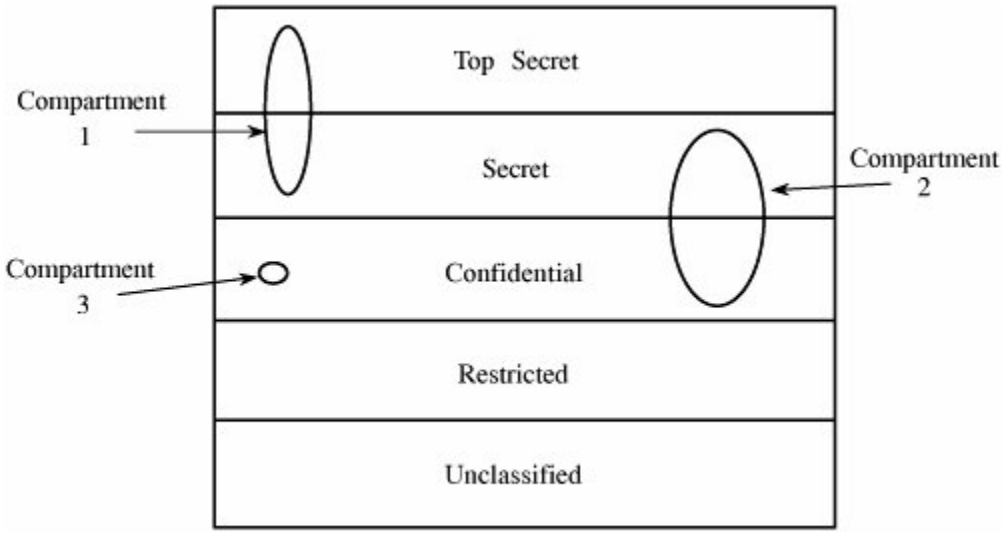


Figure 5.2. Compartments and Sensitivity Levels.

We are able to assign names to recognize the compartments, such as for example snowshoe, crypto, and Sweden. An individual piece of details could be coded with zero, one, two, or even more compartment names, with regards to the groups to which it relates. The connection between facts and compartments will be shown in Figure 5.3. For instance, one little bit of information might be a list of magazines

on cryptography, whereas another may identify the growth of snowshoes in Sweden. The area of this very first piece of details is crypto; the second reason is snowshoe, Sweden.

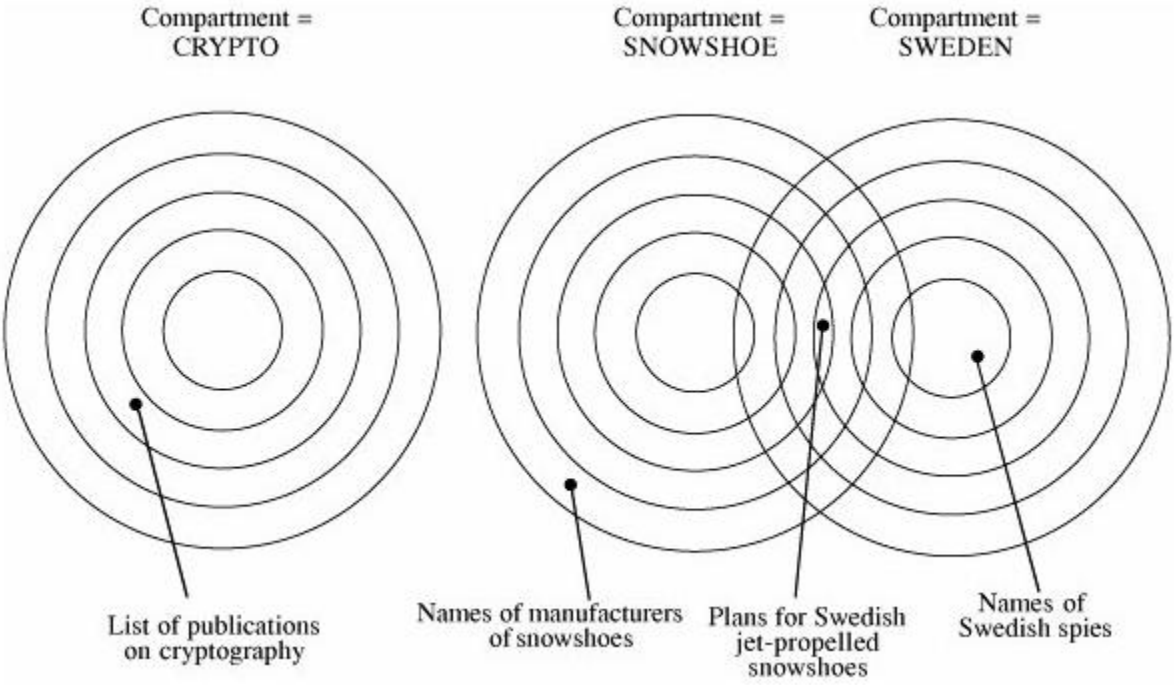


Figure 5.3. Association of Information and Compartments.

The collaboration <rank; compartments> is named the **class**or **classification** of a bit of fact. By designating details in this manner, we are able to enforce need-to-know both by safety measures levels and by subject.

A person seeking usage of sensitive information should be cleared. A clearance can be an indication a person is respected to access info up to a certain degree of sensitivity which the person must know certain types of sensitive data. The clearance of a topic is portrayed as a mix <rank; compartments>. This collaboration has a similar form because of the classification of a bit of information

Now we have a tendency to introduce a relation ≤ known as dominance, on the sets of sensitive objects and subjects. For an issue s ANd an object o,

$s \leq o$ if and only if
$rank_s \leq rank_o$ and
$compartments_s \subseteq compartments_o$

We say that o dominates s (or s is dominated by o) if s ≤o; the relation≥ is that the opposite. Dominance is employed to limit the sensitivity and content of data a theme will access. a theme will browse Associate in the Nursing object given that

the clearance level of the topic is a minimum of as high as that of the knowledge and

the subject contains a got to understand all compartments that the knowledge is assessed

These conditions square measure such as language that the topic dominates the item.

To see however the dominance relation works, think about the concentrical circles in Figure 5-3. in step with the relationships delineated there, data classified as <secret;> can be browsed by somebody cleared for access to <top secret;> or <secret;>, however not by somebody with a <top secret;> clearance or somebody cleared for <confidential;> or <secret;>.

Military security enforces each sensitivity needs and need-to-know needs. Sensitivity needs square measure referred to as stratified needs as a result of they mirror the hierarchy of sensitivity levels; need-to-know restrictions square measure unranked as a result of compartments don't essentially mirror a hierarchical data structure. This combinable model is suitable for a setting within which access is stiffly controlled by a central authority. Someone, typically referred to as a security officer, controls clearances and classifications, that aren't typically up to people to change.

Commercial enterprises possess significant security problems. They fret that professional espionage will show information to challengers about services under development. Moreover, corporations tend to be eager to guard information about the facts of corporate fund. So despite the fact that the commercial planet is usually much less rigidly and significantly less hierarchically structured compared to the military globe, we still discover lots of the same principles in commercial safety policies. For instance, a large group, like a corporation or perhaps a university, could be divided into categories or sections, each in charge of several disjoint tasks. There can also be some corporate-level duties, such as data processing and personnel exercises. Data things at any degree may have various degrees of

awareness, such as open, proprietary, or inner; here, the titles can vary greatly among organizations, no widespread hierarchy applies.

Let us expect that public details is less delicate than proprietary, which is less very sensitive than internal. Tasks and departments are usually fairly well segregated, with some overlap as men and women work on several projects. Corporate-level duties have a tendency to overlie tasks and sections, as people through the entire corporation might need accounting or staff data. However, perhaps corporate data could have degrees of level of sensitivity. Tasks themselves may expose a qualification of level of sensitivity: Workers on task old-standby haven't any need to find out about task new-product, while workers on new-product could have usage of all info on old-standby. Therefore, a commercial design of info might appear to be Figure 5.4.

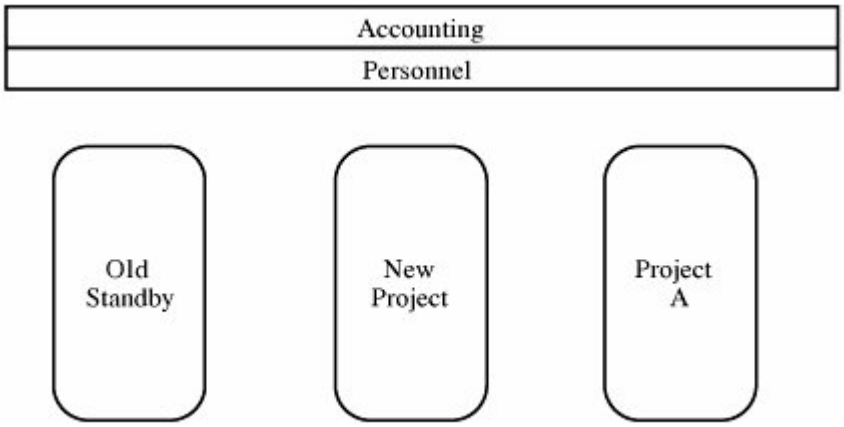


Figure 5.4. Commercial View of Sensitive Information

Two significant variations exist between professional and military info security. First, beyond your military, there's normally no formalized belief of clearances: An individual focusing on a commercial task does not need approval for job MARS access by way of a central security official. Typically, a worker isn't conferred another degree of having faith in by being granted access to inner data. Second, since there is no formal idea of clearance, the guidelines for allowing accessibility are not as much regularized. For instance, if an older manager decides a person needs usage of a bit of MARS internal files, the boss will instruct you to definitely allow the gain access to, either one-time or carrying on. Thus, there is absolutely no dominance function for some commercial information admittance since there is no formal idea of an industrial clearance.

So far, a lot of our discussion provides focused just on read entry, which addresses confidentiality in safety measures. Actually, this narrow watch is true for a lot of the existing job in computer safety measures. On the other hand, integrity and supply are at very least as essential as confidentiality in most cases. Insurance policies for integrity and accessibility are considerably less well designed than those for confidentiality, both in military and professional realms. In both instances that follow, we discover some cases of integrity concerns.

ClarkWilson Commercial Protection Policy

In many professional applications, integrity could be at least just as significant as confidentiality. The correctness of data processing records, the precision of legal job, and the correct timing of procedures are the fact of their grounds. Clark and Wilson suggested an insurance plan for what they contact well-formed transactions, that they assert are just as important within their field as is usually confidentiality within a military realm.

To understand why to consider a corporation that requests and will pay for resources. A representation from the procurement process may be this:

1. A buying clerk results in an order for your supply, sending duplicates of the buy to both suppliers as well as the receiving department.

2. The supplier boats the products, which reach the receiving division. An Obtaining clerk assessment the delivery means that the correct level of the right piece has been acquired, and warning signs a delivery kind. The delivery contact form and the initial order go directly to the accounting department.

3. The supplier directs an invoice for the accounting team. A data processing clerk compares the invoice with the initial order (concerning price along with other terms) along with the delivery type (concerning quantity and piece) and problems a check towards the supplier.

Separation of Duty

A second commercial stability policy involves parting of duty. Clark and Wilson lifted this issue within their analysis of professional security prerequisites, and Lee and Nash and Poland put into the concept.

To observe how it functions, we proceed our exemplory case of small businesses ordering merchandise. In the business, several people may be authorized to concern orders, receive products, and write investigations. However, we'd not want

exactly the same person to concern the order, have the goods, and produce the check, since there is potential for maltreatment. Therefore, we may want to set up a plan that specifies that three distinct individuals matter the order, have the goods, and publish the check, despite the fact that the three may be authorized to accomplish these tasks. This essential division of tasks is called parting of duty.

Separation of obligation is commonly achieved manually through twin signatures. Clark and Wilson triples will be "stateless," and therefore a triple doesn't have a framework of prior procedures; triples are not capable of passing control data to various other triples. So, if one individual is authorized to execute procedures TP1 and TP2, the Clark and Wilson triples cannot avoid the same man or woman from undertaking both TP1 and TP2 on confirmed data item. On the other hand, it is rather easy to carry out distinctness if it's stated as an insurance plan requirement.

Brewer and Nash identified a security coverage called the Chinese language Wall that demonstrates certain commercial demands for information entry protection. The safety measures requirements reflect concerns relevant to those individuals in legal, professional medical, investment, or data processing firms who may be subject to discord of fascination. A issue of interest is present when a particular person in one corporation can obtain vulnerable information about persons, products, or expert services in competing organizations.

The security insurance plan develops on three degrees of abstraction.

Objects. At the cheapest level are primary objects, such as for example files. Each record contains data concerning only 1 company.

Company categories. At another level, all things concerning a specific company will be grouped together.

Conflict sessions. At the best level, all sets of objects for contending companies happen to be clustered.

With this design, each item belongs to a distinctive company party, and each provider group is within a unique turmoil class. A turmoil class may comprise a number of company groups. For instance, suppose you're an advertising business with clients in a number of fields: chocolate firms, lenders, and airlines. You might like to store info on chocolate firms Suchard and Cadbury; on banking companies Citicorp, Deutsche Bank or investment company, and Credit rating Lyonnais; and on flight SAS. You intend to prevent your personnel from inadvertently disclosing information to litigant about this client's competitors, which means you establish

the guideline that no worker will know vulnerable information about contending companies. Utilizing the Chinese Wall structure hierarchy, you'll form six corporation groups (one for every provider) and three discord classes: Suchard, Cadbury, Citicorp, Deutsche Bank, Credit Lyonnais, and SAS.

The hierarchy leads a simple accessibility control coverage: An individual can access any data so long as that person hasn't accessed information from the different provider in exactly the same conflict class. That's, access is permitted if either the thing requested is at the same corporation group being an object which has previously been reached or the thing required belongs to a turmoil class which has never before happen to be accessed. Inside our example, initially, it is possible to access any items. Suppose you study from a document on Suchard. A succeeding request for usage of any bank or even to SAS will be given, but a question to gain access to Cadbury files will be denied. The next gain access to, or SAS files, does not have an impact on foreseeable future accesses. But in the event that you then obtain a document on Credit rating Lyonnais, you'll be blocked from long term accesses to Deutsche Loan provider or Citicorp. In the future, as found in Figure 5.5, it is possible to access objects simply pertaining to Suchard, SAS, Credit score Lyonnais, or perhaps a newly defined issue class.

IDOL

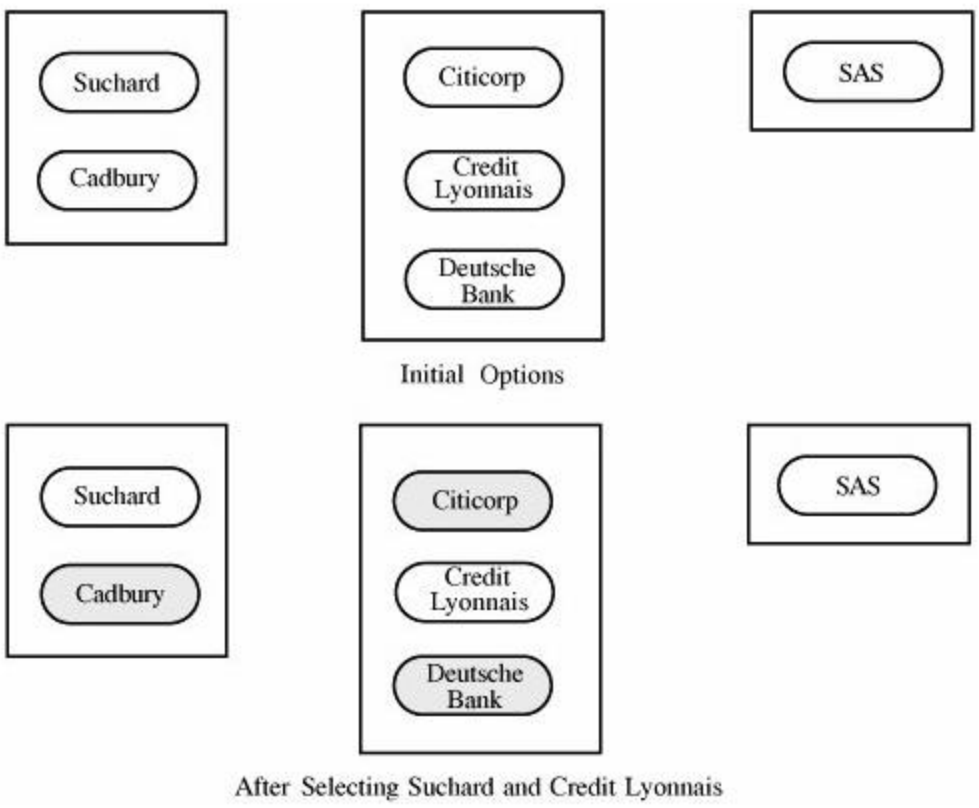


Figure 5-5. Chinese Wall Security Policy.

The Chinese Wall membrane is really a commercially motivated confidentiality policy. It really is unlike almost every other commercial insurance policies, which concentrate on integrity. Additionally, it is interesting because accessibility permissions modification dynamically: As a topic accesses some things, other objects that could previously have already been accessible are eventually denied.

5.3. Models of Security

In protection and elsewhere, styles can be used to describe, analysis, or analyze a specificsituation or partnership. McLean provides good summary of models for safety. In particular, safety models are accustomed to

test a specific insurance plan for completeness and consistency

document an insurance plan

aid conceptualize and pattern an implementation

check out whether an execution matches its requirements

Multilevel Security

Ideally, you want to build a type to represent a variety of sensitivities also to reflect the necessity to separate content rigorously from things to that they should not have admission. For instance, think of an election as well as the sensitivity of info mixed up in voting method. The names from the candidates are most likely not vulnerable. If the outcomes have not however been produced, the title of the champion is somewhat hypersensitive. If one prospect obtained an embarrassingly reduced amount of votes, the vote count up may be even more sensitive. Finally, just how a particular personal voted is incredibly sensitive. Users may also be ranked by the amount of awareness of data to that they can have gain access to.

For obvious factors, the military is rolling out extensive processes for securing facts. A generalization in the military style of information security in addition has been adopted like a model of files security in a operating-system. Bell and La Padula have been first to spell it out the properties with the military version in numerical notation, and Denning earliest formalized the composition of this version. In 2005, Bell went back to the initial model to point out its factor to computer safety. He observed which the model demonstrated the necessity to understand security specifications before beginning program design, build safety measures into definitely not onto the machine, develop a protection toolbox, and style the system to safeguard itself. The generalized version is named the lattice style of security and safety because its components form a numerical structure known as a lattice. (Notice Sidebar 5-1.) On this section, we explain the military case in point and then work with it to describe the lattice design.

The military safety model will be representative of a far more general scheme, named a lattice. The dominance connection ≤ defined within the military model may be the relation with the lattice. The relationship ≤ can be transitive and antisymmetric. The biggest component of the lattice may be the classification <top secret; all compartments> , and the tiniest element can be <unclassified; no compartments> ; both of these components respectively dominate and so are dominated by all factors. Therefore, the armed service model is really a lattice.

Many other buildings are lattices. For instance, we noted previously that a professional security coverage may contain files sensitivities such as for example public, amazing, and internal, along with the natural purchasing that public files

are less vulnerable than proprietary, that happen to be less very sensitive than inner. These three quantities also shape a lattice.

Security specialists contain chosen to basic security systems on the lattice since it naturally represents improving degrees. A security and safety system made to implement lattice products may be used in a armed forces environment. However, it is also used in industrial environments with unique labels for your degrees of level of sensitivity. Therefore, lattice representation of awareness levels pertains to many computing circumstances.

Sidebar 5-1: What Is a Lattice?

Alattice is really a mathematical design of elements structured by a connection among them, symbolized by way of a relational operator. We utilize the notation ≤ to denote this connection, and we declare that b ≥a way a similar thing like a≤ b. A connection is named a partial ordering when it's both transitive and antisymmetric. These words mean that for each three factors a, b, and c, the next two rules maintain:

transitive: In case a≤ b and b≤ c, a≤ c

antisymmetric: In case a≤ b and b ≤a, a = b

In the lattice, don't assume all pair of factors needs to get comparable; that's, there could be components a and b that neither a ≤b nor b≥ a. Even so, every couple of elements has an upper bound, namely, a component at least mainly because large as (≥) both a and b. Quite simply, despite the fact that a and b could be noncomparable under≤ , inside a lattice there's an upper sure element u in a way that a≤ u and b ≤u. Moreover, in a very lattice, every couple of elements possesses a lower bound, a component l dominated by both a and b; that's, l ≤ a and l ≤ b.

Think about the lattice in Figure 5.6, which signifies all aspects of the quantity 60. The relational operator presents the partnership "is really a element of." So, the notation a≤ b implies that a divides b or, equivalently, b is really a multiple of an. The lattice exhibits us that the quantity 60 dominates all the components; 12 dominates 4, 6, 2, 3, and 1; 20 dominates 4, 10, and 5; etc. We can furthermore note that some elements aren't comparable. For example, 2 and 5 aren't comparable and they are not directly linked by lines within the diagram.

IDOL

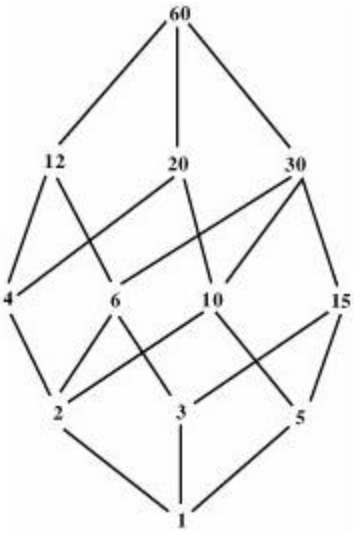


Figure 5.6. Sample Lattice.

Lattices are ideal for depicting relationships, plus they appear mostly when the romance shows an improvement in power, material, or worth. But many normal relationships form just half of a lattice. Within the relationships "is significantly less than," "is really a subset of," "reports to (for personnel)," or "is really a descendant of," there's a unique least higher bound (for instance, a standard ancestor) however, not a greatest lower destined for each match.

The Bell and La Padula design is really a formal description with the allowable pathways of information movement in a risk-free technique. The model's objective is to distinguish allowable conversation when retaining secrecy is essential. The model continues to be used to identify security prerequisites for devices concurrently handling files at different awareness levels. This design is really a formalization on the military security insurance plan and was main towards the U.S. Section of Defense's analysis criteria, described soon after in this section.

We are thinking about secure information moves because they illustrate acceptable links between topics and items of different degrees of sensitivity. One goal for security-level evaluation is to allow us to create systems that may accomplish concurrent computation on information at two unique sensitivity levels. For instance, we may desire to use one device for top-secret and private data at exactly the same time. The programs control top-secret data will be prevented from seeping top-secret data towards the confidential data, plus the confidential users will be prevented from being able to access the top-secret files. So, the BellLa

Padula unit is useful because the basis for the look of methods that handle information of numerous sensitivities.

To understand the way the BellLa Padula style works, look at a security program with the next properties. The machine covers a couple of themes S and a couple of things O. Each content s in S and each thing o in O includes a fixed security school C(s) and C(o) (denoting clearance and classification degree). The protection classes are purchased by a connection . (Be aware: The sessions may contact form a lattice, despite the fact that the BellLa Padula type can connect with even less limited cases.)

Two attributes characterize the safe flow of facts.

Simple Security Property. A topic s could have read usage of an thing o only when $C(o) \leq C(s)$.

In the military services model, this home says how the security course (clearance) of a person receiving a little bit of information should be at least mainly because high because the category (classification) of the info.

***- Property (named the " star property").** A topic s who have read usage of an thing o could have write usage of an subject p only when $C(o) \leq C(p)$.

In the armed forces model, this house says which the contents of any sensitive object could be written and then objects at the very least as high.

In the armed service design, one interpretation on the *-property is a person obtaining data at one degree may move that facts along and then people at degrees no less than the amount of the info. The *-residence avoids write-down, which happens when a area of interest with usage of high-level data exchanges that files by publishing it into a low-level object.

Basically, the *-real estate requires a person receiving data at one levels not talk to men and women cleared at degrees lower than the amount of the informationnot also about the conditions! This example highlights that this real estate is more powerful than necessary to assure security; exactly the same is also correct in computing devices. The BellLa Padula type is extremely conventional: It guarantees security also at the trouble of user friendliness or other components.

The implications of the two properties are usually shown in Figure 5.7. The classifications of subject matter (symbolized by squares) and items (displayed by circles) will be mentioned by their postures: Because the classification of

something increases, it really is shown higher within the figure. The move of information is normally horizontal (to and from exactly the same degree) and upwards (from lower ranges to raised). A downward circulation is acceptable only when the extremely cleared subject will not move any high-sensitivity info for the lower-sensitivity object.

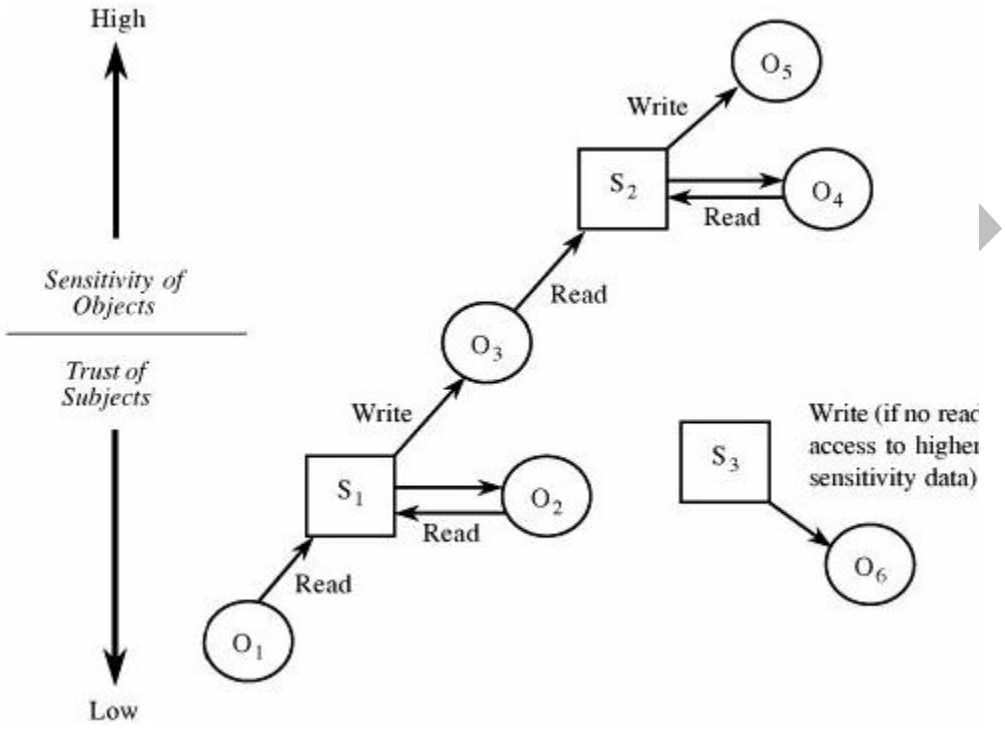


Figure 5.7. Secure Flow of Information.

For computing methods, downward movement of information is definitely difficult just because a computer plan cannot readily separate between having examine a bit of information and possessing read a bit of information that inspired what was in the future created. (McLean in job linked to Goguen and Meseguer ,gifts an interesting counter-top for the *-real estate of Bell and La Padula. He implies considering noninterference, which may be loosely referred to as tracing the consequences of inputs on outputs. If we are able to trace all result effects, we are able to ascertain conclusively whether a specific low-level output was basically "contaminated" with high-level suggestions.)

Biba Integrity Model

The BellLa Padula style applies and then secrecy of facts: The type identifies paths which could lead to incorrect disclosure of info. Nevertheless, the integrity of info is important, also. Biba created a design for preventing incorrect modification of files.

The Biba version may be the counterpart (often called the double) in the BellLa Padula unit. Biba identifies "integrity ranges," that happen to be analogous towards the sensitivity degrees of the BellLa Padula type. Subjects and items are ordered by an integrity classification structure, denoted I(s) and I(o). The properties are

Simple Integrity Property. Content s can adjust (include write usage of) subject o only when I(s) ≥I actually(o)

Integrity *-Property or home. If content s has study access to thing o with integrity stage I(o), s might have write usage of object p only when I(o) ≥I(p)

These two regulations cover untrustworthy details in an all natural way. Imagine John may be untruthful often. If John can make or improve a document, other folks should distrust the reality of the assertions in that doc. Consequently, an untrusted issue who may have write usage of an object minimizes the integrity of this object. Similarly, folks are rightly skeptical of a written report predicated on unsound evidence. The reduced integrity of an source object suggests low integrity for just about any object in line with the source object.

This unit addresses the integrity matter the fact that BellLa Padula design ignores. Even so, in doing this, the Biba style ignores secrecy. Secrecy-based protection systems have already been much more totally studied than contain integrity-based systems. The existing trend would be to sign up for secrecy and integrity problems in security devices, although no generally accepted formal products achieve this bargain.

Lampson and Graham and Denning created the idea of a formal program of protection regulations. Graham and Denning produced a model getting generic protection real estate. This model types the basis for just two later types of security systems.

The GrahamDenning design operates on a couple of subjects S, a couple of objects O, a couple of privileges R, and an gain access to control matrix A good. The matrix provides one row for every subject and something column for every issue and each thing. The privileges of a topic on another area of interest or an subject are shown with the contents of some the matrix. For every object, one issue

specified the "owner" features special rights; for every subject, another theme specified the "controller" offers special rights.

The GrahamDenning type offers eight primitive safeguard rights. These privileges are usually phrased as orders that may be issued by themes, with results on other subject matter or objects.

Create object enables the commanding at the mercy of introduce a fresh object to the machine.

Create Subject, delete thing, and delete topic area have the related aftereffect of creating or destroying a topic or object.

Read access rightallows a topic to look for the current access protection under the law of a topic to an subject.

Grant access right suited allows who owns an object to mention any access protection under the law for an item to another issue.

Delete access right allows a topic to delete the right of another area of interest for an thing, so long as the deleting issue either are the owners of the thing or controls the topic from which accessibility should be removed.

Transfer access right allows a topic to transfer among its rights to have an object to some other subject. Each best suited could be transferable or nontransferable. In case a subject gets a transferable ideal, the subject may then transfer that correct (either transferable or definitely not) to some other subjects. In case a subject obtains a nontransferable perfect, it can utilize the proper but cannot exchange that to other subjects.

These rules happen to be shown in Table 5.2 which ultimately shows prerequisite situations for performing each command and its own effect. The accessibility control matrix is really a [s,o], where s is really a area of interest and o can be an object. The topic executing each control is certainly denoted x. A transferable best suited will be denoted r*; a nontransferable ideal is prepared r.

Table 5.2. Protection System Commands.

Command	**Precondition**	**Effect**
Create object o		Add column for o in A; place owner in A[x,o]

Create subject s		Add row for s in A; place control in A[x,s]
Delete object o	Owner in A[x,o]	Delete column o
Delete subject s	Control in A[x,s]	Delete row s
Read access right of s on o	Control in A[x,s] or owner in A[x,o]	Copy A[s,o] to x
Delete access right r of s on o	Control in A[x,s] or owner in A[x,o]	Remove r from A[s,o]
Grant access right r to s on o	Owner in A[x,o]	Add r to A[s,o]
Transfer access right r or r* to s on o	r* in A[x,o]	Add r or r* to A[s,o]

This group of rules supplies the properties essential to model the entry control mechanisms of your protection system. For instance, this system can stand for a reference screen or a technique of posting between two untrustworthy, mutually dubious subsystems.

HarrisonRuzzoUllman Results

Harrison, Ruzzo, and Ullman suggested a variation around the GrahamDenning unit. This revised version answered several inquiries concerning the forms of protection confirmed system can provide. Suppose you're about to work with a particular operating-system and you wish to know if a granted user can ever before be granted a particular kind of entry. For example, you might be establishing protection amounts in Home windows or MVS. You create the access adjustments and then consult whether end user X will actually get access to subject Y. The three experts developed their style so that we may have the ability to answer questions such as this one.

The **HarrisonRuzzoUllman** type (referred to as the HRU design) is dependant on orders, where each order involves situations and primitive functions. The structure of an command is really as follows.

command *name*(o_1,o_2,...,o_k)
    if r_1 in A[s_1,o_1] and
    r_2 in A[s_2,o_2] and
    ...
    r_m in A[s_m,o_m]
    then
    op_1
    op_2
    ...
    op_n
end

This command can be structured such as a procedure, with variables o1 through okay. The notation of this HRU model can be slightly not the same as the GrahamDenning unit; in HRU every topic area is an item, too. Hence, the columns in the access management matrix are the topics and all of the objects that aren't subjects. Because of this, all the guidelines of a demand are tagged o, although they may be either things or nonsubject items. Each r is really a generic right, as with the GrahamDenning version. Each op is really a primitive operation, described in the list following. The admittance matrix is proven in Table 5.3.

Table 5.3. Access Matrix in HRU Model.

	Objects					
Subjects	**S_1**	**S_2**	**S_3**	**O_1**	**O_2**	**O_3**
S_1	Control	Own, Suspend, Resume		Own	Own	Read, Propagate

S_2		Control			Extend	Own
S_3			Control	Read, Write	Write	Read

The primitive businesses op, much like those of the GrahamDenning type, are the following:

- create subject s
- create object o
- destroy subject s
- destroy object o
- enter right r into A[s,o]

delete right r from A[s,o]The interpretations of the operations will be what their brands imply. A **protection system** is really a set of subjects, objects, rights, and commands

Harrison et al. demonstrate these operations are satisfactory to model various examples of security systems, like the Unix protection device and an indirect entry mode created by Graham and Denning. So, just like the GrahamDenning style, the HRU unit can signify "reasonable" interpretations of security.

Two important effects produced by Harrison et al. own key implications for makers of protection techniques.

The first derive from HRU implies that

Within the modeled system, where commands are limited to a single functioning each, you'll be able to decide whether confirmed subject can ever before obtain a certain to an object.

Therefore, we are able to decide (that's, we can learn beforehand) whether a low-level theme can ever get read usage of a high-level item, for example.

The second effect is fewer encouraging. Harrison et al. express thatIf commands aren't limited to one functioning each, it isn't often decidable whether confirmed protection method can confer confirmed right.

Thus, we can not determine generally whether a topic can obtain a specific to an object.

IDOL

For example, consider protection within the Unix operating-system. The Unix safeguard scheme is not at all hard; other protection devices are more sophisticated. As the Unix protection design requires several operation per command word inside the HRU model, there may be no general treatment to find out whether a particular access right could be given to a topic.

The HRU effect is essential but bleak. Actually, the HRU outcome can be lengthened. There could be an algorithm to choose the access correct question for a specific collection of defense systems, but also thousands of algorithms cannot choose the access appropriate question for several protection systems. Nevertheless, the negative benefits do not declare that no selection process exists for just about any protection system. Actually, for certain particular protection systems, it really is decidable whether confirmed access right could be conferred.

Subsequently, the HRU email address details are negative for basic procedures but usually do not rule out the chance of making judgements about particular safeguard systems.

TakeGrant Systems

One final style of a protection program may be the takegrant system, presented by Jones and widened by Lipton and Snyder.

This model has got simply four primitive function : create, revoke, take, and grant. Create and revoke act like operations from GrahamDenning and HRU styles; take and offer are new forms of operations. These procedures are presented nearly all naturally by using graphs.

As in various other systems, permit S be considered a set of themes and O be considered a set of items; objects could be either energetic (things) or unaggressive (nonobject things). Permit R be considered a set of protection under the law. Each subject matter or object can be denoted by way of a node of an graph; the privileges of a specific subject to a specific object will be denoted by way of a labeled, directed border from the topic to the thing. Figure 5.8 reveals a good example of subject, object, and rights.

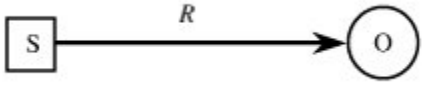


Figure 5.8. Subject, Object, and Rights.

Let s function as subject performing each one of the functions. The four businesses are thought as follows. The consequences of these procedures are displayed in Figure 5.9.

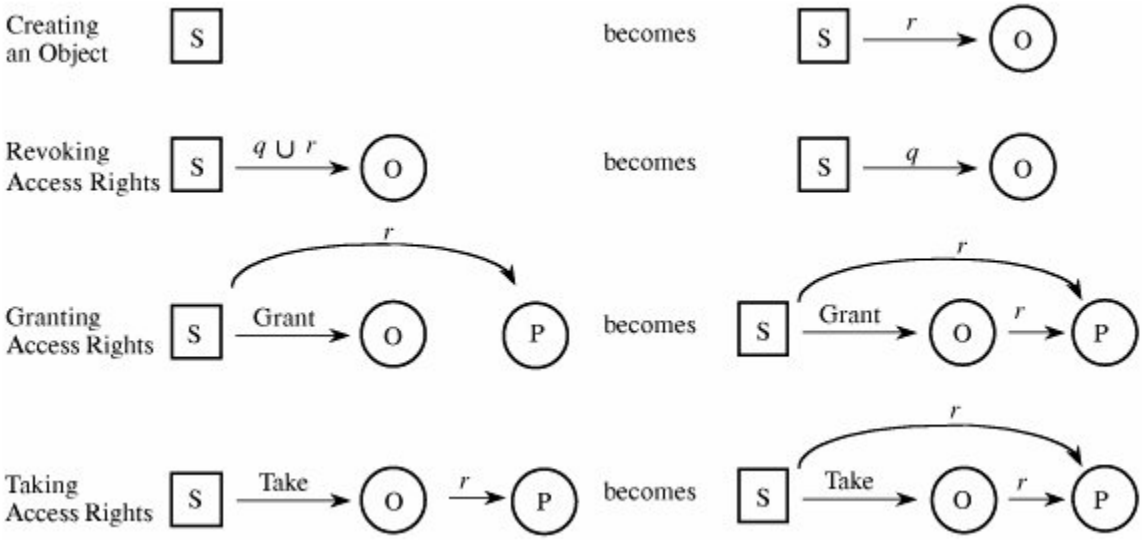


Figure 5.9. Creating an Object; Revoking, Granting, and Taking Access Rights.

Create(o,r). A fresh node with content label o is put into the graph. From s to o is really a directed border with content label r, denoting the privileges of s on o

Revoke(o,r). The privileges r happen to be revoked from s on o. The advantage from s to o seemed to be tagged q r; the brand is substituted by q. Informally, we point out that s can revoke its protection under the law to accomplish r on o.

Offer(o,p,r). Subject matter s grants or loans to o admittance privileges r on p. A particular right is give. Content s can give to o entry privileges r on p only when s has offer privileges on o and s possesses r privileges on p. Informally, s can offer (talk about) some of its privileges with o, so long as s gets the right to offer privileges to o. An advantage from o to p can be added, with brand r.

Take(o,p,r). Subject matter s requires from o admittance protection under the law r on p. A particular right is get. Subject s may take from o accessibility protection under the law r on p only when s has get directly on o and o features r protection under the law on p. Informally, s may take any protection under the law o has, so long as s gets the right to consider privileges from o. An advantage from s to p will be added, with tag r.

This group of operations is also shorter compared to the functions of either of both previous models. On the other hand, take and give are more intricate rights.

Snyder implies that in this technique certain protection inquiries are decidable; on top of that, they're decidable in realistic (significantly less than exponential) period. In, Snyder considers two queries:

1. Can we make a decision whether confirmed subject can discuss an thing with another theme?

2. Can we determine whether confirmed subject can grab usage of an item from another subject matter?

Clearly, they are important queries to answer in regards to a protection program, for they present whether the accessibility control mechanisms happen to be risk-free against unauthorized disclosure.

The solution to Snyder's 1st question is usually yes. Sharing may appear only if other subjects together possess the desired use of the thing and the initial subject is linked to each one of the group of different subjects by way of a path of corners having a specific kind. An algorithm that picks up sharability runs with time proportional to how big is the graph of this case.

Snyder also right answers the second concern affirmatively, in times heavily influenced by the capability to share. Consequently, an algorithm can choose whether access could be stolen by immediate interest the algorithm to choose sharability.

Landwehr highlights that this takegrant style assumes the most severe about customers: In case a user can give access privileges, the unit assumes that an individual will. Assume a user can make a data file and grant usage of it to everyone. For the reason that situation, every individual could allow usage of every thing by almost every other consumer. This worst-case assumption restricts the applicability from the model to conditions of controlled writing of information. Generally, on the other hand, the takegrant unit is useful since it identifies ailments under which a consumer can obtain usage of an object.

5.4. Trusted Operating System Design

Operating systems independently (no matter their protection constraints) have become difficult to create. They handle various duties, are at the mercy of interruptions and framework switches, and must lessen overhead in order not to

decrease individual computations and relationships. Adding the duty for stability enforcement for the operating system greatly increases the problems of building an operating-system.

Nevertheless, the necessity for effective security and safety is becoming extra pervasive, and great software engineering concepts tell us that it's better to design and style the protection in at the start than to shoehorn it in by the end. (Observe Sidebar 5-3 for much more about good design and style principles.) Therefore, this section targets the look of os's for a higher degree of protection. First, we take a look at the basic design and style of a typical multipurpose operating-system. Then, we take into consideration isolation, by which an operating-system supports both revealing and separating customer domains. We try particular at the look of an working system's kernel; the way the kernel was created suggests whether protection will be presented effectively. We research two various interpretations on the kernel, and we consider split or ring-structured patterns.

Trusted System Design Elements

Good design key points are always best for security, once we have noted over. But a number of important design principles are very particular to security and safety and needed for building a stable, trusted operating-system. These principles have already been articulated nicely by Saltzer and Saltzer and Schroeder:

Least privilege. Each end user and each course should operate utilizing the fewest privileges attainable. In this manner, the harm from an inadvertent or harmful attack is reduced.

Economy of mechanism. The design of this protection system ought to be small, easy, and straightforward. This type of protection system could be carefully examined, exhaustively tested, conceivably confirmed, and relied on.

Open design. The protection system must not be determined by the ignorance of prospective attackers; the system should be general public, based on secrecy of fairly few key products, like a password stand. An open style is also designed for extensive general public scrutiny, thereby furnishing independent verification of the look security.

Complete mediation.. Every accessibility attempt should be checked. Both immediate access attempts (demands) and endeavors to circumvent the entry checking mechanism is highly recommended, and the system should be situated such that it can't be circumvented.

Permission based. The default state ought to be denial of accessibility. A conservative artist identifies the things that needs to be accessible, instead of those that shouldn't.

Separation of privilege. Ideally, usage of objects should be determined by several condition, such as for example user authentication and also a cryptographic key. In this manner, a person who defeats one safeguard system won't have complete access.

Least common mechanism. Shared objects deliver potential stations for information stream. Systems employing actual or logical parting reduce the chance from sharing.

Simplicity. If a safeguard mechanism is simple to use, it really is unlikely to become avoided.

a multiprogramming operating-system performs several features that relate with security. To observe how, examine Figure 5.10, which illustrates how an operating-system interacts with customers, provides resources, and allocates resources.

IDOL

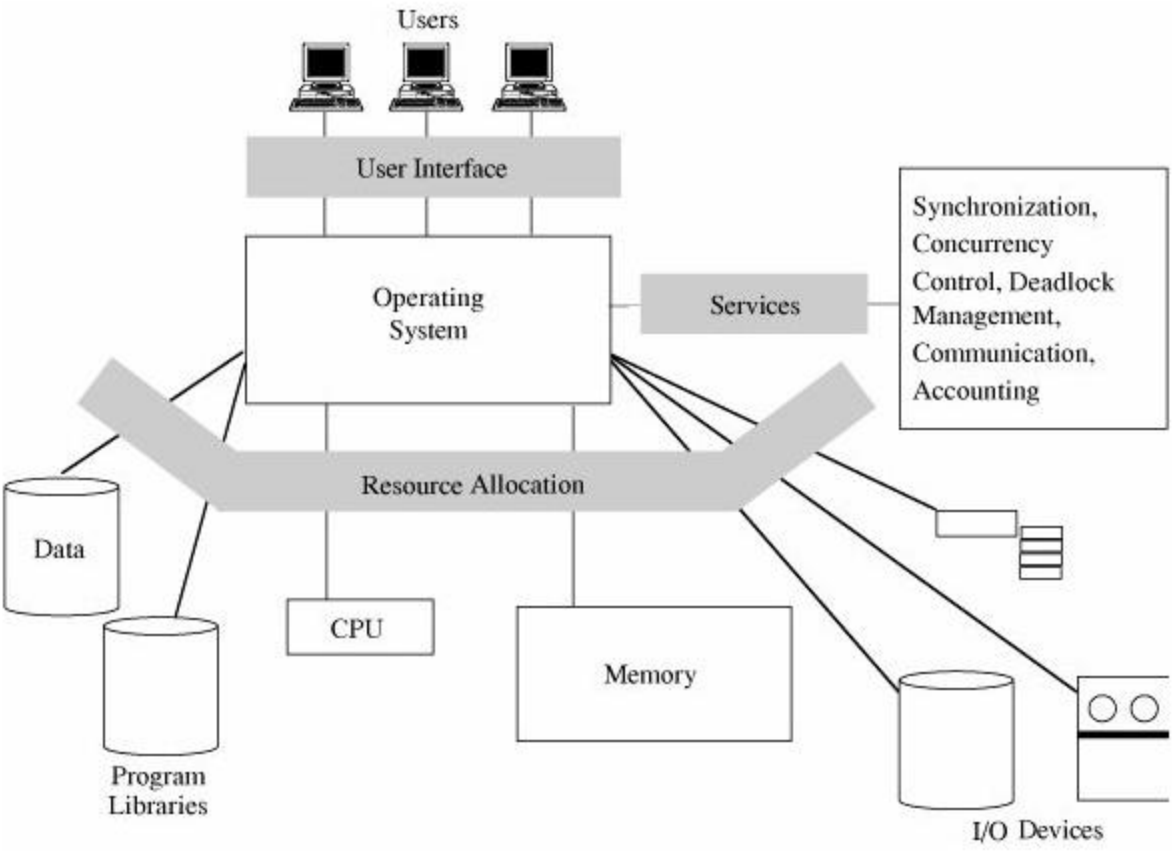


Figure 5.10. Overview of an Operating System's Functions.

We can note that the machine addresses several specific functions that require computer protection:

User authentication. The operating-system must recognize each individual who requests accessibility and must ascertain that an individual is in fact who she or he purports to become. The most frequent authentication mechanism is certainly password comparison.

Memory protection. Each user's system must manage in some of memory secured against unauthorized accesses. The safeguard will certainly avoid outsiders' accesses, also it may also manage a user's private access to limited parts of this program space. Differential stability, such as for example read, create, and execute,

could be applied to elements of a user's storage. Memory protection is normally performed by components mechanisms, such as for example paging or segmentation.

File and I/O device access control. The operating-system must protect end user and system data files from entry by unauthorized consumers. Similarly, I/O product use should be protected. Data security is usually attained by table lookup, much like an access command matrix.

Allocation and gain access to control to basic objects. Users require general objects, such as for example constructs allowing concurrency and invite synchronization. However, usage of these objects should be controlled in order that one user doesn't have a negative influence on other users. Once again, table lookup may be the common means where this protection is certainly provided.

Enforced sharing. Solutions should be distributed around users as correct. Sharing results in the necessity to promise integrity and uniformity. Table lookup, coupled with integrity controls such as for example monitors or deal processors, is frequently used to aid controlled sharing.

Guaranteed fair services. All users count on CPU usage along with other service to get provided in order that no user will be indefinitely starved from obtaining service. Equipment clocks match scheduling disciplines to supply fairness. Hardware amenities and data dining tables combine to supply control.

Interprocess communication and synchronization. Performing processes sometimes have to communicate with different processes or even to synchronize their accesses to distributed resources. Os's provide these companies by acting to be a bridge between techniques, responding to method demands for asynchronous interaction with other procedures or synchronization. Interprocess connection will be mediated by entry control tables.

Protected operating-system protection data. The operating-system must maintain information by which it could enforce security. Naturally if these info are not shielded against unauthorized entry (read, modify, and delete), the operating-system cannot provide enforcement. Several techniques, integrating encryption, hardware handle, and isolation, assistance isolation of operating-system protection data.

Security Features of Trusted Operating Systems

Unlike regular os's, trusted systems integrate technology to handle both capabilities and assurance. The look of a reliable system is fragile, involving collection of a proper and consistent group of features as well as an appropriate amount of assurance which the features have already been assembled and executed correctly. Figure 5.11 illustrates what sort of trusted operating-system differs from a typical one. Evaluate it with Figure 5-10. Detect how objects are usually accompanied or bounded by an gain access to control mechanism, providing far more safeguard and parting than does the standard operating system. Furthermore, memory is divided by end user, and files and method libraries have operated sharing and parting.

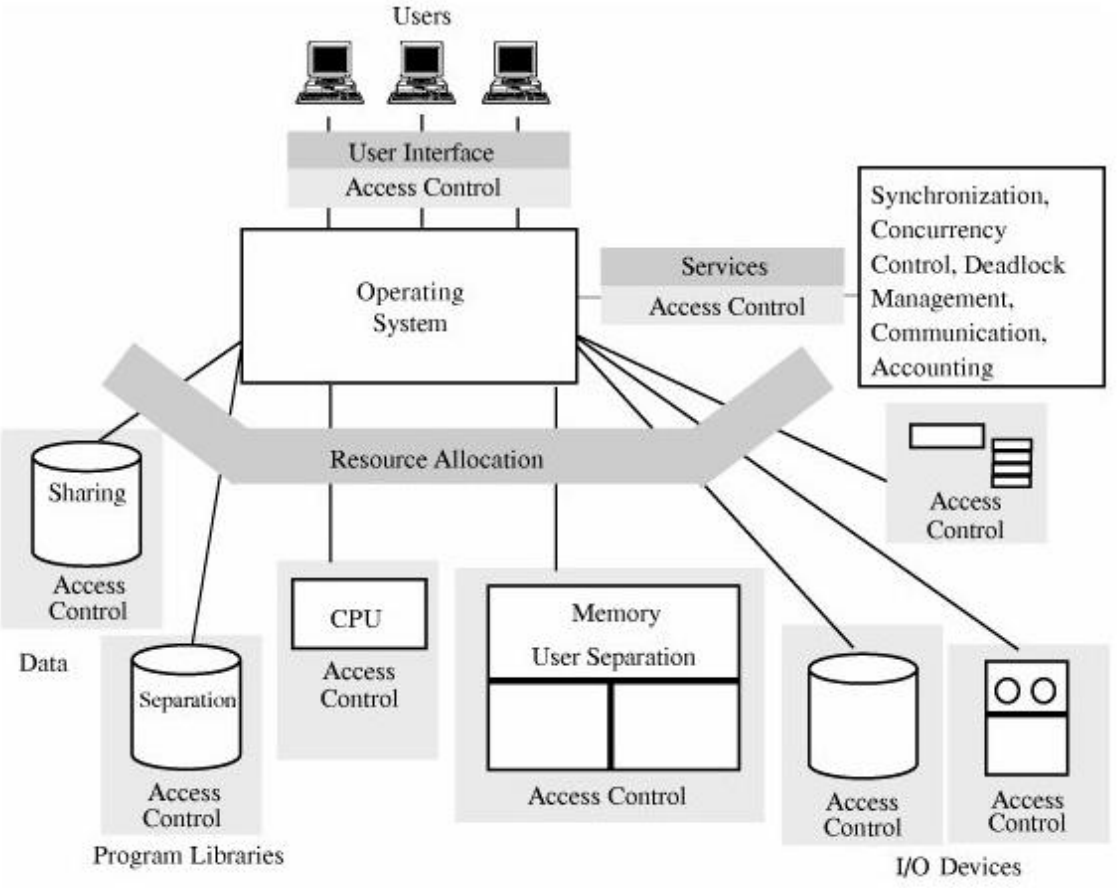


Figure 5.11. Security Functions of a Trusted Operating System.

In this section, we consider in more detail the key features of a trusted operating system, including

user identification and authentication

mandatory access control

discretionary access control

object reuse protection

complete mediation

trusted path

audit

audit log reduction

intrusion detection

We consider each of these features in turn.

Identification and Authentication

Identification reaches the main of a lot of computer security. We should have the ability to tell who's requesting usage of an subject, and we should have the ability to confirm the subject's id. As we discover shortly, most gain access to control, whether compulsory or discretionary, is dependant on accurate identification. Identification will involve two ways: learning who the accessibility requester can be and verifying the requester is definitely who he/she/it says to be. That's, you want to establish an identification and authenticate or confirm that identity. Dependable operating systems demand secure identification of people, and every individual must be distinctively identified.

Mandatory and Discretionary Access Control

Mandatory access control (MAC)implies that access control insurance plan decisions are created beyond the handle of the average person owner of the object. A middle authority can determine what information is usually to be available by whom, and an individual cannot change entry rights. A good example of MAC develops in military protection, where a person data owner will not decide who

includes a top-secret clearance; neither can the individual owner alter the classification of your object from " inside info " to secret.

In comparison, **discretionary access command (DAC),** as its label implies, leaves some access control towards the discretion in the object's owner or even to anyone else who's authorized to regulate the object's admittance. The dog owner can ascertain who must have access rights with an subject and what those privileges should be. Industrial environments typically make use of DAC to permit anyone within a designated team, and sometimes extra named individuals, to improve access. For instance, a company might establish admittance controls so the accounting group might have access to staff files. However the corporation could also let Ana and Jose to gain access to those files, as well, in their assignments as directors with the Inspector General's business office. Typically, DAC admittance rights can transform dynamically. Who owns the accounting document may include Renee and take out Walter from set of allowed accessors, as enterprise needs dictate.

MAC and DACcan both be employed to exactly the same object. MAC offers precedence over DAC, and therefore of all those who find themselves approved for MAC access, only those that also go DAC will in actuality be permitted to access the thing. For instance, a file could be classified secret, and therefore only persons cleared for key access could access the data file. But of these thousands of people granted secret gain access to by the federal government, only men and women on job "deer recreation area" or within the "environmental" party or at site "Fort Hamilton" are in fact allowed access.

Object Reuse Protection

One way a computing system sustains its efficiency would be to reuse things. The operating-system controls source allocation, so when a resource can be freed for employ by other customers or plans, the operating-system permits another user or course to gain access to the source of information. But reusable items must be cautiously managed, lest they develop a significant vulnerability. To understand why, consider what takes place when a fresh file is established. Usually, space for that file originates from a pool area of freed, used space on the disk or different storage gadget. Released space can be returned for the pool "filthy," that's, still containing the info from the prior user. Because many users would create to a document before trying to study from it, the brand new user's info obliterate the prior owner's, so there is absolutely no improper disclosure of the prior user's

information. On the other hand, a malicious customer may claim a great deal of disk space and scavenge for hypersensitive data. This sort of attack is named object reuse. The thing is not limited by disk; it could occur with key memory, cpu registers and storage space, other magnetic marketing (such as for example disks and tapes), or any reusable storage moderate.

To prevent item reuse leakage, os's clear (that's, overwrite) all place being reassigned before enabling the next person to have usage of it. Magnetic mass media are particularly susceptible to this threat. Incredibly precise and costly equipment will often separate the newest data from the info previously documented, from the info before that, etc. This threat, referred to as **magnetic remanence**, is definitely beyond the opportunity of this e book. Regardless, the operating-system must take accountability for "cleaning" the tool before permitting usage of it. (Find Sidebar 5-4 for another kind of consistent data.)

Complete Mediation

For essential or discretionary gain access to control to work, all accesses should be controlled. It really is insufficient to regulate access and then files in the event the attack will get access through memory space or another port or perhaps a network or perhaps a covert channel. The look and implementation problems of a reliable operating system goes up significantly as extra paths for gain access to must be managed. Highly trusted os's perform complete mediation, and therefore all accesses are usually checked.

Trusted Path

One way regarding a malicious user in order to gain inappropriate access will be to "spoof" users, which makes them think they are interacting with the best safety measures enforcement system while going to fact their keystrokes and even commands are being blocked and analyzed. For example of this, a malicious spoofer may well place a phony end user ID and password technique between the user plus the legitimate system. As the particular illegal system queries typically the user for identification info, the spoofer captures typically the real user ID and even password; the spoofer may use these bona fide entry data to obtain the system down the road, possibly with malicious intent. Hence, for critical operations many of these as setting a pass word or changing access accord, users want an distinguished communication, called a relied on path, to ensure of which they are

supplying guarded information only to the legitimate receiver. On several trusted systems, the end user invokes a trusted course by pressing an special key sequence that, simply by design, is intercepted immediately by security enforcement computer software; on other trusted techniques, security-relevant changes can become made only at program startup, before any techniques other than the basic safety enforcement code run.

Accountability and Audit

A security-relevant action may be simply because simple being an individual entry to an object, for example a file, or it may possibly be as major because a change to the particular central access control data source affecting all subsequent has access to. Accountability usually entails keeping a log of security-relevant events that have took place, listing each event in addition to the person responsible intended for the addition, deletion, or even change. This audit journal must obviously be shielded from outsiders, each security-relevant event must be noted.

Audit Log Reduction

In theory, the general notion involving an audit log is definitely appealing as it allows liable parties to gauge all steps that affect all guarded elements of the machine. Nevertheless in practice an exam log may be as well challenging to handle, owing in order to volume and analysis. In order to see why, considercarefully precisely what information would have to be able to be collected and assessed. In the extreme (such as where the information involved can affect a new business' viability or a new nation's security), we may possibly argue that every changes or even each personality read from a document is potentially security appropriate; the modification could impact the integrity of data, or the single personality could divulge the simply really sensitive part associated with an entire file. In addition because the path associated with control through the program is usually affected by the info the particular program processes, the pattern of individual instructions is likewise potentially security relevant. Throughout the event that a good audit record were in order to be designed for every accessibility to a single personality from a file plus for every instruction carried out, the audit log would likely be enormous. (In simple fact, it would be unattainable to audit every coaching, because then the review commands themselves would have got to be audited. In return, these commands would get implemented by instructions that will would must be audited, and even so on forever.)

In most trusted devices, the catch is simplified by a great audit of only the particular opening (first access to) and closing of (last access to) files or even similar objects. Similarly, items such as individual memory space locations, hardware registers, plus instructions are not audited. Even with these limitations, audit logs tend to be able to be very large. The simple word processor may well open fifty or additional support modules (separate files) in order to begins, it may well create and delete some sort of dozen or more momentary files during execution, also it may open many even more drivers to handle certain tasks for instance complex format or printing. Thus, one particular simple program can certainly trigger a hundred files to get opened and closed, in addition to complex systems can lead to thousands of files to become accessed in a comparatively short period of time. On the some other hand, some systems continually read from or revise a single file. The bank teller may approach transactions against the standard customer accounts file through the entire day; precisely what is significant is certainly not that the teller reached the accounts file, although which entries in the particular file were accessed. Hence, audit at the amount of file opening and final is in many cases as well much data and throughout other cases not plenty of to meet security wants.

A final difficulty may be the "needle in a haystack" phenomenon. Even if typically the audit data may be confined to the right quantity, typically many legitimate has access to and possibly one attack may occur. Finding the one particular attack access out regarding a thousand legitimate has access to can be difficult. A new corollary to the problem is usually the one of figuring out who or what does indeed the analysis. Does typically the system administrator sit and even analyze all data within the audit log? Or perhaps do the developers create a program to evaluate the data? If the particular latter, how can all of us automatically recognize a routine of unacceptable behavior? These types of issues are open inquiries being addressed not just by simply security specialists but in addition by simply experts in artificial intellect and pattern recognition.

Intrusion Detection

Closely connected to audit reduction will be the ability to detect safety measures lapses, ideally while they will occur. As we include seen in the Point out Department example, there might well be a lot of data in the audit sign for a human to be able to analyze, however the computer could help correlate independent information. Intrusion detection software creates patterns of normal program

usage, triggering an alert any time the use seems abnormal. After a new decade of promising study leads to intrusion detection, items are now commercially obtainable. Some trusted systems consist of a primitive degree involving intrusion detection software.

Kernelized Design

The kernel is the portion of an os that functions the lowest-level functions. Throughout standard operating-system design, the particular kernel implements operations many of these as synchronization, interprocess connection, message passing, and disrupt handling. The kernel is usually also called a nucleus or core. The idea of designing an functioning system around a nucleus is described by Lampson and Sturgis and by Popek and Kline.

A **security kernel** is definitely responsible for enforcing typically the security mechanisms of typically the entire operating system. The particular safety kernel provides typically the security interfaces among typically the hardware, operating system, along with other parts of the processing system. Typically, the functioning system is made so of which the security kernel is usually contained within the working system kernel. Security kernels are discussed in fine detail by Ames.

There are several great design reasons why safety functions might be isolated inside a security kernel.

- **Coverage**. Every access to the protected object must go through the security kernel. Within a system designed throughout this way, the operating-system can use the safety kernel to make sure that every entry is checked.

- **Separation.** Separating security mechanisms both coming from the rest of typically the operating-system and from the particular user space makes that easier to protect individuals mechanisms from penetration simply by the main system or typically the users.

- **Unity.** All safety measures functions are performed by simply a single set involving code, so it is usually easier to trace the reason for any problems that occur with one of these functions.

- **Modifiability.** Modifications to the safety components are easier to help to make and easier to test out.

- **Compactness.** Since it performs just security functions, the safety measures kernel is likely to be able to be relatively small.

- **Verifiability.** Being relatively small, typically the security kernel could be assessed rigorously. For example, elegant methods can be utilized to ensure that most security situations (such while states and state changes) have been covered simply by the design.

Spot typically the similarity between these positive aspects and the design aims of operating systems that will we described earlier. These types of characteristics also depend throughout many ways on modularity, On the additional hand, implementing a protection kernel may degrade program performance because the nucleus adds another layer regarding interface between user courses and os resources. In addition, the presence of the kernel does not promise that it has all safety measures functions or that this has been implemented appropriately. And perhaps a safety kernel can be really large.

How do we all balance these positive plus negative aspects of utilizing a security kernel? The design and style and usefulness of some sort of security kernel depend fairly on the overall strategy to the operating anatomy's design. There are a lot of design choices, each associated with which falls as one particular of two types: Both the kernel is created as an conjunction using the operating system, or even it is the groundwork of the entire functioning system. We will look additional closely each and every single design choice

Reference Monitor

The most important element of a security nucleus is the Reference Monitoran eye on, the portion that settings accesses to objects .A research monitor is not automatically a single part of signal; rather, it is the particular assortment of access controls with regard to devices, files, memory, interprocess communication, and other types of objects. As proven in Figure 5.12, the reference monitor acts just like a brick wall throughout the operating system or reliable software.

IDOL

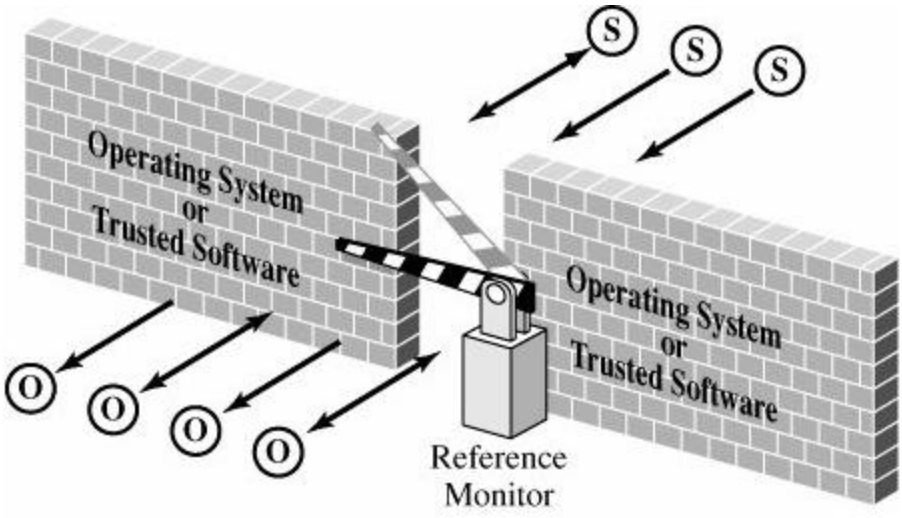


Figure 5.12. Reference Monitor.

A reference monitor has to be

- **tamperproof,** that will be, impossible to weaken or even disable
- **unbypassable,** that will be, always invoked when accessibility to any object is needed
- **analyzable,** that is, compact enough to be put through to analysis and tests, the completeness which could be ensured

A reference monitor can control obtain effectively only if this should not be modified or circumvented with a rogue process, plus it is the simply point through which almost all access requests must go. Furthermore, the reference keep track of must function correctly in case it is to meet its crucial role throughout enforcing security. Because the particular probability of correct habits decreases because the complexity and even size of a course enhance, the best assurance associated with correct policy enforcement is usually to build a small, very simple, understandable reference monitor.

The particular reference monitor is not necessarily the only security device of a trusted operating-system. Other parts of typically the security suite include taxation, identification, and authentication digesting, as well as typically the setting of enforcement variables, like who the permitted subjects are and which usually

objects they are authorized to access. The some other security parts interact along with the reference monitor, getting data from the research monitor or providing that with the data that needs to operate.

Trusted Computing Base

The trustworthy computing base, or TCB, may be the subject we give to almost everything within the trusted operating method necessary to enforce the particular security policy. Alternatively, we all say that the TCB includes the parts associated with the trusted operating program on which we hinge for correct enforcement involving policy. We can consider of the TCB while a coherent whole inside the following way. Imagine you divide a reliable os into the elements that are inside the TCB and those that are usually not, and you also allow typically the most skillful malicious computer programmers to write all the particular non-TCB parts. Since the particular TCB handles all the particular security, there is nothing at all the malicious non-TCB pieces can perform to impair typically the correct security policy observance of the TCB. This kind of definition gives you a feeling that the TCB types the fortress-like shell of which protects whatever in typically the system needs protection. Yet the analogy also explains the meaning of reliable in trusted operating technique: Our trust in typically the security of the completely system depends on typically the TCB.

It is possible to see that will it is essential intended for the TCB to become both correct and. As a result, to understand how in order to design a good TCB, we give attention to the split between the TCB and even non-TCB elements of typically the operating system and expend our effort on making sure the correctness of typically the TCB.

TCB Functions

Just what comprises the TCB? We could answer this question simply by listing system elements in which security enforcement may depend:

- **hardware,** including cpus, memory, registers, and I/O devices

- **some notion regarding processes**, so that we are able to separate and protect security-critical processes

- **primitive files,** including the security access control databases and identification/authentication data

- **protected memory,** so that typically the reference monitor can end up being protected against tampering

- **some interprocess communication**, in order that diverse parts of the TCB can pass data in order to and activate other pieces. For example, the reference point monitor can invoke plus pass data securely in order to the audit routine.

It may look as if this checklist encompasses most of typically the operating system, but within fact the TCB is definitely only a tiny subset. Intended for example, although the TCB requires access to documents of enforcement data, this does not need a good entire file structure involving hierarchical directories, virtual equipment, indexed files, and multidevice files. Thus, it may include a primitive record manager to manage only the particular small, simple files required for the TCB. The greater complex file manager to deliver externally visible files might be outside the TCB. Figure 5.13 shows a standard division into TCB and even non-TCB sections.

IDOL

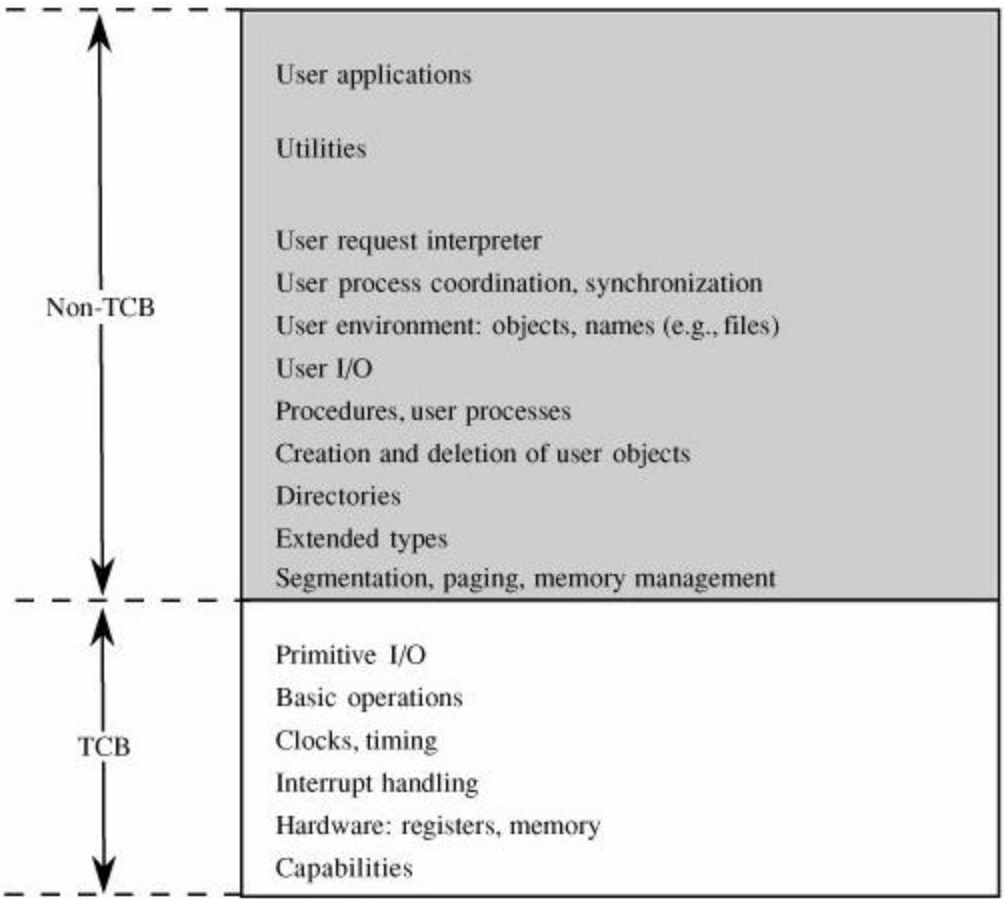


Figure 5.13. TCB and Non-TCB Code.

The TCB, which must maintain typically the secrecy and integrity regarding each domain, monitors several basic interactions.

- **Process activation.** In a multiprogramming atmosphere, activation and deactivation involving processes occur frequently. Transforming from process to one other requires a complete modification of registers, relocation routes, file access lists, procedure status information, and also other tips, much of which is usually security-sensitive information.

- **Execution domain switching.** Processes running inside one domain often employ processes consist of domain names to obtain more very sensitive data or services.

- **Memory protection.** Because each website includes code and files trapped in memory, the TCB must monitor memory referrals to make sure secrecy and ethics for every domain.

- **I/O functioning.** In certain systems, software will be involved with each personality transferred in an I/O operation. This software hooks up an user program throughout the outermost domain to be able to an I/O device inside of the innermost (hardware) domain name. Thus, I/O operations may cross all domains.

TCB Design

The trademark the particular operating system into TCB and non-TCB aspects is usually convenient for designers plus developers because it methods that all security-relevant computer code is located in one particular (logical) part. But the particular distinction is more as compared to just logical. To make certain the particular security enforcement cannot turn out to be afflicted with non-TCB code, TCB code must run throughout some protected state that will distinguishes it. Thus, typically the structuring into TCB in addition to non-TCB must be performed consciously. However, once this particular structuring has been performed, code outside of the TCB might be changed whenever, without having affecting the TCB's potential to enforce security. This kind of ability to change assists developers because it implies that major parts associated with the operating systemutilities, system drivers, user interface professionals, along with the likecan be adjusted or replaced any period; only the TCB program code must be controlled a lot more carefully. Finally, for any person evaluating the security associated with a trusted operating-system, a new division into TCB in addition to non-TCB simplifies evaluation significantly because non-TCB code will need not be considered.

TCB Implementation

Security-related activities will be likely to be executed in different places. Protection is potentially related in order to every memory access, every single I/O operation, every record or program access, each initiation or termination regarding an user, each interprocess communication. In modular working systems, these separate routines can be handled inside independent modules. Each involving these separate modules, in that case, has both security-related in addition to other functions.

Collecting most security functions into the particular TCB may destroy typically the modularity of an present operating system. A specific TCB may also end up being too big to become analyzed easily. Nevertheless, some sort of designer may decide in order to separate the security capabilities of an existing functioning system, creating a safety kernel. This form regarding kernel is depicted inside Figure 5.14.

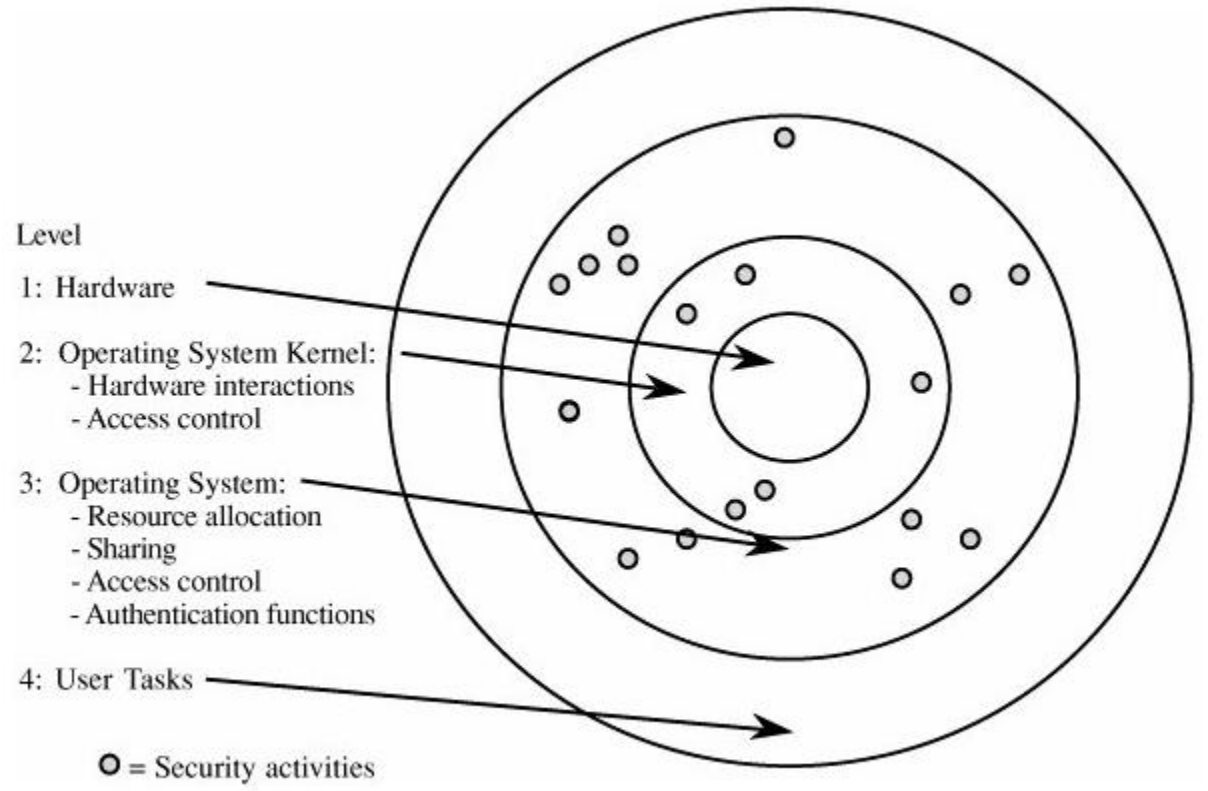


Figure 5.14. Combined Security Kernel/Operating System.

A even more sensible approach would be to design and style the security kernel very first and then design the particular main system around it. This kind of technique utilized by Honeywell in the type of the prototype for its safeguarded operating system, Scomp. That will system contained only 20 modules to perform the particular primitive security functions, plus it consisted of significantly less than 1, 000 traces of higher-level-language source signal. After the actual security

nucleus of Scomp was constructed, its functions grew in order to contain approximately 10, 1000 lines of code.

Found in a security-based design, the particular security kernel forms a good interface layer, just on top of system hardware. The security kernel monitors all operating-system hardware accesses and works all protection functions. The particular safety kernel, which depends on support from components, allows the operating technique itself to handle the majority of functions not related to be able to security. In this method, the security kernel could be small and efficient. While a byproduct of this specific partitioning, computing software offers at least three delivery domains: security kernel, running system, and user. Notice Figure 5.15.

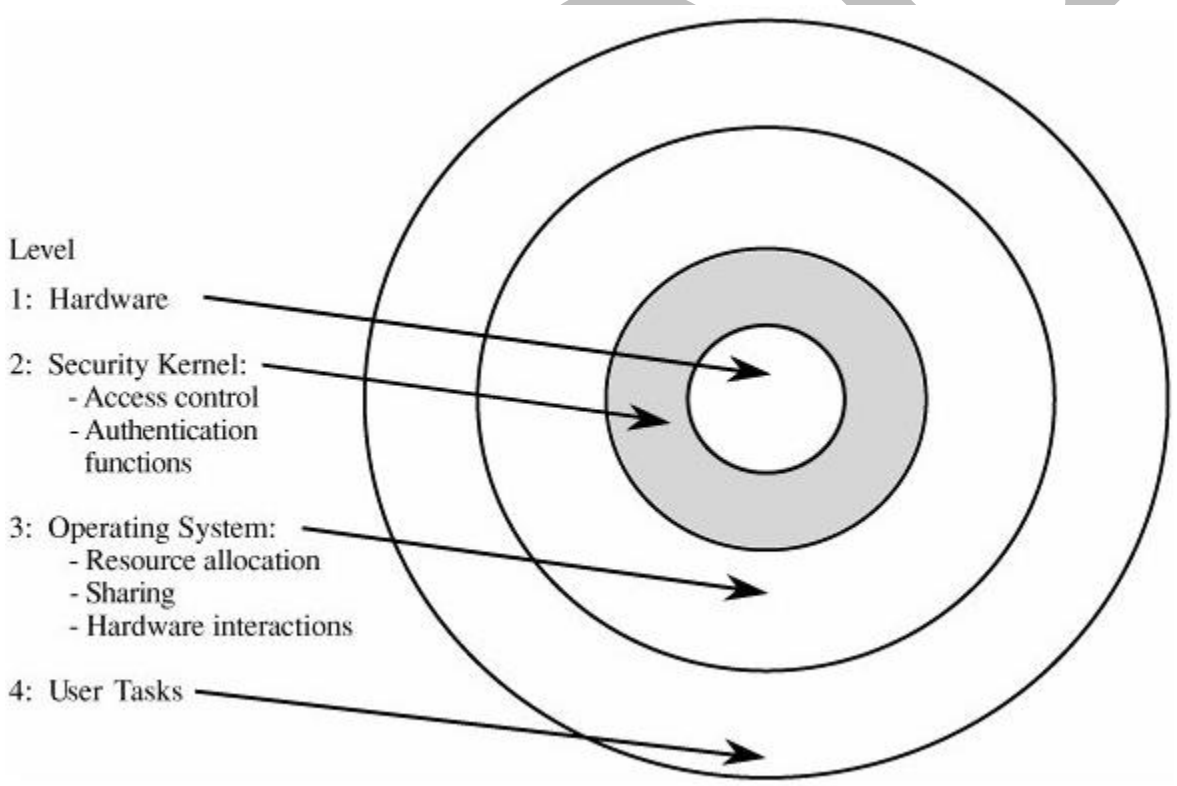


Figure 5.15. Separate Security Kernel.

Separation/Isolation

list several ways to separate one particular process from others: actual physical, temporal, cryptographic, and reasonable separation. With physical parting, two different processes make use of two different hardware features. For example, sensitive calculation may be performed on the subject of a reserved computing

program; nonsensitive tasks are go on a public technique. Hardware separation offers various attractive features, including help for multiple independent posts of execution, memory security, mediation of I/O, and even at least three various degrees of execution freedom. Temporal separation occurs whenever different processes are function at different times. With regard to instance, some military methods run nonsensitive jobs in between 8: 00 a. d. and noon, with hypersensitive computation from noon to be able to 5: 00 p. d. Encryption is used intended for cryptographic separation, so a couple of different processes can become run at the equivalent time because unauthorized consumers cannot access sensitive files in a readable contact form. Logical separation, also referred to as isolation, is provided any time a process for instance a guide monitor separates one customer's objects from the kinds from another user. Protected computing systems have recently been built with these varieties of separation.

Multiprogramming operating systems should separate each user from just about all others, allowing only thoroughly controlled interactions between the particular users. Most systems happen to be designed to provide some sort of single environment for almost all. In other words, a single copy of the working system is available for proper use by many users, because shown in Figure 5.16. The operating system is usually often separated into 2 distinct pieces, located in the highest and minimum addresses of memory.

IDOL

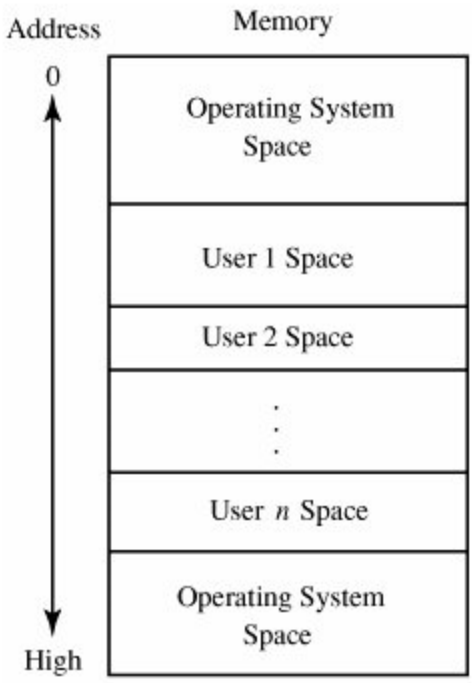


Figure 5-16. Conventional Multiuser Operating System Memory.

Virtualization

Virtualization is an effective tool for trusted program designers because it enables users to reach complex items in a carefully handled manner. By virtualization we all mean that the main system emulates or simulates an accumulation of a computer system's solutions. We say that some sort of virtual machine is some sort of collection of real or perhaps simulated hardware facilities: the [central] cpu that runs an teaching set, an amount involving directly addressable storage, plus some I/O devices. These types of facilities support the setup of programs.

Obviously, online resources must be preserved real hardware or application, but the real solutions do not need to be the exact same as the simulated types. There are many good examples of this kind of simulation. With regard to instance, printers tend to be controlled on direct access gadgets for sharing in multiuser environments. Several small devices can be simulated together with one large one. Using demand paging, some noncontiguous memory can support some sort of much larger contiguous digital memory space. And this is common even about PCs

to simulate place on slower disks using faster memory. In these kinds of ways, the operating-system supplies the virtual resource for the user, while the safety kernel precisely controls customer accesses.

Multiple Virtual Recollection Spots

The IBM MVS/ESA operating-system uses virtualization to be able to provide logical separation of which gives the user the particular impression of physical parting. IBM MVS/ESA is a new paging system such that will each user's logical tackle space is separated coming from that of others by simply the page mapping device. Additionally, MVS/ESA includes the particular operating system in each and every user's logical address area, so an user works about what seems to become a complete, separate equipment.

Most paging systems exhibit an user only typically the user's virtual address area; the operating system is definitely outside the user's electronic addressing space. Yet , the particular operating system is area of the logical space of every single MVS/ESA user. Therefore, in order to the user MVS/ESA looks like a single-user program, as shown in Figure 5.17.

IDOL

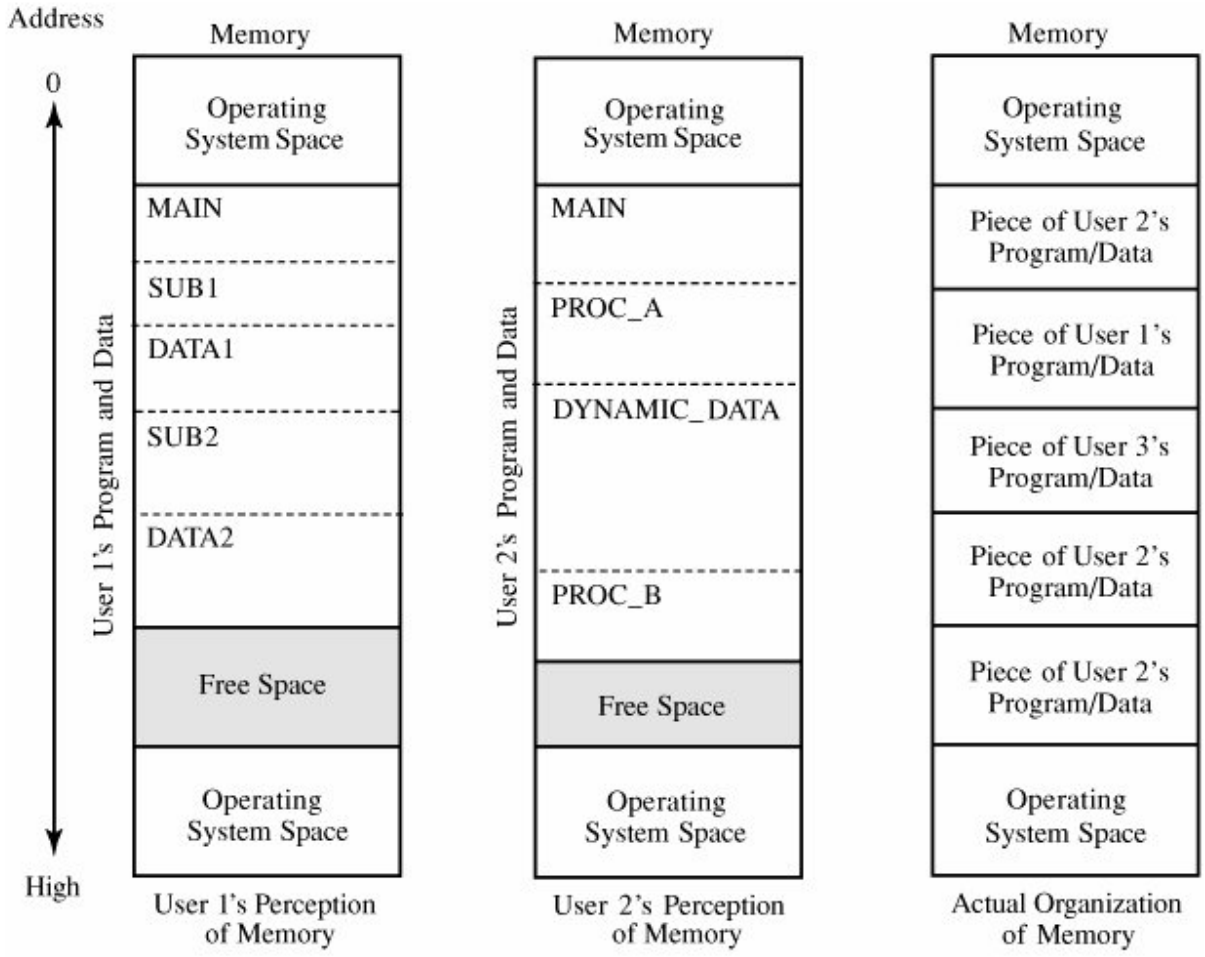


Figure 5-17. Multiple Virtual Addressing Spaces.

A primary good thing about MVS/ESA is memory administration. Each user's virtual storage space can be as huge as total addressable storage, in excess of sixteen million bytes. And safety is a second edge of this representation regarding memory. Because each customer's logical address space consists of the operating-system, the wearer's perception features running upon a separate machine, which usually could even be real.

Virtual Machines

The IBM Processor Resources/System Manager (PR/SM) system provides a stage of protection that is certainly more powerful still. A conventional operating-system has hardware facilities plus devices that are beneath the direct control associated with the operating system, like shown in Figure 5.18. PR/SM provides an whole

virtual machine to each and every user, to ensure that each consumer not only has reasonable memory but also features logical I/O devices, reasonable files, and other reasonable resources. PR/SM performs this particular feat by strictly isolating resources.

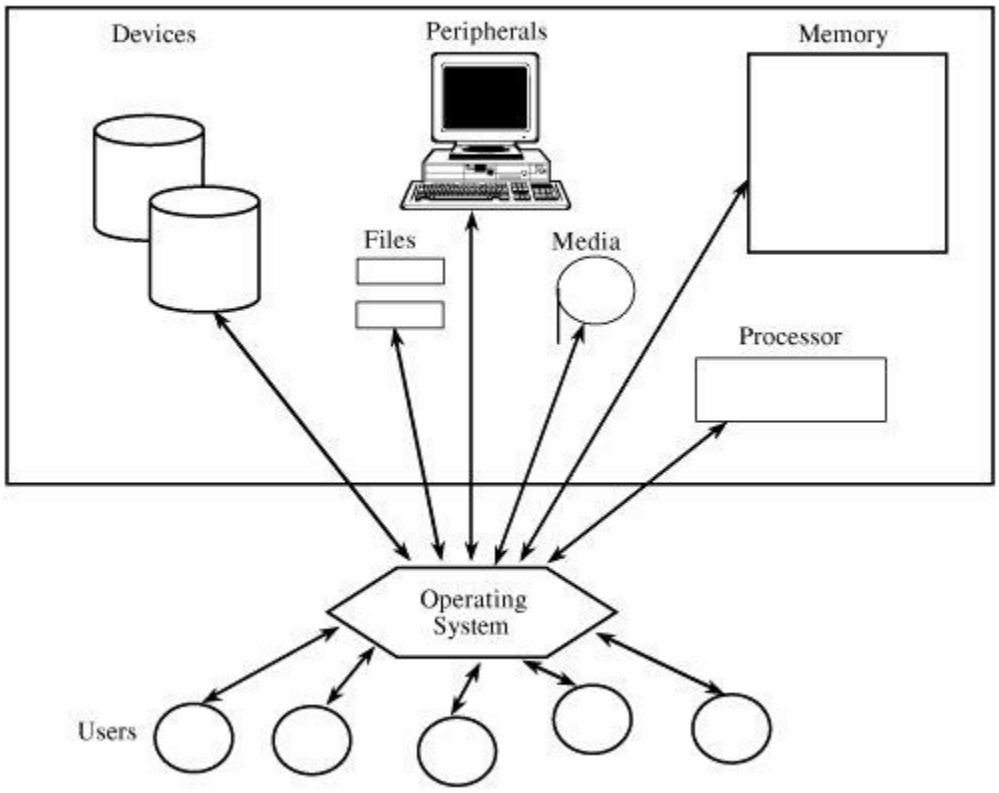


Figure 5.18. Conventional Operating System.

The PR/SM method is an organic extension regarding the concept of digital memory. Virtual memory provides user a memory room that may be logically separated coming from real memory; a digital storage is usually greater than real memory, because well. A virtual device gives the user some sort of full pair of hardware benefits; that is, a total machines that may be considerably distinctive from the real equipment. These virtual hardware assets are also logically segregated from those of other folks. The relationship of online machines to real types is shown in Figure 5-19.

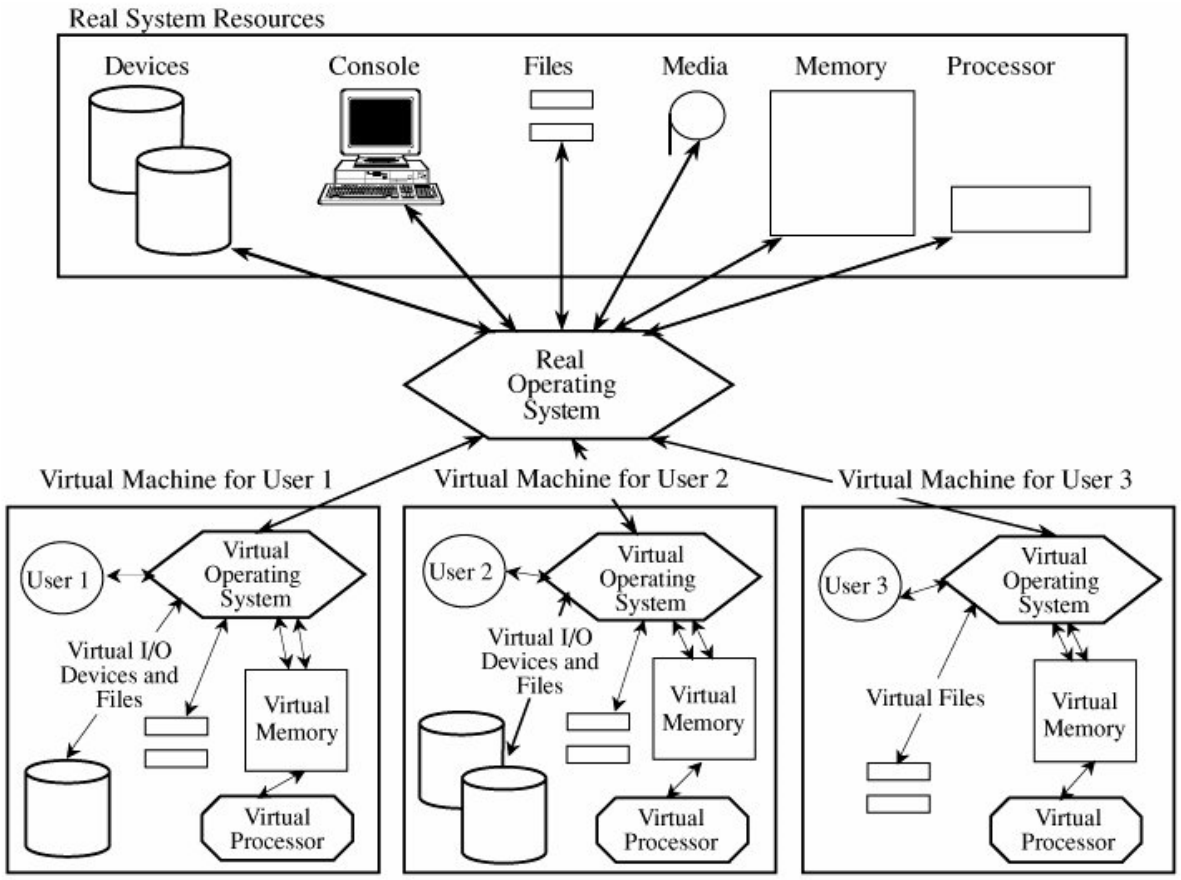


Figure 5.19. Virtual Machine.

Both MVS/ESA plus PR/SM improve the remoteness of each user by other users and from your hardware of the technique. Naturally, this added complexness boosts the overhead incurred along with these amounts of translation plus protection. Within the next section many of us study alternative designs of which reduce the complexity associated with providing security within a running system.

Layered Design

Because described previously, a kernelized operating system consists associated with at least four amounts: hardware, kernel, operating-system, in addition to user. Each of these kinds of layers can include sublayers. For example, in the kernel has got five distinct layers. With the user level, it is far from uncommon to have phony system programs, such because database managers or visual user interface shells, of which constitute separate layers involving security themselves.

Layered Trust

As we have noticed earlier with this chapter (in Figure 5-15), the split view of the secure functioning system can be portrayed as a series regarding concentric circles, with the particular most sensitive operations inside the innermost layers. Then, the particular trustworthiness and access privileges of a process will be judged by typically the process's proximity for the centre: The more trusted operations are closer to typically the center. But we may also depict the trustworthy operating system in tiers as a stack, along with the security functions nearest to the hardware. These kinds of a system is displayed in Figure 5-20.

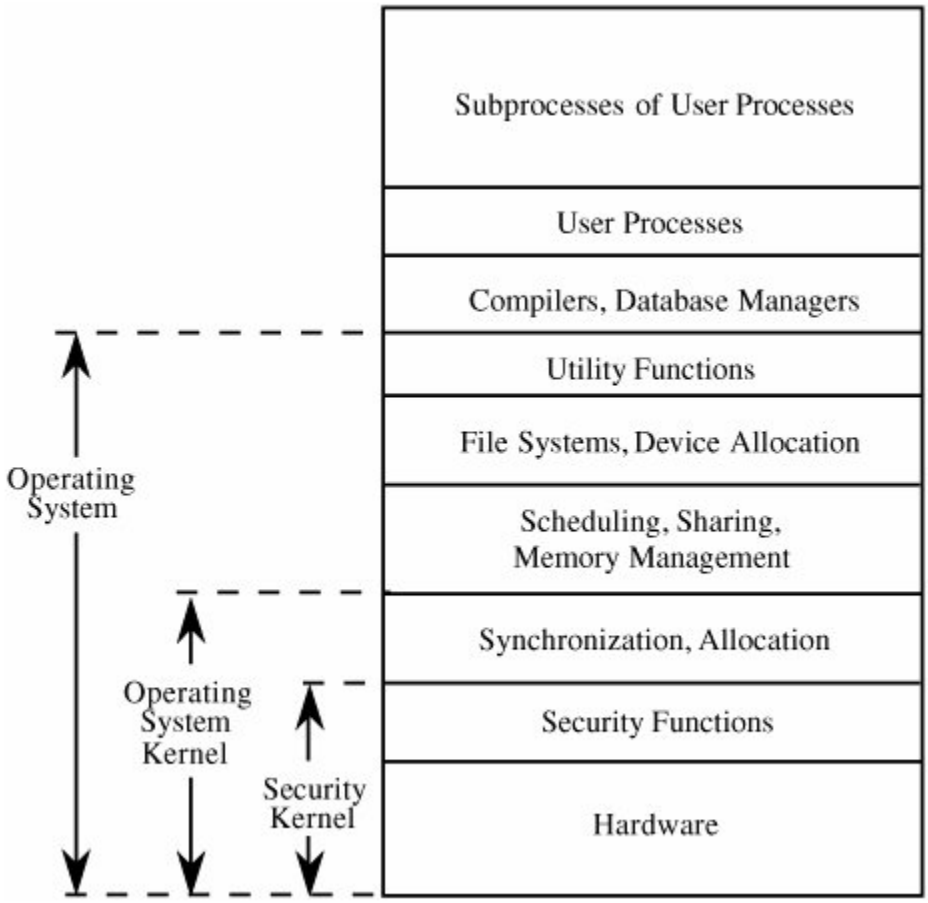


Figure 5.20. Layered Operating System.

Inside this design, some pursuits related to protection features are performed away from protection kernel. For example, consumer authentication may include being able to access a password table, tough you supply an username and password, verifying the correctness associated with the password, and therefore forth. The

disadvantage associated with performing each one of these operations within the security kernel will be that some of the particular operations (such as format the userterminal interaction plus searching for the consumer inside a table of recognized users) do not bring about high security.

On the other hand, we can implement the single logical function inside several different modules; all of us call this a split design. Trustworthiness and entry rights are the base the layering. In additional words, an individual function might be performed with a collection of modules operating within different layers, as proven in Figure 5-21. Typically the modules of each coating perform operations of some sort of certain degree of level of sensitivity.

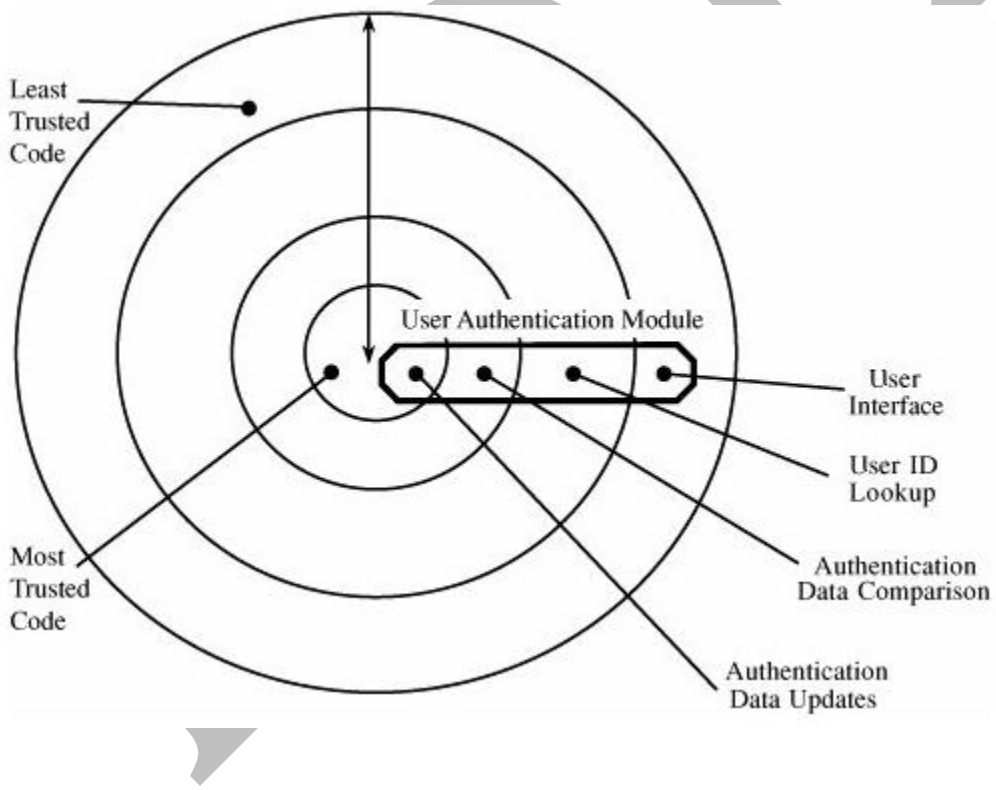


Figure 5.21. Modules Operating In Different Layers.

Neumann describes the layered construction useful for the Provably Safeguarded Main system (PSOS). As displayed in Table 5.4, several lower-level layers present many or all of their very own functionality to higher amounts, but each layer appropriately encapsulates those things under itself.

Table 5.4. PSOS Design Hierarchy.

Level	Function	Hidden by Level	Visible to User
16	User request interpreter		Yes
15	User environments and name spaces		Yes
14	User I/O		Yes
13	Procedure records		Yes
12	User processes and visible I/O		Yes
11	Creation and deletion of user objects		Yes
10	Directories	11	Partially
9	Extended types	11	Partially
8	Segments	11	Partially
7	Paging	8	No
6	System processes and I/O	12	No
5	Primitive I/O	6	No
4	Arithmetic and other basic operations		Yes
3	Clocks	6	No
2	Interrupts	6	No
1	Registers and addressable memory	7	Partially
0	Capabilities		Yes

From [NEU86], © IEEE, 1986. Used with permission.

IDOL

A layered strategy is one method to achieve encapsulation, discussed in Chapter 3 or more. Layering is known since a good operating technique design. Each layer utilizes the more central levels as services, every coating provides a certain amount of functionality to the tiers farther out. In this particular way, we can "peel off" each layer but still have a logically full system with less operation. Layering presents an excellent example of how in order to advantage and balance design and style characteristics.

Another justification with regard to layering is damage command. To find out why, consider Neumann's 2 types of risk, shown inside Tables 5.5 and 5.6. Inside a conventional, nonhierarchically developed system (shown in Table 5-5), any problemhardware disappointment, software flaw, or unforeseen condition, even in a new supposedly non-security-relevant portioncan result in disaster since the effect associated with the problem is unrestrained also because the anatomy's design implies that we can not be confident that any kind of given function has zero (indirect) security effect.

Table 5-5. Conventionally (Nonhierarchically) Designed System.

Level	Functions	Risk
All	Noncritical functions	Disaster possible
All	Less critical functions	Disaster possible
All	Most critical functions	Disaster possible

IDOL

Table 5-6. Hierarchically Designed System.

Level	Functions	Risk
2	Noncritical functions	Few disasters likely from noncritical software
1	Less critical functions	Some failures possible from less critical functions, but because of separation, effect limited
0	Most critical functions	Disasters possible but unlikely if system simple enough to be analyzed extensively

Simply by contrast, as shown throughout Table 5.6, hierarchical building has two benefits:

- Hierarchical structuring permits identification regarding the most critical pieces, which can then end up being analyzed intensely for correctness, and so the number of difficulties should be smaller.

- Isolation limits effects of issues to the hierarchical amounts at and above typically the point from the problem, and so the effects of several problems should be limited.

These design propertiesthe nucleus, separation, isolation, and hierarchical structurehave been the schedule for many trustworthy technique prototypes. They have endured the test of moment as best design in addition to implementation practices.

IDOL

5.5. Assurance in Trusted Operating Systems

This chapter provides moved our dialogue from the overall to this. We begun by studying the latest models of of protection devices. By enough time we reached the final section, we reviewed three principlesisolation, stability kernel, and split structureused in building secure os's, and we viewed in detail in the approaches used by developers of particular os's. Now, we guess that an operating-system provider has had these considerations into consideration and claims to truly have a secure design. It really is time for all of us to consider confidence, ways of persuading others a model, style, and implementation happen to be correct.

Typical OPERATING-SYSTEM Flaws

Regularly throughout our research of operating-system security features, we've used the key phrase "exploit a vulnerability." Through the entire years, various vulnerabilities have already been uncovered in lots of operating systems. They will have gradually happen to be corrected, and your body of understanding of likely weak areas has grown.

Known Vulnerabilities

In this segment, we discuss usual vulnerabilities which have been uncovered in os's. Our goal isn't to supply a "how-to" tips for prospective penetrators of os's. Rather, we review these flaws to comprehend the careful research necessary in making and testing os's. User interaction may be the largest single way to obtain operating-system vulnerabilities, for a number of reasons:

- The interface is conducted by independent, brilliant equipment subsystems. The humancomputer program often falls beyond your safety kernel or safety measures restrictions applied by an operating-system.

- Program code to connect to users is frequently much more sophisticated plus much more dependent on the precise device components than code for just about any other element of the computing program. Therefore, it really is harder to examine this program code for correctness, aside from to validate it formally.

- User interactions tend to be character oriented. Once again, in the fascination of fast information transfer, the os's designers could have tried to get shortcuts by restricting the amount of instructions executed by operating-system during actual

info transfer. Occasionally the instructions taken away are the ones that enforce security guidelines as each identity is transferred.

A second notable weakness in operating-system security displays an ambiguity in gain access to policy. Similarly, you want to separate customers and guard their individual assets. Alternatively, users be determined by shared usage of libraries, utility plans, common information, and system dining tables. The differentiation between isolation and posting is not constantly clear on the policy level, therefore the distinction can't be sharply attracted at implementation.

A third potential difficulty area is imperfect mediation. Recall that Saltzer advised an operating-system design where every requested entry was inspected for right authorization. Even so, some systems verify access only one time per interface operation, method execution, or device interval. The system can be acquired to implement total protection, however the policy choice on when to invoke the system is not finished. Therefore, within the lack of any explicit need, system designers take up the "most effective" enforcement; that's, one that will result in the least usage of machine resources.

Generality is really a fourth safety weakness, specially among commercial os's for large processing systems. Implementers make an effort to provide a opportinity for users to personalize their operating-system installation also to allow installing software packages compiled by other companies. A few of these plans, which themselves use within the operating-system, must perform with exactly the same access privileges because the operating system. For instance, there are plans offering stricter access handle than the normal control available from your operating-system. The "hooks" where these packages happen to be installed may also be trapdoors for just about any user to permeate the operating-system.

Types of Exploitations

Earlier, we reviewed why an individual interface is really a weak point in lots of major os's. We start out our illustrations by discovering this weakness in more detail. On some devices, after access may be checked to start a user procedure, the operation proceeds without following checking, resulting in basic time-of-check to time-of-use imperfections. Checking access authorization with each identity transferred is really a substantial overhead for any protection technique. The command usually resides within the user's storage. Any user can transform the foundation or destination deal with of the control after the functioning has got commenced. Because gain access to was already checked once, the brand new

address will undoubtedly be used without even more checkingit isn't checked whenever a piece of files is moved. By exploiting this flaw, consumers have been in a position to transfer files to or from any recollection address they really want.

Another exemplory case of exploitation will involve a procedural issue. In one technique a particular supervisor function was initially reserved for installing other security deals. When carried out, this supervisor contact returned handle to an individual in privileged method. The functions allowable for the reason that mode weren't monitored closely, therefore the supervisor call could possibly be used for gain access to control or for just about any other high-security technique access. This supervisor call expected some work to execute, nonetheless it was fully on the system. Extra checking must have been utilized to authenticate this program performing the supervisor submission. As a substitute, the access protection under the law for any subject matter coming into under that supervisor need might have been limited by the objects essential to perform the event with the added program.

The time-of-check to time-of-use mismatch can add security problems, as well. In an episode predicated on this vulnerability, admittance permission is examined for a specific user to gain access to an object, like a buffer. But between your time the gain access to is approved along with the access actually arises, the user shifts the designation of the thing, so that rather than accessing the permitted object, an individual today accesses another, undesirable, one.

Assurance Methods

Once we appreciate the possible vulnerabilities in something, we can put on assurance ways to look for the vulnerabilities and mitigate or eradicate their effects. In such a section, we think of three such strategies, showing how they provide us confidence in a very system's correctness: screening, confirmation, and validation. Nothing of these is certainly entire or foolproof, and each features benefits and drawbacks. However, used in combination with knowing, each can participate in an important position in deriving general assurance on the systems' security.

Testing

Testing, , may be the most widely acknowledged assurance method. As Boebert observes, conclusions from tests derive from the actual merchandise being evaluated, definitely not on some abstraction or precursor of the merchandise. This

realism is really a security advantage. Even so, conclusions predicated on testing are always limited, for the next reasons:

- Assessment can display the lifetime of an issue, but passing testing does not illustrate the lack of problems.

- Testing sufficiently within reasonable moment or effort can be difficult as the combinatorial explosion of inputs and interior states makes screening very complex.

- Testing based simply on observable outcomes, not on the inner structure of something, does not make sure any amount of completeness.

- Testing in line with the internal composition of something involves modifying the merchandise by adding program code to draw out and display inner states. That more functionality impacts the product's actions and will itself be considered a way to obtain vulnerabilities or cover up other vulnerabilities.

- Testing real-time or intricate systems presents the issue of monitoring all says and triggers. This issue makes it difficult to replicate and analyze challenges described as testers continue.

Ordinarily, we think about testing with regards to the creator: unit examining a component, integration testing to make sure that modules function correctly together, function tests to track correctness across all areas of a given work, and system assessment to combine components with software. In the same way, regression testing is conducted to be sure a change to 1 part of something will not degrade any functionality. But also for other tests, adding acceptance tests, an individual or consumer administers tests to find out if that which was ordered is what's delivered. Thus, a significant aspect of confidence is considering if the tests run work for the application form and degree of security. The type and forms of testing echo the developer's assessment approach: which studies address what concerns.

Similarly, you should recognize that evaluation is almost generally constrained by way of a project's spending plan and routine. The constraints generally mean that assessment is incomplete for some reason. Because of this, we look at notions of check coverage, evaluation completeness, and assessing effectiveness within a testing strategy. The greater complete and helpful our testing, a lot more confidence we've in the program. More info on testing are available in Pfleeger and Atlee

Penetration Testing

A testing strategy usually used in laptop security is named penetration trials, tiger team evaluation, or honest hacking. In this process, a workforce of specialists in the utilization and design and style of os's tries to split the system getting examined. The tiger staff knows well the normal vulnerabilities in os's and computing techniques, as explained in previous areas and chapters. With this particular knowledge, the workforce attempts to recognize and exploit the system's certain vulnerabilities. The task of penetration testers carefully resembles what a genuine attacker might perform

Penetration testing can be both a skill and a research. The artistic area requires careful research and creativeness in selecting the test conditions. But the technological side involves rigor, order, perfection, and group. As Weissman observes, there's an organized technique for hypothesizing and verifying defects. It isn't, as some might suppose, a arbitrary punching contest.

Using penetration screening is similar to asking a auto mechanic to look more than a used car over a sales great deal. The mechanic has learned potential weak places and checks as much of them as you possibly can. Chances are that a great mechanic will see significant troubles, but getting a problem (and repairing it) is not any promise that no additional problems are usually lurking in other areas of the machine. For instance, when the mechanic investigations the fuel method, the coolant system, along with the brakes, there is absolutely no assurance that the muffler can be good. Just as, an operating-system that fails a penetration test out may own faults, but something that will not fail isn't guaranteed to turn out to be fault-free. Even so, penetration testing pays to and often detects faults that may have been disregarded by other styles of screening. One possible reason behind the results of penetration evaluation is its employ under real-life circumstances. Users often working out a system with techniques that its makers never expected or intended. Therefore penetration testers can exploit this real-life atmosphere and knowledge to be sure kinds of challenges visible.

Penetration testing is certainly favored by the commercial neighborhood who think knowledgeable hackers will check (attack) a niche site and find issues in hours or even days. These folks don't realize that finding defects in complex program code can take 2 or 3 weeks if not calendar months. Indeed, the initial military red groups to test stability in software methods had been convened for 4- to 6-30 days workout routines. Anderson et al. explain the restriction of penetration tests. To get

one flaw in an area of just one 1 million inputs may necessitate screening all 1 million options; unless the area is reasonably constrained, this search can be prohibitive. Karger and Schell explain that even with they well informed testers of a bit of malicious program code they put in something, the testers were not able to get it. Penetration assessment isn't a magic way of finding fine needles in haystacks.

Formal Verification

The most strenuous method of examining security is certainly through conventional verification,. Formal confirmation uses regulations of mathematical reasoning to demonstrate a system has selected security houses. In formal confirmation, the operating-system is modeled plus the operating system ideas are referred to as assertions. The assortment of products and assertions can be regarded as a theorem, that is then confirmed. The theorem asserts which the operating system is certainly correct. That's, formal confirmation confirms the fact that operating system supplies the security features it will and little or nothing else.

Proving correctness of a whole operating system is really a formidable task, typically requiring months and even years of work by several individuals. Computer programs known as theorem provers can help in this work, although much individual activity continues to be needed. The quantity of work expected and the techniques used are properly beyond the range of this reserve. However, we demonstrate the general basic principle of confirmation by presenting a straightforward example that utilizes proofs of correctness.

Consider the move diagram of Figure 5.22, illustrating the reasoning in an application to look for the smallest of a couple of n ideals, A[1] through A good[n]. The circulation chart includes a single identified starting point, an individual identified ending stage, and five interior blocks, consisting of an if-then composition including a loop.

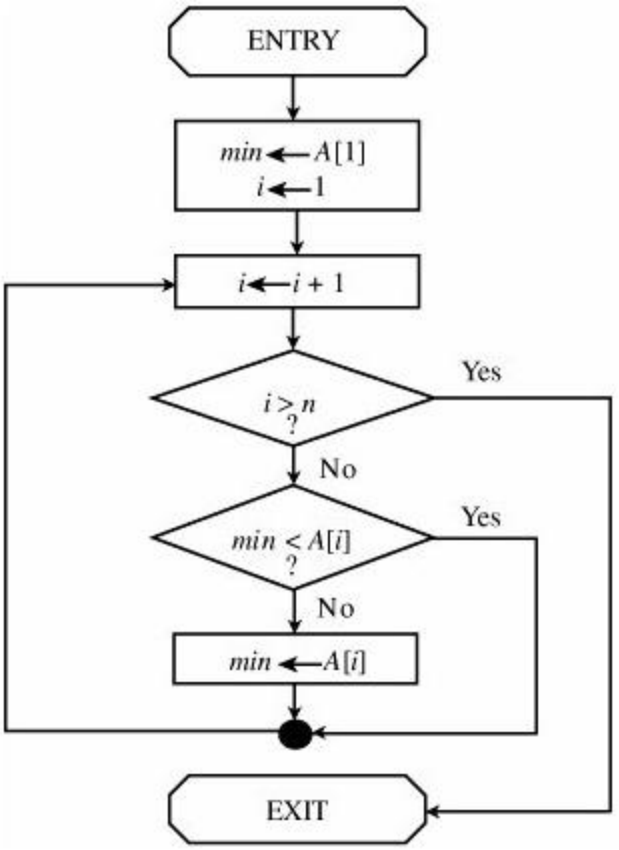


Figure 5.22. Flow Diagram for Finding the Minimum Value.

In program confirmation, we rewrite this program as some assertions concerning the program's factors and values. The original assertion is really a statement of ailments on entry towards the module. Up coming, we identify some intermediate assertions from the work on the module. We likewise determine an concluding assertion, a declaration of the anticipated result. Ultimately, we demonstrate that the original assertion turns logically for the intermediate assertions that subsequently lead logically towards the ending assertion.

We can officially verify the illustration in Shape 5-22 through the use of four assertions. The initial assertion, P, is really a statement of original conditions, assumed to become true on access to the task.

n > 0 (P)

The next assertion, Q, may be the result of using the initialization program code in the initial box.

n > 0 and (Q)

$1 \leq i \leq n$ and

$min \leq A[1]$

The 3rd assertion, R, may be the loop assertion. It asserts what's true in the beginning of every iteration from the loop.

$n > 0$ and (R)

$1 \leq i \leq n$ and

for several j, $1 \leq j \leq i - 1$, $min \leq A[j]$

The ultimate assertion, S, may be the concluding assertion, the assertion of conditions accurate at that time the loop leave occurs.

$n > 0$ and (S)

$i = n + 1$ and

for several j, $1 \leq j \leq n$, $min \leq A[j]$

These four assertions, found in figure 5.23, catch the essence from the flow chart. The next phase in the confirmation process involves demonstrating the logical development of the four assertions. That's, we must demonstrate that, presuming P holds true on entry to the procedure, Q holds true after conclusion of the initialization segment, R holds true the very first time the loop will be entered, R holds true each time with the loop, and the reality of R means that S holds true with the termination from the loop

IDOL

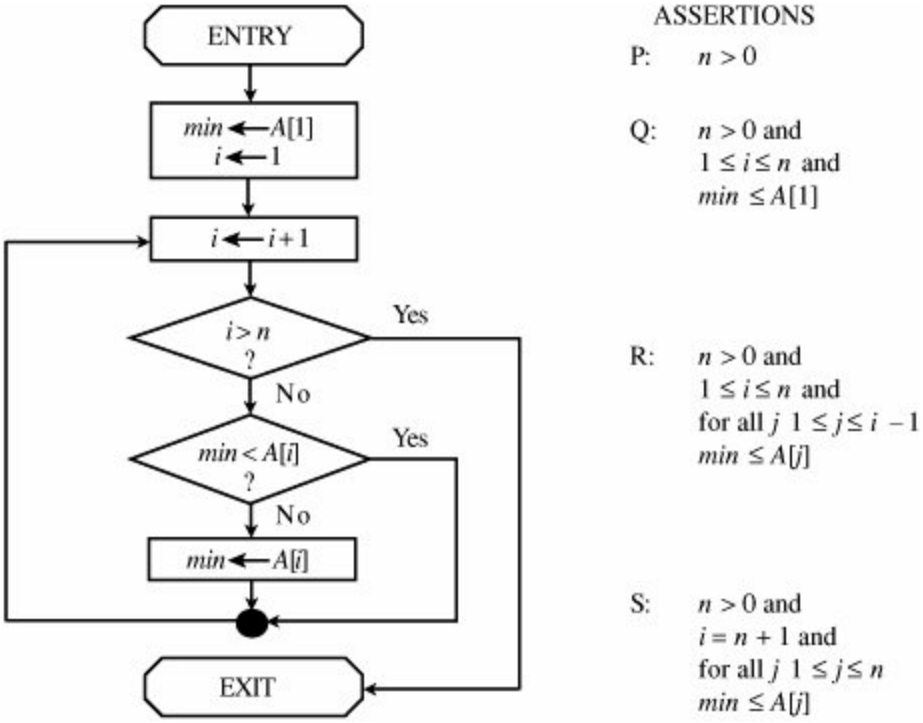


Figure 5.23. Verification Assertions.

Clearly, Q practices from P plus the semantics of both statements in the next box. Whenever we enter in the loop for the very first time, i = 2, therefore i - 1 = 1. Hence, the assertion about min applies limited to j = 1, which comes after from Q. To confirm that R remains to be legitimate with each execution with the loop, we are able to use the concept of numerical induction. The foundation with the induction is the fact R was real the very first time from the loop. With each iteration in the loop the worthiness of i rises by 1, so it's necessary to display simply that min ≤ A[i actually] because of this new price of we. That proof comes after from this is of the evaluation and replacement claims. Therefore, R holds true with each iteration with the loop. Eventually, S employs from the ultimate iteration worth of R. This task completes the conventional verification that flow graph exits with the tiniest price of A[1] through A[n] in min.

The algorithm (certainly not the confirmation) shown here's frequently used for example in the initial couple of weeks of introductory encoding classes. It really is quite simple; actually, after researching the algorithm for a short while, most students encourage themselves how the algorithm is appropriate. The confirmation itself takes a lot longer to explain; in addition, it takes far more lengthy to write

compared to the algorithm itself. Consequently, this proof-of-correctness instance highlights two primary difficulties with elegant verification strategies:

- Time. The techniques of formal confirmation are frustrating to perform. Proclaiming the assertions at each move and verifying the reasonable flow in the assertions happen to be both slow functions.

Complexity. Formal confirmation is a sophisticated process. For a few systems with many says and transitions, it really is hopeless to attempt to state and confirm the assertions. This example is especially real for systems which have not been made with formal verification at heart.

Validation

Formal verification is really a particular instance of this more general method of assuring correctness: confirmation. As we have observed in Section 3, there are lots of ways to demonstrate that each of an system's functions performs correctly. Validation may be the counterpart to confirmation, assuring that the machine developers have executed all requirements. Hence, validation makes certain that the developer is definitely building the proper product (based on the specs), and confirmation checks the grade of the execution .There are many various ways to validate an operating-system.

Open Source

A debate has opened up in the program development area over so-called open up source os's (along with other programs), ones that the source program code is freely produced for public evaluation. The arguments happen to be predictable: With open up source, several critics can peruse the program code, presumably finding imperfections, whereas shut (proprietary) source helps it be more challenging for attackers to get and exploit imperfections.

The Linux operating-system is the best example of open up source software, even though way to obtain its predecessor Unix seemed to be also accessible. The open resource idea is finding on: In accordance with a study by IDG Study, reported within the Washington Blog post, 27 per-cent of high-end machines now manage Linux, instead of 41 percent for your Microsoft operating-system, and the open up source Apache internet server outruns Microsoft Web Details Server by 63 percentage to 20 percentage.

Lawton lists extra benefits of available source:

- Expense: As the source code can be acquired to the general public, if the dog owner charges a higher fee, the general public will trade the program unofficially.

- Top quality: The program code can be examined by countless reviewers that are unrelated for the development energy or the company that developed the program.

- Help: Because the public finds imperfections, it may in addition be in the very best situation to propose the fixes for all those flaws.

- Extensibility: The general public can readily body how to increase code to meet up new needs and will reveal those extensions with various other users.

Opponents of open release dispute that presenting the attacker understanding of the look and execution of a bit of code enables a seek out shortcomings and a blueprint for his or her exploitation. Many industrial vendors have compared open source for a long time, and Microsoft happens to be being really vocal in its opposition. Craig Mundie, mature vice us president of Microsoft, claims open source computer software "puts at an increased risk the continuing vitality with the independent software industry". Microsoft favors a design under which it could share source program code of a few of its goods with selected spouses, while still keeping intellectual property privileges. The Alexis de Tocqueville Establishment argues that "terrorists attempting to hack or disrupt U.S. laptop networks will dsicover it easier if the government attempts to change to 'wide open resource' as some communities propose," citing risks against air visitors control or monitoring systems.

But noted personal computer security researchers dispute that wide open or closed supply is not the true issue to look at. Marcus Ranum, leader of Network Airline flight Recorder, has mentioned, "I don't believe making [computer software] open resource contributes to rendering it better by any means. What makes very good software is definitely single-minded concentrate." Eugene Spafford of Purdue College agrees, declaring, "What really can determine whether it's trustable is high quality and care. Was basically it designed properly? Was it designed using proper resources? Did individuals who constructed it use self-control and not squeeze in a lot of characteristics?" Ross Anderson of Cambridge College argues that "you can find more pressing protection challenges for the open up source area. The connections between security and safety and openness is usually entangled with

efforts to use safety mechanisms for industrial benefits, to entrench monopolies, to regulate copyright, and most importantly to regulate interoperability."

Evaluation

Most system buyers (that's, customers or system buyers) aren't security experts. They want the security capabilities, but they aren't usually with the capacity of verifying the accuracy and reliability or adequacy of check coverage, examining the validity of the proof correctness, or figuring out in any some other way a system appropriately implements a safety measures policy. Thus, it really is useful (and often essential) with an independent alternative party assess an operating system's safety. Independent authorities can review certain requirements, design, execution, and proof assurance for something. Because it is effective to truly have a standard technique for an assessment, several schemes have already been devised for structuring an unbiased review.

5.6 Review Question

1. A principle of typically the BellLa Padula model seemed to be not mentioned in this specific chapter. Called the tranquillity principle, it states that this classification of a subject matter or object does not really change although it is being referenced. Explain the objective of the tranquillity principle. Do you know the implications associated with a model where the comfort principle is not genuine?

2. Subjects can obtain objects, but they could also access other subject matter. Describe what sort regarding reference monitor would handle access in the circumstance of a subject doing work on another subject. Explain what sort of research monitor would control gain access to in the case involving two subjects interacting.

3. List the original source and conclusion of all information runs in each of typically the following statements.

a. sum: sum= a+b+c;
b. if a+b < c+d then queen: =0 else q: =1;
c. write (a, b, c);

d. read (a, b, c);
e. case (k) of

   0: d: = 10;

   1 , 2: d: = 20;

   other: d: = 30;

   end; /* case */

f. for i: =min to max do k: =2*k+1;
g. repeat

   a[i]: =0;

   i: =i-1;

   until i ≤ 0;

4. Does the particular system of all subsets of a finite arranged under the operation "subset of" () form a new lattice? Why or the reason why not?

5. Can the user cleared for <secret;{dog, cat, pig}> have accessibility to documents classified inside of each of the next ways within the military safety model?

1. <top secret;dog>
2. <secret;{dog}>
3. <secret;{dog,cow}>
4. <secret;{moose}>
5. <confidential;{dog,pig,cat}>
6. <confidential;{moose}>

6th. According to the BellLa Padula model, what constraints are placed on a couple of active subjects (for instance, two processes) that have to have to send and get signals to and by one another? Justify your response.

7. Write an established of rules combining typically the secrecy controls from the BellLa Padula model together with the honesty controls of the Biba model.

IDOL

8. Demonstrate a way for limited transfer regarding rights in the GrahamDenning model. A limit associated with one is adequate. Of which is, give an approach by which A can easily transfer to B appropriate R, with the accessibility that B can exchange that right to virtually any one other subject. Typically the subject to which M transfers the right are not able to transfer the right, neither can B transfer that again.

9. Explain just what is necessary to supply temporal separation. That is usually, what conditions must become met to ensure two process to be adequately divided?

10. Does the regular Unix operating system work with a nondiscretionary access command? Explain your answer.

10. Why is labeling involving objects a security necessity? That is, why are unable to the trusted computing basic just maintain an accessibility control table with articles for every single object and every single subject?

12. Label honesty is a technique that will ensures that the content label to each object is altered only by the respected computing base. Suggest a new method to implement content label integrity for an information file. Suggest a technique to implement label sincerity for a callable process.

13. Describe a condition when you might desire to allow the safety kernel to violate one particular of the security attributes of the BellLa Padula model.

14. Explain this specific is of the phrase granularity in comparison with access management. Discuss the tradeoff among granularity and efficiency.

15. Explain what sort regarding semaphore could be employed to implement a hidden channel in concurrent running. Explain how concurrent running primitives, for example fork and even join, might be used to be able to implement a covert funnel in concurrent processing.

16. The Unix main system constructions files by using a new tree. Each file will be at a leaf associated with the tree, plus the data file is identified with the (unique) path from the underlying to the leaf. Each and every interior node is the "subdirectory, " which identifies the names with the pathways leading from that client. A user can stop access through a client by restricting access to be able to

IDOL

the subdirectory. Devise the method that uses this kind of structure to implement some sort of discretionary access policy.

17.In the Unix data file system described in this kind of chapter, could a nondiscretionary access policy be described so that an end user has access to some sort of file as long as the consumer has access to most subdirectories higher (closer towards the root) in the record structure? What would get the effect of this particular policy?

18. I/O looks as the source associated with several successful methods involving penetration. Discuss why I/O is hard to safeguarded within a computing system.

5.7 References

1. Security in Computing, Fourth Edition By Charles P. Pfleeger - Pfleeger Consulting Group, Shari Lawrence Pfleeger - RAND Corporation Publisher: Prentice Hall

2. Cryptography and Network Security - Principles and Practice fifth edition Stallings William Publisher: Pearson

3. Cryptography And Network Security 3rd Edition behrouz a forouzan and debdeep mukhopadhyay 3/E Publisher: McGraw Hill Education

4. Cryptography and Network Security, 3e Atul Kahate Publisher: McGraw Hill

IDOL

Chapter 6. Database and Data Mining Security

IDOL

6.0 Introduction

Protecting data reaches the heart of several secure systems, and several users (persons, programs, or devices) depend on a database supervision system (DBMS) to control the protection. Because of this, we spend this chapter towards the security of data source management systems, for example of how software security could be designed and executed for a particular task. There's substantial current fascination with DBMS protection because databases are usually newer than encoding and os's. Databases are crucial to many organization and government corporations, holding information that indicate the organization's center competencies. Normally, when business techniques are reengineered to create them far better and much more in melody with innovative or revised objectives, among the first systems to get careful scrutiny may be the set of directories supporting the business enterprise processes. Thus, directories tend to be more than software-related repositories. Their firm and contents are believed valuable corporate investments that must definitely be carefully protected.

However, the safety provided by data source management systems has already established mixed results. As time passes, we have increased our knowledge of database security issues, and several fine controls have already been designed. But, as you will notice, you may still find more security problems for which you can find no available adjustments.

6.1. Introduction to Databases

We start by describing a data source and defining terminology linked to its work with. We bring on cases from what's named the relational data source because it is among the hottest types. However, all of the concepts described below apply to any kind of database. We first of all define the essential concepts and use them to go over security concerns.

Concept of a Database

A database is really a collection of information and a couple of rules that plan the info by specifying specific relationships on the list of files. Through these guidelines, the user explains a logical structure for the info. The data things are kept in a record, but the correct physical format in the file is definitely of no issue to an individual. A data source administrator is really a person who identifies the

guidelines that organize the info and also handles who must have usage of what elements of the data. An individual interacts with the data source through a plan called a repository manager or perhaps a database management method (DBMS), informally referred to as a front finish.

Components of Databases

The database data file consists of files, all of which consists of one related band of data. As proven in the example of this in Table 6-1, an archive in a brand and address data file includes one label and target. Each record is made up of fields or components, the elementary information things themselves. The areas in the brand and address report are NAME, Target, CITY, Talk about, and ZIP (where ZIP may be the U.S. postal program code). This repository may very well be a two-dimensional desk, where a report is really a row and each discipline of an archive is an component of the table.

Table 6.1. Example of a Database.

ADAMS	212 Market St.	Columbus	OH	43210
BENCHLY	501 Union St.	Chicago	IL	60603
CARTER	411 Elm St.	Columbus	OH	43210

Not every databases is conveniently represented as an individual, compact stand. The databases in Figure 6.1 logically includes three data files with possibly unique employs. These three data files could possibly be represented as you large stand, but that depiction might not improve the energy of or usage of the data.

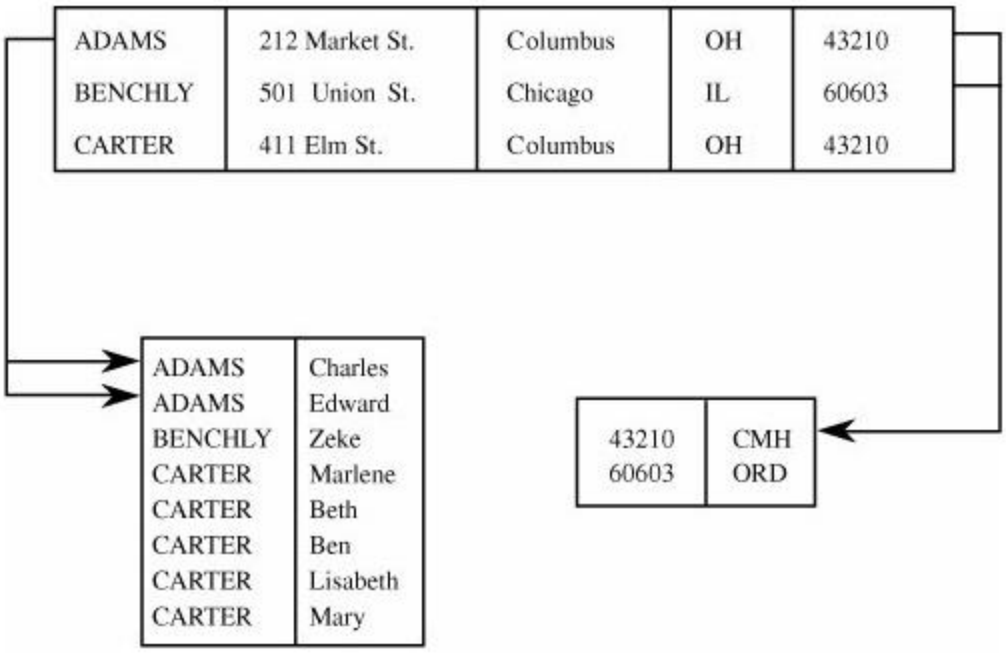


Figure 6-1. Related Parts of a Database.

The logical framework of a databases is named a schema. A specific user could have access to just area of the database, named a subschema. The entire schema of this database in Figure 6-1 is complete in Table 6.2. The three different blocks from the figure are types of subschemas, although different subschemas of the database could be defined. We are able to employ schemas and subschemas to provide to users simply those factors they desire or have to see. For instance, if Stand 6-1 symbolizes the personnel at an organization, the subschema on the low left can record employee titles without revealing private information such as residence address.

Table 6-2. Schema of Database Shown in Figure 6.1.						
Name	**First**	**Address**	**City**	**State**	**Zip**	**Airport**
ADAMS	Charles	212 Market St.	Columbus	OH	43210	CMH
ADAMS	Edward	212 Market St.	Columbus	OH	43210	CMH
BENCHLY	Zeke	501 Union St.	Chicago	IL	60603	ORD
CARTER	Marlene	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Beth	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Ben	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Lisabeth	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Mary	411 Elm St.	Columbus	OH	43210	CMH

The rules of your database recognize the columns with brands. The name of every column is named an attribute in the database. A connection is a group of columns. For instance, using the data source in Table 6-2, we note that NAMEZIP is really a relation formed by firmly taking the Label and ZIP columns, as displayed in Desk 6-3. The connection specifies clusters of connected data beliefs in quite similar way the relationship "mother of" specifies a connection among sets of humans. In this particular case, each cluster includes a pair of factors, a NAME including a ZIP. Other relationships can have extra columns, consequently each cluster might be a triple, a 4-tuple, or an n-tuple (for a few price n) of components.

Queries

Users connect to database administrators through commands for the DBMS that get, modify, include, or delete job areas and records in the database. A command word is named a query. Database supervision systems have exact guidelines of syntax for questions. Many query languages work with an English-like notation, and several derive from SQL, a organised query language actually produced by IBM. We've written the case queries in this particular section to resemble British sentences in order that they are clear to see. For instance, the query

SELECT NAME = 'ADAMS'

retrieves all details having the worth ADAMS inside the NAME field.

The consequence of performing a query is really a subschema. One method to style a subschema of the database is usually by selecting information meeting certain problems. For example, we would select records where ZIP=43210, producing the effect shown in Table 6-4.

Table 6.4. Result of Select Query.

Name	First	Address	City	State	Zip	Airport
ADAMS	Charles	212 Market St.	Columbus	OH	43210	CMH
ADAMS	Edward	212 Market St.	Columbus	OH	43210	CMH
CARTER	Marlene	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Beth	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Ben	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Lisabeth	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Mary	411 Elm St.	Columbus	OH	43210	CMH

Other, more technical, selection criteria will be possible, with reasonable operators such as for example and (Λ) and or (V), and evaluations such as significantly less than (<). A good example of a go for query is

SELECT (ZIP='43210') Λ(NAME='ADAMS')

After having picked records, we might project these details onto a number of attributes. The go for operation identifies particular rows in the database, plus a project functioning extracts the worth from certain job areas (columns) of these records. The consequence of a select-project functioning is the group of values of given attributes to the selected records. For instance, we might pick records meeting the problem ZIP=43210 and task the outcomes onto the capabilities NAME and Initial, as in Table 6.5. The effect is the set of first and previous names of individuals whose addresses have got zip program code 43210.

Table 6-5. Results of Select-Project Query.	
ADAMS	Charles
ADAMS	Edward
CARTER	Marlene
CARTER	Beth
CARTER	Ben
CARTER	Lisabeth
CARTER	Mary

Observe that we don't need to project onto exactly the same attribute(s) which the selection is performed. For example, we are able to create a query applying ZIP and Title but project the effect onto First of all:

SHOW First of all WHERE (ZIP='43210') Λ (NAME='ADAMS')

The result will be a list of the initial names of individuals whose last brands happen to be ADAMS and ZIP is usually 43210.

We can furthermore combine two subschema on a standard element with a join query. The consequence of this operation is really a subschema whose details have exactly the same value for the normal element. For instance, Figure 6.2 implies that the subschema NAMEZIP along with the subschema ZIPAIRPORT could be joined on the normal field ZIP to create the subschema NAMEAIRPORT.

IDOL

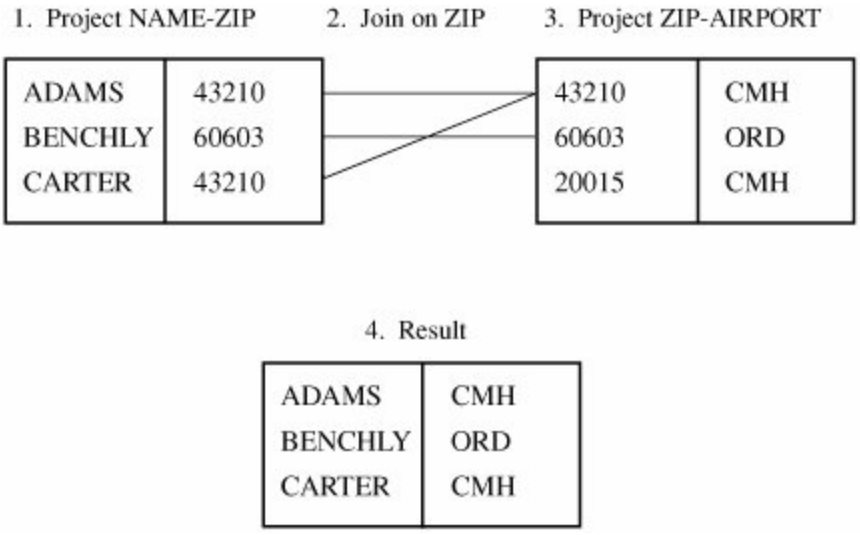


Figure 6-2. Results of Select-Project-Join Query.

Advantages of Applying Databases

The logical concept behind a databases is definitely this: A databases is a solo collection of information, stored and taken care of at one main area, to which lots of people have admission as needed. On the other hand, the actual execution may incorporate some other physical safe-keeping arrangement or admittance. The substance of an excellent database is usually that the users don't realize the physical agreements; the unified reasonable arrangement can be all they find. Because of this, a database gives several benefits over a straightforward file program:

shared access, in order that many users may use one prevalent, centralized group of data

minimal redundancy, in order that individual users don't need to collect and keep maintaining their own models of data

data consistency, in order that a change to some data value influences all customers of the info value

data integrity, in order that data values will be protected against unintentional or malicious unwanted changes

controlled access, in order that only authorized consumers are permitted to view or even to modify files values

A DBMS was created to provide these benefits efficiently. However, normally happens, the goals can conflict with one another. In particular, once we shall see, protection interests can issue with effectiveness. This clash isn't surprising because methods taken up to enforce security usually increase the processing system's dimension or complexity. What's surprising, though, is the fact security interests could also decrease the system's capability to provide files to customers by limiting selected queries that could otherwise seem to be innocuous.

6.2. Security Requirements

The basic security and safety requirements of data source systems aren't unlike those of different computing systems we've studied. The essential problemsaccess command, exclusion of spurious information, authentication of consumers, and reliabilityhave came out in lots of contexts up to now in this reserve. Following is really a list of specifications for database safety measures.

Physical database integrity. The info of a databases are immune system to physical challenges, such as ability failures, and somebody can reconstruct the databases if it's destroyed by way of a catastrophe.

Logical databases integrity. The construction of the data source is maintained. With rational integrity of your database, an adjustment to the worthiness of one discipline does not influence other fields, for instance.

Element integrity. The info within each element happen to be accurate.

Auditability. You'll be able to keep tabs on who or what has got accessed (or improved) sun and rain in the repository.

Access management. A user is usually allowed to obtain only authorized information, and different customers can be limited to different settings of entry (such as for example read or publish).

Individual authentication. Every consumer is positively diagnosed, both for the audit path and for agreement to access specific data.

Availability. Customers can gain access to the database generally and all of the data that they are certified.

We briefly verify each one of these requirements.

Integrity of the Database

If a database is to function as a middle repository of files, users should be able to believe in the accuracy and reliability of the info values. This problem means that the data source administrator should be assured that up-dates are performed simply by authorized people. It also means that the data should be protected from problem, either by another illegal program steps or by another force such as for example fire or perhaps a power disappointment. Two situations make a difference the integrity of your database: once the whole database will be damaged (as occurs, for instance, if its storage space medium is ruined) or when personal data items happen to be unreadable.

Integrity in the database all together is the duty from the DBMS, the operating-system, along with the (human being) computing system manager. Through the perspective in the operating system along with the computing system manager, directories and DBMSs will be files and plans, respectively. Therefore, a proven way of guarding the database all together is to on a regular basis regress to something easier all documents on the machine. These regular backups could be adequate adjustments against catastrophic failure.

It is sometimes important to have the ability to reconstruct the database at the idea of failing. For instance, once the power fails all of a sudden, a bank's clientele may be in the center of making deals or students could be amid registering online for his or her classes. In such cases, you want to have the ability to restore the systems to a well balanced stage without forcing consumers to redo their just lately completed transactions. To take care of these circumstances, the DBMS must retain a log of purchases. For example, imagine the bank operating system is designed in order that a message is usually generated in the log (electronic or papers or both) whenever a transaction is refined. In case of a system disappointment, the system can buy accurate account amounts by reverting into a backup copy from the data source and reprocessing all down the road transactions from your log.

Element Integrity

The integrity of repository elements is certainly their correctness or reliability. Ultimately, authorized customers have the effect of entering correct info into databases. Nevertheless, users and courses make mistakes accumulating data,

computing outcomes, and entering principles. Therefore, DBMSs quite often take special activity to help get errors because they are made also to correct errors once they are inserted.

This corrective activity can be ingested in three ways. First of all, the DBMS can put on field checks, actions that test out for appropriate ideals ready. A field may be required to end up being numeric, an uppercase notice, or among a couple of acceptable people. The check means that a value drops within given bounds or isn't greater than the sum of the the principles in two different fields. These investigations prevent simple problems as the info are joined.

The next means of providing data source integrity is maintaining a new **change log** for the particular database. A change sign lists every change designed to the database; it includes both original and revised values. Using this log, a database administrator could undo any changes that will were made in mistake. For example, a selection fine might erroneously end up being posted against Charles N. Robertson, instead of Charles M. Robertson, flagging Charles W. Robertson as ineligible to take part in varsity athletics. Upon discovering this problem, the database administrator purchases Charles W. 's initial eligibility value from the particular log and corrects typically the database.

Auditability

For a few programs it may be appealing to build an audit report coming from all access (read or perhaps write) to a database. Such a record can easily help to maintain the particular database's integrity, or from least to discover right after the fact who experienced affected which values and even when. A second benefit, even as we see later, is definitely that users can obtain protected data incrementally; that will is, no single accessibility reveals protected data, yet a set of continuous accesses viewed together uncovers the data, just like finding the clues inside a private investigator novel. In this circumstance, an audit trail may identify which clues a good user has already recently been given, as an explained regardless of whether to tell the end user more.

Granularity becomes a good impediment in auditing. Audited events in operating methods are actions like available file or call method; they are seldom simply because specific as write document 3 or execute coaching I. To be valuable for maintaining integrity, databases audit trails should incorporate accesses at the

report, field, and even component levels. This detail will be prohibitive for most data source applications.

Furthermore, it will be possible for a document to get accessed but not really reported to the user, as any time the user performs some sort of select operation. (Accessing a new record or an factor without transferring to typically the user your data received will be called the pass-through issue.) Also, you are able to identify the values of many elements without accessing these people directly. (For example, an individual can ask for the particular average salary in a new group of employees once you know the number involving employees within the group is usually only one.) As a result, a log coming from all data accessed directly may each overstate and understate just what an user actually is aware.

Access Control

Databases happen to be often separated logically simply by user access privileges. With regard to instance, all users could be granted access in order to general data, but simply the personnel department could acquire salary data and even only the marketing office can obtain sales info. Databases are very helpful because they centralize the particular storage and maintenance associated with data. Limited access is definitely both a responsibility in addition to a benefit of this kind of centralization.

The database supervisor specifies who should become allowed use of which information, at the view, connection, field, record, as well as component level. The DBMS need to enforce this policy, giving access to all described data or no gain access to where prohibited. Furthermore, typically the number of modes associated with access can be several. A user or software might have the right in order to read, change, delete, or even append to a worth, add or delete complete fields or records, or even reorganize the entire repository.

Superficially, access control with regard to a database seems just like access control for functioning systems or any additional element of a computer system. Nevertheless , the databases problem is more complex, since we see throughout this particular chapter. Operating system things, for instance files, are not related items, whereas records, career fields, and elements are associated. Although an user are not able to determine the contents regarding one file by reading through others, an user may be able to decide one data element simply by reading others.

The particular problem of obtaining info values from others is usually called inference, and all of us ponder over it in depth later on in this chapter.

This is important to observe that you can gain access to data by inference with out the need for immediate access to the safe object itself. Restricting inference may mean prohibiting specific paths to prevent probable inferences. Yet , restricting gain access to to control inference likewise limits queries from customers who do not want unauthorized access to principles. Moreover, attempts to check out requested accesses for potential unacceptable inferences may truly degrade the DBMS's efficiency.

Finally, size or granularity is different between os objects and database items. An access control set of several hundred files is definitely much simpler to implement compared to an access control listing for a database using several hundred files regarding perhaps a hundred career fields each. Size affects the particular efficiency of processing.

End user Authentication

The DBMS may require rigorous user authentication. For instance, a DBMS might insist that a great user pass both particular password and time-of-day investigations. This authentication supplements the particular authentication performed from the working system. Typically, the DBMS runs as an software program on top regarding the operating system. This product design means that right now there is no trusted course from the DBMS for the operating system, so the particular DBMS has to be suspicious involving any data it gets, including user authentication. Therefore, the DBMS will perform its own authentication.

Availability

A DBMS has areas of both a program plus a system. It is usually a program that makes use of other hardware and computer software resources, yet to a lot of users it is typically the only application run. Consumers often take the DBMS for granted, employing this as an essential instrument which to perform specific tasks. When the method is not availablebusy offering other users or lower to be repaired or even upgradedthe users are really aware of a DBMS's unavailability. For example, a couple of users may request typically the same record, and typically the DBMS must arbitrate; one particular user is bound in order to be

denied access regarding a while. And also the DBMS may withhold unprotected files to avoid revealing safeguarded data, leaving the seeking user unhappy. We look at these problems in extra detail later in this kind of chapter. Problems like these types of result in high accessibility requirements for a DBMS.

Integrity/Confidentiality/Availability

The three features of computer securityintegrity, discretion, and availabilityclearly relate with databases management systems. As we all have described, integrity is applicable to the individual aspects of a database as effectively as to the data source as a whole. As a consequence, integrity is a key concern inside the design involving database management systems. Operating more closely at sincerity issues in the up coming section.

Confidentiality is some sort of key issue with data source because of the inference problem, whereby an end user can access sensitive information indirectly.

6.3. Reliability and Integrity

Databases amalgamate data by many sources, and customers expect a DBMS in order to provide access to the particular data inside a reliable approach. When software engineers state that software has trustworthiness, they mean that typically the software runs for quite long amounts of time without screwing up. Users certainly expect some sort of DBMS to get reliable, given that the data are often major to business or company needs. Moreover, users trust their data to some sort of DBMS and rightly anticipate it to protect the particular data from loss or even damage. Concerns for stability and integrity are common security issues, but they will are more apparent along with databases.

A DBMS protections against loss or harm in several ways in which many of us study them within this segment. However, the controls we all consider are not overall: No control can avoid an authorized user by inadvertently entering an satisfactory but incorrect value.

Databases concerns about reliability and even integrity can be looked at from 3 dimensions:

Database integrity: worry that the database seeing that a whole is guarded against damage, as coming from the failure of a new disk drive or maybe the data

corruption of the master data source index. These concerns will be addressed by operating method integrity controls and recuperation procedures.

Element integrity: problem the value of some sort of specific data element is usually written or changed sole by authorized users. Appropriate access controls protect some sort of database from corruption by simply unauthorized users.

Element precision: concern that only perfect values are written in to the elements of a repository. Checks on the beliefs of elements can aid prevent insertion of inappropriate values. Also, constraint factors can detect incorrect ideals.

Protection Features from the Operating System

we discussed the defense an operating system offers for its users. A good accountable system administrator back up the files associated with a database periodically together with other user documents. The files are safeguarded during normal execution in opposition to outside access by typically the operating system's standard gain access to control facilities. Finally, typically the operating system performs particular integrity checks for those info as a part involving normal read and publish operations for I/O gadgets. These controls provide standard security for databases, nevertheless the database manager need to enhance them.

Two-Phase Up-date

A serious problem with regard to a database manager is definitely the failure with the calculating system in the center of modifying data. In case the data item in order to be modified was some sort of long field, half involving the field might demonstrate the new value, as the other half would consist of the old. Even when errors of this form were spotted easily (which they are not), some sort of more subtle problem happens when several fields are usually updated with out single discipline appears to be throughout obvious error. The remedy for this problem, proposed very first by Lampson and Sturgis and even adopted by most DBMSs, works on the two-phase update.

Update Technique

During the first of all phase, the intent level, the DBMS gathers the particular resources it needs to accomplish the update. It might gather data, create dummy records, open files, secure out others, and determine final answers; in quick, it does everything to be able to plan for the update, yet it makes no shifts to the database. Typically the first phase is repeatable an unlimited number involving times as it takes simply no permanent action. If typically the system fails during setup of the first stage, no harm is completed because all these actions can be restarted plus repeated after the technique resumes processing.

The final event of the primary phase, called committing, entails the writing of a new commit flag for the repository. The commit flag methods that the DBMS is definitely long gone the level of no return: Following committing, the DBMS starts making permanent changes.

The particular second phase makes typically the permanent changes. During the particular second phase, no steps from before the make can be repeated, nevertheless the update activities involving phase two can likewise be repeated as usually as needed. If the particular system fails during the particular second phase, the repository may contain incomplete information, but the system can easily repair these data simply by performing all activities from the second phase. After typically the second phase has recently been completed, the database will be again complete.

Two-Phase Update Example

Suppose a database contains a listing of any company's office items. The company's main stockroom stores papers, pens, paper videos, and so on, and the various departments requisition products as they will need them. The business buys in large to get the best costs. Each department includes a budget for business office supplies, so there's a charging mechanism where the expense of supplies is retrieved from the office. Also, the fundamental stockroom monitors levels of supplies readily available in order to order new materials when the share becomes low.

Suppose the procedure begins using a requisition from your accounting team for 50 bins of paper videos. Assume that we now have 107 containers in inventory and a fresh order is positioned if the number in stock ever before comes below

100. Listed below are the steps implemented following the stockroom gets the requisition.

1. The stockroom bank checks the database to find out that 50 bins of paper videos are readily available. Or even, the requisition will be rejected along with the transaction is completed.

2. If enough papers clips come in share, the stockroom deducts 50 from your inventory physique in the data source (107 - 50 = 57).

3. The stockroom costs accounting's supplies finances (also inside the repository) for 50 bins of paper videos.

4. The stockroom bank checks its remaining amount readily available (57) to find out whether the left over quantity will be below the reorder stage. Because it will be, a see to order even more paper clips is definitely generated, and that is usually flagged as "on purchase" inside the database.

5. A delivery buy is prepared, allowing 50 containers of paper videos to be delivered to accounting.

All five of the steps should be finished in the purchase listed for that database to become accurate and then for the transaction to get processed correctly.

Suppose failing develops while these tips are being refined. If the disappointment occurs before step one 1 is finished, there is absolutely no harm as the entire transaction could be restarted. Even so, during ways 2, 3, and 4, alterations are created to elements inside the database. In case a failure occurs after that, the values inside the database are usually inconsistent. Worse, the deal can't be reprocessed just because a requisition will be deducted twice, or perhaps a department will be charged double, or two shipping orders will be prepared.

Whenever a two-phase commit can be used, shadow values will be maintained for important data factors. A shadow files value can be computed and placed locally through the intent phase, which is copied to the specific database through the commit period. The operations over the database will be performed the following for just a two-phase commit.

Intent:

IDOL

1. Check the worthiness of COMMIT-FLAG within the database. If it's set, this stage cannot be conducted. Halt or loop, checking out COMMIT-FLAG until it isn't set.

2. Compare amount of boxes of papers clips readily available to amount requisitioned; if extra are usually requisitioned than happen to be readily available, halt.

3. Compute TCLIPS = ONHAND - REQUISITION.

4. Obtain BUDGET, the existing supplies budget staying for accounting office. Compute TBUDGET = Finances - Price, where COST may be the expense of 50 containers of clips.

5. Examine whether TCLIPS is usually below reorder level; if so, establish TREORDER = A fact; else arranged TREORDER = FALSE.

Commit:

1. Arranged COMMIT-FLAG in databases.

2. Duplicate TCLIPS to Videos in database.

3. Backup TBUDGET to Finances in database.

4. Duplicate TREORDER to REORDER in data source.

5. Prepare notice to provide paper videos to accounting division. Indicate transaction accomplished in log.

6. Unset COMMIT-FLAG.

With this case, each step of this intent phase will depend simply on unmodified beliefs from the databases and the prior outcomes of the intent period. Each variable you start with T is really a shadow variable applied only in this particular transaction. The ways of the purpose phase could be repeated an endless number of instances without influencing the integrity on the database.

After the DBMS commences the commit period, it creates a commit flag. When this flag is defined, the DBMS won't perform any actions of the objective phase. Intent tips cannot be done after committing because repository values are revised inside the commit phase. See, however, which the steps from the commit phase

IDOL

could be repeated an unrestricted number of periods, again without negative influence on the correctness in the values within the database.

The one left over flaw with this logic comes about if the machine fails after composing the "transaction entire" message within the log but before clearing the commit flag inside the database. This is a simple matter to function backward with the transaction log to get completed transactions that the commit flag continues to be set also to clear out those flags.

Redundancy/Internal Consistency

Many DBMSs manage more information to detect inner inconsistencies in information. The additional facts ranges from the few check parts to duplicate or shadow job areas, with regards to the importance of the info.

Error Recognition and Correction Codes

One type of redundancy is mistake recognition and correction rules, such as for example parity parts, Hamming rules, and cyclic redundancy investigations. These codes could be applied to solitary fields, documents, or the complete database. Whenever a data item is positioned in the repository, the appropriate test codes will be computed and placed; whenever a data item will be retrieved, an identical check code is certainly computed and set alongside the stored value. In case the values happen to be unequal, they indicate towards the DBMS an error has happened in the data source. A few of these codes explain the place in the error; others demonstrate precisely what the right value ought to be. The more info provided, the greater space necessary to store the rules.

Shadow Fields

Entire qualities or entire data could be duplicated inside a database. If the info are usually irreproducible, this next copy can offer an immediate alternative if one is detected. Naturally, redundant fields demand substantial space for storage.

Recovery

Along with these error correction functions, a DBMS can manage a log of individual accesses, particularly alterations. In case of failing, the database can be reloaded from the backup copy and everything later changes will be then applied from your audit log.

Concurrency/Consistency

Database systems tend to be multiuser methods. Accesses by two customers sharing exactly the same database should be constrained in order that neither inhibits the other. Straight forward locking is performed with the DBMS. If two customers attempt to browse the same data merchandise, there is absolutely no issue because both have the same value.

If both customers try to adjust the same files items, we usually assume that there surely is no issue because each has learned what to publish; the value being written will not depend on the prior value of the info item. Even so, this supposition isn't quite accurate.

To observe how concurrent modification will get us into problems, guess that the database includes couch reservations for a specific airline flight. Broker A, choosing a chair for traveler Mock, submits a query to get which seats remain available. The representative has learned that Mock prefers the right aisle seat, plus the agent discovers that car seats 5D, 11D, and 14D are usually open. At exactly the same time, Agent B is wanting to book seating for a family group of three going together. In reaction to a query, the databases shows that 8ABC and 11DEF will be the two remaining sets of three adjacent unassigned car seats. Adviser A submits the upgrade command

SELECT (SEAT-NO = '11D')

ASSIGN 'MOCK,E' TO PASSENGER-NAME

while Realtor B submits the revise sequence

SELECT (SEAT-NO = '11D')

ASSIGN 'EHLERS,P' TO PASSENGER-NAME

in addition to commands for car seats 11E and 11F. After that two passengers have already been booked in to the same couch (which may be uncomfortable, to state minimal).

Both agents contain acted appropriately: Each looked for a summary of empty seats, decided one seat from list, and kept up to date the database showing to whom the seating was assigned. The issue in this example is the moment delay

between browsing a value from your database and posting a modification of this value. Through the delay period, another user provides accessed exactly the same data.

To resolve this issue, a DBMS snacks the complete queryupdate routine as an individual atomic procedure. The command from agent must nowadays resemble "browse the current worth of seats PASSENGER-NAME for couch 11D; if it's 'UNASSIGNED', change it to 'MOCK,E' (or 'EHLERS,P')." The readmodify circuit must be accomplished as an continuous item without letting any other consumers usage of the PASSENGER-NAME discipline for seating 11D. The next agent's question to book wouldn't normally be looked at until following the first agent's have been completed; in those days, the worthiness of PASSENGERNAME would no more be 'UNASSIGNED'.

A final trouble in concurrent entry is readwrite. Assume one user is usually updating a worth when a 2nd user wishes to learn it. When the read is performed as the write is happening, the audience may receive files that are simply partially updated. As a result, the DBMS locks any read demands until a write offers been completed.

Monitors

The monitor may be the unit of your DBMS in charge of the structural integrity with the database. A keep an eye on can check ideals being entered to make sure their uniformity with all of those other data source or with features of this field. For instance, a keep track of might reject alphabetic personas for the numeric discipline. We discuss more than a few forms of screens.

Range Comparisons

A range comparison watch tests each different value to make sure that the value is at an acceptable collection. If the info value is beyond your range, it really is rejected rather than entered in to the database. For instance, the number of dates may be 131, "/," 112, "/," 19002099. A far more sophisticated range take a look at might limit your day part to 130 for weeks with 1 month, or it could take into account step year for Feb.

Range comparisons may also be practical for numeric volumes. For example, an income field may be limited by $200,000, or how big is a house may be

constrained being between 500 and 5,000 rectangular feet. Variety constraints may also apply to additional data getting a predictable form.

Transition Constraints

State constraints explain hawaii of the correct database. Move constraints describe disorders necessary before improvements can be put on a database. For instance, before a fresh employee could be put into the database, there should be a position amount in the repository with reputation "vacant." (That's, an empty slot machine must are present.) Furthermore, following the employee is included, exactly one slot machine game must be altered from "vacant" to the amount of the new worker.

Simple range assessments and filters could be implemented within just about all database management techniques. However, a lot more sophisticated point out and change constraints can demand special methods for testing. Like user-written procedures will be invoked with the DBMS every time an action should be checked.

6.4. Private Data

Some databases incorporate what is known as sensitive files. As an operating definition, why don't we say that vulnerable data are info that should certainly not be made general population. Determining which files items and grounds are sensitive is dependent both on the average person database along with the underlying so this means of the info. Obviously, some directories, like a public collection catalog, consist of no sensitive information; other databases, such as for example defense-related ones, happen to be totally sensitive. Both of these casesnothing vulnerable and everything sensitiveare easy and simple to handle since they can be included in access controls for the database all together. Someone either can be or isn't an authorized end user. These controls are given by the operating-system.

The more challenging problem, that is also a lot more interesting one, may be the case where some however, not every one of the elements within the database are delicate. There could be varying examples of sensitivity. For instance, a university data source might contain university student data comprising name, school funding, dorm, drug work with, sex, car parking fines, and competition. A

good example of this database is usually shown in Table 6-6. Title and dorm are most likely the least very sensitive; school funding, parking fines, and medicine use the many; sex and contest somewhere among. That is, lots of people may have respectable access to label, some to love-making and competition, and relatively very few to school funding, parking fines, or medication use. Indeed, understanding of the lifetime of some domains, such as medicine make use of, may itself turn out to be sensitive. Thus, safety concerns not merely the data factors but additionally their framework and meaning

Table 6.6. Sample Database.

Name	Sex	Race	Aid	Fines	Drugs	Dorm
Adams	M	C	5000	45	1	Holmes
Bailey	M	B	0	0	0	Grey
Chin	F	A	3000	20	0	West
Dewitt	M	B	1000	35	3	Grey
Earhart	F	C	2000	95	1	Holmes
Fein	F	C	1000	15	0	West
Groff	M	C	4000	0	3	West
Hill	F	B	5000	10	2	Holmes
Koch	F	C	0	0	1	West
Liu	F	A	0	10	2	Grey
Majors	M	C	2000	0	2	Grey

Furthermore, we should consider different examples of sensitivity. For example, although all of them are highly vulnerable, the school funding, parking fines, and drug-use domains may not contain the same forms of access constraints. Our security specifications may demand a few people come to be authorized to find out each discipline, but nobody be authorized to find out all three. The task of the entry control problem would be to limit consumers' access in order to obtain only the info to that they have legitimate admittance. Alternatively, the gain access to control problem pushes us to make sure that sensitive data aren't to be unveiled to unauthorized persons.

Several factors could make data sensitive.

- **Inherently sensitive**. The worthiness itself could be so revealing that it's sensitive. Examples will be the areas of defensive missiles or the median earnings of barbers in the town with only 1 barber.

- **From a hypersensitive source**. The foundation of the info may suggest a dependence on confidentiality. A good example is data from an informer whose identification would be jeopardized if the info were disclosed.

- **Declared hypersensitive**. The repository administrator or who owns the data could have declared the info to be very sensitive. Examples are labeled military files or the brand of the private donor of a bit of art.

- **Part of any sensitive attribute or perhaps a sensitive record**. In a very database, a whole attribute or report may be categorised as sensitive. Cases are the pay attribute of the personnel database or perhaps a record talking about a secret area mission.

- **Sensitive with regards to previously disclosed details**. Some data grow to be sensitive in the current presence of other data. For instance, the longitude coordinate of the secret silver mine unveils little, however the longitude coordinate with the latitude coordinate pinpoints the mine.

Many of these factors should be considered to ascertain the sensitivity of the info.

Access Decisions

Understand that a data source administrator is really a person who determines what data ought to be in the databases and who must have usage of it. The databases administrator considers the necessity for different consumers to know specific details and decides who must have what access. Selections of the databases administrator derive from an access plan.

The database boss or DBMS is really a program that functions on the data source and auxiliary handle information to employ the decisions from the access plan. We state that the repository manager decides allowing user x to gain access to data y. Evidently, an application or device cannot choose anything; it really is more precise to state that this program performs the recommendations where x accesses y as a means of utilizing the policy founded by the repository administrator. (You now understand why we utilize the simpler wording.) To help keep explanations concise, we once in a while describe programs as though they can perform human thought techniques.

The DBMS may take into consideration several variables when determining whether allowing an entry. These factors consist of availability of the info, acceptability in the admittance, and authenticity of an individual. We grow on these three aspects below.

Availability of Data

A number of required elements could be inaccessible. For instance, if a consumer is updating various fields, other customers' accesses to prospects fields should be blocked briefly. This blocking means that users usually do not receive inaccurate details, like a new street tackle with an good old city and status, or a latest code element with old records. Blocking is normally temporary. When undertaking an up-date, a user may need to block usage of several grounds or several data to guarantee the consistency of files for others.

Acceptability of Access

A number of values with the record could be sensitive rather than accessible by the overall end user. A DBMS shouldn't release sensitive files to unauthorized folks.

Deciding what's sensitive, however, isn't as simple since it sounds, as the fields may possibly not be directly asked for. A user could have asked for several records which contain sensitive data, however the user's purpose might have been only to task the principles from particular grounds that aren't sensitive. For instance, a user with the database found in Stand 6-6 may ask the Brand and DORM of any university student for whom FINES isn't 0. The precise price of the delicate field FINES isn't disclosed, although "not 0" is really a partial disclosure. Even though a sensitive worth isn't explicitly provided, the database director may deny accessibility on the lands that it uncovers information an individual is not approved to have.

Alternatively, an individual may choose to derive a nonsensitive statistic in the sensitive data; for instance, if the common financial aid worth does not show you any individual's school funding value, the databases management technique can safely give back the average. Even so, the average of 1 data price discloses that worth.

IDOL

Guarantee of Authenticity

Certain attributes of an individual external for the database can also be viewed as when permitting accessibility. For example, to improve security, the databases administrator may allow someone to obtain the database simply at times, such as for example during working hrs. Previous user demands can also be considered; repeated demands for exactly the same data or demands that exhaust a particular category of facts enable you to learn all components in a collection when a primary query isn't allowed. Once we shall see, vulnerable data can often be revealed by blended results from different less sensitive questions.

Forms of Disclosures

Data could be sensitive, but therefore can their attributes. In this area, we note that even descriptive information regarding data (such as for example their living or if they have a component that's zero) is really a type of disclosure.

Exact Data

The most critical disclosure may be the exact price of a hypersensitive data product itself. An individual may understand that sensitive data are increasingly being requested, or an individual may request basic data without understanding that some of it really is hypersensitive. A faulty data source manager could even deliver sensitive info by accident, minus the user's having required it. In every of these instances the result may be the identical: The security and safety of the vulnerable data is breached.

Bounds

Another exposure will be disclosing bounds on the sensitive value; that's, indicating a sensitive benefit, y, can be between two worth, L and H. Often, with a narrowing technique not necessarily unlike the binary lookup, an individual may first decide that $L \leq y \leq H$ and look at whether $L \leq y \leq H/2$, etc, thereby permitting an individual to determine con to any preferred perfection. In another circumstance, merely revealing a value like the athletic scholarship funds or the amount of CIA agents surpasses a quantity might be a critical breach of safety.

Sometimes, on the other hand, bounds certainly are a useful solution to present sensitive files. It's quite common to release higher and lower bounds for

information without identifying the precise records. For instance, an organization may declare that its wages for programmers range between $50,000 to $82,000. If you're a programmer gaining $79,700, it is possible to presume that you will be fairly nicely off, which means you have the info you want; nevertheless, the announcement will not disclose that are the highest- and lowest-paid developers.

Negative Result

Sometimes we are able to phrase a query to find out a negative outcome. That is, we are able to know that z isn't the worthiness of y. For instance, understanding that 0 isn't the total amount of felony convictions for an individual reveals that the individual was convicted of the felony. The difference between 1 and 2 or 46 and 47 felonies isn't as sensitive because the difference between 0 and 1. Consequently, disclosing a value isn't 0 could be a significant disclosure. Likewise, if a university student does not show up on the honors checklist, it is possible to infer that the individuals grade point common can be below 3.50. These details is not as well revealing, however, as the range of level level averages from 0.0 to 3.49 is quite wide.

Existence

In some instances, the presence of data can be itself a hypersensitive piece of info, whatever the actual value. For instance, an employer might not want employees to learn that their usage of long distance mobile phone lines has been monitored. In cases like this, discovering an extended DISTANCE field within a personnel document would reveal hypersensitive data.

Probable Value

Finally, it might be possible to look for the probability a certain element includes a certain price. To observe how, suppose you intend to find out if the president of america is registered within the Tory party. Realizing that the president can be in the repository, you post two queries towards the database:

How many folks have 1600 Pa Avenue as their formal residence? (Reply: 4)

How many folks have 1600 Pa Avenue as their standard residence and also have YES because the price of TORY? (Reply: 1)

From these inquiries you conclude there's a 25 percent possibility that the us president is a authorized Tory.

Summary of Partial Disclosure

We have viewed several types of how a safety problem can end up if attributes of sensitive info are revealed. Observe that a number of the techniques we introduced used information regarding the data, instead of immediate access to the info, to infer delicate results. An effective security tactic must guard against both immediate and indirect disclosure.

We have witnessed several types of how a safety problem can end result if qualities of sensitive files are revealed. Observe that a number of the techniques we introduced used information regarding the data, instead of immediate access to the info, to infer delicate results. An effective security tactic must guard against both primary and indirect disclosure.

Safety versus Precision

Our examples include illustrated how hard it is to find out which data are usually sensitive and how exactly to protect them. The problem is complicated by way of a desire to discuss nonsensitive info. For causes of confidentiality you want to disclose just those data that aren't sensitive. This outlook induces a conservative viewpoint in identifying what data to reveal: less is preferable to more.

Alternatively, consider the customers of the info. The conservative idea indicates rejecting any query that mentions a delicate field. We might thereby reject various sensible and nondisclosing concerns. For instance, a researcher might want a summary of grades for several students using drug treatments, or perhaps a statistician may require lists of wages for all adult men and for several women. These concerns probably usually do not compromise the identification of anybody. You want to disclose just as much data as you possibly can so that consumers of the repository get access to the data they want. This goal, known as precision, aims to safeguard all sensitive info while revealing just as much nonsensitive data as you possibly can.

We are able to depict the partnership between safety and accuracy with concentric circles. As Figure 6.3 displays, the sensitive info in the middle circle ought to be carefully concealed. The exterior band represents information we

willingly disclose in reaction to queries. But we realize that an individual may come up with bits of disclosed info and infer various other, more deeply invisible, data. The amount displays us that under the outer layer could be yet extra nonsensitive information that an individual cannot infer.

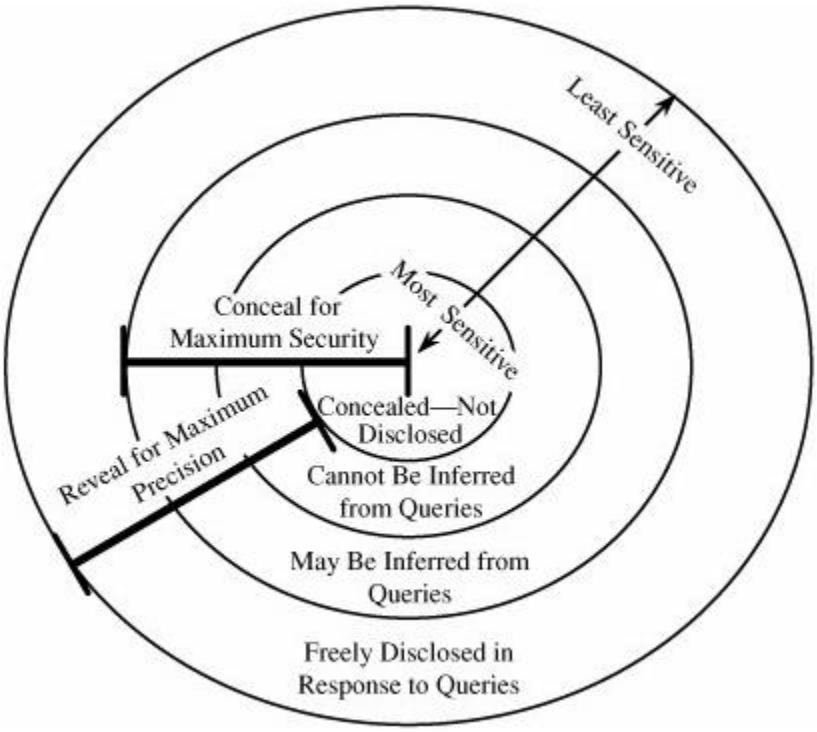


Figure 6.3. Security versus Precision.

6.5. Inference

Inference is really a solution to infer or derive hypersensitive info from nonsensitive information. The inference difficulty is a refined vulnerability in data source security.

The repository in Table 6-7 might help demonstrate the inference issue. Recall that Help is the level of financial aid students receives. FINES may be the amount of auto parking fines nonetheless owed. DRUGS may be the consequence of a drug-use study: 0 stands for never employed and 3 signifies frequent user. Clearly

this information ought to be kept private. We presume that Support, FINES, and Prescription drugs are sensitive areas, although only once the values are usually related to a particular individual. Within this section, we seem at methods to determine sensitive info values through the database.

Table 6.7. Sample Database (repeated).

Name	Sex	Race	Aid	Fines	Drugs	Dorm
Adams	M	C	5000	45	1	Holmes
Bailey	M	B	0	0	0	Grey
Chin	F	A	3000	20	0	West
Dewitt	M	B	1000	35	3	Grey
Earhart	F	C	2000	95	1	Holmes
Fein	F	C	1000	15	0	West
Groff	M	C	4000	0	3	West
Hill	F	B	5000	10	2	Holmes
Koch	F	C	0	0	1	West
Liu	F	A	0	10	2	Grey
Majors	M	C	2000	0	2	Grey

Direct Attack

In a primary attack, a person tries to find out values of hypersensitive fields by searching for them straight with concerns that yield very few records. Probably the most successful technique would be to contact form a query so particular that it complements exactly one files item.

In Table 6.7, a vulnerable query may be

List Name where

SEX=M ΛDRUGS=1

This query discloses that for report ADAMS, Drug treatments=1. However, it really is an obvious strike because it chooses men and women for whom Prescription drugs=1, plus the DBMS might reject the query since it selects details for a particular price of the very sensitive attribute DRUGS.

A less noticeable query is

List Name where

(SEX=M ΛDRUGS=1) V

(SEX ≠M Λ SEX ≠ F) V

(DORM=AYRES)

At first glance, this query seems as if it will conceal drug consumption by selecting additional non-drug-related records aswell. On the other hand, this query nonetheless retrieves only 1 record, disclosing a title that corresponds to the very sensitive DRUG price. The DBMS must know that Love-making has simply two possible principles so the next clause will pick no records. Even though that were achievable, the DBMS would should also understand that no records are present with DORM=AYRES, despite the fact that AYRES might actually be a satisfactory price for DORM.

Agencies that publish private statistical data, like the U.S. Census Bureau, usually do not reveal results whenever a few people constitute a large percentage of a class. The principle of "n things over k pct" implies that data ought to be withheld if n products signify over k percentage of the effect reported. In the last case, the main one person selected presents completely of the info documented, so there will be no ambiguity about which particular person suits the query.

Indirect Attack

Another procedure, utilized by the U.S. Census Bureau along with other organizations that accumulate sensitive data, would be to release only research. The organizations control individual titles, addresses, or various other characteristics where a single specific can be acknowledged. Only neutral studies, such as amount, count number, and mean, are usually released.

The indirect episode looks for to infer your final result predicated on a number of intermediate statistical benefits. But this process requires work beyond your database itself. Specifically, a statistical harm seeks to utilize some apparently private statistical solution to infer particular data. In the next sections, we provide several types of indirect episodes on directories that report data.

Sum

An strike by sum attempts to infer a worth from a noted sum. For instance, with the trial database in Table 6-7, it could seem secure to report college student aid

full by intercourse and dorm. This type of report is proven in Table 6-8. This apparently innocent report unveils that no feminine living in Gray is receiving school funding. Thus, we are able to infer that any feminine living in Gray (such as for example Liu) is obviously not receiving school funding. This approach normally we can determine a poor result.

Table 6-8. Sums of Financial Aid by Dorm and Sex.				
	Holmes	Grey	West	Total
M	5000	3000	4000	12000
F	7000	0	4000	11000
Total	12000	3000	8000	23000

Count

The count could be combined with sum to create some a lot more revealing results. Typically these two figures are released for your database to permit users to find out average worth. (Conversely, if matter and mean happen to be released, sum could be deduced.)

Table 6-9 reveals the matter of information for pupils by dorm and intercourse. This table will be innocuous alone. Combined with sum table, on the other hand, this table shows that both guys in Holmes and Western are receiving school funding in the quantity of $5000 and $4000, respectively. We are able to obtain the brands by choosing the subschema of Label, DORM, that is not sensitive since it delivers just low-security info on the complete database.

Table 6-9. Count of Students by Dorm and Sex.				
	Holmes	Grey	West	Total
M	1	3	1	5
F	2	1	3	6
Total	3	4	4	11

IDOL

Mean

The arithmetic mean (average) permits exact disclosure in the event the attacker can change the subject populace. Being a trivial example, take into account salary. Given the amount of staff, the mean wage for an organization as well as the mean salary of most workers except the us president, it is possible to compute the president's income.

Median

By a just a little more complicated method, we can identify an individual price from medians. The harm requires finding options having one level of intersection that occurs to be accurately in the centre, as revealed in Figure 6.4.

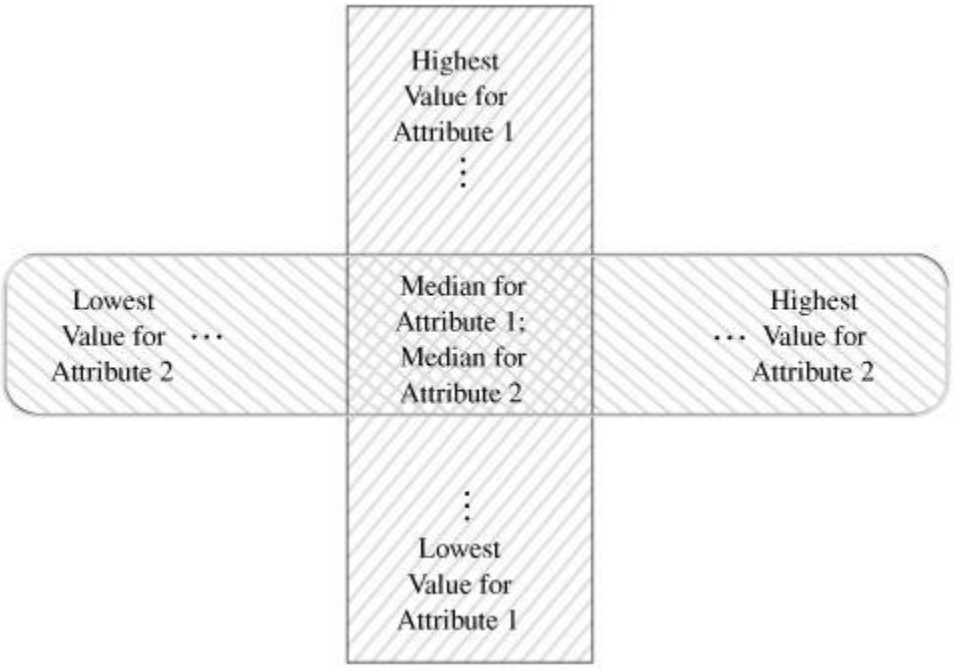


Figure 6.4. Intersecting Medians.

For example, inside our sample database, you can find five men and three individuals whose drug work with value can be 2. Arranged to be able of help, these lists happen to be shown in Table 6-10. Observe that Majors may be the only name widespread to both listings, and handily that name will be in the center of each list. A person working at medical Clinic could probably learn that Majors is really a white man whose drug-use credit score is certainly 2. That data identifies

Majors because the intersection of the two listings and pinpoints Majors' school funding as $2000. On this case, the queries

Table 6-10. Inference from Median of Two Lists.			
Name	**Sex**	**Drugs**	**Aid**
Bailey	M	0	0
Dewitt	M	3	1000
Majors	M	2	2000
Groff	M	3	4000
Adams	M	1	5000
Liu	F	2	0
Majors	M	2	2000
Hill	F	2	5000

q = median(AID where Making love = M)

p = median(AID where Prescription drugs = 2)

reveal the precise financial aid sum for Majors.

Tracker Attacks

As already described, database management methods may conceal information when a few entries constitute a large percentage of the info exposed. A tracker episode can fool the data source manager into seeking the desired data through the use of additional questions that produce smaller outcomes. The tracker offers additional records being retrieved for just two different queries; both sets of documents cancel one another out, leaving simply the statistic or info desired. The solution is by using intelligent cushioning of two questions. Quite simply, instead of attempting to identify a distinctive value, we ask n - 1 various other values (where you can find n values within the database). Provided n and n - 1, we are able to easily compute the required single element.

For instance, assume we need to know how many feminine Caucasians reside in Holmes Hall. A query posed may be

Count ((SEX=F) Λ (RACE=C) Λ (DORM=Holmes))

The database control system might check with the database, discover that the answer will be 1, and won't remedy that query because a person record dominates the consequence of the query.

However, further evaluation in the query we can track sensitive info through nonsensitive questions.

The query

q=count ((SEX=F) Λ (RACE=C) Λ (DORM=Holmes))

is of the proper execution

q = count (a Λ b Λ c)

Utilizing the rules of reasoning and algebra, we are able to change this query to

q = count(a Λ b Λ c) = count (a) count(a Λ¬ (b Λ c))

Thus, the initial query is the same as

count (SEX=F)

minus

count ((SEX=F) Λ ((RACE ≠C) Λ (DORM ≠ Holmes)))

Because count(a) = 6 and count(a Λ ¬ (b Λ c)) = 5, we are able to identify the suppressed worth quickly: 6 - 5 = 1. In addition, neither 6 nor 5 is really a sensitive count.

Linear Technique Vulnerability

A tracker is really a specific situation of a far more standard vulnerability. With just a little reasoning, algebra, and fortune in the syndication of the data source contents, it might be possible to create some queries that profits results associated with several different packages. For example, the next method of five concerns will not overtly uncover any one c value from database. On the other hand, the inquiries' equations could be solved for every of the mysterious c values, exposing them all.

IDOL

$$
\begin{aligned}
q_1 &= c_1 + c_2 + c_3 + c_4 + c_5 \\
q_2 &= c_1 + c_2 + c_4 \\
q_3 &= c_3 + c_4 \\
q_4 &= c_4 + c_5 \\
q_5 &= c_2 + c_5
\end{aligned}
$$

To observe how, use standard algebra to notice that q1 - q2 = c3 + c5, and q3 - q4 = c3 - c5. In that case, subtracting both of these equations, we acquire c5 = ((q1 - q2) - (q3 - q4))/2. After we know c5, we are able to derive others.

Actually, this attack could also be used to obtain effects apart from numerical ones. Remember that people can apply rational regulations to and (Λ) and or (V), regular operators for data source concerns, to derive ideals from a group of logical expressions. For instance, each appearance might stand for a query requesting precise data rather than counts, like the equation

q = s1 V s2 V s3 V s4 V s5

The consequence of the query is really a set of documents. Using reasoning and place algebra in a way much like our numerical illustration, we can meticulously determine the specific values for every from the si.

Adjustments for Statistical Inference Attacks

Denning and Schlorer offer a good survey of approaches for maintaining safety in directories. The controls for several statistical attacks happen to be similar. Essentially, you can find two methods to drive back inference problems: Either handles are put on the inquiries or controls happen to be applied to singular items within the databases. As we have observed, it is complicated to find out whether confirmed query discloses very sensitive data. So, query controls work primarily against immediate attacks.

Suppression and concealing happen to be two controls put on data things. With suppression, very sensitive data values aren't supplied; the query is certainly rejected without reply. With concealing, the solution provided is near but not the actual value.

These two settings reflect the compare between security and safety and accuracy. With suppression, any benefits provided are proper, yet many reactions should be withheld to keep safety. With concealing, even more results could be provided,

however the precision of the outcomes is lower. The decision between suppression and concealing depends upon the context in the database. Types of suppression and concealing follow.

Limited Response Suppression

The n-item k-percent tip eliminates particular low-frequency factors from being shown. It isn't sufficient to erase them, nevertheless, if their beliefs may also be inferred. To understand why, consider Table 6-11, which ultimately shows counts of college students by dorm and love-making.

Table 6.11. Students by Dorm and Sex.

	Holmes	Grey	West	Total
M	1	3	1	5
F	2	1	3	6
Total	3	4	4	11

The data in such a table claim that the skin cells with counts of just one 1 ought to be suppressed; their matters are as well revealing. Nonetheless it does no very good to curb the MaleHolmes cell when the price 1 could be dependant on subtracting FemaleHolmes (2) from the full total (3) to find out 1, as displayed in Table 6.12.

Table 6.12. Students by Dorm and Sex, with Low Count Suppression.

	Holmes	Grey	West	Total
M	-	3	-	5
F	2	-	3	6
Total	3	4	4	11

When one cell is suppressed in a table with totals for rows and columns, it's important to suppress a minumum of one additional cell around the row and something around the column to supply some confusion. By using this logic, all tissues (except totals) would need to be suppressed on this small sample Table. When totals aren't provided, single tissues within a row or column could be suppressed.

Combined Results

Another control offers rows or columns to safeguard sensitive values. For instance, Table 6.13 reveals several sensitive outcomes that identify individual individuals. (Despite the fact that these counts might not seem sensitive, they could be accustomed to infer sensitive info such as Title; therefore, we take into account them to come to be sensitive.)

Table 6-13. Students by Sex and Drug Use.

Sex	Drug Use			
	0	1	2	3
M	1	1	1	2
F	2	2	2	0

These counts, coupled with other results such as for example sum, enable us to infer personal drug-use values with the three males, in addition to to infer that no feminine was graded 3 for medicine use. To curb such sensitive details, you'll be able to combine the feature principles for 0 and 1, and in addition for just two 2 and 3, providing the less very sensitive results revealed in Table 6.14. In this situation, it is unattainable to recognize any single benefit.

Table 6.14. Suppression by Combining Revealing Values.		
Sex	Drug Use	
	0 or 1	2 or 3
M	2	3
F	4	2

Another method of combining results would be to present principles in ranges. For instance, instead of launching exact school funding figures, results could be released with the amounts $01999, $20003999, and $4000 and over. Even if only 1 record is displayed by a one result, the precise value of this record isn't known. Similarly, the best and lowest school funding values are hidden.

Yet another approach to merging is by rounding. This system is truly a fairly well-known exemplory case of combining by variety. If numbers will be rounded for the nearest a variety of of 10, the helpful ranges happen to be 05, 615, 1625, etc. Actual values are usually rounded upward or right down to the nearest a variety of of some platform.

Random Sample

With random test control, an outcome is not produced from the whole databases; instead the effect is computed on the random sample from the database. The trial chosen is usually large enough for being valid. As the sample isn't the whole databases, a query from this sample won't necessarily match the effect for your database. Thus, due to 5 percentage for a specific query implies that 5 percent of this records chosen for any sample because of this query had the required property. You'll expect that around 5 percentage of the complete database could have the property involved, but the real percentage could be quite different.

In order that averaging problems from repeated, equal queries are avoided, the same test set ought to be chosen for comparative queries. In this manner, all equivalent questions will produce exactly the same consequence, although that final result will be just an approximation for the whole database.

Random Data Perturbation

It is quite often beneficial to perturb the beliefs of the databases by a smaller error. For every x_i this is the true price of data product i inside the database, we are able to generate a little random error term εiand include it to x_i for statistical effects. The ? values happen to be both negative and positive, in order that some reported worth will be marginally greater than their true prices and other claimed values will undoubtedly be lower. Statistical actions such as total and mean will undoubtedly be close however, not necessarily exact. Information perturbation is simpler to utilize than random example selection since it is simpler to store all of the ? values to be able to produce exactly the same result for comparative queries.

Query Analysis

A more complex type of security makes use of query analysis. Below, a query and its own implications are examined to find out whether an outcome should be presented. As noted early on, query analysis could be very difficult. One method involves retaining a query background for each person and judging a query within the framework of what inferences happen to be possible given prior results.

Conclusion within the Inference Problem

You can find no perfect answers to the inference trouble. The methods to controlling it stick to the three pathways listed below. The initial two methods may be used either to control queries accepted or even to limit data presented in reaction to a query. The final method applies and then data released.

- Suppress obviously vulnerable information. This step can be obtained fairly effortlessly. The tendency would be to err privately of suppression, in doing so restricting the effectiveness of the repository.

- Monitor what an individual knows. Although probably leading to the best safe disclosure, this process is extremely high priced. Information should be managed on all consumers, even though the majority are not attempting to obtain sensitive files. Moreover, this process seldom considers what any two different people may know mutually and cannot tackle what a solitary user can attain by using several IDs.

- Disguise the info. Random perturbation and rounding can inhibit statistical strikes that be determined by exact beliefs for rational and algebraic adjustment. The users on the database receive just a bit incorrect or perhaps inconsistent results.

It is improbable that study will reveal a straightforward, easy-to-apply strategy that determines specifically which data could be revealed without reducing sensitive data.

Aggregation

Linked to the inference difficulty is aggregation, this means building sensitive outcomes from less very sensitive inputs. We found earlier that learning either the latitude or longitude of your gold mine does indeed you no fine. But once you learn both latitude and longitude, it is possible to identify the mine. For a far more realistic example, think about how police make use of aggregation usually in fixing crimes: They determine who possessed a motive for committing the criminal offenses, when the offense was dedicated, who acquired alibis covering that point, who had the abilities, etc. Typically, you imagine of police exploration as you start with the entire populace and narrowing the research to an individual. If the police officers do the job in parallel, you can have a summary of probable suspects, another could have an inventory with possible purpose, and another could have a summary of capable persons. Once the intersection of the lists is really a single person, the authorities have their primary suspect.

Responding to the aggregation trouble is difficult since it requires the repository management program to trail which outcomes each user experienced already acquired and conceal any outcome that would allow user derive a far more sensitive final result. Aggregation is particularly tough to counter since it can take spot outside the program. For example, imagine the security coverage is the fact that anyone might have sometimes the latitude or longitude of this mine, however, not both. Nothing stops you from receiving one, your good friend from obtaining the other, and both of you talking to one another.

Recent fascination with data mining features raised concern once again about aggregation. Files mining may be the procedure for sifting through several directories and correlating several data elements to get useful information.

Advertising and marketing companies use info mining extensively to get consumers more likely to buy a product or service.

6.6. Multilevel Databases

So far, we've considered data in mere two classes: either vulnerable or nonsensitive. We've alluded for some data items getting more delicate than others, but we've allowed simply yes-or-no gain access to. Our presentation could have implied that level of sensitivity was a work of the feature, the column where the data came out, although nothing we've done depended with this interpretation of level of sensitivity. Such a unit appears in Table 6.15, where two columns are usually diagnosed (by shading) as hypersensitive. Actually, though, sensitivity is set not only by attribute but additionally in ways that people investigate within the next section.

Table 6.15. Attribute-Level Sensitivity. (Sensitive attributes are shaded.)

Name	Department	Salary	Phone	Performance
Rogers	training	43,800	Apr-67	A2
Jenkins	research	62,900	Jun-81	D4
Poling	training	38,200	Apr-01	B1
Garland	user services	54,600	Jun-00	A4
Hilten	user services	44,500	Apr-51	B1
Davis	administration	51,400	Apr-05	A3

The Case for Differentiated Security

Consider a databases containing information on U.S. federal government expenditures. A number of the expenditures happen to be for paper videos, which is not necessarily sensitive data. Some salary expenses are at the mercy of privacy requirements. Unique salaries are hypersensitive, however the aggregate (for instance, the full total Agriculture Team payroll, which really is a matter of general public record) isn't sensitive. Fees of certain armed service operations tend to be more sensitive; for instance, the quantity america spends for ballistic

missiles, that is not public. You can find even operations regarded only to some individuals, so the amount allocated to these operations, as well as the truth that anything was allocated to such an procedure, is highly vulnerable.

Table 6.15 listings employee information. It could in fact function as circumstance that Davis is really a temporary employee employed for a particular task, and her entire record includes a different level of sensitivity from others. Perhaps the mobile phone found for Garland will be her private collection, unavailable to the general public. We are able to refine the level of sensitivity of the info by depicting it as displayed in Table 6-16.

Table 6-16. Data and Attribute Sensitivity.

Name	Department	Salary	Phone	Performance
Rogers	training	43,800	Apr-67	A2
Jenkins	research	62,900	Jun-81	D4
Poling	training	38,200	Apr-01	B1
Garland	user services	54,600	Jun-00	A4
Hilten	user services	44,500	Apr-51	B1
Davis	administration	51,400	Apr-05	A3

From this information, three attributes of database safety emerge.

- The stability of an individual element could be not the same as the protection of other components of the same document or from additional values of exactly the same attribute. That's, the security of 1 element varies from that of additional elements of exactly the same row or column. This example implies that security and safety should be executed for each personal element.

- Two levelssensitive and nonsensitiveare insufficient to stand for some security circumstances. Several marks of security could be needed. These levels may represent amounts of allowable understanding, which might overlap. Usually, the security marks form a lattice.

The security of your aggregatea total, a count, or perhaps a group of prices inside a databasemay change from the stability of the average person elements. The protection on the aggregate could be higher or less than that of the average person elements.

Granularity

Recall the military classification type applied formerly to paper files and was designed to computers. It really is simple enough to classify and monitor an individual sheet of papers or, for example, a paper document, a computer data file, or a one program or method. It is completely dissimilar to classify individual files items.

For obvious causes, a whole sheet of document is grouped at one degree, even though particular words, such as for example and, the, or of, will be innocuous in virtually any context, along with other words, such as for example codewords like Manhattan task, might be delicate in any framework. But determining the sensitivity of every value in the database is comparable to applying a awareness level to every individual word of your document.

And the thing is still more difficult. The term Manhattan alone is not hypersensitive, nor is job. However, the mix of these words generates the very sensitive codeword Manhattan task. A similar circumstances occurs in directories. Therefore, not merely can every part of a database own a distinct awareness, every mix of elements may also have a definite sensitivity. In addition, the combination could be pretty much sensitive than some of its elements.

Just what exactly would we are in need of to be able to associate a level of sensitivity stage with each benefit of a data source? First, we are in need of an access command coverage to dictate which customers may have usage of what data. Usually, to carry out this coverage each data merchandise is marked showing its access limits. Second, we are in need of a way to guarantee that the worthiness is not evolved by an unauthorized man or woman. These two specifications home address both confidentiality and integrity.

Security Issues

, we launched three general stability issues: integrity, confidentiality, and accessibility. In this part, we extend the initial two of the concepts to add their special assignments for multilevel directories.

Integrity

Even yet in a single-level databases where all elements possess the same amount of sensitivity, integrity is really a tricky problem. Regarding multilevel directories, integrity gets both more essential and more hard to achieve. Due to the *-property or home for access management, an activity that reads high-level info is not permitted to write a document at less level. Put on databases, even so, this principle states a high-level user shouldn't be able to publish a lower-level info element.

The problem with this particular interpretation arises once the DBMS should be able to learn all records inside the database and create new records for just about any of the next purposes: to accomplish backups, to check out the repository to answer questions, to reorganize the databases in accordance with a user's handling needs, or even to update all details of the data source.

When people experience this issue, they deal with it through the use of trust and good sense. Individuals who have access to hypersensitive information are mindful not to express it to uncleared folks. In a processing system, you can find two options: Either the procedure cleared at a higher stage cannot write to less level or the procedure should be a "trusted method," the personal computer equivalent of an individual with a security and safety clearance.

Confidentiality

Users trust a database provides correct information, and therefore the data will be consistent and correct. As indicated before, some method of guarding confidentiality may bring about small adjustments to the info. Although these perturbations shouldn't have an effect on statistical analyses, they could produce two several answers representing exactly the same underlying data price in reaction to two differently created queries. Within the multilevel situation, two different consumers running at two distinct levels of safety could easily get two different responses to exactly the same query. To maintain confidentiality, precision can be sacrificed.

Enforcing confidentiality furthermore results in unknowing redundancy. Assume a personnel professional performs at one degree of access agreement. The specialist recognizes that Bob Hill functions for the business. However, Bob's report does not show up on the old age settlement roster. The professional assumes this omission can be an error and produces an archive for Bob.

The reason why that no report for Bob looks is the fact Bob is really a secret real estate agent, and his work with the business is not said to be public knowledge. An archive on Bob happens to be in the record but, due to his special posture, his record isn't accessible towards the personnel consultant. The DBMS cannot reject the document from the workers specialist because doing this would disclose that there currently is this type of record with a sensitivity too much for the professional to check out. The design of the brand new record implies that nowadays there are two details for Bob Hill: one hypersensitive and one certainly not, as proven in Table 6.17. This example is named polyinstantiation, and therefore one document can appear often, with another degree of confidentiality each and every time.

Table 6-17. Polyinstantiated Records.

Name	Sensitivity	Assignment	Location
Hill, Bob	C	Program Mgr	London
Hill, Bob	TS	Secret Agent	South Bend

This problem can be exacerbated because Bob Hill is really a common enough title that there could be two differing people in the databases with that brand. Thus, merely checking the databases (from the high-sensitivity stage) for duplicate titles is not an effective way to discover records inserted unknowingly by people who have only reduced clearances.

We might furthermore find other causes, unrelated to awareness level, that bring about polyinstantiation. For instance, Mark Thyme proved helpful for Acme Company for 30 ages and retired. He could be now attracting a pension from Acme, therefore he appears like a retiree in a single personnel document. But Mark auto tires of being residence and is also rehired like a part-time service provider; this new function generates another personnel document for Symbol. Each is really a legitimate employment document. Inside our zeal to lessen

polyinstantiation, we should take care not to eliminate legitimate files such as for example these.

6.7. Proposals for Multilevel Security

As it is possible to already tell, applying multilevel safety measures for databases can be difficult, probably way more than in os's, due to the smaller granularity of the things being handled. In the rest of this segment, we study methods to multilevel protection for databases.

Separation

As we have previously seen, separation is essential to limit admittance. In this area, we study systems to implement parting in databases. Subsequently, we observe how these mechanisms can help implement multilevel protection for databases.

Partitioning

The obvious command for multilevel directories is usually partitioning. The data source is split into separate directories, each at its level of awareness. This approach is comparable to maintaining separate data files in separate record cabinets.

This control damages a basic benefit of databases: removal of redundancy and better accuracy through possessing only one industry to update. In addition, it generally does not address the issue of the high-level customer who needs usage of some low-level files coupled with high-level data.

Nevertheless, due to the difficulty of creating, maintaining, and employing multilevel databases, numerous users with information of blended sensitivities deal with their data through the use of separate, isolated directories.

Encryption

If sensitive information will be encrypted, a end user who accidentally will get them cannot interpret the info. Thus, each degree of sensitive data could be saved in a stand encrypted under an integral unique to the amount of level of sensitivity. But encryption provides certain disadvantages.

First, a individual can install a selected plaintext attack. Imagine gathering affiliation of REP or DEM is certainly saved in encrypted kind in each report. A end user who achieves usage of these encrypted grounds can simply decrypt them by developing a new report with get together=DEM and contrasting the ensuing encrypted version compared to that element in all the records. More serious, if authentication files happen to be encrypted, the harmful user can swap the encrypted type of his / her own data for your of any user. Not merely does this deliver access for your malicious user, but it addittionally excludes the authentic individual whose authentication info have been improved to that of this malicious individual. These possibilities will be shown in Figure 6.5 and 6.6.

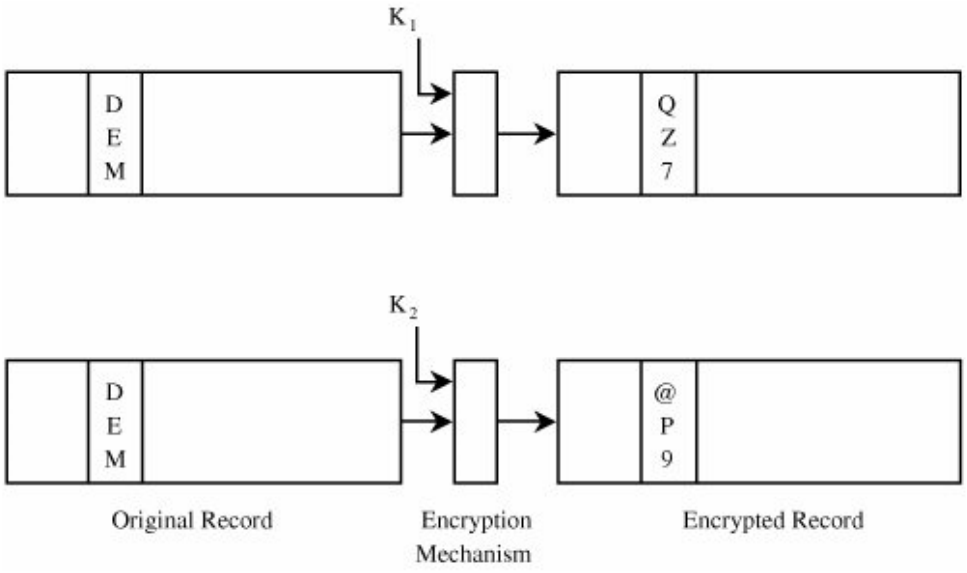


Figure 6.5. Cryptographic Separation: Different Encryption Keys

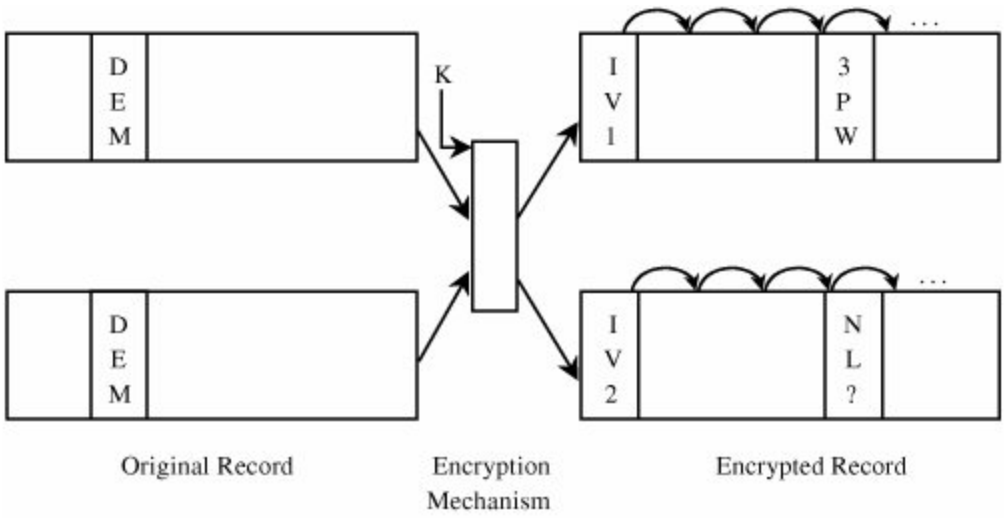


Figure 6.6. Cryptographic Separation: Block Chaining.

Using a several encryption key for every document overcomes these problems. Each record's grounds could be encrypted with another essential, or all grounds of an archive could be cryptographically linked, much like cipher stop chaining.

The disadvantage, next, is that every field should be decrypted when customers perform standard repository operations such as for example "select all files with Income > 10,000." Decrypting the Earnings field, also on rejected files, increases the time and energy to practice a query. (Think about the query that selects just one single record but that has to decrypt and review one field of every record to get the one which satisfies the query.) Consequently, encryption isn't often employed to implement parting in databases.

Integrity Lock

The integrity lock was initially proposed with the U.S. Oxygen Force Summer Research on Data Bottom Protection. The lock is really a way to deliver both integrity and constrained access for your database. The procedure was initially nicknamed "spray coloring" because each factor is figuratively decorated with a coloring that denotes its awareness. The coloring is certainly maintained together with the element, not in a very master database stand.

A model of the essential integrity lock is definitely shown in Number 6-7. As illustrated, each visible data item includes three items: the specific data product itself, a awareness label, and also a checksum. The awareness label identifies the level of sensitivity of the info, along with the checksum is usually computed across both files and sensitivity content label to avoid unauthorized adjustment of the info product or its brand. The actual files item is stashed in plaintext, for proficiency as the DBMS might need to examine many grounds when selecting data to complement a query.

The sensitivity tag should be

- unforgeable, in order that a malicious object cannot develop a new sensitivity levels for a component
- unique, in order that a malicious object cannot replicate a sensitivity stage from another element
- concealed, in order that a malicious object area cannot even ascertain the sensitivity degree of an arbitrary element

The third little bit of the integrity lock to get a field can be an error-detecting code, known as a cryptographic checksum. To ensure that a info benefit or its level of sensitivity classification is not altered, this checksum should be unique for confirmed factor, and must incorporate both element's data price then one to connect that benefit to a specific position within the database. As displayed in Figure 6-8, a proper cryptographic checksum consists of something unique for the record (the report quantity), something one of a kind to this info field in the record (the discipline attribute brand), the worthiness of this component, and the level of sensitivity classification on the aspect. These four pieces protect from anyone's transforming, copying, or going the info. The checksum could be computed with a solid encryption algorithm or hash feature.

IDOL

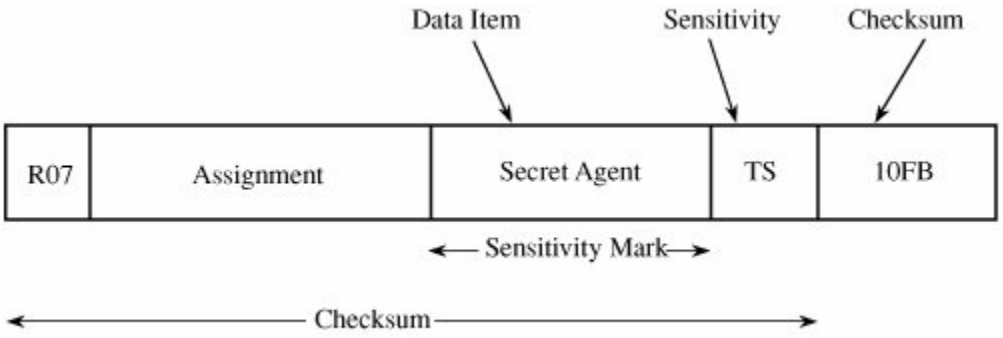


Figure 6.8. Cryptographic Checksum.

Sensitivity Lock

The level of sensitivity lock found in Figure 6.9 was created by Graubert and Kramer to meet up these ideas. A awareness lock is really a combination of a distinctive identifier (like the record variety) plus the sensitivity level. As the identifier is exclusive, each lock pertains to one particular report. Many different factors will have exactly the same sensitivity degree. A malicious subject matter shouldn't be able to discover two elements possessing identical sensitivity quantities or identical files values simply by considering the sensitivity stage part of the lock. Due to the encryption, the lock's items, especially the level of sensitivity level, are hidden from plain see. So, the lock can be connected with one specific document, and it helps to protect the secrecy on the sensitivity degree of that record

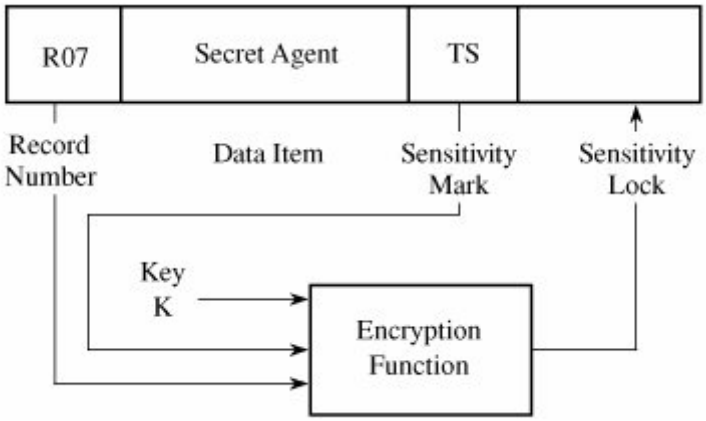


Figure 6.9. Sensitivity Lock.

Styles of Multilevel Secure Databases

This section protects different models for multilevel risk-free databases. These models show the tradeoffs among efficiency, flexibility, simplicity, and trustworthiness.

Integrity Lock

The integrity lock DBMS had been invented like a short-term treatment for the security difficulty for multilevel directories. The intention was initially in order to utilize any (untrusted) data source manager with a reliable procedure that deals with access management. The sensitive information had been obliterated or hidden with encryption that shielded both a files item and its own sensitivity. In this manner, only the gain access to procedure would have to be respected because only it might be able to attain or grant usage of sensitive information. The design of this type of system is proven in Figure 6-10.

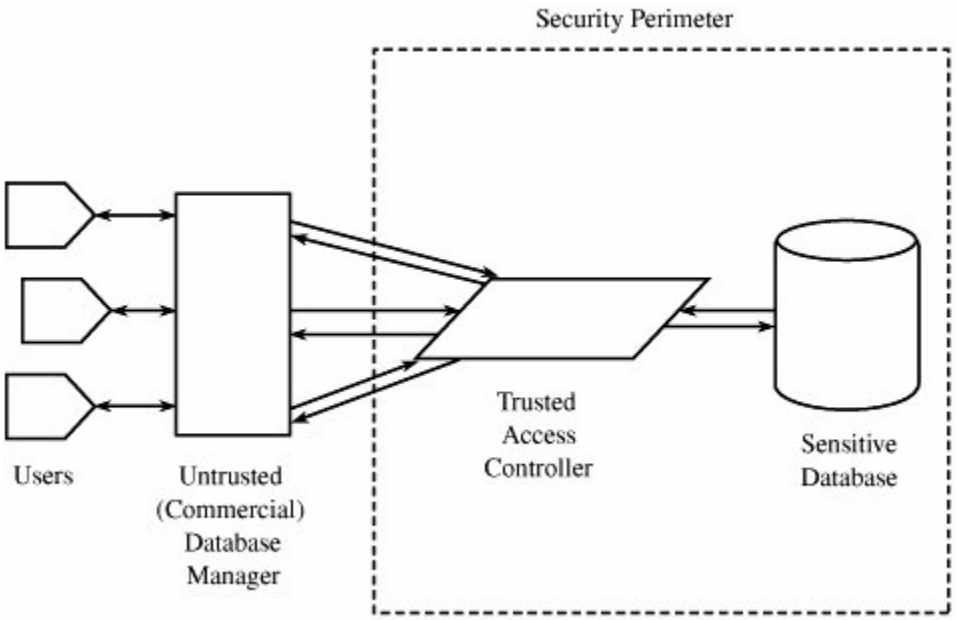


Figure 6.10. Trusted Database Manager.

The performance of integrity hair is a considerable drawback. The area needed for saving an element should be expanded to support the sensitivity content label. Because there are many pieces inside the label and something label for each element, the area required is considerable.

Problematic, too, may be the processing time productivity associated with an integrity lock. The awareness label should be decoded whenever a data element is usually passed to an individual to verify which the user's access will be allowable. Also, whenever a value is authored or customized, the label should be recomputed. Thus, considerable processing time will be consumed. In the event the database file could be sufficiently protected, the info values of the average person elements could be still left in plaintext. That technique benefits go for and project inquiries across sensitive career fields because a component need not end up being decrypted merely to determine whether it ought to be selected.

A final difficulty with this particular approach is usually that the untrusted database supervisor sees all info, so it's at the mercy of Trojan horse problems by which information could be leaked through covert stations.

Trusted Entrance End

The style of a reliable front-end process will be shown in Figure 6-11. A reliable front end can be referred to as a officer and operates similar to the reference monitor. This process, originated by Hinke and Schaefer, identifies that lots of DBMSs have already been built and placed into use without thought of multilevel security and safety. Staff members seem to be trained in employing these DBMSs, plus they may actually use them often. The front-end idea takes benefit of existing resources and expertise, improving the security of the existing systems with reduced change to the machine. The conversation between a customer, a trusted forward end, including a DBMS involves the next steps.

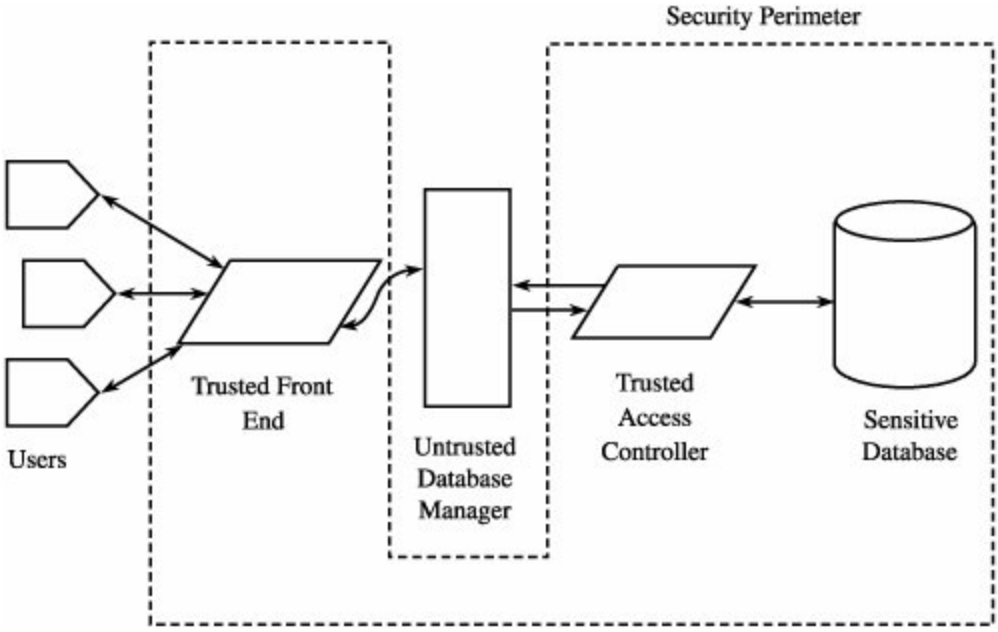


Figure 6-11. Trusted Front End.

1. A user recognizes himself or herself to leading end; leading conclusion authenticates the user's personal information.

2. An individual concerns a query to leading end.

3. front end verifies the user's authorization to information.

4. front endconcerns a query for the database manager.

5. The database supervisor performs I/O admittance, getting together with low-level access handle to achieve usage of actual data.

6. The database supervisor returns the consequence of the query towards the trusted front ending.

7. front endanalyzes the level of sensitivity levels of the info items in the effect and selects those things in keeping with the user's protection level.

8. front endtransmits decided on data towards the untrusted front finish for formatting.

9. The untrusted front end ending transmits formatted info to an individual.

The trusted front side end assists as a one-way filtration system, screening out benefits the user shouldn't be able to entry. But the plan is certainly inefficient because possibly much data is definitely retrieved and discarded as incorrect for an individual.

Commutative Filters

The idea of a commutative filtration system was suggested by Denning as being a simplification on the trusted interface towards the DBMS. Basically, the filter monitors the user's need, reformatting it if required, so that sole data of a proper sensitivity level will be returned to an individual.

A commutative filter is really a process that sorts an interface between your user along with a DBMS. Nevertheless, unlike the respected front conclusion, the filter attempts to capitalize around the efficiency of all DBMSs. The filtration system reformats the query so the database manager does indeed just as much of the task as possible, screening process out many undesirable records. The filtration system then offers a second screening to choose only files to that your user has entry.

Filters may be used for security with the record, feature, or element levels.

- When used on the record levels, the filter demands desired files plus cryptographic checksum info; after that it verifies the correctness and availability of data to get passed to an individual.

- At the feature level, the filtration bank checks whether all features inside the user's query will be accessible to an individual and, if that's the case, goes by the query for the database office manager. On go back, it deletes all job areas to that your user does not have any access rights.

- At the factor level, the machine requests desired information plus cryptographic checksum details. When they are returned, it bank checks the classification degree of every component of every report retrieved contrary to the user's level.

Suppose several physicists in Washington performs on very very sensitive projects, therefore the current user shouldn't be allowed to obtain the physicists'

titles in the data source. This restriction offers a problem with this particular query:

retrieve NAME where ((OCCUP=PHYSICIST) Λ (CITY=WASHDC))

Suppose, also, that the existing user is definitely prohibited from understanding anything about any individuals in Moscow. Utilizing a typical DBMS, the query might obtain all records, along with the DBMS would after that pass the outcomes to the user. However, once we have seen, an individual could probably infer reasons for having Moscow personnel or Washington physicists focusing on secret jobs without even being able to access those fields immediately.

The commutative filtration re-forms the initial query in the trustable way in order that sensitive information will be never extracted from database. Our trial query would become

retrieve NAME where ((OCCUP=PHYSICIST) Λ (CITY=WASHDC))

from all records R where

(NAME-SECRECY-LEVEL (R) ≤ USER-SECRECY-LEVEL) Λ

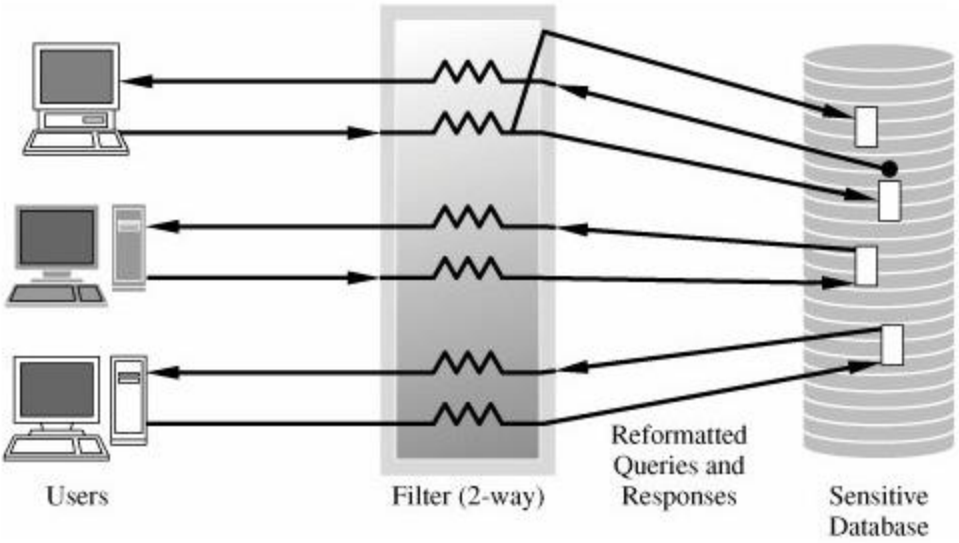
(OCCUP-SECRECY-LEVEL (R) ≤ USER-SECRECY-LEVEL) Λ

(CITY-SECRECY-LEVEL (R) ≤ USER-SECRECY-LEVEL))

The filter functions by restricting the query for the DBMS and restricting the outcomes before they're returned to an individual. In this situation, the filtration would request Label, NAME-SECRECY-LEVEL, OCCUP, OCCUP-SECRECY-LEVEL, Metropolis, and CITY-SECRECY-LEVEL worth and would then simply filter and go back to the user just those job areas and items which are of an secrecy degree acceptable for an individual.

A good example of this query filtering functioning is found in Figure 6.12. The benefit of the commutative filtration system is usually that it permits query variety, some optimization, plus some subquery controlling to be achieved from the DBMS. This delegation of tasks keeps how big is the security filtration system small, decreases redundancy between it as well as the DBMS, and boosts the overall effectiveness of the machine.

IDOL



Distributed Databases

The distributed or federated databaseis a 4th design for a risk-free multilevel database. In cases like this, a trusted prominent end controls usage of two unmodified industrial DBMSs: one for several low-sensitivity data and something for several high-sensitivity data.

The front stop requires a user's query and formulates single-level inquiries to the directories as appropriate. For any end user cleared for high-sensitivity info, the front finish submits inquiries to both substantial- and low-sensitivity directories. If the user isn't cleared for high-sensitivity files, the front ending submits a query to simply the low-sensitivity repository. If the effect is from either back-end data source alone, leading end passes the effect back to an individual. If the effect originates from both databases, leading end must combine the outcomes appropriately. For instance, when the query is really a join query getting some high-sensitivity words and some very low, the front ending has to do the same as a database sign up for itself.

The distributed databases design isn't popular as the front stop, which should be trusted, is sophisticated, potentially including a lot of the functionality of a complete DBMS itself. Furthermore, the design will not scale well to numerous degrees of level of sensitivity; each sensitivity degree of data should be maintained in its separate database.

Window/View

Traditionally, among the advantages of utilizing a DBMS for several users of several interests (however, not necessarily different level of sensitivity levels) may be the ability to develop a different view for every user. That's, each user is fixed to an image of the info reflecting just what an individual needs to notice. For instance, the registrar could see only the school assignments and levels of each pupil at a university or college, not having to see extracurricular exercises or medical files. The university wellbeing clinic, alternatively, needs medical details and drug-use info but not ratings on standardized academics tests.

The idea of a window or perhaps a view may also be an organizing rule for multilevel databases access. A windows is really a subset of an database, containing the information a user is eligible for admittance. Denning research the introduction of sights for multilevel data source security.

A view can signify an individual user's subset repository so that most of a user's concerns access simply that data source. This subset ensures that an individual does not admittance values beyond your permitted kinds, because nonpermitted ideals are not even yet in the user's databases. The view is definitely specified as a couple of relations within the database, therefore the data inside the view subset modification as data adjustment in the databases.

For instance, a travel realtor might have usage of section of an airline's airline flight information database. Documents for cargo plane tickets will be excluded, just as would the pilot's title plus the serial amount of the plane for each and every flight. Imagine the database included an attribute Kind whose value was initially either CARGO or Move (for traveler). Other capabilities may be flight number, origins, destination, departure moment, arrival time, capability, pilot, and tail quantity.

Now imagine the airline made some passenger plane tickets with lower fares that may be booked only straight through the air travel. The flight might assign their airfare numbers a far more sensitive rating to create these plane tickets unavailable to visit agents. The complete database, as well as the agent's view, may have the logical framework shown in Table 6-18.

IDOL

Table 6-18. Airline Database.

(a) Airline's View.

FLT#	ORIG	DEST	DEP	ARR	CAP	TYPE	PILOT	TAIL
362	JFK	BWI	830	950	114	PASS	Dosser	2463
397	JFK	ORD	830	1020	114	PASS	Bottoms	3621
202	IAD	LGW	1530	710	183	PASS	Jevins	2007
749	LGA	ATL	947	1120	0	CARGO	Witt	3116
286	STA	SFO	1020	1150	117	PASS	Gross	4026
...								
...								

(b) Travel Agent's View.

		FLT	ORIG	DEST	DEP	ARR	CAP	
		362	JFK	BWI	830	950	114	
		397	JFK	ORD	830	1020	114	
		202	IAD	LGW	1530	710	183	
		286	STA	SFO	1020	1150	117	
		...						
		...						

The travel agent's view of the database is expressed as

```
view AGENT-INFO
    FLTNO:=MASTER.FLTNO
    ORIG:=MASTER.ORIG
    DEST:=MASTER.DEST
    DEP:=MASTER.DEP
    ARR:=MASTER.ARR
    CAP:=MASTER.CAP
        where MASTER.TYPE='PASS'
    class AGENT
    auth retrieve
```

Because the entry class of the view is Realtor, more sensitive airfare numbers (plane tickets booked only throughout the airline) usually do not come in this view. Otherwise, we could have got eliminated the complete records for all those plane tickets by restricting the report selection which has a where clause. A check

out may include computation or organic selection standards to designate subset data.

The data shown to a individual is attained by filtering of this contents of the initial database. Attributes, documents, and elements are usually stripped away so the user sees simply acceptable products. Any feature (column) is usually withheld unless an individual is authorized to gain access to a minumum of one element. Any document (row) is definitely withheld unless an individual is authorized to gain access to a minumum of one element. Then, for several elements that even now remain, if an individual is not certified to gain access to the element, it really is substituted by UNDEFINED. This previous step will not compromise any info because the customer knows the life of the feature (there's a minumum of one element that an individual can obtain) and an individual knows the presence of the report (again, a minumum of one accessible element is accessible in the document).

Along with elements, a check out includes relationships on attributes. Additionally, a user can make new relationships from fresh and existing capabilities and components. These new relationships are attainable to other customers, subject to the typical access privileges. A person can are powered by the subset repository defined in the view only just as allowed from the operations authorized inside the view. For example, a user may be allowed to get records specified in a single view or even to retrieve and upgrade records as given in another watch. For example, the airline inside our example may limit travel companies to retrieving information.

The Sea View project referred to in may be the basis for something that integrates a reliable operating system to create a trusted repository manager. The split implementation as explained is proven in Figure 6.13. The lowest layer, the research monitor, performs data file connections, enforcing the BellLa Padula gain access to controls, and does indeed user authentication. Section of its function would be to filter data handed down to higher ranges. The second degree performs standard indexing and computation features of the databases. The third levels translates views in to the base relations of this repository. These three levels constitute the trusted processing bottom part (TCB) of the machine. The

remaining levels implement ordinary DBMS capabilities and an individual interface.

This layered technique makes landscapes both a reasonable section of a data source and an operating one. The tactic is an crucial step toward the look and implementation of the trustable database control system.

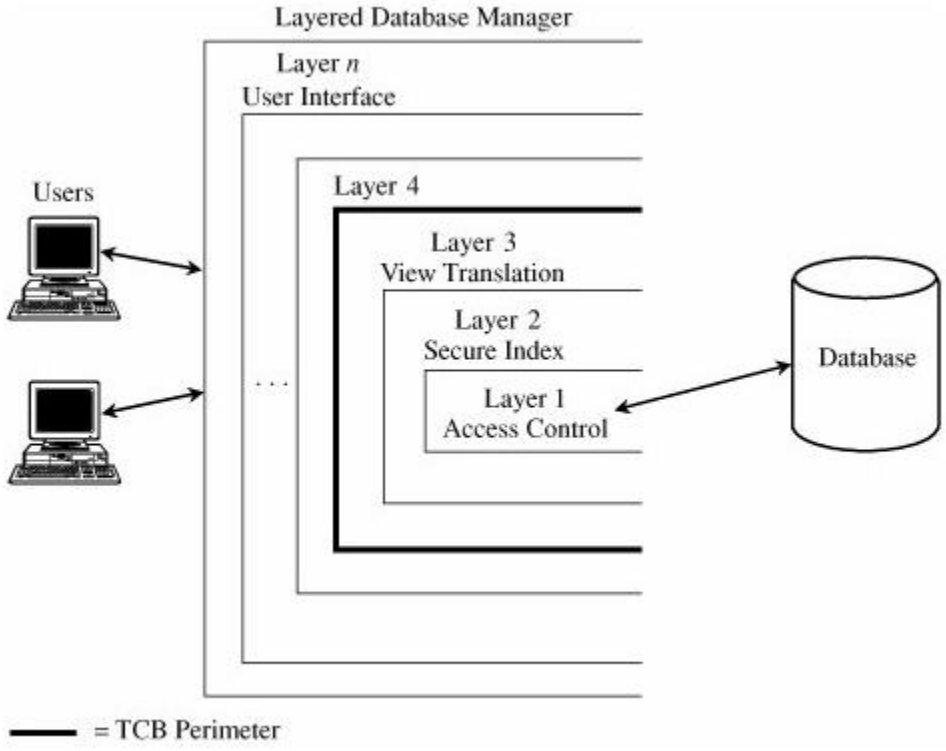


Figure 6-13. Secure Database Decomposition.

Practical Issues

The multilevel security and safety problem for directories has been examined because the 1970s. Several encouraging research results have already been identified, once we have seen on this chapter. However, much like trusted os's, the consumer requirement is not sufficient to aid many items. Civilian users haven't liked the inflexibility with the military multilevel protection model, and there were too few armed forces users. Therefore, multilevel secure directories are principally of analysis and historical attention.

The general ideas of multilevel directories are essential. We do have to be able to distinguish data in accordance with their amount of sensitivity. Similarly, we are in

need of ways of merging data of various sensitivities into one repository (or at the very least into one exclusive repository or federation of directories). And these desires will only boost as time passes as larger directories contain more delicate information, specifically for privacy concerns.

6.8. Data Mining

Databases are excellent repositories of information. More data are increasingly being collected and rescued (partly as the expense per megabyte of storage space has dropped from dollars a couple of years before to fractions of cents nowadays). Systems and the web allow spreading of directories by persons and with techniques earlier unimagined. But to get needles of details in those huge areas of haystacks of info requires wise analyzing and querying of the info. Indeed, a complete specialization, called information mining, has surfaced. In a typically automated way, files mining applications kind and look for thorough data.

Data mining utilizes statistics, machine mastering, mathematical models, style recognition, along with other techniques to find out patterns and relationships on large datasets.Information mining tools work with association (one celebration often complements another), sequences (one celebration often results in another), classification (events show patterns, for instance coincidence), clustering (some things have similar features), and forecasting (past situations foretell future kinds). The difference between a databases and a information mining application is now blurred; it is possible to probably observe how you could employ these approaches in ordinary databases queries. Generally, repository queries are regular, whereas files mining is considerably more automatic. You can develop a data source query to find out what other items are acquired by individuals who buy digital camera models and you also might recognize a preponderance of MP3 members in the effect, but you would need to observe that marriage yourself. Files mining equipment would provide the significant associations, not only between surveillance cameras and MP3 participants, but additionally among bagels, flight tickets, and jogging shoes (if this type of relationship been around). Humans need to evaluate these correlations and know what is significant.

IDOL

Data mining gifts probable interactions, but they are definitely not cause-and-effect relationships. Assume you analyzed info and located a relationship between purchase of ice ointment cones and passing away by drowning. You'll not necessarily conclude that advertising ice ointment cones will cause drowning (nor the converse). This differentiation shows why people must be involved with files mining to interpret the productivity: Only human beings can discern that additional variables are participating (for instance, season or spots where cones can be purchased).

Computer security increases from information mining. Files mining is trusted to analyze program data, for instance, audit logs, to recognize patterns linked to attacks. Locating the precursors with an attack might help develop good protection tools and tactics, and seeing what connected with an attack might help pinpoint vulnerabilities to regulate and damage which could have took place. (Among the early works of this type is certainly, and complete conferences have already been specialized in this essential and maturing subject matter.)

In this part, however, you want to examine security difficulties involving information mining. Our now-familiar triad of confidentiality, integrity, and availableness gives us hints from what these security concerns are. Confidentiality considerations start with personal privacy but also incorporate amazing and commercially very sensitive data and safeguarding the worthiness of intellectual property or home: Just how do we control what's disclosed or produced? For integrity the top issue is definitely correctnessincorrect data are usually both worthless and potentially harmful, but we have to investigate how exactly to gauge and be sure correctness. The accessibility consideration pertains to both functionality and composition: Combining directories not originally made to be combined impacts whether results can be acquired regularly as well as at all.

Level of privacy and Sensitivity

Because the target of info mining is overview results, not particular person data items, you'll not expect an issue with level of sensitivity of individual files items. Unfortunately that's not true.

Individual level of privacy can have problems with the same forms of inference and aggregation concerns we researched for directories. Because privacy,

particularly protecting just what a person considers personal information, is an crucial topic that pertains to many regions of computer security,

Not only unique privacy is damaged, however: Relationship by aggregation and inference make a difference companies, businesses, and governments, also. Take, for instance, a problem including Firestone tires as well as the Ford Explorer motor vehicle. IN-MAY 2000, the U.S. Country wide Highway Traffic Security Administration (NHTSA) observed a high occurrence of tire disappointment on Ford Explorers installed with Firestone auto tires. In certain problems the Firestone car tire tread separated; using circumstances the Ford Explorer tipped over, so when the tread segregated, the Ford was initially more prone to hint over. Buyers experienced complained to both Ford and Firestone since soon after the wheel and vehicle collaboration was positioned on the marketplace in 1990, but troubles began to occur after a style modification in 1995. Both corporations had some proof the problem, however the NHTSA overview of combined data much better showed the relationship. Maintaining information on goods' quality is really a standard management training. But the level of sensitivity of info in these directories would preclude many sharing. Even though a trustworthy natural party could possibly be identified to mine the info, the owners will be reasonably worried about what may be revealed. A lot of failures of 1 product could demonstrate a potential marketplace weakness, or perhaps a series of smaller amounts of information could reveal check marketing actions to outsiders.

information about an entity (an individual, company, organization, administration body) may possibly not be under that entity's handle. Supermarkets collect merchandise data using their company purchasers, either from individual visits or, even more usefully, across all acquisitions for a person who runs on the "customer commitment" cards. In aggregate the info show marketing benefits beneficial to the manufacturers, marketing agencies, health analysts, government food firms, financial institutions, experts, among others. But these outcomes were collected with the supermarket that may now carry out anything with the outcomes, including sell these to manufacturers' competitors, for instance.

There's been little research carried out on, or account directed at, the awareness of data from files mining. Clifton offers investigated the issue and proposed solutions that would develop close however, not exact aggregate effects that could preclude revealing hypersensitive information.

Data Correctness and Integrity

"Connecting the dots" is really a phrase currently in fashion: It identifies attracting conclusions from connections between discrete items of data. However before we can link dots, we have to do two some other considerations: gather and appropriate them. Data storage space and computer systems is to be able to collect additional dots than previously. If a name or deal with has ever came out incorrectly on the mailing list, you understand that not absolutely all collected dots happen to be accurate.

Correcting Flaws in Data

Let's have the email list for example. Your neighbor at 510 Thames Block introduced a catalog for cooking area supplies for you at 519 Thames Road with your title but handle 510 rather than 519; clearly an individual made a blunder entering your street address. You contact your kitchen supply place, and they're pleased to adjust your address on the records, since it is within their interest to mail catalogs to individuals who are thinking about them. However they bought your title and address alongside others from the mailing list, plus they have no motivation to get hold of the list operator to improve your master report. So more catalogs continue steadily to show up together with your neighbor. You can view where this report leadsmistaken addresses by no means die.

Files mining exacerbates this example. Databases need exclusive keys to greatly help with design and lookups. But different directories may not have got shared keys, so that they use some information field as though it were an integral. In our instance case, this propagated data field may be the address, hence right now your neighbor's deal with is connected with cooking (even though your neighbor requires a recipe to create tea). Luckily, this example is certainly of little result.

Consider terrorists, on the other hand. A government's cleverness service collects files on suspicious pursuits. But the labels of suspicious folks are foreign, prepared in another alphabet. When altered in to the government's alphabet, the change is

abnormal: One broker creates "Doe," another "Do," and another "Dho." Attempting to use these brands as common tips is challenging at greatest. One approach is certainly phonetic. You cluster conditions that sound related. In cases like this, however, you may generate "Jo," "Cho," "Toe," and "Tsiao," as well, in so doing implicating innocent individuals inside the terrorist lookup.Supposing a real human analyst could effectively separate each one of these and wished to appropriate the Doe/Carry out/Doh databases, you may still find two problems. Initially, the analyst might possibly not have access to the initial databases kept by other firms. Even though the analyst could easily get for the originals, the analyst may possibly never know where else these primary databases had recently been copied.

One important aim of databases would be to have an archive in one location in order that one correction acts all makes use of. With Data mining, an outcome can be an aggregate from numerous databases. There is absolutely no natural solution to function backward from the effect for the amalgamated databases to get and correct problems.

Making use of Comparable Data

Data semantics will be another important account when mining for info. Consider two physical databases with info on family revenue. Except one data source has cash flow by money, and another has the files in thousands. Even though the field labels are the very same, combining the uncooked data would bring about badly distorted studies. Consider another feature rated excessive/medium/low in a single repository and on a numerical size of just one 1 to 5 in another. Should higher/medium/low be handled as 1/3/5? Even though analysts apply that transformation, processing with some 3-stage plus some 5-point precision minimizes the grade of the outcomes. Or how will you meaningfully incorporate one database which has a particular feature with another that will not?

Eradicating False Matches

As we explained earlier, coincidence isn't relationship or causation; because a couple of things occur together will not mean either will cause the other. Info mining attempts to point out nonobvious associations in info, but information

mining applications normally use fuzzy reasoning to get these contacts. These approaches will create both fake positives (phony complements) and skipped connections (fake negatives). We have to be sensitive for the natural inaccuracy of info mining techniques and protect from putting an excessive amount of rely upon the output of the data mining use because "the computer explained so."

Correctness of outcomes and proper interpretation of these results are key security problems for info mining.

Option of Data

Interoperability among unique databases is really a third security problem for data mining. Once we just described, databases must have suitable framework and semantics to create data mining achievable. Missing or matchless data could make data mining outcomes incorrect, so possibly a better substitute is not to make a outcome. But no effect is not exactly like due to no correlation. Much like single databases, information mining programs must cope with multiple sensitivities. Attempting to combine databases with an attribute with an increase of sensitive values can result in no data and therefore no matches.

6.9 Reivew Question

1. A database transaction implements the command "set STATUS to 'CURRENT' in all records where BALANCE-OWED = 0."

   a. Describe how that transaction would be performed with the two-step commit described in this chapter.
   b. Suppose the relations from which that command was formed are (CUSTOMER-ID,STATUS) and (CUSTOMER-ID,BALANCE-OWED). How would the transaction be performed?

Suppose the relations from which that command was formed are (CUSTOMER-ID,STATUS), (CREDIT-ID,CUSTOMER-ID), (CREDIT-ID, BALANCE-OWED). How would the transaction be performed?

2. Can a database contain two identical records without a negative effect on the integrity of the database? Why or why not?

3. Explain the disadvantages of partitioning as a means of implementing multilevel security for databases.

A database management system is implemented under an operating system trusted to provide multilevel separation of users.

a. What security features of the operating system can be used to simplify the design of the database management system?

4. Suppose the operating system has rating r, where r is C2 or B1 or B3, and so on. State and defend a policy for the degree of trust in the database management system, based on the trust of the operating system.
5. What is the purpose of encryption in a multilevel secure database management system?

6.10 References

1. Security in Computing, Fourth Edition By Charles P. Pfleeger - Pfleeger Consulting Group, Shari Lawrence Pfleeger - RAND Corporation Publisher: Prentice Hall

2. Cryptography and Network Security - Principles and Practice fifth edition Stallings William Publisher: Pearson

3. Cryptography And Network Security 3rd Edition behrouz a forouzan and debdeep mukhopadhyay 3/E Publisher: McGraw Hill Education

4. Cryptography and Network Security, 3e Atul Kahate Publisher: McGraw Hill

IDOL

Chapter 7. Security in Networks

IDOL

7.0 Introducton

Networks their design, expansion, and usage are significant to our design of computing. We connect to networks daily, whenever we perform banking transaction, make calls, or drive trains and planes. The electricity companies use systems to track electric power or water utilization and bill for this. When we purchase groceries or gas, networks permit our credit card or debit card deals and billing. Living without networks will be considerably less practical, and many things to do would be difficult. Not surprisingly, after that, computing networks will be attackers' targets of preference. For their actual and possible impact, network problems attract the eye of journalists, supervisors, auditors, and everyone. For example, once you read the everyday newspapers, it's likely you'll find a storyline in regards to a network-based attack at the very least on a monthly basis. The insurance policy coverage itself evokes a feeling of evil, employing terms such as for example hijacking, spread denial of service, and our common friendstrojans, worms, and Trojan horses. Because any large-scale episode will probably put a large number of computing systems at an increased risk, with potential deficits well in to the huge amount of money, network attacks create good copy.

In this section we describe why a network is much like and various from a credit card application system or an operating-system that you've studied in previous chapters. In looking into networks, become familiar with how the principles of confidentiality, integrity, and accessibility apply in networked adjustments. At exactly the same time, so as to the essential notions of id and authentication, accessibility command, accountability, and guarantee are the schedule for network stability, just as they are in other adjustments.

Networking keeps growing and changing maybe even faster than various other computing disciplines. Subsequently, this chapter is definitely unlikely to provide you with current technology, the most recent attack, or the most recent defense mechanism; it is possible to find out about those in everyday newspapers with web sites. However the novelty and alter build on which we know right now: the essential concepts, dangers, and settings for systems. By developing a knowledge of the fundamentals, you can soak up the most existing news efficiently.

Moreover, your understanding can help you in building, safeguarding, and applying networks

7.1. Network Concepts

Networks will be both delicate and tough. To understand why, take into account the power, cable, telephone, or drinking water network that functions your home. In case a dropping tree branch breaks or cracks the power line to your house, you're without electric power until that line gets repaired; you're vulnerable to what's called an individual point of failure, because one cut to the system destroys electrical features for your whole home. Similarly, there could be one phone trunk range or normal water main that provides your home and the ones nearby; failing can keep your building, road, or area without assistance. But we've ways to keep carefully the entire community from declining. If we track back from the network out of your home to the foundation of what moves through it, we have been likely to note that several main syndication lines support a whole town or campus. That's, there is several way to find from the foundation to town, enabling technical engineers to redirect the circulation along alternative pathways. Redundancy helps it be uncommon for a whole city to reduce service from the single failure. Because of this, we point out that this type of network possesses resilience or mistake tolerance.

Complex routing algorithms reroute the move not only around failures but additionally around overloaded sections. The routing is normally done instantly; the control system is frequently supplemented by human being supervision or treatment. Various kinds of networks have high reliability by design and style, not unintentionally. But because there typically is much less redundancy near a network's endpoints than somewhere else, we state that the system has great power in the centreand fragility in the perimeter.

Through the user's viewpoint, a network may also be designed such that it appears like two endpoints with an individual connection in the centre. For instance, the municipal drinking water supply can happen to be bit more than a tank (the foundation), the pipes (the transmitting or communication method), as

well as your water sink (the location). Although this simplistic look at is functionally right, it ignores the sophisticated design, execution, and management on the "pipes." Similarly, we describe computer system networks in this particular chapter with techniques that concentrate on the security aspects but found the systems themselves in a very simplistic approach, to identify the purpose of security and stop the complexity on the systems from distracting our focus. Please take into account that our network information tend to be abstractions of a far more complex actuality.

The Network

Figure 7.1 indicates a system in its simplest type, as two products linked across some moderate by components and software program that allow the communication. In some instances, one device is really a computer (in some cases known as a "server") and another is really a simpler machine (sometimes known as a "client") empowered just with some method of input (like a keyboard) plus some means of productivity (like a screen). For instance, a powerful laptop could be a server, but a handheld individual digital helper (PDA) or perhaps a cell phone may be a network customer. Actually, because more buyer devices have become network-enabled, network security and safety issues will continue steadily to grow.

Figure 7.1. Simple View of Network.

In spite of the fact that this model characterizes an essential system, the genuine circumstance is as often as possible fundamentally progressively entangled.

The simpler client device, utilized for client to-PC communication, is frequently a PC or workstation, so the end user has extensive storage capacity and processing ability.

A network can be designed as only a solitary end user associated with a single server. In any case, more regularly, numerous end users collaborate with numerous servers.

The services of networks are regularly given by numerous PCs. As a solitary client's clients go forward and backward from customer to server, it might just go through certain PCs yet stop at others for critical interactions.

The end client is typically ignorant of huge numbers of the communications and computations occurring in the system for the client's benefit.

A single computing technique in a community is often referred to as a node, and its own processor (pc) is named a host. A link between two hosts is actually a link. Network processing consists of consumers, communications media, obvious hosts, and devices not generally noticeable to customers. In Figure 7-2, Techniques 1 through 4 are usually nodes. Inside our figure the customers are in the lettered consumer machines, perhaps getting together with Server F.

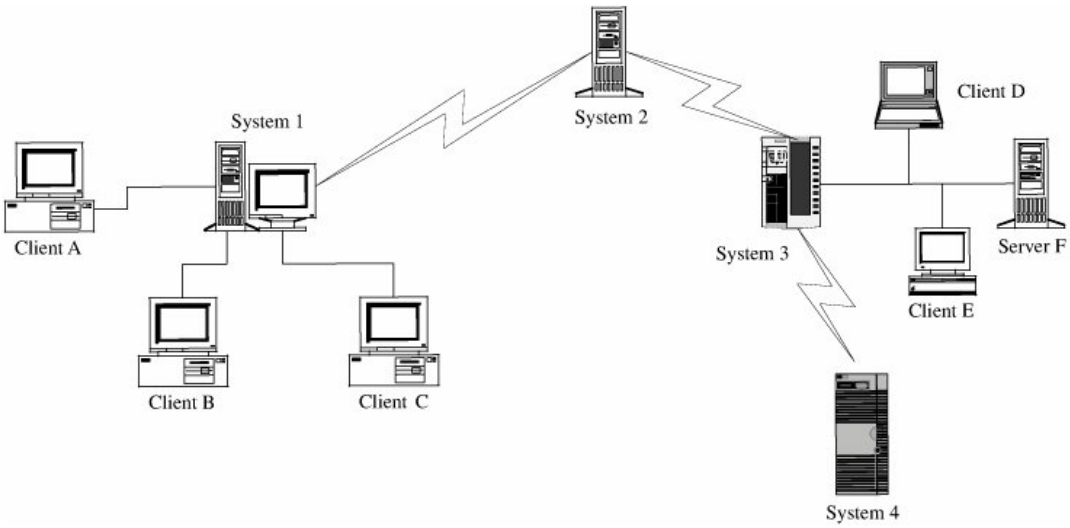


Figure 7-2. More Complex but More Typical View of Networks.

Users talk to networked devices by interacting straight with terminals, workstations, and personal computers. A workstation can be an end-user computing gadget, usually created for a single end user at the same time. Workstations frequently have strong processors and good-sized ram and storage in order to do sophisticated info manipulation (such as for example converting coded info to a visual format and showing the photo). Something is a assortment of processors, perhaps adding an assortment of workstations and self-employed processors, typically with an increase of processing power and much more storage capacity when compared to a workstation.

Networks could be described by more than a few typical attributes:

- Anonymity. You might have seen the animation image that presents your dog typing in a workstation, and declaring to another puppy, "On the net, nobody recognizes you're your dog." A community removes a lot of the clues, such as for example appearance, tone of voice, or context, where we acknowledge acquaintances.

- Automation. In a few sites, one or both endpoints, in addition to all intermediate tips, involved in confirmed communication could be machines with just minimal human guidance.

- Distance. Many systems connect endpoints which are physically far aside. But not all network relationships involve yardage, the swiftness of communication is certainly fast plenty of that humans normally cannot inform whether a remote control site is next to or far.

- Opaqueness. As the dimension of yardage is disguised ., users cannot tell whether a remote host is at the room nearby or in another country. Just as, users cannot differentiate whether they are usually linked to a node within an office, school, residence, or warehouse, or if the node's computing method will be large or little, modest or effective. In fact, consumers cannot explain to if the existing communication involves exactly the same host with that they communicated the final time.

Routing diversity. To keep up or improve dependability and overall performance, routings between two endpoints are often dynamic. That's, the same connection

may carry out one path from the network the very first time and an extremely different path the next time. Actually, a query might take a different journey from the reaction that follows a couple of seconds later.

Size and shape

Just how a network is usually configured, with regards to nodes and contacts, is named the community topology. It is possible to think about the topology because the form of the network

Both of these extremes highlight three proportions of networks which have particular bearing over a network's security.

- Boundary. The boundary distinguishes some the community from a component outside it. For a straightforward network, we are able to easily list all of the components and get an imaginary lines around it to split up what is within the network from what's outside. But list all of the hosts linked to the Internet is usually practically impossible. For instance, a line bordering the Internet would need to surround the complete globe right now, and Online connections also go through satellites in orbit round the earth. Additionally, as folks and organizations prefer to get connected or definitely not, the quantity and kind of hosts change nearly second by 2nd, with the quantity generally increasing as time passes.

- Ownership. It is difficult to learn who has each host within a network. The community administrator's firm may have the network system, including the wire and network units. However, selected hosts could be linked to a community for convenience, definitely not implying ownership.

Control. Lastly, if ownership can be uncertain, control should be, too. To observe how, decide on an arbitrary sponsor. Is it section of community A? If yes, could it be under the management of community A's administrator? Does indeed that administrator create access control guidelines for the system, or determine when its application must be improved also to what version? In fact, does indeed the administrator even understand what variant of program that host goes?

Setting of Communication

A computer community implements conversation between two endpoints. Information will be communicated either in electronic format (where data items happen to be portrayed as discrete binary ideals) or analog (where data items are usually expressed as things in a continuing range, utilizing a medium like noise or electric powered voltage). Computers commonly store and practice digital data, however, many telephone and very similar cable communications come in analog type (because telephones have been originally made to transmit speech). Once the transmission medium needs to shift analog files, the digital indicators must be changed into analog for transmitting and then back again to electronic for computation in the receiving conclusion. Some largely analog networks could even have some electronic segments, therefore the analog signals are usually digitized more often than once. These conversions will be performed by way of a modem (the word comes from modulator-demodulator), which turns a digital information stream to shades and again.

Media

Communication is empowered by several forms of media. We are able to choose among various types, such as for example along copper wire connections or optical fiber content or throughout the air, much like cellular phones. Why don't we look at each kind in turn?

Cable

Because a lot of our computer interaction has historically long been done over mobile phone lines, the most frequent network communication method today is cable. Inside our properties and office buildings, we work with a couple of insulated copper wiring, known as a twisted couple or unshielded twisted match (UTP). Copper has got good transmission attributes at a comparatively low priced. The bandwidth of UTP is bound to under 10 megabits per 2nd (Mbps), thus engineers cannot transfer a lot of communications simultaneously about the same line. Additionally, the signal power degrades since it travels throughout the copper wire, also it cannot travel extended distances without a boost.

Another preference for network interaction is definitely coaxial (coax) cable connection, the kind utilized for cable. Coax cable will be constructed with an individual wire encircled by an insulation coat. The jacket can be itself surrounded by way of a braided or spiral-wound line. The inner cable carries the transmission,

and the exterior braid functions as a soil. The most trusted computer conversation coax cable is usually Ethernet, carrying around 100 Mbps over ranges as high as 1500 feet.

Coax cable in addition is suffering from degradation of sign quality over range. Repeaters (for electronic indicators) or amplifiers (for analog indicators) could be spaced periodically across the cable to get the indication, amplify it, take away spurious signals known as "sound," and retransmit it.

Optical Fiber

A newer type of cable is constructed of very slim strands of cup. Instead of holding electricity, these fibers bring pulses of lighting. The bandwidth of optical fiber content is around 1000 Mbps, plus the signal degrades much less over fibers than over line or coax; the fibre is wonderful for a run of around 2.5 miles. Optical fiber entails less interference, much less crossover between adjacent multimedia, less expensive, and less pounds than copper. Therefore, optical fiber is normally a far greater transmission channel than copper. Therefore, as copper age range, it is staying changed by optical fibers in most connection systems. Specifically, most long-distance communication lines are actually fiber.

Wireless

Radio signals may also carry communications. Much like pagers, cellular microphones, garage front door openers, and convenient telephones, wireless stereo may be used in networks, carrying out a protocol produced for short-range telecommunications, selected the 802.11 category of standards. The cordless medium can be used for short ranges; it is specifically useful for sites where the nodes are literally close together, such as for example in a workplace or in the home. Various 802.11 products are becoming designed for home and workplace wireless networks.

Microwave

Microwave is really a form of stereo transmission especially perfect for outdoor connection. Microwave includes a channel capacity much like coax cable; that's, it carries comparable amounts of information. Its principal advantages would be that the signal is sturdy from level of transmitting to level of receipt. Thus,

microwave signals need not turn out to be regenerated with repeaters, simply because do signs on cable.

Nevertheless, a microwave transmission travels inside a straight line, showing a problem as the planet curves. Microwave alerts travel by type of view: The transmitter and recipient must be in the straight line collectively, without intervening obstacles, such as for example mountains. As found in Figure7-.3, a direct microwave signal transported between towers of fair height can take a trip a mileage of no more than 30 miles due to the earth's curvature. Consequently, microwave signals are usually "bounced" from recipient to receiver, spaced significantly less than 30 miles aside, to cover an extended distance.

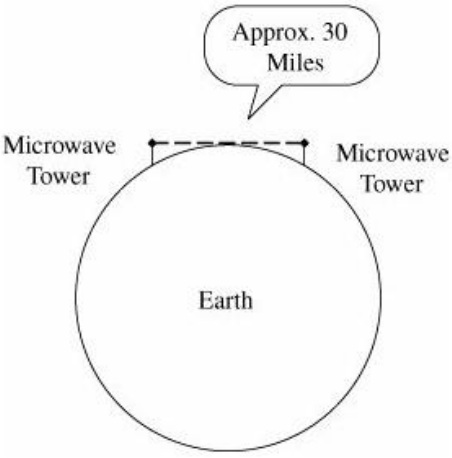


Figure 7.3. Microwave Transmission.

Infrared

Infrared communication provides signals for quick distances (around 9 miles) and in addition requires an apparent line of look. Because it will not require cabling, it really is convenient for convenient objects, such as for example laptops and links to peripherals. An infrared sign is hard to intercept since it is really a point-to-point signal. Nevertheless, it is at the mercy of "in the centre" attacks where the interceptor functions just like a repeater, getting the indication, extracting any wanted info, and retransmitting to the initial destination the initial signal or

perhaps a modified version. Due to line-of-sight prerequisites and limited mileage, infrared is normally found in a protected room, such as for example an office, where in-the-middle attacks will be complicated to conceal.

Satellite

Many communications, such as for example international calls, must travel round the earth. In the first days of cell phone technology, telephone organizations ran huge wires across the ocean's bottom, allowing calls to visit in one continent to some other. Today, we've other choices. The communication businesses spot satellites in orbits which are synchronized while using rotation of the planet earth (referred to as geosynchronous orbits), therefore the satellite seems to hover in a set situation 22,300 kilometers above the planet earth. Although the satellite television can be high-priced to launch, after in space it really is essentially free of maintenance. Furthermore, the grade of a satellite interaction link is frequently much better than an earthbound cable.

Satellites become naive transponders: Whatever they acquire they transmit out again. As a result, satellites are actually sophisticated receivers, for the reason that their sole functionality is to obtain and repeat impulses. In the user's perspective, the signal fundamentally "bounces" from the satellite and back again to earth. For instance, a sign from THE UNITED STATES journeys 22,300 a long way in to the sky and exactly the same distance back again to a spot in Europe. The procedure of bouncing a sign off a dish is revealed in Figure 7-4.

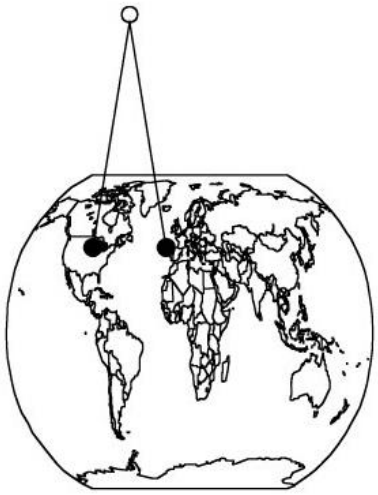


Figure 7.4. Satellite Communication.

We can job a signal to some satellite with realistic accuracy, however the satellite isn't expected to have got the same degree of accuracy and reliability when it delivers the signal back again to earth. To lessen complexity and get rid of beam concentrating, satellites typically disperse their transmissions over an extremely wide area. A fairly narrow position of dispersion through the satellite's transmitter generates a fairly wide pattern (known as the footprint) at first glance of the planet earth due to the 22,300-mile yardage from the satellite television to earth. Therefore, a typical dish transmission could be received more than a path some hundred miles vast; some handle the width of the complete continental USA in one transmission. For a few applications, such as for example satellite television, an easy footprint is attractive. But for safe communications, small the footprint, the fewer the chance of interception.

Protocols

When we work with a network, the interaction media are often translucent to us. That's, the majority of us have no idea whether our conversation is transported over copper cable, optical fiber, satellite television, microwave, or some mixture. Actually, the communication channel may differ from one transmission to another. This ambiguity is truly a positive feature of your system: its self-reliance. That's, the communication is certainly separated from the specific medium of connection. Independence can be done because we've defined methods that permit a user to see the community at a higher, abstract degree of interaction (viewing it with regards to user and information); the facts of the way the communication is completed are covered within software and hardware at both ends. The program and hardware permit us to put into practice a network in accordance with a standard protocol stack, a split architecture for marketing communications. Each layer within the stack is similar to a terminology for communicating info pertinent at that coating.

Two popular standard protocol stacks are employed often for implementing sites: the Start Methods Interconnection (OSI) along with the Transmission Control Standard protocol and Internet Standard protocol (TCP/IP) structures. We examine each one of these in turn.

ISO OSI Reference Model

The International Standards Organization (ISO) Open Systems Interconnection model consists of layers by which a network communication occurs. The OSI reference model contains the seven layers listed in Table 7-1.

Table 7-1. OSI Protocol Layer Levels.

Layer	Name	Activity
7	Application	User-level data
6	Presentation	Standardized data appearance, blocking, text compression
5	Session	Sessions or logical connections between parts of an application; message sequencing, recovery
4	Transport	Flow control, end-to-end error detection and correction, priority service
3	Network	Routing, message blocking into uniformly sized packets
2	Data Link	Reliable data delivery over physical medium; transmission error recovery, separating packets into uniformly sized frames
1	Physical	Actual communication across physical medium; individual bit transmission

How communication performs across the several layers is definitely depicted in Figure 7-5. We are able to think about the tiers as producing an assembly series, where each layer brings its own program to the connection. In concert, the levels represent the various activities that must definitely be performed for genuine transmission of a note. Separately, each level serves an objective; equivalent layers carry out similar functions to the sender and device.

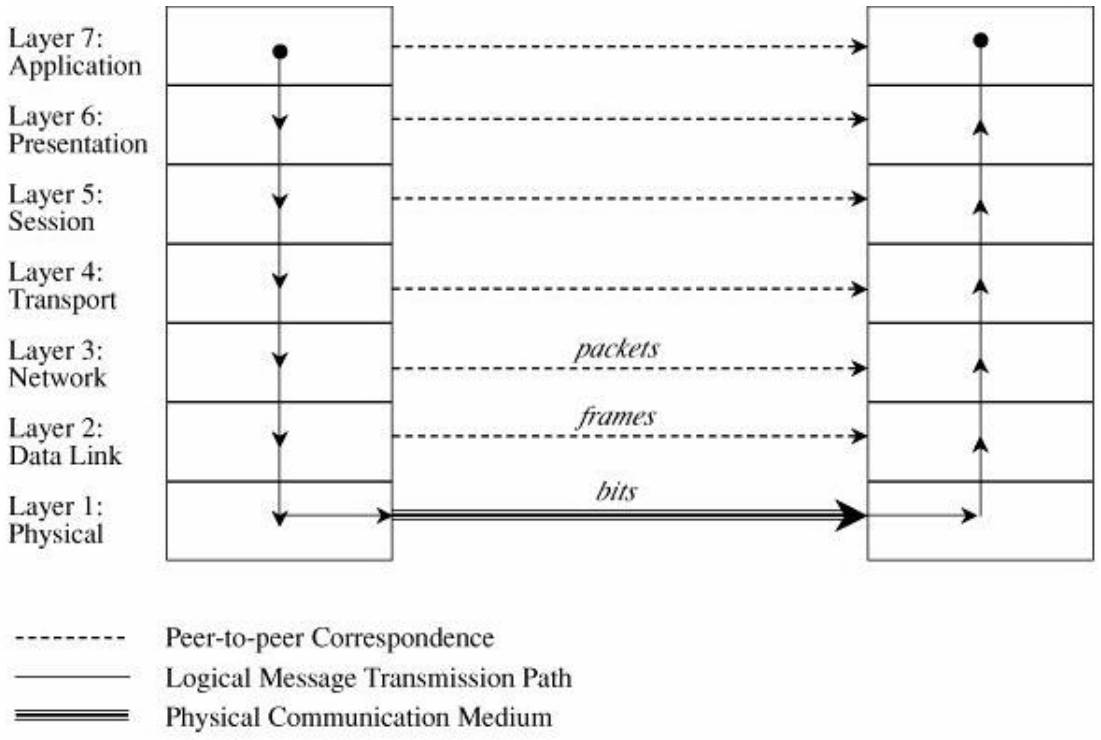


Figure 7.5. ISO OSI Network Model.

Each layer goes by info in three instructions: above which has a layer communicating even more abstractly, parallel or across to exactly the same coating in another coordinator, and below using a layer handling much less abstract (that's, more requisite) data things. The marketing communications above and here are actual interactions, as the parallel one is really a virtual communication avenue. Parallel layers will be named "peers."

Let us take a look at a simple exemplary case of protocol transmission. Guess that, to send e mail to a pal, you run a credit card application such as for example Eudora, View, or Unix email. You type a note, utilizing the application's editor, and the application form formats the subject matter into two pieces: a header that presents to whom the concept is supposed (and also other things, such as for example sender and period sent), and also a body which has the text of one's message. The application form reformats your information into a normal format in order that even though you and your good friend use different email applications, it is possible to still swap e-mail. This change is revealed in Figure 7-6.

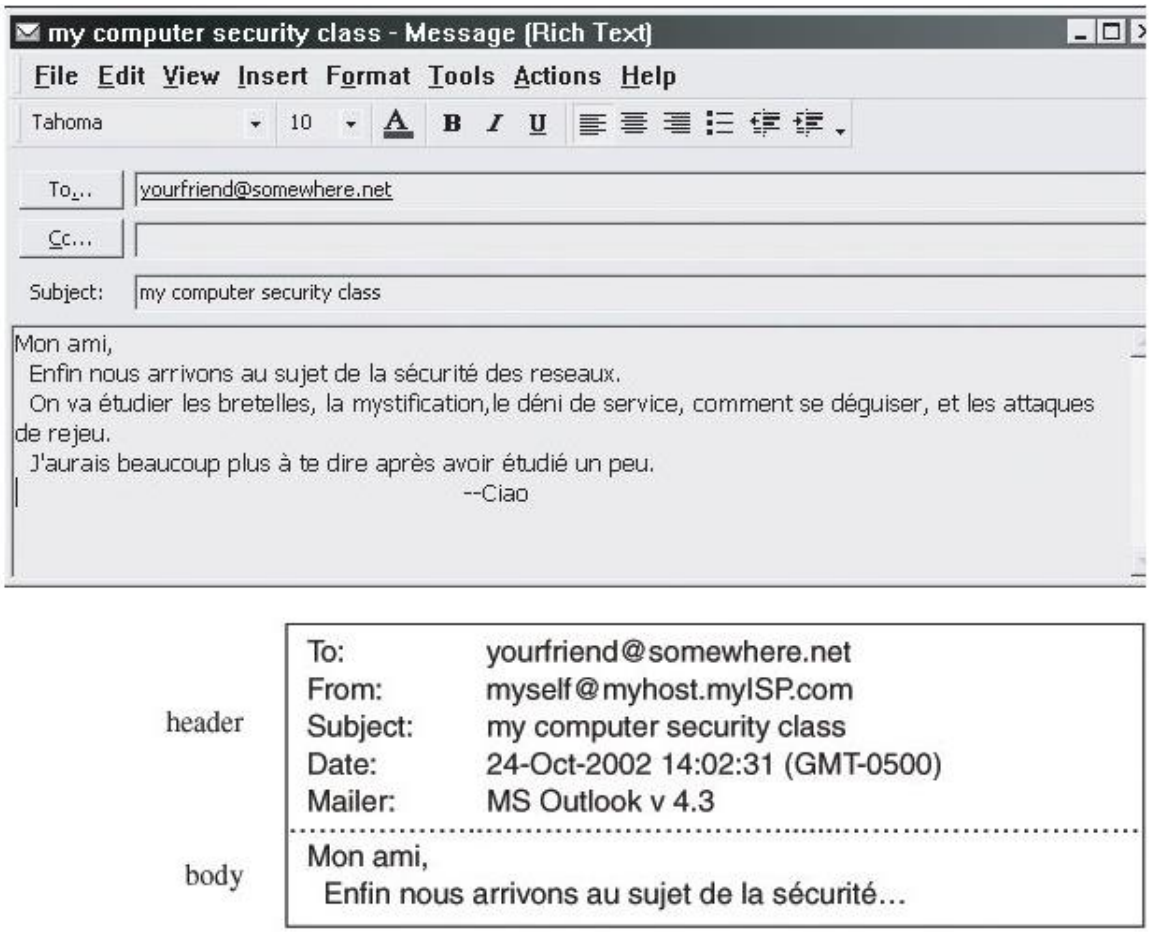


Figure 7.6. Transformation.

However, the communication is not carried just as you typed it, as fresh text. Raw content material is an extremely inefficient coding, because an alphabet utilizes relatively several 255 possible character types to have an 8-tad byte. Rather, the presentation level will probably change the uncooked text into another thing. It may carry out compression, figure conversions, and also some cryptography. An e-mail meaning is really a one-way move (from sender to device), so it's not necessarily initiating a program in which info fly backwards and forwards between your two endpoints. As the notion of any communication session isn't directly relevant with this scenario, we disregard the session layer for the present time. Occasionally, spurious alerts intrude in the communication route, as when static rustles a phone line or disturbance intrudes on the radio or television set signal. To handle this, the carry layer adds mistake detection and modification coding to filter these spurious indicators.

TCP/IP

The OSI style is really a conceptual one; it demonstrates the different things to do required for giving a communication. Nevertheless, full implementation of any seven-layer transmission bears too much over head for megabit-per-second marketing communications; the OSI process slows things right down to unacceptable levels. Because of this, TCP/IP (Transmitting Control Standard protocol/Internet Standard protocol) may be the protocol stack useful for most wide region network marketing communications. TCP/IP was designed for what grew to become the web. TCP/IP is described by protocols, not really layers, but we are able to think about it with regards to four levels: use, host-to-host (end-to-end) transportation, Internet, and bodily. In particular, a credit card application program deals just with abstract information items significant to the application form individual. Although TCP/IP is frequently used as an individual acronym, it certainly denotes two various methods: TCP implements an attached communications session together with the more standard IP transport process. In fact, one third process, UDP (customer datagram standard protocol) can be an essential area of the suite.

The transport covering receives variable-length emails from the application form layer; the transfer layer breaks or cracks them into units of controllable size, moved in packets. THE WEB layer transmits use coating packets in datagrams, transferring them to various physical connections in line with the data's location (provided within an address accompanying the info). The real layer includes device drivers to execute the specific bit-by-bit data interaction. Table 7-2 demonstrates how each covering plays a part in the complete interaction.

IDOL

Table 7-2. Internet Communication Layers.	
Layer	**Responsibilities**
Application	user interactions User interaction, addressing
Transport	Sequencing, reliability (integrity), error correction
Internet	Flow control, routing
Physical	Data communication

The TCP standard protocol must ensure the right sequencing of packets along with the integrity (appropriate transmitting) of information within packets. The process will place out-of-sequence packets in suitable order, demand retransmitting an absent packet, and acquire a fresh duplicate of a broken packet. In this manner, TCP hands and fingers a blast of correct info in proper purchase to the invoking software. But this support comes at a cost. Recording and verifying sequence volumes, verifying integrity investigations, and asking for and looking forward to retransmissions of faulty or absent packets devote some time and induce overhead. Most applications count on a flawless blast of bits, however, many software can tolerate a not as much accurate blast of data if swiftness or efficiency is crucial.

A TCP packet is really a data structure which includes a sequence variety, an acknowledgment amount allowing you to connect the packets of the communication program, flags, and supply and vacation spot port quantities. A port is really a number designating a specific application jogging on some type of computer. For instance, if Jose and Walter commence a communication, they set up a unique channel amount where their personal computers can course their respected packets to all of them. The channel range is named an interface. Each service runs on the well-known port, such as for example interface 80 for HTTP (webpages), 23 for Telnet (far off terminal relationship), 25 for SMTP (e-mail), or

IDOL

161 for SNMP (community management). More specifically, each one of these services includes a waiting procedure that screens the specified interface number and will try to execute its assistance on any files passed towards the port.

7.2. Threats in Networks

What Makes a Network Vulnerable?

An isolated residence user or perhaps a stand-alone business office with several employees can be an unlikely target for most attacks. But put in a community to the blend and the chance rises sharply. Think of how a community differs from the stand-alone surroundings:

- **Anonymity**. An attacker can support a harm from a large number of miles away rather than come into primary contact with the machine, its administrators, or customers. The actual attacker is as a result safe behind an electric shield. The invasion can be transferred through a great many other hosts in order to disguise the attack's source.

- **Many factors of attackbothtargets and origins**.. A straightforward computing system is really a self-contained unit. Accessibility controls using one machine protect the confidentiality of information on that cpu. However, whenever a file is placed in a system host far off from an individual, the info or the data file itself may go through many hosts to access an individual. One host's administrator may enforce demanding security plans, but that administrator does not have any control over some other hosts within the network. Thus, an individual must be determined by the access command mechanisms in each one of these systems. An invasion will come from any number to any sponsor, so that a big network offers numerous factors of vulnerability.

- **Sharing** Because networks allow source and workload posting, more users contain the potential to gain access to networked devices than on sole computers. Perhaps even worse, access can be afforded to even more systems, in order that access adjustments for single techniques may be insufficient in networks.

IDOL

- **Complexity of method.**We observed that a main system is a difficult software application. Reliable security is certainly difficult, or even impossible, on a big operating system, especially one not developed specifically for security and safety. A network offers several possibly dissimilar os's. Therefore, a community operating/control system may very well be more technical than an operating-system for an individual computing system. On top of that, the ordinary pc today has increased computing ability than did various office computers within the last 2 decades. The attacker may use this capacity to advantage by evoking the victim's computer to execute area of the attack's computation. And because the average computer is indeed powerful, most consumers have no idea what their personal computers are really carrying out at any time: What operations are mixed up in background when you are participating in Invaders from Mars? This complexness diminishes confidence inside the network's security.

Unknown perimeter. A network's expandability likewise implies uncertainty concerning the system boundary. One coordinator might be a node on two distinct networks, so sources on one system are accessible towards the users of another network as well. Although wide convenience is a benefits, this unidentified or uncontrolled band of possibly malicious consumers is a protection disadvantage. An identical problem develops when fresh hosts could be put into the community. Every community node should be able to respond to the possible occurrence of different, untrustablehostsFiure 7-11 highlights the issues in determining the boundaries of the network. Notice, for instance, that a person on a bunch in community D could be unaware of the actual connections from customers of networks A new and B. As well as the host in the center of systems A and B actually belongs to A, B, C, and E. If you can find different security regulations for these systems, to what guidelines is that web host subject?

Unknown perimeter. Shape 7-12 illustrates that there could be many paths in one host to some other. Guess that an individual on coordinator A1 really wants to send a note to a person on variety B3. That communication may be routed through hosts C or D before coming to web host B3. Host C might provide acceptable security, however, not D. Network consumers seldom have handle on the routing of these messages.

Thus, a system differs significantly from the stand-alone, local atmosphere. Network characteristics considerably increase the protection risk.

Who Attacks Sites?

That are the attackers? We cannot list their labels, just as we cannot know that are all the thieves in our metropolis, country, or the planet. Even though we recognized who these were, we have no idea if we're able to stop their actions. (Find Sidebar 7-3 for an initial, tenuous url between psychological features and hacking.) To possess some notion of who the attackers may be, we go back to concepts released in Section 1, where we detailed the three important the different parts of an episode: method, possibility, and motive.

Within the next sections we discover method: equipment and strategies the attackers make use of. Here we take into account earliest the motives of attackers. Concentrating on motive can provide us some notion of who might assault a networked number or end user. Four significant motives will be challenge or strength, fame, funds, and ideology.

Challenge

Why do individuals do risky or daunting items, like climb hills or swim the British Channel or take part in extreme sports? Due to the challenge. The problem is no various for someone professional on paper or using plans. The single most crucial motivation for your network attacker may be the intellectual challenge. They're intrigued with understanding the responses to MAY I defeat this system? What would occur if I attempted this process or that method?

Some attackers benefit from the intellectual activation of defeating the supposedly undefeatable. For instance, Robert Morris, who perpetrated the web worm in 1988 (detailed in Section 3), attacked supposedly being an experiment to find if he could exploit a specific vulnerability. Different attackers, like the Cult with the Dead Cow, get to show weaknesses in safety defenses in order that others can pay attention to conditioning security. Still various other attackers will be unnamed, unknown folks working persistently merely to see how very good they can will end up in performing unwelcome pursuits.

IDOL

Fame

The task of accomplishment will do for a few attackers. But different attackers seek identification for their routines. That is, area of the challenge does the deed; another component is taking credit rating for it. Oftentimes, we have no idea who the attackers are really, but they abandon behind a "calling credit card" using a brand or moniker: Mafiaboy, Kevin Mitnick, Fluffy Bunny, and participants on the Chaos Computer Pub, for instance. The actors frequently maintain some anonymity through the use of pseudonyms, however they achieve fame nonetheless. They may definitely not have the ability to brag as well openly, however they enjoy the private thrill of experiencing their attacks authored up in the news headlines media.

Funds and Espionage

As in additional settings, financial incentive motivates attackers, as well. Some attackers carry out industrial espionage, looking for info on a company's goods, consumers, or long-range strategies. We know professional espionage includes a role whenever we read about notebooks and sensitive paperwork having been raised from resort rooms when other even more valuable items have been left out. Some countries are usually notorious for making use of espionage to assist their state-run companies.

Sometimes commercial espionage is in charge of seemingly strange commercial behavior. For instance, in July 2002, newspaper publishers reported a Yale University stability audit had discovered that admissions officials from rival Princeton School broke into Yale's on-line admissions notification technique. The Princeton snoops accepted considering the confidential selections about eleven college students who had put on both universities but who hadn't yet been advised of their choices by Yale. In another circumstance, a startup business was going to activate its very first application on the net. Two days prior to the application's unveiling, the top offices have been burglarized. The only real item stolen was basically the one laptop or computer comprising the application's community design. Corporate representatives had to produce a difficult option: Go surfing realizing that a rival might then benefit from knowing the inner architecture or hold off the product's rollout before network design has been changed. They find the latter. Similarly, the principle of stability for a significant

manufacturing company features documented privately to us of data that certain of the business's competitors had taken facts. But he could acquire no measures because he cannot establish which of three competition the specific culprit was.

Industrial espionage is definitely illegal, nonetheless it occurs, partly due to the high potential get. Its lifetime and consequences could be embarrassing for the prospective companies. Thus, various incidents head out unreported, and you can find few reliable data on how many professional espionage and "dirty techniques" continue. Annually since 1997, the Pc Security Institute as well as the U.S. National Bureau of Analysis have surveyed stability professionals from organizations, government agencies, colleges, and organizations, wanting to know them to document perceptions of personal computer situations. About 500 replies are received for every study. Theft of intellectual home amounted to a complete lack of $31 million, having an average damage per event of $350 thousand, causeing this to be the group of third-highest reduction. That amount has been more than increase the amount noted within the 2004 study. (These survey email address details are anecdotal, so it's hard to bring countless conclusions. For total information on the survey notice [CSI05].) Industrial espionage, resulting in lack of intellectual property, is actually a problem.

Organized Crime

With the development in commercial price of the web, participation by arranged crime in addition has increased. In Oct 2004, police imprisoned members of your 28-individual gang of Web criminals, known as the Shadowcrew, who controlled outside of six foreign international locations and eight says in america. Six leaders of this team pled guilty to fees, concluding an illicit company that trafficked in at the very least 1.5 million taken credit and charge card numbers and led to losses more than $4 million. In July 2003, Alexey Ivanov seemed to be convicted because the supervisor of the wide-ranging, organized legal enterprise that involved in sophisticated adjustment of computer information, financial facts, and charge card quantities. Ivanov and team were in charge of an aggregate lack of about $25 million. And in Jan 2006, Jeanson Wayne Ancheta pled guilty to presenting attacked 400,000 personal computers with malicious program code and booking their make use of to others to utilize to launch episodes on others. In June 2005, the FBI and police from 10 various other countries performed over 90 queries

worldwide within "Operation Site Lower," made to disrupt and dismantle lots of the leading criminal institutions that illegally deliver and buy and sell in copyrighted program, movies, songs, and games on the net. Brazilian law enforcement officials arrested 85 persons in 2005 for World wide web fraud.

Although money can be popular to these offences, the more intriguing fact is they often entail collaborators from many countries. These extra sophisticated attacks demand several person training of a room, and so firm and individual duties follow. With possible revenue inside the huge amount of money and operations concerning thousands of charge card numbers along with other pieces of id, existing organized criminal offenses units will definitely get sucked in. As Williams claims, "[T]here keeps growing evidence that arranged crime groups are usually exploiting the brand new opportunities provided by the web."

Ideology

Within the last few years, we have been starting to get cases where attacks are usually perpetrated to enhance ideological ends. For instance, many security experts think that the Code Crimson worm of 2001 premiered by a class motivated by the strain in U.S.China and Taiwan relationships. Denning [DEN99a] offers recognized between two forms of related manners, hactivism and cyberterrorism. Hactivism includes "procedures that employ hacking approaches against a target's [system] along with the intention of disrupting ordinary operations however, not causing serious harm." In some instances, the hacking sometimes appears as giving tone of voice into a constituency that may otherwise not become heard by the business or government firm. For instance, Denning describes pursuits such as digital sit-ins, where an interest team floods an organization's site with traffic to show support of a specific position. Cyberterrorism will be more threatening than hactivism: "politically enthusiastic hacking operations designed to cause grave cause harm to such as lack of life or extreme economic harm."

Reconnaissance

Now that we've listed various motives for attacking, we consider how attackers perpetrate their disorders. Attackers usually do not ordinarily sit back with a terminal and start an attack. An inspired attacker investigates and strategies before acting. In the same way you might make investments time in studying a

jewelry retailer before getting into to steal as a result, a system attacker learns a whole lot about a prospective target before you begin the strike. We examine the precursors with an attack in order that if we are able to recognize characteristic behaviour, we may have the ability to block the invasion before it really is launched.

Because most susceptible networks are linked to the web, the attacker commences preparation by learning whenever you can about the aim for.

Port Scan

A good way to gather community information is by using a port check out, an application that, for a specific IP address, reviews which ports react to announcements and which of more than a few known vulnerabilities appear to be current. Farmer and Venema are one of the primary to spell it out the technique.

A port scan is similar to a routine real examination from the doctor, specially the initial questions employed to find out a health background. The inquiries and answers independently may not seem to be significant, however they point to parts that suggest more investigation.

Port scanning explains to an attacker three stuff: which regular ports or products and services are jogging and responding on the prospective system, what operating-system is set up on the mark technique, and what programs and variations of applications can be found. This information is definitely designed for the asking from the networked system; it could be obtained silently, anonymously, without recognition or authentication, pulling little if any focus on the scan.

Port scanning resources are plentiful, and not simply for the underground local community. The nmap scanning device by Fyodor at www.insecure.org/nmap is really a useful software that anyone can download. Granted a target, nmap will review all open jacks, the assistance they assist, and the dog owner (user Identification) in the daemon supplying the program. (The dog owner is significant since it indicates what privileges would descend upon somebody who compromised that assistance.) Another easily available scanner can be netcat, compiled by Hobbit, at www.l0pht.com/users/l0pht. (That Web address can be "letter ell," "digit zero," p-h-t.) Professional products certainly are a little more

high priced, however, not prohibitive. Well-known professional scanners are usually Nessus (Nessus Corp.), CyberCop Scanning device (Network Affiliates), Secure Scanning device (Cisco), and Web Scanner (Net Security Techniques).

Social Engineering

The port check gives an outside picture of any network where will be the windows and doors, of what exactly are they constructed, from what kinds of areas do they open up? The attacker in addition wants to know very well what is in the building. What much better way to learn than to consult?

Suppose, while relaxing in your workstation, you obtain a telephone call. "Hello, that is John Davis as a result support. We have to test some relationships on the inner network. Would you please manage the command word ipconfig/all on your own workstation and read through if you ask me the addresses it exhibits?" The demand looks innocuous. But if you don't learn John Davis and his work responsibilities effectively, the caller could possibly be an attacker collecting information on the within architecture.

Social engineering entails using social expertise and personal conversation to get you to definitely reveal security-relevant facts and perhaps possibly to do a thing that permits an harm. The idea of social anatomist would be to persuade the target to be beneficial. The attacker frequently impersonates someone in the organization who's in the bind: "My notebook has just ended up stolen and I have to change the security password I had located onto it," or "I must get out an essential report rapidly and I cannot access the following matter." This harm works especially properly in the event the attacker impersonates an individual in a higher position, like the division vice chief executive or the top of IT protection. (Their names can often be entirely on a public site, in a community registration with the web registry, or in promotion and posts.) The strike is often fond of someone low good enough for being intimidated or amazed by the high-level man or woman. A direct telephone call and expressions of superb urgency can override any all natural instinct to look at the story.

Because the sufferer has aided the attacker (plus the attacker has got profusely thanked the target), the prey will think there is nothing wrong rather than report the event. Thus, the destruction may possibly not be known for quite a while.

An attacker offers little to reduce in attempting a social executive attack. At most detrimental it will increase knowing of a possible concentrate on. If the social engineering is certainly directed against somebody who isn't skeptical, especially somebody not involved with security management, this could be successful. We as human beings like to assist others when requested politely.

Intelligence

From a slot scan the attacker recognizes what is available. From social executive, the attacker recognizes certain internal specifics. But a far more detailed floor approach would be fine. Intelligence may be the general expression for collecting facts. In security and safety it often identifies gathering discrete items of information from numerous sources and putting them together with each other like the bits of a puzzle.

One popular intelligence technique is named "dumpster diving." It will involve looking through items which have already been discarded in rubbish bins or recycling bins. It is awesome what we dispose of without great deal of thought. Blended with the remains to be from lunch may be system diagrams, printouts of safety measures device configurations, program designs and resource code, cell phone and employee listings, and more. Possibly outdated printouts could be useful. Rarely will the construction of a security and safety device change totally. More often only 1 rule is added in or erased or changed, so an attacker includes a big probability of an effective attack in line with the old information.

Operating System and Application Fingerprinting

The port check provides the attacker with very certain information. For example, an attacker may use a port check out to learn that port 80 is usually open and facilitates HTTP, the process for transmitting webpages. However the attacker will probably have many relevant questions, such as for example which professional server application is usually running, what variant, and what the actual operating-system and version happen to be. Once armed with this particular more information, the attacker can consult with a list of certain software's regarded vulnerabilities to find out which certain weaknesses to attempt to exploit.

How do the attacker solution these problems? The network methods are normal and vendor self-employed. Even now, each vendor's program code is implemented individually, so there could be minor variants in interpretation and habit. The variations usually do not make the program noncompliant with the typical, but they will vary enough to create each edition distinctive. For instance, each version could have different sequence statistics, TCP flags, and fresh options. To understand why, take into consideration that sender and device must organize with sequence amounts to implement the bond of an TCP period. Some implementations answer with confirmed sequence amount, others react with the main greater, among others respond having an unrelated number. In the same way, certain flags in a single version will be undefined or incompatible with others. What sort of system responds to some prompt (for example, by acknowledging it, asking for retransmission, or disregarding it) may also reveal the machine and version. Eventually, new features provide a strong hint: A fresh version will put into action a new characteristic but a vintage edition will reject the question. Each one of these peculiarities, sometimes referred to as the operating-system or use fingerprint, can indicate the maker and version.

For example, along with performing its interface scan, a scanning device such as for example nmap will reply with a estimate at the mark operating system. To find out more about how that is done, start to see the report at www.insecure.org/nmap/nmap-fingerprinting-article.html.

Sometimes the application form identifies itself. Generally a client-server relationship is handled totally within the application form according to standard protocol guidelines: "More information send me this site; OK but manage this support program code; thanks, I simply did." However the application cannot react to a message that will not follow the predicted form. For example, the attacker might work with a Telnet program to give meaningless messages to some other application. Ports such as for example 80 (HTTP), 25 (SMTP), 110 (POP), and 21 (FTP) may react with something similar to

Server: Netscape-Commerce/1.12

Your browser directed a non-HTTP compliant subject matter.

or

Microsoft ESMTP Email Service, Variant: 5.0.2195.3779

This reply shows the attacker which use and version happen to be running.

Bulletin Boards and Chats

The Internet is just about the greatest instrument for sharing expertise since the creation of the publishing press. It really is probably also probably the most dangerous software for sharing information.

Different underground bulletin planks and boards support change of facts. Attackers can publish their hottest exploits and methods, study what others did, and seek out more information on systems, programs, or sites. Understand that, much like everything on the net, anyone can publish anything, so there is absolutely no guarantee that the info is efficient or accurate. And you also never know who's reading from the web.

Availability of Documentation

The sellers themselves sometimes deliver information that's beneficial to an attacker. For instance, Microsoft makes a resource set up by which software vendors can check out a Microsoft merchandise to be able to develop appropriate, complementary programs. This toolkit as well gives attackers equipment to utilize in investigating something that can consequently be the concentrate on of an invasion.

Reconnaissance: Concluding Remarks

An excellent thief, that's, an effective one, spends moment understanding the framework of the mark. To get ready for perpetrating a loan provider robbery, the thief might keep an eye on the bank, finding just how many guards you can find, when they have breaks, when money shipments arrive, etc.

Threats in Transit: Eavesdropping and Wiretapping

By now, you can view an attacker can accumulate a significant quantity of information regarding a victim before you begin the actual invasion. Once the setting up is performed, the attacker is preparing to proceed. Within this section we choose the forms of attacks that may appear. Recall from Section 1 an attacker has countless ways where to harm in a very computing surroundings:

lack of confidentiality, integrity, or availableness to data, equipment or software, operations, or other belongings. Because a community involves info in transit, we appear first in the harm that may happen between a senders including a recipient.

The simplest way to attack is merely to listen inside. An attacker can decide on off this content of a connection passing inside the clear. The word eavesdrop suggests overhearing without expending any additional effort. For instance, we might claim an attacker (or perhaps a system administrator) is definitely eavesdropping by tracking all traffic moving by way of a node. The administrator may have a legitimate objective, such as enjoying for inappropriate usage of resources (for example, visiting non-work-related sites from a provider community) or connection with inappropriate get-togethers (for example, passing files for an enemy from the military laptop or computer).

A far more hostile term will be wiretap, this means intercepting marketing communications through some work. Passive wiretapping is merely "being attentive," similar to eavesdropping. But productive wiretapping indicates injecting something in to the communication. For instance, Marvin could exchange Manny's communications along with his own or make communications purported being from Manny. Formerly derived from hearing in on telegraph and mobile phone communications, the word wiretapping generally conjures up a bodily act where a tool extracts information since it flows on the wire. However in fact no real contact is essential. A wiretap can be carried out covertly in order that neither the sender nor the device of a interaction recognizes that the material have already been intercepted.

Wiretapping works in different ways with regards to the communication medium utilized. Let us seem more diligently at each probable choice.

Cable

At most local levels, all signals within an Ethernet or various other LAN can be found on the cable television for anybody to intercept. Each LAN connection (like a computer panel) includes a unique street address; each board and its own drivers are designed to brand all packets from its sponsor with its exclusive address (being a sender's "return address") also to take from the web simply those packets attended to to its coordinator.

IDOL

But removing just those packets dealt with to confirmed host is really a issue of politeness; there's little to avoid an application from evaluating each packet since it goes by. A tool referred to as a packet sniffer can get all packets within the LAN. Alternatively, among the interface cards could be reprogrammed to really have the supposedly unique street address of another pre-existing card within the LAN in order that two different credit cards will both fetch packets for just one address. (In order to avoid recognition, the rogue credit card must put back online copies of this packets it includes intercepted.) Luckily (for the present time), LANs are often used simply in environments which are fairly welcoming, so most of these attacks arise infrequently.

Clever attackers may take benefit of a wire's qualities and read through packets without the physical manipulation. Regular wire (and several other electronic elements) emit rays. By a procedure named inductance an intruder can touch a line and go through radiated indicators without making actual physical connection with the cable connection. A cable's indicators travel only brief distances, plus they can be obstructed by some other conductive materials. The gear needed to grab signals is cheap and an easy task to obtain, thus inductance threats certainly are a serious issue for cable-based systems. For the harm to operate, the intruder should be fairly near to the cable; this type of attack is as a result limited to circumstances with reasonable bodily access.

When the attacker isn't close sufficiently to benefit from inductance, then extra hostile measures could be warranted. Easy and simple type of intercepting a cable TV is by immediate cut. In case a cable is definitely severed, all provider on it prevents. Within the mend, an attacker can simply splice in a second cable that subsequently receives a duplicate of all alerts along the main cable. You can find ways to be considered a little less noticeable but accomplish exactly the same goal. For instance, the attacker might thoroughly expose a number of the outer conductor, hook up to it, then diligently expose a number of the interior conductor and hook up to it. Both these operations change the resistance, known as the impedance, with the cable. In the initial case, the maintenance itself alters the impedance, as well as the impedance change could be explained (or hidden) within the repair. In the next case, just a little social executive can make clear the transformation

Signals on the network happen to be multiplexed, and therefore several signal is carried at confirmed time. For instance, two analog (audio) signals could be mixed, like two shades in a very musical chord, and two electronic signals could be blended by interleaving, like handmade cards staying shuffled. A LAN provides different packets, but info on the WAN could be heavily multiplexed since it leaves its delivering host. As a result, a wiretapper over a WAN must be able not merely to intercept the required communication but additionally to draw out it from others with which it really is multiplexed. While this is done, your time and effort involved means it'll be used sparingly.

Microwave

Microwave signals aren't transported along a cable; they are transmit through the surroundings, making them considerably more attainable to outsiders. Generally, a transmitter's indication is targeted on its matching receiver. The transmission path is rather wide, to be certain of striking the recipient, as proven in Figure 7.13. From the stability standpoint, the broad swath can be an invitation to mischief. Not merely can someone intercept a microwave transmitting by interfering with the type of look between sender and device, someone may also pick up a whole transmitting from an antenna situated near but slightly off of the direct focus stage.

A microwave signal is normally certainly not shielded or isolated to avoid interception. Microwave can be, therefore, an extremely insecure medium. On the other hand, due to the large level of traffic taken by microwave back links, it really is unlikelybut definitely not impossiblethat someone can separate a person transmission from all of the others interleaved in it. A privately owned or operated microwave link, taking only communications for just one organization, isn't so well covered by volume.

IDOL

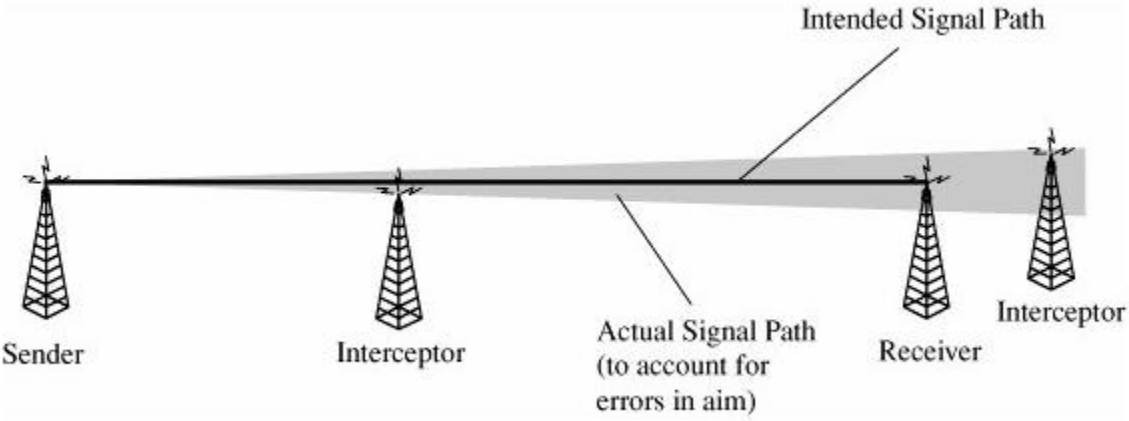


Figure 7.13. Path of Microwave Signals.

Satellite Communication

Satellite communication includes a similar issue of staying dispersed over a location higher than the intended stage of reception. Diverse satellites have diverse characteristics, however, many signals could be intercepted within an area different hundred miles huge and one thousand miles long. Subsequently, the prospect of interception is sustained than with microwave indicators. However, because dish communications are usually heavily multiplexed, the chance is smaller than anybody communication will undoubtedly be intercepted.

Optical Fiber

Optical fiber presents two significant protection advantages over various other transmission media. Very first, the complete optical network should be tuned carefully whenever a new connection is manufactured. Therefore, no-one can touch an optical program without recognition. Clipping just one single fiber in a lot of money will destroy the total amount in the community.

Second, optical fibers carries light vitality, not electricity. Light source will not emanate a magnetic discipline as electricity will. So, an inductive touch is impossible with an optical fiber cable television.

Just using fiber content, however, will not guarantee security, any longer than does making use of encryption. The repeaters, splices, and taps along a cable connection are places of which data could be available easier than in the fibers

IDOL

wire itself. The cable connections from computing devices to the fibre can also be details for penetration. Alone, fiber is a lot better than cable, nonetheless it has vulnerabilities also.

Wireless

Wireless networking is now extremely popular, with justification. With cordless (also called WiFi), folks are not linked with a wired interconnection; they are absolve to roam throughout an workplace, house, or making while maintaining a link. Universities, offices, and also home customers like having the ability to hook up to a network minus the cost, problem, and trouble of running wire connections. The down sides of cordless arise in the power of intruders to intercept and spoof a link.

As we observed earlier, wireless marketing communications travel by stereo. In America, wireless computer relationships share exactly the same frequencies as storage area door openers, native radios (typically utilized as baby screens), some cord-less telephones, along with other very short length applications. Even though frequency band will be crowded, few programs are expected to get on the music group from any one user, thus contention or disturbance isn't an issue.

But the big threat isn't interference; it really is interception. A radio signal is solid for about 100 to 200 legs. To understand those figures, photograph a typical ten-story workplace, ten office buildings "wide" by five office buildings "deep," much like many structures in workplace parks or on college campuses. Believe you create a wireless bottom station (device) in the part of the very best floor. That stop could receive alerts transmitted from the contrary corner of the bottom floor. In case a similar building have been adjacent, the transmission may be received during that building, too

A strong signal could be picked up conveniently. And with a cheap, tuned antenna, a radio signal could be picked up more than a few miles away. Quite simply, someone who wished to pick up your unique signal could achieve this from several roads away. Parked within a truck or truck, the interceptor could check your communications for a long time without arousing suspicion.

Interception

Interception of cordless traffic is definitely a menace, through either unaggressive or effective wiretapping. You might respond to that risk by let's assume that encryption will treat it. Unfortunately, encryption isn't always useful for wireless communication, plus the encryption included in some wireless products is not just as strong since it ought to be to deter a separate attacker.

Theft of Service

Wireless likewise admits another problem: the chance of rogue usage of a network link. Many hosts operate the Dynamic Coordinator Configuration Process (DHCP), where litigant negotiates a one-time Ip and connection with a bunch. This protocol pays to in business office or campus options, where not absolutely all users (customers) are effective anytime. A small amount of IP addresses could be shared among consumers. Fundamentally the addresses can be purchased in a pool. A fresh client requests a link and an Ip through DHCP, along with the server assigns one through the pool.

This structure admits a large issue with authentication. Unless the number authenticates consumers before assigning a link, any requesting customer is allocated an Ip and network admittance. (Generally, this assignment arises before the person on your client workstation actually recognizes and authenticates to some server, so furthermore there may possibly not be an authenticatable identification the fact that DHCP server can require.) The problem is so severe that in a few urban centers a map can be acquired, showing many sites accepting wireless associations.

A user wanting free of charge Internet access could get it by simply finding a cellular LAN supplying DHCP services. But could it be legal? In distinct conditions Benjamin Smith III in Florida in July 2005 and Dennis Kauchak in Illinois in March 2006 had been convicted of remotely being able to access some type of computer wirelessly minus the owner's authorization. Kauchak has been sentenced into a $250 fine. Consequently, while you have the ability to connect, it could not be lawful to take action.

Alternatively, some places or organizations help make wireless access openly available as a residential area service. Free cordless cities contain Albuquerque and Honolulu in America, Oulu in Finland, along with the key districts of metropolitan areas such as for example Hamburg, Germany, and Adelaide,

Australia. The metropolitan areas hope that giving free gain access to will spur IT expansion and attract vacationers and business travellers.

Overview of Wiretapping

There are lots of points of which network traffic can be acquired to the interceptor. Figure 7.14 illustrates how marketing communications are exposed off their origin with their destination.

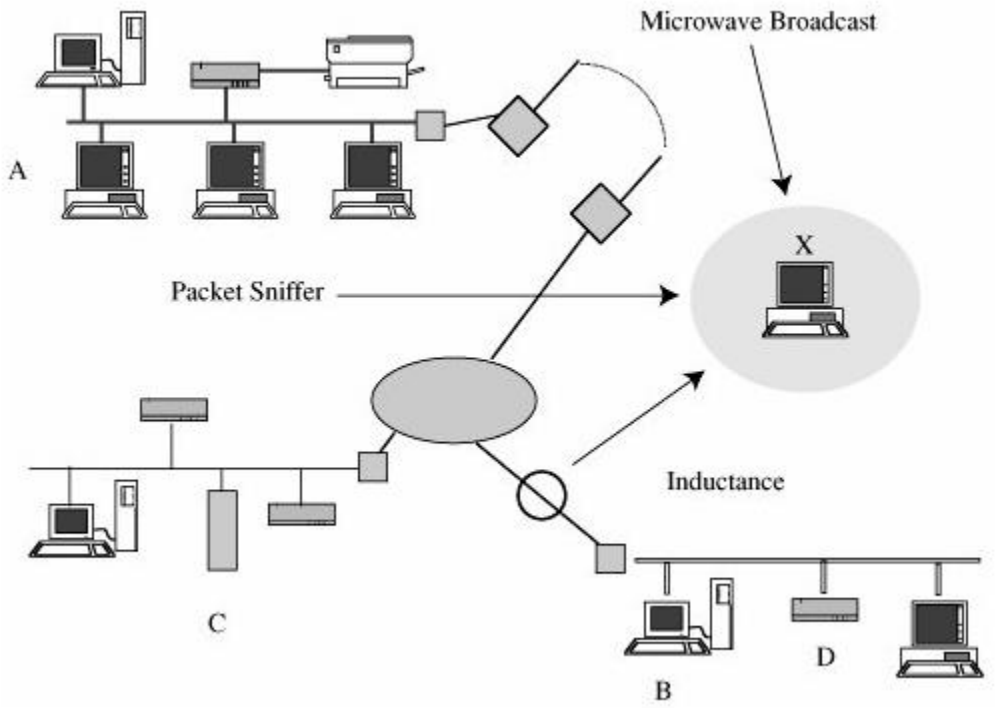


Figure 7.14. Wiretap Vulnerabilities.

From a safety standpoint, you need to assume that communication hyperlinks between community nodes could be broken. Because of this, commercial network customers employ encryption to safeguard the confidentiality of these communications, once we demonstrate later in such a chapter. Local system communications could be encrypted, although for functionality reasons it might be preferable to safeguard local contacts with strong actual and administrative safety measures instead.

Protocol Flaws

Internet protocols happen to be publicly uploaded for scrutiny by the complete Internet area. Each accepted process is well known by its Obtain Comment (RFC) amount. Many issues with protocols have already been identified by distinct reviewers and corrected prior to the protocol was recognized as a typical.

But protocol meanings are created and evaluated by fallible human beings. Likewise, protocols are usually integrated by fallible human beings. For instance, TCP connections happen to be established through collection numbers. Your client (initiator) directs a sequence variety to open a link, the server responds with this number including a sequence amount of its, and your client responds together with the server's sequence quantity. Suppose an individual can imagine a client's following sequence number. See your face could impersonate your client within an interchange. Sequence volumes are incremented on a regular basis, so it may be easy to anticipate the next quantity. Impersonation

In most cases, there is a less strenuous approach than wiretapping for acquiring home elevators a system: Impersonate someone else or procedure. Why chance tapping a range, or why bother extracting one connection out of several, when you can obtain the identical data directly? Impersonation is really a more significant hazard in a broad area system than in an area one. Local men and women often have improved ways to obtain gain access to as another customer; they can, for instance, simply be seated at an unattended workstation. Even now, impersonation attacks shouldn't be ignored actually on geographic area networks, because geographic area networks are occasionally mounted on wider area systems without anyone's 1st thinking from the security implications.

Within an impersonation, an attacker offers many choices:

- Suppose the identification and authentication information on the target.
- Pick the individuality and authentication information on the target from the previous interaction or from wiretapping.
- Circumvent or disable the authentication system at the prospective computer.
- Make use of a target that won't be authenticated.
- Apply a goal whose authentication information are known.

Let us take a look at each choice.

Authentication Foiled by Guessing

Chapter 4 claimed the outcomes of several tests showing that lots of users select easy-to-guess passwords. In Chapter 3, we found that the web worm of 1988 capitalized on accurately that flaw. Morris's worm tried out to impersonate each person on a concentrate on machine by hoping, in order, a small number of variations of an individual name, a summary of about 250 frequent passwords and, ultimately, the words within a dictionary. Sadly, several users' accounts remain available to these easy assaults.

A second way to obtain password guesses can be default passwords. Countless systems are originally set up with default company accounts having Visitor or ADMIN as login IDs; associated these IDs are usually well-known passwords such as for example "guest" or "null" or "password" make it possible for the administrator to create the machine. Administrators often overlook to remove or disable these trading accounts, or at the very least to improve the passwords.

In a reputable environment, such as for example a business office LAN, a security password may simply be considered a signal that an individual does not wish others to utilize the workstation or accounts. Often the password-protected workstation is made up of sensitive data, such as for example employee wages or information regarding new products. Consumers may believe the password will do to help keep out an inquisitive colleague; they find no reason to safeguard against concerted strikes. On the other hand, if that dependable environment is linked to an untrustworthy wider-area community, all consumers with straightforward passwords become quick targets. In fact, some systems aren't originally linked to a wider community, so their consumers commence in a much less exposed circumstance that clearly alters when the relationship occurs.

Dead accounts provide a final way to obtain guessable passwords. To observe how, suppose Teacher Romine, a faculty representative, takes abandon for per year to instruct at another university or college. The existing bank account may reasonably turn out to be kept on keep, awaiting the professor's go back. But an attacker, reading through a university papers online, realizes that an individual is away. Right now the attacker utilizes social anatomist on the machine administration ("Hello that is Professor Romine dialing from my short-term office

at Talk about School. I haven't applied my take into account a long time, but now I want something as a result urgently. I've forgotten the security password. Can you make sure you reset it to ICECREAM? No? Properly, send me a fresh password by e-mail to my bank account r1@stateuniv.edu.") Additionally, the attacker can check out several passwords before password guessing restriction is exceeded. The machine after that locks the bill administratively, as well as the attacker runs on the social engineering strike. In every these techniques the attacker may flourish in resetting or obtaining a password.

Authentication Thwarted by Eavesdropping or Wiretapping

Due to the rise in spread and client-server processing, some users have admission privileges on more than a few connected machines. To safeguard against arbitrary outsiders making use of these accesses, authentication is necessary between hosts. This gain access to can involve an individual directly, or it could be done automatically with respect to the user by way of a host-to-host authentication standard protocol. In any case, the bank account and authentication information on the subject will be passed for the destination web host. When this info are offered the network, they're subjected to anyone watching the communication in the network. These identical authentication details could be used again by an impersonator until they're changed.

Because transmitting a security password in the clean is a important vulnerability, protocols have already been developed so the password itself by no means retains a user's workstation. But, once we have seen in a number of other places, the facts are important.

Microsoft LAN Director was an early on method for employing networks. It possessed a password alternate mechanism where the password itself seemed to be never transmitted inside the clear; instead just a cryptographic hash of it had been transmitted. A security password could contain around 14 characters. It might include top- and lowercase characters, digits, and particular people, for 67 alternatives in virtually any one posture, and 6714 opportunities for a complete 14-persona password quite a good work factor. Even so, those 14 characters weren’t diffused over the entire hash; these were sent in independent substrings, representing 14 characters 17 and 814. A 7-personality or shorter security

password acquired all nulls in the next substring and has been immediately recognizable. An 8-figure password acquired 1 figure and 6 nulls in the next substring, therefore 67 guesses would discover the one character. Even yet in the best circumstance, a 14-personality password, the task factor dropped from 6714 to 677 + 677 = 2 * 677. These job factors differ by way of a factor of around 10 billion. LAN Supervisor authentication was maintained in many later on systems (like Windows NT) being an option to help backward compatibility with techniques such as Glass windows 95/98. This training is an excellent exemplary case of why protection and cryptography have become precise and should be monitored by specialists from principle through design and style and implementation.

Authentication Foiled by Avoidance

Obviously, authentication works well only once it functions. A fragile or flawed authentication enables usage of any technique or one who can circumvent the authentication.

In a classic operating-system flaw, the buffer for typed personas in a security password was of resolved size, keeping track of all character types typed, consisting of backspaces for modification. If an end user typed more

Nonexistent Authentication

If two computer systems are employed by exactly the same users to retail outlet data and manage processes and when each possesses authenticated its consumers on first gain access to, you might believe that computer-to-computer or nearby user-to-remote method authentication is pointless. These two personal computers and their consumers are an honest environment where the added difficulty of recurring authentication seems abnormal.

Even so, this assumption isn't valid. To understand why, think about the UNIX operating-system. In UNIX, the document .rhosts lists respected hosts and .rlogin listings trusted users that are allowed entry without authentication. The data files are designed to support computer-to-computer relationship by users who've recently been authenticated at their principal hosts. These "trusted hosts" may also be exploited by outsiders who access one system via an authentication weakness (like a guessed security password) and transfer to some other system

IDOL

that allows the authenticity of the user who originates from something on its respected list.

An attacker could also realize that something provides some identities necessitating no authentication. Some devices have got "guest" or "anonymous" trading accounts to permit outsiders to gain access to things the methods want to discharge to anyone. For instance, a loan provider might post an ongoing listing of forex rates, a catalogue with an on line catalog will make that catalog designed for anyone to look for, or a business might allow usage of a few of its information. Aindividual can sign in as "guest" and get publicly available things. Typically, no security password is necessary, or an individual is shown a note requesting that an individual sort "GUEST" (or your title, which really signifies any string that appears like a brand) when called for a password. Each one of these accounts allows usage of unauthenticated users.

Well-Known Authentication

Authentication data ought to be unique and tough to suppose. But however, the capability of one well-known authentication plan occasionally usurps the security. For instance, one computer producer planned to utilize the same security password to permit its remote preservation personnel to gain access to some of its computers owned by some of its customers across the world. Fortunately, security industry experts pointed out the actual risk before that thought was set up.

The system management process (SNMP) is trusted for remote control of network products, such as for example routers and switches that help no ordinary consumers. SNMP runs on the "neighborhood string," basically a security password for the city of devices that may interact with each other. But network products are designed specifically for quick installation with reduced configuration, and several network administrators usually do not modify the default area string set up on a router or change. This laxity creates these devices over the network perimeter available to many SNMP problems.

Some vendors nevertheless ship personal computers with one technique administration account mounted, using a default security password. Or the methods feature a demonstration or check account, without required security

password. Some administrators neglect to alter the passwords or erase these accounts.

Trusted Authentication

Finally, authentication may become an issue when identification is usually delegated to additional trusted sources. For example, a data file may indicate who is able to be respected on a specific number. Or the authentication system for one program can "attest to" a consumer. We noted before the way the UNIX .rhosts, .rlogin, and /etc/hosts/equiv documents suggest hosts or consumers that are respected on some other hosts. While these benefits are of help to users who've accounts on numerous devices or for system management, routine maintenance, and operation, they need to be used meticulously. All of them represents a possible hole by which a remote control user or a distant attacker an achieve accessibility.

Spoofing

Guessing or elsewhere obtaining the community authentication credentials of the entity (a customer, an account, an activity, a node, and a tool) allows an attacker to make a full communication beneath the entity's identification. Impersonation falsely presents an appropriate entity inside a communication. Closely associated is definitely spoofing, when an attacker falsely keeps on one end of your networked interchange. Types of spoofing will be masquerading, time hijacking, and man-in-the-middle assaults.

Masquerade

In the masquerade one variety pretends to become another. A standard example is Link confusion. Names of domain can easily end up being confused, or somebody can simply mistype certain titles. Therefore xyz.com, xyz.org, and xyz.internet may be three different corporations, or one real organization (for instance, xyz.com) and two masquerade makes an attempt from somebody who registered the comparable domain names. Labels with or without hyphens (coca-cola.com versus cocacola.com) and effortlessly mistyped brands (l0pht.com versus lopht.com, or citibank.com versus citybank.com) will be applicants for masquerading.

From attacker's perspective, the enjoyment in masquerading gets into before the cover up is removed. For instance, suppose you intend to attack a genuine bank, First Azure Bank or investment company of Chicago. The specific bank gets the domain BlueBank.com, and that means you register the domain Blue-Bank.com. Upcoming, you set up a website at Blue-Bank.com, most likely using the true Blue Bank emblem that you down loaded to make your website look whenever you can like that on the Chicago bank. Lastly, you ask visitors to log in making use of their name, account amount, and security password or PIN. (This redirection may appear in lots of ways. For example, it is possible to purchase a banner that links to your internet site rather than the actual banks, or it is possible to deliver e-mail to Chicago occupants and request them to go to your website.) After accumulating personal info from several loan provider users, it is possible to drop the bond, pass the bond to the real Blue Bank or Investment Company, or continue steadily to collect more info. You may have the ability to transfer this relationship smoothly for an authenticated usage of the real Glowing blue Bank so the user certainly not realizes the deviation.

Phishing is the fraudulent try and gain sensitive informationincluding usernames, passwords and credit score card details by disguising oneself as a truthful entity in an electronic communication. Typically executed by way of e - mail spoofing or on the instant messaging, it frequently directs customers to enter into personal records at a fake internet site which fits the look and sense of the valid web page.

Phishing is an example of social engineering techniques getting used to misinform customers. Users are regularly lured by using communications purporting to be from trusted events together with social web websitesauction sites, banks, on line payment processors or IT administrators.

Attempts to cope with phishing incidents encompass law, person training, public cognizance, and technical safety features (the latter being because of phishing attacks regularly exploiting weaknesses in modern net safety).

An example of a phishing electronic mail, disguised as an reliable e-mail from a (fictional) bank. The sender is attempting to trick the recipient into revealing exclusive statistics by way of "confirming" it at the phisher's internet site. Note the misspelling of the phrases obtained and discrepancy as acquired and discrepancy, respectively. It is also really worth noting that, although the URL of

the bank's webpage appears to be valid, the hyperlink could actually be pointed on the phisher's web site.

Session Hijacking

Computer session in normal day-to-day speak is a transient interaction you have with a website. For example, the time between you first log into your bank account, after which log off after your operation, is a session.

During a session hijacking, a malicious hacker locations himself in among your computer and the website's server (Facebook for example), while you're engaged in an active session.

At this factor, the malicious hacker actively monitors the whole thing that occurs to your account, and can even kick you out and take manage of it.

The biggest gain of a session hijacking is that the malicious attacker can enter the server and get right of entry to its records without having to hack a registered account. In addition, he can also make changes on the server to assist him hack it in the future or to simplify a facts-stealing operation.

In cryptography and computer protection, a person-in-the-middle attack (MITM) is an attack wherein the attacker secretly relays and likely alters the communications among events who accept as true with they may be directly speaking with every other. One example of a MITM assault is lively eavesdropping, wherein the attacker makes independent connections with the sufferers and relays messages between them to make them consider they may be speak immediately to each other over a private connection, while in truth the complete communications is managed with the aid of the attacker. The attacker need to be capable of intercept all applicable messages passing between the 2 partiesand inject new ones. This is straightforward in many instances; as an example, an attacker within reception range of an unencrypted wireless access point (Wi-Fi) may want to insert themselves as a man-in-the-middle.

As it targets to avoid mutual authentication, a MITM attack can be triumphant best whilst the attacker impersonates each endpoint sufficiently nicely to fulfill their expectations. Most cryptographic protocols include some form of endpoint authentication in particular to save you MITM attack. For instance, TLS can

authenticate one or each events the use of at the same time depended on certificate authority.

Suppose Alice desires to talk with Bob. Meanwhile, Mallory desires to intercept the communique to eavesdrop and optionally to supply a fake message to Bob.

First, Alice asks Bob for his public key. If Bob sends his public key to Alice, however Mallory is able to intercept it, an MITM attack can begin. Mallory sends Alice a solid message that looks to originate from Bob, however rather includes Mallory's public key.

Alice, believing this public key to be Bob's, encrypts her message with Mallory's key and sends the enciphered message back to Bob. Mallory again intercepts, deciphers the message the use of her private key, probably alters it if she desires, and re-enciphers it the use of the general public key she intercepted from Bob whilst he in the beginning attempted to send it to Alice. When Bob gets the newly enciphered message, he believes it got here from Alice.

1. Alice sends a message to Bob, that's intercepted by way of Mallory:

Alice "Hi Bob, it is Alice. Give me your key." → Mallory Bob

2. Mallory relays this message to Bob; Bob cannot inform it isn't always definitely from Alice:

Alice Mallory "Hi Bob, its Alice. Give me your key." → Bob

3. Bob responds along with his encryption key:

Alice Mallory ← [Bob's key] Bob

4. Mallory replaces Bob's key together with her very own, and relays this to Alice, claiming that it's far Bob's key:

Alice ← [Mallory's key] Mallory Bob

5. Alice encrypts a message with what she believes to be Bob's key, wondering that best Bob can examine it:

Alice "Meet me at the bus forestall!" [Encrypted with Mallory's key] → Mallory Bob

6. However, as it become surely encrypted with Mallory's key, Mallory can decrypt it, examine it, adjust it (if favored), re-encrypt with Bob's key, and ahead it to Bob:

Alice Mallory "Meet me on the van down by way of the river!" [Encrypted with Bob's key] → Bob

7. Bob thinks that this message is a at ease conversation from Alice.

This instance suggests the need for Alice and Bob to have a few way to ensure that they're really every using each other's public keys, rather than the public key of an attacker. Otherwise, such assaults are usually possible, in principle, in opposition to any message sent the use of public-key era. A variety of strategies can assist shield in opposition to MITM attacks.

Message Confidentiality Threats

An attacker can without problems violate message confidentiality (and possibly integrity) because of the general public nature of networks. Eavesdropping and impersonation assaults can cause a confidentiality or integrity failure. Here we bear in mind numerous other vulnerabilities which could have an effect on confidentiality.

An attacker can without problems violate message confidentiality (and possibly integrity) due to the public nature of networks. Eavesdropping and impersonation attacks can lead to a confidentiality or integrity failure. here we take into account several different vulnerabilities that can affect confidentiality.

Misdelivery

Sometimes messages are misdelivered due to a few flaw in the network hardware or software program. Maximum regularly, messages are misplaced totally, that is an integrity or availability issue. Every so often, but, a destination address is changed or a few handler malfunctions, inflicting a message to be added to a person other than the intended recipient. All of these "random" activities are quite unusual.

IDOL

Extra common than network flaws are human mistakes. It's far a long way too easy to mistype an cope with inclusive of 100064,30652 as 10064,30652 or 100065,30642, or to type "idw" or "iw" in preference to "diw" for David Ian Walker, who is referred to as Ian via his buddies. There may be truly no justification for a computer network administrator to discover people with the aid of meaningless lengthy numbers or cryptic initials when "iwalker" would be a ways much less prone to human error.

Publicity

To guard the confidentiality of a message, we need to track it all the manner from its creation to its disposal. Along the manner, the content of a message may be exposed in brief buffers; at switches, routers, gateways, and intermediate hosts throughout the community; and inside the workspaces of strategies that build, format, and present the message. In in advance chapters, we considered confidentiality exposures in programs and running systems. All of those exposures apply to networked environments as properly. Furthermore, a malicious attacker can use any of those exposures as a part of a trendy or targeted assault on message confidentiality.

Passive wiretapping is one supply of message publicity. So also is subversion of the shape through which a verbal exchange is routed to its destination. Ultimately, intercepting the message at its supply, destination, or at any intermediate node can lead to its publicity.

raffic Flow Analysis

Sometimes now not only is the message itself sensitive but the reality that a message exists is additionally sensitive. For example, if the enemy for the duration of wartime sees a large amount of community site visitors between headquarters and a unique unit, the enemy may be capable to infer that great action is being planned involving that unit. In a business setting, messages despatched from the president of one company to the president of a competitor may want to lead to speculation about a takeover or conspiracy to restore prices. Or communications from the high minister of one usa to every other with whom diplomatic family members had been suspended could lead to inferences about a rapprochement between the countries. In these cases, we want to shield each the content of messages and the header data that identifies sender and receiver.

Message Integrity Threats

In many cases, the integrity or correctness of a conversation is at least as essential as its confidentiality. In reality for some situations, such as passing authentication data, the integrity of the communication is paramount. In different cases, the want for integrity is much less obvious. Next we reflect onconsideration on threats primarily based on disasters of integrity in communication.

Falsification of Messages

Increasingly, human beings rely on digital messages to justify and direct actions. For example, if you receive a message from a right pal asking you to meet at the pub for a drink subsequent Tuesday evening, you will probably be there at the appointed time. Likewise, you will comply with a message from your supervisor telling you to cease work on challenge A and dedicate your electricity rather to undertaking B. As long as it is reasonable, we tend to act on an electronic message simply as we would on a signed letter, a smartphone call, or a face-to-face communication.

However, an attacker can take advantage of our have faith in messages to misinform us. In particular, an attacker may

Change some or all of the content of a message

Replace a message entirely, inclusive of the date, time, and sender/receiver identification

Reuse (replay) an historical message

Combine pieces of special messages into one

Change the apparent supply of a message

Redirect a message

Destroy or delete a message

These attacks can be perpetrated in the ways we have already examined, including

Active wiretap

Trojan horse

IDOL

Impersonation

Preempted host

Preempted workstation

Noise

Signals dispatched over communications media are field to interference from traffic on the same media, as good as from natural sources, such as lightning, electric motors, and animals. Such unintended interference is known as noise. These forms of noise are inevitable, and they can threaten the integrity of data in a message.

Fortunately, communications protocols were deliberately designed to overcome the negative effects of noise. For example, the TCP/IP protocol suite ensures detection of nearly all transmission errors. Processes in the communications stack notice errors and prepare for retransmission, all invisible to the larger-level purposes. As a result, noise is scarcely a consideration for users in security-critical purposes.

Format Failures

Network communications work on the grounds that of well-designed protocols that define how two computers communicate with a minimum of human intervention. The structure of a message, the dimension of a data unit, a sequence of the message, even the meaning of a single bit is precisely described in a prescribed standard. The entire network works handiest considering all people obeys these rules and regulations.

Virtually everybody, that is. Attackers purposely break the rules to see what is going to happen. Or the attacker may search to exploit an undefined situation within the standard. The application could realize the violation of structure and lift an error indicator. Oftentimes, however, the malformation causes a software failure, which is able to result in a protection compromise, just what the attacker wants. In this part, we appear at a few kinds of malformation.

Protocol Failures and Implementation Flaws

Each and every protocol is a specification of a service to be provided; the provider is then implemented in application, may be incorrect. Network protocol application is general to the operating Systems, so flaws in that software can motive trendy harm for the reason that of the privileges with which the application runs and the impact of the application on many users at once. Particular Network protocol implementations had been the source of many protection flaws; peculiarly difficult have been SNMP (Network management), DNS (addressing service), and e-mail offerings akin to SMTP and S/MIME. Although unique vendors have applied the code for these services themselves, they probably are centered on a common (flawed) prototype. For illustration, the CERT advisory for SNMP flaws (Vulnerability Note 107186) lists roughly 200 exceptional implementations to which the advisory applies.

Or the protocol itself may be incomplete. If the protocol does no longer specify what action to soak up a designated concern, companies may just produce an exceptional outcome. So an interaction on windows, for instance, might prevail while the equal interaction on a UNIX process would fail.

The protocol can have an unknown protection flaw. In a basic example, Bellovin points out a weak point in the way packet sequence numbers are assigned attacker might intrude into communication in this type of manner that the intrusion is authorized as the real conversation and the actual sender is rejected.

Attackers can take advantage of all of a majority of these errors.

Website Vulnerabilities

A web site is in particular inclined on the grounds that it's almost wholly exposed to the consumer. If you happen to use an application, you do not generally get to view the software's code. With a website online, the attacker can download the web page's code for offline be trained over time. With a program, you may have little capability to manipulate in what order you access parts of the application, however, an online attacker gets to manage in what order pages are accessed,

possibly even having access to web page 5 without first having run pages 1 via 4. The attacker may additionally decide upon what information to give and might run experiments with unique information values to peer how the web site will react. In short, the attacker has some benefits that can be difficult to control.

The record of web page vulnerabilities is simply too lengthy to explore completely here. Hoglund and McGraw, Andrews and Whitaker, and Howard et al. offer fine analyses of the right way to find and fix flaws in the web software. Be definite to check the code progress issues, seeing that many code procedures there (such as buffer overflows and inadequate parameter checking) are applicable right here.

Web Site Defacement

some of the largest identified attacks are the web site defacement attack. Since of the tremendous number of web sites which have been defaced and the visibility of the outcome, the attacks are most of the time said within the fashionable press.

A defacement is normal now not best on the grounds that of its visibility but additionally when you consider that of the benefit with which one may also be completed. Websites are designed in order that their code is downloaded, enabling an attacker to obtain the whole hypertext report and all applications directed to the users within the loading procedure. An attacker may view programmers' comments left in as they built or maintained the code. The down load approach just about offers the attacker the blueprints to the site.

The benefit and enchantment of defacement are stronger by using the seeming plethora of vulnerabilities that web sites offer an attacker. For example, between December 1999 and June 2001 (the primary 18 months after its release), Microsoft furnished 17 security patches for its Internet Information Server (IIS) version 4.0 And variant 4.0 used to be an improvement for three prior versions, so theoretically Microsoft had a high-quality deal of time prior to determining its security flaws.

Buffer Overflows

Buffer overflow is alive and well on web sites, too. The attacker readily feeds a software some distance more information than it expects to obtain. A buffer dimension is passed, and the excess data spill over into adjoining code and data areas.

Maybe the satisfactory-recognized web server buffer overflow is the file name hindrance referred to as iishack. This attack is so good known that's has been written right into a method. To execute the procedure, an attacker provides as parameters the web page to be attacked and the URL of software the attacker wants that server to execute.

Other internet servers are susceptible to particularly lengthy parameter fields, similar to passwords of length 10,000 or a protracted URL padded with area or null characters.

Application Code Errors

A consumer's browser carries on an intricate, undocumented protocol interchange with purposes on the web server. To make its job less difficult, the online server passes context strings to the consumer, making the user's browser reply with full context. A challenge arises when the person can adjust that context.

To look why, recall our fictitious looking website called CDs-R-Us, promoting compact discs. At any given time, a server at that website may have a thousand or more transactions in more than a few states of completion. The website shows a page of items to order, the user selects one, the web page displays more items, the person selects one other, the web page shows extra items, the consumer selects two extra, and so on until the person is finished determining. Many men and women go on to whole the order via specifying cost and shipping information. However other humans use websites like this one as a web-based catalog or guide, and not using a real intention of ordering. For example, they can use this website to discover the price of the cutting-edge CD from Cherish the ladies; they may be able to use an online e-book service to investigate how many books by using Iris Murdoch are in print. And even supposing the user is a bona fide consumer, many times internet connections fail, leaving the transaction

incomplete. For these explanations, the webserver normally maintains track of the popularity of an incomplete order in parameter fields appended to the URL. These fields journey from the server to the browser and again to the server with each and every person resolution or page request.

Anticipate you could have chosen one CD and are watching at a second web page. The online server has passed you a URL similar to

http://www.CDs-r-us.Com/buy.Asp?I1=459012&p1=1599

This URL method you've chosen CD quantity 459012, and its price is $15.99. You now opt for a second and the URL becomes

http://www.CDs-r-us.Com/

purchase.Asp?I1=459012&p1=1599&i2=365217&p2=1499

However in case you are a clever attacker, you appreciate which you can edit the URL within the tackle window of your browser. As a consequence, you exchange each and every of 1599 and 1499 to 199. And when the server totals up your order, lo and behold, your two CDs price most effective $1.99 every.

Server-Side Include

A possibly more critical challenge is called a Server-Side Include. The concern takes knowledge of the truth that web sites may also be prepared to invoke an exact fuction. For example, many pages use web instructions to ship an email message in the "contact us" a part of the displayed web page. The commands, equivalent to electronic mail, if, goto, and incorporate, are positioned in a field that's interpreted in HTML.

One of the vital Server-Side Include commands is exec, to execute an arbitrary file on the server. For illustration, the Server-Side Include command

<!#exec cmd="/usr/bin/telnet &">

opens a Telnet session from the server strolling within the name of (that's, with the privileges of) the server. An attacker could in finding it interesting to execute instructions equivalent to chmod (exchange entry rights to an object), sh (set up a command shell), or cat (reproduction to a file).

IDOL

7.3. Network Security Controls

The set of security attacks is certainly long, and the news headlines media carry consistent accounts of severe security situations. From these, you might be prepared to conclude that system security can be hopeless. Fortunately, that's not the case. Prior chapters have displayed several approaches for addressing security problems, such as for example encryption for confidentiality and integrity, reference point monitors for accessibility management, and overlapping control buttons for defense comprehensive. These strategies may also be useful in guarding networks. This part presents many great defenses open to the network safety engineer. Subsequent areas provide thorough explanations for three specifically significant controlsfirewalls, intrusion diagnosis devices, and encrypted e-mail.

Security Threat Analysis

Remember the three measures of a stability threat evaluation in other scenarios. Initially, we scrutinize all of the parts of something so that we realize what each portion does and exactly how it interacts with other areas. Next, we take into account possible harm to confidentiality, integrity, and availableness. Eventually, we hypothesize the forms of attacks which could cause this destruction. We can have the same measures with a community. We start by looking at the average person elements of a system:

Local nodes attached via

Local communications back links to a

Local area system, which also offers

Local data safe-keeping,

Local procedures, and

Local devices.

The local system is also linked to a

Network gateway gives access via

IDOL

Network communications links to

Network control information,

System routers, and

Network resources, such as for example databases.

These functional demands are common for network customers. However now we look once more at these pieces, this time around conjuring the unwanted effects threat agents could cause. We posit a destructive agent call him Hector who really wants to attack networked marketing communications between two customers, Andy and Bo. What might Hector carry out?

Read communications. The messages delivered and received are usually uncovered inside Andy's device, at all locations through the community, and inside Bo's equipment. Therefore, a confidentiality invasion can be installed from practically anywhere in the network.

Modify communications from Andy to Bo. Once more, the messages are usually exposed by any means places from the network.

Forge communications allegedly from Andy to Bo. This step is even less difficult than changing a communication just because a forgery could be inserted from anywhere in the system. It do not need to originate with all the ostensible sender, also it does not need that a conversation be captured in transit. Since Andy will not deliver his marketing communications in person and since Bo may never have fulfilled Andy, Bo provides little time frame for judging whether a conversation purportedly directed by Andy is certainly authentic.

Inhibit communications from Andy to Bo. Right here again, Hector can perform this outcome by invading Andy's equipment, Bo's equipment, routers between them, or marketing communications links. He is able to also disrupt marketing communications generally by flooding the system or disrupting any exceptional path within the network.

Inhibit all communications passing by way of a point. If the idea resides on a distinctive way to or from the node, all site visitors to or from that node is definitely blocked. If the road is not distinctive, obstructing it shifts site visitors to various other nodes, most likely overburdening them.

Read data at some machine C between Andy and Bo. Hector can impersonate Andy (who's authorized to gain access to information at C). Bo might issue a note that seems outside of figure for Andy, but device C will nonetheless apply the accessibility handles for Andy. Additionally, Hector can invade (go an application on) device C to override admittance controls. Finally, he is able to search the community for machines which have weak or incorrectly administered access settings.

Modify or destroy data at C. Below once again, Hector can impersonate Andy and carry out anything Andy could perform. In the same way, Hector can make an effort to circumvent controls.

We summarize these dangers with an inventory:

- intercepting info in traffic
- accessing plans or info at distant hosts
- modifying plans or info at distant hosts
- modifying information in transit
- inserting communications
- impersonating a user
- inserting a duplicate of a prior communication
- blocking determined traffic
- preventing all traffic
- running an application at a distant host

Why are these attacks feasible? Dimension, anonymity, ignorance, misunderstanding, difficulty, dedication, and encoding all contribute. But we've help accessible; we appear next at certain hazards and their countermeasures. Afterwards in this section we check out how these countermeasures fit in together into particular tools.

Design and style and Implementation

Architecture

IDOL

As with consequently lots of the areas we've studied, planning could possibly be the strongest control. Specifically, when we create or enhance computer-based systems, we are able to give consideration to their overall structures and intend to "build in" security and safety among the key constructs. Likewise, the structures or design and style of a community can have a tremendous influence on its security.

Segmentation

In the same way segmentation was a robust security command in os's, it can reduce the prospect of harm inside a system in two crucial techniques: Segmentation decreases the amount of threats, also it limits the quantity of damage an individual vulnerability makes it possible for.

Assume your community implements electronic business for customers of the web. The fundamental elements of your network could be

- AWorld Wide Web server, to take care of customers' HTTP sessions
- Application code, to provide your things and services for sale
- a databases of goods, as well as perhaps an accompanying supply to the count up of stock readily available and being wanted from suppliers
- A data source of purchases taken

If each one of these activities were to perform on one device, your network will be in big trouble: Any bargain or failure of this machine would demolish your entire business capability.

A far more secure design utilizes multiple sections, as demonstrated in Figure 7-19. Imagine one little bit of hardware is usually to be an internet server box subjected to access by everyone. To reduce the chance of assault from beyond your system, that field should not likewise have other, more very sensitive, functions onto it, such as individual authentication or usage of a sensitive files repository. Separate sections and servers corresponding for the principles of minimum opportunity and encapsulation reduce the problems should any subsystem end up being compromised

IDOL

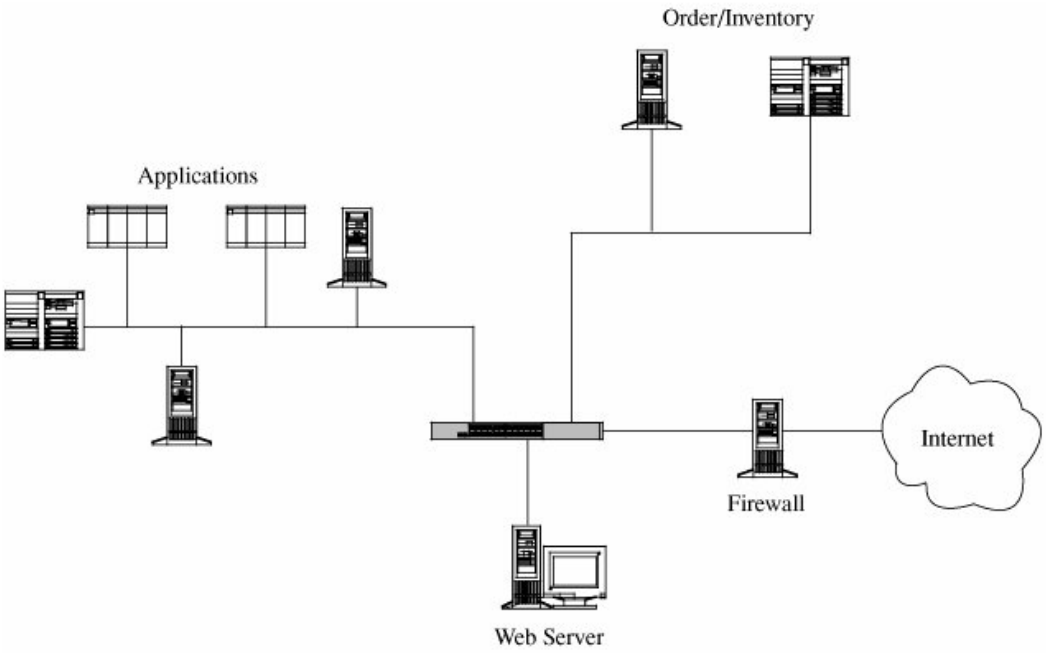


Figure 7-19. Segmented Architecture.

Separate access will be another solution to segment the system. For example, assume a network has been useful for three reasons: utilizing the "live" production method, testing another production variant, and developing following systems. In the event the network is very well segmented, external consumers can access simply the live method, testers should obtain only the check system, and coders should access simply the development method. Segmentation allows these three populations to coexist without risking that, for example, a programmer will inadvertently modify the production program.

Redundancy

Another essential architectural control is definitely redundancy: allowing for a function to become performed on several node, in order to avoid "putting all of the eggs in a single basket." For instance, the look of figure 7-19 has only 1 web server; shed it and everything connectivity is dropped. A better design and style could have two servers, employing what is known as failover method. In failover function the servers talk to each other routinely, each deciding if another is still dynamic. If one fails, another takes over running for both of these. Although

performance is certainly cut approximately in two when a malfunction occurs, at the very least some processing has been done.

Single Points of Failure

Ideally, the structures should produce the network defense to failure. Actually, the structures should at the very least ensure that the machine tolerates failure within an acceptable method (such as for example slowing down however, not stopping control, or recovering and restarting imperfect transactions). One method to evaluate the community architecture's tolerance of disappointment is to search for single things of failure. That's, we should consult when there is a single stage in the system that, if it have been to fall short, could deny usage of all or perhaps a significant area of the network. So, for instance, a single databases in one spot is susceptible to all the problems that could influence that location. Very good network design minimizes single tips of inability. Distributing the databaseplacing duplicates than it on different community segments, maybe even in different real locationscan decrease the risk of considerable harm from the failure at anybody point. There's often substantial over head in implementing this type of design; for instance, the independent directories should be synchronized. But typically we can cope with the failure-tolerant characteristics easier than with the hurt the effect of a failed single website link.

Architecture is important in implementing a great many other controls. We explain architectural features once we introduce other adjustments through the entire remainder of the chapter.

Mobile Agents

Mobile program code and hostile real estate agents are potential ways of attack, as identified earlier with this chapter. However, they are able to also be makes for good. Great agents might search for unsecured wireless accessibility, application vulnerabilities, or embedded harmful program code. Schneider and Zhou [SCH05] research distributed trust, by way of a corps of conversing, state-sharing agents. The theory is easy: Just like soldiers, you understand some agents will undoubtedly be stopped among others will undoubtedly be subverted because of the enemy, however, many agents will stay unchanged. The corps can get over Byzantine problems .Schneider and Zhou propose a style in which no-one agent is crucial to the entire success however the overall group could be trusted.

Encryption

Encryption is just about the most significant and versatile software for a system security expert. We've seen in previous chapters that encryption can be powerful for supplying level of privacy, authenticity, integrity, and restricted access to info. Because networks typically involve sustained risks, they often times secure information with encryption, possibly in conjunction with other controls.

Before we commence to study the usage of encryption to counter community security threats, why don't we consider these things? First, understand that encryption isn't a panacea or silver precious metal bullet. A flawed program style with encryption continues to be a flawed technique design. Second, observe that encryption protects simply what's encrypted (that ought to be apparent but isn't). Information are subjected between a user's fingertips plus the encryption procedure before they're transmitted, and they're exposed again after they have already been decrypted over the remote end. The very best encryption cannot drive back a destructive Trojan horses that intercepts information before the level of encryption. Lastly, encryption is not any better than its major supervision. If an attacker can suppose or deduce a poor encryption key, the overall game is over. Individuals who don't realize encryption sometimes slip-up it for fairy particles to sprinkle on something for magic security. This book wouldn't normally be wanted if many of these fairy dust been around.

In network programs, encryption could be applied frequently between two hosts (known as url encryption) or between two programs (known as end-to-end encryption). We take into consideration each below. With either type of encryption, key submission is always an issue. Encryption keys should be sent to the sender and recipient in a safe manner. In such a section, we as well investigate approaches for safe key circulation in networks. Eventually, we review a cryptographic center for a community computing environment

Link Encryption

In hyperlink encryption, data will be encrypted right before the system sites them within the physical communications back link. In cases like this, encryption happens at layer one or two 2 inside the OSI unit. (An identical situation comes about with TCP/IP methods.) Likewise, decryption occurs in the same way the

communication finds and gets into the receiving computer system. A style of link encryption can be shown in Figure 7-20.

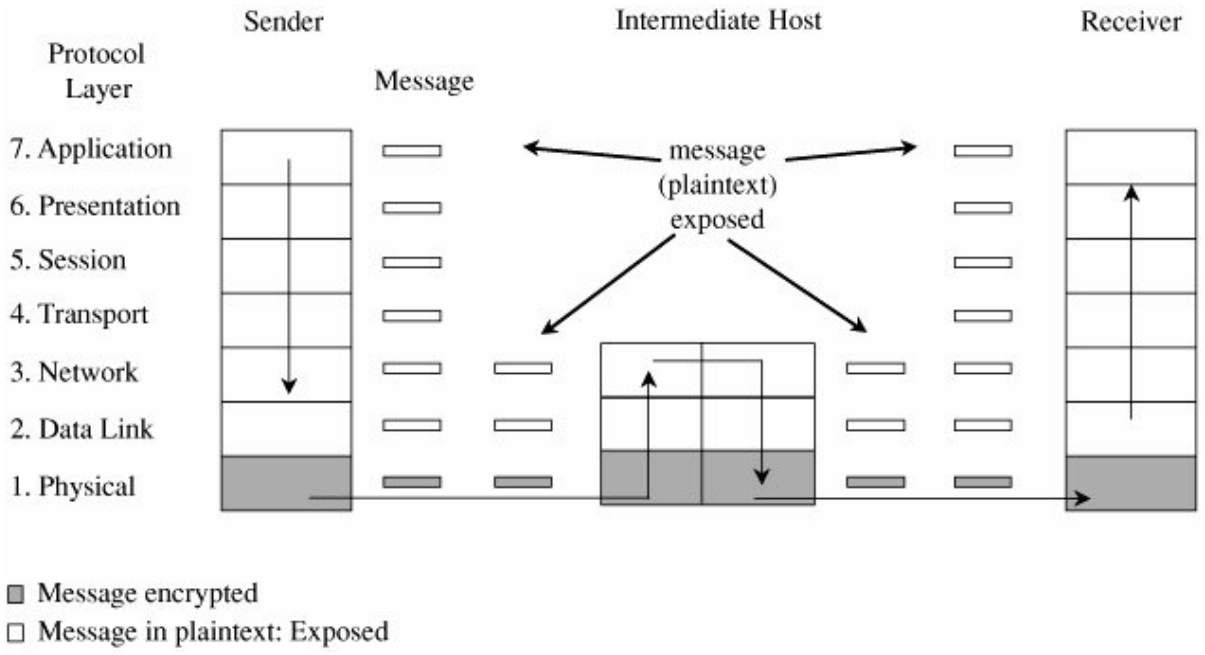


Figure 7-20. Link Encryption.

Encryption shields the communication in transit between two pcs, but the concept is at plaintext in the hosts. (A note in plaintext can be reported to be "within the clear.") Observe that as the encryption is included in the bottom protocol coating, the message is definitely exposed in every other layers in the sender and recipient. If we've good physical security and safety, we may not necessarily be too worried about this publicity; the exposure arises in the senders or receiver's number or workstation, secured by alarms or secured doors, for instance. Nevertheless, you need to observe that the message is definitely uncovered in two levels of most intermediate hosts by which the communication may pass. This exposure comes about because routing and addressing aren't read in the bottom layer, but simply at higher levels. The message is certainly in the obvious inside the intermediate hosts, and something of the hosts may possibly not be especially trustworthy.

Link encryption is certainly invisible to an individual. The encryption will become a transmission assistance performed by way of a low-level network standard protocol layer, exactly like meaning routing or transmitting error detection. Figure 7.21 shows an average link encrypted concept, with all the shaded areas encrypted. Because a number of the data hyperlink header and truck is applied prior to the block will be encrypted, section of all of those blocks will be shaded. Because the message M will be managed at each coating, header and management information is included on the transmitting side and taken out on the getting side. Components encryption devices perform swiftly and reliably; in cases like this, link encryption is definitely invisible for the operating system in addition to for the operator.

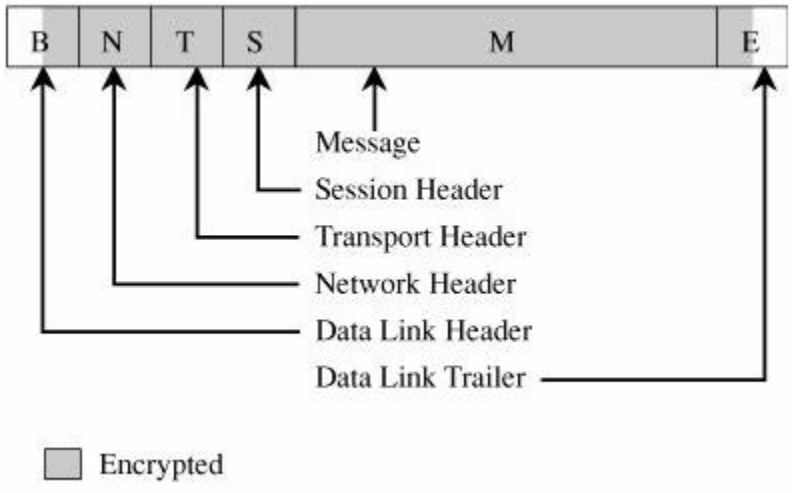


Figure 7.21. Message under Link Encryption.

Link encryption is particularly appropriate once the transmission line may be the point of best vulnerability. If all hosts on the network are moderately secure however the communications medium is usually shared with additional users or isn't secure, url encryption can be an easy control to utilize.

End-to-End Encryption

As its brand signifies, end-to-end encryption offers security in one end of any transmission to another. The encryption could be applied by way of a hardware device between your user as well as the host. Otherwise, the encryption can be carried out by software operating on the variety computer. In any case, the

encryption is conducted at the best levels (coating 7, application, or simply at level 6, display) of this OSI unit. A style of end-to-end encryption is usually shown in Figure 7.22

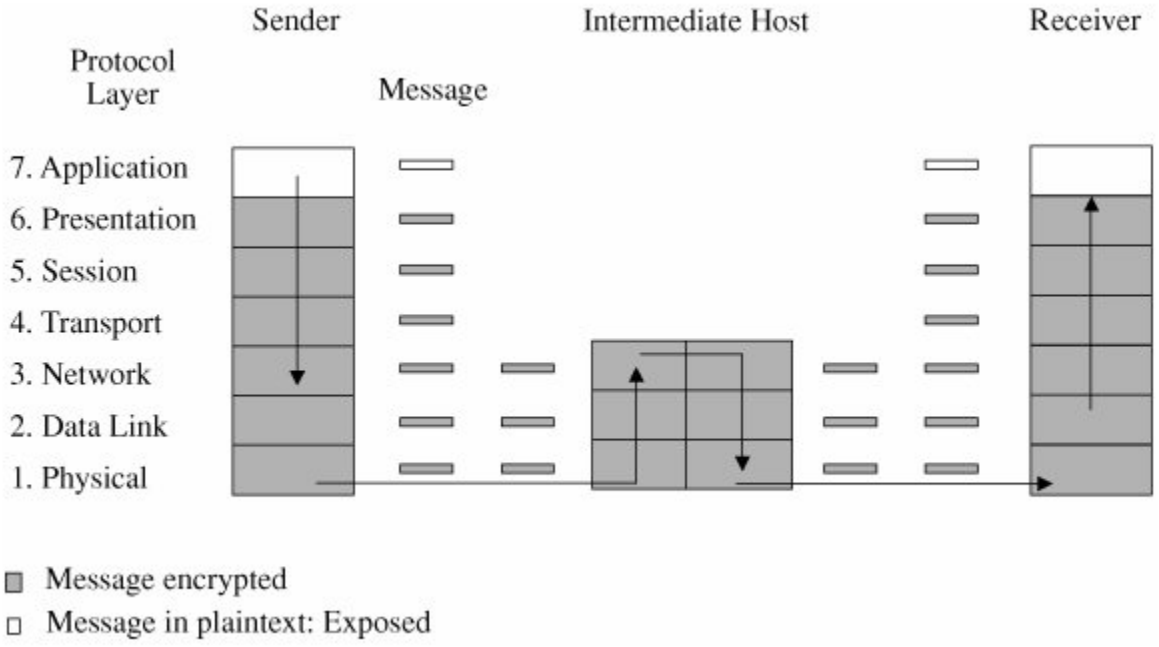


Figure 7.22. End-to-End Encryption.

Because the encryption precedes all of the routing and transmitting processing from the layer, the meaning is carried in encrypted variety throughout the community. The encryption addresses prospective imperfections in lower levels in the move model. In case a lower part should neglect to preserve stability and reveal information it has acquired, the data's confidentiality isn't endangered. Figure 7.23 shows an average information with end-to-end encryption, once again along with the encrypted discipline shaded.

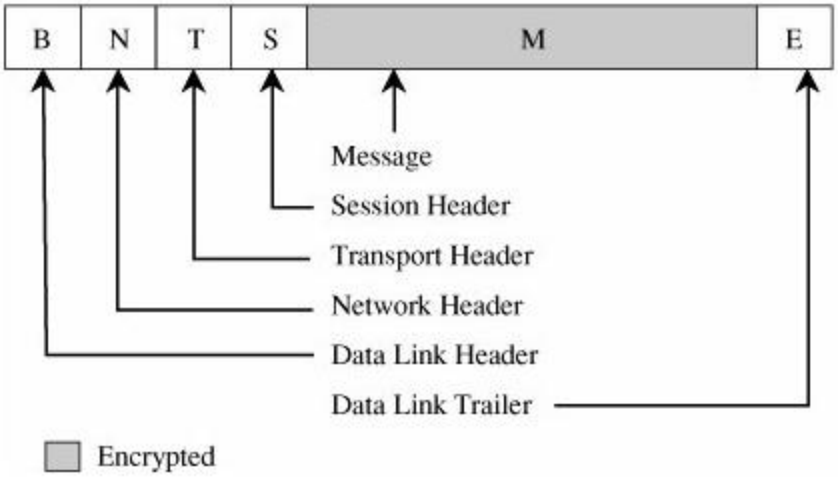


Figure 7.23. End-to-End Encrypted Message.

When end-to-end encryption can be used, messages dispatched through many hosts are guarded. The data articles of the information continues to be encrypted, as displayed in Number 7-24, plus the message is definitely encrypted (shielded against disclosure) during transit. Therefore, despite the fact that a note must go through possibly insecure nodes (such as for example C through G) on the road between A and B, the information is guarded against disclosure during transit.

Comparison of Encryption Methods

Simply encrypting a note is not definite assurance that you won't be disclosed during or after transmitting. In most cases, however, the effectiveness of encryption is sufficient protection, taking into consideration the odds of the interceptor's splitting the encryption as well as the timeliness of this message. Much like many areas of security, we should balance the effectiveness of protection with the probability of attack. (Become familiar with more about controlling these dangers in Section 8.)

With website link encryption, encryption is certainly invoked for several transmissions along a specific link. Typically, confirmed host has only 1 link right into a network, and therefore all network visitors initiated on that web host will undoubtedly be encrypted by that web host. But this encryption structure implies that almost every other host acquiring these communications must have a very

cryptographic center to decrypt the communications. In addition, all hosts must discuss keys. A note may go through a number of intermediate hosts on the path to its final vacation spot. If the information is definitely encrypted along some back links of a community however, not others, then area of the benefit of encryption is shed. Therefore, hyperlink encryption is normally done on all hyperlinks of a system if it's performed in any way.

In comparison, end-to-end encryption is definitely put on "logical hyperlinks," that happen to be programs between two techniques, at a rate properly above the bodily path. Because the intermediate hosts along a transmitting path need not encrypt or decrypt a note, they will have no dependence on cryptographic facilities. Therefore, encryption can be used limited to those information and applications that it is required. Moreover, the encryption can be carried out with computer software, so we are able to put it selectively, one use at the same time or to one message inside a given application.

The selective benefit of end-to-end encryption can be a disadvantage concerning encryption tips. Under end-to-end encryption, there's a virtual cryptographic route between each couple of users. To supply proper security and safety, each couple of users should talk about a distinctive cryptographic key. The amount of keys required is usually thus add up to the amount of pairs of customers, that is n * (n - 1)/2 for n customers. This number raises rapidly because the number of consumers increases. Nevertheless, this matter assumes that sole key encryption can be used. With an open public key system, only 1 pair of tips is necessary per recipient.

As proven in Table 7-5, url encryption is definitely faster, better for an individual, and uses much less secrets. End-to-end encryption is usually more flexible, may be used selectively, is performed at an individual level, and will be included with the application form. Neither form is certainly right for several situations.

IDOL

Table 7-5. Comparison of Link and End-to-End Encryption.	
Link Encryption	**End-to-End Encryption**
Security within hosts	
Data exposed in sending host	Data encrypted in sending host
Data exposed in intermediate nodes	Data encrypted in intermediate nodes
Role of user	
Applied by sending host	Applied by sending process
Invisible to user	User applies encryption
Host maintains encryption	User must find algorithm
One facility for all users	User selects encryption
Typically done in hardware	Either software or hardware implementation
All or no data encrypted	User chooses to encrypt or not, for each data item
Implementation concerns	
Requires one key per host pair	Requires one key per user pair
Provides node authentication	Provides user authentication

In some instances, both types of encryption could be applied. A consumer who does not really trust the grade of the hyperlink encryption supplied by something can put on end-to-end encryption as well. Something administrator who's worried about the security associated with an end-to-end encryption plan applied by a credit card application program may also install a hyperlink encryption machine. If both encryptions are usually relatively quick, this duplication of security and safety has little unfavorable effect.

Virtual Exclusive Networks

Link encryption may be used to provide a network's consumers the sense they are on an exclusive network, even though it is section of a public community. Because of this, the approach is named a virtual exclusive system (or VPN).

Typically, physical safety measures and administrative security and safety are strong sufficient to protect transmitting in the perimeter of a network. Thus, the best exposure for your user is between your user's workstation or consumer as well as the perimeter of this host community or server.

A firewall can be an access device that rests between two systems or two system segments. It filter systems all traffic between your shielded or "inside" community and a not as much reliable or "outside" system or portion. (We analyze firewalls at length later in this particular chapter.)

Many firewalls may be used to carry out a VPN. Whenever a user initially establishes an interaction with all the firewall, an individual can demand a VPN period using the firewall. The user's consumer along with the firewall discuss a program encryption key, along with the firewall and your client subsequently apply that major to encrypt all site visitors between your two. In this manner, the larger system is restricted and then those given exceptional access from the VPN. Quite simply, it seems to an individual that the system is private, though it is not. While using VPN, we declare that the connection passes via an encrypted tunnel or tunnel. Establishment of the VPN is found in Figure7.25.

IDOL

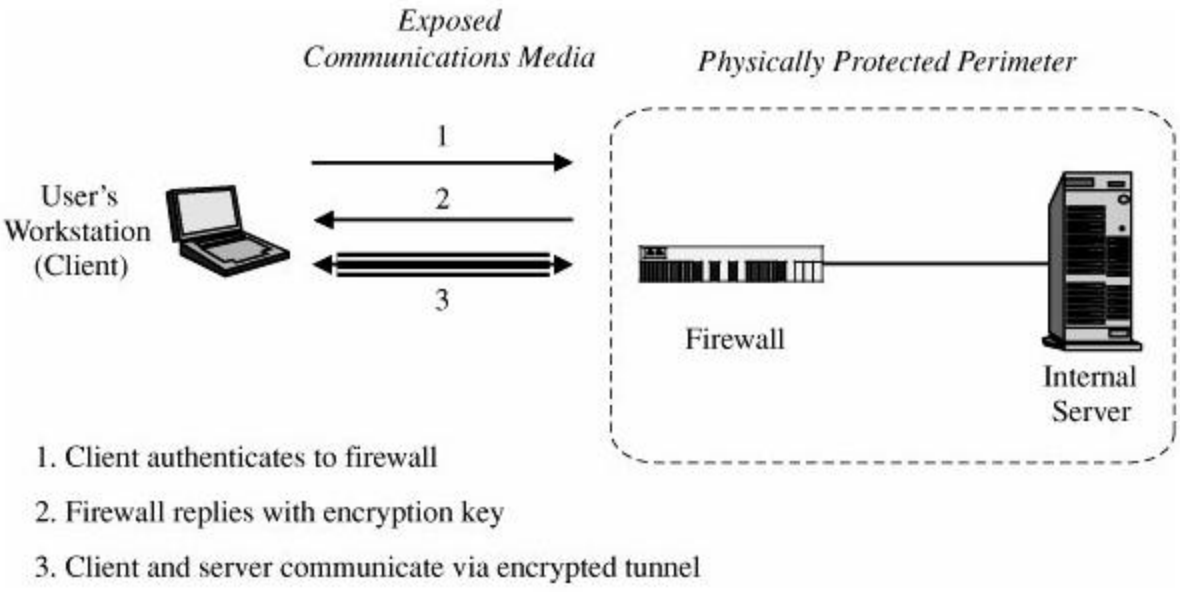


Figure 7.25. Establishing a Virtual Private Network.

Virtual private sites are created once the firewall interacts having an authentication service in the perimeter. The firewall may move user authentication information for the authentication server and, upon verification of this authenticated personal information, the firewall supplies the user with ideal security privileges. For instance, a known respected person, such as for example an employee or perhaps a system administrator, could be allowed to obtain resources unavailable to general customers. The firewall implements this gain access to control based on the VPN. A VPN with privileged gain access to is found in Figure 7-26. For the reason that amount, the firewall moves to the inner server the (privileged) personal information of Customer 2.

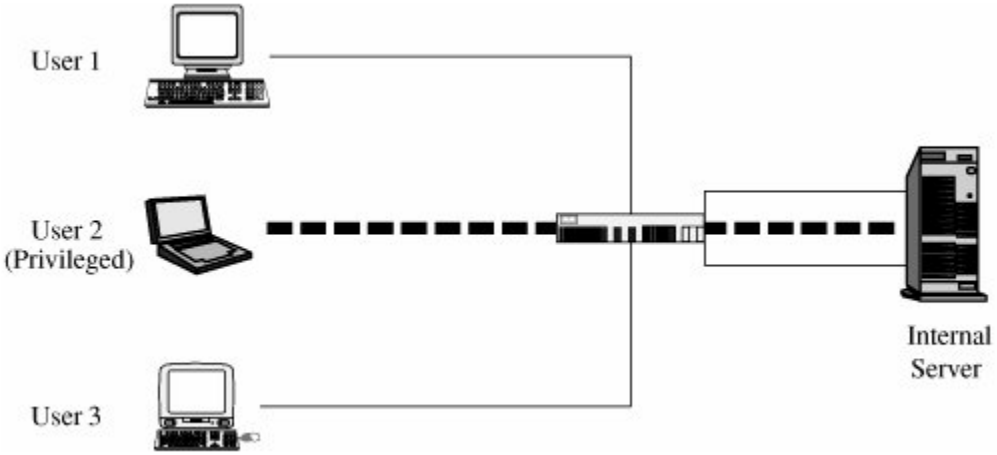


Figure 7.26. VPN to Allow Privileged Access

PKI and Certificates

A public key facilities, or PKI, is really a process intended to enable consumers to implement general public key cryptography, typically in a big (and sometimes, distributed) environment. PKI provides each user a couple of services, linked to identification and accessibility control, the following:

-. Create certificates associating a user's personal information which has a (people) cryptographic key

-. Hand out certificates from its database

-. Indication certificates, incorporating its credibility for the authenticity in the certificate

-. Confirm (or deny) a certificate is usually valid

-. Invalidate certificates for customers who no more are allowed gain access to or whose personal key is exposed

PKI is frequently regarded as a standard, however in fact this is a set of guidelines, products, and techniques that keep some area for interpretation. The insurance policies define the guidelines under that your cryptographic techniques should operate. Specifically, the policies identify the way to handle keys and beneficial information and how exactly to match degree of control to degree of risk. The

techniques dictate the way the keys ought to be generated, handled, and used. Eventually, the products really implement the guidelines, and they create, store, and handle the keys.

PKI creates entities, named certificate regulators that employ the PKI plan on certificates. The overall idea is a certificate authority is certainly trusted, so customers can delegate the development, issuance, approval, and revocation of certificates for the authority, much as you would work with a trusted bouncer to permit only some individuals to get into a constrained nightclub. The precise actions of a certificate authority are the following:

- managing public key element certificates because of their experience of living cycle

- issuing certificates by binding a user's or system's personality to a general public key with an electronic signature

- arranging expiration schedules for certificates

- guaranteeing that certificates happen to be revoked when important by posting certificate revocation lists

The functions of any certificate authority can be carried out in-house or by way of a commercial service or perhaps a trusted alternative party.

PKI also entails a registration specialist that acts being a program between an individual including a certificate specialist. The registration expert catches and authenticates the individuality of an end user and submits a certificate get to the correct certificate authority. In such a sense, the sign up authority is similar to the U.S. Postal Support; the postal program acts being an agent of this U.S. STATE DEPT. make it possible for U.S. people to acquire passports (public U.S. authentication) by giving the appropriate types, verifying identification, and requesting the specific passport (comparable to a certificates) from the correct passport-issuing workplace (the certificate expert). Much like passports, the grade of registration authority can determine the amount of trust that may be put into the certificates which are issued. PKI works with most naturally in a very hierarchically planned, centrally controlled company, like a government agency.

IDOL

Most PKI operations work with certificates that bind id to an integral. But research has been done to broaden the idea of certificate into a broader characterization of qualifications. For instance, Credit Cards Company could be interested in verifying your fiscal position than your id; a PKI program may include a certificate that's predicated on binding the fiscal status with an integral. THE EASY Distributed Security System (SDSI) takes this process, including identification certificates, group account certificates, and name-binding certificates. Around this writing, you can find drafts of two associated requirements: ANSI typical X9.45 and the easy Public Key Facilities (SPKI); the second option has just a set of prerequisites plus a certificate format.

PKI is near but not but a mature procedure. Many issues should be resolved, specifically since PKI offers yet to get integrated commercially on a big scale. Table 7-6 lists more than a few issues for being addressed once we find out about PKI. Even so, some things have grown to be clear. Very first, the certificate power should be accredited and confirmed by an unbiased entire body. The certificate authority's personal key ought to be stashed in a tamper-resistant safety measures module. Then, usage of the qualification and registration specialists should be firmly controlled, through strong individual authentication such as for example smart cards

IDOL

Issue	Questions
Flexibility	How do we implement interoperability and stay consistent with other PKI implementations?
	Open, standard interfaces?
	Compatible security policies?
	How do we register certificates?
	Face-to-face, e-mail, web, network?
	Single or batch (e.g., national identity cards, bank cards)?
Ease of use	How do we train people to implement, use, maintain PKI?
	How do we configure and integrate PKI?
	How do we incorporate new users?
	How do we do backup and disaster recovery?
Support for security policy	How does PKI implement an organization's security policy?
	Who has which responsibilities?
Scalability	How do we add more users?
	Add more applications?
	Add more certificate authorities?
	Add more registration authorities?
	How do we expand certificate types?
	How do we expand registration mechanisms?

Table 7-6. Issues Relating to PKI.

The security involved with guarding the certificates includes administrative procedures. For instance, several operator ought to be necessary to authorize certification demands. Controls ought to be set up to find hackers preventing them from issuing bogus certificate demands. These settings might include

IDOL

electronic signatures and tough encryption. Eventually, a safe audit trail is essential for reconstructing certificate data should the technique fail and then for recovering in case a hacking attack will in fact corrupt the authentication method.

SSH Encryption

SSH (secure shell) is really a pair of methods (versions 1 and 2), formerly identified for UNIX but additionally available under Glass windows 2000, that delivers an authenticated and encrypted way to the shell or operating-system demand interpreter. Both SSH variations replace UNIX resources such as for example Telnet, rlogin, and rsh for remote control access. SSH safeguards against spoofing assaults and changes of info in communication.

The SSH process includes negotiation between native and remote websites for encryption algorithm (for instance, DES, Thought, AES) and authentication (adding public main and Kerberos).

SSL Encryption

The SSL (Secure Sockets Part) protocol was initially originally created by Netscape to safeguard connection between a browser and server. Additionally it is known right now as TLS, for carry layer safety measures. SSL interfaces between programs (such as for example browsers) plus the TCP/IP protocols to supply server authentication, optional customer authentication, and an encrypted marketing communications channel between consumer and server. Customer and server work out a mutually recognized collection of encryption for period encryption and hashing; prospects include things like triple DES and SHA1, or RC4 using a 128-bit essential and MD5.

To utilize SSL, your client demands an SSL time. The server responds using its public key certification so the client can establish the authenticity of this server. Your client returns section of a symmetric time key encrypted beneath the server's public main. Both server and customer compute the treatment key, and they turn to encrypted conversation, using the provided session key.

The protocol is easy but effective, which is the most trusted secure communication standard protocol on the net. However, understand that SSL protects simply from the client's internet browser for the server's decryption stage (that is often and then the server's firewall or, somewhat stronger, for the

computer that works the web request). Data will be exposed in the user's keyboard towards the browser and through the entire recipient's company. Azure Gem Security is rolling out a product named LocalSSL that encrypts info after it's been typed before operating system offers it for the client's browser, hence thwarting any key logging Trojan horses that has been implanted within the user's computer system to show everything an individual types.

IPSec

As noted in the past, the address place online is running out there. As names of domain and apparatus proliferate, the initial, 30-year-old, 32-little address framework of the web is filling. A new construction, named IPv6 (version 6 on the IP protocol collection), solves the addressing trouble. This restructuring in addition offered a fantastic opportunity for the web Engineering Task Push (IETF) to handle serious security prerequisites.

As part of the IPv6 collection, the IETF used IPSec, or the IP Safety Protocol Suite. Made to address requisite shortcomings such as for example being at the mercy of spoofing, eavesdropping, and procedure hijacking, the IPSec process defines a typical means for coping with encrypted information. IPSec is applied in the IP layer, so that it affects all levels above it, specifically TCP and UDP. As a result, IPSec needs no adjustment to the prevailing large numbers of TCP and UDP practices.

IPSec is rather much like SSL, for the reason that it facilitates authentication and confidentiality in a manner that will not necessitate significant transformation either above it (in software) or below it (within the TCP practices). Like SSL, it had been designed to get independent of particular cryptographic protocols also to permit the two communicating functions to acknowledge a mutually reinforced set of methods.

The foundation of IPSec is usually what is referred to as a security relationship, which is fundamentally the set of stability parameters for just a secured communication route. It is about much like an SSL program. A security organization includes

- Encryption algorithm and function (for instance, DES in block-chaining function)

- Encryption key
- Encryption parameters, like the initialization vector
- Authentication standard protocol and key
- Lifespan with the association, allowing long-running sessions to choose a fresh cryptographic key normally as needed
- Target of the contrary finish of association

Sensitivity degree of protected information (usable for labeled data)

A host, like a network server or perhaps a firewall, may have several security organizations in place for concurrent marketing communications with different remote control hosts. A safety association is determined by a security and safety parameter catalog (SPI), a info element that's basically a pointer right into a table of safety associations.

The fundamental info set ups of IPSec will be the AH (authentication header) along with the ESP (encapsulated stability payload). The ESP replaces (includes) the traditional TCP header and info part of a packet, as found in Figure 7.27. The real header and truck depend on the info link and actual layer communications moderate, such as for example Ethernet.

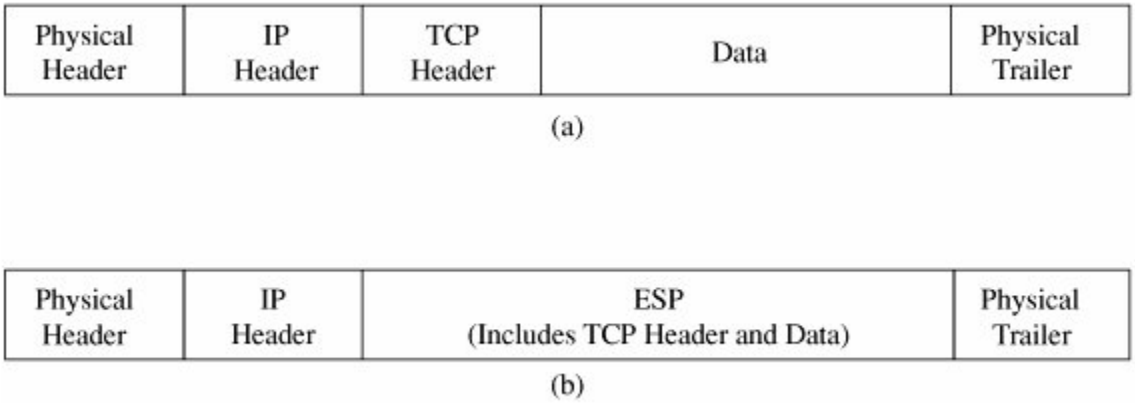


Figure 7.27. Packets: (a) Conventional Packet; (b) IPSec Packet.

The ESP has both an authenticated part and an encrypted section, as proven in Figure 7.28. The series number will be incremented by one for every packet

carried to exactly the same address utilizing the similar SPI, to preclude packet replay episodes

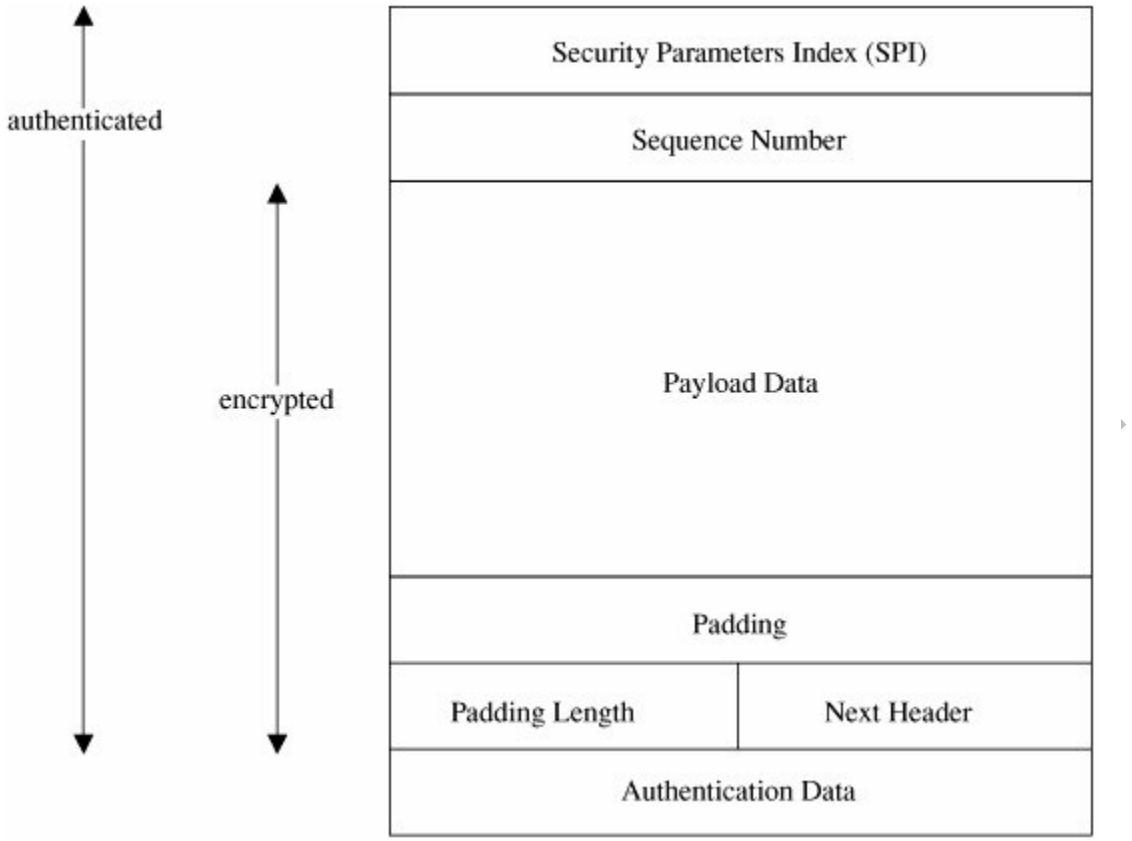


Figure 7.28. Encapsulated Security Packet.

As with just about all cryptographic software, the critical component is key administration. IPSec addresses this want with ISAKMP or Internet Security Association Key Management Protocol. Like SSL, ISAKMP needs that a distinctive key be developed for each protection organization. The ISAKMP standard protocol is simple, versatile, and scalable. In IPSec, ISAKMP is usually carried out through IKE or ISAKMP major exchange. IKE offers a way to acknowledge and manage practices, algorithms, and tips. For key trade between unrelated events IKE makes use of the DiffieHellman design). In DiffieHellman, each one of the two parties , X and Y, chooses a big prime and send a large number g raised to the power of the prime to the other. That's, X delivers g^x and Y delivers g^y. They both increase what they obtain to the energy they maintained: Y raises g^x to $(gx)^y$ and X raises g^y to

$(gy)^x$, that happen to be both the very same; voil?, they promote a magic formula $(gx)^y = (gy)^x$. (The computation is usually slightly more difficult, being done in a very finite discipline mod(n), thus an attacker cannot point the secret quickly.) Making use of their shared secret, both parties now change identities and certificates to authenticate those identities. Eventually, they derive a discussed cryptographic important and get into a security connection.

The key swap is very successful: The change can be completed in two announcements, having an optional two extra communications for authentication. Because this can be a public key technique, only two secrets are needed for every single couple of communicating get-togethers. IKE offers submodes for authentication (initiation) and then for establishing new tips in a prevailing security association.

IPSec can create cryptographic sessions numerous purposes, incorporating VPNs, software, and lower-level community management (such as for example routing). The practices of IPSec have already been published and thoroughly scrutinized. Focus on the protocols started in 1992. These were first publicized in 1995, plus they had been finalized in 1998 (RFCs 24012409).

Signed Code

As we have observed, someone can put malicious active program code on an internet site for being downloaded by unsuspecting consumers. Running while using opportunity of whoever downloading it, such lively code can perform serious harm, from deleting data to giving e-mail text messages to fetching Trojan horses to executing understated and hard-to-detect mischief. Today's craze is to let applications and up-dates being downloaded from fundamental sites, therefore the risk of downloading it something malicious keeps growing.

A partial not complete approach to minimizing this risk is by using signed program code. A trustworthy alternative party appends an electronic signature to a bit of program code, supposedly connoting even more trustworthy program code. A signature framework in a very PKI really helps to validate the trademark.

Who might the reliable party get? A well-known maker will be recognizable like a program code signer. But what of the tiny and virtually undiscovered manufacturer of a tool driver or perhaps a code add-in? When the code vendor

can be unknown, it generally does not help that owner signs its program code; miscreants can article their own agreed upon code, too.

In March 2001, Verisign introduced it possessed erroneously released two code-signing certificates beneath the title of Microsoft Corp. to somebody who purported to bebut was initially nota Microsoft staff. These certificates had been in circulation for nearly two months prior to the error was found. Even with Verisign found the mistake and canceled the certificates, a person would understand the certificates have been revoked just by verifying Verisign's list. A lot of people would not issue a program code download agreed upon by Microsoft

Encrypted E-mail

An electronic email message is similar to the back of your post greeting card. The mail provider (and everyone inside the postal technique through whose palms the card moves) can go through not only the address but additionally everything within the message field. To safeguard the privacy in the meaning and routing info, we can apply encryption to safeguard the confidentiality on the message as well as perhaps its integrity.

As we have observed in several various other software, the encryption may be the easy part; major management may be the more difficult problem. The two dominating approaches to essential management will be the usage of a hierarchical, certificate-based PKI option for key change and the usage of a set, individual-to-individual exchange technique. The hierarchical approach is named S/MIME and is utilized by many professional mail-handling programs, such as for example Microsoft Swap or Eudora. The average person method is named PGP and is really a industrial add-on. We appear more cautiously at encrypted e-mail in the later portion of this chapter.

Content Integrity

Content integrity arrives as an additional benefit with cryptography. No-one can change encrypted files in a significant way without bursting the encryption. This will not say, on the other hand, that encrypted info cannot be changed. Changing also one little bit of an encrypted files stream affects the effect after decryption,

typically in a manner that very seriously alters the producing plaintext. We have to consider three prospective threats:

- Malicious adjustment that changes articles in a important way
- Destructive or nonmalicious changes that changes information in a manner that is not always meaningful
- nonmalicious changes that changes information in a manner that will never be detected

Encryption addresses the initial of these hazards very effectively. To handle the others, we are able to use other handles.

Error Correcting Codes

We can apply error diagnosis and error modification codes to protect against modification in the transmission. The rules are their brands imply: Error recognition codes find when one has took place, and error modification codes can in fact correct mistakes without needing retransmission of the initial message. The mistake code is sent combined with the original data, therefore the receiver can recomputed the mistake code and check out whether the acquired result fits the expected benefit.

The simplest mistake detection code is really a parity check. A supplementary bit is put into an existing band of data bits based on their amount or a special OR. Both forms of parity are known as even and unusual. With possibly parity the excess bit is certainly 0 if the sum of the data parts is actually and 1 in case the sum is unusual; that's, the parity touch is set so the amount of all data pieces in addition to the parity bit is definitely even. Strange parity may be the similar except the amount is odd. For instance, the data flow 01101101 could have a straight parity little bit of 1 (and a strange parity little bit of 0) because 0+1+1+0+1+1+0+1 = 5 + 1 = 6 (or 5 + 0 = 5 for strange parity). A parity little can show you the changes of an individual bit. Nevertheless, parity will not discover two-bit errors cases where two parts in an organization are changed. That's, the usage of a parity touch depends on the assumption that single-bit problems will happen infrequently, so it's most unlikely that two pieces would be evolved.

Parity signals just that a touch has been improved; it generally does not identify which touch has been improved.

There are some other kinds of mistake detection codes, such as for example hash rules and Huffman rules. A number of the more complex rules can identify multiple-bit mistakes (several bits changed inside a data party) and could have the ability to pinpoint which pieces have been improved.

Parity and easy error recognition and correction rules are accustomed to detect nonmalicious adjustments in situations where there could be faulty transmission devices, communications noises and disturbance, or other resources of spurious modifications to data.

Cryptographic Checksum

Malicious modification should be handled in a manner that avoids the attacker from changing the error diagnosis mechanism along with the data parts themselves. One method to do this is by using a method that shrinks and changes the data, based on the value of the info bits.

To observe how such a technique might work, think of an error diagnosis code as being a many-to-one transformation. That's, any error recognition code decreases a stop of files to an inferior digest whose benefit depends upon each bit within the block. The percentage of lowering (that's, the percentage of original measurement of the stop to transformed measurement) pertains to the code's success in detecting problems. If a program code minimizes an 8-little bit data block into a 1-bit result, after that 1 / 2 of the 28 suggestions values chart to 0 and one half to at least one 1, supposing a uniform circulation of outputs. Quite simply, you can find 28/2 = 27 = 128 diverse bit patterns that produce exactly the same 1-bit consequence. The much less inputs that chart to a specific output, the much less methods the attacker can transform an input worth without impacting on its output. So, a 1-little result is as well weak for most applications. In the event the output is usually three bits rather than one, subsequently each output final result originates from 28/23 or 25 = 32 inputs. Small amount of inputs to confirmed output is essential for blocking destructive modification.

A cryptographic checksum (oftentimes called a note digest) is really a cryptographic work that creates a checksum. The cryptography avoids the

attacker from adjusting the data stop (the plaintext) and in addition modifying the checksum benefit (the ciphertext) to complement. Two major employs of cryptographic checksums happen to be code tamper security and information integrity safety in transit. For program code protection, something administrator computes the checksum of every program record on something and then in the future computes fresh checksums and compares the worth. Because executable program code usually will not modify, the administrator can identify unanticipated improvements from, for instance, malicious code episodes. Likewise, a checksum on info in communication recognizes data which have been changed in transmitting, maliciously or unintentionally.

Strong Authentication

As we have observed in prior chapters, os's and database supervision methods enforce a safety measures insurance policy that specifies whowhich folks, groups, subjects can admittance which methods and objects. Middle to that coverage is authentication: realizing and being guaranteed of the correctness of identities.

Networked environments want authentication, too. Within the network case, even so, authentication could be more difficult to accomplish securely due to the chance for eavesdropping and wiretapping, which are less prevalent in nonnetworked surroundings. Also, both edges of an interaction might need to be authenticated to one another: Before you decide to send your security password across a community, you intend to know that you're really conversing with the remote control host you anticipate. Lampson provides the issue of authentication in autonomous, sent out systems; the true problem, he highlights, is how exactly to develop have faith in of community entities with that you've no basis to get a relationship. Why don't we look more meticulously at authentication strategies appropriate for used in networks?

One-Time Password

The wiretap danger means that a password could possibly be intercepted from the user who gets into a security password across an unprotected community. A one-time security password can protect from wiretapping and spoofing of the remote host.

IDOL

As a names indicates, a one-time security password is wonderful for one only use. To observe how it works, think about the easiest case, where the user and number both get access to identical listings of passwords, just like the one-time pad for cryptography from chapter 2. An individual would enter the initial password for the initial login, another one for another login, etc. So long as the password listings remained secret so when long as nobody could suppose one security password from another, a security password attained through wiretapping will be useless. However, much like the one-time cryptographic pads, individuals have trouble preserving these password listings.

To cope with this problem, we can easily use a password token, a device that creates a password that is usually unpredictable but that may be validated within the getting end. The simplest kind of password token is the synchronous one, such because the SecurID device by RSA Security, Inc. This kind of device displays a randomly number, generating a fresh number every minute. Each and every user is issued some sort of different device (that creates a different random range sequence). The user scans the amount from the device's display and types that in as being a one-time security password. The computer for the acquiring end executes the protocol to generate the security password appropriate for the existing minute; in case the user's pass word matches the one calculated remotely, the user will be authenticated. Because the products could get out associated with alignment if one time clock runs slightly faster as compared to the other, these gadgets use fairly natural regulations to account for small drift.

What are advantages plus disadvantages of this strategy? First, it is simple to use. It generally counters the possibility involving a wiretapper reusing the password. Which has a strong password-generating algorithm, it truly is immune in order to spoofing. Nevertheless, the technique fails if the consumer loses the generating unit or, worse, if the particular device falls into a good attacker's hands. Because the new password is created only once a moment, right now there is a small (one-minute) window of vulnerability in the course of which an eavesdropper can certainly reuse an intercepted username and password.

ChallengeResponse Systems

To counter-top the loss and recycling problems, a more advanced one-time password scheme employs challenge and response, once we first studied in Phase

4. A challenge plus response device looks want a simple pocket online car loan calculator. The user first authenticates to the device, normally using a PIN. Typically the remote system sends some sort of random number, called the particular "challenge," that this end user enters into the unit. The device responds to be able to that number with one more number, which the user after that transmits towards the system.

Typically the system prompts an individual together with a new challenge regarding each use. Thus, this particular device eliminates the little windowpane of vulnerability by which the user could reuse the time-sensitive authenticator. An electrical generator that falls in to the incorrect hands is useless with no the PIN. However, an individual must always have the particular response generator to record in, and a busted device denies service in order to an individual. Finally, these gadgets do not address the particular possibility of a dodgy remote host.

Digital Distributed Authentication

In the 1980s, Digital Equipment Corporation known the problem of seeking to authenticate nonhuman choices in a computing method. For instance, a method might retrieve a customer query, which after that will it reformats, perhaps boundaries, and submits to some sort of database manager. Both the particular database manager plus the issue processor want to become sure which a particular interaction channel is authentic among the two. Neither associated with these servers is working under the direct handle or supervision of a new human (although each procedure was, naturally, somehow started by a human). Individual forms of access handle are thus inappropriate.

Electronic created a simple structure in this requirement, effective towards the following threats:

- Impersonation of a server simply by a rogue process, regarding either of the a couple of servers involved in the particular authentication
- Interception or customization of data exchanged among servers
- Replay of a new previous authentication

The structures assumes that each machine has its own exclusive key and that the particular corresponding public key will be available to or kept by every other

approach that may need to set up an authenticated channel. In order to begin an authenticated conversation between servers A plus server B, A transmits a request to W, encrypted under B's open public key. B decrypts typically the request and replies together with a message encrypted below A's public key. To stop replay, A and N can append an unique number to the subject matter to get encrypted.

A plus B can establish the private channel by one particular of them choosing the encryption key (for some sort of secret key algorithm) plus sending it to typically the other within the authenticating communication. Once the authentication will be complete, all communication beneath that secret key can easily be assumed to become as secure as had been the original dual open public key exchange. To shield the privacy in the station, Gasser recommends a different cryptographic processor, such as some sort of key card, so that will private keys will never ever be exposed outside typically the processor.

Two implementation issues remain to become solved: (a) How can a possibly large number of community keys be distributed and even (b) how can typically the public keys be dispersed in a way of which ensures the secure products of a process along with the key? Digital acknowledged that a key hardware (perhaps with multiple replications) was necessary to disperse keys. The 2nd difficulty is definitely addressed with certificates in addition to a certification hierarchy, while described in Chapter two.

Both of these style decisions are to some sort of certain degree implied by simply the nature of typically the remaining portion of the protocol. A various approach was taken by simply Kerberos, as we notice in the following parts.

Kerberos

Kerberos is a system of which supports authentication in dispersed systems. Originally created to job with secret key security, Kerberos, in its newest version, uses public major technology to back up key change. The Kerberos system has been designed at Massachusetts Start of Technology.

Kerberos is usually used for authentication among intelligent processes, such since client-to-server tasks, or the user's workstation to some other hosts. Kerberos is centered on the idea of which a central server gives authenticated

tokens, called ticket, to requesting applications. The ticket is surely an unforgeable, nonreplayable, authenticated object. That will be, it is a protected data structure naming a good user and a services that user is authorized to obtain. In addition, it consists of a time value and several control information.

The very first step in using Kerberos is to establish the session with the Kerberos server, as shown within Figure 7.29. A wearer's workstation sends the customer's identity towards the Kerberos machine when a user firelogs in. The Kerberos machine verifies that the consumer is authorized. The Kerberos server sends two communications:

1. for the user's workstation, a period key element SG for used in communication along with the ticket-granting server (G) along with a ticket T_G for that ticket-granting server; S_G is certainly encrypted beneath the user's security password: $E(S_G + T_G, pw)$

In Kerberos variation 5, just S_G is usually encrypted; in Kerberos variation 4, both session key along with the ticket have been encrypted when delivered to an individual.

2. For the ticket-granting server, a backup of the treatment key S_G along with the identity of an individual (encrypted under an integral shared between your Kerberos server plus the ticket-granting server)

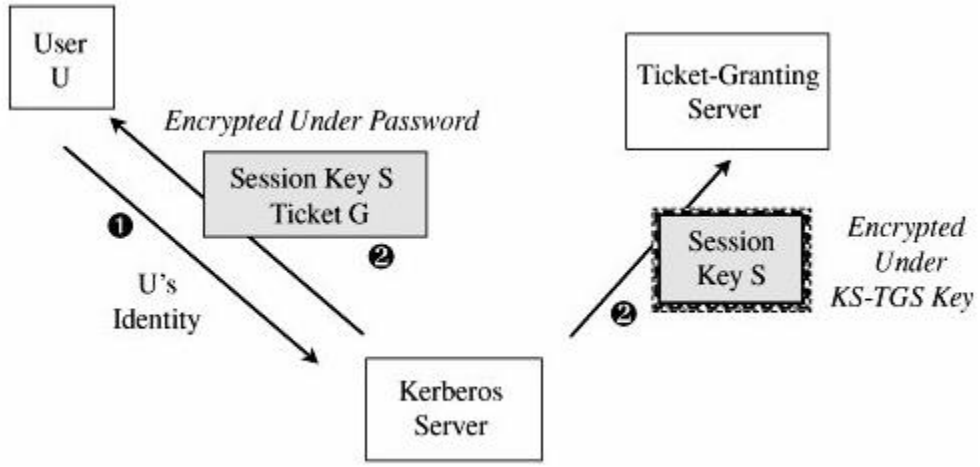


Figure 7.29. Initiating a Kerberos Session.

In the event the workstation can decrypt $E(S_G + T_G, pw)$ through the use of pw, the security password typed by an individual, then the individual has succeeded within an authentication with all the workstation.

Observe that passwords are kept on the Kerberos server, definitely not with the workstation, and that the user's security password did not need to be passed over the network, even yet in encrypted form. Retaining passwords centrally however, not passing them over the network is really a security advantage.

Next, an individual would want to exercise various other services from the distributed system, such as for example accessing a record. Using the key element S_G supplied by the Kerberos server, an individual U demands a ticket to gain access to file F from ticket-granting server. As demonstrated in Figure 7.30, following the ticket-granting server verifies U's accessibility permission, it results a ticket including a session main. The ticket has U's authenticated identification (inside the ticket U extracted from the Kerberos server), an recognition of F (the document to be reached), the entry rights (for instance, to learn), a period key S_F for any file server to utilize while conversing this record to U, and an expiration particular date for the solution. The ticket is usually encrypted under an integral shared exclusively between your ticket-granting servers along with the record server. This solution cannot be examine, revised, or forged by an individual U (or other people). The ticket-granting server must, consequently, provide U which has a duplicate of S_F, the Session key for that file server. Demands for usage of other solutions and servers happen to be handled similarly.

IDOL

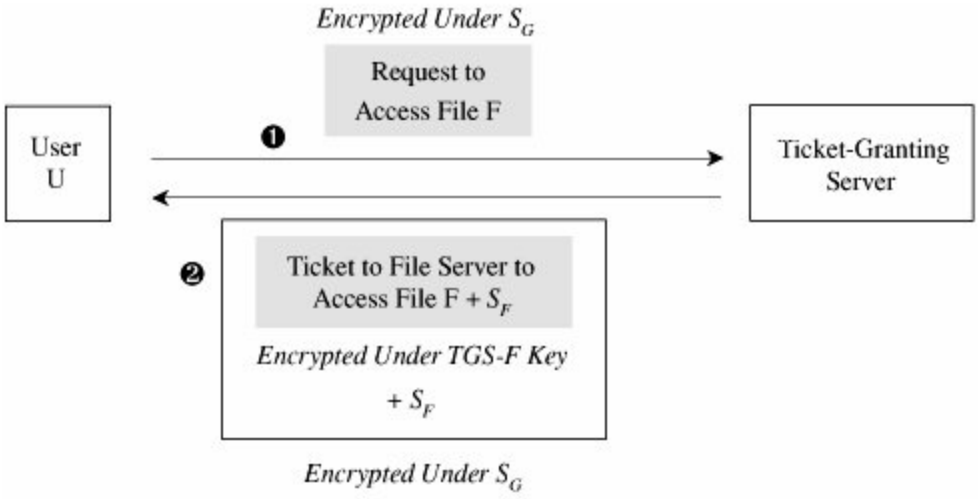


Figure 7.30. Obtaining a Ticket to Access a File.

Kerberos was diligently designed to hold up against attacks in spread environments:

- No passwords communicated for the network. As previously identified, a user's security password is stored simply in the Kerberos server. The user's security password is not dispatched in the user's workstation once the consumer initiates a period. (Certainly, a user's original password should be sent beyond your network, such as for example in a notice.)

- Cryptographic defense against spoofing. Each accessibility request will be mediated from the ticket-granting server, which understands the identity of this requester, in line with the authentication performed primarily because of the Kerberos server and on the truth that the user could present a submission encrypted under an integral that were encrypted beneath the user's password.

- Limited amount of validity. Each solution is given for a restricted time frame; the ticket includes a timestamp with which a acquiring server will decide the ticket's validity. In this manner, certain long-term problems, such as for example brute push cryptanalysis, will most likely be neutralized as the attacker won't have time to finished the attack.

- Timestamps to avoid replay problems. Kerberos requires efficient usage of a common clock. Each user's demand to some server will be stamped with enough time of the need. A server finding a request compares this time around to

the present moment and fulfills the question only if time is reasonably near to the current period. This time-checking helps prevent most replay episodes, because the attacker's presentation of this ticket will undoubtedly be delayed too much time.

- Mutual authentication. An individual of something can be guaranteed of any server's authenticity by asking for an authenticating reaction from server. An individual sends a solution to a server and delivers the server a submission encrypted beneath the session key for your server's assistance; the ticket plus the session key have been supplied by the ticket-granting server. The server can decrypt the solution only when it gets the unique primary it shares together with the ticket-granting server. In the ticket may be the session key that is the only indicates the server offers of decrypting the user's demand. When the server can go back to the user a note encrypted under this exact same session essential but comprising 1 + the user's timestamp, the server should be authentic. As a result of this shared authentication, a server can offer a unique route to an end user and an individual may not have to encrypt marketing communications on that route to ensure ongoing authenticity. Steering clear of encryption saves amount of time in the communication.

Kerberos isn't a perfect response to security challenges in distributed techniques.

- Kerberos requires constant availability of a reliable ticket-granting server. As the ticket-granting server may be the basis of gain access to handle and authentication, constant usage of that server is vital. Both consistency (components or software disappointment) and overall performance (ability and swiftness) problems should be addressed.

- Authenticity of machines requires a reliable relationship between your ticket-granting server and every server. The ticket-granting server must reveal a distinctive encryption key element with each "trustworthy" server. The ticket-granting server (or that server's individual administrator) should be convinced on the authenticity of this server. In an area environment, this amount of trust is definitely warranted. Within a widely distributed surroundings, an administrator at one web page can hardly ever justify rely upon the authenticity of machines at other internet sites.

- Kerberos requires well-timed transactions. To avoid replay strikes, Kerberos restricts the validity of any solution. A replay episode could succeed over validity, nevertheless. And setting the time fairly is difficult: Too much time increases the contact with replay problems, while too quick requires prompt individual actions and hazards providing an individual with a solution that won't come to be honored when provided into a server. In the same way, subverting a server's clock enables reuse of your expired ticket.

- A subverted workstation can conserve and afterwards replay individual passwords. This vulnerability is available in any technique where passwords, encryption secrets, or additional constant, sensitive data is entered inside the clear on the workstation that could be subverted.

- Security password guessing gets results. A user's original ticket is went back beneath the user's security password. An attacker can send a short authentication request for the Kerberos server and make an effort to decrypt the reply by guessing in the password.

- Kerberos will not scale very well. The architectural style of Kerberos, found in Figure 7-31, assumes one Kerberos server and something ticket-granting server, and also a collection of different servers, all of which shares a distinctive key using the ticket-granting server. Putting another ticket-granting server, for instance, to enhance effectiveness or consistency, would need duplicate keys or perhaps a second set for several servers. Duplication escalates the risk of subjection and complicates major updates, and next keys a lot more than double the task for every server to do something on a solution.

Kerberos is really a complete choice. All programs must work with Kerberos authentication and admittance control. Currently, several applications make use of Kerberos authentication, therefore integration of Kerberos into a preexisting environment requires adjustment of existing software, which is definitely not feasible.

IDOL

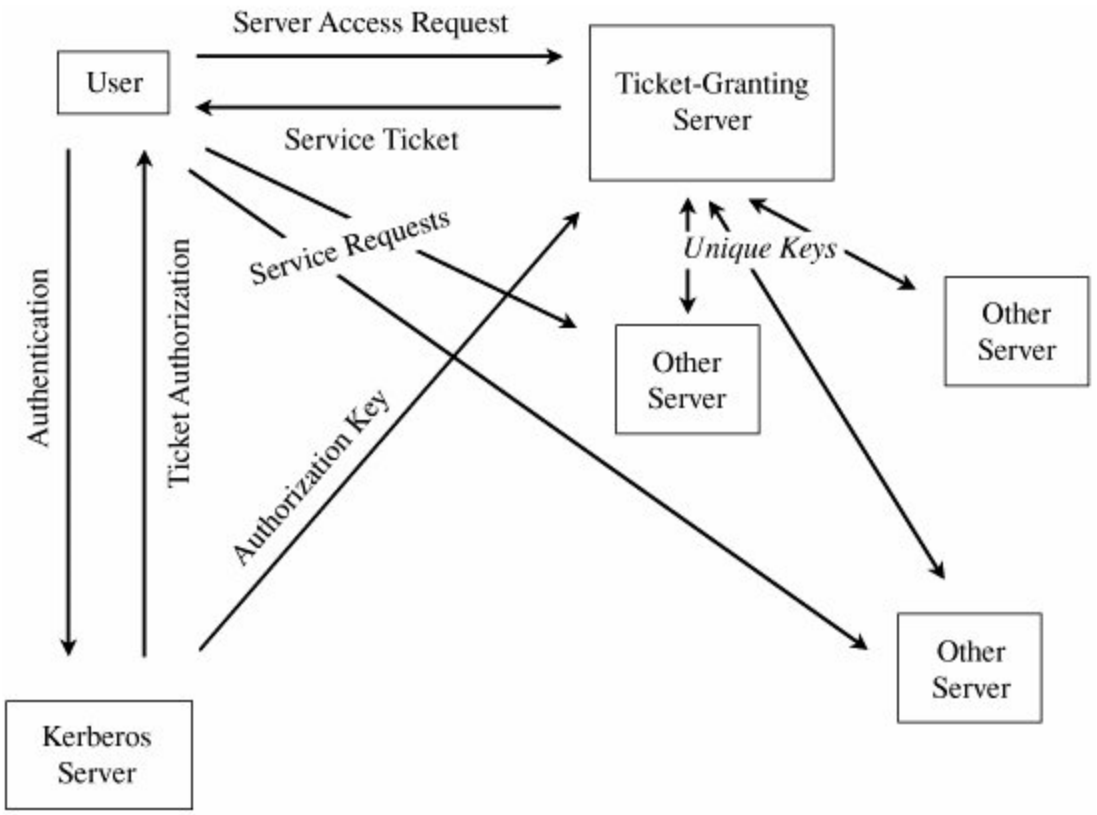


Figure 7.31. Access to Services and Servers in Kerberos.

Access Controls

Authentication handles the who of security and safety policy enforcement; entry adjustments enforce the what and exactly how.

ACLs on Routers

Routers carry out the major job of directing community site visitors either to subnetworks they handle or to some other routers for succeeding delivery to additional subnetworks. Routers switch outside IP addresses into interior Macintosh addresses of hosts on an area subnetwork.

Suppose a bunch has been spammed (flooded) with packets from the malicious rogue variety. Routers could be configured with entry control listings to deny usage of certain hosts from specific hosts. Therefore, a router could erase all packets having a source address from the rogue host along with a destination target of the prospective host.

This approach possesses three problems, even so. First of all, routers in large sites perform a large amount of work: They need to take care of every packet getting into and moving away from the network. Incorporating ACLs towards the router demands the router to assess every packet contrary to the ACLs. One ACL provides job, degrading the router's efficiency; as additional ACLs are added in, the router's efficiency may become undesirable. The second difficulty can be an efficiency matter: Due to the volume of do the job they do, routers are made to perform only important providers. Logging of action is usually not really done over a router due to the volume of site visitors and the efficiency charges logging would entail. With ACLs, it might be useful to understand how many packets have been being deleted, to learn if a specific ACL could possibly be removed (in that way improving overall performance). But without logging it really is impossible to learn whether an ACL has been used. Both of these problems together imply ACLs on routers will be most reliable against specific acknowledged threats but they shouldn't be used indiscriminately.

The final restriction on adding ACLs on routers considerations the nature with the risk. A router inspects simply source and location addresses. An attacker typically does not expose an actual supply address. To show you the real origin address will be equal to a lender robber's making his home target and a information of where he projects to retail outlet the stolen funds.

Because someone can simply forge any origin address over a UDP datagram, various attacks make use of UDP practices with false origin addresses so the attack can't be blocked easily by way of a router having an ACL. Router ACLs are of help only when the attacker delivers several datagrams with exactly the same forged source tackle.

In rule, a router is a superb point of admittance control since it grips every packet getting into and moving away from a subnetwork. In particular situations, largely for inner subnetworks, ACLs may be used effectively to limit certain traffic moves, for example, to make sure that only specific hosts (addresses) get access to an internal community management subnetwork. But also for large-scale, general visitors screening, routers will be less beneficial than firewalls.

Firewalls

A firewall does indeed the screening that's less befitting a router to accomplish. A router's major function is dealing with, whereas a firewall's main function can be filtering. Firewalls may also do auditing. A lot more essential, firewalls can analyze a whole packet's contents, like the data section, whereas a router can be involved only with supply and destination Apple pc and IP addresses. Because they're an extremely essential network security management, we analyze firewalls within an entire section afterwards in this section.

Wireless Security

Because wireless processing is so shown, it requires options to protect marketing communications between some type of computer (called your client) and a radio base place or access stage. Remembering that these communications happen to be on predefined stereo frequencies, you may expect an eavesdropping attacker to attempt to intercept and impersonate. Items to protect have found the access level, authenticating the remote control computer for the access level, and vice versa, and safeguarding the communication flow.

SSID

As described early on in this section, the Service Collection Identifier or SSID may be the identification of access point; this is a string as high as 32 characters. Definitely the SSIDs have to be unique in confirmed area to tell apart one wireless system from another. The factory-installed default for earlier versions of cordless access points had not been unique, such as for example "wireless," "tsunami" or "Linksys" (a brandname name); now virtually all factory defaults certainly are a serial number special to these devices.

Litigant and an admittance point take part in a handshake to find one another: Basically the client states, "I'm looking to hook up to access stage S" as well as the access point states, "I'm access stage S; hook up to me." The purchase of the two steps is essential. In what's called "open function," an entry point can constantly broadcast its elegance, indicating that it's open for the next phase in establishing a link. Open mode is really a poor security training since it advertises the label of a gain access to indicate which an attacker might add. "Closed" or "stealth method"

reverses the purchase of the process: Your client must send a sign seeking an entry point with a specific SSID prior to the access stage responds compared to that one query having an invitation for connecting.

But closed setting does not stop understanding of the SSID. The original exchange "searching for S," "I'm S" takes place in the apparent and can be acquired to anyone who runs on the sniffer to intercept cordless communications in collection. Therefore, anyone who sniffs the SSID can preserve the SSID (that is seldom changed used) to utilize later.

WEP

The second part of securing a radio communication involves usage of encryption. The initial 802.11 cellular regular relied upon a cryptographic process called wired equal personal privacy or WEP. WEP had been meant to deliver users privacy equal to that of a separate wire, that's, immunity to many eavesdropping and impersonation problems. WEP makes use of an encryption major shared between your client as well as the access level. To authenticate an end user, the access level sends an arbitrary number to your client, which the customer encrypts utilizing the shared key element and returns for the access point. In the future, your client and access level are authenticated and will communicate utilizing their shared encryption main. Several problems can be found with this apparently simple approach.

Very first, the WEP normal uses the 64- or 128-little encryption key. An individual enters the main element in any practical form, generally in hexadecimal or being an alphanumeric string that's converted to lots. Coming into 64 or 128 pieces in hex calls for choosing and keying in 16 or 32 icons correctly for your client and access level. And in addition, hex strings like C0DE C0DE.. (That is clearly a no between C and D) are normal. Passphrases are susceptible to a dictionary strike.

Even if the main element is strong, it certainly has a helpful length of simply 40 or 104 pieces due to the way it really is found in the algorithm. A brute pressure harm against a 40-little key succeeds swiftly. Even for that 104-bit version, imperfections inside the RC4 algorithm and its own use (notice [BOR01, FLU01, and ARB02]) beat WEP security. Different tools, you start with WEPCrack and AirSnort, enable an attacker to break a WEP encryption, typically in a minute. At

the 2005 discussion, the FBI exhibited the decrease with which a WEP-secured cellular session could be broken.

Therefore, in 2001 the IEEE begun design of a fresh authentication and encryption design for wireless. Regrettably, some wireless gadgets still available on the market allow simply the false protection of WEP.

WPA and WPA2

The choice to WEP is usually WiFi Protected Gain access to or WPA, authorised in 2003. The IEEE normal 802.11i is currently referred to as WPA2, accepted in 2004, and can be an expansion of WPA. So how exactly does WPA boost upon WEP?

First, WEP utilizes an encryption primary that's unchanged before user enters a fresh key at your client and access level. Cryptologists detest unchanging encryption tips because a set key provides attacker a great deal of ciphertext to attempt to analyze and the required time in which to investigate it. WPA includes a key change solution, called Temporal Key element Integrity System (TKIP), where the encryption essential is changed quickly on each packet.

Second, WEP utilizes the encryption major being an authenticator, albeit insecurely. WPA utilizes the extensible authentication standard protocol (EAP) where authentication can be carried out by security password, token, certificate, or some other mechanism. For little network (residence) consumers, this probably nevertheless means a distributed secret, that is not ideal. Consumers are inclined to selecting weak tips, such as brief numbers or go phrases at the mercy of a dictionary episode.

The encryption algorithm for WEP is certainly RC4, which includes cryptographic imperfections both in key element length and design and style . In WEP the initialization vector for RC4 is 24 pieces, a size thus little that collisions typically occur; furthermore, there is absolutely no test against initialization vector reuse. WPA2 provides AES just as one encryption algorithm (although RC4 can be still backed for compatibility causes).

WEP carries a 32-little bit integrity check independent from the info portion. But as the WEP encryption is usually at the mercy of cryptanalytic invasion, the integrity test was also subject matter, consequently an attacker could enhance

content as well as the corresponding check and never have to know the affiliated encryption essential. WPA carries a 64-tad integrity be sure is encrypted.

The setup process for WPA and WPA2 is a lot better quality than that for WEP. Installation for WPA consists of three protocol measures: authentication, a four-way handshake (to make sure that your client can produce cryptographic keys also to generate and mount secrets for both encryption and integrity on both comes to an end), and an optional class key element handshake (for multicast connection.) .

WPA and WPA2 deal with the security and safety deficiencies recognised in WEP create a strong circumstance for public key element cryptography in cordless sensor sites, and an identical argument could be made for various other wireless software (even though heavier computation requirements of public main encryption is really a limiting issue on wireless equipment with limited cpu capabilities.)

Alarms and Alerts

The logical watch of network safety looks like figure 7-32, where both a router as well as a firewall provide tiers of safeguard for the inner network. Now why don't we add yet another layer to the defense?

An intrusion recognition system is really a device that's placed in the protected community to watch what occurs inside the system. If an attacker goes by throughout the router and goes by throughout the firewall, an intrusion diagnosis system supplies the opportunity to find the attack at the start, happening, or after it includes occurred. Intrusion recognition systems switch on an alarm, that may take defensive measures.

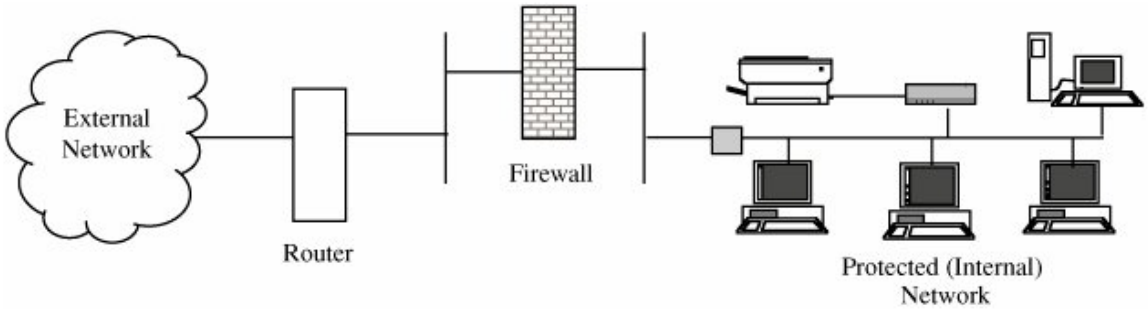


Figure 7-32. Layered Network Protection.

Honeypots

How will you get a mouse? You placed a capture with bait (foodstuff the mouse locates desirable) and get the mouse after it really is lured in to the trap. It is possible to catch some type of computer attacker exactly the same way.

In an exceedingly interesting publication, Cliff Stoll highlights the storyplot of luring and monitoring what of attacker. Cheswick and Bellovin inform a similar history. These two instances describe the usage of a honeypot: some type of computer system available to attackers.

You set up a honeypot for a number of reasons:

- To monitor what attackers do, in order to find out about new strikes (to enable you to improve your defenses against these fresh attacks)

- To lure an attacker to a location in which you might be able to find out enough to recognize and prevent the attacker

- To offer an desirable but diversionary playground, wanting the fact that attacker will depart your real program alone

A honeypot does not have any special features. This is a computer system or perhaps a network segment, packed with servers and equipment and data. It might be protected which has a firewall, although you need the attackers to possess some access. There could be some monitoring capabilities, done carefully so the monitoring isn't evident for the attacker.

The two complicated top features of a honeypot will be adding a believable, desirable false surroundings and confining and overseeing the attacker surreptitiously. Spitzner did extensive work acquiring and examining honeypots. He feels just like the attacker, figuring the particular attacker would want to see within an invaded laptop, but as McCarty [highlights, it will always be a contest between attacker and defender. Spitzner in addition tries to go a lot of his information off the prospective platform so the attacker will never be alert to the evaluation and definitely not have the ability to modify or remove the data compiled. Raynal talks about how to evaluate the data accumulated.

IDOL

Traffic Flow Security

So far, we've looked at adjustments that cover the most frequent network hazards: cryptography for eavesdropping, authentication options for impersonation, and intrusion recognition systems for strikes in progress, structures for structural imperfections. Earlier in such a chapter, we outlined threats, consisting of a risk of traffic movement inference. In case the attacker can identify an exceptional level of site visitors between two tips, the attacker may infer the positioning of a meeting about to take place.

The countermeasure to site visitors flow threats would be to disguise the visitors flow. One method to disguise traffic stream, albeit costly as well as perhaps crude, would be to ensure a reliable volume of visitors between two things. If site visitors between A and B can be encrypted so the attacker can discover only the amount of packets moving, A and B can consent to go away recognizable (in their mind) but meaningless encrypted site visitors. When A offers much to talk to B, you will see very few meaningless packets; when connection is light source, A will pad the visitor's stream numerous spurious packets.

A more sophisticated method of traffic flow safety is named onion routing. Look at a message that's covered in several layers, just like the layers of the onion. A really wants to send a note to B but doesn't desire anyone in or intercepting visitors on the system to learn A is conversing with B. Consequently A calls for the concept to B, wraps it within a bundle for D to send out to B. Then simply, A wraps that package deal in another program for C to deliver to D. Ultimately, A delivers this offer to C. This technique is proven in Body 7-33. The inner wrappings are encrypted under an integral befitting the intermediate receiver Receiving the bundle, C is aware of it originated from A, although C will not know in case a may be the originator or an intermediate level. C in that case unwraps the exterior layer and considers it ought to be delivered to D. At this time, C cannot recognize if D may be the final receiver or just an intermediary. C directs the communication to D, who unwraps another layer. D recognizes neither where in fact the package originally originated from nor where its ultimate destination is definitely. D forwards the program to B, its amazing recipient.

With this structure, any intermediate recipients those apart from the initial sender and amazing receiver know neither where in fact the package deal

originated nor where it'll find yourself. This scheme gives confidentiality of articles, source, vacation spot, and routing.

7. 4. Firewalls

What Is a Firewall?

A fire wall is a device that will filters all traffic among a protected or "inside" network and a significantly less trustworthy or "outside" system. Usually a firewall works on a dedicated system; since it is some sort of single point through which often traffic is channeled, efficiency is very important, which means non firewall functions really should not be done in the same machine. Just because a firewall is executable program code, an attacker could give up that code and carry out from the firewall's unit. Thus, the fewer parts of code on typically the device, the fewer equipment the attacker could have by simply compromising the firewall. Fire wall code usually runs about a proprietary or thoroughly minimized operating system.

Typically the purpose of a fire wall is to keep "bad" things outside a secured environment. To accomplish of which, firewalls implement a safety measures policy that may be specifically made to address what negative things might happen. Intended for instance, the policy may be to avoid any gain access to from outside (while nevertheless allowing visitors pass by the inside to typically the outside). Alternatively, the insurance plan might permit accesses simply from certain places, coming from certain users, or with regard to certain activities. Area of the obstacle of protecting a community with a firewall will be determining which security coverage meets the needs regarding the installation.

People within the firewall community (users, developers, and security experts) disagree about how some sort of firewall should work. Specifically, the community is broken down of a firewall's default conduct. We are able to describe the couple of schools of thought while "that which is certainly not expressly forbidden is permitted" (default permit) and "that which is not specially permitted is forbidden" (default deny). Users, always enthusiastic about new features, prefer typically the former. Security experts, depending on several decades associated with experience, strongly counsel the particular latter. An

administrator employing or configuring a fire wall must choose one associated with the two approaches, even though the administrator could expand the policy by establishing the firewall's parameters.

Design and style of Firewalls

Remember through Chapter 5 that the reference monitor must end up being

- always invoked
- tamperproof
- small and simple enough with regard to rigorous analysis

A fire wall is a special sort of reference monitor. By meticulously positioning a firewall inside a network, we can assure that all network has access to that we wish to command must pass through that. This restriction meets typically the "always invoked" condition. Some sort of firewall is typically effectively isolated, making it extremely immune to modification. Normally a firewall is integrated on a separate computer system, with direct connections just to the outside in addition to inside networks. This remoteness is expected to fulfill the "tamperproof" requirement. In addition to firewall designers highly suggest keeping the functionality involving the firewall simple.

Types of Firewalls

Firewalls include a wide range involving capabilities. Forms of firewalls incorporate

- packet filtering gateways or even screening routers
- stateful inspection firewalls
- application proxies
- guards
- personal firewalls

Each sort does different things; no person is necessarily "right" as well as the some others "wrong.” With this area, we examine each sort to see what that is,

how functions, plus what its strengths plus weaknesses are. In standard, screening routers are likely to put into action rather simplistic security guidelines, whereas guards and web proxy gateways have a more potent set of choices with regard to security policy. Simplicity inside a security policy will be not a bad point; the key question to request when choosing a kind of fire wall is what threats the installation needs to table.

Just because a firewall is a new type of host, that often is as pré-réglable as being a good-quality workstation. Although a screening router may be fairly primitive, the particular tendency is to number even routers on finish computers with operating methods because editors and additional programming tools assist throughout configuring and maintaining typically the router. However, firewall designers are minimalists: They consider to eliminate from the particular firewall all that is definitely not strictly necessary for that firewall's functionality. There is usually a great reason for this little constraint: to offer as small assistance as possible to some successful attacker. Thus, firewalls tend not to need user accounts so of which, for example, they have got no password file to be able to conceal. Indeed, the almost all desirable firewall is one particular that runs contentedly found in a back room; apart from periodic scanning of their audit logs, there will be seldom reason to feel it.

Packet Filtering Gateway

A packet filtering portal or screening router is certainly the simplest, and inside of some situations, the the majority of effective type of fire wall. A packet filtering entrance controls access to bouts based on packet address (source or destination) or special transport protocol type (such as HTTP web traffic). As described earlier found in this chapter, putting ACLs on routers may seriously impede their performance. Nevertheless a separate firewall at the rear of (on the local side) of the router might screen traffic before that reaches the protected community. Figure 7-34 shows some sort of packet filter that prevents access from (or to) addresses in one community; the filter allows HTTP traffic but blocks targeted traffic using the Telnet process.

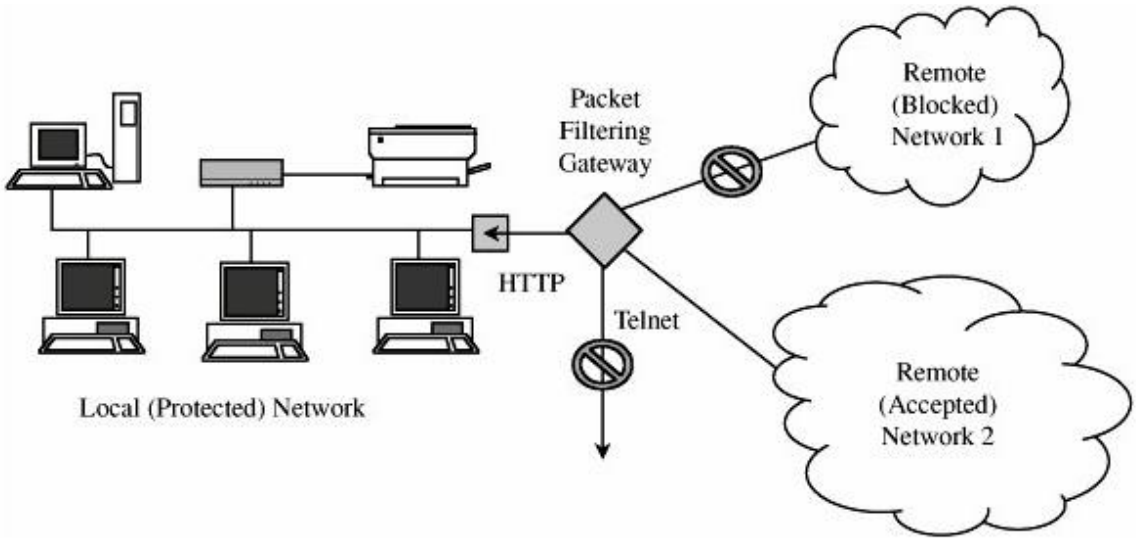


Figure 7.34. Packet Filter Blocking Addresses and Protocols.

For example, suppose a worldwide company has three LANs at three locations all over the world, as shown in **Figure** 7-35. In this illustration, the router has a couple of sides: inside and exterior. We admit the regional LAN is inside typically the router, and the 2 connections to distant LANs through wide area sites are on the outside the house. The company may want connection only among the 3 LANs of the business network. It could make use of a screening router in the LAN at a 100.24.4.0 to let in only communications most likely going towards the host at 100.24.4.0 in addition to to allow out simply communications addressed either in order to address 144.27.5.3 or 192.19.33.0.

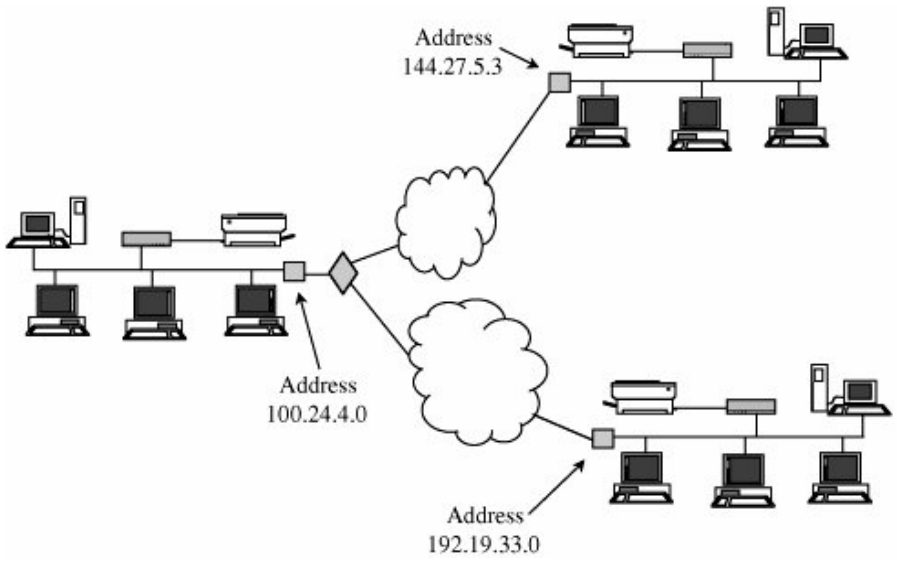


Figure 7-35. Three Connected LANs.

Packet filter systems do not "see inside" a packet; they stop or accept packets exclusively on the basis involving the IP addresses in addition to ports. Thus, any particulars in the packet's info field (for example, permitting certain Telnet commands whilst blocking other services) will be beyond the capability regarding a packet filter.

Packet filters is capable of doing the very crucial service of guaranteeing the validity of in addresses. Inside of hosts typically believe other on the inside hosts for all your reasons referred to as qualities of LANs. However the only way an internal host can recognize another inside coordinator is definitely by the tackle shown in the foundation field of a note. Supply addresses in packets could be forged, so an internal application might believe it was conversing with another number inside instead of another forger. A packet filtration sits between your inside community and the exterior net, so that it can know in case a packet from the exterior is forging an internal address, as proven in Figure 7-36. A verification packet filter may be configured to stop all packets from the exterior that said their source street address was an internal address. In such a case in point, the packet filtration blocks all packets saying ahead from any street address of the proper execution 100.50.25.x (but, needless to say, it permits in virtually any packets with location 100.50.25.x).

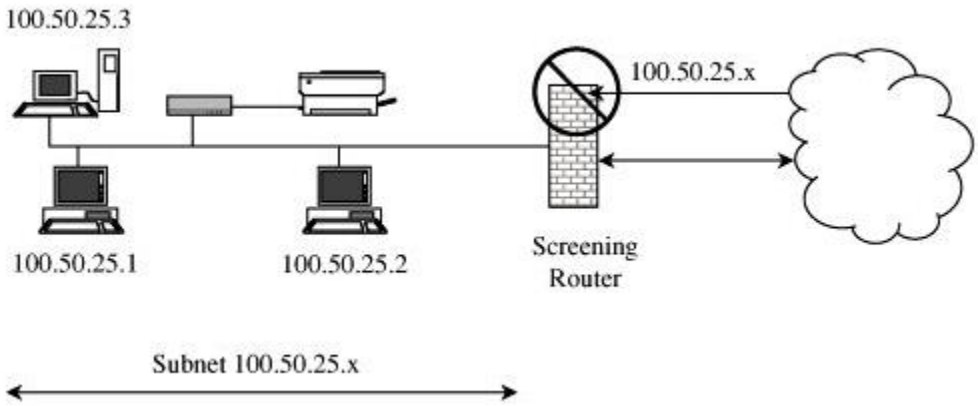


Figure 7-36. Filter Screening Outside Addresses.

The primary drawback of packet filtering routers is really a combination of ease and difficulty. The router's assessment is simplistic; to execute complex filtering, the filtering guidelines set must be very complete. A detailed regulations set will undoubtedly be complex and for that reason prone to problem. For example, obstructing all interface 23 visitors (Telnet) is easy and clear-cut. But if some Telnet visitors is usually to be allowed, each Ip from which it really is allowed should be specified in the guidelines; in this manner, the rule place can become pretty long.

Stateful Inspection Firewall

Filtering firewalls focus on packets individually, taking or rejecting each packet and shifting to another. They will have no idea of "state" or "context" in one packet to another. A Stateful Inspection Firewall maintains status information in one packet to some other in the suggestions stream.

One classic tactic utilized by attackers would be to break an assault into several packets by forcing some packets to possess very short measures in order that a firewall cannot discover the signature of your attack break up across several packets. (Understand that while using TCP methods, packets can get to any order, along with the protocol suite is in charge of reassembling the packet supply in proper buy before transferring it along to the application form.) A Stateful

Inspection Firewall would trail the series of packets and disorders in one packet to some other to thwart this attack.

Application Proxy

Packet filters seem only in the headers of packets, definitely not at the info in the packets. Thus, a packet filtration system would go away anything to port 25 , supposing its screening guidelines allow inbound contacts to that slot. But applications are usually complex and quite often contain errors. More serious, applications (like the e-mail delivery adviser) often work with respect to all users, so that they require privileges of most users (for instance, to store inbound mail messages in order that inside consumers can study them). A flawed software, jogging with all customers' privileges, could cause much damage.

A credit card application proxy gateway, also known as a bastion coordinator, is really a firewall that simulates the (appropriate) ramifications of an application so the application receives sole requests to do something effectively. A proxy gateway is really a two-headed gadget: It appears to the within as if it's the outside (location) relationship, while to the exterior it responds in the same way the insider would.

A credit card application proxy works pseudo applications. For example, when e-mail is used in a spot, a sending method at one internet site and also a receiving process in the destination communicate by way of a standard protocol that establishes the legitimacy of a mail transfer and actually exchanges the mail information. The standard protocol between sender and vacation spot is carefully identified. A proxy gateway fundamentally intrudes in the center of this protocol change, seeming such as a destination in connection with all the sender that's beyond your firewall, and seeming just like the sender in connection with the true destination inside. The proxy in the Centre has the possibility to screen the email transfer, making certain only suitable e-mail protocol instructions are delivered to the destination.

For example of program proxying, think about the FTP (record transfer) protocol. Certain protocol orders fetch (receive) files from the remote location, retail outlet (set) documents onto a remote control host, list documents (ls) in a very directory on the remote web host, and position the procedure (disc) at a specific stage in a listing tree on the remote variety. Some administrators should permit makes but

block places, and to listing only certain documents or prohibit adjusting out of a specific directory (in order that an outsider could get only files from the prespecified index). The proxy would simulate both edges of this standard protocol exchange. For instance, the proxy might take get directions, reject put orders, and filter the neighborhood reaction to a get to list documents.

To understand the true reason for a proxy gateway, why don't we consider several cases.

- A company really wants to set up a web based price list in order that outsiders can easily see the merchandise and prices presented. It really wants to make sure that (a) no outsider can transform the costs or product record and (b) outsiders can obtain only the purchase price list, no of the even more sensitive files located inside.

- A school really wants to allow its college students to get any facts from INTERNET resources on the net. To help offer efficient service, the institution wants to know very well what sites have already been stopped at and what data from the websites have already been fetched; particularly common files will undoubtedly be cached locally.

- A government organization wants to react to queries by way of a database management technique. However, due to inference disorders against directories, the agency really wants to restrict inquiries that come back the entail of a couple of less than five values.

- A business with multiple office buildings really wants to encrypt the info part of all e-mail to addresses at its different offices. (A equivalent proxy in the remote conclusion will take away the encryption.)

Each one of these requirements could be satisfied with a proxy. In the initial situation, the proxy would observe the file exchange protocol data to make sure that only the purchase price list file has been accessed, which file could simply be read, certainly not altered. The school's need could be achieved by way of a logging procedure within the browser. The agency's have could be pleased by way of a special-purpose proxy that interacted together with the database management technique, performing queries but additionally obtaining the amount of values that the response seemed to be computed and including a

random modest error period to benefits from small example sizes. The necessity for minimal login could possibly be handled by way of a specially published proxy that needed strong person authentication (like a challenge response method), which countless operating systems usually do not require. These capabilities are demonstrated in Figure 7-37.

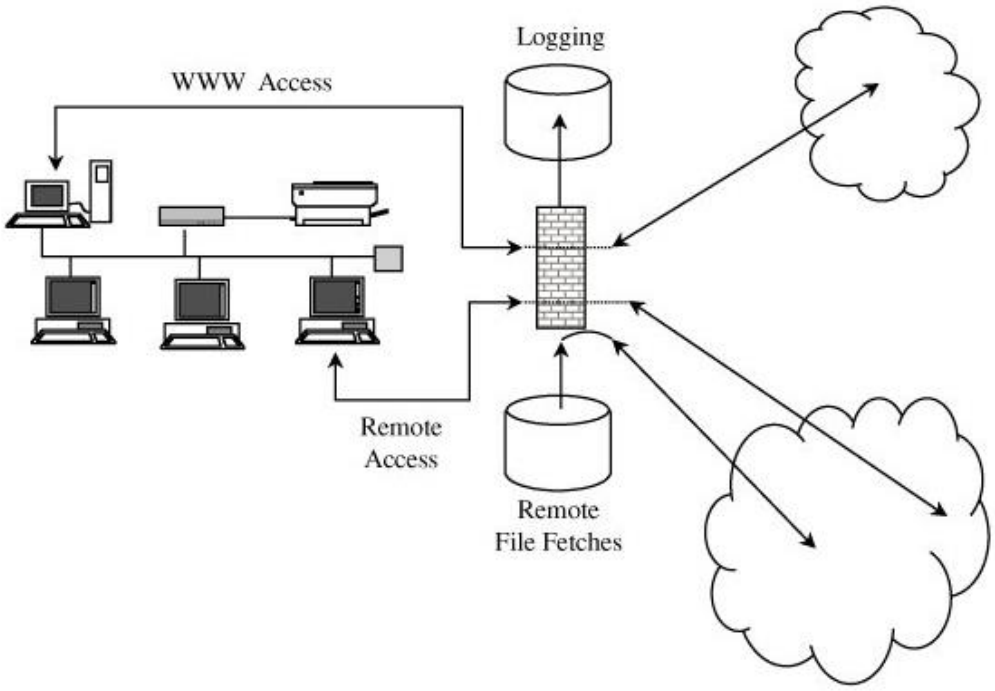


Figure 7-37. Actions of Firewall Proxies.

The proxies for the firewall could be tailored to particular requirements, such as for example logging information regarding accesses. They are able to even present a standard user interface from what could be dissimilar internal features. Suppose the inner network includes a mixture of operating-system types, none which support robust authentication by way of a challenge response token. The proxy can demand from customers solid authentication (brand, security password, and challenge response), validate the challenge response itself, and pass on simply simple label and security password authentication specifics in the proper execution required by way of a specific interior host's operating-system.

The differentiation between a proxy as well as a screening router is usually that the proxy interprets the standard protocol stream to a credit card application, to

control measures with the firewall based on things visible inside the protocol, not only on outside header data.

Guard

A guard is really a sophisticated firewall. Such as a proxy firewall, it gets protocol data devices, interprets them, and goes by through exactly the same or different process data systems that achieve either exactly the same result or perhaps a modified end result. The guard makes a decision what services to execute around the user's behalf relative to its available know-how, such as for example whatever it could reliably find out of the (outside) user's identification, previous interactions, etc. The amount of command a guard can offer is limited simply by what will be computable. But guards and proxy firewalls will be similar enough which the variation between them may also be fuzzy. That's, we can put functionality into a proxy firewall until it starts off to look nearly the same as a guard.

Guard activities could be very sophisticated, just as illustrated in the next examples:

- A university really wants to allow its learners to utilize e-mail up to limit of numerous messages roughly many figures of e-mail within the last so a number of days. Although this effect could be attained by changing e-mail handlers, it really is more easily executed by monitoring the normal point by which all e-mail moves, the mail transport protocol.

- A school wishes its students in order to access the internet but, due to the slow swiftness of its link with the web, it'll allow only a lot of personas per downloaded picture (that's, allowing text method and simple artwork, but disallowing complicated graphics, animation, tunes, or so on).

- A library really wants to make available specific documents but, to aid fair usage of copyrighted matter, it'll allow a individual to retrieve simply the first numerous characters of your document. From then on amount, the catalogue will require the person to cover a fee that'll be forwarded to the writer.

- A company really wants to allow its personnel to fetch documents via ftp. Even so, to prevent launch of viruses, it'll first complete all incoming documents through a Trojan scanner. Despite the fact that several files will undoubtedly be

nonexecutable content material or graphics, the business administrator believes that the trouble of checking them (that ought to pass) will undoubtedly be negligible.

Each one of these scenarios could be implemented to be a modified proxy. As the proxy decision is dependent on some quality from the communication info, we call up the proxy an officer. Since the safety policy implemented with the guard is rather more complex compared to the action of your proxy, the guard's program code is also more complicated and therefore extra exposed to problem. Simpler firewalls contain fewer possible methods to fail or get subverted.

Personal Firewalls

Firewalls typically secure a (sub) community of several hosts. University pupils and personnel in offices will be behind a genuine firewall. Increasingly, house users, individual individuals, and smaller businesses use cable connection modems or DSL cable connections with endless, always-on access. These folks require a firewall, but another firewall computer to safeguard an individual workstation can appear too intricate and expensive. These folks require a firewall's features at less price.

An individual firewall can be an application software that runs over a workstation to obstruct unwanted traffic, normally from the system. An individual firewall can supplement the task of the standard firewall by screening process the type of data an individual host encourage, or it could compensate for having less a normal firewall, just as an exclusive DSL or cable connection modem connection.

In the same way a system firewall screens inbound and outgoing visitors for that community, an individual firewall screens visitors about the same workstation. A workstation could possibly be vulnerable to harmful code or harmful active brokers (ActiveX control buttons or Java applets), leakage of personalized data stored for the workstation, and vulnerability scans to recognize potential weaknesses. Professional implementations of private firewalls consist of Norton Particular Firewall from Symantec, McAfee Individual Firewall, and Area Alarm from Area Labs (right now had by Checkpoint).

The non-public firewall is set up to enforce some insurance plan. For example, an individual may decide that one sites, such as for example computers on the

business network, are extremely trustworthy, but almost every other sites aren't. The user identifies an insurance plan permitting download of program code, unrestricted data posting, and management entry from the organization segment, however, not from other websites. Personal firewalls may also create logs of accesses, which may be useful to verify in the event something harmful does indeed slip from the firewall.

Comparison of Firewall Types

We can summarize the differences among the several types of firewalls we have studied in depth. The comparisons are shown in Table 7.8.

Table 7-8. Comparison of Firewall Types.

Packet Filtering	Stateful Inspection	Application Proxy	Guard	Personal Firewall
Simplest	More complex	Even more complex	Most complex	Similar to packet filtering firewall
Sees only addresses and service protocol type	Can see either addresses or data	Sees full data portion of packet	Sees full text of communication	Can see full data portion of packet
Auditing difficult	Auditing possible	Can audit activity	Can audit activity	Canand usually doesaudit activity
Screens based on connection rules	Screens based on information across packetsin either header or data field	Screens based on behavior of proxies	Screens based on interpretation of message content	Typically, screens based on information in a single packet, using header or data
Complex addressing rules can make configuration tricky	Usually preconfigured to detect certain attack signatures	Simple proxies can substitute for complex addressing rules	Complex guard functionality can limit assurance	Usually starts in "deny all inbound" mode, to which user adds trusted addresses as they appear

Example Firewall Configurations

Let us look at several examples to understand how to use firewalls. We present situations designed to show how a firewall complements a sensible security policy and architecture.

The simplest use of a firewall is shown in Figure 7.38. This environment has a screening router positioned between the internal LAN and the outside network connection. In many cases, this installation is adequate when we need only screen the address of a router.

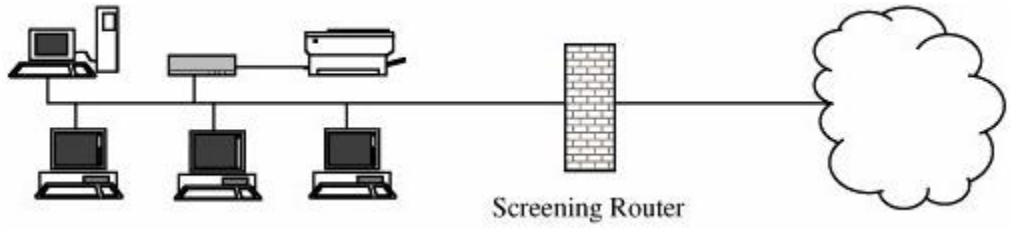


Figure 7.38. Firewall with Screening Router.

Nevertheless , to use a proxy machine, this organization is usually not ideal. Similarly, configuring a router for a intricate set of approved or even rejected addresses is difficult. If the firewall router is successfully attacked, then all traffic on the LAN that the fire wall is connected is noticeable. To reduce this exposure, a proxy firewall is frequently installed on its personal LAN, as shown inside Figure 7.39. In this specific way the only targeted traffic visible on that LOCAL AREA NETWORK is the traffic going into and from the firewall.

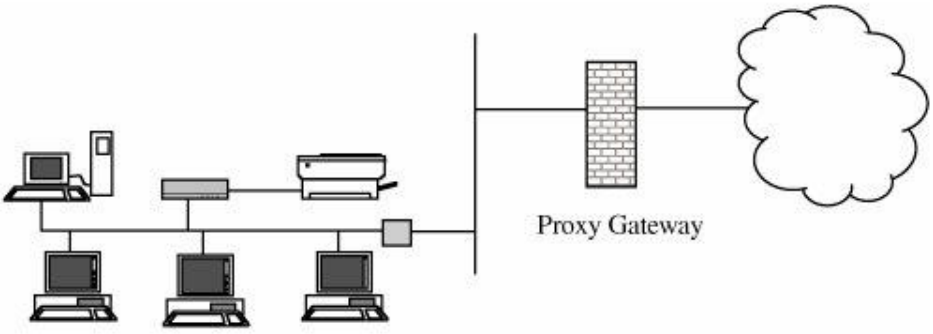


Figure 7.39. Firewall on Separate LAN.

For even more protection, we can add a verification router to this construction, as shown in Figure 7.40. Here, the testing router ensures address correctness to the proxy firewall (so that the proxy firewall cannot be tricked by an outside opponent forging an address through an inside host); typically the proxy firewall filters visitors according to its proxy rules. Also, if the screening router is subverted, the particular traffic to the proxy firewall is visible not any of the very sensitive information on the inner protected LAN.

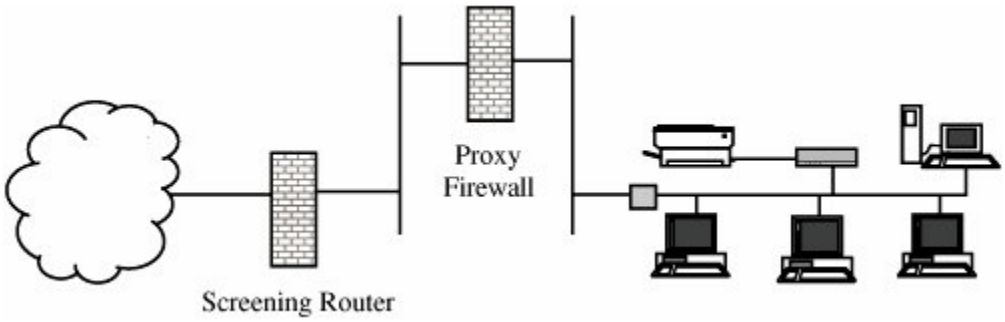


Figure 7.40 Firewall with Proxy and Screening Router.

Combining a computer virus scanner with an individual firewall is certainly both helpful and efficient. Usually, users forget to perform virus scanners each day, but they remember to perform them occasionally, such as for example sometime through the week. However, allowing the virus scanning device execution for the user's memory implies that the scanner picks up a problem just following the fact such as whenever a virus is downloaded within an e-mail attachment. With all the

mix of a virus scanning device and an individual firewall, the firewall directs all inbound e-mail to the herpes virus scanning device, which examines every connection as soon as it reaches the prospective variety and before it really is opened.

An individual firewall works on the very computer it really is trying to secure. Thus, an inspired attacker will probably endeavor an undetected episode that could disable or reconfigure the firewall for future years. Still, specifically for cable connection modem, DSL, along with other "often on" relationships, the static workstation is really a visible and susceptible target to have an ever-present attack neighborhood. An individual firewall can offer reasonable coverage to clients that aren't behind a community firewall.

Let us check out several examples to comprehend how to apply firewalls. We offer situations made to show what sort of firewall suits a sensible safety measures policy and structures.

The simplest usage of a firewall will be shown in Figure 7-38. This surroundings has a verification router positioned between your inside LAN and the exterior network connection. Oftentimes, this installation is definitely adequate whenever we need only display the address of your router.

Although these instances happen to be simplifications, they demonstrate the forms of configurations firewalls defend. Next, we evaluate the forms of episodes against which firewalls can and cannot protect.

What Firewalls CanandCannotBlock

As we have observed, firewalls aren't complete answers to all computer stability complications. A firewall defends just the perimeter of its atmosphere against assaults from outsiders who wish to execute program code or access information on the devices in the shielded environment. Remember these things about firewalls.

- Firewalls can shield an environment only when the firewalls manage the complete perimeter. That's, firewalls work only when no unmediated cable connections breach the perimeter. If actually one inside sponsor connects to another address, by way of a modem for instance, the entire on the inside net is prone throughout the modem and its own host.

- Firewalls usually do not protect data beyond your perimeter; data which have properly handed down (outbound) from the firewall are simply as exposed as though there have been no firewall.

- Firewalls will be the most visible section of an assembly to the exterior, so they will be the most attractive goal for attack. Because of this, several different tiers of protection, named defense comprehensive, are much better than relying on the effectiveness of just a one firewall.

- Firewalls should be correctly set up, that configuration should be updated because the internal and outside environment adjustments, and firewall action reports should be reviewed occasionally for proof attempted or flourishing intrusion.

- Firewalls are goals for penetrators. While a firewall was created to withstand attack, it isn't impenetrable. Developers intentionally retain a firewall little and simple in order that even though a penetrator breaks or cracks it, the firewall doesn't have further tools, such as for example compilers, linkers, loaders, and so on, to keep an attack.

- Firewalls exercise just minor control on the content accepted to the within, and therefore inaccurate information or malicious program code must be manipulated by other stands for in the perimeter.

Firewalls are essential tools in safeguarding an environment linked to a network. Nevertheless, the environment should be seen as a whole, all achievable exposures should be considered, along with the firewall must match a larger, complete security tactic. Firewalls solely cannot secure a host.

7.5. Intrusion Detection Systems

Following the perimeter handles, firewall, and authentication and accessibility controls block particular actions, some customers are admitted to employ a computing system. Many of these controls are precautionary: They stop known bad items from happening. Many reports show that most pc security incidents are usually due to insiders, individuals who would not become blocked by way of a firewall. And insiders need access with important privileges to accomplish their daily work opportunities. Almost all damage from insiders isn't malicious; it really is honest people producing honest mistakes. Then simply, too, you can find the

potential harmful outsiders who've somehow transferred the monitors of firewalls and entry controls. Avoidance, although necessary, isn't a complete personal computer security control; recognition during an occurrence copes with injury that can't be prevented beforehand. Halme and Bauer market research the number of controls to handle intrusions.

Intrusion detection devices complement these precautionary controls because the next type of protection. An intrusion recognition system (IDS) is really a device, usually another separate laptop that monitors action to identify destructive or suspicious activities. Kemmerer and Vigna review the annals of IDSs. An IDS is really a sensor, such as a smoking detector, that boosts an security alarm if specific items occur. A style of an IDS is usually shown in Figure 7.41. The ingredients in the number will be the four basic components of an intrusion diagnosis system, in line with the Common Intrusion Diagnosis Platform of [STA96]. An IDS will get fresh inputs from receptors. It helps you to save those inputs, analyzes them, and needs some controlling steps.

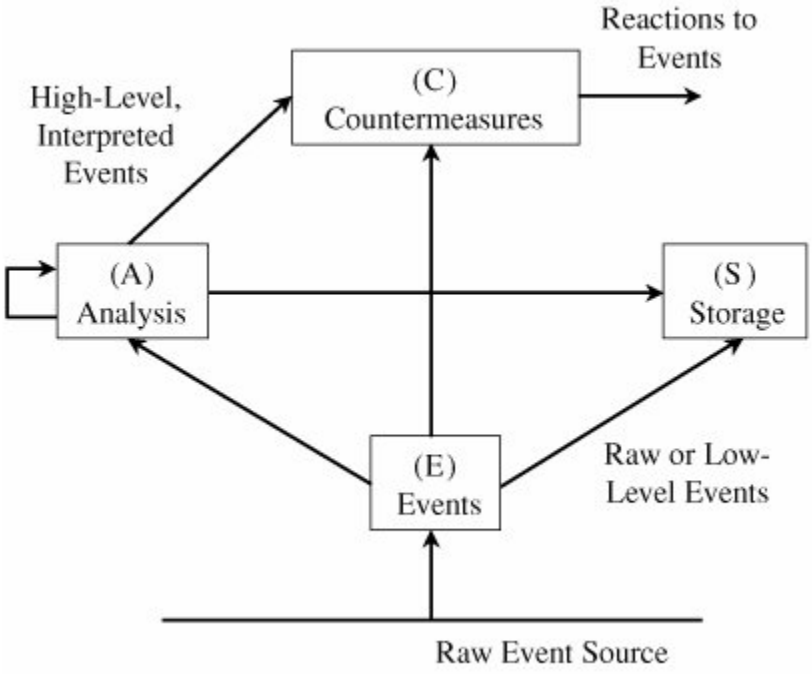


Figure 7.41. Common Components of an Intrusion Detection Framework.

IDSs perform variety of capabilities:

- monitoring consumers and program activity
- auditing system settings for vulnerabilities and misconfigurations
- evaluating the integrity of important system and documents
- recognizing known harm patterns in technique activity
- identifying abnormal exercise through statistical analysis
- managing audit paths and highlighting person violation of coverage or usual activity
- correcting system settings errors
- putting in and operating traps to report information regarding intruders

Nobody IDS performs many of these functions. Why don't we look more carefully at the forms of IDSs and their used in providing security.

Types of IDSs

The two standard forms of intrusion detection methods are signature centered and heuristic. Signature-based intrusion recognition systems perform uncomplicated pattern-matching and review situations that suit a pattern matching to a recognized attack variety. Heuristic intrusion diagnosis systems, also called anomaly based, create a model of appropriate habits and flag exceptions compared to that model; for future years, the administrator can tag a flagged habit as acceptable so the heuristic IDS will right now treat that formerly unclassified conduct as acceptable.

Intrusion detection units can be community based or variety structured. A network-based IDS is really a stand-alone device mounted on the system to monitor visitors throughout that community; a host-based IDS works about the same workstation or consumer or host, to safeguard that one sponsor.

Early intrusion recognition systems worked following the fact, by researching logs of method activity to identify prospective misuses that got took place. The administrator could examine the results in the IDS to get and repair weaknesses in the machine. Now, on the other hand, intrusion detection techniques operate instantly (or near real-time), watching exercise and bringing up alarms with time with the administrator to use protective action.

Signature-Based Intrusion Detection

A simple signature to get a known attack kind might describe some TCP SYN packets delivered to many different jacks in succession and sometimes close to each other, as will be the situation for a slot check. An intrusion recognition system may possibly find nothing different in the initial SYN, claim, to dock 80, and another (from exactly the same source deal with) to dock 25. But as increasingly more ports acquire SYN packets, specifically ports that aren't open, this style reflects a feasible port scan. In the same way, some implementations from the standard protocol stack fail should they obtain an ICMP packet having a data amount of 65535 bytes, consequently this type of packet will be a pattern that to watch.

The issue with signature-based diagnosis may be the signatures themselves. An attacker will attempt to modify a simple attack so that you won't match the recognized signature of this attack. For instance, the attacker may switch lowercase to uppercase words or convert symbolic such as for example "blank room" to its figure code comparative %20. The IDS must actually work from the canonical type of the data supply to be able to notice that %20 complements a pattern which has a blank room. The attacker may add malformed packets how the IDS will dsicover, to intentionally result in a structure mismatch; the standard protocol handler stack will dispose of the packets due to the malformation. Each one of these variations could possibly be discovered by an IDS, but extra signatures require extra work with the IDS, which decreases performance.

Needless to say, signature-based IDSs cannot identify a new strike that a signature isn't yet installed inside the database. Every invasion type starts off as a fresh pattern sometime, plus the IDS is certainly helpless to alert of its lifetime.

Signature-based intrusion diagnosis systems have a tendency to use statistical evaluation. This approach utilizes statistical equipment both to acquire sample proportions of key signals (such as for example amount of exterior activity, amount of active processes, amount of transactions) also to determine if the collected measurements match the predetermined episode signatures.

Ultimately, signatures should fit every instance of your attack, match understated variations in the attack, however, not match traffic that's not section of an attack. Even so, this goal will be fantastic but unreachable.

Heuristic Intrusion Detection

Because signatures happen to be limited to particular, known attack habits, another type of intrusion detection gets to be useful. Rather than looking for fits, heuristic intrusion recognition looks for habit that is unusual. The original job of this type focused on the average person, looking for characteristics of this person that may be helpful in knowing normal and unusual behavior. For instance, one person might always start the day off by studying e-mail, write various documents utilizing a word processor chip, and occasionally regress to something easier files. These activities would be typical. This user will not seem to work with many administrator resources. If see your face tried to gain access to sensitive system supervision utilities, this different behavior may be a idea that another person was acting beneath the user's identity.

If we think about a compromised program used, it starts clean up, without intrusion, also it ends dirty, completely compromised. There could be no stage in the track of use where the system evolved from tidy to dirty; it had been much more likely that little filthy events occurred, sometimes at first and increasing because the system became deeper compromised. Anybody of those activities might be appropriate by itself, however the accumulation of these and the buy and speed of which they occurred might have been indicators that something undesirable was going on. The inference engine motor of any intrusion detection method performs continuous examination of the machine, elevating an alert once the system's dirtiness surpasses a threshold.

Inference engines function in two techniques. Some, referred to as state-based intrusion recognition systems, start to see the system going right through changes of general state or settings. They make an effort to detect once the system provides veered into unsafe settings. Others make an effort to map current task onto a style of unacceptable exercise and increase an alarm once the activity has a resemblance to the model. They are known as model-based intrusion diagnosis systems. Later function sought to create a dynamic style of behavior, to support variation and development in someone's actions as time passes. The approach compares real task with a recognized representation of normality.

Alternatively, intrusion recognition can work coming from a model of identified bad activity. For instance, except for several utilities (login, modification

password, create consumer), any attempt to gain access to a password data file can be suspect. This type of intrusion detection is recognized as misuse intrusion recognition. In this job, the real action is likened against a acknowledged suspicious area.

All heuristic intrusion diagnosis activity is labeled in another of three groups: very good/benign, dubious, or unknown. As time passes, specific forms of actions can shift from one of the categories to some other, corresponding for the IDS's understanding whether certain activities are appropriate or not.

Much like pattern-matching, heuristic intrusion diagnosis is bound by the quantity of information the machine has noticed (to classify measures into the appropriate category) and exactly how well the existing actions match one of these brilliant categories.

Stealth Mode

An IDS is really a network system (or, regarding a host-based IDS, an application running on the network system). Any community device is possibly vulnerable to community attacks. How beneficial would an IDS be if it itself had been deluged which has a denial-of-service assault? If an attacker been successful in logging directly into a system in the protected community, wouldn't attempting to disable the IDS function as next step?

To counter those complications, most IDSs work in stealth method, whereby an IDS offers two system interfaces: one with the network (or system segment) being supervised and another to generate notifications and perhaps various other administrative desires. The IDS utilizes the monitored software as input just; it never transmits packets out during that interface. Typically, the interface can be configured so the device does not have any published address from the monitored interface; that's, a router cannot road anything compared to that address directly, as the router will not know this type of device exists. It's the excellent passive wiretap. In the event the IDS must create an alert, it utilizes only the security alarm interface on a totally separate control community. Such a structures is demonstrated in Figure 7-42.

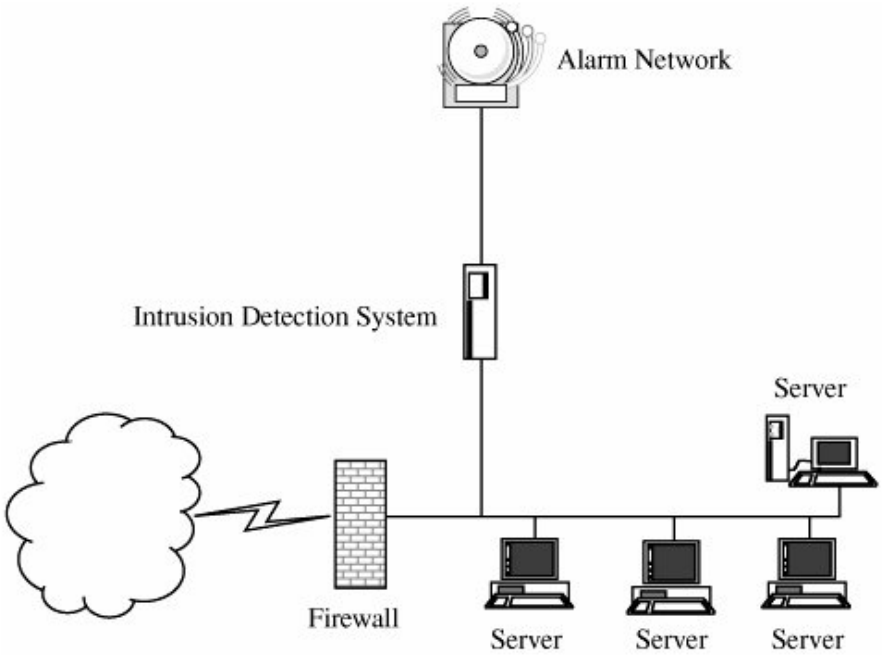


Figure 7.42. Stealth Mode IDS Connected to Two Networks.

Additional IDS Types

Some security technicians consider other products being IDSs as well. For example, to detect undesirable code modification, plans can assess the active edition of an application code using a saved version of your digest of this program code. The tripwire method [KIM98] may be the renowned software program (or static files) comparison course. You work tripwire on a fresh system, also it produces a hash benefit for each document; and then you conserve these hash worth in a safe place (offline, in order that no intruder can improve them while changing a system document). In the event that you later suspect one's body might have been affected, you rerun tripwire, supplying it the kept hash beliefs. It recomputed the hash worth and records any mismatches, which may indicate files which were changed.

Program vulnerability scanners, such as for example ISS Scanning device or Nessus, could be operate against a community. They look for regarded vulnerabilities and survey flaws found.

As we have observed, a honeypot is really a faux environment designed to lure an attacker. It could be viewed as an IDS, in the impression that this honeypot may

document an intruder's activities and even try to track who the attacker is certainly from steps, packet info, or connections.

Objectives for Intrusion Diagnosis Systems

The two varieties of intrusion detection pattern corresponding and heuristic represent unique approaches, all of which has benefits and drawbacks. Actual IDS goods often blend both approaches.

Ultimately, an IDS ought to be fast, straightforward, and precise, while at exactly the same time being complete. It will detect all strikes with little efficiency charges. An IDS might use some or all of the next design techniques:

- Filtration on packet headers
- Filtration system on packet content
- Maintain network state
- Use organic, multipacket signatures
- Use minimal amount of signatures with utmost effect
- Filter instantly, online
- Hide its presence
- Work with optimal sliding period window size to complement signatures

Responding to Alarms

Whatever the style, an intrusion recognition system boosts an security alarm when it detects a suit. The security alarm can range between something modest, such as for example writing an email within an audit log, to something important, such as for example paging the machine security administrator. Certain implementations permit the user to find out what action the machine should undertake what events.

What are feasible responses? The number is unlimited and may come to be anything the administrator can see right now (and plan). Generally, responses belong to three major types (any or which may be used within a response):

- Monitor, collect info, perhaps increase level of data collected

IDOL

- Protect, act to lessen exposure
- Contact a human

Monitoring is suitable for an episode of moderate (first) impact. Possibly the real goal would be to check out the intruder, to check out what resources are increasingly being seen or what attempted episodes are tried out. Another monitoring probability is to capture all traffic from the given supply for future examination. This approach ought to be invisible towards the attacker. Protecting often means increasing access handles and even creating a source unavailable (for instance, shutting off a system connection or creating a file unavailable). The machine may also sever the community interconnection the attacker can be using. As opposed to monitoring, protecting is quite noticeable to the attacker. Ultimately, calling a individuals allows unique discrimination. The IDS may take an initial protective action promptly while also producing an aware of a human being who might take seconds, short minutes, or much longer to respond.

False Results

Intrusion detection methods are not best, and mistakes will be their biggest difficulty. Although an IDS might discover an intruder appropriately more often than not, it could stumble in two various ways: by bringing up an security alarm for a thing that is not actually an harm (named a false favorable, or form I error inside the statistical group) or not necessarily raising an security alarm for a genuine attack (a phony negative, or variety II problem). Way too many false positives implies the administrator will undoubtedly be less confident in the IDS's warnings, probably leading to a genuine alarm's being overlooked. But wrong negatives imply that real attacks will be moving the IDS without steps. We point out that the amount of wrong positives and phony negatives signifies the level of sensitivity of the machine. Just about all IDS implementations permit the administrator to tune the system's level of sensitivity, to strike a satisfactory balance between phony advantages and disadvantages.

IDS Advantages and Limitations

Intrusion detection methods are evolving items. Research began within the middle-1980s and goods had appeared by mid-1990s. Nevertheless, this area carries on to improve as new study influences the look of products.

On the benefit, IDSs find an ever-growing amount of serious problems. So when we find out about problems, we are able to put their signatures for the IDS model. Hence, as time passes, IDSs continue steadily to improve. At exactly the same time, they are being cheaper and simpler to administer.

On the drawback, keeping away from an IDS is really a first main concern for profitable attackers. An IDS that's not well defended is certainly useless. Luckily for us, stealth setting IDSs are tough even to get on an interior network, aside from to compromise.

IDSs search for regarded weaknesses, whether through habits of known assaults or types of normal behavior. Equivalent IDSs could have indistinguishable vulnerabilities, and their choice criteria may overlook similar attacks. Focusing on how to evade a specific style of IDS can be an important little bit of intelligence passed in the attacker community. Needless to say, once manufacturers notice a shortcoming within their products, they make an effort to fix it. Luckily for us, commercial IDSs will be very good at identifying disorders.

Another IDS restriction is its level of sensitivity, which is tough to calculate and fine-tune. IDSs won't be excellent, so locating the proper balance is crucial.

A final limitation isn't of IDSs by itself, but is among work with. An IDS will not run itself; somebody has to check its background and react to its alarms. An administrator can be foolish to get and mount an IDS and ignore it.

Generally, IDSs are great improvements to a network's stability. Firewalls block visitors to particular plug-ins or addresses; in addition they constrain certain practices to restrict their effects. But by description, firewalls need to allow some visitors to key in a protected spot. Enjoying what that visitors actually does in the protected area can be an IDS's employment, which it can quite well.

7.6. Secure E-Mail

The final handle we consider comprehensive is protected e-mail. Consider how much you utilize e-mail and just how much you depend on the exactness of its details. How can you react in the event that you received a note from your trainer saying that as you had done therefore properly in your lessons so far, you're excused from carrying out any further do the job in it? Imagine if that message had been a joke from the classmate? We depend on e-mail's confidentiality and

integrity for very sensitive and important marketing communications, even though common e-mail has minimal confidentiality or integrity. Within this section we check out how to include confidentiality and integrity safeguard to regular e-mail.

Security for E-mail

E-mail is essential for today's business, as well a convenient moderate for marketing communications among ordinary consumers. But, once we noted before, e-mail is quite public, subjected at every level from sender's workstation towards the recipient's screen. In the same way you would definitely not put hypersensitive or private applying for grants a postcard, you need to also recognize that e-mail emails are subjected and designed for others to learn.

Sometimes we wish e-mail to become more secure. To explain and implement a far more secure web form, we start by analyzing the exposures of common e-mail.

Hazards to E-mail

Consider risks to e-mail:

- Communication interception (confidentiality)
- Concept interception (clogged delivery)
- Subject matter interception and succeeding replay
- Message articles modification
- Message source modification
- Message information forgery by outsider
- Message source forgery by outsider
- Message articles forgery by recipient
- Message origins forgery by recipient
- Denial of communication transmission

Confidentiality and content material forgery tend to be treated by encryption. Encryption may also assist in a protection against replay, although we'd also need

to use a standard protocol where each message is made up of something unique that's encrypted. Symmetric encryption cannot drive back forgery by way of a receiver, since both sender and receiver share a standard key; however, open public key techniques can permit a receiver decrypt however, not encrypt. Due to lack of handle over the mid items of a system, senders or receivers commonly cannot drive back blocked delivery.

Prerequisites and Solutions

If we have been to produce a list of certain requirements for risk-free e-mail, our hope list would are the following protections.

- *message confidentiality* (the information is not uncovered en route towards the receiver)
- message integrity (the particular receiver sees will be what was delivered)
- *sender authenticity* (the device is usually confident who the sender seemed to be)
- Nonrepudiation (the sender cannot deny possessing sent the subject matter)

Not absolutely all these qualities are essential for every concept, but a perfect secure e-mail package deal allows these capabilities being invoked selectively

Designs

The typical for encrypted e-mail originated by the web Culture, through its structures panel (IAB) and analysis (IRTF) and anatomist (IETF) task causes. The encrypted e-mail methods are documented being an Internet regular in records 1421, 1422, 1423, and 1424 this common is actually the 3rd refinement of the initial specification.

Among the design aims for encrypted e-mail was basically allowing security-enhanced information to visit as ordinary announcements through the prevailing Internet e-mail method. This requirement means that the large pre-existing e-mail network wouldn't normally require change to support security. Consequently, all protection comes about in the body of a note.

Confidentiality

Because the security has several elements, we get started our description of these by looking very first at how exactly to provide confidentiality improvements. The sender decides a (random) symmetric algorithm encryption main. After that, the sender encrypts a backup of the complete message for being transmitted, integrating FROM:, TO:, SUBJECT:, and Night out: headers. Next, the sender prepends plaintext headers. For key element administration, the sender encrypts the communication key beneath the recipient's public major, and attaches that towards the message as well. The process of fabricating an encrypted e-mail information is displayed in Figure 7.43.

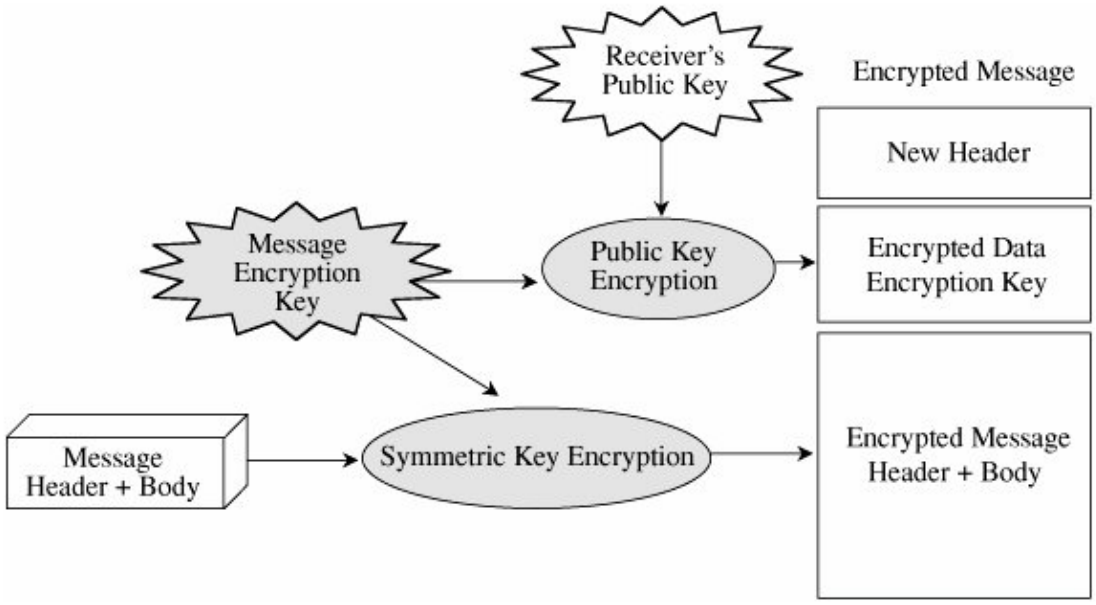


Figure 7-43. Overview of Encrypted E-mail Processing.

Encryption could deliver any string as outcome. Various e-mail handlers' count on that message visitors will not include characters apart from the standard printable characters. System e-mail handlers make use of unprintable heroes as control indicators in the site visitor's stream. In order to avoid problems in transmitting,

encrypted e-mail changes the complete cipher text communication to printable personas. A good example of an encrypted e-mail subject matter is demonstrated in Figure 7.44. Spot the three helpings: an exterior (plaintext) header, a part where the information encryption key could be transferred, along with the encrypted concept itself. (The encryption is usually demonstrated with shading.)

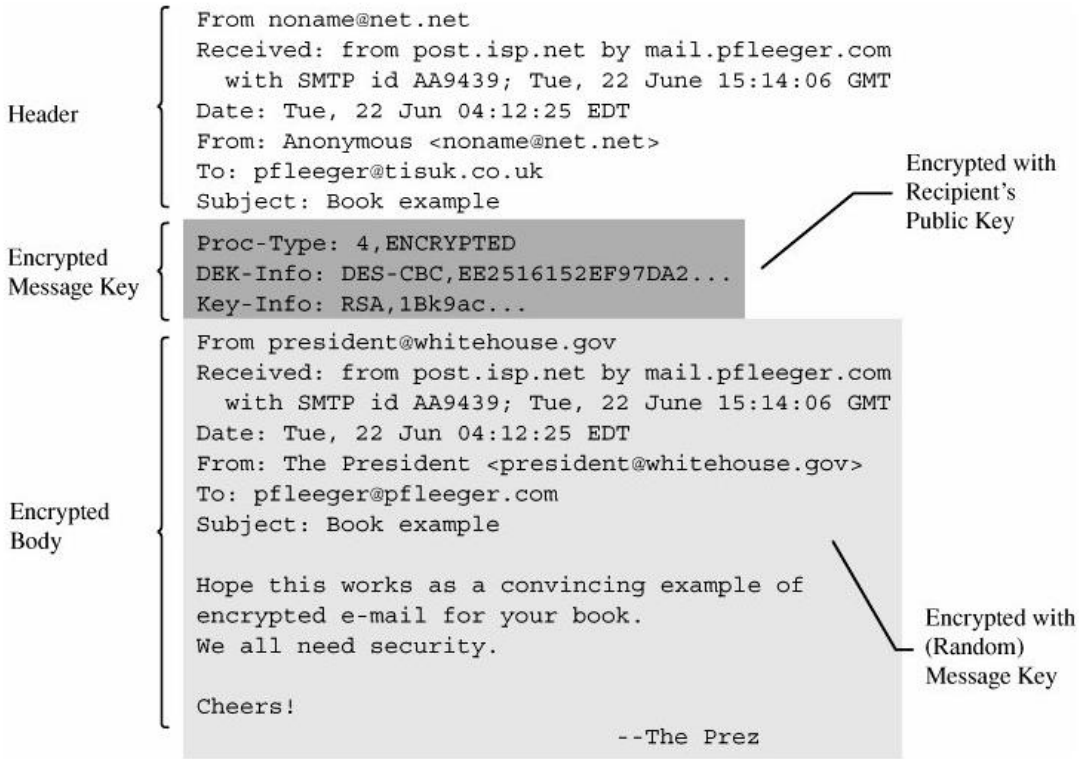


Figure 7.44. Encrypted E-mailSecured Message.

The encrypted e-mail regular works most simply as just explained, making use of both symmetric and asymmetric encryption. The typical is also described for symmetric encryption just: To utilize symmetric encryption, the sender and device must have formerly established a contributed secret encryption main. The processing variety ("Proc-Type") field explains to what privacy advancement services have already been applied. In the info exchange key industry ("DEK-Info"), the type of key swap (symmetric or asymmetric) is definitely shown. The main element exchange ("Key-Info") industry contains the information encryption key element, encrypted under this discussed encryption major. The field likewise recognizes the originator (sender) so the receiver can establish which provided symmetric key had been used. If the main element exchange technique had been

to utilize asymmetric encryption, the main element exchange discipline would support the message encryption discipline, encrypted beneath the recipient's public primary. Also included may be the sender's document (useful for determining authenticity as well as for generating replies).

The encrypted e-mail normal supports several encryption algorithms, applying popular algorithms such as for example DES, triple DES, and AES for meaning confidentiality, and RSA and DiffieHellman for crucial exchange.

Other Stability Features

Along with confidentiality, we might want various types of integrity for protected e-mail.

Encrypted e-mail information always carry an electronic signature, therefore the authenticity and nonrepudiability from the sender is guaranteed. The integrity can be assured due to a hash work (called a note integrity check out, or MIC) inside the digital signature bank. Optionally, encrypted e-mail information could be encrypted for confidentiality.

Notice in Figure 7.44 the header in the message (inside the encrypted section) varies from that outside the house. A sender's personality or the specific subject of a note can be hidden in the encrypted portion.

The encrypted e-mail handling can combine with common e-mail packages, thus an individual can send both improved and nonenhanced communications, as demonstrated in Figure 7-45. In case the sender decides to include enhancements, a supplementary little bit of encrypted e-mail handling is invoked for the sender's stop; the receiver must remove the improvements. But without improvements, messages flow throughout the email handlers as common.

S/MIME (reviewed later on this segment) can hold the swap of apart from just texts: help for voice, artwork, video, along with other kinds of sophisticated message parts.

IDOL

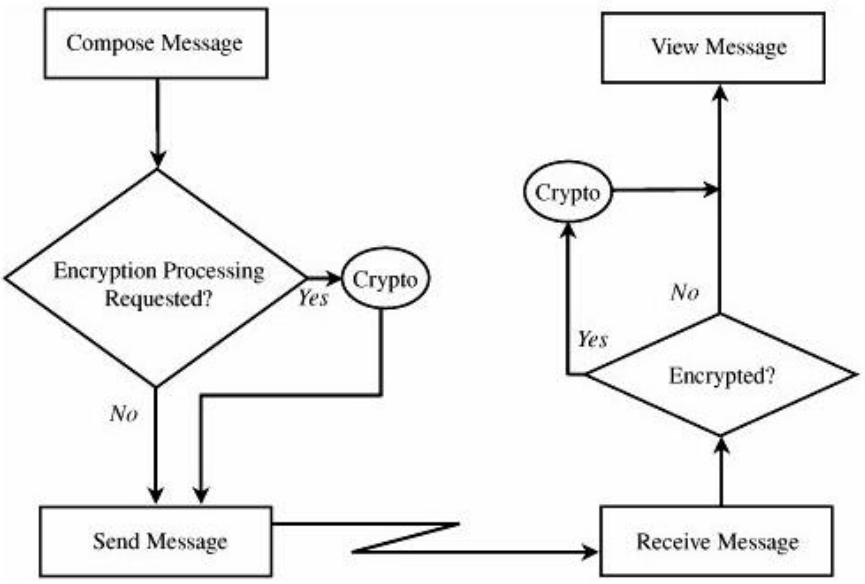


Figure 7.45. Encrypted E-mail Processing in Message Transmission.

Encryption for Secure E-mail

The significant problem with encrypted e-mail can be key administration. The certificate system described in Section 2 is great for exchanging tips as well as for associating a personal information with a open public encryption key. The issue with certificates is definitely creating the hierarchy. Countless organizations own hierarchical buildings. The encrypted e-mail problem is transferring beyond the individual organization with an interorganizational hierarchy. Exactly due to the issue of imposing a hierarchy on the nonhierarchical globe, PGP originated as an easier type of encrypted e-mail.

Encrypted e-mail supplies strong end-to-end safety measures for e-mail. Triple DES, AES, and RSA cryptography are very strong, particularly if RSA can be used with an extended bit crucial (1024 bits or even more). The vulnerabilities left over with encrypted e-mail result from the points not really protected: the endpoints. An attacker with entry could subvert a sender's or receiver's device, modifying the program code that does indeed the privacy improvements or organizing to drip a cryptographic key element.

Case in point Secure E-mail Systems

Encrypted e-mail applications can be found from many resources. Several colleges (like Cambridge School in England as well as the School of Michigan in america) and businesses (BBN, RSA-DSI, and Trusted Facts Systems) are suffering from either prototype or professional types of encrypted e-mail.

PGP

PGP means Pretty Good Level of privacy. It was created by Phil Zimmerman in 1991. Initially a free bundle, it grew to be a commercial product or service after being acquired by Network Affiliates in 1996. A freeware edition is still obtainable. PGP is accessible, both in professional types and freeware, which is heavily utilized by individuals exchanging personal e-mail.

PGP addresses the main element distribution problem using what is named a "band of put your trust in" or perhaps a user's "keyring." One individual directly provides public key to some other, or the next consumer fetches the first's open public key from the server. Some individuals contain their PGP common keys in the bottom of e-mail emails. And one particular person can give another person's major to one third (as well as a fourth, and so forth). Thus, the main element association problem results in being among caveat emptor: "Allow purchaser beware." EASILY am reasonably positive an e-mail message seriously originates from you and contains not happen to be tampered with, I'll use your linked public key. EASILY trust you, I might also rely on the secrets you provide me for other folks. The model reduces intellectually once you give me all of the keys you acquired from men and women, who subsequently gave you all of the keys they received from still other folks, who gave all of them their keys, etc.

You warning sign each primary you offer me. The tips you offer me could also have been agreed upon by other folks. I opt to have confidence in the veracity of an key-and-identity combination, predicated on who signed the main element.

PGP will not mandate an insurance plan for establishing put your trust in. Rather, each individual is absolve to decide how many to believe in each key acquired.

The PGP running does some or every one of the following actions, based on whether confidentiality, integrity, authenticity, or some mix of these is determined:

Create an arbitrary session key to get a symmetric algorithm

Encrypt the concept, using the procedure key (for subject matter confidentiality).

Encrypt the program key beneath the recipient's public essential.

Generate a note digest or hash in the message; signal the hash by encrypting it while using sender's private essential (for meaning integrity and authenticity).

Affix the encrypted period main to the encrypted subject matter and digest.

Transmit the information to the receiver.

The receiver reverses these ways to get and validate the information content.

S/MIME

An Internet common governs how e-mail is usually sent and acquired. The overall MIME specification identifies the file format and dealing with of e-mail parts. S/MIME (Secure Multipurpose World wide web Mail Extensions) may be the Internet regular for safe e-mail attachments.

S/MIME is very much indeed like PGP and its own predecessors, PEM (Privacy-Enhanced Email) and RIPEM. S/MIME have been adopted in professional e-mail packages, such as for example Eudora and Microsoft Perspective.

The principal variation between S/MIME and PGP may be the method of crucial exchange. Primary PGP depends upon each user's exchanging tips with all possible recipients and creating a wedding ring of dependable recipients; in addition, it requires establishing a qualification of rely upon the authenticity of this keys for all those recipients. S/MIME utilizes hierarchically validated certificates, generally symbolized in X.509 structure, for key alternate. So, with S/MIME, the sender and receiver need not have exchanged tips in advance so long as they have a standard certifier they both faith.

S/MIME works together with a number of cryptographic algorithms, such as for example DES, AES, and RC2 for symmetric encryption.

S/MIME performs safety transformations nearly the same as those for PGP. PGP was initially originally created for plaintext communications, but S/MIME deals with (secures) a variety of attachments, such as for example documents (for instance, spreadsheets, images, presentations, videos, and noise). Since it is built-

into many industrial e-mail deals, S/MIME will probably dominate the risk-free e-mail market.

7.7 Review Questions

1. Describe a social engineering attack you could use to obtain a user's password.
2. Is a social engineering attack more likely to succeed in person, over the telephone, or through e-mail? Justify your answer
3. A port scanner is a tool useful to an attacker to identify possible vulnerabilities in a potential victim's system. Cite a situation in which someone who is not an attacker could use a port scanner for a nonmalicious purpose.
4. Compare copper wire, microwave, optical fiber, infrared, and (radio frequency) wireless in their resistance to passive and active wiretapping.
5. What is a "man in the middle" attack? Cite a real-life example (not from computer networking) of such an attack. Suggest a means by which sender and receiver can preclude a man-in-the-middle attack. (a) Cite a means not requiring cryptography. (b) Cite a means involving cryptography but also ensuring that the man in the middle cannot get in the middle of the key exchange
6. Signing of mobile code is a suggested approach for addressing the vulnerability of hostile code. Outline what a code-signing scheme would have to do.
7. Does a VPN use link encryption or end-to-end? Justify your answer.
8. Why is a firewall a good place to implement a VPN? Why not implement it at the actual server(s) being accessed?
9. Can encrypted e-mail provide verification to a sender that a recipient has read an e-mail message? Why or why not?
10. Can message confidentiality and message integrity protection be applied to the same message? Why or why not?

IDOL

11. What are the advantages and disadvantages of an e-mail program (such as Eudora or Outlook) that automatically applies and removes protection to e-mail messages between sender and receiver?

7.8 References

1. Security in Computing, Fourth Edition By Charles P. Pfleeger - Pfleeger Consulting Group, Shari Lawrence Pfleeger - RAND Corporation Publisher: Prentice Hall

2. Cryptography and Network Security - Principles and Practice fifth edition Stallings William Publisher: Pearson

3. Cryptography And Network Security 3rd Edition behrouz a forouzan and debdeepmukhopadhyay 3/E Publisher: McGraw Hill Education

4. Cryptography and Network Security, 3e AtulKahate Publisher: McGraw Hill

IDOL

Chapter 8

Database Security - I

IDOL

IDOL

8.0 Objectives

We begin this chapter with a brief summary of database terminology. Then we consider the security requirements for database management systems. Two major security problems - integrity and secrecy, are explained in a database context.

8.1 Introduction

Protecting data is at the heart of many secure systems, and many users (people, programs, or systems) rely on a database management system (DBMS) to manage the protection. There is substantial current interest in DBMS security because databases are newer than programming and operating systems. Databases are essential to many business and government organizations, holding data that reflect the organization's core competencies. Often, when business processes are reengineered to make them more effective and more in tune with new or revised goals, one of the first systems to receive careful scrutiny is the set of databases supporting the business processes. Thus, databases are more than software-related repositories. Their organization and contents are considered valuable corporate assets that must be carefully protected. However, the protection provided by database management systems has had mixed results. Over time, we have improved our understanding of database security problems, and several good controls have been developed. But, as you will see, there are still more security concerns for which there are no available controls.

8.2 Introduction to Databases

We begin by describing a database and defining terminology related to its use. We draw on examples from what is called the relational database because it is one of the most widely used types. However, all the concepts described here apply to any type of database. We first define the basic concepts and then use them to discuss security concerns.

8.2.1 Concept of a Database

A database is a collection of *data* and a set of *rules* that organize the data by specifying certain relationships among the data. Through these rules, the user describes a *logical* format for the data. The data items are stored in a file, but the precise *physical* format of the file is of no concern to the user. A database administrator is a person who defines the rules that organize the data and also controls who should have access to what parts of the data. The user interacts with the database through a program called a database manager or a database management system (DBMS), informally known as a front end.

8.2.2 Components of Databases

The database file consists of records, each of which contains one related group of data. As shown in the example in Table 8-1, a record in a name and address file consists of one name and address. Each record contains fields or elements, the elementary data items themselves. The fields in the name and address record are NAME, ADDRESS, CITY, STATE, and ZIP (where ZIP is the U.S. postal code). This database can be viewed as a two-dimensional table, where a record is a row and each field of a record is an element of the table.

ADAMS	212 Market St.	Columbus	OH	43210
BENCHLY	501 Union St.	Chicago	IL	60603
CARTER	411 Elm St.	Columbus	OH	43210

Table 8-1 Example of a database

Not every database is easily represented as a single, compact table. The database in Figure 8-1 logically consists of three files with possibly different uses. These three files could be represented as one large table, but that depiction may not improve the utility of or access to the data.

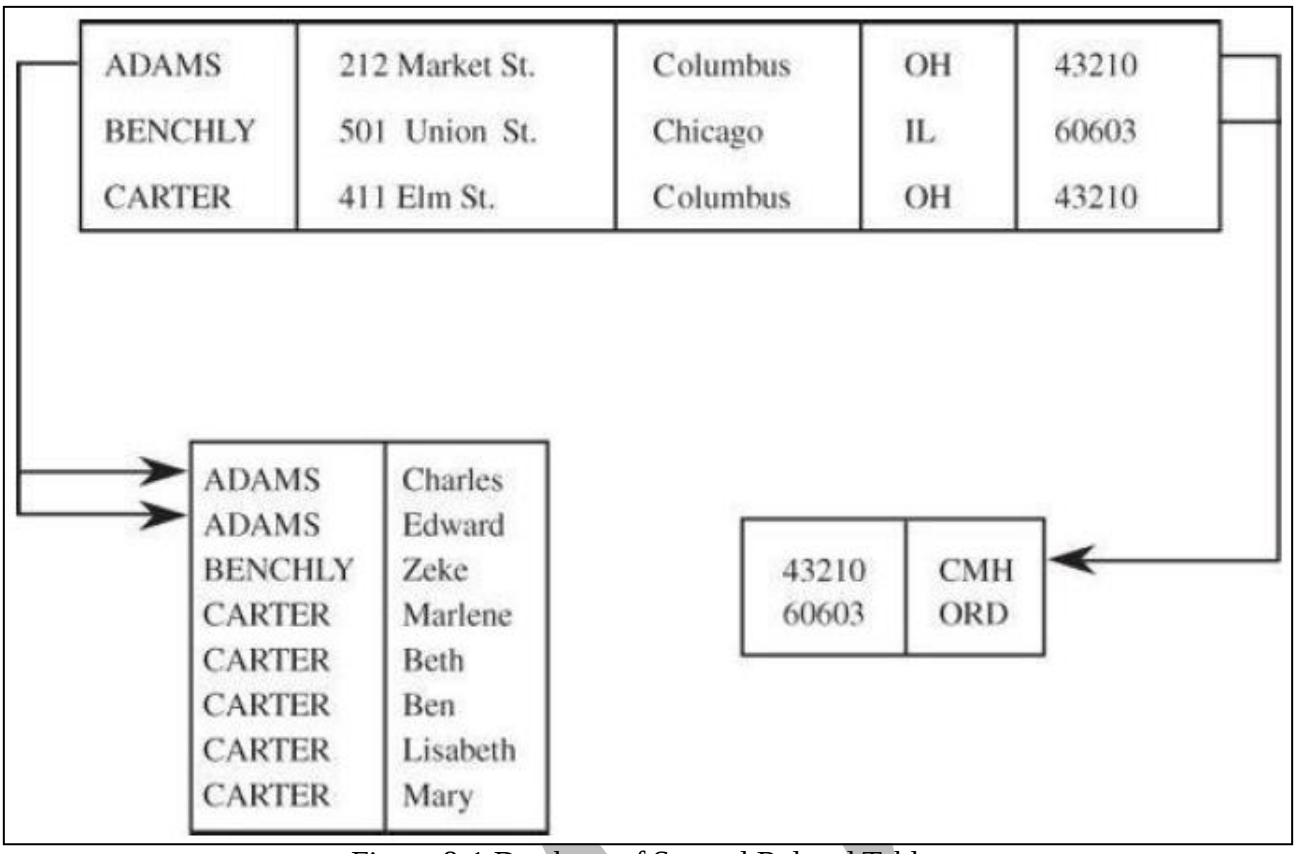


Figure 8-1 Database of Several Related Tables

The logical structure of a database is called a schema. A particular user may have access to only part of the database, called a subschema. The overall schema of the database in Figure 8-1 is detailed in Table 8-2. The three separate blocks of the figure are examples of subschemas, although other subschemas of this database can be defined. We can use schemas and subschemas to present to users only those elements they wish or need to see. For example, if Table 8-1 represents the employees at a company, the subschema on the lower left can list employee names without revealing personal information such as home address.

Name	First	Address	City	State	Zip	Airport
ADAMS	Charles	212 Market St.	Columbus	OH	43210	CMH
ADAMS	Edward	212 Market St.	Columbus	OH	43210	CMH
BENCHLY	Zeke	501 Union St.	Chicago	IL	60603	ORD
CARTER	Marlene	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Beth	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Ben	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Lisabeth	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Mary	411 Elm St.	Columbus	OH	43210	CMH

Table 8-2 Schema of Database from Figure 8-1

The rules of a database identify the columns with names. The name of each column is called an attribute of the database. A relation is a set of columns. For example, using the database in Table 8-

2, we see that NAME-ZIP is a relation formed by taking the NAME and ZIP columns, as shown in Table 8-3. The relation specifies clusters of related data values in much the same way that the relation "mother of" specifies a relationship among pairs of humans. In this example, each cluster contains a pair of elements, a NAME and a ZIP. Other relations can have more columns, so each cluster may be a triple, a 4-tuple, or an *n*-tuple (for some value *n*) of elements.

Name	Zip
ADAMS	43210
BENCHLY	60603
CARTER	43210

Table 8-3 Relation in a Database

8.2.3 Queries

Users interact with database managers through commands to the DBMS that retrieve, modify, add, or delete fields and records of the database. A command is called a query. Database management systems have precise rules of syntax for queries. Most query languages use an English-like notation, and many are based on SQL, a structured query language originally developed by IBM. We have written the example queries in this chapter to resemble English sentences so that they are easy to understand. For example, the query

SELECT NAME = 'ADAMS'

retrieves all records having the value *ADAMS* in the NAME field. The result of executing a query is a subschema. One way to form a subschema of a database is by selecting records meeting certain conditions. For example, we might select records in which ZIP=43210, producing the result shown in Table 8-4.

Name	First	Address	City	State	Zip	Airport
ADAMS	Charles	212 Market St.	Columbus	OH	43210	CMH
ADAMS	Edward	212 Market St.	Columbus	OH	43210	CMH
CARTER	Marlene	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Beth	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Ben	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Lisabeth	411 Elm St.	Columbus	OH	43210	CMH
CARTER	Mary	411 Elm St.	Columbus	OH	43210	CMH

Table 8-4 Results of a SELECT Query

8.2.4 Advantages of Using Databases

The logical idea behind a database is this: A database is a single collection of data, stored and maintained at one central location, to which many people have access as needed. However, the actual implementation may involve some other physical storage arrangement or access. The essence of a good database is that the users are unaware of the physical arrangements; the unified logical arrangement is all they see. As a result, a database offers many advantages over a simple file system:

- ***Shared access:*** so that many users can use one common, centralized set of data
- ***Minimal redundancy:*** so that individual users do not have to collect and maintain their own

sets of data

- ***Data consistency:*** so that a change to a data value affects all users of the data value
- ***Data integrity:*** so that data values are protected against accidental or malicious undesirable changes
- ***Controlled access:*** so that only authorized users are allowed to view or to modify data values

A DBMS is designed to provide these advantages efficiently. However, as often happens, the objectives can conflict with each other. In particular, security interests can conflict with performance. This clash is not surprising because measures taken to enforce security often increase the computing system's size or complexity. What is surprising, though, is that security interests may also reduce the system's ability to provide data to users by limiting certain queries that would otherwise seem innocuous.

8.3 Security Requirements

The basic security requirements of database systems are not unlike those of other computing systems we have studied about in the previous chapters. The basic problems are access control, exclusion of spurious data, authentication of users, and reliability which have been discussed in earlier chapters are also valid for database systems, along with the following list of requirements for database security.

- ***Physical* database integrity:** The data of a database are immune to physical problems, such as power failures, and someone can reconstruct the database if it is destroyed through a catastrophe.
- ***Logical database integrity:*** The structure of the database is preserved. With logical integrity of a database, a modification to the value of one field does not affect other fields, for example.
- ***Element integrity:*** The data contained in each element are accurate.
- **Auditability:** It is possible to track who or what has accessed (or modified) the elements in the database.
- ***Access control:*** A user is allowed to access only authorized data, and different users can be restricted to different modes of access (such as read or write).
- ***User authentication:*** Every user is positively identified, both for the audit trail and for permission to access certain data.
- ***Availability:*** Users can access the database in general and all the data for which they are authorized.

8.3.1 Integrity of the Database

If a database is to serve as a central repository of data, users must be able to trust the accuracy of the data values. This condition implies that the database administrator must be assured that updates are performed only by authorized individuals. It also implies that the data must be protected from corruption, either by an outside illegal program action or by an outside force such as fire or a power failure. Two situations can affect the integrity of a database: when the whole database is damaged (as happens, for example, if its storage medium is damaged) or when individual data items are unreadable. Integrity of the database as a whole is the responsibility of the DBMS, the operating system, and the (human) computing system manager. From the perspective of the operating system and the computing system manager, databases and DBMSs are files and programs, respectively. Therefore, one way of protecting the database as a whole is to regularly back up all files on the system. These periodic backups can be adequate controls against catastrophic failure. Sometimes it is important to be able to reconstruct the database at the point of a failure. For instance, when the power fails suddenly, a bank's clients may be in the middle of making transactions or students may be in the midst of registering online for their classes. In these cases, we want to be able to restore the systems to a stable point without forcing users to redo their recently completed transactions. To handle these situations, the DBMS must maintain a log of transactions. For example, suppose the

banking system is designed so that a message is generated in a log (electronic or paper or both) each time a transaction is processed. In the event of a system failure, the system can obtain accurate account balances by reverting to a backup copy of the database and reprocessing all later transactions from the log.

8.3.2 Element Integrity

The integrity of database elements is their correctness or accuracy. Ultimately, authorized users are responsible for entering correct data into databases. However, users and programs make mistakes collecting data, computing results, and entering values. Therefore, DBMSs sometimes take special action to help catch errors as they are made and to correct errors after they are inserted.

This corrective action can be taken in three ways. First, the DBMS can apply field checks, activities that test for appropriate values in a position. A field might be required to be numeric, an uppercase letter, or one of a set of acceptable characters. The check ensures that a value falls within specified bounds or is not greater than the sum of the values in two other fields. These checks prevent simple errors as the data are entered.

A second integrity action is provided by access control. To see why, consider life without databases. Data files may contain data from several sources, and redundant data may be stored in several different places. For example, a student's home address may be stored in many different campus files: at class registration, for dining hall privileges, at the bookstore, and in the financial aid office. Indeed, the student may not even be aware that each separate office has the address on file. If the student moves from one residence to another, each of the separate files requires correction. Without a database, there are several risks to the data's integrity. First, at a given time, there could be some data files with the old address (they have not yet been updated) and some simultaneously with the new address (they have already been updated). Second, there is always the possibility that the data fields were changed incorrectly, again leading to files with incorrect information. Third, there may be files of which the student is unaware, so he or she does not know to notify the file owner about updating the address information. These problems are solved by databases. They enable collection and control of this data at one central source, ensuring the student and users of having the correct address. However, the centralization is easier said than done. Who owns this shared central file? Who has authorization to update which elements? What if two people apply conflicting modifications? What if modifications are applied out of sequence? How are duplicate records detected? What action is taken when duplicates are found? These are policy questions that must be resolved by the database administrator by having formal processes which are needed for managing changes in databases.

The third means of providing database integrity is maintaining a change log for the database. A change log lists every change made to the database; it contains both original and modified values. Using this log, a database administrator can undo any changes that were made in error. For example, a library fine might erroneously be posted against Charles W. Robertson, instead of Charles M. Robertson, flagging Charles W. Robertson as ineligible to participate in varsity athletics. Upon discovering this error, the database administrator obtains Charles W's original eligibility value from the log and corrects the database.

8.3.3 Auditability

For some applications it may be desirable to generate an audit record of all access (read or write) to a database. Such a record can help to maintain the database's integrity, or at least to discover after the fact who had affected which values and when. A second advantage is that users can access protected data incrementally; that is, no single access reveals protected data, but a set of sequential accesses viewed together reveals the data, much like discovering the clues in a detective novel. In this case, an audit trail can identify which clues a user has already been given, as a guide to whether to tell the user more. As we noted earlier, granularity becomes an impediment in auditing. Audited events in operating systems are actions like *open file* or *call procedure*; they are seldom as specific

as *write record* 3 or *execute instruction I*. To be useful for maintaining integrity, database audit trails should include accesses at the record, field, and even element levels. This detail is prohibitive for most database applications. Furthermore, it is possible for a record to be accessed but not reported to a user, as when the user performs a select operation. (Accessing a record or an element without transferring to the user the data received is called the pass-through problem.) Also, you can determine the values of some elements without accessing them directly. (For example, you can ask for the average salary in a group of employees when you know the number of employees in the group is only one.) Thus, a log of all records accessed directly may both overstate and understate what a user actually knows.

8.3.4 Access Control

Databases are often separated logically by user access privileges. For example, all users can be granted access to general data, but only the personnel department can obtain salary data and only the marketing department can obtain sales data. Databases are very useful because they centralize the storage and maintenance of data. Limited access is both a responsibility and a benefit of this centralization. The database administrator specifies who should be allowed access to which data, at the view, relation, field, record, or even element level. The DBMS must enforce this policy, granting access to all specified data or no access where prohibited. Furthermore, the number of modes of access can be many. A user or program may have the right to read, change, delete, or append to a value, add or delete entire fields or records, or reorganize the entire database. Superficially, access control for a database seems like access control for operating systems or any other component of a computing system. However, the database problem is more complicated. Operating system objects, such as files, are unrelated items, whereas records, fields, and elements are related. Although a user cannot determine the contents of one file by reading others, a user might be able to determine one data element just by reading others. The problem of obtaining data values from others is called inference, and we consider it in depth later in this chapter. It is important to notice that you can access data by inference without needing direct access to the secure object itself. Restricting inference may mean prohibiting certain paths to prevent possible inferences. However, restricting access to control inference also limits queries from users who do not intend unauthorized access to values. Moreover, attempts to check requested accesses for possible unacceptable inferences may actually degrade the DBMS's performance. Finally, size or granularity is different between operating system objects and database objects. An access control list of several hundred files is much easier to implement than an access control list for a database with several hundred files of perhaps a hundred fields each. Size affects the efficiency of processing.

8.3.5 User Authentication

The DBMS can require rigorous user authentication. For example, a DBMS might insist that a user pass both specific password and time-of-day checks. This authentication supplements the authentication performed by the operating system. Typically, the DBMS runs as an application program on top of the operating system. This system design means that there is no trusted path from the DBMS to the operating system, so the DBMS must be suspicious of any data it receives, including user authentication. Thus, the DBMS is forced to do its own authentication.

8.3.6 Availability

A DBMS has aspects of both a program and a system. It is a program that uses other hardware and software resources, yet to many users it is the only application run. Users often take the DBMS for granted, employing it as an essential tool with which to perform particular tasks. But when the system is not available, busy serving other users or down to be repaired or upgraded, the users are very aware of a DBMS's unavailability. For example, two users may request the same record, and the DBMS must arbitrate; one user is bound to be denied access for a while. Or the DBMS may withhold unprotected data to avoid revealing protected data, leaving the requesting user unhappy.

Problems like these result in high availability requirements for a DBMS.

8.3.7 Integrity/Confidentiality/Availability

The three aspects of computer security - integrity, confidentiality, and availability, clearly relate to database management systems. As we have described, integrity applies to the individual elements of a database as well as to the database as a whole. Thus, integrity is a major concern in the design of database management systems. We look more closely at integrity issues in the next section. Confidentiality is a key issue with databases because of the inference problem, whereby a user can access sensitive data indirectly. Inference and access control are covered later in this chapter. Finally, availability is important because of the shared access motivation underlying database development. However, availability conflicts with confidentiality. The last sections of the chapter address availability in an environment in which confidentiality is also important.

8.4 Reliability and Integrity

Databases amalgamate data from many sources, and users expect a DBMS to provide access to the data in a reliable way. When software engineers say that software has reliability, they mean that the software runs for very long periods of time without failing. Users certainly expect a DBMS to be reliable, since the data usually are key to business or organizational needs. Moreover, users entrust their data to a DBMS and rightly expect it to protect the data from loss or damage. Concerns for reliability and integrity are general security issues, but they are more apparent with databases. A DBMS guards against loss or damage in several ways. However, the controls we consider are not absolute: No control can prevent an authorized user from inadvertently entering an acceptable but incorrect value. Database concerns about reliability and integrity can be viewed from three dimensions:

Database integrity: *This is* concerned that the database as a whole is protected against damage, as from the failure of a disk drive or the corruption of the master database index. These concerns are addressed by operating system integrity controls and recovery procedures.

Element integrity: *This is* concerned that the value of a specific data element is written or changed only by authorized users. Proper access controls protect a database from corruption by unauthorized users.

Element accuracy: *This is* concerned that only correct values are written into the elements of a database. Checks on the values of elements can help prevent insertion of improper values. Also, constraint conditions can detect incorrect values.

8.4.1 Protection Features from the Operating System

In an earlier chapter, we discussed the protection an operating system provides for its users. A responsible system administrator backs up the files of a database periodically along with other user files. The files are protected during normal execution against outside access by the operating system's standard access control facilities. Finally, the operating system performs certain integrity checks for all data as a part of normal read and write operations for I/O devices. These controls provide basic security for databases, but the database manager must enhance them.

8.4.2 Two-Phase Update

A serious problem for a database manager is the failure of the computing system in the middle of modifying data. If the data item to be modified was a long field, half of the field might show the new value, while the other half would contain the old. Even if errors of this type were spotted easily (which they are not), a more subtle problem occurs when several fields are updated and no single field appears to be in obvious error. The solution to this problem, proposed first by Lampson and Sturgis and adopted by most DBMSs, uses a two-phase update.

8.4.2.1 Update Technique

During the first phase, called the intent phase, the DBMS gathers the resources it needs to perform the update. It may gather data, create dummy records, open files, lock out other users, and calculate final answers; in short, it does everything to prepare for the update, but it makes no changes to the database. The first phase is repeatable an unlimited number of times because it takes no permanent action. If the system fails during execution of the first phase, no harm is done because all these steps can be restarted and repeated after the system resumes processing. The last event of the first phase, called committing, involves the writing of a commit flag to the database. The commit flag means that the DBMS has passed the point of no return: After committing, the DBMS begins making permanent changes.

The second phase makes the permanent changes. During the second phase, no actions from before the commit can be repeated, but the update activities of phase two can also be repeated as often as needed. If the system fails during the second phase, the database may contain incomplete data, but the system can repair these data by performing all activities of the second phase. After the second phase has been completed, the database is again complete.

8.4.2.2 Two-Phase Update Example

Suppose a database contains an inventory of a company's office supplies. The company's central stockroom stores paper, pens, paper clips, and the like, and the different departments requisition items as they need them. The company buys in bulk to obtain the best prices. Each department has a budget for office supplies, so there is a charging mechanism by which the cost of supplies is recovered from the department. Also, the central stockroom monitors quantities of supplies on hand so as to order new supplies when the stock becomes low.

Suppose the process begins with a requisition from the accounting department for 50 boxes of paper clips. Assume that there are 107 boxes in stock and a new order is placed if the quantity in stock ever falls below 100. Here are the steps followed after the stockroom receives the requisition.

1. The stockroom checks the database to determine that 50 boxes of paper clips are on hand. If not, the requisition is rejected and the transaction is finished.
2. If enough paper clips are in stock, the stockroom deducts 50 from the inventory figure in the database (107 - 50 = 57).
3. The stockroom charges accounting's supplies budget (also in the database) for 50 boxes of paper clips.
4. The stockroom checks its remaining quantity on hand (57) to determine whether the remaining quantity is below the reorder point. Because it is, a notice to order more paper clips is generated, and the item is flagged as "on order" in the database.
5. A delivery order is prepared, enabling 50 boxes of paper clips to be sent to accounting.

All five of these steps must be completed in the order listed for the database to be accurate and for the transaction to be processed correctly.

Suppose a failure occurs while these steps are being processed. If the failure occurs before step 1 is complete, there is no harm because the entire transaction can be restarted. However, during steps 2, 3, and 4, changes are made to elements in the database. If a failure occurs then, the values in the database are inconsistent. Worse, the transaction cannot be reprocessed because a requisition would be deducted twice, or a department would be charged twice, or two delivery orders would be prepared.

When a two-phase commit is used, shadow values are maintained for key data points. A shadow data value is computed and stored locally during the intent phase, and it is copied to the actual database during the commit phase. The operations on the database would be performed as follows for a two-phase commit.

Intent:

1. Check the value of COMMIT-FLAG in the database. If it is set, this phase cannot be performed.

Halt or loop, checking COMMIT-FLAG until it is not set.
2. Compare number of boxes of paper clips on hand to number requisitioned; if more are requisitioned than are on hand, halt.
3. Compute TCLIPS = ONHAND - REQUISITION.
4. Obtain BUDGET, the current supplies budget remaining for accounting department. Compute TBUDGET = BUDGET - COST, where COST is the cost of 50 boxes of clips.
5. Check whether TCLIPS is below reorder point; if so, set TREORDER = TRUE; else set TREORDER = FALSE.

Commit:
1. Set COMMIT-FLAG in database.
2. Copy TCLIPS to CLIPS in database.
3. Copy TBUDGET to BUDGET in database.
4. Copy TREORDER to REORDER in database.
5. Prepare notice to deliver paper clips to accounting department. Indicate transaction completed in log.
6. Unset COMMIT-FLAG.

With this example, each step of the intent phase depends only on unmodified values from the database and the previous results of the intent phase. Each variable beginning with T is a shadow variable used only in this transaction. The steps of the intent phase can be repeated an unlimited number of times without affecting the integrity of the database.
Once the DBMS begins the commit phase, it writes a commit flag. When this flag is set, the DBMS will not perform any steps of the intent phase. Intent steps cannot be performed after committing because database values are modified in the commit phase. Notice, however, that the steps of the commit phase can be repeated an unlimited number of times, again with no negative effect on the correctness of the values in the database.
The one remaining flaw in this logic occurs if the system fails after writing the "transaction complete" message in the log but before clearing the commit flag in the database. It is a simple matter to work backward through the transaction log to find completed transactions for which the commit flag is still set and to clear those flags.

8.4.3 Redundancy/Internal Consistency

Many DBMSs maintain additional information to detect internal inconsistencies in data. The additional information ranges from a few check bits to duplicate or shadow fields, depending on the importance of the data.

8.4.3.1 Error Detection and Correction Codes

One form of redundancy is error detection and correction codes, such as parity bits, Hamming codes, and cyclic redundancy checks. These codes can be applied to single fields, records, or the entire database. Each time a data item is placed in the database, the appropriate check codes are computed and stored; each time a data item is retrieved, a similar check code is computed and compared to the stored value. If the values are unequal, they signify to the DBMS that an error has occurred in the database. Some of these codes point out the place of the error; others show precisely what the correct value should be. The more information provided, the more space required to store the codes.

8.4.3.2 Shadow Fields

Entire attributes or entire records can be duplicated in a database. If the data are irreproducible, this second copy can provide an immediate replacement if an error is detected. Obviously, redundant fields require substantial storage space.

8.4.4 Recovery

In addition to these error correction processes, a DBMS can maintain a log of user accesses, particularly changes. In the event of a failure, the database is reloaded from a backup copy and all later changes are then applied from the audit log.

8.4.5 Concurrency/Consistency

Database systems are often multi-user systems. Accesses by two users sharing the same database must be constrained so that neither interferes with the other. Simple locking is done by the DBMS. If two users attempt to read the same data item, there is no conflict because both obtain the same value. If both users try to modify the same data items, we often assume that there is no conflict because each knows what to write; the value to be written does not depend on the previous value of the data item. However, this supposition is not quite accurate.

To see how concurrent modification can get us into trouble, suppose that the database consists of seat reservations for a particular airline flight. Agent A, booking a seat for passenger Mock, submits a query to find which seats are still available. The agent knows that Mock prefers a right aisle seat, and the agent finds that seats `5D, 11D, and 14D` are open. At the same time, Agent B is trying to book seats for a family of three traveling together. In response to a query, the database indicates that `8ABC` and `11DEF` are the two remaining groups of three adjacent unassigned seats.

Agent A submits the update command

```
SELECT (SEAT-NO = '11D')
ASSIGN 'MOCK,E' TO PASSENGER-NAME
```

while Agent B submits the update sequence

```
SELECT (SEAT-NO = '11D')
ASSIGN 'EHLERS,P' TO PASSENGER-NAME
```

as well as commands for seats `11E` and `11F`. Then two passengers have been booked into the same seat (which would be uncomfortable, to say the least). Both agents have acted properly: Each sought a list of empty seats, chose one seat from the list, and updated the database to show to whom the seat was assigned. The difficulty in this situation is the time delay between reading a value from the database and writing a modification of that value. During the delay time, another user has accessed the same data.

To resolve this problem, a DBMS treats the entire query-update cycle as a single atomic operation. The command from the agent must now resemble "read the current value of seat `PASSENGER-NAME` for seat `11D`; if it is `'UNASSIGNED'`, modify it to `'MOCK,E' (or 'EHLERS,P')`". The read-modify cycle must be completed as an uninterrupted item without allowing any other users access to the `PASSENGER-NAME` field for seat `11D`. The second agent's request to book would not be considered until after the first agent's had been completed; at that time, the value of `PASSENGERNAME` would no longer be `'UNASSIGNED'`.

A final problem in concurrent access is read-write. Suppose one user is updating a value when a second user wishes to read it. If the read is done while the write is in progress, the reader may receive data that are only partially updated. Consequently, the DBMS locks any read requests until a write has been completed.

8.4.6 Monitors

The monitor is the unit of a DBMS responsible for the structural integrity of the database. A monitor can check values being entered to ensure their consistency with the rest of the database or with characteristics of the particular field. For example, a monitor might reject alphabetic characters

for a numeric field. We discuss several forms of monitors.

8.4.6.1 Range Comparisons

A range comparison monitor tests each new value to ensure that the value is within an acceptable range. If the data value is outside the range, it is rejected and not entered into the database. For example, the range of dates might be 131, "/," 112, "/," 19002099. An even more sophisticated range check might limit the day portion to 130 for months with 30 days, or it might take into account leap year for February. Range comparisons are also convenient for numeric quantities. For example, a salary field might be limited to $200,000, or the size of a house might be constrained to be between 500 and 5,000 square feet. Range constraints can also apply to other data having a predictable form.

Range comparisons can be used to ensure the internal consistency of a database. When used in this manner, comparisons are made between two database elements. For example, a grade level from A+ would be acceptable if the record described a student at an elementary school, whereas only 912 would be acceptable for a record of a student in high school. Similarly, a person could be assigned a job qualification score of 75100 only if the person had completed college or had had at least ten years of work experience. Filters or patterns are more general types of data form checks. These can be used to verify that an automobile plate is two letters followed by four digits, or the sum of all digits of a credit card number is a multiple of 9. Checks of these types can control the data allowed in the database. They can also be used to test existing values for reasonableness. If you suspect that the data in a database have been corrupted, a range check of all records could identify those having suspicious values.

8.4.6.2 State Constraints

State constraints describe the condition of the entire database. At no time should the database values violate these constraints. Phrased differently, if these constraints are not met, some value of the database is in error.

In the section on two-phase updates, we saw how to use a commit flag, which is set at the start of the commit phase and cleared at the completion of the commit phase. The commit flag can be considered a state constraint because it is used at the end of every transaction for which the commit flag is not set. Earlier in this chapter, we described a process to reset the commit flags in the event of a failure after a commit phase. In this way, the status of the commit flag is an integrity constraint on the database. For another example of a state constraint, consider a database of employees' classifications. At any time, at most one employee is classified as "president". Furthermore, each employee has an employee number different from that of every other employee. If a mechanical or software failure causes portions of the database file to be duplicated, one of these uniqueness constraints might be violated. By testing the state of the database, the DBMS could identify records with duplicate employee numbers or two records classified as "president".

8.4.6.3 Transition Constraints

State constraints describe the state of a correct database. Transition constraints describe conditions necessary before changes can be applied to a database. For example, before a new employee can be added to the database, there must be a position number in the database with status "vacant." (That is, an empty slot must exist.) Furthermore, after the employee is added, exactly one slot must be changed from "vacant" to the number of the new employee. Simple range checks and filters can be implemented within most database management systems. However, the more sophisticated state and transition constraints can require special procedures for testing. Such user-written procedures are invoked by the DBMS each time an action must be checked.

8.5 Sensitive Data

Some databases contain what is called sensitive data. As a working definition, let us say that sensitive data are data that should not be made public. Determining which data items and fields are sensitive depends both on the individual database and the underlying meaning of the data. Obviously, some databases, such as a public library catalog, contain no sensitive data; other databases, such as defense-related ones, are totally sensitive. These two cases nothing sensitive and everything sensitive are the easiest to handle because they can be covered by access controls to the database as a whole. Someone either is or is not an authorized user. These controls are provided by the operating system. The more difficult problem, which is also the more interesting one, is the case in which *some but not all* of the elements in the database are sensitive. There may be varying degrees of sensitivity. For example, a university database might contain student data consisting of name, financial aid, dorm, drug use, sex, parking fines, and race. An example of this database is shown in Table 8-5. Name and dorm are probably the least sensitive; financial aid, parking fines, and drug use the most; sex and race somewhere in between. That is, many people may have legitimate access to name, some to sex and race, and relatively few to financial aid, parking fines, or drug use. Indeed, knowledge of the existence of some fields, such as drug use, may itself be sensitive. Thus, security concerns not only the data elements but also their context and meaning. Furthermore, we must take into account different degrees of sensitivity. For instance, although they are all highly sensitive, the financial aid, parking fines, and drug-use fields may not have the same kinds of access restrictions. Our security requirements may demand that a few people be authorized to see each field, but no one be authorized to see all three. The challenge of the access control problem is to limit users' access so that they can obtain only the data to which they have legitimate access. Alternatively, the access control problem forces us to ensure that sensitive data are not to be released to unauthorized people. Several factors can make data sensitive.

- ***Inherently sensitive:*** The value itself may be so revealing that it is sensitive. Examples are the locations of defensive missiles or the median income of barbers in a town with only one barber.
- ***From a sensitive source:*** The source of the data may indicate a need for confidentiality. An example is information from an informer whose identity would be compromised if the information were disclosed.
- ***Declared sensitive:*** The database administrator or the owner of the data may have declared the data to be sensitive. Examples are classified military data or the name of the anonymous donor of a piece of art.
- **Part of a sensitive *attribute* or a sensitive *record:*** In a database, an entire attribute or record may be classified as sensitive. Examples are the salary attribute of a personnel database or a record describing a secret space mission.
- **Sensitive *in relation to previously disclosed information:*** Some data become sensitive in the presence of other data. For example, the longitude coordinate of a secret gold mine reveals little, but the longitude coordinate in conjunction with the latitude coordinate pinpoints the mine.

All of these factors must be considered to determine the sensitivity of the data.

8.5.1 Access Decisions

Remember that a database administrator is a person who decides what data should be in the database and who should have access to it. The database administrator considers the need for different users to know certain information and decides who should have what access. Decisions of the database administrator are based on an access policy. The database manager or DBMS is a program that operates on the database and auxiliary control information to implement the decisions of the access policy. We say that the database manager decides to permit user x to access data y. Clearly, a program or machine cannot decide anything; it is more precise to say that the program performs the instructions by which x accesses y as a way of implementing the policy established by the database administrator. To keep explanations concise, we occasionally describe programs as if

they can carry out human thought processes. The DBMS may consider several factors when deciding whether to permit an access. These factors include availability of the data, acceptability of the access, and authenticity of the user. We expand on these three factors below.

8.5.1.1 Availability of Data

One or more required elements may be inaccessible. For example, if a user is updating several fields, other users' accesses to those fields must be blocked temporarily. This blocking ensures that users do not receive inaccurate information, such as a new street address with an old city and state, or a new code component with old documentation. Blocking is usually temporary. When performing an update, a user may have to block access to several fields or several records to ensure the consistency of data for others. Notice, however, that if the updating user aborts the transaction while the update is in progress, the other users may be permanently blocked from accessing the record. This indefinite postponement is also a security problem, resulting in denial of service.

8.5.1.2 Acceptability of Access

One or more values of the record may be sensitive and not accessible by the general user. A DBMS should not release sensitive data to unauthorized individuals. Deciding what is sensitive, however, is not as simple as it sounds, because the fields may not be directly requested. A user may have asked for certain records that contain sensitive data, but the user's purpose may have been only to project the values from particular fields that are not sensitive. Even when a sensitive value is not explicitly given, the database manager may deny access on the grounds that it reveals information the user is not authorized to have. Alternatively, the user may want to derive a non-sensitive statistic from the sensitive data; for example, if the average financial aid value does not reveal any individual's financial aid value, the database management system can safely return the average. However, the average of one data value discloses that value.

8.5.1.3 Assurance of Authenticity

Certain characteristics of the user external to the database may also be considered when permitting access. For example, to enhance security, the database administrator may permit someone to access the database only at certain times, such as during working hours. Previous user requests may also be taken into account; repeated requests for the same data or requests that exhaust a certain category of information may be used to find out all elements in a set when a direct query is not allowed. Sensitive data can sometimes be revealed by combined results from several less sensitive queries.

8.5.2 Types of Disclosures

Data can be sensitive, but so can their characteristics. In this section, we see that even descriptive information about data (such as their existence or whether they have an element that is zero) is a form of disclosure.

8.5.2.1 Exact Data

The most serious disclosure is the *exact value of a sensitive data item* itself. The user may know that sensitive data are being requested, or the user may request general data without knowing that some of it is sensitive. A faulty database manager may even deliver sensitive data by accident, without the user's having requested it. In all of these cases the result is the same: The security of the sensitive data has been breached.

8.5.2.2 Bounds

Another exposure is disclosing bounds on a sensitive value; that is, indicating that a sensitive value, y, is between two values, L and H. Sometimes, by using a narrowing technique not unlike the binary search, the user may first determine that $L \leq y \leq H$ and then see whether $L \leq y \leq H/2$, and so forth, thereby permitting the user to determine y to any desired precision. In another case, merely

revealing that a value such as the athletic scholarship budget or the number of CIA agents exceeds a certain amount may be a serious breach of security.

8.5.2.3 Negative Result

Sometimes we can word a query to determine a negative result. That is, we can learn that *z* is *not* the value of *y*. For example, knowing that 0 is not the total number of felony convictions for a person reveals that the person was convicted of a felony. The distinction between 1 and 2 or 46 and 47 felonies is not as sensitive as the distinction between 0 and 1. Therefore, disclosing that a value is not 0 can be a significant disclosure. Similarly, if a student does not appear on the honors list, you can infer that the person's grade point average is below 3.50. This information is not too revealing, however, because the range of grade point averages from 0.0 to 3.49 is rather wide.

8.5.2.4 Existence

In some cases, the existence of data is itself a sensitive piece of data, regardless of the actual value. For example, an employer may not want employees to know that their use of long distance telephone lines is being monitored. In this case, discovering a LONG DISTANCE field in a personnel file would reveal sensitive data.

8.5.2.5 Probable Value

Finally, it may be possible to determine the probability that a certain element has a certain value. To see how, suppose you want to find out whether the president of the United States is registered in the Tory party. Knowing that the president is in the database, you submit two queries to the database:

- How many people have 1600 Pennsylvania Avenue as their official residence? (Response: 4)
- How many people have 1600 Pennsylvania Avenue as their official residence and have YES as the value of TORY? (Response: 1)

From these queries you conclude there is a 25 percent likelihood that the president is a registered Tory.

8.5.2.6 Summary of Partial Disclosure

We have seen several examples of how a security problem can result if characteristics of sensitive data are revealed. Notice that some of the techniques we presented used information *about* the data, rather than direct access to the data, to infer sensitive results. A successful security strategy must protect from both direct and indirect disclosure.

8.5.3 Security versus Precision

Our examples have illustrated how difficult it is to determine which data are sensitive and how to protect them. The situation is complicated by a desire to share non-sensitive data. For reasons of confidentiality we want to disclose only those data that are not sensitive. Such an outlook encourages a conservative philosophy in determining what data to disclose: less is better than more. On the other hand, consider the users of the data. The conservative philosophy suggests rejecting any query that mentions a sensitive field. We may thereby reject many reasonable and non-disclosing queries. For example, a researcher may want a list of grades for all students using drugs, or a statistician may request lists of salaries for all men and for all women. These queries probably do not compromise the identity of any individual. We want to disclose as much data as possible so that users of the database have access to the data they need. This goal, called precision, aims to protect all sensitive data while revealing as much non-sensitive data as possible.

We can depict the relationship between security and precision with concentric circles. As Figure 8-2 shows, the sensitive data in the central circle should be carefully concealed. The outside band represents data we willingly disclose in response to queries. But we know that the user may put together pieces of disclosed data and infer other, more deeply hidden, data. The figure shows us that

beneath the outer layer may be yet more non-sensitive data that the user cannot infer. The ideal combination of security and precision allows us to maintain perfect confidentiality with maximum precision; in other words, we disclose all and only the non-sensitive data. But achieving this goal is not as easy as it might seem.

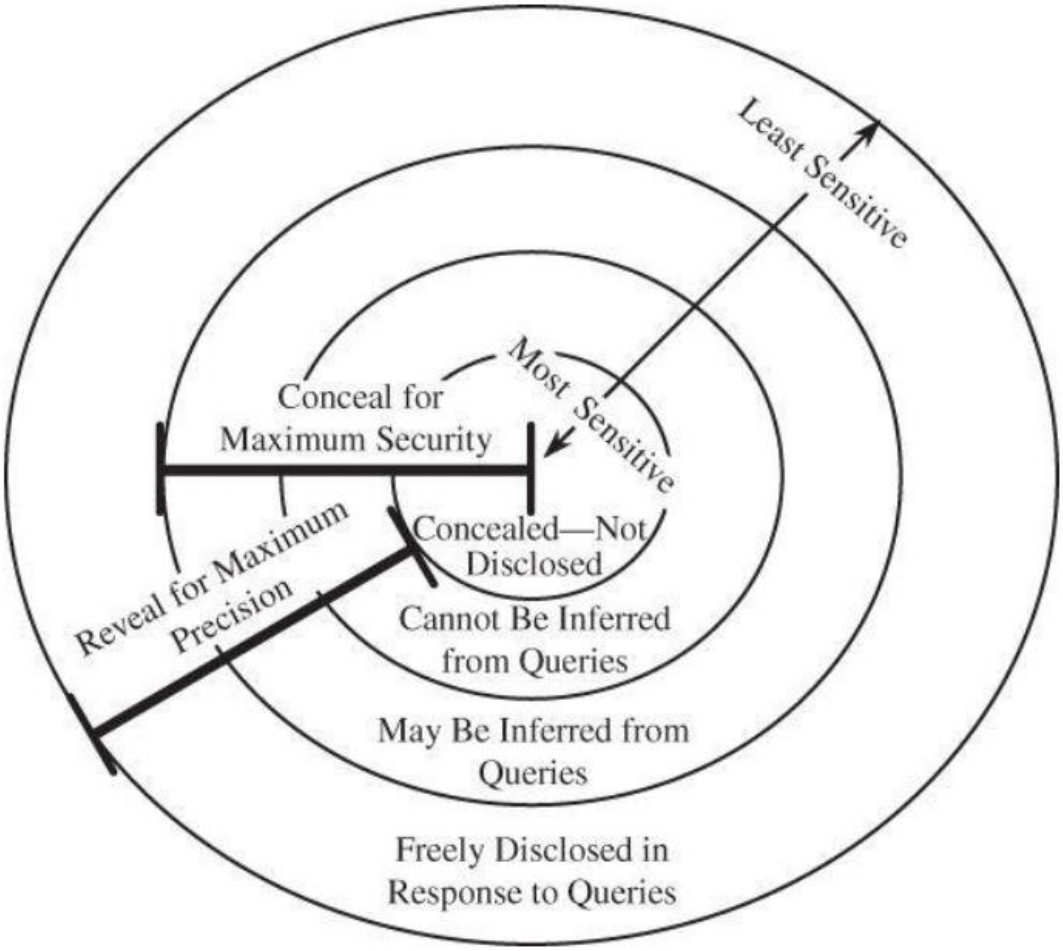


Figure 8-2 Security versus Precision

8.6 Summary

- Reliability, correctness, and integrity are three closely related concepts in databases. Users trust the DBMS to maintain their data correctly, so integrity issues are very important to database security. This chapter has addressed the confidentiality and integrity problems specific to database applications for database management systems.
- Both confidentiality and integrity are important to users of databases. Confidentiality can be broken by indirect disclosure of a negative result or of the bounds of a value. Integrity of the entire database is a responsibility of the DBMS software; this problem is handled by most major commercial systems through backups, redundancy, change logs, and two-step updates.
- Integrity of an individual element of the database is the responsibility of the database administrator who defines the access policy.

8.7 Review Questions

a) What are the components of a database?
b) Discuss the advantages of using databases.
c) What are the requirements for database security?

d) Write a short note on the element integrity for database security.
e) Explain the two-phase update technique for database integrity.
f) How can concurrent access and consistency be maintained in databases?
g) Write a short note on monitors.
h) What is sensitive data? Explain the factors that make data sensitive.
i) What are the different types of disclosures of data?
j) Differentiate between security and precision.

8.8 Bibliography, References and Further Reading

- *Security in Computing by* C. P. Pfleeger, and S. L. Pfleeger, Pearson Education.
- *Computer Security: Art and Science by* Matt Bishop, Pearson Education.
- *Cryptography And Network Security: Principles and practice by* Stallings
- *Network Security by* Kaufman, Perlman, Speciner
- *Network Security : A Beginner's Guide by* Eric Maiwald, TMH
- *Java Network Security by* Macro Pistoia, Pearson Education
- *Principles of information security by* Whitman, Mattord, Thomson

IDOL

Chapter 9

Database Security - II

IDOL

9.0 Objectives

In this chapter, we study two major (but related) database security problems, the inference problem and the multilevel problem. Both problems are complex, and there are no immediate solutions. However, by understanding the problems, we become more sensitive to ways of reducing potential threats to the data.

9.1 Introduction

Protecting data is at the heart of many secure systems, and many users (people, programs, or systems) rely on a database management system (DBMS) to manage the protection. There is substantial current interest in DBMS security because databases are newer than programming and operating systems. However, the protection provided by database management systems has had mixed results. Over time, we have improved our understanding of database security problems, and several good controls have been developed. But, as you will see, there are still more security concerns for which there are no available controls.

9.2 Inference

Inference is a way to infer or derive sensitive data from non-sensitive data. The inference problem is a subtle vulnerability in database security. The database in Table 9-1 can help illustrate the inference problem. AID is the amount of financial aid a student is receiving. FINES is the amount of parking fines still owed. DRUGS is the result of a drug-use survey: 0 means never used and 3 means frequent user. Obviously, this information should be kept confidential. We assume that AID, FINES, and DRUGS are sensitive fields, although only when the values are related to a specific individual. In this section, we look at ways to determine sensitive data values from the database.

IDOL

Name	Sex	Race	Aid	Fines	Drugs	Dorm
Adams	M	C	5000	45.	1	Holmes
Bailey	M	B	0	0.	0	Grey
Chin	F	A	3000	20.	0	West
Dewitt	M	B	1000	35.	3	Grey
Earhart	F	C	2000	95.	1	Holmes
Fein	F	C	1000	15.	0	West
Groff	M	C	4000	0.	3	West
Hill	F	B	5000	10.	2	Holmes
Koch	F	C	0	0.	1	West
Liu	F	A	0	10.	2	Grey
Majors	M	C	2000	0.	2	Grey

Table 9-1 Database to illustrate inferences

9.2.1 Direct Attack

In a direct attack, a user tries to determine values of sensitive fields by seeking them directly with queries that yield few records. The most successful technique is to form a query so specific that it matches exactly one data item. In Table 9-1, a sensitive query might be

```
List NAME where
     SEX=M ∧ DRUGS=1
```

This query discloses that for record ADAMS, DRUGS=1. However, it is an obvious attack because it selects people for whom DRUGS=1, and the DBMS might reject the query because it selects records for a specific value of the sensitive attribute DRUGS. A less obvious query is

```
List NAME where
      (SEX = M ∧ DRUGS = 1) ∨
      (SEX ≠ M ∧ SEX ≠ F) ∨
      (DORM = AYRES)
```

On the surface, this query looks as if it should conceal drug usage by selecting other non-drug-related records as well. However, this query still retrieves only one record, revealing a name that corresponds to the sensitive DRUG value. The DBMS needs to know that SEX has only two possible values so that the second clause will select no records. Even if that were possible, the DBMS would also need to know that no records exist with DORM=AYRES, even though AYRES might in fact be an acceptable value for DORM. Organizations that publish personal statistical data, such as the U.S. Census Bureau, do not reveal results when a small number of people make up a large proportion of a category. The rule of "*n* items over *k* percent" means that data should be withheld if *n* items represent over *k* percent of the result reported. In the previous case, the one person selected represents 100 percent of the data reported, so there would be no ambiguity about which person matches the query.

9.2.2 Indirect Attack

Another procedure, used by the U.S. Census Bureau and other organizations that gather sensitive data, is to release only statistics. The organizations suppress individual names, addresses, or other characteristics by which a single individual can be recognized. Only neutral statistics, such as sum, count, and mean, are released. The indirect attack seeks to infer a final result based on one or more intermediate statistical results. But this approach requires work outside the database itself. In particular, a statistical attack seeks to use some apparently anonymous statistical measure to infer individual data. In the following sections, we present several examples of indirect attacks on databases that report statistics.

9.2.2.1 Sum

An attack by sum tries to infer a value from a reported sum. For example, with the sample database in Table 9-1, it might seem safe to report student aid total by sex and dorm. Such a report is shown in Table 9-2. This seemingly innocent report reveals that no female living in Grey is receiving financial aid. Thus, we can infer that any female living in Grey (such as Liu) is certainly not receiving financial aid. This approach often allows us to determine a negative result.

	Holmes	Grey	West	Total
M	5000	3000	4000	12000
F	7000	0	4000	11000
Total	12000	3000	8000	23000

Table 9-2 Table showing negative result

9.2.2.2 Count

The count can be combined with the sum to produce some even more revealing results. Often these two statistics are released for a database to allow users to determine average values. (Conversely, if count and mean are released, sum can be deduced.) Table 9-3 shows the count of records for students by dorm and sex. This table is innocuous by itself. Combined with the sum table, however, this table demonstrates that the two males in Holmes and West are receiving financial aid in the amount of $5000 and $4000, respectively. We can obtain the names by selecting the subschema of NAME, DORM, which is not sensitive because it delivers only low-security data on the entire database.

Sex	Holmes	Grey	West	Total
M	1	3	1	5
F	2	1	3	6
Total	3	4	4	11

Table 9-3 Inference from Count and Sum Results

9.2.2.3 Mean

The arithmetic mean (average) allows exact disclosure if the attacker can manipulate the subject population. As a trivial example, consider salary. Given the number of employees, the mean salary

for a company and the mean salary of all employees except the president, it is easy to compute the president's salary.

9.2.2.4 Median

By a slightly more complicated process, we can determine an individual value from the **median**, the midpoint of an ordered list of values. The attack requires finding selections having one point of intersection that happens to be exactly in the middle. For example, in our sample database, there are five males and three persons whose drug use value is 2. Arranged in order of aid, these lists are shown in table 9-4. Notice that Majors is the only name common to both lists, and conveniently that name is in the middle of each list. Someone working at the Health Clinic might be able to find out that Majors is a white male whose drug-use score is 2. That information identifies Majors as the intersection of these two lists and pinpoints Majors' financial aid as $2000. In this example, the queries

```
q = median(AID where SEX = M)
p = median(AID where DRUGS = 2)
```

reveal the exact financial aid amount for Majors.

Name	Sex	Drugs	Aid
Bailey	M	0	0
Dewitt	M	3	1000
Majors	M	2	2000
Groff	M	3	4000
Adams	M	1	5000
Liu	F	2	0
Majors	M	2	2000
Hill	F	2	5000

Table 9-4 Drug Use and Aid Result

9.2.2.5 Tracker Attacks

Database management systems may conceal data when a small number of entries make up a large proportion of the data revealed. A tracker attack can fool the database manager into locating the desired data by using additional queries that produce small results. The tracker adds additional records to be retrieved for two different queries; the two sets of records cancel each other out, leaving only the statistic or data desired. The approach is to use intelligent padding of two queries. In other words, instead of trying to identify a unique value, we request n - 1 other values (where there are n values in the database). Given n and n - 1, we can easily compute the desired single element.

9.2.2.6 Linear System Vulnerability

A tracker is a specific case of a more general vulnerability. With a little logic, algebra, and luck in the distribution of the database contents, it may be possible to construct a series of queries that

returns results relating to several different sets. For example, the following system of five queries does not overtly reveal any single *c* value from the database. However, the queries' equations can be solved for each of the unknown *c* values, revealing them all. In fact, this attack can also be used to obtain results *other than* numerical ones. Recall that we can apply logical rules to *and* (∧) and *or* (∨), typical operators for database queries, to derive values from a series of logical expressions. For example, each expression might represent a query asking for precise data instead of counts.

9.2.2.7 Controls for Statistical Inference Attacks

Denning and Schlörer present a very good survey of techniques for maintaining security in databases. The controls for all statistical attacks are similar. Essentially, there are two ways to protect against inference attacks: Either controls are applied to the queries or controls are applied to individual items within the database. As we have seen, it is difficult to determine whether a given query discloses sensitive data. Thus, query controls are effective primarily against direct attacks.

Suppression and concealing are two controls applied to data items. With suppression, sensitive data values are not provided; the query is rejected without response. With concealing, the answer provided is *close to* but not exactly the actual value. These two controls reflect the contrast between security and precision. With suppression, any results provided are correct, yet many responses must be withheld to maintain security. With concealing, more results can be provided, but the precision of the results is lower. The choice between suppression and concealing depends on the context of the database.

9.2.2.8 Limited Response Suppression

The *n*-item *k*-percent rule eliminates certain low-frequency elements from being displayed. It is not sufficient to delete them, however, if their values can also be inferred. When one cell is suppressed in a table with totals for rows and columns, it is necessary to suppress at least one additional cell on the row and one on the column to provide some confusion. Using this logic, all cells (except totals) would have to be suppressed in this small sample table. When totals are not provided, single cells in a row or column can be suppressed.

9.2.2.9 Combined Results

Another control combines rows or columns to protect sensitive values. For example, Table 9-5 shows several sensitive results that identify single individuals. These counts, combined with other results such as sum, permit us to infer individual drug-use values for the three males, as well as to infer that no female was rated 3 for drug use. To suppress such sensitive information, it is possible to combine the attribute values for 0 and 1, and also for 2 and 3, producing the less sensitive results shown in Table 9-6. In this instance, it is impossible to identify any single value.

Table 9-5 Students by Sex and Drug Use

	Drug Use			
Sex	**0**	**1**	**2**	**3**
M	1	1	1	2
F	2	2	2	0

Table 9-6 Suppression by Combining Revealing Values

	Drug Use	
Sex	**0 or 1**	**2 or 3**
M	2	3
F	4	2

9.2.2.10 Conclusion on the Inference Problem

There are no perfect solutions to the inference problem. The approaches to controlling it follow the three paths listed below. The first two methods can be used either to limit queries accepted or to limit data provided in response to a query. The last method applies only to data released.

- *Suppress obviously sensitive information:* This action can be taken fairly easily. The tendency is to err on the side of suppression, thereby restricting the usefulness of the database.
- T*rack what the user knows:* Although possibly leading to the greatest safe disclosure, this approach is extremely costly. Information must be maintained on all users, even though most are not trying to obtain sensitive data. Moreover, this approach seldom takes into account what any two people may know together and cannot address what a single user can accomplish by using multiple IDs.
- *Disguise the data:* Random perturbation and rounding can inhibit statistical attacks that depend on exact values for logical and algebraic manipulation. The users of the database receive slightly incorrect or possibly inconsistent results.

It is unlikely that research will reveal a simple, easy-to-apply measure that determines exactly which data can be revealed without compromising sensitive data. Nevertheless, an effective control for the inference problem is just knowing that it exists. As with other problems in security, recognition of the problem leads to understanding of the purposes of controlling the problem and to sensitivity to the potential difficulties caused by the problem.

9.2.3 Aggregation

Related to the inference problem is aggregation, which means building sensitive results from less sensitive inputs. We saw earlier that knowing either the latitude or longitude of a gold mine does you no good. But if you know both latitude and longitude, you can pinpoint the mine. For a more realistic example, consider how police use aggregation frequently in solving crimes: They determine who had a motive for committing the crime, when the crime was committed, who had alibis covering that time, who had the skills, and so forth. Typically, you think of police investigation as starting with the entire population and narrowing the analysis to a single person. But if the police officers work in parallel, one may have a list of possible suspects, another may have a list with possible motive, and another may have a list of capable persons. When the intersection of these lists is a single person, the police have their prime suspect.

Addressing the aggregation problem is difficult because it requires the database management system to track which results each user had already received and conceal any result that would let the user derive a more sensitive result. Aggregation is especially difficult to counter because it can take place outside the system. For example, suppose the security policy is that anyone can have *either* the latitude or longitude of the mine, but not both. Nothing prevents you from getting one, your friend from getting the other, and the two of you talking to each other.

Recent interest in data mining has raised concern again about aggregation. Data mining is the process of sifting through multiple databases and correlating multiple data elements to find useful information. Marketing companies use data mining extensively to find consumers likely to buy a product.

Aggregation was of interest to database security researchers at the same time as was inference. As we have seen, some approaches to inference have proven useful and are currently being used. But

there have been few proposals for countering aggregation.

9.3 Multilevel Databases

So far, we have considered data in only two categories: either sensitive or non-sensitive. We have alluded to some data items being more sensitive than others, but we have allowed only yes-or-no access. Our presentation may have implied that sensitivity was a function of the *attribute*, the column in which the data appeared, although nothing we have done depended on this interpretation of sensitivity. In fact, though, sensitivity is determined not just by attribute but also in ways that we investigate in the next section.

9.3.1 The Case for Differentiated Security

Consider a database containing data on U.S. government expenditures. Some of the expenditures are for paper clips, which is not sensitive information. Some salary expenditures are subject to privacy requirements. Individual salaries are sensitive, but the aggregate (for example, the total Agriculture Department payroll, which is a matter of public record) is not sensitive. Expenses of certain military operations are more sensitive; for example, the total amount the United States spends for ballistic missiles, which is not public. There are even operations known only to a few people, and so the amount spent on these operations, or even the fact that anything was spent on such an operation, is highly sensitive.

From this description, three characteristics of database security emerge.

- The security of a single element may be different from the security of other elements of the same record or from other values of the same attribute. That is, the security of one element may differ from that of other elements of the same row or column. This situation implies that security should be implemented for each individual element.
- Two levels sensitive and non-sensitive are inadequate to represent some security situations. Several grades of security may be needed. These grades may represent ranges of allowable knowledge, which may overlap. Typically, the security grades form a lattice.
- The security of an aggregate, a sum, a count, or a group of values in a database may differ from the security of the individual elements. The security of the aggregate may be higher or lower than that of the individual elements.

These three principles lead to a model of security not unlike the military model of security encountered earlier, in which the sensitivity of an object is defined as one of *n* levels and is further separated into compartments by category.

9.3.2 Granularity

Recall that the military classification model applied originally to paper documents and was adapted to computers. It is fairly easy to classify and track a single sheet of paper or, for that matter, a paper file, a computer file, or a single program or process. It is entirely different to classify individual data items. For obvious reasons, an entire sheet of paper is classified at one level, even though certain words, such as *and*, *the*, or *of*, would be innocuous in any context, and other words, such as codewords like *Manhattan project*, might be sensitive in any context. But defining the sensitivity of each value in a database is similar to applying a sensitivity level to each individual word of a document.

And the problem is still more complicated. The word *Manhattan* by itself is not sensitive, nor is *project*. However, the combination of these words produces the sensitive codeword *Manhattan project*. A similar situation occurs in databases. Therefore, not only can every *element* of a database have a distinct sensitivity, every *combination of elements* can also have a distinct sensitivity. Furthermore, the combination can be more or less sensitive than any of its elements. So what would we need in order to associate a sensitivity level with each value of a database? First, we need an

access control policy to dictate which users may have access to what data. Typically, to implement this policy each data item is marked to show its access limitations. Second, we need a means to guarantee that the value has not been changed by an unauthorized person. These two requirements address both confidentiality and integrity.

9.3.3 Security Issues

In Chapter 1, we introduced three general security concerns: integrity, confidentiality, and availability. In this section, we extend the first two of these concepts to include their special roles for multilevel databases.

9.3.3.1 Integrity

Even in a single-level database in which all elements have the same degree of sensitivity, integrity is a tricky problem. In the case of multilevel databases, integrity becomes both more important and more difficult to achieve. Because of the *-property for access control, a process that reads high-level data is not allowed to write a file at a lower level. Applied to databases, however, this principle says that a high-level user should not be able to write a lower-level data element. The problem with this interpretation arises when the DBMS must be able to read all records in the database and write new records for any of the following purposes: to do backups, to scan the database to answer queries, to reorganize the database according to a user's processing needs, or to update all records of the database.

When people encounter this problem, they handle it by using trust and common sense. People who have access to sensitive information are careful not to convey it to uncleared individuals. In a computing system, there are two choices: Either the process cleared at a high level cannot write to a lower level or the process must be a "trusted process," the computer equivalent of a person with a security clearance.

9.3.3.2 Confidentiality

Users trust that a database will provide correct information, meaning that the data are consistent and accurate. As indicated earlier, some means of protecting confidentiality may result in small changes to the data. Although these perturbations should not affect statistical analyses, they may produce two different answers representing the same underlying data value in response to two differently formed queries. In the multilevel case, two different users operating at two different levels of security might get two different answers to the same query. To preserve confidentiality, precision is sacrificed.

Enforcing confidentiality also leads to unknowing redundancy. Suppose a personnel specialist works at one level of access permission. The specialist knows that Bob Hill works for the company. However, Bob's record does not appear on the retirement payment roster. The specialist assumes this omission is an error and creates a record for Bob. The reason that no record for Bob appears is that Bob is a secret agent, and his employment with the company is not supposed to be public knowledge. A record on Bob actually is in the file but, because of his special position, his record is not accessible to the personnel specialist. The DBMS cannot reject the record from the personnel specialist because doing so would reveal that there already is such a record at a sensitivity too high for the specialist to see. The creation of the new record means that there are now two records for Bob Hill: one sensitive and one not. This situation is called polyinstantiation, meaning that one record can appear (be instantiated) many times, with a different level of confidentiality each time. In our zeal to reduce polyinstantiation, we must be careful not to eliminate legitimate records such as these.

9.4 Proposals for Multilevel Security

As you can already tell, implementing multilevel security for databases is difficult, probably more

so than in operating systems, because of the small granularity of the items being controlled. In the remainder of this section, we study approaches to multilevel security for databases.

9.4.1 Separation

As we have already seen, separation is necessary to limit access. In this section, we study mechanisms to implement separation in databases. Then, we see how these mechanisms can help to implement multilevel security for databases.

9.4.1.1 Partitioning

The obvious control for multilevel databases is partitioning. The database is divided into separate databases, each at its own level of sensitivity. This approach is similar to maintaining separate files in separate file cabinets. This control destroys a basic advantage of databases: elimination of redundancy and improved accuracy through having only one field to update. Furthermore, it does not address the problem of a high-level user who needs access to some low-level data combined with high-level data. Nevertheless, because of the difficulty of establishing, maintaining, and using multilevel databases, many users with data of mixed sensitivities handle their data by using separate, isolated databases.

9.4.1.2 Encryption

If sensitive data are encrypted, a user who accidentally receives them cannot interpret the data. Thus, each level of sensitive data can be stored in a table encrypted under a key unique to the level of sensitivity. But encryption has certain disadvantages. First, a user can mount a chosen plaintext attack. Suppose party affiliation of REP or DEM is stored in encrypted form in each record. A user who achieves access to these encrypted fields can easily decrypt them by creating a new record with party=DEM and comparing the resulting encrypted version to that element in all other records. Worse, if authentication data are encrypted, the malicious user can substitute the encrypted form of his or her own data for that of any other user. Not only does this provide access for the malicious user, but it also excludes the legitimate user whose authentication data have been changed to that of the malicious user.

Using a different encryption key for each record overcomes these defects. Each record's fields can be encrypted with a different key, or all fields of a record can be cryptographically linked, as with cipher block chaining. The disadvantage, then, is that each field must be decrypted when users perform standard database operations such as "select all records with SALARY > 10,000." Decrypting the SALARY field, even on rejected records, increases the time to process a query. Thus, encryption is not often used to implement separation in databases.

9.4.1.3 Integrity Lock

The integrity lock was first proposed at the U.S. Air Force Summer Study on Data Base Security. The lock is a way to provide both integrity and limited access for a database. The operation was nicknamed "spray paint" because each element is figuratively painted with a colour that denotes its sensitivity. The colouring is maintained with the element, not in a master database table. Each apparent data item consists of three pieces: the actual data item itself, a sensitivity label, and a checksum.

The sensitivity label defines the sensitivity of the data, and the checksum is computed across both data and sensitivity label to prevent unauthorized modification of the data item or its label. The actual data item is stored in plaintext, for efficiency because the DBMS may need to examine many fields when selecting records to match a query. The sensitivity label should be

- *unforgeable,* so that a malicious subject cannot create a new sensitivity level for an element
- *unique,* so that a malicious subject cannot copy a sensitivity level from another element
- *concealed,* so that a malicious subject cannot even determine the sensitivity level of an

arbitrary element

The third piece of the integrity lock for a field is an error-detecting code, called a cryptographic checksum. To guarantee that a data value or its sensitivity classification has not been changed, this checksum must be unique for a given element, and must contain both the element's data value and something to tie that value to a particular position in the database. An appropriate cryptographic checksum includes something unique to the record (the record number), something unique to this data field within the record (the field attribute name), the value of this element, and the sensitivity classification of the element. These four components guard against anyone's changing, copying, or moving the data. The checksum can be computed with a strong encryption algorithm or hash function.

9.4.1.4 Sensitivity Lock

A sensitivity lock is a combination of a unique identifier (such as the record number) and the sensitivity level. Because the identifier is unique, each lock relates to one particular record. Many different elements will have the same sensitivity level. A malicious subject should not be able to identify two elements having identical sensitivity levels or identical data values just by looking at the sensitivity level portion of the lock. Because of the encryption, the lock's contents, especially the sensitivity level, are concealed from plain view. Thus, the lock is associated with one specific record, and it protects the secrecy of the sensitivity level of that record.

9.4.2 Designs of Multilevel Secure Databases

This section covers different designs for multilevel secure databases. These designs show the trade-offs among efficiency, flexibility, simplicity, and trustworthiness.

9.4.2.1 Integrity Lock

The integrity lock DBMS was invented as a short-term solution to the security problem for multilevel databases. The intention was to be able to use any (untrusted) database manager with a trusted procedure that handles access control. The sensitive data were obliterated or concealed with encryption that protected both a data item and its sensitivity. In this way, only the access procedure would need to be trusted because only it would be able to achieve or grant access to sensitive data.

The efficiency of integrity locks is a serious drawback. The space needed for storing an element must be expanded to contain the sensitivity label. Because there are several pieces in the label and one label for every element, the space required is significant. Problematic, too, is the processing time efficiency of an integrity lock. The sensitivity label must be decoded every time a data element is passed to the user to verify that the user's access is allowable. Also, each time a value is written or modified, the label must be recomputed. Thus, substantial processing time is consumed. If the database file can be sufficiently protected, the data values of the individual elements can be left in plaintext. That approach benefits select and project queries across sensitive fields because an element need not be decrypted just to determine whether it should be selected. A final difficulty with this approach is that the untrusted database manager sees all data, so it is subject to Trojan horse attacks by which data can be leaked through covert channels.

9.4.2.2 Trusted Front End

A trusted front end is also known as a guard and operates much like the reference monitor. This approach, originated by Hinke and Schaefer, recognizes that many DBMSs have been built and put into use without consideration of multilevel security. Staff members are already trained in using these DBMSs, and they may in fact use them frequently. The front-end concept takes advantage of existing tools and expertise, enhancing the security of these existing systems with minimal change to the system. The interaction between a user, a trusted front end, and a DBMS involves the

following steps:

a) A user identifies himself or herself to the front end; the front end authenticates the user's identity.
b) The user issues a query to the front end.
c) The front end verifies the user's authorization to data.
d) The front end issues a query to the database manager.
e) The database manager performs I/O access, interacting with low-level access control to achieve access to actual data.
f) The database manager returns the result of the query to the trusted front end.
g) The front end analyzes the sensitivity levels of the data items in the result and selects those items consistent with the user's security level.
h) The front end transmits selected data to the untrusted front end for formatting.
i) The untrusted front end transmits formatted data to the user.

The trusted front end serves as a one-way filter, screening out results the user should not be able to access. But the scheme is inefficient because potentially much data is retrieved and then discarded as inappropriate for the user.

9.4.2.3 Commutative Filters

A commutative filter is a process that forms an interface between the user and a DBMS. However, unlike the trusted front end, the filter tries to capitalize on the efficiency of most DBMSs. The filter reformats the query so that the database manager does as much of the work as possible, screening out many unacceptable records. The filter then provides a second screening to select only data to which the user has access. Filters can be used for security at the record, attribute, or element level.

- When used at the record level, the filter requests desired data plus cryptographic checksum information; it then verifies the accuracy and accessibility of data to be passed to the user.
- At the attribute level, the filter checks whether all attributes in the user's query are accessible to the user and, if so, passes the query to the database manager. On return, it deletes all fields to which the user has no access rights.
- At the element level, the system requests desired data plus cryptographic checksum information. When these are returned, it checks the classification level of every element of every record retrieved against the user's level.

The commutative filter re-forms the original query in a trustable way so that sensitive information is never extracted from the database. The filter works by restricting the query to the DBMS and then restricting the results before they are returned to the user. The advantage of the commutative filter is that it allows query selection, some optimization, and some subquery handling to be done by the DBMS. This delegation of duties keeps the size of the security filter small, reduces redundancy between it and the DBMS, and improves the overall efficiency of the system.

9.4.2.4 Distributed Databases

The distributed or federated database is a fourth design for a secure multilevel database. In this case, a trusted front-end controls access to two unmodified commercial DBMSs: one for all low-sensitivity data and one for all high-sensitivity data. The front end takes a user's query and formulates single-level queries to the databases as appropriate. For a user cleared for high-sensitivity data, the front end submits queries to both the high- and low-sensitivity databases. But if the user is not cleared for high-sensitivity data, the front end submits a query to only the low-sensitivity database. If the result is obtained from either back-end database alone, the front end passes the result back to the user. If the result comes from both databases, the front end has to combine the results appropriately. For example, if the query is a join query having some high-sensitivity terms and some low, the front end has to perform the equivalent of a database join itself.

The distributed database design is not popular because the front end, which must be trusted, is complex, potentially including most of the functionality of a full DBMS itself. In addition, the

design does not scale well to many degrees of sensitivity; each sensitivity level of data must be maintained in its own separate database.

9.4.2.5 Window/View

Traditionally, one of the advantages of using a DBMS for multiple users of different interests (but not necessarily different sensitivity levels) is the ability to create a different view for each user. That is, each user is restricted to a picture of the data reflecting only what the user needs to see. For example, the registrar may see only the class assignments and grades of each student at a university, not needing to see extracurricular activities or medical records. The university health clinic, on the other hand, needs medical records and drug-use information but not scores on standardized academic tests.

The notion of a window or a view can also be an organizing principle for multilevel database access. A window is a subset of a database, containing exactly the information that a user is entitled to access. A view can represent a single user's subset database so that all of a user's queries access only that database. This subset guarantees that the user does not access values outside the permitted ones, because nonpermitted values are not even in the user's database. The view is specified as a set of relations in the database, so the data in the view subset change as data change in the database.

A view may involve computation or complex selection criteria to specify subset data. The data presented to a user is obtained by filtering of the contents of the original database. Attributes, records, and elements are stripped away so that the user sees only acceptable items. Any attribute (column) is withheld unless the user is authorized to access at least one element. Any record (row) is withheld unless the user is authorized to access at least one element. Then, for all elements that still remain, if the user is not authorized to access the element, it is replaced by UNDEFINED. This last step does not compromise any data because the user knows the existence of the attribute (there is at least one element that the user can access) and the user knows the existence of the record (again, at least one accessible element exists in the record). In addition to elements, a view includes relations on attributes. Furthermore, a user can create new relations from new and existing attributes and elements. These new relations are accessible to other users, subject to the standard access rights. A user can operate on the subset database defined in a view only as allowed by the operations authorized in the view. As an example, a user might be allowed to retrieve records specified in one view or to retrieve and update records as specified in another view.

9.4.3 Practical Issues

The multilevel security problem for databases has been studied since the 1970s. Several promising research results have been identified, as we have seen in this chapter. However, as with trusted operating systems, the consumer demand has not been sufficient to support many products. Civilian users have not liked the inflexibility of the military multilevel security model, and there have been too few military users. Consequently, multilevel secure databases are primarily of research and historical interest.

The general concepts of multilevel databases are important. We do need to be able to separate data according to their degree of sensitivity. Similarly, we need ways of combining data of different sensitivities into one database (or at least into one virtual database or federation of databases). And these needs will only increase over time as larger databases contain more sensitive information, especially for privacy concerns.

9.5 Summary

- This chapter has addressed two aspects of security for database management systems: the inference problem for statistical databases, and problems of including users and data of different sensitivity levels in one database.
- The inference problem in a statistical database arises from the mathematical relationships

between data elements and query results. We studied controls for preventing statistical inference, including limited response suppression, perturbation of results, and query analysis.

- One very complex control involves monitoring all data provided to a user in order to prevent inference from independent queries.
- Multilevel secure databases must provide both confidentiality and integrity. Separation can be implemented physically, logically, or cryptographically. We explored five approaches for ensuring confidentiality in multilevel secure databases: integrity lock, trusted front end, commutative filters, distributed databases, and restricted views. Other solutions are likely to evolve as the problem is studied further.

9.6 Review Questions

a) Write a short note on inference.
b) Explain direct attack of inference.
c) What are indirect attacks of inference? What is its types?
d) What is aggregation? Differentiate between aggregation and inference.
e) Write a short note on multilevel databases.
f) How can separation limit access in databases?
g) List and explain the designs for multilevel secure databases.

9.7 Bibliography, References and Further Reading

- *Security in Computing by* C. P. Pfleeger, and S. L. Pfleeger, Pearson Education.
- *Computer Security: Art and Science by* Matt Bishop, Pearson Education.
- *Cryptography And Network Security: Principles and practice by* Stallings
- *Network Security by* Kaufman, Perlman, Speciner
- *Network Security : A Beginner's Guide by* Eric Maiwald, TMH
- *Java Network Security by* Macro Pistoia, Pearson Education
- *Principles of information security by* Whitman, Mattord, Thomson

IDOL

Chapter 10

Security in Networks - I

IDOL

10.1 Objectives

At the end of this chapter, you will be able to understand:

- Networks vs. stand-alone applications and environments: differences and similarities
- Threats against networked applications, including denial of service, web site defacements, malicious mobile code, and protocol attacks

10.2 Introduction

Networks, their design, development, and usage are critical to our style of computing. We interact with networks daily, when we perform banking transactions, make telephone calls, or ride trains and planes. The utility companies use networks to track electricity or water usage and bill for it. When we pay for groceries or gasoline, networks enable our credit or debit card transactions and billing. Life without networks would be considerably less convenient, and many activities would be impossible. Not surprisingly, then, computing networks are attackers' targets of choice. Because of their actual and potential impact, network attacks attract the attention of journalists, managers, auditors, and the general public.

In this chapter we describe what makes a network similar to and different from an application program or an operating system. In investigating networks, you will learn how the concepts of confidentiality, integrity, and availability apply in networked settings. At the same time, you will see that the basic notions of identification and authentication, access control, accountability, and assurance are the basis for network security, just as they have been in other settings.

10.3 Network Concepts

To study network threats and controls, we first must review some of the relevant networking terms and concepts. Networks are both fragile and strong. To see why, think about the power, cable television, telephone, or water network that serves your home. If a falling tree branch breaks the power line to your home, you are without electricity until that line is repaired; you are vulnerable to what is called a single point of failure, because one cut to the network destroys electrical functionality for your entire home. From the user's perspective, a network is sometimes designed so that it looks like two endpoints with a single connection in the middle. For example, the municipal water supply may appear to be little more than a reservoir (the source), the pipes (the transmission or communication medium), and your water faucet (the destination). Although this simplistic view is functionally correct, it ignores the complex design, implementation, and management of the "pipes." In a similar way, we describe computer networks in this chapter in ways that focus on the security concepts but present the networks themselves in a simplistic way, to highlight the role of security and prevent the complexity of the networks from distracting our attention.

10.3.1 The Network

A network in its simplest form, is two devices connected across some medium by hardware and software that enable the communication. In some cases, one device is a computer (sometimes called a "server") and the other is a simpler device (sometimes called a "client") enabled only with some means of input (such as a keyboard) and some means of output (such as a screen).

Although this model defines a basic network, the actual situation is frequently significantly more complicated.

- The simpler client device, employed for user-to-computer communication, is often a PC or workstation, so the client has considerable storage and processing capability.
- A network can be configured as just a single client connected to a single server. But more typically, many clients interact with many servers.

- The network's services are often provided by many computers. As a single user's communication travels back and forth from client to server, it may merely pass through some computers but pause at others for significant interactions.
- The end user is usually unaware of many of the communications and computations taking place in the network on the user's behalf.

A single computing system in a network is often called a node, and its processor (computer) is called a host. A connection between two hosts is known as a link. Network computing consists of users, communications media, visible hosts, and systems not generally visible to end users. Users communicate with networked systems by interacting directly with terminals, workstations, and computers. A workstation is an end-user computing device, usually designed for a single user at a time.

Networks can be described by several typical characteristics:

- *Anonymity:* You may have seen the cartoon image that shows a dog typing at a workstation, and saying to another dog, "On the Internet, nobody knows you're a dog." A network removes most of the clues, such as appearance, voice, or context, by which we recognize acquaintances.
- *Automation:* In some networks, one or both endpoints, as well as all intermediate points, involved in a given communication may be machines with only minimal human supervision.
- *Distance:* Many networks connect endpoints that are physically far apart. Although not all network connections involve distance, the speed of communication is fast enough that humans usually cannot tell whether a remote site is near or far.
- *Opaqueness:* Because the dimension of distance is hidden, users cannot tell whether a remote host is in the room next door or in a different country. In the same way, users cannot distinguish whether they are connected to a node in an office, school, home, or warehouse, or whether the node's computing system is large or small, modest or powerful. In fact, users cannot tell if the current communication involves the same host with which they communicated the last time.
- *Routing diversity:* To maintain or improve reliability and performance, routings between two endpoints are usually dynamic. That is, the same interaction may follow one path through the network the first time and a very different path the second time. In fact, a query may take a different path from the response that follows a few seconds later.
- *Boundary:* The boundary distinguishes an element of the network from an element outside it. For a simple network, we can easily list all the components and draw an imaginary line around it to separate what is in the network from what is outside. But listing all the hosts connected to the Internet is practically impossible. For example, a line surrounding the Internet would have to surround the entire globe today, and Internet connections also pass through satellites in orbit around the earth. Moreover, as people and organizations choose to be connected or not, the number and type of hosts change almost second by second, with the number generally increasing over time.
- *Ownership:* It is often difficult to know who owns each host in a network. The network administrator's organization may own the network infrastructure, including the cable and network devices. However, certain hosts may be connected to a network for convenience, not necessarily implying ownership.
- *Control:* Finally, if ownership is uncertain, control must be, too. To see how, pick an arbitrary host. Is it part of network A? If yes, is it under the control of network A's administrator? Does that administrator establish access control policies for the network, or determine when its software must be upgraded and to what version? Indeed, does the administrator even know what version of software that host runs?

10.3.2 Media

Communication is enabled by several kinds of media. We can choose among several types, such as along copper wires or optical fiber or through the air, as with cellular phones. Let us look at each type in turn.

Cable

Because much of our computer communication has historically been done over telephone lines, the most common network communication medium today is wire. Inside our homes and offices, we use a pair of insulated copper wires, called a twisted pair or unshielded twisted pair (UTP). Copper has good transmission properties at a relatively low cost. The bandwidth of UTP is limited to under 10 megabits per second (Mbps), so engineers cannot transmit a large number of communications simultaneously on a single line. Moreover, the signal strength degrades as it travels through the copper wire, and it cannot travel long distances without a boost.

Another choice for network communication is coaxial (coax) cable, the kind used for cable television. Coax cable is constructed with a single wire surrounded by an insulation jacket. The jacket is itself surrounded by a braided or spiral-wound wire. The inner wire carries the signal, and the outer braid acts as a ground. The most widely used computer communication coax cable is Ethernet, carrying up to 100 Mbps over distances of up to 1500 feet. Coax cable also suffers from degradation of signal quality over distance. Repeaters (for digital signals) or amplifiers (for analog signals) can be spaced periodically along the cable to pick up the signal, amplify it, remove spurious signals called "noise," and retransmit it.

Optical Fiber

A newer form of cable is made of very thin strands of glass. Instead of carrying electrical energy, these fibers carry pulses of light. The bandwidth of optical fiber is up to 1000 Mbps, and the signal degrades less over fiber than over wire or coax; the fiber is good for a run of approximately 2.5 miles. Optical fiber involves less interference, less crossover between adjacent media, lower cost, and less weight than copper. Thus, optical fiber is generally a much better transmission medium than copper.

Wireless

Radio signals can also carry communications. Similar to pagers, wireless microphones, garage door openers, and portable telephones, wireless radio can be used in networks, following a protocol developed for short-range telecommunications, designated the 802.11 family of standards. The wireless medium is used for short distances; it is especially useful for networks in which the nodes are physically close together, such as in an office building or at home. Many 802.11 devices are becoming available for home and office wireless networks.

Microwave

Microwave is a form of radio transmission especially well-suited for outdoor communication. Microwave has a channel capacity similar to coax cable; that is, it carries similar amounts of data. Its principal advantage is that the signal is strong from point of transmission to point of receipt. Therefore, microwave signals do not need to be regenerated with repeaters, as do signals on cable.

Infrared

Infrared communication carries signals for short distances (up to 9 miles) and also requires a clear line of sight. Because it does not require cabling, it is convenient for portable objects, such as laptop computers and connections to peripherals. An infrared signal is difficult to intercept because it is a point-to-point signal. Because of line-of-sight requirements and limited distance, infrared is typically used in a protected space, such as an office, in which in-the-middle attacks would be difficult to conceal.

Satellite

Many communications, such as international telephone calls, must travel around the earth. In the early days of telephone technology, telephone companies ran huge cables along the ocean's bottom, enabling calls to travel from one continent to another. Today, we have other alternatives. The communication companies place satellites in orbits that are synchronized with the rotation of the earth (called geosynchronous orbits).

Table 10-1 Communication Media Strengths and Weakness

Medium	Strengths	Weaknesses
Wire	• Widely used • Inexpensive to buy, install, maintain	• Susceptible to emanation • Susceptible to physical wiretapping
Optical fiber	• Immune to emanation • Difficult to wiretap	• Potentially exposed at connection points
Microwave	• Strong signal, not seriously affected by weather	• Exposed to interception along path of transmission • Requires line of sight location • Signal must be repeated approximately every 30 miles (50 kilometers)
Wireless (radio, WiFi)	• Widely available • Built into many computers	• Signal degrades over distance; suitable for short range • Signal interceptable in circular pattern around transmitter
Satellite	• Strong, fast signal	• Delay due to distance signal travels up and down • Signal exposed over wide area at receiving end

10.3.3 Protocols

When we use a network, the communication media are usually transparent to us. The communication medium may change from one transmission to the next. This ambiguity is actually a positive feature of a network: its *independence*. That is, the communication is separated from the actual medium of communication. Independence is possible because we have defined protocols that allow a user to view the network at a high, abstract level of communication (viewing it in terms of user and data); the details of *how* the communication is accomplished are hidden within software and hardware at both ends. The software and hardware enable us to implement a network according to a protocol stack, a layered architecture for communications. Each layer in the stack is much like a language for communicating information relevant at that layer. Two popular protocol stacks are used frequently for implementing networks: the Open Systems Interconnection (OSI) and the Transmission Control Protocol and Internet Protocol (TCP/IP) architecture.

ISO OSI Reference Model

The International Standards Organization (ISO) Open Systems Interconnection model consists of layers by which a network communication occurs. The OSI reference model contains the seven layers. This seven-layer model starts with an application that prepares data to be transmitted through a network. The data move down through the layers, being transformed and repackaged; at the lower layers, control information is added in headers and trailers. Finally, the data are ready to travel on a physical medium, such as a cable or through the air on a microwave or satellite link. On the receiving end, the data enter the bottom of the model and progress up through the layers where control information is examined and removed, and the data are reformatted. Finally, the data arrive at an application at the top layer of the model for the receiver.

TCP/IP

The OSI model is a conceptual one; it shows the different activities required for sending a communication. However, full implementation of a seven-layer transmission carries too much overhead for megabit-per-second communications; the OSI protocol slows things down to unacceptable levels. For this reason, TCP/IP (Transmission Control Protocol/Internet Protocol) is the protocol stack used for most wide area network communications. TCP/IP was invented for what became the Internet. TCP/IP is defined by protocols, not layers, but we can think of it in terms of four layers: application, host-to-host (end-to-end) transport, Internet, and physical. In particular, an application program deals only with abstract data items meaningful to the application user. Although TCP/IP is often used as a single acronym, it really denotes two different protocols: TCP implements a connected communications session on top of the more basic IP transport protocol. In fact, a third protocol, UDP (user datagram protocol) is also an essential part of the suite. The transport layer receives variable-length messages from the application layer; the transport layer breaks them down into units of manageable size, transferred in packets. The Internet layer transmits application layer packets in datagrams, passing them to different physical connections based on the data's destination. The physical layer consists of device drivers to perform the actual bit-by-bit data communication.

10.3.4 Types of Network

A network is a collection of communicating hosts. But to understand the network and how it works, we have several key questions to ask, such as How many hosts? Communicating by what means? To answer these questions, we are helped by an understanding of several types of subclasses of networks, since they commonly combine into larger networks. The subclasses are general notions, not definitive distinctions. But since the terms are commonly used, we present several common network subclasses that have significant security properties.

Local Area Networks

As the name implies, a local area network (or LAN) covers a small distance, typically within a single building. Usually a LAN connects several small computers, such as personal computers, as well as printers and perhaps some dedicated file storage devices. The primary advantage of a LAN is the opportunity for its users to share data and programs and to share access to devices such as printers.

Wide Area Networks

A wide area network, or WAN, differs from a local area network in terms of both size or distance (as its name implies, it covers a wider geographic area than does a LAN) and control or ownership (it is more likely *not* to be owned or controlled by a single body). Still, there tends to be some unifying principle to a WAN. The hosts on a WAN may all belong to a company with many offices, perhaps even in different cities or countries, or they may be a cluster of independent organizations within a few miles of each other, who share the cost of networking hardware. These examples also show how WANs themselves differ. Some are under close control and maintain a high degree of logical and physical isolation (typically, these are WANs controlled by one organization), while others are only marriages of convenience.

Internetworks (Internets)

Networks of networks, or internetwork networks, are sometimes called internets. An internet is a connection of two or more separate networks, in that they are separately managed and controlled. The most significant internetwork is known as the Internet, because it connects so many of the other public networks. The Internet is, in fact, a federation of networks, loosely controlled by the Internet Society (ISOC) and the Internet Corporation for Assigned Names and Numbers (ICANN). These

organizations enforce certain minimal rules of fair play to ensure that all users are treated equitably, and they support standard protocols so that users can communicate.

10.4 Threats in Networks

This section describes some of the threats you have already hypothesized and perhaps presents you with some new ones. But the general thrust is the same: threats aimed to compromise confidentiality, integrity, or availability, applied against data, software, and hardware by nature, accidents, non-malicious humans, and malicious attackers.

10.4.1 What makes a network vulnerable

An isolated home user or a stand-alone office with a few employees is an unlikely target for many attacks. But add a network to the mix and the risk rises sharply. Consider how a network differs from a stand-alone environment:

Anonymity: An attacker can mount an attack from thousands of miles away and never come into direct contact with the system, its administrators, or users. The potential attacker is thus safe behind an electronic shield. The attack can be passed through many other hosts in an effort to disguise the attack's origin. And computer-to-computer authentication is not the same for computers as it is for humans.

Many points of attack both targets and origins: A simple computing system is a self-contained unit. Access controls on one machine preserve the confidentiality of data on that processor. However, when a file is stored in a network host remote from the user, the data or the file itself may pass through many hosts to get to the user. One host's administrator may enforce rigorous security policies, but that administrator has no control over other hosts in the network. Thus, the user must depend on the access control mechanisms in each of these systems. An attack can come from any host to any host, so that a large network offers many points of vulnerability.

Sharing: Because networks enable resource and workload sharing, more users have the potential to access networked systems than on single computers. Perhaps worse, access is afforded to *more systems*, so that access controls for single systems may be inadequate in networks.

Complexity of system: A network combines two or more possibly dissimilar operating systems. Therefore, a network operating/control system is likely to be more complex than an operating system for a single computing system. Furthermore, the ordinary desktop computer today has greater computing power than did many office computers in the last two decades. The attacker can use this power to advantage by causing the victim's computer to perform part of the attack's computation. And because an average computer is so powerful, most users do not know what their computers are really doing at any moment: What processes are active in the background while you are playing *Invaders from Mars*? This complexity diminishes confidence in the network's security.

Unknown perimeter: A network's expandability also implies uncertainty about the network boundary. One host may be a node on two different networks, so resources on one network are accessible to the users of the other network as well. Although wide accessibility is an advantage, this unknown or uncontrolled group of possibly malicious users is a security disadvantage. A similar problem occurs when new hosts can be added to the network. Every network node must be able to react to the possible presence of new, untrustable hosts.

Unknown path: There may be many paths from one host to another. Suppose that a user on host A1 wants to send a message to a user on host B3. That message might be routed through hosts C or D before arriving at host B3. Host C may provide acceptable security, but not D. Network users seldom have control over the routing of their messages.

10.4.2 Who attacks networks

Who are the attackers? To have some idea of who the attackers might be, we return to concepts introduced in earlier chapter, where we described the three necessary components of an attack: method, opportunity, and motive. Here we consider first the motives of attackers. Focusing on motive may give us some idea of who might attack a networked host or user. Four important motives are challenge or power, fame, money, and ideology.

Challenge

Why do people do dangerous or daunting things, like climb mountains or swim the English Channel or engage in extreme sports? Because of the challenge. The situation is no different for someone skilled in writing or using programs. The single most significant motivation for a network attacker is the intellectual challenge. He or she is intrigued with knowing the answers to Can I defeat this network? What would happen if I tried this approach or that technique? Some attackers enjoy the intellectual stimulation of defeating the supposedly undefeatable. For example, Robert Morris, who perpetrated the Internet worm in 1988, attacked supposedly as an experiment to see if he could exploit a particular vulnerability. Other attackers, such as the Cult of the Dead Cow, seek to demonstrate weaknesses in security defenses so that others will pay attention to strengthening security. Still other attackers are unnamed, unknown individuals working persistently just to see how far they can go in performing unwelcome activities. However, only a few attackers find previously unknown flaws. The vast majority of attackers repeat well-known and even well-documented attacks, sometimes only to see if they work against different hosts.

Fame

The challenge of accomplishment is enough for some attackers. But other attackers seek recognition for their activities. That is, part of the challenge is doing the deed; another part is taking credit for it. In many cases, we do not know who the attackers really are, but they leave behind a "calling card" with a name or moniker: Mafiaboy, Kevin Mitnick, Fluffy Bunny, and members of the Chaos Computer Club, for example. The actors often retain some anonymity by using pseudonyms, but they achieve fame nevertheless. They may not be able to brag too openly, but they enjoy the personal thrill of seeing their attacks written up in the news media.

Money and Espionage

As in other settings, financial reward motivates attackers, too. Some attackers perform industrial espionage, seeking information on a company's products, clients, or long-range plans. We know industrial espionage has a role when we read about laptops and sensitive papers having been lifted from hotel rooms when other more valuable items were left behind. Some countries are notorious for using espionage to aid their state-run industries. Industrial espionage, leading to loss of intellectual property, is clearly a problem.

Organized Crime

With the growth in commercial value of the Internet, participation by organized crime has also increased. In October 2004, police arrested members of a 28-person gang of Internet criminals, called the Shadowcrew, who operated out of six foreign countries and eight states in the United States. Six leaders of that group pled guilty to charges, closing an illicit business that trafficked in at least 1.5 million stolen credit and bank card numbers and resulted in losses in excess of $4 million. These more sophisticated attacks require more than one person working out of a bedroom, and so organization and individual responsibilities follow. With potential revenue in the millions of dollars and operations involving hundreds of thousands of credit card numbers and other pieces of identity, existing organized crime units are sure to take notice.

Ideology

In the last few years, we are starting to find cases in which attacks are perpetrated to advance ideological ends. For example, many security analysts believe that the Code Red worm of 2001 was launched by a group motivated by the tension in U.S. China relations. Denning has distinguished between two types of related behaviours, hactivism and cyberterrorism. Hactivism involves "operations that use hacking techniques against a target's [network] with the intent of disrupting normal operations but not causing serious damage." In some cases, the hacking is seen as giving voice to a constituency that might otherwise not be heard by the company or government organization. Cyberterrorism is more dangerous than hactivism: "politically motivated hacking operations intended to cause grave harm such as loss of life or severe economic damage." Security and terrorism experts are seeing increasing use of the Internet as an attack vector, as a communications medium among attackers, and as a point of attack.

10.4.3 Reconnaissance

Now that we have listed many motives for attacking, we turn to how attackers perpetrate their attacks. Attackers do not ordinarily sit down at a terminal and launch an attack. A clever attacker investigates and plans before acting. A network attacker learns a lot about a potential target before beginning the attack. We study the precursors to an attack so that if we can recognize characteristic behaviour, we may be able to block the attack before it is launched. Because most vulnerable networks are connected to the Internet, the attacker begins preparation by finding out as much as possible about the target.

Port Scan

An easy way to gather network information is to use a port scan, a program that, for a particular IP address, reports which ports respond to messages and which of several known vulnerabilities seem to be present. Port scanning tells an attacker three things: which standard ports or services are running and responding on the target system, what operating system is installed on the target system, and what applications and versions of applications are present. This information is readily available for the asking from a networked system; it can be obtained quietly, anonymously, without identification or authentication, drawing little or no attention to the scan.

Social Engineering

The port scan gives an external picture of a network, where are the doors and windows, of what are they constructed, to what kinds of rooms do they open? The attacker also wants to know what is inside the building. What better way to find out than to ask? Suppose, while sitting at your workstation, you receive a phone call. "Hello, this is John Davis from IT support. We need to test some connections on the internal network. Could you please run the command ipconfig/all on your workstation and read to me the addresses it displays?" The request sounds innocuous. But unless you know John Davis and his job responsibilities well, the caller could be an attacker gathering information on the inside architecture. Social engineering involves using social skills and personal interaction to get someone to reveal security-relevant information and perhaps even to do something that permits an attack. The point of social engineering is to persuade the victim to be helpful. The attacker often impersonates someone inside the organization who is in a bind: "My laptop has just been stolen and I need to change the password I had stored on it," or "I have to get out a very important report quickly and I can't get access to the following thing." This attack works especially well if the attacker impersonates someone in a high position, such as the division vice president or the head of IT security. (Their names can sometimes be found on a public web site, in a network registration with the Internet registry, or in publicity and articles.) The attack is often directed at someone low enough to be intimidated or impressed by the high-level person. A direct phone call and expressions of great urgency can override any natural instinct to check out the story. Because the victim has helped the attacker, the victim will think nothing is wrong and not report the incident.

Thus, the damage may not be known for some time. An attacker has little to lose in trying a social engineering attack. At worst it will raise awareness of a possible target. But if the social engineering is directed against someone who is not skeptical, especially someone not involved in security management, it may well succeed.

Intelligence

From a port scan the attacker knows what is open. From social engineering, the attacker knows certain internal details. But a more detailed floor plan would be nice. Intelligence is the general term for collecting information. In security it often refers to gathering discrete bits of information from various sources and then putting them together like the pieces of a puzzle. One commonly used intelligence technique is called "dumpster diving." It involves looking through items that have been discarded in rubbish bins or recycling boxes. It is amazing what we throw away without thinking about it. Mixed with the remains from lunch might be network diagrams, printouts of security device configurations, system designs and source code, telephone and employee lists, and more. Even outdated printouts may be useful. Seldom will the configuration of a security device change completely. More often only one rule is added or deleted or modified, so an attacker has a high probability of a successful attack based on the old information.

Gathering intelligence may also involve eavesdropping. Trained spies may follow employees to lunch and listen in from nearby tables as co-workers discuss security matters. Or spies may befriend key personnel to co-opt, coerce, or trick them into passing on useful information. Most intelligence techniques require little training and minimal investment of time. If an attacker has targeted a particular organization, spending a little time to collect background information yields a big payoff.

Operating System and Application Fingerprinting

The port scan supplies the attacker with very specific information. For instance, an attacker can use a port scan to find out that port 80 is open and supports HTTP, the protocol for transmitting web pages. But the attacker is likely to have many related questions, such as which commercial server application is running, what version, and what the underlying operating system and version are. Once armed with this additional information, the attacker can consult a list of specific software's known vulnerabilities to determine which particular weaknesses to try to exploit. How can the attacker answer these questions? The network protocols are standard and vendor independent. Still, each vendor's code is implemented independently, so there may be minor variations in interpretation and behaviour. The variations do not make the software noncompliant with the standard, but they are different enough to make each version distinctive. A new version will implement a new feature, but an old version will reject the request. All these peculiarities, sometimes called the operating system or application fingerprint, can mark the manufacturer and version.

Bulletin Boards and Chats

The Internet is probably the greatest tool for sharing knowledge since the invention of the printing press. It is probably also the most dangerous tool for sharing knowledge. Numerous underground bulletin boards and chat rooms support exchange of information. Attackers can post their latest exploits and techniques, read what others have done, and search for additional information on systems, applications, or sites. Remember that, as with everything on the Internet, anyone can post anything, so there is no guarantee that the information is reliable or accurate. And you never know who is reading from the Internet.

Availability of Documentation

The vendors themselves sometimes distribute information that is useful to an attacker. For example, Microsoft produces a resource kit by which application vendors can investigate a Microsoft product to develop compatible, complementary applications. This toolkit also gives attackers tools to use in investigating a product that can subsequently be the target of an attack.

10.4.4 Threats in Transit: Eavesdropping and Wiretapping

By now, you can see that an attacker can gather a significant amount of information about a victim before beginning the actual attack. Once the planning is done, the attacker is ready to proceed. The easiest way to attack is simply to listen in. An attacker can pick off the content of a communication passing in the clear. The term eavesdrop implies overhearing without expending any extra effort. For example, we might say that an attacker (or a system administrator) is eavesdropping by monitoring all traffic passing through a node. The administrator might have a legitimate purpose, such as watching for inappropriate use of resources (for instance, visiting non-work-related web sites from a company network) or communication with inappropriate parties (for instance, passing files to an enemy from a military computer).

A more hostile term is wiretap, which means intercepting communications through some effort. Passive wiretapping is just "listening," much like eavesdropping. But active wiretapping means injecting something into the communication. Wiretapping works differently depending on the communication medium used.

Cable

At the most local level, all signals in an Ethernet or other LAN are available on the cable for anyone to intercept. Each LAN connector (such as a computer board) has a unique address; each board and its drivers are programmed to label all packets from its host with its unique address (as a sender's "return address") and to take from the net only those packets addressed to its host. But removing only those packets addressed to a given host is mostly a matter of politeness; there is little to stop a program from examining each packet as it goes by. A device called a packet sniffer can retrieve all packets on the LAN. Alternatively, one of the interface cards can be reprogrammed to have the supposedly unique address of another existing card on the LAN so that two different cards will both fetch packets for one address. (To avoid detection, the rogue card will have to put back on the net copies of the packets it has intercepted.) Fortunately (for now), LANs are usually used only in environments that are fairly friendly, so these kinds of attacks occur infrequently.

Wireless

Wireless networking is becoming very popular, with good reason. With wireless (also known as WiFi), people are not tied to a wired connection; they are free to roam throughout an office, house, or building while maintaining a connection. Universities, offices, and even home users like being able to connect to a network without the cost, difficulty, and inconvenience of running wires. The difficulties of wireless arise in the ability of intruders to intercept and spoof a connection. But the major threat is not interference; it is interception. A wireless signal is strong for approximately 100 to 200 feet. A strong signal can be picked up easily. And with an inexpensive, tuned antenna, a wireless signal can be picked up several miles away. In other words, someone who wanted to pick up your particular signal could do so from several streets away. Parked in a truck or van, the interceptor could monitor your communications for quite some time without arousing suspicion.

Interception

Interception of wireless traffic is always a threat, through either passive or active wiretapping. You may react to that threat by assuming that encryption will address it. Unfortunately, encryption is not always used for wireless communication, and the encryption built into some wireless devices is not as strong as it should be to deter a dedicated attacker.

Theft of Service

Wireless also admits a second problem: the possibility of rogue use of a network connection. Many hosts run the Dynamic Host Configuration Protocol (DHCP), by which a client negotiates a one-

time IP address and connectivity with a host. This protocol is useful in office or campus settings, where not all users (clients) are active at any time. A small number of IP addresses can be shared among users. Essentially the addresses are available in a pool. A new client requests a connection and an IP address through DHCP, and the server assigns one from the pool. This scheme admits a big problem with authentication. Unless the host authenticates users before assigning a connection, any requesting client is assigned an IP address and network access.

10.4.5 Protocol Flaws

Internet protocols are publicly posted for scrutiny by the entire Internet community. Each accepted protocol is known by its Request for Comment (RFC) number. Many problems with protocols have been identified by sharp reviewers and corrected before the protocol was established as a standard. But protocol definitions are made and reviewed by fallible humans. Likewise, protocols are implemented by fallible humans. For example, TCP connections are established through sequence numbers. The client (initiator) sends a sequence number to open a connection, the server responds with that number and a sequence number of its own, and the client responds with the server's sequence number. Suppose someone can guess a client's next sequence number. That person could impersonate the client in an interchange. Sequence numbers are incremented regularly, so it can be easy to predict the next number.

10.4.6 Impersonation

In many instances, there is an easier way than wiretapping for obtaining information on a network: Impersonate another person or process. Why risk tapping a line, or why bother extracting one communication out of many, if you can obtain the same data directly? Impersonation is a more significant threat in a wide area network than in a local one. Local individuals often have better ways to obtain access as another user; they can, for example, simply sit at an unattended workstation. Still, impersonation attacks should not be ignored even on local area networks, because local area networks are sometimes attached to wider area networks without anyone's first thinking through the security implications. In an impersonation, an attacker has several choices:

- Guess the identity and authentication details of the target.
- Pick up the identity and authentication details of the target from a previous communication or from wiretapping.
- Circumvent or disable the authentication mechanism at the target computer.
- Use a target that will not be authenticated.
- Use a target whose authentication data are known.

Authentication Foiled by Guessing

The results of several studies showing that many users choose easy-to-guess passwords. The Internet worm of 1988 capitalized on exactly that flaw. Morris's worm tried to impersonate each user on a target machine by trying, in order, a handful of variations of the user name, a list of about 250 common passwords and, finally, the words in a dictionary. Sadly, many users' accounts are still open to these easy attacks. A second source of password guesses is default passwords. Many systems are initially configured with default accounts having GUEST or ADMIN as login IDs; accompanying these IDs are well-known passwords such as "guest" or "null" or "password" to enable the administrator to set up the system. Administrators often forget to delete or disable these accounts, or at least to change the passwords. Dead accounts offer a final source of guessable passwords. The attacker can try several passwords until the password guessing limit is exceeded. The system then locks the account administratively, and the attacker uses a social engineering attack. In all these ways the attacker may succeed in resetting or discovering a password.

Authentication Thwarted by Eavesdropping or Wiretapping
Because of the rise in distributed and client-server computing, some users have access privileges on several connected machines. To protect against arbitrary outsiders using these accesses, authentication is required between hosts. This access can involve the user directly, or it can be done automatically on behalf of the user through a host-to-host authentication protocol. In either case, the account and authentication details of the subject are passed to the destination host. When these details are passed on the network, they are exposed to anyone observing the communication on the network. These same authentication details can be reused by an impersonator until they are changed.

Authentication Foiled by Avoidance
Obviously, authentication is effective only when it works. A weak or flawed authentication allows access to any system or person who can circumvent the authentication. Many network hosts, especially those that connect to wide area networks, run variants of Unix System V or BSD Unix. In a local environment, many users are not aware of which networked operating system is in use; still fewer would know of, be capable of, or be interested in exploiting flaws. However, some hackers regularly scan wide area networks for hosts running weak or flawed operating systems. Thus, connection to a wide area network, especially the Internet, exposes these flaws to a wide audience intent on exploiting them.

Non-existent Authentication
If two computers are used by the same users to store data and run processes and if each has authenticated its users on first access, you might assume that computer-to-computer or local user-to-remote process authentication is unnecessary. These two computers and their users are a trustworthy environment in which the added complexity of repeated authentication seems excessive. These "trusted hosts" can also be exploited by outsiders who obtain access to one system through an authentication weakness (such as a guessed password) and then transfer to another system that accepts the authenticity of a user who comes from a system on its trusted list. An attacker may also realize that a system has some identities requiring no authentication. Some systems have "guest" or "anonymous" accounts to allow outsiders to access things the systems want to release to anyone.

Well-Known Authentication
Authentication data should be unique and difficult to guess. But unfortunately, the convenience of one well known authentication scheme sometimes usurps the protection. The system network management protocol (SNMP) is widely used for remote management of network devices, such as routers and switches, that support no ordinary users. SNMP uses a "community string," essentially a password for the community of devices that can interact with one another. But network devices are designed especially for quick installation with minimal configuration, and many network administrators do not change the default community string installed on a router or switch. This laxity makes these devices on the network perimeter open to many SNMP attacks. Some vendors still ship computers with one system administration account installed, having a default password. Or the systems come with a demonstration or test account, with no required password. Some administrators fail to change the passwords or delete these accounts.

Trusted Authentication
Finally, authentication can become a problem when identification is delegated to other trusted sources. For instance, a file may indicate who can be trusted on a particular host. Or the authentication mechanism for one system can "vouch for" a user. We noted earlier how the Unix *.rhosts*, *.rlogin*, and */etc/hosts/equiv* files indicate hosts or users that are trusted on other hosts. While these features are useful to users who have accounts on multiple machines or for network management, maintenance, and operation, they must be used very carefully. Each of them

represents a potential hole through which a remote user or a remote attacker can achieve access.

Spoofing
Guessing or otherwise obtaining the network authentication credentials of an entity (a user, an account, a process, a node, a device) permits an attacker to create a full communication under the entity's identity. Impersonation falsely represents a valid entity in a communication. Closely related is spoofing, when an attacker falsely carries on one end of a networked interchange. Examples of spoofing are masquerading, session hijacking, and man-in-the-middle attacks.

Masquerade
In a masquerade one host pretends to be another. A common example is URL confusion. Domain names can easily be confused, or someone can easily mistype certain names. Thus xyz.com, xyz.org, and xyz.net might be three different organizations, or one bona fide organization (for example, xyz.com) and two masquerade attempts from someone who registered the similar domain names. From the attacker's point of view, the fun in masquerading comes before the mask is removed.
A variation of this attack is called phishing. You send an e-mail message, perhaps with the real logo of Blue Bank, and an enticement to click on a link, supposedly to take the victim to the Blue Bank web site. The enticement might be that your victim's account has been suspended or that you offer your victim some money for answering a survey (and need the account number and PIN to be able to credit the money), or some other legitimate-sounding explanation.
In another version of a masquerade, the attacker exploits a flaw in the victim's web server and is able to overwrite the victim's web pages. Although there is some public humiliation at having one's site replaced, perhaps with obscenities or strong messages opposing the nature of the site (for example, a plea for vegetarianism on a slaughterhouse web site), most people would not be fooled by a site displaying a message absolutely contrary to its aims. However, a clever attacker can be more subtle. Instead of differentiating from the real site, the attacker can try to build a false site that resembles the real one, perhaps to obtain sensitive information (names, authentication numbers, credit card numbers) or to induce the user to enter into a real transaction.

Session Hijacking
Session hijacking is intercepting and carrying on a session begun by another entity. Suppose two entities have entered into a session but then a third entity intercepts the traffic and carries on the session in the name of the other. A different type of example involves an interactive session, for example, using Telnet. If a system administrator logs in remotely to a privileged account, a session hijack utility could intrude in the communication and pass commands as if they came from the administrator.

Man-in-the-Middle Attack
Our hijacking example requires a third party involved in a session between two entities. A man-in-the-middle attack is a similar form of attack, in which one entity intrudes between two others. The difference between man-in-the-middle and hijacking is that a man-in-the-middle usually participates from the start of the session, whereas a session hijacking occurs after a session has been established. The difference is largely semantic and not too significant.

10.4.7 Message Confidentiality Threats
An attacker can easily violate message confidentiality (and perhaps integrity) because of the public nature of networks. Eavesdropping and impersonation attacks can lead to a confidentiality or integrity failure. Here we consider several other vulnerabilities that can affect confidentiality.

Misdelivery
Sometimes messages are misdelivered because of some flaw in the network hardware or software. Most frequently, messages are lost entirely, which is an integrity or availability issue. Occasionally, however, a destination address is modified or some handler malfunctions, causing a message to be delivered to someone other than the intended recipient. All of these "random" events are quite uncommon. More frequent than network flaws are human errors. It is far too easy to mistype an address. There is simply no justification for a computer network administrator to identify people by meaningless long numbers or cryptic initials.

Exposure
To protect the confidentiality of a message, we must track it all the way from its creation to its disposal. Along the way, the content of a message may be exposed in temporary buffers; at switches, routers, gateways, and intermediate hosts throughout the network; and in the workspaces of processes that build, format, and present the message. In earlier chapters, we considered confidentiality exposures in programs and operating systems. All of these exposures apply to networked environments as well. Furthermore, a malicious attacker can use any of these exposures as part of a general or focused attack on message confidentiality. Passive wiretapping is one source of message exposure. So, also is subversion of the structure by which a communication is routed to its destination. Finally, intercepting the message at its source, destination, or at any intermediate node can lead to its exposure.

Traffic Flow Analysis
Sometimes not only is the message itself sensitive but the fact that a message *exists* is also sensitive. For example, if the enemy during wartime sees a large amount of network traffic between headquarters and a particular unit, the enemy may be able to infer that significant action is being planned involving that unit. In a commercial setting, messages sent from the president of one company to the president of a competitor could lead to speculation about a takeover or conspiracy to fix prices. Or communications from the prime minister of one country to another with whom diplomatic relations were suspended could lead to inferences about a rapprochement between the countries. In these cases, we need to protect both the *content* of messages and the *header* information that identifies sender and receiver.

10.4.8 Message Integrity Threats
In many cases, the *integrity* or correctness of a communication is at least as important as its confidentiality. In fact, for some situations, such as passing authentication data, the integrity of the communication is paramount. In other cases, the need for integrity is less obvious. Next we consider threats based on failures of integrity in communication.

Falsification of Messages
Increasingly, people depend on electronic messages to justify and direct actions. For example, if you receive a message from a good friend asking you to meet at the pub for a drink next Tuesday evening, you will probably be there at the appointed time. As long as it is reasonable, we tend to act on an electronic message just as we would on a signed letter, a telephone call, or a face-to-face communication. However, an attacker can take advantage of our trust in messages to mislead us. In particular, an attacker may

- change some or all of the content of a message
- replace a message entirely, including the date, time, and sender/receiver identification
- reuse (replay) an old message
- combine pieces of different messages into one
- change the apparent source of a message

- redirect a message
- destroy or delete a message

These attacks can be perpetrated in the ways we have already examined, including

- active wiretap
- Trojan horse
- Impersonation
- preempted host
- preempted workstation

Noise

Signals sent over communications media are subject to interference from other traffic on the same media, as well as from natural sources, such as lightning, electric motors, and animals. Such unintentional interference is called noise. These forms of noise are inevitable, and they can threaten the integrity of data in a message. Fortunately, communications protocols have been intentionally designed to overcome the negative effects of noise. For example, the TCP/IP protocol suite ensures detection of almost all transmission errors. Processes in the communications stack detect errors and arrange for retransmission, all invisible to the higher-level applications. Thus, noise is scarcely a consideration for users in security-critical applications.

10.4.9 Format Failures

Network communications work because of well-designed protocols that define how two computers communicate with a minimum of human intervention. The format of a message, size of a data unit, sequence of interactions, even the meaning of a single bit is precisely described in a standard. The whole network works only because everyone obeys these rules. Almost everyone, that is. Attackers purposely break the rules to see what will happen. Or the attacker may seek to exploit an undefined condition in the standard. Software may detect the violation of structure and raise an error indicator. Sometimes, however, the malformation causes a software failure, which can lead to a security compromise, just what the attacker wants. In this section we look at several kinds of malformation.

Malformed Packets

Packets and other data items have specific formats, depending on their use. Field sizes, bits to signal continuations, and other flags have defined meanings and will be processed appropriately by network service applications called protocol handlers. These services do not necessarily check for errors, however. For example, in 2003 Microsoft distributed a patch for its RPC (Remote Procedure Call) service. If a malicious user initiated an RPC session and then sent an incorrectly formatted packet, the entire RPC service failed, as well as some other Microsoft services. Attackers try all sorts of malformations of packets. Of course, many times the protocol handler detects the malformation and raises an error condition, and other times the failure affects only the user (the attacker). But when the error causes the protocol handler to fail, the result can be denial of service, complete failure of the system, or some other serious result.

Protocol Failures and Implementation Flaws

Each protocol is a specification of a service to be provided; the service is then implemented in software, which may be flawed. Network protocol software is basic to the operating system, so flaws in that software can cause widespread harm because of the privileges with which the software runs and the impact of the software on many users at once. Certain network protocol implementations have been the source of many security flaws; especially troublesome have been SNMP (network management), DNS (addressing service), and e-mail services such as SMTP and S/MIME. Although different vendors have implemented the code for these services themselves, they often are based on a common (flawed) prototype. Or the protocol itself may be incomplete. If

the protocol does not specify what action to take in a particular situation, vendors may produce different results. So, an interaction on Windows, for example, might succeed while the same interaction on a Unix system would fail. The protocol may have an unknown security flaw. Attackers can exploit all of these kinds of errors.

10.4.10 Website Vulnerabilities

A web site is especially vulnerable because it is almost completely exposed to the user. If you use an application program, you do not usually get to view the program's code. With a web site, the attacker can download the site's code for offline study over time. With a program, you have little ability to control in what order you access parts of the program, but a web attacker gets to control in what order pages are accessed. The attacker can also choose what data to supply and can run experiments with different data values to see how the site will react. In short, the attacker has some advantages that can be challenging to control.

Web Site Defacement

One of the most widely known attacks is the web site defacement attack. Because of the large number of sites that have been defaced and the visibility of the result, the attacks are often reported in the popular press. A defacement is common not only because of its visibility but also because of the ease with which one can be done. Web sites are designed so that their code is downloaded, enabling an attacker to obtain the full hypertext document and all programs directed to the client in the loading process. An attacker can even view programmers' comments left in as they built or maintained the code. The download process essentially gives the attacker the blueprints to the web site.

Buffer Overflows

Buffer overflow is alive and well on web pages, too. The attacker simply feeds a program far more data than it expects to receive. A buffer size is exceeded, and the excess data spill over into adjoining code and data locations. Perhaps the best-known web server buffer overflow is the file name problem known as iishack. This attack is so well known that is has been written into a procedure (see *http://www.technotronic.com*). To execute the procedure, an attacker supplies as parameters the site to be attacked and the URL of a program the attacker wants that server to execute.

Dot-Dot-Slash

Web server code should always run in a constrained environment. Ideally, the web server should never have editors, xterm and Telnet programs, or even most system utilities loaded. By constraining the environment in this way, even if an attacker escapes from the web server application, no other executable programs will help the attacker use the web server's computer and operating system to extend the attack. The code and data for web applications can be transferred manually to a web server or pushed as a raw image. But many web applications programmers are naïve. They expect to need to edit a web application in place, so they install editors and system utilities on the server to give them a complete environment in which to program.

A second, less desirable, condition for preventing an attack is to create a fence confining the web server application. With such a fence, the server application cannot escape from its area and access other potentially dangerous system areas (such as editors and utilities). The server begins in a particular directory subtree, and everything the server needs is in that same subtree.

Enter the dot-dot. In both Unix and Windows, '..' is the directory indicator for "predecessor." And '../..' is the grandparent of the current location. So someone who can enter file names can travel back up the directory tree one .. at a time. Cerberus Information Security analysts found just that vulnerability in the webhits.dll extension for the Microsoft Index Server.

Application Code Errors
A user's browser carries on an intricate, undocumented protocol interchange with applications on the web server. To make its job easier, the web server passes context strings to the user, making the user's browser reply with full context. A problem arises when the user can modify that context. An example is the time-of-check to time-of-use flaw that we discussed in earlier chapters. The server sets (checks) the price of the item when you first display the price, but then it loses control of the checked data item and never checks it again. This situation arises frequently in server application code because application programmers are generally not aware of security and typically do not anticipate malicious behaviour.

Server-Side Include
A potentially more serious problem is called a server-side include. The problem takes advantage of the fact that web pages can be organized to invoke a particular function automatically. For example, many pages use web commands to send an e-mail message in the "contact us" part of the displayed page. The commands, such as e-mail, if, goto, and include, are placed in a field that is interpreted in HTML. One of the server-side include commands is exec, to execute an arbitrary file on the server. For instance, the server-side include command opens a Telnet session from the server running in the name of the server. An attacker may find it interesting to execute commands such as *chmod* (change access rights to an object), *sh* (establish a command shell), or *cat* (copy to a file).

10.4.11 Denial of Service
Availability attacks, sometimes called denial-of-service or DOS attacks, are much more significant in networks than in other contexts. There are many accidental and malicious threats to availability or continued service.

Transmission Failure
Communications fail for many reasons. For instance, a line is cut. Or network noise makes a packet unrecognizable or undeliverable. A machine along the transmission path fails for hardware or software reasons. A device is removed from service for repair or testing. A device is saturated and rejects incoming data until it can clear its overload. Many of these problems are temporary or automatically fixed (circumvented) in major networks, including the Internet. However, some failures cannot be easily repaired. From a malicious standpoint, you can see that anyone who can sever, interrupt, or overload capacity to you can deny you service. The physical threats are pretty obvious. We consider instead several electronic attacks that can cause a denial of service.

Connection Flooding
The most primitive denial-of-service attack is flooding a connection. If an attacker sends you as much data as your communications system can handle, you are prevented from receiving any other data. Even if an occasional packet reaches you from someone else, communication to you will be seriously degraded. More sophisticated attacks use elements of Internet protocols. In addition to TCP and UDP, there is a third class of protocols, called ICMP or Internet Control Message Protocols. Normally used for system diagnostics, these protocols do not have associated user applications.

Echo-Chargen
This attack works between two hosts. Chargen is a protocol that generates a stream of packets; it is used to test the network's capacity. The attacker sets up a chargen process on host A that generates its packets as echo packets with a destination of host B. Then, host A produces a stream of packets to which host B replies by echoing them back to host A. This series puts the network infrastructures

of A and B into an endless loop. If the attacker makes B both the source and destination address of the first packet, B hangs in a loop, constantly creating and replying to its own messages.

Ping of Death
A ping of death is a simple attack. Since ping requires the recipient to respond to the ping request, all the attacker needs to do is send a flood of pings to the intended victim. The attack is limited by the smallest bandwidth on the attack route. The attacker cannot mathematically flood the victim alone. But the attack succeeds if the numbers are reversed. The ping packets will saturate the victim's bandwidth.

Smurf
The smurf attack is a variation of a ping attack. It uses the same vehicle, a ping packet, with two extra twists. First, the attacker chooses a network of unwitting victims. The attacker spoofs the source address in the ping packet so that it appears to come from the victim. Then, the attacker sends this request to the network in broadcast mode by setting the last byte of the address to all 1s; broadcast mode packets are distributed to all hosts on the network.

Syn Flood
Another popular denial-of-service attack is the syn flood. This attack uses the TCP protocol suite, making the session-oriented nature of these protocols work against the victim. The attacker can deny service to the target by sending many SYN requests and never responding with ACKs, thereby filling the victim's SYN_RECV queue.

Teardrop
The teardrop attack misuses a feature designed to improve network communication. In the teardrop attack, the attacker sends a series of datagrams that cannot fit together properly. One datagram might say it is position 0 for length 60 bytes, another position 30 for 90 bytes, and another position 41 for 173 bytes. These three pieces overlap, so they cannot be reassembled properly. In an extreme case, the operating system locks up with these partial data units it cannot reassemble, thus leading to denial of service.

Traffic Redirection
A router is a device that forwards traffic on its way through intermediate networks between a source host's network and a destination's network. So if an attacker can corrupt the routing, traffic can disappear. To see how, keep in mind that, in spite of its sophistication, a router is simply a computer with two or more network interfaces. Suppose a router advertises to its neighbours that it has the best path to every other address in the whole network. Soon all routers will direct all traffic to that one router. The one router may become flooded, or it may simply drop much of its traffic. In either case, a lot of traffic never makes it to the intended destination.

DNS Attacks
Our final denial-of-service attack is actually a class of attacks based on the concept of domain name server. By overtaking a name server or causing it to cache spurious entries (called DNS cache poisoning), an attacker can redirect the routing of any traffic, with an obvious implication for denial of service.

10.4.12 Distributed Denial of Service
The denial-of-service attacks we have listed are powerful by themselves, but an attacker can construct a two-stage attack that multiplies the effect many times. This multiplicative effect gives power to distributed denial of service. To perpetrate a distributed denial-of-service (or DDoS)

attack, an attacker does two things. In the first stage, the attacker uses any convenient attack to plant a Trojan horse on a target machine. That Trojan horse does not necessarily cause any harm to the target machine, so it may not be noticed. The attacker repeats this process with many targets. Each of these target systems then becomes what is known as a zombie. The target systems carry out their normal work, unaware of the resident zombie. At some point the attacker chooses a victim and sends a signal to all the zombies to launch the attack. Then, instead of the victim's trying to defend against one denial-of-service attack from one malicious host, the victim must try to counter *n* attacks from the *n* zombies all acting at once.

10.4.13 Threats in Active or Mobile Code

Active code or mobile code is a general name for code that is pushed to the client for execution. There are many different kinds of active code.

Cookies

Strictly speaking, cookies are not active code. They are data files that can be stored and fetched by a remote server. However, cookies can be used to cause unexpected data transfer from a client to a server, so they have a role in a loss of confidentiality. So a cookie is something that takes up space on your disk, holding information about you that you cannot see, forwarded to servers you do not know whenever the server wants it, without informing you. The philosophy behind cookies seems to be "Trust us, it's good for you."

Scripts

Clients can invoke services by executing scripts on servers. Typically, a web browser displays a page. As the user interacts with the web site via the browser, the browser organizes user inputs into parameters to a defined script; it then sends the script and parameters to a server to be executed. But all communication is done through HTML. The server cannot distinguish between commands generated from a user at a browser completing a web page and a user's handcrafting a set of orders. The malicious user can monitor the communication between a browser and a server to see how changing a web page entry affects what the browser sends and then how the server reacts. With this knowledge, the malicious user can manipulate the server's actions.

Active Code

Displaying web pages started simply with a few steps: generate text, insert images, and register mouse clicks to fetch new pages. Soon, people wanted more elaborate action at their web sites: toddlers dancing atop the page, a three-dimensional rotating cube, images flashing on and off, colours changing, totals appearing. Some of these tricks, especially those involving movement, take significant computing power; they require a lot of time and communication to download from a server. But typically, the client has a capable and underutilized processor, so the timing issues are irrelevant. To take advantage of the processor's power, the server may download code to be executed on the client. This executable code is called active code. The two main kinds of active code are Java code and ActiveX controls.

A hostile applet is downloadable Java code that can cause harm on the client's system. Because an applet is not screened for safety when it is downloaded and because it typically runs with the privileges of its invoking user, a hostile applet can cause serious damage. Using ActiveX controls, objects of arbitrary type can be downloaded to a client. If the client has a viewer or handler for the object's type, that viewer is invoked to present the object. To prevent arbitrary downloads, Microsoft uses an authentication scheme under which downloaded code is cryptographically signed and the signature is verified before execution. But the authentication verifies only the source of the code, not its correctness or safety.

Auto Exec by Type
Data files are processed by programs. For some products, the file type is implied by the file extension, such as .doc for a Word document, .pdf (Portable Document Format) for an Adobe Acrobat file, or .exe for an executable file. On many systems, when a file arrives with one of these extensions, the operating system automatically invokes the appropriate processor to handle it. A malicious agent might send you a file named innocuous.doc, which you would expect to be a Word document. Because of the .doc extension, Word would try to open it. An attacker can disguise a malicious active file under a nonobvious file type.

Bots
Bots, hackerese for robots, are pieces of malicious code under remote control. These code objects are Trojan horses that are distributed to large numbers of victims' machines. Because they may not interfere with or harm a user's computer (other than consuming computing and network resources), they are often undetected. Structured as a loosely coordinated web, a network of bots, called a botnet, is not subject to failure of any one bot or group of bots, and with multiple channels for communication and coordination, they are highly resilient. Botnets are used for distributed denial-of-service attacks, launching attacks from many sites in parallel against a victim. They are also used for spam and other bulk email attacks, in which an extremely large volume of e-mail from any one point might be blocked by the sending service provider.

10.4.14 Complex Attacks
As if these vulnerabilities were not enough, two other phenomena multiply the risk. Scripts let people perform attacks even if the attackers do not understand what the attack is or how it is performed. Building blocks let people combine components of an attack, almost like building a house from prefabricated parts.

Script Kiddies
Attacks can be scripted. A simple smurf denial-of-service attack is not hard to implement. But an underground establishment has written scripts for many of the popular attacks. With a script, attackers need not understand the nature of the attack or even the concept of a network. The hacker community is active in creating scripts for known vulnerabilities. People who download and run attack scripts are called script kiddies. As the rather derogatory name implies, script kiddies are not well respected in the attacker community because the damage they do requires almost no creativity or innovation. Nevertheless, script kiddies can cause serious damage, sometimes without even knowing what they do.

Building Blocks
This chapter's attack types do not form an exhaustive list, but they represent the kinds of vulnerabilities being exploited, their sources, and their severity. A good attacker knows these vulnerabilities and many more. An attacker simply out to cause minor damage to a randomly selected site could use any of the techniques we have described, perhaps under script control. A dedicated attacker who targets one location can put together several pieces of an attack to compound the damage. Often, the attacks are done in series so that each part builds on the information gleaned from previous attacks. For example, a wiretapping attack may yield reconnaissance information with which to form an ActiveX attack that transfers a Trojan horse that monitors for sensitive data in transmission. Putting the attack pieces together like building blocks expands the number of targets and increases the degree of damage.

10.5 Summary

This chapter covers a very large and important area of computer security: networks and distributed applications. As the world becomes more connected by networks, the significance of network security will certainly continue to grow. Security issues for networks are visible and important, but their analysis is similar to the analysis done for other aspects of security. That is, we ask questions about what we are protecting and why we are protecting it.

A network has many different vulnerabilities, but all derive from an underlying model of computer, communications, and information systems security. Threats are raised against the key aspects of security: confidentiality, integrity, and availability.

Network assets include the network infrastructure, applications programs and, most importantly, data. Recall that threats are actions or situations that offer potential harm to or loss of confidentiality, integrity, or availability, in the form of interception (eavesdropping or passive wiretapping), modification (active wiretapping, falsification, and compromise of authenticity), and denial of service. In stand-alone computing, most agents have a strong motive for an attack. But in networks we see new threat agents; anyone can be a victim of essentially a random attack. The strongest network controls are solid authentication, access control, and encryption.

Networks usually employ many copies of the same or similar software, with a copy on each of several (or all) machines in the network. This similarity, combined with connectivity, means that any fault in one copy of a program can create vulnerabilities spread across many machines. Mass-market software often has flaws, and each flaw can be studied and exploited by an attacker. In large networks, a huge number of potential attackers can probe the software extensively; the result is that a network often includes many identified faults and software patches to counter them.

10.6 Review Questions

a) Explain the characteristics of the network.
b) Explain the different types of media.
c) List the strengths and weaknesses of different types of communication medium.
d) What are the different types of networks?
e) What are the differences between network and stand-alone environment?
f) What are the motives to attack a network?
g) How do attackers perpetrate their attacks?
h) Write a short note on eavesdropping and wiretapping.
i) What is impersonation? What choices does an attacker have during impersonation?
j) Explain the threats to message confidentiality.
k) Explain the threats to message integrity.
l) Write a short note on website vulnerabilities.
m) Explain the various availability attacks (DOS attacks).
n) Explain the different threats in active code.

10.7 Bibliography, References and Further Reading

- *Security in Computing by* C. P. Pfleeger, and S. L. Pfleeger, Pearson Education.
- *Computer Security: Art and Science by* Matt Bishop, Pearson Education.
- *Cryptography And Network Security: Principles and practice by* Stallings
- *Network Security by* Kaufman, Perlman, Speciner

- *Network Security : A Beginner's Guide by* Eric Maiwald, TMH
- *Java Network Security by* Macro Pistoia, Pearson Education
- *Principles of information security by* Whitman, Mattord, Thomson

IDOL

Chapter 11

Security in Networks - II

IDOL

11.0 Objectives

At the end of this chapter, you will be able to understand:

- Controls against network attacks: physical security, policies and procedures, and a range of technical controls
- Firewalls: design, capabilities, limitations
- Intrusion detection systems
- Private e-mail: PGP and S/MIME

11.1 Introduction

In this chapter we consider three categories of controls: First, as you can well imagine, the familiar control of encryption is a strong tool for preserving both confidentiality and integrity in networks. We describe architecturally how encryption can be used and then introduce two specific applications of cryptography to networking: encrypted communication between a browser and its websites, called SSL encryption, and encrypted links within a network, called a virtual private network or VPN. Then we introduce a network-protection tool called a firewall, which is really just an instantiation of the familiar reference monitor. We end the study of controls with another device, called an intrusion detection or protection system, that monitors network traffic to identify and counter specific malicious network threats.

11.2 Network Security Controls

The list of security attacks is long, and the news media carry frequent accounts of serious security incidents. Previous chapters have presented several strategies for addressing security concerns, such as encryption for confidentiality and integrity, reference monitors for access control, and overlapping controls for defense in depth. These strategies are also useful in protecting networks. This section presents many excellent defenses available to the network security engineer. Subsequent sections provide detailed explanations for three particularly important controls, firewalls, intrusion detection systems, and encrypted e-mail.

11.2.1 Security Threat Analysis

Recall the three steps of a security threat analysis in other situations. First, we scrutinize all the parts of a system so that we know what each part does and how it interacts with other parts. Next, we consider possible damage to confidentiality, integrity, and availability. Finally, we hypothesize the kinds of attacks that could cause this damage. We can take the same steps with a network.

Why are all these attacks possible? Size, anonymity, ignorance, misunderstanding, complexity, dedication, and programming all contribute. But we have help at hand; we look next at specific threats and their countermeasures.

11.2.2 Design and Implementation

Concepts from the work of the early trusted operating systems projects have natural implications for networks as well. And assurance relates to networked systems. In general, the Open Web Applications project has documented many of the techniques people can use to develop secure web applications. Thus, having addressed secure programming from several perspectives.

11.2.3 Architecture

When we build or modify computer-based systems, we can give some thought to their overall architecture and plan to "build in" security as one of the key constructs. Similarly, the architecture or design of a network can have a significant effect on its security.

Segmentation

Just as segmentation was a powerful security control in operating systems, it can limit the potential for harm in a network in two important ways: Segmentation reduces the number of threats, and it limits the amount of damage a single vulnerability can allow. Separate access is another way to segment the network.

Redundancy

Another key architectural control is redundancy: allowing a function to be performed on more than one node, to avoid "putting all the eggs in one basket."

Single Points of Failure

Ideally, the architecture should make the network immune to failure. In fact, the architecture should at least make sure that the system tolerates failure in an acceptable way (such as slowing down but not stopping processing or recovering and restarting incomplete transactions). One way to evaluate the network architecture's tolerance of failure is to look for single points of failure.

Mobile Agents

Mobile code and hostile agents are potential methods of attack. However, they can also be forces for good. Good agents might look for unsecured wireless access, software vulnerabilities, or embedded malicious code.

11.2.4 Encryption

Encryption is probably the most important and versatile tool for a network security expert. We have seen in earlier chapters that encryption is powerful for providing privacy, authenticity, integrity, and limited access to data. Because networks often involve even greater risks, they often secure data

with encryption, perhaps in combination with other controls. Before we begin to study the use of encryption to counter network security threats, let us consider these points. First, remember that encryption is not a panacea or silver bullet. A flawed system design with encryption is still a flawed system design. Second, notice that encryption protects only what is encrypted. Finally, encryption is no more secure than its key management. If an attacker can guess or deduce a weak encryption key, the game is over.

Link Encryption

In link encryption, data are encrypted just before the system places them on the physical communications link. In this case, encryption occurs at layer 1 or 2 in the OSI model. (A similar situation occurs with TCP/IP protocols.) Similarly, decryption occurs just as the communication arrives at and enters the receiving computer. Link encryption is invisible to the user. The encryption becomes a transmission service performed by a low-level network protocol layer, just like message routing or transmission error detection.

End-to-End Encryption

As its name implies, end-to-end encryption provides security from one end of a transmission to the other. The encryption can be applied by a hardware device between the user and the host. Alternatively, the encryption can be done by software running on the host computer. In either case, the encryption is performed at the highest levels (layer 7, application, or perhaps at layer 6, presentation) of the OSI model. Since the encryption precedes all the routing and transmission processing of the layer, the message is transmitted in encrypted form throughout the network.

Table 11-1 Comparison of Link and End-to-End Encryption

Link Encryption	End-to-End Encryption
Security within hosts	
Data partially exposed in sending host	Data protected in sending host
Data partially exposed in intermediate nodes	Data protected through intermediate nodes
Role of user	
Applied by sending host	Applied by user application
Invisible to user	User application encrypts
Host administrators select encryption	User selects algorithm
One facility for all users	Each user selects
Can be done in software or hardware	Usually software implementation; occasionally performed by user add-on hardware
All or no data encrypted	User can selectively encrypt individual data items
Implementation considerations	
Requires one key per pair of hosts	Requires one key per pair of users
Provides node authentication	Provides user authentication

Virtual Private Networks

Link encryption can be used to give a network's users the sense that they are on a private network, even when it is part of a public network. For this reason, the approach is called a virtual private

network (or VPN). Typically, physical security and administrative security are strong enough to protect transmission inside the perimeter of a network. Thus, the greatest exposure for a user is between the user's workstation or client and the perimeter of the host network or server. A firewall is an access control device that sits between two networks or two network segments. It filters all traffic between the protected or "inside" network and a less trustworthy or "outside" network or segment. With the VPN, we say that the communication passes through an encrypted tunnel or tunnel.

PKI and Certificates

A public key infrastructure, or PKI, is a process created to enable users to implement public key cryptography, usually in a large (and frequently, distributed) setting. PKI offers each user a set of services, related to identification and access control, as follows:

- Create certificates associating a user's identity with a (public) cryptographic key
- Give out certificates from its database
- Sign certificates, adding its credibility to the authenticity of the certificate
- Confirm (or deny) that a certificate is valid
- Invalidate certificates for users who no longer are allowed access or whose private key has been exposed

PKI sets up entities, called certificate authorities, that implement the PKI policy on certificates. The general idea is that a certificate authority is trusted, so users can delegate the construction, issuance, acceptance, and revocation of certificates to the authority. The security involved in protecting the certificates involves administrative procedures. For example, more than one operator should be required to authorize certification requests. Controls should be put in place to detect hackers and prevent them from issuing bogus certificate requests. These controls might include digital signatures and strong encryption. Finally, a secure audit trail is necessary for reconstructing certificate information should the system fail and for recovering if a hacking attack does indeed corrupt the authentication process.

SSH Encryption

SSH (secure shell) is a pair of protocols (versions 1 and 2), originally defined for Unix but also available under Windows 2000, that provides an authenticated and encrypted path to the shell or operating system command interpreter. Both SSH versions replace Unix utilities such as Telnet, *rlogin*, and *rsh* for remote access. SSH protects against spoofing attacks and modification of data in communication. The SSH protocol involves negotiation between local and remote sites for encryption algorithm (for example, DES, IDEA, AES) and authentication (including public key and Kerberos).

SSL Encryption

The SSL (Secure Sockets Layer) protocol was originally designed by Netscape to protect communication between a web browser and server. It is also known now as TLS, for transport layer security. SSL interfaces between applications (such as browsers) and the TCP/IP protocols to provide server authentication, optional client authentication, and an encrypted communications channel between client and server. Client and server negotiate a mutually supported suite of encryption for session encryption and hashing; possibilities include triple DES and SHA1, or RC4 with a 128-bit key and MD5. The protocol is simple but effective, and it is the most widely used secure communication protocol on the Internet. However, remember that SSL protects only from the client's browser to the server's decryption point (which is often only to the server's firewall or, slightly stronger, to the computer that runs the web application). Data are exposed from the user's keyboard to the browser and throughout the recipient's company.

IPSec

IPSec is somewhat similar to SSL, in that it supports authentication and confidentiality in a way that does not necessitate significant change either above it (in applications) or below it (in the TCP protocols). Like SSL, it was designed to be independent of specific cryptographic protocols and to allow the two communicating parties to agree on a mutually supported set of protocols. The basis of IPSec is what is called a security association, which is essentially the set of security parameters for a secured communication channel. As with most cryptographic applications, the critical element is key management. IPSec addresses this need with ISAKMP or Internet Security Association Key Management Protocol.

Signed Code

Someone can place malicious active code on a web site to be downloaded by unsuspecting users. A partial not complete approach to reducing this risk is to use signed code. A trustworthy third party appends a digital signature to a piece of code, supposedly connoting more trustworthy code. A signature structure in a PKI helps to validate the signature.

Encrypted E-mail

An electronic mail message is much like the back of a post card. The mail carrier (and everyone in the postal system through whose hands the card passes) can read not just the address but also everything in the message field. To protect the privacy of the message and routing information, we can use encryption to protect the confidentiality of the message and perhaps its integrity.

11.2.5 Content Integrity

Content integrity comes as a bonus with cryptography. No one can change encrypted data in a meaningful way without breaking the encryption. This does not say, however, that encrypted data cannot be modified. Changing even one bit of an encrypted data stream affects the result after decryption, often in a way that seriously alters the resulting plaintext. We need to consider three potential threats:

- malicious modification that changes content in a meaningful way
- malicious or nonmalicious modification that changes content in a way that is not necessarily meaningful
- nonmalicious modification that changes content in a way that will not be detected

Encryption addresses the first of these threats very effectively. To address the others, we can use other controls.

Error Correcting Codes

We can use error detection and error correction codes to guard against modification in a transmission. The codes work as their names imply: Error detection codes detect when an error has occurred, and error correction codes can actually correct errors without requiring retransmission of the original message. The error code is transmitted along with the original data, so the recipient can recompute the error code and check whether the received result matches the expected value. The simplest error detection code is a parity check. An extra bit is added to an existing group of data bits depending on their sum or an exclusive OR. The two kinds of parity are called even and odd.

There are other kinds of error detection codes, such as hash codes and Huffman codes. Some of the more complex codes can detect multiple-bit errors (two or more bits changed in a data group) and may be able to pinpoint which bits have been changed. Parity and simple error detection and correction codes are used to detect nonmalicious changes in situations in which there may be faulty transmission equipment, communications noise and interference, or other sources of spurious changes to data.

Cryptographic Checksum

Malicious modification must be handled in a way that prevents the attacker from modifying the error detection mechanism as well as the data bits themselves. One way to do this is to use a technique that shrinks and transforms the data, according to the value of the data bits. A cryptographic checksum (sometimes called a message digest) is a cryptographic function that produces a checksum. The cryptography prevents the attacker from changing the data block (the plaintext) and also changing the checksum value (the ciphertext) to match. Two major uses of cryptographic checksums are code tamper protection and message integrity protection in transit.

11.2.6 Strong Authentication

Networked environments need authentication, too. In the network case, however, authentication may be more difficult to achieve securely because of the possibility of eavesdropping and wiretapping, which are less common in nonnetworked environments. Also, both ends of a communication may need to be authenticated to each other: Before you send your password across a network, you want to know that you are really communicating with the remote host you expect.

One-Time Password

The wiretap threat implies that a password could be intercepted from a user who enters a password across an unsecured network. A one-time password can guard against wiretapping and spoofing of a remote host. As the name implies, a one-time password is good for one use only. What are the advantages and disadvantages of this approach? First, it is easy to use. It largely counters the possibility of a wiretapper reusing a password. With a strong password-generating algorithm, it is immune to spoofing. However, the system fails if the user loses the generating device or, worse, if the device falls into an attacker's hands. Because a new password is generated only once a minute, there is a small (one-minute) window of vulnerability during which an eavesdropper can reuse an intercepted password.

Challenge Response Systems

To counter the loss and reuse problems, a more sophisticated one-time password scheme uses challenge and response. A challenge and response device looks like a simple pocket calculator. The user first authenticates to the device, usually by means of a PIN. The remote system sends a random number, called the "challenge," which the user enters into the device. The device responds to that number with another number, which the user then transmits to the system. The system prompts the user with a new challenge for each use. Thus, this device eliminates the small window of vulnerability in which a user could reuse a time-sensitive authenticator.

Digital Distributed Authentication

In the 1980s, Digital Equipment Corporation recognized the problem of needing to authenticate nonhuman entities in a computing system. For example, a process might retrieve a user query, which it then reformats, perhaps limits, and submits to a database manager. Both the database manager and the query processor want to be sure that a particular communication channel is authentic between the two. Neither of these servers is running under the direct control or supervision of a human (although each process was, of course, somehow initiated by a human). Human forms of access control are thus inappropriate. Digital created a simple architecture for this requirement, effective against the following threats:

- *impersonation* of a server by a rogue process, for either of the two servers involved in the authentication
- *interception or modification* of data exchanged between servers
- *replay* of a previous authentication

The architecture assumes that each server has its own private key and that the corresponding public key is available to or held by every other process that might need to establish an authenticated

channel.

Kerberos

Kerberos is a system that supports authentication in distributed systems. Originally designed to work with secret key encryption, Kerberos, in its latest version, uses public key technology to support key exchange. The Kerberos system was designed at Massachusetts Institute of Technology. Kerberos is used for authentication between intelligent processes, such as client-to-server tasks, or a user's workstation to other hosts. Kerberos is based on the idea that a central server provides authenticated tokens, called tickets, to requesting applications. A ticket is an unforgeable, nonreplayable, authenticated object. That is, it is an encrypted data structure naming a user and a service that user is allowed to obtain. It also contains a time value and some control information.

Kerberos was carefully designed to withstand attacks in distributed environments:

- *No passwords communicated on the network: A* user's password is stored only at the Kerberos server. The user's password is not sent from the user's workstation when the user initiates a session.
- *Cryptographic protection against spoofing:* Each access request is mediated by the ticket-granting server, which knows the identity of the requester, based on the authentication performed initially by the Kerberos server and on the fact that the user was able to present a request encrypted under a key that had been encrypted under the user's password.
- *Limited period of validity:* Each ticket is issued for a limited time period; the ticket contains a timestamp with which a receiving server will determine the ticket's validity. In this way, certain long-term attacks, such as brute force cryptanalysis, will usually be neutralized because the attacker will not have time to complete the attack.
- *Timestamps to prevent replay attacks:* Kerberos requires reliable access to a universal clock. Each user's request to a server is stamped with the time of the request. A server receiving a request compares this time to the current time and fulfills the request only if the time is reasonably close to the current time. This time-checking prevents most replay attacks, since the attacker's presentation of the ticket will be delayed too long.
- *Mutual authentication:* The user of a service can be assured of any server's authenticity by requesting an authenticating response from the server. The user sends a ticket to a server and then sends the server a request encrypted under the session key for that server's service; the ticket and the session key were provided by the ticket-granting server. The server can decrypt the ticket only if it has the unique key it shares with the ticket-granting server.

Kerberos is not a perfect answer to security problems in distributed systems.

- *Kerberos requires continuous availability of a trusted ticket-granting server:* Because the ticket-granting server is the basis of access control and authentication, constant access to that server is crucial. Both reliability (hardware or software failure) and performance (capacity and speed) problems must be addressed.
- *Authenticity of servers requires a trusted relationship between the ticket-granting server and every server:* The ticket-granting server must share a unique encryption key with each "trustworthy" server. The ticket-granting server (or that server's human administrator) must be convinced of the authenticity of that server. In a local environment, this degree of trust is warranted. In a widely distributed environment, an administrator at one site can seldom justify trust in the authenticity of servers at other sites.
- *Kerberos requires timely transactions:* To prevent replay attacks, Kerberos limits the validity of a ticket. A replay attack could succeed during the period of validity, however. And setting the period fairly is hard: Too long increases the exposure to replay attacks, while too short requires prompt user actions and risks providing the user with a ticket that will not be honoured when presented to a server. Similarly, subverting a server's clock allows reuse of an expired ticket.
- *A subverted workstation can save and later replay user passwords:* This vulnerability exists

in any system in which passwords, encryption keys, or other constant, sensitive information is entered in the clear on a workstation that might be subverted.

- *Password guessing works:* A user's initial ticket is returned under the user's password. An attacker can submit an initial authentication request to the Kerberos server and then try to decrypt the response by guessing at the password.
- *Kerberos does not scale well:* The architectural model of Kerberos assumes one Kerberos server and one ticket-granting server, plus a collection of other servers, each of which shares a unique key with the ticket-granting server. Adding a second ticket-granting server, for example, to enhance performance or reliability, would require duplicate keys or a second set for all servers. Duplication increases the risk of exposure and complicates key updates, and second keys more than double the work for each server to act on a ticket.
- *Kerberos is a complete solution:* All applications must use Kerberos authentication and access control. Currently, few applications use Kerberos authentication, and so integration of Kerberos into an existing environment requires modification of existing applications, which is not feasible.

11.2.7 Access Controls

Authentication deals with the *who* of security policy enforcement; access controls enforce the *what* and *how*.

ACLs on Routers

Routers perform the major task of directing network traffic either to subnetworks they control or to other routers for subsequent delivery to other subnetworks. Routers convert external IP addresses into internal MAC addresses of hosts on a local subnetwork. Suppose a host is being spammed (flooded) with packets from a malicious rogue host. Routers can be configured with access control lists to deny access to particular hosts from particular hosts. So, a router could delete all packets with a source address of the rogue host and a destination address of the target host.

This approach has three problems, however. First, routers in large networks perform a lot of work: They have to handle every packet coming into and going out of the network. Adding ACLs to the router requires the router to compare every packet against the ACLs. The second problem is also an efficiency issue: Because of the volume of work they perform, routers are designed to perform only essential services. Logging of activity is usually not done on a router because of the volume of traffic and the performance penalty logging would entail. With ACLs, it would be useful to know how many packets were being deleted, to know if a particular ACL could be removed (thereby improving performance). But without logging it is impossible to know whether an ACL is being used. The final limitation on placing ACLs on routers concerns the nature of the threat. A router inspects only source and destination addresses. An attacker usually does not reveal an actual source address. To reveal the real source address would be equivalent to a bank robber's leaving his home address and a description of where he plans to store the stolen money. Because someone can easily forge any source address on a UDP datagram, many attacks use UDP protocols with false source addresses so that the attack cannot be blocked easily by a router with an ACL.

Firewalls

A firewall does the screening that is less appropriate for a router to do. A router's primary function is addressing, whereas a firewall's primary function is filtering. Firewalls can also do auditing. Even more important, firewalls can examine an entire packet's contents, including the data portion, whereas a router is concerned only with source and destination MAC and IP addresses.

11.2.8 Wireless Security

Because wireless computing is so exposed, it requires measures to protect communications between a computer (called the client) and a wireless base station or access point. Remembering that all these communications are on predefined radio frequencies, you can expect an eavesdropping attacker to try to intercept and impersonate. Pieces to protect are finding the access point, authenticating the remote computer to the access point, and vice versa, and protecting the communication stream.

SSID

The Service Set Identifier or SSID is the identification of an access point; it is a string of up to 32 characters. Obviously the SSIDs need to be unique in a given area to distinguish one wireless network from another. A client and an access point engage in a handshake to locate each other. In what is called "open mode," an access point can continually broadcast its appeal, indicating that it is open for the next step in establishing a connection. Open mode is a poor security practice because it advertises the name of an access point to which an attacker might attach. "Closed" or "stealth mode" reverses the order of the protocol: The client must send a signal seeking an access point with a particular SSID before the access point responds to that one query with an invitation to connect.

WEP

The second step in securing a wireless communication involves use of encryption. The original 802.11 wireless standard relied upon a cryptographic protocol called wired equivalent privacy or WEP. WEP was meant to provide users privacy equivalent to that of a dedicated wire, that is, immunity to most eavesdropping and impersonation attacks. WEP uses an encryption key shared between the client and the access point. First, the WEP standard uses either a 64- or 128-bit encryption key. The user enters the key in any convenient form, usually in hexadecimal or as an alphanumeric string that is converted to a number.

WPA and WPA2

The alternative to WEP is WiFi Protected Access or WPA, approved in 2003. The IEEE standard 802.11i is now known as WPA2, approved in 2004, and is an extension of WPA. How does WPA improve upon WEP? The setup protocol for WPA and WPA2 is much more robust than that for WEP. Setup for WPA involves three protocol steps: authentication, a four-way handshake (to ensure that the client can generate cryptographic keys and to generate and install keys for both encryption and integrity on both ends), and an optional group key handshake (for multicast communication.)

11.2.9 Alarms and Alerts

The logical view of network protection has both a router and a firewall which provides layers of protection for the internal network. Now let us add one more layer to this defense. An intrusion detection system is a device that is placed inside a protected network to monitor what occurs within the network. If an attacker passes through the router and passes through the firewall, an intrusion detection system offers the opportunity to detect the attack at the beginning, in progress, or after it has occurred. Intrusion detection systems activate an alarm, which can take defensive action.

11.2.10 Honeypots

How do you catch a mouse? You set a trap with bait (food the mouse finds attractive) and catch the mouse after it is lured into the trap. You can catch a computer attacker the same way. You put up a honeypot for several reasons:

- to watch what attackers do, in order to learn about new attacks (so that you can strengthen your defenses against these new attacks)
- to lure an attacker to a place in which you may be able to learn enough to identify and stop

the attacker

- to provide an attractive but diversionary playground, hoping that the attacker will leave your real system alone

A honeypot has no special features. It is just a computer system or a network segment, loaded with servers and devices and data. It may be protected with a firewall, although you want the attackers to have some access. There may be some monitoring capability, done carefully so that the monitoring is not evident to the attacker. The two difficult features of a honeypot are putting up a believable, attractive false environment and confining and monitoring the attacker surreptitiously.

11.2.11 Traffic Flow Security

If the attacker can detect an exceptional volume of traffic between two points, the attacker may infer the location of an event about to occur. The countermeasure to traffic flow threats is to disguise the traffic flow. One way to disguise traffic flow, albeit costly and perhaps crude, is to ensure a steady volume of traffic between two points. A more sophisticated approach to traffic flow security is called onion routing. Packages for onion routing can be any network transmissions. The most popular uses, however, are covert email, and private web browsing. The Tor project distributes free software and enlists an open network that uses onion routing to defend against traffic analysis. Tor (which stands for The Onion Router) protects by transferring communications around a distributed network of over 5,000 relays run by volunteers all around the world: It prevents outsiders watching Internet connections from learning what sites a user visits, and it prevents sites from learning the user's physical location.

11.3 Firewalls

Firewalls were officially invented in the early 1990s, but the concept really reflects the reference monitor from two decades earlier.

11.3.1 What is a Firewall?

A firewall is a device that filters all traffic between a protected or "inside" network and a less trustworthy or "outside" network. Usually a firewall runs on a dedicated device; because it is a single point through which traffic is channelled, performance is important, which means non-firewall functions should not be done on the same machine. Because a firewall is executable code, an attacker could compromise that code and execute from the firewall's device. Thus, the fewer pieces of code on the device, the fewer tools the attacker would have by compromising the firewall. Firewall code usually runs on a proprietary or carefully minimized operating system. The purpose of a firewall is to keep "bad" things outside a protected environment. To accomplish that, firewalls implement a security policy that is specifically designed to address what bad things might happen. For example, the policy might be to prevent any access from outside (while still allowing traffic to pass *from* the inside *to* the outside). Alternatively, the policy might permit accesses only from certain places, from certain users, or for certain activities. Part of the challenge of protecting a network with a firewall is determining which security policy meets the needs of the installation.

We can describe the two schools of thought as "that which is not expressly forbidden is permitted" (default permit) and "that which is not expressly permitted is forbidden" (default deny). Users, always interested in new features, prefer the former. Security experts, relying on several decades of experience, strongly counsel the latter. An administrator implementing or configuring a firewall must choose one of the two approaches, although the administrator can often broaden the policy by setting the firewall's parameters.

11.3.2 Design of Firewalls

Remember that a reference monitor must be

- always invoked
- tamperproof
- small and simple enough for rigorous analysis

A firewall is a special form of reference monitor. By carefully positioning a firewall within a network, we can ensure that all network accesses that we want to control must pass through it. This restriction meets the "always invoked" condition. A firewall is typically well isolated, making it highly immune to modification. Usually a firewall is implemented on a separate computer, with direct connections only to the outside and inside networks. This isolation is expected to meet the "tamperproof" requirement. And firewall designers strongly recommend keeping the functionality of the firewall simple.

11.3.3 Types of Firewalls

Firewalls have a wide range of capabilities. Types of firewalls include

- packet filtering gateways or screening routers
- stateful inspection firewalls
- application proxies
- guards
- personal firewalls

Each type does different things; no one is necessarily "right" and the others "wrong." In this section, we examine each type to see what it is, how it works, and what its strengths and weaknesses are. Simplicity in a security policy is not a bad thing; the important question to ask when choosing a type of firewall is what threats an installation needs to counter.

Packet Filtering Gateway

A packet filtering gateway or screening router is the simplest, and in some situations, the most effective type of firewall. A packet filtering gateway controls access to packets on the basis of packet address (source or destination) or specific transport protocol type (such as HTTP web traffic). As described earlier in this chapter, putting ACLs on routers may severely impede their performance. But a separate firewall behind (on the local side) of the router can screen traffic before it gets to the protected network.

Packet filters do not "see inside" a packet; they block or accept packets solely on the basis of the IP addresses and ports. Thus, any details in the packet's data field (for example, allowing certain Telnet commands while blocking other services) is beyond the capability of a packet filter. Packet filters can perform the very important service of ensuring the validity of inside addresses. A packet filter sits between the inside network and the outside net, so it can know if a packet from the outside is forging an inside address. A screening packet filter might be configured to block all packets from the *outside* that claimed their source address was an *inside* address.

Stateful Inspection Firewall

Filtering firewalls work on packets one at a time, accepting or rejecting each packet and moving on to the next. They have no concept of "state" or "context" from one packet to the next. A stateful inspection firewall maintains state information from one packet to another in the input stream. One classic approach used by attackers is to break an attack into multiple packets by forcing some packets to have very short lengths so that a firewall cannot detect the signature of an attack split across two or more packets. A stateful inspection firewall would track the sequence of packets and conditions from one packet to another to thwart such an attack.

Application Proxy

An application proxy gateway, also called a bastion host, is a firewall that simulates the (proper) effects of an application so that the application receives only requests to act properly. A proxy gateway is a two-headed device: It looks to the inside as if it is the outside (destination) connection, while to the outside it responds just as the insider would.

An application proxy runs pseudoapplications. For instance, when electronic mail is transferred to a location, a sending process at one site and a receiving process at the destination communicate by a protocol that establishes the legitimacy of a mail transfer and then actually transfers the mail message. The protocol between sender and destination is carefully defined. A proxy gateway essentially intrudes in the middle of this protocol exchange, seeming like a destination in communication with the sender that is outside the firewall, and seeming like the sender in communication with the real destination on the inside. The proxy in the middle has the opportunity to screen the mail transfer, ensuring that only acceptable e-mail protocol commands are sent to the destination.

The proxies on the firewall can be tailored to specific requirements, such as logging details about accesses. They can even present a common user interface to what may be dissimilar internal functions. Suppose the internal network has a mixture of operating system types, none of which support strong authentication through a challenge response token. The proxy can demand strong authentication (name, password, and challenge response), validate the challenge response itself, and then pass on only simple name and password authentication details in the form required by a specific internal host's operating system.

Guard

A guard is a sophisticated firewall. Like a proxy firewall, it receives protocol data units, interprets them, and passes through the same or different protocol data units that achieve either the same result or a modified result. The guard decides what services to perform on the user's behalf in accordance with its available knowledge, such as whatever it can reliably know of the (outside) user's identity, previous interactions, and so forth. The degree of control a guard can provide is limited only by what is computable. But guards and proxy firewalls are similar enough that the distinction between them is sometimes fuzzy. Since the security policy implemented by the guard is somewhat more complex than the action of a proxy, the guard's code is also more complex and therefore more exposed to error. Simpler firewalls have fewer possible ways to fail or be subverted.

11.3.4 Personal Firewalls

A personal firewall is an application program that runs on a workstation to block unwanted traffic, usually from the network. A personal firewall can complement the work of a conventional firewall by screening the kind of data a single host will accept, or it can compensate for the lack of a regular firewall, as in a private DSL or cable modem connection. Just as a network firewall screens incoming and outgoing traffic for that network, a personal firewall screens traffic on a single workstation. A workstation could be vulnerable to malicious code or malicious active agents (ActiveX controls or Java applets), leakage of personal data stored on the workstation, and vulnerability scans to identify potential weaknesses.

The personal firewall is configured to enforce some policy. For example, the user may decide that certain sites, such as computers on the company network, are highly trustworthy, but most other sites are not. Combining a virus scanner with a personal firewall is both effective and efficient. A personal firewall runs on the very computer it is trying to protect. Thus, a clever attacker is likely to attempt an undetected attack that would disable or reconfigure the firewall for the future. Still, especially for cable modem, DSL, and other "always on" connections, the static workstation is a visible and vulnerable target for an ever-present attack community. A personal firewall can provide

reasonable protection to clients that are not behind a network firewall.

11.3.5 Comparison of Firewall Types

We can summarize the differences among the several types of firewalls we have studied in depth. The comparisons are shown in Table 10-3. Firewall types are arranged generally from least sophisticated on the left to more so on the right, with the exception of personal firewalls, which are more like an enterprise packet filter.

Table 11-2 Comparison of Firewall Types

Packet Filter	Stateful Inspection	Application Proxy	Circuit Gateway	Guard	Personal Firewall
Simplest decision-making rules, packet by packet	Correlates data across packets	Simulates effect of an application program	Joins two subnetworks	Implements any conditions that can be programmed	Similar to packet filter, but getting more complex
Sees only addresses and service protocol type	Can see addresses and data	Sees and analyzes full data portion of pack	Sees addresses and data	Sees and analyzes full content of data	Can see full data portion
Auditing limited because of speed limitations	Auditing possible	Auditing likely	Auditing likely	Auditing likely	Auditing likely
Screens based on connection rules	Screens based on information across multiple packets—in either headers or data	Screens based on behavior of application	Screens based on address	Screens based on interpretation of content	Typically, screens based on content of each packet individually, based on address or content
Complex addressing rules can make configuration tricky	Usually preconfigured to detect certain attack signatures	Simple proxies can substitute for complex decision rules, but proxies must be aware of application's behavior	Relatively simple addressing rules; make configuration straightforward	Complex guard functionality; can be difficult to define and program accurately	Usually starts in mode to deny all inbound traffic; adds addresses and functions to trust as they arise

11.3.6 What Firewalls Can- and Cannot- Block

As we have seen, firewalls are not complete solutions to all computer security problems. A firewall protects only the perimeter of its environment against attacks from outsiders who want to execute code or access data on the machines in the protected environment. Keep in mind these points about firewalls.

- Firewalls can protect an environment only if the firewalls control the entire perimeter. That is, firewalls are effective only if no unmediated connections breach the perimeter. If even one inside host connects to an outside address, by a modem for example, the entire inside net is vulnerable through the modem and its host.
- Firewalls do not protect data outside the perimeter; data that have properly passed (outbound) through the firewall are just as exposed as if there were no firewall.

- Firewalls are the most visible part of an installation to the outside, so they are the most attractive target for attack. For this reason, several different layers of protection, called defense in depth, are better than relying on the strength of just a single firewall.
- Firewalls must be correctly configured, that configuration must be updated as the internal and external environment changes, and firewall activity reports must be reviewed periodically for evidence of attempted or successful intrusion.
- Firewalls are targets for penetrators. While a firewall is designed to withstand attack, it is not impenetrable. Designers intentionally keep a firewall small and simple so that even if a penetrator breaks it, the firewall does not have further tools, such as compilers, linkers, loaders, and the like, to continue an attack.
- Firewalls exercise only minor control over the content admitted to the inside, meaning that inaccurate data or malicious code must be controlled by other means inside the perimeter.

Firewalls are important tools in protecting an environment connected to a network. However, the environment must be viewed as a whole, all possible exposures must be considered, and the firewall must fit into a larger, comprehensive security strategy. Firewalls alone cannot secure an environment.

11.4 Intrusion Detection Systems

After the perimeter controls, firewall, and authentication and access controls block certain actions, some users are admitted to use a computing system. Most of these controls are preventive: They block known bad things from happening. Many studies have shown that most computer security incidents are caused by insiders, people who would not be blocked by a firewall. And insiders require access with significant privileges to do their daily jobs. The vast majority of harm from insiders is not malicious; it is honest people making honest mistakes. Then, too, there are the potential malicious outsiders who have somehow passed the screens of firewalls and access controls. Prevention, although necessary, is not a complete computer security control; detection during an incident copes with harm that cannot be prevented in advance.

Intrusion detection systems complement these preventive controls as the next line of defense. An intrusion detection system (IDS) is a device, typically another separate computer, that monitors activity to identify malicious or suspicious events. An IDS is a sensor, like a smoke detector, that raises an alarm if specific things occur. An IDS receives raw inputs from sensors. It saves those inputs, analyzes them, and takes some controlling action.

IDSs perform a variety of functions:

- monitoring users and system activity
- auditing system configuration for vulnerabilities and misconfigurations
- assessing the integrity of critical system and data files
- recognizing known attack patterns in system activity
- identifying abnormal activity through statistical analysis
- managing audit trails and highlighting user violation of policy or normal activity
- correcting system configuration errors
- installing and operating traps to record information about intruders

No one IDS performs all of these functions. Let us look more closely at the kinds of IDSs and their use in providing security.

11.4.1 Types of IDSs

The two general types of intrusion detection systems are signature based and heuristic. Signature-based intrusion detection systems perform simple pattern-matching and report situations that match

a pattern corresponding to a known attack type. Heuristic intrusion detection systems, also known as anomaly based, build a model of acceptable behaviour and flag exceptions to that model; for the future, the administrator can mark a flagged behaviour as acceptable so that the heuristic IDS will now treat that previously unclassified behaviour as acceptable.
Intrusion detection devices can be network based or host based. A network-based IDS is a stand-alone device attached to the network to monitor traffic throughout that network; a host-based IDS runs on a single workstation or client or host, to protect that one host.

Signature-Based Intrusion Detection
A simple signature for a known attack type might describe a series of TCP SYN packets sent to many different ports in succession and at times close to one another, as would be the case for a port scan. An intrusion detection system would probably find nothing unusual in the first SYN, say, to port 80, and then another (from the same source address) to port 25. But as more and more ports receive SYN packets, especially ports that are not open, this pattern reflects a possible port scan. Similarly, some implementations of the protocol stack fail if they receive an ICMP packet with a data length of 65535 bytes, so such a packet would be a pattern for which to watch.
The problem with signature-based detection is the signatures themselves. An attacker will try to modify a basic attack in such a way that it will not match the known signature of that attack. For example, the attacker may convert lowercase to uppercase letters or convert a symbol such as "blank space" to its character code equivalent %20. The IDS must necessarily work from a canonical form of the data stream in order to recognize that %20 matches a pattern with a blank space. The attacker may insert malformed packets that the IDS will see, to intentionally cause a pattern mismatch; the protocol handler stack will discard the packets because of the malformation. Each of these variations could be detected by an IDS, but more signatures require additional work for the IDS, which reduces performance.
Of course, signature-based IDSs cannot detect a new attack for which a signature is not yet installed in the database. Every attack type starts as a new pattern at some time, and the IDS is helpless to warn of its existence. Signature-based intrusion detection systems tend to use statistical analysis. This approach uses statistical tools both to obtain sample measurements of key indicators and to determine whether the collected measurements fit the predetermined attack signatures. Ideally, signatures should match every instance of an attack, match subtle variations of the attack, but not match traffic that is not part of an attack. However, this goal is grand but unreachable.

Heuristic Intrusion Detection
Because signatures are limited to specific, known attack patterns, another form of intrusion detection becomes useful. Instead of looking for matches, heuristic intrusion detection looks for behaviour that is out of the ordinary. The inference engine of an intrusion detection system performs continuous analysis of the system, raising an alert when the system's dirtiness exceeds a threshold. Inference engines work in two ways. Some, called state-based intrusion detection systems, see the system going through changes of overall state or configuration. They try to detect when the system has veered into unsafe modes. Others try to map current activity onto a model of unacceptable activity and raise an alarm when the activity resembles the model. These are called model-based intrusion detection systems. This approach has been extended to networks. Alternatively, intrusion detection can work from a model of known bad activity. For example, except for a few utilities (login, change password, create user), any other attempt to access a password file is suspect. This form of intrusion detection is known as misuse intrusion detection.
All heuristic intrusion detection activity is classified in one of three categories: good/benign, suspicious, or unknown. Over time, specific kinds of actions can move from one of these categories to another, corresponding to the IDS's learning whether certain actions are acceptable or not. As with pattern-matching, heuristic intrusion detection is limited by the amount of information the system has seen (to classify actions into the right category) and how well the current actions fit into

one of these categories.

Stealth Mode

An IDS is a network device (or, in the case of a host-based IDS, a program running on a network device). Any network device is potentially vulnerable to network attacks. How useful would an IDS be if it itself were deluged with a denial-of-service attack? If an attacker succeeded in logging in to a system within the protected network, wouldn't trying to disable the IDS be the next step? To counter those problems, most IDSs run in stealth mode, whereby an IDS has two network interfaces: one for the network (or network segment) being monitored and the other to generate alerts and perhaps other administrative needs. The IDS uses the monitored interface as input only; it *never* sends packets out through that interface. Often, the interface is configured so that the device has no published address through the monitored interface; that is, a router cannot route anything to that address directly, because the router does not know such a device exists. It is the perfect passive wiretap. If the IDS needs to generate an alert, it uses only the alarm interface on a completely separate control network.

Other IDS Types

Some security engineers consider other devices to be IDSs as well. For instance, to detect unacceptable code modification, programs can compare the active version of a software code with a saved version of a digest of that code. The *tripwire* program is the most well-known software (or static data) comparison program. You run *tripwire* on a new system, and it generates a hash value for each file; then you save these hash values in a secure place (offline, so that no intruder can modify them while modifying a system file). If you later suspect your system may have been compromised, you rerun *tripwire*, providing it the saved hash values. It recomputes the hash values and reports any mismatches, which would indicate files that were changed. System vulnerability scanners, such as *ISS Scanner* or *Nessus*, can be run against a network. They check for known vulnerabilities and report flaws found. As we have seen, a honeypot is a faux environment intended to lure an attacker. It can be considered an IDS, in the sense that the honeypot may record an intruder's actions and even attempt to trace who the attacker is from actions, packet data, or connections.

11.4.2 Goals for Intrusion Detection Systems

The two styles of intrusion detection pattern matching and heuristic represent different approaches, each of which has advantages and disadvantages. Actual IDS products often blend the two approaches. Ideally, an IDS should be fast, simple, and accurate, while at the same time being complete. It should detect all attacks with little performance penalty. An IDS could use some or all of the following design approaches:

- Filter on packet headers
- Filter on packet content
- Maintain connection state
- Use complex, multipacket signatures
- Use minimal number of signatures with maximum effect
- Filter in real time, online
- Hide its presence
- Use optimal sliding time window size to match signatures

Responding to Alarms

Whatever the type, an intrusion detection system raises an alarm when it finds a match. The alarm can range from something modest, such as writing a note in an audit log, to something significant, such as paging the system security administrator. Particular implementations allow the user to

determine what action the system should take on what events. What are possible responses? The range is unlimited and can be anything the administrator can imagine (and program). In general, responses fall into three major categories (any or all of which can be used in a single response):

- Monitor, collect data, perhaps increase amount of data collected
- Protect, act to reduce exposure
- Call a human

Monitoring is appropriate for an attack of modest (initial) impact. Perhaps the real goal is to watch the intruder, to see what resources are being accessed or what attempted attacks are tried. Another monitoring possibility is to record all traffic from a given source for future analysis. This approach should be invisible to the attacker. Protecting can mean increasing access controls and even making a resource unavailable. In contrast to monitoring, protecting may be very visible to the attacker. Finally, calling a human allows individual discrimination. The IDS can take an initial defensive action immediately while also generating an alert to a human who may take seconds, minutes, or longer to respond.

False Results

Intrusion detection systems are not perfect, and mistakes are their biggest problem. Although an IDS might detect an intruder correctly most of the time, it may stumble in two different ways: by raising an alarm for something that is not really an attack (called a false positive, or type I error in the statistical community) or not raising an alarm for a real attack (a false negative, or type II error). Too many false positives means the administrator will be less confident of the IDS's warnings, perhaps leading to a real alarm's being ignored. But false negatives mean that real attacks are passing the IDS without action. We say that the degree of false positives and false negatives represents the sensitivity of the system. Most IDS implementations allow the administrator to tune the system's sensitivity, to strike an acceptable balance between false positives and negatives.

11.4.3 IDS Strengths and Limitations

Intrusion detection systems are evolving products. Research began in the mid-1980s and products had appeared by the mid-1990s. However, this area continues to change as new research influences the design of products. On the upside, IDSs detect an ever-growing number of serious problems. And as we learn more about problems, we can add their signatures to the IDS model. Thus, over time, IDSs continue to improve. At the same time, they are becoming cheaper and easier to administer. On the downside, avoiding an IDS is a first priority for successful attackers. An IDS that is not well defended is useless. Fortunately, stealth mode IDSs are difficult even to find on an internal network, let alone to compromise. IDSs look for known weaknesses, whether through patterns of known attacks or models of normal behaviour. Similar IDSs may have identical vulnerabilities, and their selection criteria may miss similar attacks. Knowing how to evade a particular model of IDS is an important piece of intelligence passed within the attacker community. Of course, once manufacturers become aware of a shortcoming in their products, they try to fix it. Fortunately, commercial IDSs are pretty good at identifying attacks. Another IDS limitation is its sensitivity, which is difficult to measure and adjust. IDSs will never be perfect, so finding the proper balance is critical. A final limitation is not of IDSs *per se*, but is one of use. An IDS does not run itself; someone has to monitor its track record and respond to its alarms. An administrator is foolish to buy and install an IDS and then ignore it.

In general, IDSs are excellent additions to a network's security. Firewalls block traffic to particular ports or addresses; they also constrain certain protocols to limit their impact. But by definition, firewalls have to allow some traffic to enter a protected area. Watching what that traffic actually does inside the protected area is an IDS's job, which it does quite well.

11.5 Secure E-mail

The final control we consider in depth is secure e-mail. Think about how much you use e-mail and how much you rely on the accuracy of its contents. How would you react if you received a message from your instructor saying that because you had done so well in your course so far, you were excused from doing any further work in it? What if that message were a joke from a classmate? We rely on e-mail's confidentiality and integrity for sensitive and important communications, even though ordinary e-mail has almost no confidentiality or integrity. In this section we investigate how to add confidentiality and integrity protection to ordinary e-mail.

11.5.1 Security for E-mail

E-mail is vital for today's commerce, as well a convenient medium for communications among ordinary users. But, as we noted earlier, e-mail is very public, exposed at every point from the sender's workstation to the recipient's screen. Just as you would not put sensitive or private thoughts on a postcard, you must also acknowledge that e-mail messages are exposed and available for others to read. Sometimes we would like e-mail to be more secure. To define and implement a more secure form, we begin by examining the exposures of ordinary e-mail.

Threats to E-mail

Consider threats to electronic mail:

- message interception (confidentiality)
- message interception (blocked delivery)
- message interception and subsequent replay
- message content modification
- message origin modification
- message content forgery by outsider
- message origin forgery by outsider
- message content forgery by recipient
- message origin forgery by recipient
- denial of message transmission

Confidentiality and content forgery are often handled by encryption. Encryption can also help in a defense against replay, although we would also have to use a protocol in which each message contains something unique that is encrypted. Symmetric encryption cannot protect against forgery by a recipient, since both sender and recipient share a common key; however, public key schemes can let a recipient decrypt but not encrypt. Because of lack of control over the middle points of a network, senders or receivers generally cannot protect against blocked delivery.

11.5.2 Requirements and Solutions

If we were to make a list of the requirements for secure e-mail, our wish list would include the following protections:

- *message confidentiality* (the message is not exposed en route to the receiver)
- *message integrity* (what the receiver sees is what was sent)
- *sender authenticity* (the receiver is confident who the sender was)
- *nonrepudiation* (the sender cannot deny having sent the message)

Not all these qualities are needed for every message, but an ideal secure e-mail package would allow these capabilities to be invoked selectively.

IDOL

11.5.3 Designs

The standard for encrypted e-mail was developed by the Internet Society, through its architecture board (IAB) and research (IRTF) and engineering (IETF) task forces. The encrypted e-mail protocols are documented as an Internet standard in documents 1421, 1422, 1423, and 1424. This standard is actually the third refinement of the original specification. One of the design goals for encrypted e-mail was allowing security-enhanced messages to travel as ordinary messages through the existing Internet e-mail system. This requirement ensures that the large existing e-mail network would not require change to accommodate security. Thus, all protection occurs within the body of a message.

Confidentiality

Because the protection has several aspects, we begin our description of them by looking first at how to provide confidentiality enhancements. The sender chooses a (random) symmetric algorithm encryption key. Then, the sender encrypts a copy of the entire message to be transmitted, including FROM:, TO:, SUBJECT:, and DATE: headers. Next, the sender prepends plaintext headers. For key management, the sender encrypts the message key under the recipient's public key and attaches that to the message as well. Encryption can potentially yield any string as output. Many e-mail handlers expect that message traffic will not contain characters other than the normal printable characters. Network e-mail handlers use unprintable characters as control signals in the traffic stream. To avoid problems in transmission, encrypted e-mail converts the entire ciphertext message to printable characters.

The encrypted e-mail standard works most easily, using both symmetric and asymmetric encryption. The encrypted e-mail standard supports multiple encryption algorithms, using popular algorithms such as DES, triple DES, and AES for message confidentiality, and RSA and Diffie-Hellman for key exchange.

Other Security Features

In addition to confidentiality, we may want various forms of integrity for secure e-mail. Encrypted e-mail messages always carry a digital signature, so the authenticity and non-repudiability of the sender is assured. The integrity is also assured because of a hash function (called a message integrity check, or MIC) in the digital signature. Optionally, encrypted e-mail messages can be encrypted for confidentiality.

Encryption for Secure E-mail

The major problem with encrypted e-mail is key management. The certificate scheme is excellent for exchanging keys and for associating an identity with a public encryption key. The difficulty with certificates is building the hierarchy. Many organizations have hierarchical structures. The encrypted e-mail dilemma is moving beyond the single organization to an interorganizational hierarchy. Precisely because of the problem of imposing a hierarchy on a non-hierarchical world, PGP was developed as a simpler form of encrypted e-mail. Encrypted e-mail provides strong end-to-end security for electronic mail. Triple DES, AES, and RSA cryptography are quite strong, especially if RSA is used with a long bit key. The vulnerabilities remaining with encrypted e-mail come from the points not covered: the endpoints. An attacker with access could subvert a sender's or receiver's machine, modifying the code that does the privacy enhancements or arranging to leak a cryptographic key.

11.5.4 Example Secure E-mail Systems

Encrypted e-mail programs are available from many sources. Several universities (including Cambridge University in England and The University of Michigan in the United States) and companies (BBN, RSA-DSI, and Trusted Information Systems) have developed either prototype or

commercial versions of encrypted e-mail.

PGP

PGP stands for Pretty Good Privacy. It was invented by Phil Zimmerman in 1991. Originally a free package, it became a commercial product after being bought by Network Associates in 1996. A freeware version is still available. PGP is widely available, both in commercial versions and freeware, and it is heavily used by individuals exchanging private e-mail. PGP addresses the key distribution problem with what is called a "ring of trust" or a user's "keyring." One user directly gives a public key to another, or the second user fetches the first's public key from a server. Some people include their PGP public keys at the bottom of e-mail messages. And one person can give a second person's key to a third (and a fourth, and so on). Thus, the key association problem becomes one of caveat emptor: "Let the buyer beware." If I am reasonably confident that an e-mail message really comes from you and has not been tampered with, I will use your attached public key. If I trust you, I may also trust the keys you give me for other people. The model breaks down intellectually when you give me all the keys you received from people, who in turn gave you all the keys they got from still other people, who gave them all their keys, and so forth. PGP does not mandate a policy for establishing trust. Rather, each user is free to decide how much to trust each key received.

The PGP processing performs some or all of the following actions, depending on whether confidentiality, integrity, authenticity, or some combination of these is selected:

- Create a random session key for a symmetric algorithm.
- Encrypt the message, using the session key (for message confidentiality).
- Encrypt the session key under the recipient's public key.
- Generate a message digest or hash of the message; sign the hash by encrypting it with the sender's private key (for message integrity and authenticity).
- Attach the encrypted session key to the encrypted message and digest.
- Transmit the message to the recipient.

The recipient reverses these steps to retrieve and validate the message content.

S/MIME

An Internet standard governs how e-mail is sent and received. The general MIME specification defines the format and handling of e-mail attachments. S/MIME (Secure Multipurpose Internet Mail Extensions) is the Internet standard for secure e-mail attachments. S/MIME is very much like PGP and its predecessors, PEM (Privacy-Enhanced Mail) and RIPEM. S/MIME has been adopted in commercial e-mail packages, such as Eudora and Microsoft Outlook. The principal difference between S/MIME and PGP is the method of key exchange. Basic PGP depends on each user's exchanging keys with all potential recipients and establishing a ring of trusted recipients; it also requires establishing a degree of trust in the authenticity of the keys for those recipients. S/MIME uses hierarchically validated certificates, usually represented in X.509 format, for key exchange. Thus, with S/MIME, the sender and recipient do not need to have exchanged keys in advance as long as they have a common certifier they both trust.

S/MIME works with a variety of cryptographic algorithms, such as DES, AES, and RC2 for symmetric encryption. S/MIME performs security transformations very similar to those for PGP. PGP was originally designed for plaintext messages, but S/MIME handles (secures) all sorts of attachments, such as data files (for example, spreadsheets, graphics, presentations, movies, and sound). Because it is integrated into many commercial e-mail packages, S/MIME is likely to dominate the secure e-mail market.

11.6 Example Protocols

Much of the software currently used to protect the confidentiality of information are not true cryptosystems. Instead, they are applications to which cryptographic protocols have been added.

This is perhaps particularly true of Internet protocols; some experts claim that the Internet and its corresponding protocols were designed without any consideration for security, which was added later as an afterthought. Whether or not this is true, the lack of threats in the environment in which it was launched allowed the Internet to grow rapidly. But as the number of threats grew, so did the need for additional security measures.

11.6.1 SSL

Netscape developed the **Secure Sockets Layer (SSL)** protocol to use public key encryption to secure a channel over the Internet, thus enabling secure communications. Most popular browsers, including Internet Explorer, use SSL. In addition to providing data encryption, integrity, and server authentication, SSL can, when properly configured, provide client authentication.

The SSL protocol works as follows: during a normal client/server HTTP session, the client requests access to a portion of the Web site that requires secure communications, and the server sends a message to the client indicating that a secure connection must be established. The client sends its public key and security parameters. This handshaking phase is complete when the server finds a public key match and sends a digital certificate to the client in order to authenticate itself. Once the client verifies that the certificate is valid and trustworthy, the SSL session is established. Until the client or the server terminates the session, any amount of data can be transmitted securely.

SSL provides two protocol layers within the TCP framework: SSL Record Protocol and Standard HTTP. The **SSL Record Protocol** is responsible for the fragmentation, compression, encryption, and attachment of an SSL header to the plaintext prior to transmission. Received encrypted messages are decrypted and reassembled for presentation to the higher levels of the protocol. The SSL Record Protocol provides basic security and communication services to the top levels of the SSL protocol stack.

11.6.2 PEM

A number of cryptosystems have been adapted to work with the dominant e-mail protocols in an attempt to incorporate some degree of security into this notoriously insecure communication medium. Some of the more popular adaptations included Secure Multipurpose Internet Mail Extensions, Privacy Enhanced Mail (PEM), and Pretty Good Privacy (PGP). Secure Multipurpose Internet Mail Extensions (S/MIME) builds on the encoding format of the Multipurpose Internet Mail Extensions (MIME) protocol and uses digital signatures based on public key cryptosystems to secure e-mail. Privacy Enhanced Mail (PEM) was proposed by the Internet Engineering Task Force (IETF) and is a standard that uses DES3 symmetric key encryption and RSA for key exchanges and digital signatures. Pretty Good Privacy (PGP) was developed by Phil Zimmermann and uses the IDEA cipher for message encoding. PGP also uses RSA for symmetric key exchange and digital signatures.

PEM employs a range of cryptographic techniques to allow for confidentiality, sender authentication, and message integrity. The message integrity aspects allow the user to ensure that a message hasn't been modified during transport from the sender. The sender authentication allows a user to verify that the PEM message that they have received is truly from the person who claims to have sent it. The confidentiality feature allows a message to be kept secret from people to whom the message was not addressed.

11.6.3 IPSec

Internet Protocol Security (IPSec) is an open-source protocol framework for security development within the TCP/IP family of protocol standards. It is used to secure communications across IP-based networks such as LANs, WANs, and the Internet. The protocol is designed to

protect data integrity, user confidentiality, and authenticity at the IP packet level. IPSec is the cryptographic authentication and encryption product of the IETF's IP Protocol Security Working Group. It is often described as the security system from IP version 6 (the future version of the TCP/IP protocol), retrofitted for use with IP version 4 (the current version). IPSec is defined in Request for Comments (RFC) 1825, 1826, and 1827 and is widely used to create virtual private networks (VPNs). IPSec itself is actually an open framework. IPSec includes the IP Security protocol itself, which specifies the information to be added to an IP packet as well as how to encrypt packet data; and the Internet Key Exchange, which uses an asymmetric-based key exchange and negotiates the security associations. IPSec operates in two modes: transport and tunnel. In **transport mode** only the IP data are encrypted, not the IP headers. This allows intermediate nodes to read the source and destination addresses. In **tunnel mode** the entire IP packet is encrypted and is then placed into the content portion of another IP packet. This requires other systems at the beginning and end of the tunnel to act as proxies and to send and receive the encrypted packets. These systems then transmit the decrypted packets to their true destinations. IPSec uses several different cryptosystems:

- Diffie-Hellman key exchange for deriving key material between peers on a public network
- Public key cryptography for signing the Diffie-Hellman exchanges to guarantee the identity of the two parties
- Bulk encryption algorithms, such as DES, for encrypting the data
- Digital certificates signed by a certificate authority to act as digital ID cards

Within IPSec, IP layer security is achieved by means of an application header protocol or an encapsulating security payload protocol. The **application header (AH) protocol** provides system-to-system authentication and data integrity verification but does not provide secrecy for the content of a network communication. The **encapsulating security payload (ESP) protocol** provides secrecy for the contents of network communications as well as system-to-system authentication and data integrity verification. When two networked systems form an association that uses encryption and authentication keys, algorithms, and key lifetimes, they can implement either the AH or the ESP protocol, but not both.

The AH protocol is designed to provide data integrity and IP packet authentication. Although AH does not provide confidentiality protection, IP packets are protected from replay attacks and address spoofing as well as other types of cyberattacks against open networks.

The encapsulating security payload protocol provides confidentiality services for IP packets across insecure networks. ESP can also provide the authentication services of AH. ESP in tunnel mode can be used to establish a virtual private network, assuring encryption and authentication between networks communicating via the Internet. In tunnel mode, the entire IP packet is encrypted with the attached ESP header. A new IP header is attached to the encrypted payload, providing the required routing information. Figure 10-1 shows the packet format of the IPSec AH and ESP protocol.

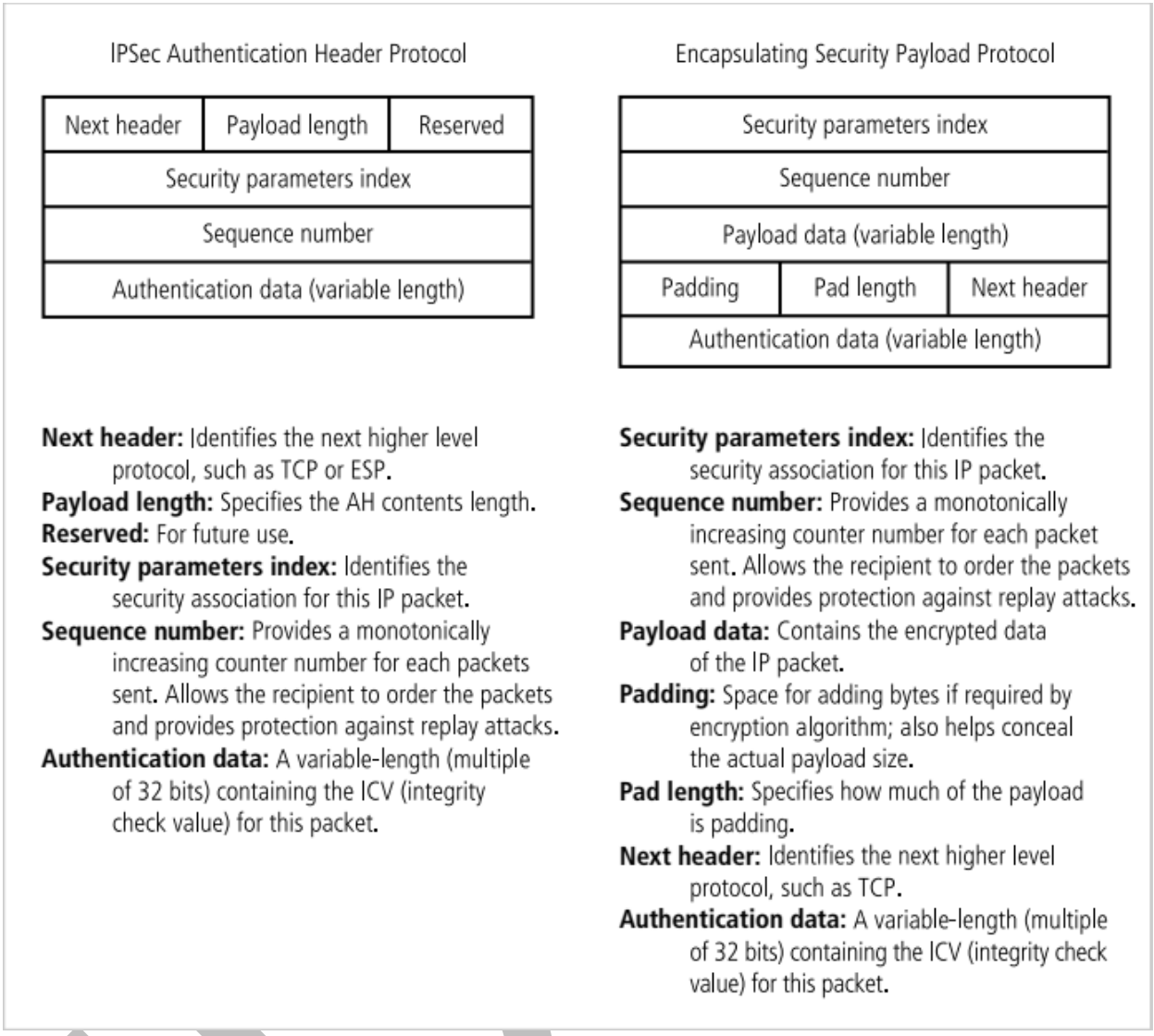


Figure 11-1 IPSec Headers

11.7 Summary

Security in networks is the combination and culmination of everything we know about security, and certainly everything we have discussed in this book so far. A network's security depends on all the cryptographic tools at our disposal, good program development processes, operating system controls, trust and evaluation and assurance methods, and inference and aggregation controls.

Networks and their security remind us that good software engineering practices can go a long way toward making software difficult to attack. When a network and its components are structured, designed, and architected well, the resulting system presents solid defenses and avoids potential single points of failure. And a well-engineered network is easy to change as it evolves; because it is easier to understand, changes seldom introduce unintentional flaws.

Many of the controls useful for stand-alone systems are also useful in networks. But three controls are specific to networks: firewalls, intrusion detection systems, and secure e-mail. These controls have evolved from many years of research, both in security and in other computer science realms. They emphasize why we should know not only the history of security but also the relevance of other computing research. For example, firewalls are just an updated form of reference monitor. Similarly, intrusion detection profits from more fundamental research into pattern matching and expert systems. And secure e-mail is really a carefully designed application of cryptography. You might think that controls such as these are the result of strokes of genius. But in fact, they reflect the

long-term nature of knowledge and engineering practice; new ways to provide security build on a growing base of understanding and experience.
Until now we have stressed technical controls, which can be very effective in protecting our computing assets. But many security losses come from trusted insiders either honest people making honest, human mistakes or dishonest insiders able to capitalize on their knowledge or privileges. In the next chapter we consider administrative controls, such as security policies, user awareness, and risk analysis, as a way to address the insider threat.

11.8 Review Questions

a) What are the key architecture controls in network security?
b) What is Link encryption and end-to-end encryption? Compare them.
c) How can content integrity be maintained?
d) Write a short note on PKI and certificates.
e) Write a short note on Kerberos.
f) What measures are required for Wireless security?
g) Explain Honeypots.
h) What is firewall? Explain the different types of firewalls.
i) Compare the different types of firewalls.
j) What can and cannot be blocked by firewalls?
k) What is IDS? Explain the types of IDSs.
l) What are the goals for IDSs?
m) Write a short note on Secure E-mail.
n) Write a short note on IPSec protocol.

11.9 Bibliography, References and Further Reading

- *Security in Computing by* C. P. Pfleeger, and S. L. Pfleeger, Pearson Education.
- *Computer Security: Art and Science by* Matt Bishop, Pearson Education.
- *Cryptography And Network Security: Principles and practice by* Stallings
- *Network Security by* Kaufman, Perlman, Speciner
- *Network Security : A Beginner's Guide by* Eric Maiwald, TMH
- *Java Network Security by* Macro Pistoia, Pearson Education
- *Principles of information security by* Whitman, Mattord, Thomson

IDOL

Chapter 12

Security Planning and Risk Analysis

IDOL

12.0 Objectives

In this chapter, we move above the technical knowledge and introduce the management aspects of security planning and risk analysis.

12.1 Introduction

Security is a combination of technical, administrative, and physical controls, as we first pointed out in the earlier chapters. So far, we have considered technical controls almost exclusively. But stop and think for a moment: What good is a firewall if there is no power to run it? How effective is a public key infrastructure if someone can walk off with the certificate server? And why have elaborate access control mechanisms if your employee mails a sensitive document to a competitor? The administrative and physical controls may be less glamorous than the technical ones, but they are surely as important. In this and the next chapter we complete

our study of security controls by considering administrative and physical aspects. We look at four related areas:

- *Planning:* What advance preparation and study lets us know that our implementation meets our security needs for today and tomorrow?
- *Risk analysis:* How do we weigh the benefits of controls against their costs, and how do we justify any controls?
- Policy: How do we establish a framework to see that our computer security needs continue to be met?
- *Physical control:* What aspects of the computing environment have an impact on security?

These four areas are just as important to achieving security as are the latest firewall or coding practice.

12.2 Security Planning

Years ago, when most computing was done on mainframe computers, data processing centres were responsible for protection. Responsibility for security rested neither with the programmers nor the users but instead with the computing centres themselves. These centres developed expertise in security, and they implemented many protection activities in the background, without users having to be conscious of protection needs and practices. Since the early 1980s, the introduction of personal computers and the general ubiquity of computing have changed the way many of us work and interact with computers. In particular, a significant amount of the responsibility for security has shifted to the user and away from the computing center. But many users are unaware of (or choose to ignore) this responsibility, so they do not deal with the risks posed or do not implement simple measures to prevent or mitigate problems. Unfortunately, there are many common examples of this neglect. Moreover, it is exacerbated by the seemingly hidden nature of important data: Things we would protect if they were on paper are ignored when they are stored electronically. For example, a person who carefully locks up paper copies of company confidential records overnight may leave running a personal computer or terminal on an assistant's or manager's desk. In this situation, a curious or malicious person walking past can retrieve confidential memoranda and data. Similarly, the data on laptops and workstations are often more easily available than on older, more isolated systems. For instance, the large and cumbersome disk packs and tapes from a few years ago have been replaced by media such as diskettes, zip disks, and CDs, which hold a similar volume of data but fit easily in a pocket or briefcase. Moreover, we all recognize that a box of CDs or diskettes may contain many times more data than a printed report. But since the report is an apparent, visible exposure and the CD or diskette is not, we leave the computer media in plain view, easy to borrow or steal. In all cases, whether the user initiates some computing action or simply interacts with an active application, every application has confidentiality, integrity, and availability requirements that relate to the data, programs, and computing machinery. In these situations, users suffer from lack of sensitivity: They often do not appreciate the security risks associated with using computers.

For these reasons, every organization using computers to create and store valuable assets should perform thorough and effective security planning. A security plan is a document that describes how an organization will address its security needs. The plan is subject to periodic review and revision as the organization's security needs change. A good security plan is an official record of current security practices, plus a blueprint for orderly change to improve those practices. By following the plan, developers and users can measure the effect of proposed changes, leading eventually to further improvements. The impact of the security plan is important, too. A carefully written plan, supported by management, notifies employees that security is important to management. Thus, the security plan has to have the appropriate content and produce the desired effects.

In this section we focus on three aspects of writing a security plan: what it should contain, who writes it, and how to obtain support for it. Then, we address two specific cases of security plans: business continuity plans, to ensure that an organization continues to function in spite of a computer security incident, and incident response plans, to organize activity to deal with the crisis of an incident.

12.2.1 Contents of a Security Plan

A security plan identifies and organizes the security activities for a computing system. The plan is both a description of the current situation and a plan for improvement. Every security plan must address seven

issues.

1. ***policy***, indicating the goals of a computer security effort and the willingness of the people involved to work to achieve those goals
2. ***current state***, describing the status of security at the time of the plan
3. ***requirements***, recommending ways to meet the security goals
4. ***recommended controls***, mapping controls to the vulnerabilities identified in the policy and requirements
5. ***accountability***, describing who is responsible for each security activity
6. ***timetable***, identifying when different security functions are to be done
7. ***continuing attention***, specifying a structure for periodically updating the security plan

There are many approaches to creating and updating a security plan. Some organizations have a formal, defined security planning process, much as they might have a defined and accepted development or maintenance process. Others look to security professionals for guidance on how to perform security planning. But every security plan contains the same basic material, no matter the format. The following sections expand on the seven parts of a security plan.

1. Policy

A security plan must state the organization's policy on security. A security policy is a high -level statement of purpose and intent. Initially, you might think that all policies would be the same: to prevent security breaches. But in fact, the policy is one of the most difficult sections to write well. As we discuss later in this chapter, there are trade-offs among the strength of the security, the cost, the inconvenience to users, and more. For example, we must decide whether to implement very stringent and possibly unpopular controls that prevent all security problems or simply mitigate the effects of security breaches once they happen. For this reason, the policy statement must answer three essential questions:

- *Who* should be allowed access?
- To what system and organizational *resources* should access be allowed?
- What *types* of access should each user be allowed for each resource?

The policy statement should specify the following:

- The organization's *goals* on security. For example, should the system protect data from leakage to outsiders, protect against loss of data due to physical disaster, protect the data's integrity, or protect against loss of business when computing resources fail? What is the higher priority: serving customers or securing data?
- Where the *responsibility* for security lies. For example, should the responsibility rest with a small computer security group, with each employee, or with relevant managers?
- The organization's *commitment* to security. For example, who provides security support for staff, and where does security fit into the organization's structure?

2. Current Security Status

To be able to plan for security, an organization must understand the vulnerabilities to which it may be exposed. The organization can determine the vulnerabilities by performing a risk analysis: a careful investigation of the system, its environment, and the things that might go wrong. The risk analysis forms the basis for describing the current status of security. The status can be expressed as a listing of organizational assets, the security threats to the assets, and the controls in place to protect the assets. The status portion of the plan also defines the limits of responsibility for security. It describes not only which assets are to be protected but also who is responsible for protecting them. The plan may note that some groups may be excluded from responsibility; for example, joint ventures with other organizations may designate one organization to provide security for all member organizations. The plan also defines the boundaries of responsibility, especially when networks are involved. For instance, the plan should clarify who provides the security for a network router or for a leased line to a remote site. Even though the security plan should be thorough, there will necessarily be vulnerabilities that are not considered. These vulnerabilities are not always the result of ignorance or naïveté; rather, they can arise from the addition of new equipment or data as the system evolves. They can also result from new situations, such as when a system is used in ways not anticipated by its designers. The security plan should detail the process to be followed when someone identifies a new vulnerability. In particular, instructions should explain how to integrate controls for that vulnerability into the existing security procedures.

IDOL

3. Requirements

The heart of the security plan is its set of security requirements: functional or performance demands placed on a system to ensure a desired level of security. The requirements are usually derived from organizational needs. Sometimes these needs include the need to conform to specific security requirements imposed from outside, such as by a government agency or a commercial standard. A constraint is an aspect of the security policy that constrains, circumscribes, or directs the implementation of the requirements. A control is an action, device, procedure, or technique that removes or reduces a vulnerability. To see the difference between requirements, constraints, and controls, consider the six "requirements" of the U.S. Department of Defense's TCSEC, introduced earlier. These six items are listed below in Table 12-1.

Table 12-1. The Six “Requirements” of the TCSEC.

Security policy	There must be an explicit and well-defined security policy enforced by the system.
Identification	Every subject must be uniquely and convincingly identified. Identification is necessary so that subject/object access can be checked.
Marking	Every object must be associated with a label that indicates its security level. The association must be done so that the label is available for comparison each time an access to the object is requested.
Accountability	The system must maintain complete, secure records of actions that affect security. Such actions include introducing new users to the system, assigning or changing the security level of a subject or an object, and denying access attempts.
Assurance	The computing system must contain mechanisms that enforce security, and it must be possible to evaluate the effectiveness of these mechanisms.
Continuous protection	The mechanisms that implement security must be protected against unauthorized change.

Given our definitions of requirement, constraint, and control, it is easy to see that the first "requirement" of the TCSEC is really a constraint: the security policy. The second and third "requirements" describe mechanisms for enforcing security, not descriptions of required behaviours. That is, the second and third "requirements" describe explicit implementations, not a general characteristic or property that the system must have. However, the fourth, fifth, and sixth TCSEC "requirements" are indeed true requirements. They state that the system must have certain characteristics, but they do not enforce a particular implementation. These distinctions are important because the requirements explain *what* should be accomplished, not *how*. That is, the requirements should always leave the implementation details to the designers, whenever possible. For example, rather than writing a requirement that certain data records should require passwords for access (an implementation decision), a security planner should state only that access to the data records should be restricted (and note to whom the access should be restricted). This more flexible requirement allows the designers to decide among several other access controls (such as access control lists) and to balance the security requirements with other system requirements, such as performance and reliability. Figure 12-1 illustrates how the different aspects of system analysis support the security planning process.

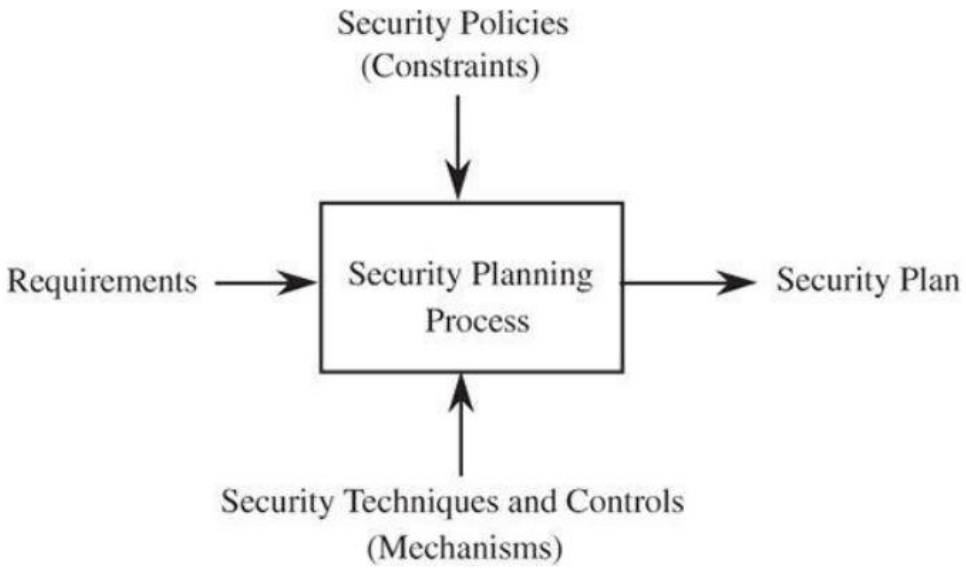


Figure 12-1. Inputs to the Security Plan.

As with the general software development process, the security planning process must allow customers or users to specify desired functions, independent of the implementation. The requirements should address all aspects of security: confidentiality, integrity, and availability. They should also be reviewed to make sure that they are of appropriate quality. In particular, we should make sure that the requirements have these characteristics:

- *Correctness:* Are the requirements understandable? Are they stated without error?
- *Consistency:* Are there any conflicting or ambiguous requirements?
- *Completeness:* Are all possible situations addressed by the requirements?
- *Realism:* Is it possible to implement what the requirements mandate?
- *Need:* Are the requirements unnecessarily restrictive?
- *Verifiability:* Can tests be written to demonstrate conclusively and objectively that the requirements have been met? Can the system or its functionality be measured in some way that will assess the degree to which the requirements are met?
- *Traceability:* Can each requirement be traced to the functions and data related to it so that changes in a requirement can lead to easy re-evaluation?

The requirements may then be constrained by budget, schedule, performance, policies, governmental regulations, and more. Given the requirements and constraints, the developers then choose appropriate controls.

4. Recommended Controls

The security requirements lay out the system's needs in terms of what should be protected. The security plan must also recommend what controls should be incorporated into the system to meet those requirements. Throughout this book you have seen many examples of controls, so we need not review them here. As we see later in this chapter, we can use risk analysis to create a map from vulnerabilities to controls. The mapping tells us how the system will meet the security requirements. That is, the recommended controls address implementation issues: how the system will be designed and developed to meet stated security requirements.

5. Responsibility for Implementation

A section of the security plan should identify which people are responsible for implementing the security requirements. This documentation assists those who must coordinate their individual responsibilities with those of other developers. At the same time, the plan makes explicit who is accountable should some requirement not be met or some vulnerability not be addressed. That is, the plan notes who is responsible for implementing controls when a new vulnerability is discovered or a new kind of asset is introduced. People building, using, and maintaining the system play many roles. Each role can take some responsibility for one or more aspects of security. Consider, for example, the groups listed here.

- *Personal computer users* may be responsible for the security of their own machines. Alternatively,

the security plan may designate one person or group to be coordinator of personal computer security.

- *Project leaders* may be responsible for the security of data and computations.
- *Managers* may be responsible for seeing that the people they supervise implement security measures.
- *Database administrators* may be responsible for the access to and integrity of data in their databases.
- *Information officers* may be responsible for overseeing the creation and use of data; these officers may also be responsible for retention and proper disposal of data.
- *Personnel staff members* may be responsible for security involving employees, for example, screening potential employees for trustworthiness and arranging security training programs.

6. Timetable

A comprehensive security plan cannot be executed instantly. The security plan includes a timetable that shows how and when the elements of the plan will be performed. These dates also give milestones so that management can track the progress of implementation. If the implementation is to be a phased development, the plan should also describe how the security requirements will be implemented over time. Even when overall development is not phased, it may be desirable to implement the security aspects of the system over time. For example, if the controls are expensive or complicated, they may be acquired and implemented gradually. Similarly, procedural controls may require staff training to ensure that everyone understands and accepts the reason for the control. The plan should specify the order in which the controls are to be implemented so that the most serious exposures are covered as soon as possible. A timetable also gives milestones by which to judge the progress of the security program. Furthermore, the plan must be extensible. Conditions will change: New equipment will be acquired, new degrees and modes of connectivity will be requested, and new threats will be identified. The plan must include a procedure for change and growth, so that the security aspects of changes are considered as a part of preparing for the change, not for adding security after the change has been made. The plan should also contain a schedule for periodic review. Even though there may have been no obvious, major growth, most organizations experience modest change every day. At some point the cumulative impact of the change is enough to require the plan to be modified.

7. Continuing Attention

Good intentions are not enough when it comes to security. We must not only take care in defining requirements and controls, but we must also find ways for evaluating a system's security to be sure that the system is as secure as we intend it to be. Thus, the security plan must call for reviewing the security situation periodically. As users, data, and equipment change, new exposures may develop. In addition, the current means of control may become obsolete or ineffective. The inventory of objects and the list of controls should periodically be scrutinized and updated, and risk analysis performed anew. The security plan should set times for these periodic reviews, based either on calendar time (such as, review the plan every nine months) or on the nature of system changes (such as, review the plan after every major system release).

12.2.2 Security Planning Team Members

Who performs the security analysis, recommends a security program, and writes the security plan? As with any such comprehensive task, these activities are likely to be performed by a committee that represents all the interests involved. The size of the committee depends on the size and complexity of the computing organization and the degree of its commitment to security. Organizational behaviour studies suggest that the optimum size for a working committee is between five and nine members. Sometimes a larger committee may serve as an oversight body to review and comment on the products of a smaller working committee. Alternatively, a large committee might designate subcommittees to address various sections of the plan. Security in operating systems and networks requires the cooperation of the systems administration staff. Program security measures can be understood and recommended by applications programmers. Physical security controls are implemented by those responsible for general physical security, both against human attacks and natural disasters. Finally, because controls affect system users, the plan should incorporate users' views, especially with regard to usability and the general desirability of controls. Thus, no matter how it is organized, a security planning team should represent each of the following groups:

- computer hardware group
- system administrators
- systems programmers

- applications programmers
- data entry personnel
- physical security personnel
- representative users

In some cases, a group can be adequately represented by someone who is consulted at appropriate times, rather than a committee member from each possible constituency being enlisted.

12.2.3 Assuring Commitment to a Security Plan

After the plan is written, it must be accepted and its recommendations carried out. Acceptance by the organization is key; a plan that has no organizational commitment is simply a plan that collects dust on the shelf. Commitment to the plan means that security functions will be implemented and security activities carried out. Three groups of people must contribute to making the plan a success.

- The planning team must be sensitive to the needs of each group affected by the plan.
- Those affected by the security recommendations must understand what the plan means for the way they will use the system and perform their business activities. In particular, they must see how what they do can affect other users and other systems.
- Management must be committed to using and enforcing the security aspects of the system.

Education and publicity can help people understand and accept a security plan. Acceptance involves not only the letter but also the spirit of the security controls. If people understand the need for recommended controls and accept them as sensible, they will use the controls properly and effectively. If people think the controls are bothersome, capricious, or counterproductive, they will work to avoid or subvert them. Management commitment is obtained through understanding. But this understanding is not just a function of what makes sense technologically; it also involves knowing the cause and the potential effects of lack of security. Managers must also weigh trade-offs in terms of convenience and cost. The plan must present a picture of how cost effective the controls are, especially when compared to potential losses if security is breached without the controls. Thus, proper presentation of the plan is essential, in terms that relate to management as well as technical concerns. Remember that some managers are not computing specialists. Instead, the system supports a manager who is an expert in some other business function, such as banking, medical technology, or sports. In such cases, the security plan must present security risks in language that the managers understand. Sometimes outside experts can bridge the gap between the managers' business and security. Management is often reticent to allocate funds for controls until the value of those controls is explained. The results of a risk analysis can help communicate the financial trade-offs and benefits of implementing controls. By describing vulnerabilities in financial terms and in the context of ordinary business activities (such as leaking data to a competitor or an outsider), security planners can help managers understand the need for controls.

The plans we have just discussed are part of normal business. They address how a business handles computer security needs. Similar plans might address how to increase sales or improve product quality, so these planning activities should be a natural part of management. Next, we turn to two particular kinds of business plans that address specific security problems: coping with and controlling activity during security incidents.

12.2.4 Business Continuity Plans

Small companies working on a low profit margin can literally be put out of business by a computer incident. Large, financially sound businesses can weather a modest incident that interrupts their use of computers for a while, although it is painful to them. But even rich companies do not want to spend money unnecessarily. The analysis is sometimes as simple as *no computers means no customers means no sales means no profit.* Government agencies, educational institutions, and non-profit organizations also have limited budgets, which they want to use to further their needs. They may not have a direct profit motive but being able to meet the needs of their customers, the public, students, and constituents partially determines how well they will fare in the future. All kinds of organizations must plan for ways to cope with emergency situations.

A business continuity plan documents how a business will continue to function during a computer security incident. An ordinary security plan covers computer security during normal times and deals with protecting against a wide range of vulnerabilities from the usual sources. A business continuity plan deals with situations having two characteristics:

- *catastrophic situations*, in which all or a major part of a computing capability is suddenly unavailable
- *long duration*, in which the outage is expected to last for so long that business will suffer

There are many situations in which a business continuity plan would be helpful. Here are some examples that typify what you might find in reading your daily newspaper:

- A fire destroys a company's entire network.
- A seemingly permanent failure of a critical software component renders the computing system unusable.
- A business must deal with the abrupt failure of its supplier of electricity, telecommunications, network access, or other critical service.
- A flood prevents the essential network support staff from getting to the operations centre.

As you can see, these examples are likely to recur, and each disables a vital function. You may also have noticed how often "the computer" is blamed for an inability to provide a service or product. For instance, the clerk in a shop is unable to use the cash register because "the computer is down." You may have a CD in your hand, plus exactly the cash to pay for it. But the clerk will not take your money and send you on your way. Often, computer service is restored shortly. But sometimes it is not. The key to coping with such disasters is advance planning and preparation, identifying activities that will keep a business viable when the computing technology is disabled. The steps in business continuity planning are these:

- Assess the business impact of a crisis.
- Develop a strategy to control impact.
- Develop and implement a plan for the strategy

Assess Business Impact

To assess the impact of a failure on your business, you begin by asking two key questions:

- What are the *essential assets*? What are the things that will prevent the business from doing business? Answers are typically of the form "the network," "the customer reservations database," or "the system controlling traffic lights."
- What could *disrupt use* of these assets? The vulnerability is more important than the threat agent. For example, whether destroyed by a fire or zapped in an electrical storm, the network is nevertheless down. Answers might be "failure," "corrupted," or "loss of power."

You probably will find only a handful of key assets when doing this analysis. Do not overlook people and the things they need for support, such as documentation and communications equipment. Another way to think about your assets is to ask yourself, "What is the minimum set of things or activities needed to keep business operational, at least to some degree?" If a manual system would compensate for a failed computer system, albeit inefficiently, you may want to consider building such a manual system as a potential critical asset.

Develop Strategy

The continuity strategy investigates how the key assets can be safeguarded. In some cases, a backup copy of data or redundant hardware or an alternative manual process is good enough. Sometimes, the most reasonable answer is reduced capacity. Ideally, you would like to continue business with no loss. But with catastrophic failures, usually only a portion of the business function can be preserved. In this case, you must develop a strategy appropriate for your business and customers. For instance, you can decide whether it is better to preserve half of function A and half of B, or most of A and none of B. You also must consider the time frame in which business is done. Some catastrophes last longer than others. For example, rebuilding after a fire is a long process and implies a long time in disaster mode. Your strategy may have several steps, each dependent on how long the business is disabled. Thus, you may take one action in response to a one-hour outage, and another if the outage might last a day or longer. Because you are planning in advance, you have the luxury of being able to think about possible circumstances and evaluate alternatives. The result of a strategy analysis is a selection of the best actions, organized by circumstances. The strategy can then be used as the basis for your business continuity plan.

Develop Plan

The business continuity plan specifies several important things:

- who is in charge when an incident occurs
- what to do

- who does it

The plan justifies making advance arrangements, such as acquiring redundant equipment, arranging for data backups, and stockpiling supplies, before the catastrophe. The plan also justifies advance training so that people know how they should react. In a catastrophe there will be confusion; you do not want to add confused people to the already severe problem. The person in charge declares the state of emergency and instructs people to follow the procedures documented in the plan. The person in charge also declares when the emergency is over and conditions can revert to normal. Thus, the business continuity planning addresses how to maintain some degree of critical business activity in spite of a catastrophe. Its focus is on keeping the business viable. It is based on the asset survey, which focuses on only a few critical assets and serious vulnerabilities that could threaten operation for a long or undetermined period of time. The focus of the business continuity plan is to keep the business going while someone else addresses the crisis. That is, the business continuity plan does not include calling the fire department or evacuating the building, important though those steps are. The focus of a business continuity plan is the *business* and how to keep it functioning to the degree possible in the situation. Handling the emergency is someone else's problem.

12.2.5 Incident Response Plans

An incident response plan tells the staff how to deal with a security incident. In contrast to the business continuity plan, the goal of incident response is handling the current security incident, without regard for the business issues. The security incident may at the same time be a business catastrophe, as addressed by the business continuity plan. But as a specific security event, it might be less than catastrophic but could be a serious breach of security, such as a hacker attack or a case of internal fraud. An incident could be a single event, a series of events, or an ongoing problem. An incident response plan should

- define what constitutes an *incident*
- identify who is responsible for *taking charge* of the situation
- describe the plan of *action*

The plan usually has three phases: advance planning, triage, and running the incident. A fourth phase, review, is useful after the situation abates so that this incident can lead to improvement for future incidents.

Advance Planning

As with all planning functions, advance planning works best because people can think logically, unhurried, and without pressure. What constitutes an incident may be vague. We cannot know the details of an incident in advance. Typical characteristics include harm or risk of harm to computer systems, data, or processing; initial uncertainty as to the extent of damage; and similar uncertainty as to the source or method of the incident. For example, you can see that the file is missing, or the home page has been defaced, but you do not know how or by whom or what other damage there may be. In organizations that have not done incident planning, chaos may develop at this point. Someone calls the network manager. Someone sends e-mail to the help desk. Someone calls the FBI, the CERT, the newspapers, or the fire department. People start to investigate on their own, without coordinating with the relevant staff in other departments, agencies, or businesses. And there is a lot of conversation, rumour, and misinformation: more heat than light. With an incident response plan in place, everybody is trained in advance to call the designated leader. There is an established list of people to call, in order, in case the first person is unavailable. The leader decides what to do next, and he or she begins by determining if this is a real incident or a false alarm. Indeed, natural events sometimes look like incidents, and the facts of the situation should be established first. If the leader decides this may be a real incident, he or she invokes the response team.

Response Team

The response team is the set of people charged with responding to the incident. The response team may include:

- *director*: person in charge of the incident, who decides what actions to take and when to terminate the response. The director is typically a management employee.
- *lead technician*: person who directs and coordinates the response. The lead technician decides where to focus attention, analyzes situation data, documents the incident and how it was handled, and calls for other technical people to assist with the analysis.
- *advisor(s)*: legal, human resources, or public relations staff members as appropriate.

In a small incident a single person can handle more than one of these roles. Nevertheless, it is important that a single person be in charge, a single person who directs the response work, a single point of contact for "insiders" (employees, users), and a single point of contact for "the public." To develop policy and identify a response team, you need to consider certain matters:

- *Legal issues:* An incident has legal ramifications. In some countries, computer intrusions are illegal, so law enforcement officials must be involved in the investigation. In other places, you have discretion in deciding whether to ask law enforcement to participate. In addition to criminal action, you may be able to bring a civil case. Both kinds of legal action have serious implications for the response. For example, evidence must be gathered and maintained in specific ways to be usable in court. Similarly, laws may limit what you can do against the alleged attacker: Cutting off a connection is probably acceptable but launching a retaliatory denial-of-service attack may not be.
- *Preserving evidence:* The most common reaction in an incident is to assume the cause was internal or accidental. For instance, you may surmise that the hardware has failed or that the software isn't working correctly. The staff may be directed to change the configuration, reload the software, reboot the system, or similarly attempt to resolve the problem by adjusting the software. Unfortunately, each of these acts can irreparably distort or destroy evidence. When dealing with a possible incident, do as little as possible before "dusting for fingerprints."
- *Records:* It may be difficult to remember what you have already done: Have you already reloaded a particular file? What steps got you to the prompt asking for the new DNS server's address? If you call in an outside forensic investigator or the police, you will need to tell exactly what you have already done.
- *Public relations:* In handling an incident your organization should speak with one voice. You risk sending confusing messages if too many people speak. It is especially important that only one person speak publicly if legal action may be taken. An unguarded comment may tip off the attacker or have a negative effect on the case. You can simply say that an incident occurred, tell briefly and generally what it was, and state that the incident is now under control and normal operation is resuming.

After the Incident Is Resolved

Eventually, the incident response team closes the case. At this point it will hold a review after the incident to consider two things:

- *Is any security control action to be taken*? Did an intruder compromise a system because security patches were not up-to-date; if so, should there be a procedure to ensure that patches are applied when they become available? Was access obtained because of a poorly chosen password; if so, should there be a campaign to educate users on how to strong passwords? If there were control failures, what should be done to prevent similar attacks in the future?
- *Did the incident response plan work?* Did everyone know whom to notify? Did the team have needed resources? Was the response fast enough? What should be done differently next time? The incident response plan ensures that incidents are handled promptly, efficiently, and with minimal harm.

12.3 Risk Analysis

Good, effective security planning includes a careful risk analysis. A risk is a potential problem that the system or its users may experience. We distinguish a risk from other project events by looking for three things:

- *A loss associated with an event*. The event must generate a negative effect: compromised security, lost time, diminished quality, lost money, lost control, lost understanding, and so on. This loss is called the risk impact.
- *The likelihood that the event will occur*. The probability of occurrence associated with each risk is measured from 0 (impossible) to 1 (certain). When the risk probability is 1, we say we have a problem.
- *The degree to which we can change the outcome*. We must determine what, if anything, we can do to avoid the impact or at least reduce its effects. Risk control involves a set of actions to reduce or eliminate the risk.

We usually want to weigh the pros and cons of different actions we can take to address each risk. To that end,

we can quantify the effects of a risk by multiplying the risk impact by the risk probability, yielding the risk exposure. For example, if the likelihood of virus attack is 0.3 and the cost to clean up the affected files is $10,000, then the risk exposure is $3,000. So, we can use a calculation like this one to decide that a virus checker is worth an investment of $100, since it will prevent a much larger potential loss. Clearly, risk probabilities can change over time, so it is important to track them and plan for events accordingly.
Risk is inevitable in life: Crossing the street is risky but that does not keep us from doing it. We can identify, limit, avoid, or transfer risk but we can seldom eliminate it. In general, we have three strategies for dealing with risk:

- *avoiding* the risk, by changing requirements for security or other system characteristics
- *transferring* the risk, by allocating the risk to other systems, people, organizations, or assets; or by buying insurance to cover any financial loss should the risk become a reality
- *assuming* the risk, by accepting it, controlling it with available resources, and preparing to deal with the loss if it occurs

Thus, costs are associated not only with the risk's potential impact but also with reducing it. Risk leverage is the difference in risk exposure divided by the cost of reducing the risk. In other words, risk leverage is

$$\frac{(\text{risk exposure before reduction}) - (\text{risk exposure after reduction})}{(\text{cost of risk reduction})}$$

If the leverage value of a proposed action is not high enough, then we look for alternative but less costly actions or more effective reduction techniques. Risk analysis is the process of examining a system and its operational context to determine possible exposures and the potential harm they can cause. Thus, the first step in a risk analysis is to identify and list all exposures in the computing system of interest. Then, for each exposure, we identify possible controls and their costs. The last step is a cost benefit analysis: Does it cost less to implement a control or to accept the expected cost of the loss? In the remainder of this section, we describe risk analysis, present examples of risk analysis methods, and discuss some of the drawbacks to performing risk analysis.

12.3.1 The Nature of Risk

In our everyday lives, we take risks. In crossing the road, eating oysters, or playing the lottery, we take the chance that our actions may result in some negative result such as being injured, getting sick, or losing money. Consciously or unconsciously, we weigh the benefits of taking the action with the possible losses that might result. Just because there is a risk to a certain act we do not necessarily avoid it; we may look both ways before crossing the street, but we do cross it. In building and using computing systems, we must take a more organized and careful approach to assessing our risks. Many of the systems we build and use can have a dramatic impact on life and health if they fail. For this reason, risk analysis is an essential part of security planning. We cannot guarantee that our systems will be risk free; that is why our security plans must address actions needed should an unexpected risk become a problem. And some risks are simply part of doing business; for example, as we have seen, we must plan for disaster recovery, even though we take many steps to avoid disasters in the first place. When we acknowledge that a significant problem cannot be prevented, we can use controls to reduce the seriousness of a threat. For example, you can back up files on your computer as a defence against the possible failure of a file storage device. But as our computing systems become more complex and more distributed, complete risk analysis becomes more difficult and time consuming and more essential.

12.3.2 Steps of a Risk Analysis

Risk analysis is performed in many different contexts; for example, environmental and health risks are analyzed for activities such as building dams, disposing of nuclear waste, or changing a manufacturing process. Risk analysis for security is adapted from more general management practices, placing special emphasis on the kinds of problems likely to arise from security issues. By following well-defined steps, we can analyze the security risks in a computing system. The basic steps of risk analysis are listed below:

- Identify assets.
- Determine vulnerabilities.
- Estimate likelihood of exploitation.

- Compute expected annual loss.
- Survey applicable controls and their costs.
- Project annual savings of control.

These steps are described in detail in the following sections:

Step 1: Identify Assets

Before we can identify vulnerabilities, we must first decide what we need to protect. Thus, the first step of a risk analysis is to identify the assets of the computing system. The assets can be considered in categories, as listed below:

- *hardware*: processors, boards, keyboards, monitors, terminals, microcomputers, workstations, tape drives, printers, disks, disk drives, cables, connections, communications controllers, and communications media
- *software*: source programs, object programs, purchased programs, in-house programs, utility programs, operating systems, systems programs (such as compilers), and maintenance diagnostic programs
- *data*: data used during execution, stored data on various media, printed data, archival data, update logs, and audit records
- *people:* skills needed to run the computing system or specific programs
- *documentation*: on programs, hardware, systems, administrative procedures, and the entire system
- *supplies*: paper, forms, laser cartridges, magnetic media, and printer fluid

It is essential to tailor this list to your own situation. No two organizations will have the same assets to protect, and something that is valuable in one organization may not be as valuable to another. For example, RAND Corporation's Vulnerability Assessment and Mitigation (VAM) methodology includes additional assets, such as:

- the enabling infrastructure
- the building or vehicle in which the system will reside
- the power, water, air, and other environmental conditions necessary for proper functioning
- human and social assets, such as policies, procedures, and training

The VAM methodology is a process supported by a tool to help people identify assets, vulnerabilities, and countermeasures. In a sense, the list of assets is an inventory of the system, including intangibles and human resource items. For security purposes, this inventory is more comprehensive than the traditional inventory of hardware and software often performed for configuration management or accounting purposes. The point is to identify all assets necessary for the system to be usable.

Step 2: Determine Vulnerabilities

The next step in risk analysis is to determine the vulnerabilities of these assets. This step requires imagination; we want to predict what damage might occur to the assets and from what sources. We can enhance our imaginative skills by developing a clear idea of the nature of vulnerabilities. This nature derives from the need to ensure the three basic goals of computer security: confidentiality, integrity, and availability. Thus, a vulnerability is any situation that could cause loss of confidentiality, integrity, and availability. We want to use an organized approach to considering situations that could cause these losses for a particular object. Software engineering offers us several techniques for investigating possible problems. Hazard analysis explores failures that may occur and faults that may cause them. These techniques have been used successfully in analyzing safety-critical systems. However, additional techniques are tailored specifically to security concerns. To organize the way we consider threats and assets, we can use a matrix such as the one shown in Table 12-2. One vulnerability can affect more than one asset or cause more than one type of loss. The table is a guide to stimulate thinking, but its format is not rigid.

Table 12-2. Assets and Security Properties.

Asset	Confidentiality	Integrity	Availability
Hardware			
Software			
Data			
People			
Documentation			
Supplies			

In thinking about the contents of each matrix entry, we can ask the following questions:

- What are the effects of unintentional errors? Consider typing the wrong command, entering the wrong data, using the wrong data item, discarding the wrong listing, and disposing of output insecurely.
- What are the effects of wilfully malicious insiders? Consider disgruntled employees, bribery, and curious browsers.
- What are the effects of outsiders? Consider network access, dial-in access, hackers, people walking through the building, and people sifting through the trash.
- What are the effects of natural and physical disasters? Consider fires, storms, floods, power outages, and component failures.

Table 12-3 is a version of the previous table with some of the entries filled in. It shows that certain general problems can affect the assets of a computing system. In a given installation, it is necessary to determine what can happen to specific hardware, software, data items, and other assets.

Table 12-3. Assets and Attacks.

IDOL

Asset	Secrecy	Integrity	Availability
Hardware		overloaded destroyed tampered with	failed stolen destroyed unavailable
Software	stolen copied pirated	impaired by Trojan horse modified tampered with	deleted misplaced usage expired
Data	disclosed accessed by outsider inferred	damaged – software error – hardware error – user error	deleted misplaced destroyed
People			quit retired terminated on vacation
Documentation			lost stolen destroyed
Supplies			lost stolen damaged

Alas, there is no simple checklist or easy procedure to list all vulnerabilities. Tools can help us conceive of vulnerabilities by providing a structured way to think. For example, RAND's VAM methodology suggests that assets have certain properties that make them vulnerable. The properties exist in three categories: aspects of the design or architecture, aspects of behaviour, and general attributes. Table 12-4 lists these properties in more detail. Notice that the properties apply to many kinds of systems and at various places within a given system. These attributes can be used to build a matrix, each of whose entries may suggest one or more vulnerabilities. Using that matrix for example, the design attribute *limits, finiteness* applied to a *cyber object*, a *software program* could lead you to suspect buffer overflow vulnerabilities, or *uniqueness* for a *hardware object* could signal a single point of failure. To use this methodology, you would work through the matrix, thinking of each contributing attribute on each asset class to derive the set of vulnerabilities.

Table 12-4. Attributes Contributing to Vulnerabilities.

Design/Architecture	Behavioral	General
• Singularity – Uniqueness – Centrality – Homogeneity • Separability • Logic/implementation errors; fallibility • Design sensitivity, fragility, limits, finiteness • Unrecoverability	• Behavioral sensitivity/fragility • Malevolence • Rigidity • Malleability • Gullibility, deceivability, naïveté • Complacency • Corruptibility, controllability	• Accessible, detectable, identifiable, transparent, interceptable • Hard to manage or control • Self-unawareness and unpredictability • Predictability

Step 3: Estimate Likelihood of Exploitation

The third step in conducting a risk analysis is determining how often each exposure is likely to be exploited. Likelihood of occurrence relates to the stringency of the existing controls and the likelihood that someone or something will evade the existing controls. There are several approaches to computing the probability that an event will occur: classical, frequency, and subjective. Each approach has its advantages and disadvantages, and we must choose the approach that best suits the situation (and its available information). In security, it is often not possible to directly evaluate an event's probability by using classical techniques. However, we can try to apply frequency probability by using observed data for a specific system. Local failure rates are fairly easy to record, and we can identify which failures resulted in security breaches or created new vulnerabilities. In particular, operating systems can track data on hardware failures, failed login attempts, numbers of accesses, and changes in the sizes of data files. Another alternative is to estimate the number of occurrences in a given time period. We can ask an analyst familiar with the system to approximate the number of times a described event occurred in the last year, for example. Although the count is not exact, the analyst's knowledge of the system and its usage may yield reasonable estimates. Of course, the two methods described depend on the fact that a system is already built and has been in use for some period of time. In many cases, and especially for proposed systems, the usage data are not available. In this case, we may ask an analyst to estimate likelihood by reviewing a table based on a similar system; this approach is incorporated in several formal security risk processes.

Step 4: Compute Expected Loss

By this time, we have gained an understanding of the assets we value, their possible vulnerabilities, and the likelihood that the vulnerabilities will be exploited. Next, we must determine the likely loss if the exploitation does indeed occur. As with likelihood of occurrence, this value is difficult to determine. Some costs, such as the cost to replace a hardware item, are easy to obtain. The cost to replace a piece of software can be approximated reasonably well from the initial cost to buy it (or specify, design, and write it). However, we must take care to include hidden costs in our calculations. For instance, there is a cost to others of not having a piece of hardware or software. Similarly, there are costs in restoring a system to its previous state, reinstalling software, or deriving a piece of information. These costs are substantially harder to measure. In addition, there may be hidden costs that involve legal fees if certain events take place. For example, some data require protection for legal reasons. Personal data, such as police records, tax information, census data, and medical information, are so sensitive that there are criminal penalties for releasing the data to unauthorized people. Other data are company confidential; their release may give competitors an edge on new products or on likely changes to the stock price. Some financial data, especially when they reflect an adverse event, could seriously affect public confidence in a bank, an insurance company, or a stock brokerage. It is difficult to determine the cost of releasing these data. If a computing system, a piece of software, or a key person is unavailable, causing a particular computing task to be delayed, there may be serious consequences. If a program that prints paychecks is delayed, employees' confidence in the company may be shaken, or some employees may face penalties from not being able to pay their own bills. If customers cannot make transactions because the computer is down, they may choose to take their business to a competitor. For some time-critical services involving human lives, such as a hospital's life-support systems or a space station's guidance systems, the costs of failure are infinitely high. Thus, we must analyze the ramifications of a computer security failure. The following questions can prompt us to think about issues of explicit and hidden cost related to security. The answers may not produce precise cost figures, but they will help identify the sources of various types of costs.

- What are the legal obligations for preserving the confidentiality or integrity of a given data item?
- What business requirements and agreements cover the situation? Does the organization have to pay a penalty if it cannot provide a service?
- Could release of a data item cause harm to a person or organization? Would there be the possibility of legal action if harm were done?
- Could unauthorized access to a data item cause the loss of future business opportunity? Might it give a competitor an unfair advantage? What would be the estimated loss in revenue?
- What is the psychological effect of lack of computer service? Embarrassment? Loss of credibility? Loss of business? How many customers would be affected? What is their value as customers?
- What is the value of access to data or programs? Could this computation be deferred? Could this computation be performed elsewhere? How much would it cost to have a third party do the computing elsewhere?

- What is the value to someone else of having access to data or programs? How much would a competitor be willing to pay for access?
- What other problems would arise from loss of data? Could the data be replaced or reconstructed? With what amount of work?

These are not easy costs to evaluate. Nevertheless, they are needed to develop a thorough understanding of the risks. Furthermore, the vulnerabilities in computer security are often considerably higher than managers expect. Realistic estimates of potential harm can raise concern and suggest places in which attention to security is especially needed.

Step 5: Survey and Select New Controls

By this point in our risk analysis, we understand the system's vulnerabilities and the likelihood of exploitation. We turn next to an analysis of the controls to see which ones address the risks we have identified. We want to match each vulnerability with at least one appropriate security technique. Once we do that, we can use our expected loss estimates to help us decide which controls, alone or in concert, are the most cost effective for a given situation.

Choosing Controls

In this analysis controls can overlap, as for example, when a human guard and a locked door both protect against unauthorized access. Neither of these is redundant, because the human guard can handle exceptional situations (for example, when a legitimate user loses a key), but the lock prevents access if the guard is distracted. Also, one control may cover multiple vulnerabilities, so encrypting a set of data may protect both confidentiality and integrity. Controls have positive and negative effects: Encryption, for example, protects confidentiality, but it also takes time and introduces key management issues. Thus, when selecting controls, you have to consider the full impact. Controls are not perfect. They can fail: Guards can be bribed or fall asleep, encryption can be broken, and access control devices can malfunction. Some controls are stronger than others. For example, a physical device is generally stronger than a written policy (policies are nevertheless useful).

Which Controls Are Best?

Typically, there is no single best set of controls. One control is stronger, another is more usable, another prevents harm instead of detecting it afterwards, and still another protects against several types of vulnerabilities. As you have inferred, risk analysis involves building a multidimensional array: assets, vulnerabilities, likelihoods, controls. Mapping controls to vulnerabilities may involve using graph theory to select a minimal set of controls that address all vulnerabilities. The advantage of careful, systematic documentation of all these data is that each choice can be analyzed, and the side effects of changes are apparent. If this process sounds difficult, it is, but it need not be overwhelming. Listing all assets is less important than listing the top few, probably five to ten. Postulating all vulnerabilities is less important than recognizing several classes of harm and representative causes. With a manageable number of assets and vulnerabilities, determining controls (some of which may already be in place) need not be extensive, as long as some control covers each major vulnerability.

Step 6: Project Savings

By this point in our risk analysis, we have identified controls that address each vulnerability in our list. The next step is to determine whether the costs outweigh the benefits of preventing or mitigating the risks. Recall that we multiply the risk probability by the risk impact to determine the risk exposure. The risk impact is the loss that we might experience if the risk were to turn into a real problem. There are techniques to help us determine the risk exposure. The effective cost of a given control is the actual cost of the control (such as purchase price, installation costs, and training costs) minus any expected loss from using the control (such as administrative or maintenance costs). Thus, the true cost of a control may be positive if the control is expensive to administer or introduces new risk in another area of the system. Or the cost can even be negative if the reduction in risk is greater than the cost of the control. For example, suppose a department has determined that some users have gained unauthorized access to the computing system. It is feared that the intruders might intercept or even modify sensitive data on the system. One approach to addressing this problem is to install a more secure data access control program. Even though the cost of the access control software is high, its cost is easily justified when compared to its value. Because the entire cost of the package is charged in the first year, even greater benefits are expected for subsequent years.

12.3.3 Arguments for and against Risk Analysis

Risk analysis is a well-known planning tool, used often by auditors, accountants, and managers. In many situations, such as obtaining approval for new drugs, new power plants, and new medical devices, a risk analysis is required by law in many countries. There are many good reasons to perform a risk analysis in preparation for creating a security plan.

- *Improve awareness:* Discussing issues of security can raise the general level of interest and concern among developers and users. Especially when the user population has little expertise in computing, the risk analysis can educate users about the role security plays in protecting functions and data that are essential to user operations and products.
- *Relate security mission to management objectives:* Security is often perceived as a financial drain for no gain. Management does not always see that security helps balance harm and control costs.
- *Identify assets, vulnerabilities, and controls:* Some organizations are unaware of their computing assets, their value to the organization, and the vulnerabilities associated with those assets. A systematic analysis produces a comprehensive list of assets, valuations, and risks.
- *Improve basis for decisions:* A security manager can present an argument such as "I think we need a firewall here" or "I think we should use token-based authentication instead of passwords." Risk analysis augments the manager's judgment as a basis for the decision.
- *Justify expenditures for security:* Some security mechanisms appear to be very expensive and without obvious benefit. A risk analysis can help identify instances where it is worth the expense to implement a major security mechanism. Justification is often derived from examining the much larger risks of *not* spending for security.

However, despite the advantages of risk analysis, there are several arguments against using it to support decision making.

- *False sense of precision and confidence:* The heart of risk analysis is the use of empirical data to generate estimates of risk impact, risk probability, and risk exposure. The danger is that these numbers will give us a false sense of precision, thereby giving rise to an undeserved confidence in the numbers. However, in many cases the numbers themselves are much less important than their relative sizes. Whether an expected loss is $100,000 or $150,000 is relatively unimportant. It is much more significant that the expected loss is far above the $10,000 or $20,000 budget allocated for implementing a particular control. Moreover, anytime a risk analysis generates a large potential loss, the system deserves further scrutiny to see if the root cause of the risk can be addressed.
- *Hard to perform:* Enumerating assets, vulnerabilities, and controls requires creative thinking. Assessing loss frequencies and impact can be difficult and subjective. A large risk analysis will have many things to consider. Risk analysis can be restricted to certain assets or vulnerabilities, however.
- *Immutability:* It is typical on many software projects to view processes like risk analysis as an irritating fact of life a step to be taken in a hurry so that the developers can get on with the more interesting jobs related to designing, building, and testing the system. For this reason, risk analyses, like contingency plans and five-year plans, have a tendency to be filed and promptly forgotten. But if an organization takes security seriously, it will view the risk analysis as a living document, updating it at least annually or in conjunction with major system upgrades.
- *Lack of accuracy:* Risk analysis is not always accurate, for many reasons. First, we may not be able to calculate the risk probability with any accuracy, especially when we have no past history of similar situations. Second, even if we know the likelihood, we cannot always estimate the risk impact very well. The risk management literature is replete with papers about describing the scenario, showing that presenting the same situation in two different ways to two equivalent groups of people can yield two radically different estimates of impact. And third, we may not be able to anticipate all the possible risks. For example, bridge builders did not know about the risks introduced by torque from high winds until the Tacoma Narrows Bridge twisted in the wind and collapsed. After studying the colossal failure of this bridge and discovering the cause, engineers made mandatory the inclusion of torque in their simulation parameters. Similarly, we may not know enough about software, security, or the context in which the system is to be used, so there may be gaps in our risk analysis that cause it to be inaccurate.

This lack of accuracy is often cited as a deficiency of risk analysis. But this lack is a red herring. Risk

DRAFT

analysis is useful as a planning tool, to compare and contrast options. We may not be able to predict events accurately, but we can use risk analysis to weigh the tradeoffs between one action and another. When risk analysis is used in security planning, it highlights which security expenditures are likely to be most cost effective. This investigative basis is important for choosing among controls when money available for security is limited. And our risk analysis should improve as we build more systems, evaluate their security, and have a larger experience base from which to draw our estimates. A risk analysis has many advantages as part of security plan or as a tool for less formal security decision making. It ranges from very subjective and imprecise to highly quantitative. It is useful for generating and documenting thoughts about likely threats and possible countermeasures. Finally, it supports rational decision making about security controls.

12.4 Summary

- The administration of security draws on skills slightly different from the technical skills. The security administrator must understand not just security assets, threats, vulnerabilities, and controls, but management and implementation. In this chapter we examined how security is administered.
- First, security planning is a process that drives the rest of security administration. A security plan is a structure that allows things to happen in a studied, organized manner. General security plans explain how the organization will match threats to controls and to assets. Business continuity plans focus on the single issue of maintaining some ability to do business. Incident response plans cover how to keep a security event, such as a breach or attack, from running out of control. All plans offer the advantage that you can think about a situation in advance, with a clear mind, when you can weigh options easily.
- Risk assessment is a technique supporting security planning. In a risk assessment, you list vulnerabilities and controls, and then balance the cost of each control against the potential harm it can block. Risk assessments let you calculate the savings of security measures, instead of their costs, as is more frequently the case. Not all risk can be blocked. With a thorough risk assessment, you can know what risks you choose to accept.

12.7 Review Questions

a) Explain the contents of a security plan.
b) Explain the six requirements of TCSEC.
c) Explain the characteristics for the requirements of security plan.
d) Write a short note on Business Continuity Plan.
e) Write a short note on Incident Response Plans.
f) What is Risk Analysis? Explain its nature.
g) Explain the steps in risk analysis.
h) Why should you perform risk analysis?
i) What are the disadvantages of performing risk analysis?

12.8 Bibliography, References and Further Reading

- *Security in Computing by* C. P. Pfleeger, and S. L. Pfleeger, Pearson Education.
- *Computer Security: Art and Science by* Matt Bishop, Pearson Education.
- *Cryptography And Network Security: Principles and practice by* Stallings
- *Network Security by* Kaufman, Perlman, Speciner
- *Network Security : A Beginner's Guide by* Eric Maiwald, TMH

- *Java Network Security by* Macro Pistoia, Pearson Education
- *Principles of information security by* Whitman, Mattord, Thomson

IDOL

Chapter 13

Security Policies and Physical Security

IDOL

13.0 Objectives

In this chapter, we understand how to establish a framework for security needs and understand the impact that computing environment has on the physical security.

13.1 Introduction

Security is a combination of technical, administrative, and physical controls, as we first pointed out in the earlier chapters. So far, we have considered technical controls almost exclusively. But stop and think for a moment: What good is a firewall if there is no power to run it? How effective is a public key infrastructure if

someone can walk off with the certificate server? And why have elaborate access control mechanisms if your employee mails a sensitive document to a competitor? The administrative and physical controls may be less glamorous than the technical ones, but they are surely as important. In this and the next chapter we complete our study of security controls by considering administrative and physical aspects. We look at four related areas:

- *Planning:* What advance preparation and study lets us know that our implementation meets our security needs for today and tomorrow?
- *Risk analysis:* How do we weigh the benefits of controls against their costs, and how do we justify any controls?
- Policy: How do we establish a framework to see that our computer security needs continue to be met?
- *Physical control:* What aspects of the computing environment have an impact on security?

These four areas are just as important to achieving security as are the latest firewall or coding practice.

13.2 Organizational Security Policies

A key element of any organization's security planning is an effective security policy. A security policy must answer three questions: *who* can access *which resources* in *what manner*? A security policy is a high-level management document to inform all users of the goals of and constraints on using a system. A policy document is written in broad enough terms that it does not change frequently. The information security policy is the foundation upon which all protection efforts are built. It should be a visible representation of priorities of the entire organization, definitively stating underlying assumptions that drive security activities. The policy should articulate senior management's decisions regarding security as well as asserting management's commitment to security. To be effective, the policy must be understood by everyone as the product of a directive from an authoritative and influential person at the top of the organization. People sometimes issue other documents, called procedures or guidelines, to define how the policy translates into specific actions and controls. In this section, we examine how to write a useful and effective security policy.

13.2.1 Purpose

Security policies are used for several purposes, including the following:

- recognizing sensitive information assets
- clarifying security responsibilities
- promoting awareness for existing employees
- guiding new employees

13.2.2 Audience

A security policy addresses several different audiences with different expectations. That is, each group – users, owners, and beneficiaries use the security policy in important but different ways.

Users

Users legitimately expect a certain degree of confidentiality, integrity, and continuous availability in the computing resources provided to them. Although the degree varies with the situation, a security policy should reaffirm a commitment to this requirement for service. Users also need to know and appreciate what is considered acceptable use of their computers, data, and programs. For users, a security policy should define acceptable use.

Owners

Each piece of computing equipment is owned by someone, and the owner may not be a system user. An owner provides the equipment to users for a purpose, such as to further education, support commerce, or enhance productivity. A security policy should also reflect the expectations and needs of owners.

Beneficiaries

A business has paying customers or clients; they are beneficiaries of the products and services offered by that business. At the same time, the general public may benefit in several ways: as a source of employment or by provision of infrastructure. For example, the government has customers: the citizens of its country, and "guests" who have visas enabling entry for various purposes and times. A university's customers include its students and faculty; other beneficiaries include the immediate community (which can take advantage of lectures and concerts on campus) and often the world population (enriched by the results of research and service). To varying degrees, these beneficiaries depend, directly or indirectly, on the existence of or access to computers, their data and programs, and their computational power. For this set of beneficiaries, continuity and integrity of computing are very important. In addition, beneficiaries value confidentiality and correctness of the data involved. Thus, the interests of beneficiaries of a system must be reflected in the system's security policy.

Balance Among All Parties

A security policy must relate to the needs of users, owners, and beneficiaries. Unfortunately, the needs of these groups may conflict. A beneficiary might require immediate access to data, but owners or users might not want to bear the expense or inconvenience of providing access at all hours. Continuous availability may be a goal for users, but that goal is inconsistent with a need to perform preventive or emergency maintenance. Thus, the security policy must balance the priorities of all affected communities.

13.2.3 Contents

A security policy must identify its audiences: the beneficiaries, users, and owners. The policy should describe the nature of each audience and their security goals. Several other sections are required, including the purpose of the computing system, the resources needing protection, and the nature of the protection to be supplied. We discuss each one in turn.

Purpose

The policy should state the purpose of the organization's security functions, reflecting the requirements of beneficiaries, users, and owners. For example, the policy may state that the system will "protect customers' confidentiality or preserve a trust relationship," "ensure continual usability," or "maintain profitability." There are typically three to five goals, such as:

- Promote efficient business operation.
- Facilitate sharing of information throughout the organization.
- Safeguard business and personal information.
- Ensure that accurate information is available to support business processes.
- Ensure a safe and productive place to work.
- Comply with applicable laws and regulations.

The security goals should be related to the overall goal or nature of the organization. It is important that the system's purpose be stated clearly and completely because subsequent sections of the policy will relate back to these goals, making the policy a goal-driven product.

Protected Resources

A risk analysis will have identified the assets that are to be protected. These assets should be listed in the policy, in the sense that the policy lays out which items it addresses. For example, will the policy apply to all computers or only to those on the network? Will it apply to all data or only to client or management data? Will security be provided to all programs or only the ones that interact with customers? If the degree of protection varies from one service, product, or data type to another, the policy should state the differences. For example, data that uniquely identify clients may be protected more carefully than the names of cities in which clients reside.

Nature of the Protection

The asset list tells us *what* should be protected. The policy should also indicate *who* should have access to the protected items. It may also indicate *how* that access will be ensured and *how* unauthorized people will be denied access. All the mechanisms described in this book are at your disposal in deciding which controls should protect which objects. In particular, the security policy should state what

degree of protection should be provided to which kinds of resources.

13.2.4 Characteristics of a Good Security Plan

If a security policy is written poorly, it cannot guide the developers and users in providing appropriate security mechanisms to protect important assets. Certain characteristics make a security policy a good one.

Coverage

A security policy must be comprehensive: It must either apply to or explicitly exclude all possible situations. Furthermore, a security policy may not be updated as each new situation arises, so it must be general enough to apply naturally to new cases that occur as the system
is used in unusual or unexpected ways.

Durability

A security policy must grow and adapt well. In large measure, it will survive the system's growth and expansion without change. If written in a flexible way, the existing policy will be applicable to new situations. However, there are times when the policy must change (such as when government regulations mandate new security constraints), so the policy must be changeable when it needs to be. An important key to durability is keeping the policy free from ties to specific data or protection mechanisms that almost certainly will change. It is preferable to describe assets needing protection in terms of their function and characteristics, rather than in terms of specific implementation. Better still, we can separate the elements of the policy, having one policy statement for student grades and another for customers' proprietary data. Similarly, we may want to define one policy that applies to preserving the confidentiality of relationships, and another protecting the use of the system through strong authentication.

Realism

The policy must be realistic. That is, it must be possible to implement the stated security requirements with existing technology. Moreover, the implementation must be beneficial in terms of time, cost, and convenience; the policy should not recommend a control that works but prevents the system or its users from performing their activities and functions. It is important to make economically worthwhile investments in security, just as for any other careful business investment.

Usefulness

An obscure or incomplete security policy will not be implemented properly, if at all. The policy must be written in language that can be read, understood, and followed by anyone who must implement it or is affected by it. For this reason, the policy should be succinct, clear, and direct.

13.2.5 Examples

To understand the nature of security policies, we study a few examples to illustrate some of the points just presented.

Data Sensitivity Policy

Our first example is from an organization that decided to classify all its data resources into four levels, based on how severe might be the effect if a resource were damaged. These levels are sensitive, personal or protected, company confidential and open. Then, the required protection was based on the resource's level. Finally, the organization analyzed its threats, their possible severities, and countermeasures, and their effectiveness, within each of the four levels. Although the phrases describing the degree of damage are open to interpretation, the intent of these levels is clear: All information assets are to be classified as sensitive, personal, confidential, or open, and protection requirements for these four types are detailed in the remainder of the organization's policy document.

Government Agency IT Security Policy

The U.S. Department of Energy (DOE), like many government units, has established its own security policy. The following excerpt is from the policy on protecting classified material, although the form is appropriate

for many unclassified uses as well. It is the policy of DOE that classified information and classified ADP [automatic data processing] systems shall be protected from unauthorized access (including the enforcement of need-to-know protections), alteration, disclosure, destruction, penetration, denial of service, subversion of security measures, or improper use as a result of espionage, criminal, fraudulent, negligent, abusive, or other improper actions. The DOE shall use all reasonable measures to protect ADP systems that process, store, transfer, or provide access to classified information, to include but not limited to the following: physical security, personnel security, telecommunications security, administrative security, and hardware and software security measures. This order establishes this policy and defines responsibilities for the development, implementation, and periodic evaluation of the DOE program. The policy then continues for several more pages to list specific responsibilities for specific people. The cited paragraph is comprehensive, covering practically every possible source (espionage, crime, fraud, etc.) of practically every possible harm (unauthorized access, alteration, destruction, etc.), and practically every possible kind of control (physical, personnel, etc.).
The generality of the header paragraph is complemented by subsequent paragraphs giving specific responsibilities:

- "Each data owner shall determine and declare the required protection level of information . . ."
- "Each security officer shall . . . perform a risk assessment to identify and document specific . . . assets, . . . threats, . . . and vulnerability . . ."
- "Each manager shall...establish procedures to ensure that systems are continuously monitored...to detect security infractions . . ." and so on.

Internet Security Policy
The Internet does not have a governing security policy per se, because it is a federation of users. Nevertheless, the Internet Society drafted a security policy for its members. The policy contains the following interesting portions.

- Users are individually responsible for understanding and respecting the security policies of the systems (computers and networks) they are using. Users are individually accountable for their own behaviour.
- Users have a responsibility to employ available security mechanisms and procedures for protecting their own data. They also have a responsibility for assisting in the protection of the systems they use.
- Computer and network service providers are responsible for maintaining the security of the systems they operate. They are further responsible for notifying users of their security policies and any changes to these policies.
- Vendors and system developers are responsible for providing systems which are sound and which embody adequate security controls.
- Users, service providers, and hardware and software vendors are responsible for cooperating to provide security.
- Technical improvements in Internet security protocols should be sought on a continuing basis. At the same time, personnel developing new protocols, hardware or software for the Internet are expected to include security considerations as part of the design and development process.

These statements clearly state to whom they apply and for what each party is responsible.

13.3 Physical Security

Much of this book has focused on technical issues in security and their technical solutions: firewalls, encryption techniques, and more. But many threats to security involve human or natural disasters, events that should also be addressed in the security plan. For this reason, in this section we consider how to cope with the nontechnical things that can go wrong. There are two pieces to the process of dealing with nontechnical problems: preventing things that can be prevented and recovering from the things that cannot be prevented. Physical security is the term used to describe protection needed outside the computer system. Typical physical security controls include guards, locks, and fences to deter direct attacks. In addition, there are other kinds of protection against less direct disasters, such as floods and power outages; these, too, are part of physical security. As we will see, many physical security measures can be provided simply by good common sense, a characteristic that Mark Twain noted "is a most uncommon virtue."

IDOL

13.3.1 Natural Disasters

Computers are subject to the same natural disasters that can occur to homes, stores, and automobiles. They can be flooded, burned, melted, hit by falling objects, and destroyed by earthquakes, storms, and tornadoes. Additionally, computers are sensitive to their operating environment, so excessive heat or inadequate power is also a threat. It is impossible to prevent natural disasters, but through careful planning it is possible to reduce the damage they inflict. Some measures can be taken to reduce their impact. Because many of these perils cannot be prevented or predicted, controls focus on limiting possible damage and recovering quickly from a disaster. Issues to be considered include the need for offsite backups, the cost of replacing equipment, the speed with which equipment can be replaced, the need for available computing power, and the cost or difficulty of replacing data and programs.

Flood

Water from a natural flood comes from ground level, rising gradually, and bringing with it mud and debris. Often, there is time for an orderly shutdown of the computing system; at worst, the organization loses some of the processing in progress. At other times, such as when a dam breaks, a water pipe bursts, or the roof collapses in a storm, a sudden flood can overwhelm the system and its users before anything can be saved. Water can come from above, below, or the side. The machinery may be destroyed or damaged by mud and water, but most computing systems are insured and replaceable by the manufacturer. Managers of unique or irreplaceable equipment who recognize the added risk sometimes purchase or lease duplicate redundant hardware systems to ensure against disruption of service. Even when the hardware can be replaced, we must be concerned about the stored data and programs. The system administrator may choose to label storage media in a way that makes it easy to identify the most important data. For example, green, yellow, and red labels may show which disks are the most sensitive, so that all red disks are moved from the data centre during a storm. Similarly, large plastic bags and waterproof tape can be kept near important equipment and media; they are used to protect the hardware and storage media in case of a burst pipe or other sudden flood. The real issue is protecting data and preserving the ability to compute. The only way to ensure the safety of data is to store backup copies in one or more safe locations.

Fire

Fire is more serious than water; often there is not as much time to react, and human lives are more likely to be in immediate danger. To ensure that system personnel can react quickly, every user and manager should have a plan for shutting down the system in an orderly manner. Such a process takes only a few minutes but can make recovery much easier. This plan should include individual responsibilities for all people: some to halt the system, others to protect crucial media, others to close doors on media cabinets. Provision should be made for secondary responsibilities, so that onsite staff can perform duties for those who are not in the office. Water is traditionally used to put out fires, but it is not a good idea for use in computer rooms. In fact, more destruction can be the result of sprinklers than of the fires themselves. A fire sensor usually activates many sprinklers, dousing an entire room, even when the fire is merely some ignited paper in a wastebasket and of no threat to the computing system. Many computing centres use carbon dioxide extinguishers or an automatic system that sprays a gas such as Halon to smother a fire but leave no residue. Unfortunately, these gas systems work by displacing the oxygen in the room, choking the fire but leaving humans unable to breathe. Consequently, when these protection devices are activated, humans must leave, disabling efforts to protect media. The best defense for situations like these is careful placement of the computing facility. A windowless location with fire-resistant access doors and non-flammable full-height walls can prevent some fires from spreading from adjacent areas to the computing room. With a fire and smoke-resistant facility, personnel merely shut down the system and leave, perhaps carrying out the most important media. Fire prevention is quite effective, especially because most computer goods are not especially flammable. Advance planning, reinforced with simulation drills, can help make good use of the small amount of time available before evacuation is necessary.

Other Natural Disasters

Computers are subject to storms, earthquakes, volcanoes, and similar events. Although not natural disasters, building collapse, explosion, and damage from falling objects can be considered in the same category. These kinds of catastrophes are difficult to predict or estimate. But we know these catastrophes will occur. Security managers cope with them in several ways:

- developing contingency plans so that people know how to react in emergencies and business can continue
- insuring physical assets, computers, buildings, devices, supplies against harm
- preserving sensitive data by maintaining copies in physically separated locations

13.3.2 Power Loss

Computers need their food, electricity and they require a constant, pure supply of it. With a direct power loss, all computation ceases immediately. Because of possible damage to media by sudden loss of power, many disk drives monitor the power level and quickly retract the recording head if power fails. For certain time-critical applications, loss of service from the system is intolerable; in these cases, alternative complete power supplies must be instantly available.

Uninterruptible Power Supply

One protection against power loss is an uninterruptible power supply. This device stores energy during normal operation so that it can return the backup energy if power fails. One form of uninterruptible power supply uses batteries that are continually charged when the power is on but which then provide power when electricity fails. However, size, heat, flammability, and low output can be problems with batteries. Some uninterruptible power supplies use massive wheels that are kept in continuous motion when electricity is available. When the power fails, the inertia in the wheels operates generators to produce more power. Size and limited duration of energy output are problems with this variety of power supply. Both forms of power supplies are intended to provide power for a limited time, just long enough to allow the current state of the computation to be saved so that no computation is lost.

13.3.3 Surge Suppressor

Another problem with power is its "cleanness." Although most people are unaware of it, a variation of 10 percent from the stated voltage of a line is considered acceptable, and some power lines vary even more. A particular power line may always be 10 percent high or low. In many places, lights dim momentarily when a large appliance, such as an air conditioner, begins operation. When a large motor starts, it draws an exceptionally large amount of current, which reduces the flow to other devices on the line. When a motor stops, the sudden termination of draw can send a temporary surge along the line. Similarly, lightning strikes may send a momentary large pulse. Thus, instead of being constant, the power delivered along any electric line shows many brief fluctuations, called drops, spikes, and surges. A drop is a momentary reduction in voltage, and a spike or surge is a rise. For computing equipment, a drop is less serious than a surge. Most electrical equipment is tolerant of rather large fluctuations of current. These variations can be destructive to sensitive electronic equipment, however. Simple devices called "surge suppressors" filter spikes from an electric line, blocking fluctuations that would affect computers. These devices cost from $20 to $100; they should be installed on every computer, printer, or other connected component. More sensitive models are typically used on larger systems. As mentioned previously, a lightning strike can send a surge through a power line. To increase protection, personal computer users usually unplug their machines when they are not in use, as well as during electrical storms. Another possible source of destruction is lightning striking a telephone line. Because the power surge can travel along the phone line and into the computer or peripherals, the phone line should be disconnected from the modem during storms. These simple measures may save much work as well as valuable equipment.

13.3.4 Human Vandals

Because computers and their media are sensitive to a variety of disruptions, a vandal can destroy hardware, software, and data. Human attackers may be disgruntled employees, bored operators, saboteurs, people seeking excitement, or unwitting bumblers. If physical access is easy to obtain, crude attacks using axes or bricks can be very effective. Physical attacks by unskilled vandals are often easy to prevent; a guard can stop someone approaching a computer installation with a threatening or dangerous object. When physical access is difficult, more subtle attacks can be tried, resulting in quite serious damage. People with only some

sophisticated knowledge of a system can short-circuit a computer with a car key or disable a disk drive with a paper clip. These items are not likely to attract attention until the attack is completed.

Unauthorized Access and Use

Films and newspaper reports exaggerate the ease of gaining access to a computing system. Still, as distributed computing systems become more prevalent, protecting the system from outside access becomes more difficult and more important. Interception is a form of unauthorized access; the attacker intercepts data and either breaks confidentiality or prevents the data from being read or used by others. In this context, interception is a passive attack. But we must also be concerned about active interception, in the sense that the attacker can change or insert data before allowing it to continue to its destination.

Theft

It is hard to steal a large mainframe computer. Not only is carrying it away difficult but finding a willing buyer and arranging installation and maintenance also require special assistance. However, printed reports, tapes, or disks can be carried easily. If done well, the loss may not be detected for some time. Personal computers, laptops, and personal digital assistants (PDAs, such as Palms or Blackberries) are designed to be small and portable. Diskettes and tape backup cartridges are easily carried in a shirt pocket or briefcase. Computers and media that are easy to carry are also easy to conceal. We can take one of three approaches to preventing theft: preventing access, preventing portability, or detecting exit.

Preventing Access

The surest way to prevent theft is to keep the thief away from the equipment. However, thieves can be either insiders or outsiders. Therefore, access control devices are needed both to prevent access by unauthorized individuals and to record access by those authorized. A record of accesses can help identify who committed a theft. The oldest access control is a guard, not in the database management system sense but rather in the sense of a human being stationed at the door to control access to a room or to equipment. Guards offer traditional protection; their role is well understood, and the protection they offer is adequate in many situations. However, guards must be on duty continuously to be effective; providing breaks implies at least four guards for a 24-hour operation, with extras for vacation and illness. A guard must personally recognize someone or recognize an access token, such as a badge. People can lose or forget badges; terminated employees and forged badges are also problems. Unless the guard makes a record of everyone who has entered a facility, there is no way to know who (employee or visitor) has had access in case a problem is discovered.

The second oldest access control is a lock. This device is even easier, cheaper, and simpler to manage than a guard. However, it too provides no record of who has had access, and difficulties arise when keys are lost or duplicated. At computer facilities, it is inconvenient to fumble for a key when your hands are filled with tapes or disks, which might be ruined if dropped. There is also the possibility of piggybacking: a person walks through the door that someone else has just unlocked. Still, guards and locks provide simple, effective security for access to facilities such as computer rooms.

More exotic access control devices employ cards with radio transmitters, magnetic stripe cards (similar to 24-hour bank cards), and smart cards with chips containing electronic circuitry that makes them difficult to duplicate. Because each of these devices interfaces with a computer, it is easy for the computer to capture identity information, generating a list of who entered and left the facility, when, and by which routes. Some of these devices operate by proximity, so that a person can carry the device in a pocket or clipped to a collar; the person obtains easy access even when hands are full. Because these devices are computer controlled, it is easy to invalidate an access authority when someone quits or reports the access token lost or stolen. The nature of the application or service determines how strict the access control needs to be. Working in concert with computer-based authentication techniques, the access controls can be part of defense in depth using multiple mechanisms to provide security.

Preventing Portability

Portability is a mixed blessing. We can now carry around in our pockets devices that provide as much computing power as mainframes did twenty years ago. Portability is in fact a necessity in devices such as PDAs and mobile phones. And we do not want to permanently affix our personal computers to our desks, in case they need to be removed for repair or replacement. Thus, we need to find ways to enable portability without promoting theft. One antitheft device is a pad connected to cable, similar to those used to secure bicycles. The pad is glued to the desktop with extremely strong adhesive. The cables loop around the equipment and are locked in place. Releasing the lock permits the equipment to be moved. An alternative is

to couple the base of the equipment to a secure pad, in much the same way that televisions are locked in place in hotel rooms. Yet a third possibility is a large, lockable cabinet in which the personal computer and its peripherals are kept when they are not in use. Some people argue that cables, pads, and cabinets are unsightly and, worse, they make the equipment inconvenient to use. Another alternative is to use movement-activated alarm devices when the equipment is not in use. Small alarms are available that can be locked to a laptop or PDA. When movement is detected, a loud, annoying whine or whistle warns that the equipment has been disturbed. Such an alarm is especially useful when laptops must be left in meeting or presentation rooms overnight or during a break. Used in concert with guards, the alarms can offer reasonable protection at reasonable cost.

Detecting Theft

For some devices, protection is more important than detection. We want to keep someone from stealing certain systems or information at all costs. But for other devices, it may be enough to detect that an attempt has been made to access or steal hardware or software. For example, chaining down a disk makes it unusable. Instead, we try to detect when someone tries to leave a protected area with the disk or other protected object. In these cases, the protection mechanism should be small and unobtrusive. One such mechanism is similar to the protection used by many libraries, bookstores, or department stores. Each sensitive object is marked with a special label. Although the label looks like a normal pressure-sensitive one, its presence can be detected by a machine at the exit door if the label has not been disabled by an authorized party, such as a librarian or sales clerk. Similar security code tags are available for vehicles, people, machinery, and documents. Some tags are enabled by radio transmitters. When the detector sounds an alarm, someone must apprehend the person trying to leave with the marked object.

13.3.5 Interception of Sensitive Information

When disposing of a draft copy of a confidential report containing its sales strategies for the next five years, a company wants to be especially sure that the report is not reconstructable by one of its competitors. When the report exists only as hard copy, destroying the report is straightforward, usually accomplished by shredding or burning. But when the report exists digitally, destruction is more problematic. There may be many copies of the report in digital and paper form and in many locations (including on the computer and on storage media). There may also be copies in backups and archived in e-mail files. In this section, we look at several ways to dispose of sensitive information.

Shredding

Shredders have existed for a long time, as devices used by banks, government agencies, and others organizations to dispose of large amounts of confidential data. Although most of the shredded data is on paper, shredders can also be used for destroying printer ribbons and some types of disks and tapes. Shredders work by converting their input to thin strips or pulp, with enough volume to make it infeasible for most people to try to reconstruct the original from its many pieces. When data are extremely sensitive, some organizations burn the shredded output for added protection.

Overwriting Magnetic Data

Magnetic media present a special problem for those trying to protect the contents. When data are stored on magnetic disks, the ERASE or DELETE functions often simply change a directory pointer to free up space on the disk. As a result, the sensitive data are still recorded on the medium, and they can be recovered by analysis of the directory. A more secure way to destroy data on magnetic devices is to overwrite the data several times, using a different pattern each time. This process removes enough magnetic residue to prevent most people from reconstructing the original file. However, "cleaning" a disk in this fashion takes time. Moreover, a person using highly specialized equipment might be able to identify each separate message, much like the process of peeling off layers of wallpaper to reveal the wall beneath.

Degaussing

Degaussers destroy magnetic fields. Passing a disk or other magnetic medium through a degausser generates a magnetic flux so forceful that all magnetic charges are instantly realigned, thereby fusing all the separate layers. A degausser is a fast way to cleanse a magnetic medium, although there is still question as to whether it is adequate for use in the most sensitive of applications. For most users, a degausser is a fast way to

neutralize a disk or tape, permitting it to be reused by others.

Protecting Against Emanation: Tempest

Computer screens emit signals that can be detected from a distance. In fact, any components, including printers, disk drives, and processors, can emit information. Tempest is a U.S. government program under which computer equipment is certified as emission-free (that is, no detectable emissions). There are two approaches for preparing a device for Tempest certification: enclosing the device and modifying the emanations. The obvious solution to preventing emanations is to trap the signals before they can be picked up. Enclosing a device in a conductive case, such as copper, diffuses all the waves by conducting them throughout the case. Copper is a good conductor, and the waves travel much better through copper than through the air outside the case, so the emissions are rendered harmless. This solution works very well with cable, which is then enclosed in a solid, emanation-proof shield. Typically, the shielded cable is left exposed so that it is easy to inspect visually for any signs of tapping or other tampering. The shielding must be complete. That is, it does little good to shield a length of cable but not also shield the junction box at which that cable is connected to a component. The line to the component and the component itself must be shielded, too. The shield must enclose the device completely. If top, bottom, and three sides are shielded, emanations are prevented only in those directions. However, a solid copper shield is useless in front of a computer screen. Covering the screen with a fine copper mesh in an intricate pattern carries the emanation safely away. This approach solves the emanation problem while still maintaining the screen's usability.

Entire computer rooms or even whole buildings can be shielded in copper so that large computers inside do not leak sensitive emanations. Although it seems appealing to shield the room or building instead of each component, the scheme has significant drawbacks. A shielded room is inconvenient because it is impossible to expand the room easily as needs change. The shielding must be done carefully, because any puncture is a possible point of emanation. Furthermore, continuous metal pathways, such as water pipes or heating ducts, act as antennas to convey the emanations away from their source. Emanations can also be designed in such a way that they cannot be retrieved. This process is similar to generating noise in an attempt to jam or block a radio signal. With this approach, the emanations of a piece of equipment must be modified by addition of spurious signals. Additional processors are added to Tempest equipment specifically to generate signals that fool an interceptor. The exact Tempest modification methods are classified. As might be expected, Tempest-enclosed components are larger and heavier than their unprotected counterparts. Tempest testing is a rigorous program of the U.S. Department of Defense. Once a product has been approved, even a minor design modification, such as changing from one manufacturer's power supply to an equivalent one from another manufacturer, invalidates the Tempest approval. Therefore, these components are costly, ranging in price from 10 percent to 300 percent more than similar non-Tempest products. They are most appropriate in situations in which the data to be confined are of great value, such as top-level government information. Other groups with less dramatic needs can use other less rigorous shielding.

13.3.6 Contingency Planning

The key to successful recovery is adequate preparation. Seldom does a crisis destroy irreplaceable equipment; most computing systems, personal computers to mainframes are standard, off-the-shelf systems that can be easily replaced. Data and locally developed programs are more vulnerable because they cannot be quickly substituted from another source. Let us look more closely at what to do after a crisis occurs.

Backup

In many computing systems, some data items change frequently, whereas others seldom change. For example, a database of bank account balances changes daily, but a file of depositors' names and addresses changes much less often. Also the number of changes in a given period of time is different for these two files. These variations in number and extent of change relate to the amount of data necessary to reconstruct these files in the event of a loss. A backup is a copy of all or a part of a file to assist in re-establishing a lost file. In professional computing systems, periodic backups are usually performed automatically, often at night when system usage is low. Everything on the system is copied, including system files, user files, scratch files, and directories, so that the system can be regenerated after a crisis. This type of backup is called a complete backup. Complete backups are done at regular intervals, usually weekly or daily, depending on the criticality of the information or service provided by the system.

Major installations may perform revolving backups, in which the last several backups are kept. Each time a backup is done, the oldest backup is replaced with the newest one. There are two reasons to perform revolving backups: to avoid problems with corrupted media and to allow users or developers to retrieve old versions of a file. Another form of backup is a selective backup, in which only files that have been changed (or created) since the last backup are saved. In this case, fewer files must be saved, so the backup can be done more quickly. A selective backup combined with an earlier complete backup gives the effect of a complete backup in the time needed for only a selective backup. For each type of backup, we need the means to move from the backup forward to the point of failure. That is, we need a way to restore the system in the event of failure. In critical transaction systems, we address this need by keeping a complete record of changes since the last backup. Sometimes, the system state is captured by a combination of computer- and paper-based recording media. For example, if a system handles bank teller operations, the individual tellers duplicate their processing on paper records the deposit and withdrawal slips that accompany your bank transactions; if the system fails, the staff restores the latest backup version and reapplies all changes from the collected paper copies. Or the banking system creates a paper journal, which is a log of transactions printed just as each transaction completes.

Personal computer users often do not appreciate the need for regular backups. Even minor crises, such as a failed piece of hardware, can seriously affect personal computer users. With a backup, users can simply change to a similar machine and continue work.

Offsite Backup

A backup copy is useless if it is destroyed in the crisis, too. Many major computing installations rent warehouse space some distance from the computing system, far enough away that a crisis is not likely to affect the offsite location at the same time. As a backup is completed, it is transported to the backup site. Keeping a backup version separate from the actual system reduces the risk of its loss. Similarly, the paper trail is also stored somewhere other than at the main computing facility. Personal computer users concerned with integrity can take home a copy of important disks as protection or send a copy to a friend in another city. If both secrecy and integrity are important, a bank vault, or even a secure storage place in another part of the same building can be used. The worst place to store a backup copy is where it usually is stored: right next to the machine.

Networked Storage

With today's extensive use of networking, using the network to implement backups is a good idea. Storage providers sell space in which you can store data; think of these services as big network-attached disk drives. You rent space just as you would consume electricity: You pay for what you use. The storage provider needs to provide only enough total space to cover everyone's needs, and it is easy to monitor usage patterns and increase capacity as combined needs rise. Networked storage is perfect for backups of critical data because you can choose a storage provider whose physical storage is not close to your processing. In this way, physical harm to your system will not affect your backup. You do not need to manage tapes or other media and physically transport them offsite.

Cold Site

Depending on the nature of the computation, it may be important to be able to recover from a crisis and resume computation quickly. A bank, for example, might be able to tolerate a four-hour loss of computing facilities during a fire, but it could not tolerate a ten-month period to rebuild a destroyed facility, acquire new equipment, and resume operation. Most computer manufacturers have several spare machines of most models that can be delivered to any location within 24 hours in the event of a real crisis. Sometimes the machine will come straight from assembly; other times the system will have been in use at a local office. Machinery is seldom the hard part of the problem. Rather, the hard part is deciding where to put the equipment in order to begin a temporary operation. A cold site or shell is a facility with power and cooling available, in which a computing system can be installed to begin immediate operation. Some companies maintain their own cold sites, and other cold sites can be leased from disaster recovery companies. These sites usually come with cabling, fire prevention equipment, separate office space, telephone access, and other features. Typically, a computing center can have equipment installed and resume operation from a cold site within a week of a disaster.

Hot Site

If the application is more critical or if the equipment needs are more specialized, a hot site may be more appropriate. A hot site is a computer facility with an installed and ready-to run computing system. The system has peripherals, telecommunications lines, power supply, and even personnel ready to operate on short notice. Some companies maintain their own; other companies subscribe to a service that has available one or more locations with installed and running computers. To activate a hot site, it is necessary only to load software and data from offsite backup copies. Numerous services offer hot sites equipped with every popular brand and model of system. They provide diagnostic and system technicians, connected communications lines, and an operations staff. The hot site staff also assists with relocation by arranging transportation and housing, obtaining needed blank forms, and acquiring office space. Because these hot sites serve as backups for many customers, most of whom will not need the service, the annual cost to any one customer is fairly low. The cost structure is like insurance: The likelihood of an auto accident is low, so the premium is reasonable, even for a policy that covers the complete replacement cost of an expensive car. Notice, however, that the first step in being able to use a service of this type is a complete and timely backup.

13.4 Summary

- The administration of security draws on skills slightly different from the technical skills. The security administrator must understand not just security assets, threats, vulnerabilities, and controls, but management and implementation. In this chapter we examined how security is administered.
- An organizational security policy is a document that specifies the organization's goals regarding security. It lists policy elements that are statements of actions that must or must not be taken to preserve those goals. Policy documents often lead to implementational procedures. Also, user education and awareness activities ensure that users are aware of policy restrictions.
- Physical security concerns the physical aspects of computing: the devices themselves and harm that can come to them because of the buildings in which they are contained. Physical security addresses two branches of threats: natural threats to buildings and the infrastructure, and human threats. Redundancy and physical controls address physical security threats.
- The administration of security has a strong human component, from the writing of plans and policies, to the mental work in performing a risk analysis, to the human guards that implement or reinforce many physical controls.
- By no means have we covered all of physical security in this brief introduction. Professionals become experts at individual aspects, such as fire control or power provision. We have to protect the facility against many sorts of disasters, from weather to chemical spills and vehicle crashes to explosions. It is impossible to predict what will occur or when.
- The physical security manager has to consider all assets and a wide range of harm. Malicious humans seeking physical access are a different category of threat agent. With them, you can consider motive or objective: is it theft of equipment, disruption of processing, interception of data, or access to service? Fences, guards, solid walls, and locks will deter or prevent most human attacks. But you always need to ask where weaknesses remain; a solid wall has a weakness in every door and window.
- The primary physical controls are strength and duplication. Strength means overlapping controls implementing a defense-in-depth approach so that if one control fails, the next one will protect. People who built ancient castles practiced this philosophy with moats, walls, drawbridges, and arrow slits. Duplication means eliminating single points of failure. Redundant copies of data protect against harm to one copy from any cause. Spare hardware components protect against failures.

13.5 Review Questions

a) What are the various audiences and how is balance managed among all the audiences?
b) Explain the contents of a good security plan.
c) What are the characteristics of a good security plan?
d) What natural disasters are computers prone to and how can they be saved?
e) What are the ways in which human vandals can cause problems in physical security?

f) How can sensitive information be intercepted?
g) Write a short note on contingency planning.

13.6 Bibliography, References and Further Reading

- *Security in Computing by* C. P. Pfleeger, and S. L. Pfleeger, Pearson Education.
- *Computer Security: Art and Science by* Matt Bishop, Pearson Education.
- *Cryptography And Network Security: Principles and practice by* Stallings
- *Network Security by* Kaufman, Perlman, Speciner
- *Network Security : A Beginner's Guide by* Eric Maiwald, TMH
- *Java Network Security by* Macro Pistoia, Pearson Education
- *Principles of information security by* Whitman, Mattord, Thomson

IDOL

Chapter 14

Privacy

IDOL

14.0 Objectives

After reading this chapter, the reader will be able to understand

- the privacy aspect of security and
- authentication effects on privacy.

14.1 Introduction

Computers did not invent or even cause privacy issues; we had those long before computers and probably even before written language. But computers' high-speed processing and data storage and transmission capabilities made possible data collection and correlation that affect privacy. Because privacy is part of confidentiality, it is an aspect of computer security.

Privacy is a human right, although people can legitimately disagree over when or to what extent privacy is deserved; this disagreement may have cultural, historical, or personal roots. Laws and ethics can set the baseline for and enforce expectations of privacy. But inherently, the right to privacy depends on the situation and the affected parties. And just as confidentiality, integrity, and availability can conflict, so too can privacy and other aspects of security. We won't take a position on when a right to privacy should be enforceable because that is outside the scope of this book. You might characterize the presentation of this chapter as "assuming a particular right to privacy exists, what are its implications in computing and information technology?" We as citizens help decide the contours of privacy rights; we as computer security experts implement those decisions in computer systems.

Privacy is also a broad topic, affected by computing but not just a security topic. We don't want to try to survey all possible privacy issues in this chapter, just those inextricably linked to computer security. In this chapter we look at the meaning of information privacy. We examine identification and authentication, two familiar aspects of computing that have significant privacy implications.

14.2 Privacy Concepts

In this section we examine privacy, first from its general or common usage and then as it applies in technological situations.

14.2.1. Aspects of Information Privacy

Information privacy has three aspects: sensitive data, affected parties, and controlled disclosure. In fact, these aspects are similar to the three elements of access control from earlier chapters: subject, object, and access rights. We examine these three in turn.

Controlled Disclosure

What is privacy? A good working definition is that privacy is the right to control who knows certain aspects about you, your communications, and your activities. In other words, you voluntarily choose who can know things about you and what those things are. People ask you for your telephone number: your auto mechanic, a clerk in a store, your tax authority, a new business contact, or a cute person in a bar. You consider why the person wants the number and decide whether to give it out. But the key point is *you* decide. So, privacy is something over which you have considerable influence.

You do not have complete control, however. Once you give your number to someone else, your control is diminished because it depends in part on what someone else does. As soon as you give out your number, you transfer authority and control to someone else. You may say “don't give my number to anyone else”, “use discretion”, or “I am sensitive about my privacy”, but you do not control the other person. You have to trust the other person to comply with your wishes, whether you state them explicitly or not. This problem is similar to the propagation problem of computer security: Anyone who has access to an object can copy, transfer, or propagate that object or its content to others without restriction.

Sensitive Data

Someone asks you for your shoe size; you might answer, “I'm a very private person and cannot imagine why you would want to know such an intimate detail” or you could say “10C”; some

people find that data more sensitive than others. We know things people usually consider sensitive, such as financial status, certain health data, unsavoury events in their past, and the like, so if you learn something you consider sensitive about someone, you will keep it quiet. But most of us are not too sensitive about our shoe size, so we don't normally protect that if we learn it about someone else. Of course, if a friend told me not to pass that along, I wouldn't. It is not up to me to question why someone else considers something private. Here are examples (in no particular order) of data many people consider private.

- identity, the ownership of private data and the ability to control its disclosure
- finances, credit, bank details
- legal matters
- medical conditions, drug use, DNA, genetic predisposition to illnesses
- voting, opinions, membership in advocacy organizations
- preferences: religion, sexuality
- biometrics, physical characteristics, polygraph results, fingerprints
- diaries, poems, correspondence, recorded thoughts
- privileged communications with professionals such as lawyers, accountants, doctors, counselors, and clergy
- performance: school records, employment ratings
- activities: reading habits, web browsing, music, art, videos
- air travel data, general travel data, a person's location (present and past)
- communications: mail, e-mail, telephone calls, spam
- history: "youthful indiscretions," past events
- illegal activities, criminal records

Privacy is also affected by who you are. When you are in a room of people you don't know, perhaps at a reception, someone may come up to you and say, “So you are the man who baked that beautiful cake over there; I really appreciate your skills as a pastry chef”. It feels kind of nice to get that kind of recognition. Conversely, a friend was frequently on local television; she far preferred having dinner at home instead of going to a restaurant because she had grown tired of people rushing up to her saying “you're [Olga], I see you all the time on TV”. Public personalities cherish the aspects of privacy they retain. World champion athletes cannot avoid having their results made public, whereas you might not want everyone to know how poorly you finished in the last event. Culture also influences what people consider sensitive. In general, a person's privacy expectations depend on context: who is affected and what the prevailing norm of privacy is.

Affected Subject

This brings us to another point about privacy: Individuals, groups, companies, organizations, and governments all have data they consider sensitive. So far, we have described privacy from the standpoint of a person. Companies may have data they consider private or sensitive: product plans, key customers, profit margins, and newly discovered technologies. For organizations such as companies, privacy usually relates to gaining and maintaining an edge over the competition. Other organizations, for example, schools, hospitals, or charities, may need to protect personal data on their students, patients, or donors, or they may want to control negative news, and so forth. Governments consider military and diplomatic matters sensitive, but they also recognize a responsibility to keep confidential data they collect from citizens, such as tax information. We may use terms like subject or owner to cover privacy issues affecting people, groups, and the like.

Privacy is an aspect of confidentiality. As we have learned throughout this book, the three security goals of confidentiality, integrity, and availability conflict, and confidentiality frequently conflicts with availability. If you choose not to have your telephone number published in a directory, that also means some people will not be able to reach you by telephone.

IDOL

14.2.2. Computer Related Privacy Problems

You may notice that many of the kinds of sensitive data and many of the points about privacy have nothing to do with computers. These sensitivities and issues predate computers. Computers and networks have only affected the feasibility of some unwanted disclosures. Public records offices have long been open for people to study the data held there, but the storage capacity and speed of computers have given us the ability to amass, search, and correlate. Search engines have given us the ability to find one data item out of billions, the equivalent of finding one sheet of paper out of a warehouse full of boxes of papers. Furthermore, the openness of networks and the portability of technology (such as laptops, PDAs, cell phones, and memory devices) have greatly increased the risk of disclosures affecting privacy. Eight dimensions of privacy (specifically as it relates to the web, although the definitions carry over naturally to other types of computing) are as follows:

- *Information collection*: Data are collected only with knowledge and explicit consent.
- *Information usage*: Data are used only for certain specified purposes.
- *Information retention*: Data are retained for only a set period of time.
- *Information disclosure*: Data are disclosed to only an authorized set of people.
- *Information security*: Appropriate mechanisms are used to ensure the protection of the data.
- *Access control*: All modes of access to all forms of collected data are controlled.
- *Monitoring*: Logs are maintained showing all accesses to data.
- *Policy changes*: Less restrictive policies are never applied after-the-fact to already obtained data.

Here are the privacy issues that have come about through use of computers.

Data Collection

As we have previously said, advances in computer storage make it possible to hold and manipulate huge numbers of records. Disks on ordinary consumer PCs are measured in gigabytes (10^9 bytes), and commercial storage capacities often measure in terabytes (10^{12} bytes). In 2006, EMC Corporation announced a storage product whose capacity exceeds one petabyte (10^{15} bytes). Indiana University plans to acquire a supercomputer with one petabyte of storage, and the San Diego Supercomputer Center has online storage of one petabyte and offline archives of seven petabytes. Estimates of Google's stored data are also in the petabyte range. We have both devices to store massive amounts of data and the data to fill those devices. Whereas physical space limited storing (and locating) massive amounts of printed data, electronic data take relatively little space. We never throw away data; we just move it to slower secondary media or buy more storage.

No Informed Consent

Where do all these bytes come from? Although some are from public and commercial sources (newspapers, web pages, digital audio, and video recordings) and others are from intentional data transfers (tax returns, a statement to the police after an accident, readers' survey forms, school papers), still others are collected without announcement. Telephone companies record the date, time, duration, source, and destination of each telephone call. ISPs track sites visited. Some sites keep the IP address of each visitor to the site. The user is not necessarily aware of this third category of data collection and thus cannot be said to have given informed consent.

Loss of Control

We realize that others may keep data we give them. When you order merchandise online, you know you have just released your name, probably some address and payment data, and the items you purchased. Or when you use a customer appreciation card at a store, you know the store can associate your identity with the things you buy. Having acquired your data, a merchant can redistribute it to anyone. The fact that you booked one brand of hotel room through a travel agent

IDOL

could be sold to other hotels. If you frequently telephone someone in one city and have taken several plane trips to that city, local stores, restaurants, or tourist attractions in that city might want your name. You have little control over dissemination (or re-dissemination) of your data. We do not always appreciate the ramifications of lost control. Suppose in a moment of anger you dash off a strong note to someone. Although 100 years ago you would have written the note on paper and 50 years ago you would have voiced the comment by telephone, now you post the message to a blog.

Next suppose you have a change of heart and you want to retract your angry note. Let us consider how you would deal with these three forms of the communication. For the written note, you write a letter of apology, your recipient tears up your first note, and no trace remains. In the second case you telephone to apologize and all that remains is a memory. As for the blog, you delete your posting. However, several other people might have seen your original posting and copied it to blogs or other web sites that you do not control. Search engines might have found the original or copies. And other people might have picked up your words and circulated them in e-mail. Thus, with letters and phone calls, we can usually obliterate something we want to retract. But once something is out of your control on the web, it may never be deleted. This example concerned something you wrote.

A similar situation concerns something written about you. Someone else has posted something on the web that is personal about you and you want it removed. Even if the poster agrees, you may not be able to remove all its traces. Finally, some people are finding they reveal more than they should on sites like myspace.com. Prospective employees are being turned down for jobs because of things they have written. The web is a great historical archive, but because of archives, caches, and mirror sites, things posted on the web may never go away.

A second issue of loss of control concerns data exposure. Suppose a company holds data about you and that company's records are exposed in a computer attack. The company may not be responsible for preventing harm to you, compensating you if you are harmed, or even informing you of the event.

Ownership of the Data

In the cases just described, customer details are being marketed. Information about you is being sold and you have no control; nor do you get to share in the profit. Even before computers customer data were valuable. Mailing lists and customer lists were company assets that were safeguarded against access by the competition. Sometimes companies rented their mailing lists when there was not a conflict with a competitor. But in those cases, the subject of the data, the name on the list, did not own the right to be on the list or not. With computers the volume and sources of data have increased significantly, but the subject still has no rights.

These issues, loss of control, no informed consent, no ownership of data have significant privacy implications. The way we address these kinds of issues is with policies, written statements of practice that inform all affected parties of their rights.

14.3 Privacy Principles and Policies

In the United States, interest in privacy and computer databases dates back at least to the early 1970s. (It is worth noting that the U.S. Watergate burglary occurred in 1972. Shortly after, reports surfaced that Nixon maintained an enemies list and had used IRS records as a means of combating adversaries. Thus, people in the United States were sensitive about privacy at that time. Public concern for privacy has varied over the years.) In the early 1970s, a committee developed privacy principles that have affected U.S. laws and regulations and that also set the path for privacy legislation in other countries.

14.3.1. Fair Information Policies

In 1973 Willis Ware of the RAND Corporation chaired a committee to advise the Secretary of the U.S. Department of Human Services on privacy issues. The report proposes a set of principles of fair information practice:

- *Collection limitation:* Data should be obtained lawfully and fairly.
- *Data quality:* Data should be relevant to their purposes, accurate, complete, and up-to-date.
- *Purpose specification:* The purposes for which data will be used should be identified and the data destroyed if no longer necessary to serve that purpose.
- *Use limitation:* Use for purposes other than those specified is authorized only with consent of the data subject or by authority of law.
- *Security safeguards:* Procedures to guard against loss, corruption, destruction, or misuse of data should be established.
- *Openness:* It should be possible to acquire information about the collection, storage, and use of personal data systems.
- *Individual participation:* The data subject normally has a right to access and to challenge data relating to her.
- *Accountability:* A data controller should be designated and accountable for complying with the measures to give effect to the principles.

These principles describe the rights of individuals, not requirements on collectors; that is, the principles do not require protection of the data collected. Ware raises the problem of linking data in multiple files and of overusing keys, such as social security numbers, that were never intended to be used to link records. And although he saw that society was moving toward a universal identity number, he feared that movement would be without plan (and hence without control). He was right, even though he could not have foreseen the amount of data exchanged 30 years later. Turn and Ware consider protecting the data themselves, recognizing that collections of data will be attractive targets for unauthorized access attacks. They suggest four ways to protect stored data:

- Reduce exposure by limiting the amount of data maintained, asking for only what is necessary and using random samples instead of complete surveys.
- Reduce data sensitivity by interchanging data items or adding subtle errors to the data (and warning recipients that the data have been altered).
- Anonymize the data by removing or modifying identifying data items.
- Encrypt the data.

14.3.2. U.S. Privacy Laws

Ware and his committee expected these principles to apply to all collections of personal data on individuals. Unfortunately, that is not the way the legislation developed. The Ware committee report led to the 1974 Privacy Act (5 USC 552a), which embodies most of these principles, although that law applies only to data maintained by the U.S. government. The Privacy Act is a broad law, covering all data collected by the government. It is the strongest U.S. privacy law because of its breadth: It applies to all personal data held anywhere in the government.

The United States subsequently passed laws protecting data collected and held by other organizations, but these laws apply piecemeal, by individual data type. Consumer credit is addressed in the Fair Credit Reporting Act, healthcare information in the Health Insurance Portability and Accountability Act (HIPAA), financial service organizations in the GrammLeachBliley Act (GLBA), children's web access in the Children's Online Privacy Protection Act (COPPA), and student records in the Federal Educational Rights and Privacy Act. Not surprisingly these separate laws are inconsistent in protecting privacy. Laws and regulations do help in some aspects of privacy protection. Antón et al. investigated the impact of the HIPAA law by analyzing companies' posted privacy policies before and after the privacy provisions of the law

became effective. They found the following in policies posted after HIPAA:

- Statements on data transfer (to other organizations) were more explicit than before HIPAA.
- Consumers still had little control over the disclosure or dissemination of their data.
- Statements were longer and more complex, making them harder for consumers to understand.
- Even within the same industry branch (such as drug companies), statements varied substantially, making it hard for consumers to compare policies.
- Statements were unique to specific web pages, meaning they covered more precisely the content and function of a particular page.

A problem with many laws is that the target areas of the laws still overlap: Which law (if any) would require privacy protection of a university student's health center bills paid by credit card? The laws have different protection and handling requirements, so it is important to determine which law applies to a single piece of data. Also, gaps between laws are not covered. As new technologies (such as computers, the Internet, or cell phones) are developed, either existing privacy laws have to be reinterpreted by the courts to apply to the new technologies or new laws have to be passed, which takes time. Sometimes the privacy provisions of a law are a second purpose, somewhat disguised by the first purpose of the law. As an example, the primary purpose of HIPAA was to ensure that people who left or were terminated from one job had health insurance to cover them until they got another job; the privacy aspects were far less prominent as the law was being developed.

14.3.3. Controls on Commercial Web Sites

The e-Government Act places strong controls on government data collection through web sites. As we described, privacy outside the government is protected by law in some areas, such as credit, banking, education, and healthcare. But there is no counterpart to the e-Government Act for private companies.

No Deceptive Practices

The Federal Trade Commission has the authority to prosecute companies that engage in deceptive trade or unfair business practices. If a company advertises in a false or misleading way, the FTC can sue. The FTC has used that approach on web privacy: If a company advertises a false privacy protection that is, if the company says it will protect privacy in some way but does not do so the FTC considers that false advertising and can take legal action. Because of the FTC, privacy notices at the bottom of web sites do have meaning. This practice leads to a bizarre situation, however. A company is allowed to collect personal information and pass it in any form to anyone, as long as the company's privacy policy said it would do so, or at least the policy did not say it would not do so. Vowing to maintain privacy and intentionally not doing so is an illegal deceptive practice. Stating an intention to share data with marketing firms or “other third parties” makes such sharing acceptable, even though the third parties could be anyone.

Examples of Deceptive Practices

The FTC settled a prosecution in 2005 against CartManager International, a firm that runs familiar web shopping cart software to collect items of an order, obtain the purchaser's name and address, and determine shipping and payment details. This software runs as an application under other well-known retail merchants' web sites to handle order processing. Some of these other retailers had privacy statements on their web sites saying, in effect, that they would not sell or distribute customers' data, but CartManager did sell the data it collected. The FTC held that the relationship to CartManager was invisible to users, and so the policy from the online merchants applied also to CartManager.

In another case, Antón analyzed the privacy policy posted on the web site of Jet Blue airlines and

found it misleading. Jet Blue stated that it would not disclose passenger data to third parties. It then released passenger data, "in response to a special request from the Department of Defense" to Torch Concepts, which in turn passed it to the Defense Department to use to test passenger screening algorithms for airline security. The data in question involved credit card information: Clearly the only reason for Jet Blue to have collected those data from passengers was to process charges for airline tickets. The analysis by Antón is interesting for two reasons: First, Jet Blue violated its own policy. Second, the Department of Defense may have circumvented the e-Government Act by acquiring from a private company data it would not have been able to collect as a government entity. The purpose for which the data were originally collected was ordinary business and accounting activities of Jet Blue. Using those same records to screen for terrorists was outside the scope of the original data collection. Commercial sites have no standard of content comparable to the FTC recommendation from the e-Government Act. Some companies display solid and detailed privacy statements that they must obey. On the other hand, you may find no statement at all, which gives the company the greatest flexibility because it is impossible to lie when saying nothing. Cranor makes some recommendations for useful web privacy policies.

14.3.4. Non – U.S. Privacy Principles

In 1981, the Council of Europe (an international body of 46 European countries, founded in 1949) adopted Convention 108 for the protection of individuals with regard to the automatic processing of personal data, and in 1995, the European Union (E.U.) adopted Directive 95/46/EC on the processing of personal data. Directive 95/46/EC, often called the European Privacy Directive, requires that rights of privacy of individuals be maintained and that data about them be

- processed fairly and lawfully
- collected for specified, explicit and legitimate purposes and not further processed in a way incompatible with those purposes
- adequate, relevant, and not excessive in relation to the purposes for which they are collected and/or further processed
- accurate and, where necessary, kept up to date; every reasonable step must be taken to ensure that inaccurate or incomplete data having regard for the purposes for which they were collected or for which they are further processed, are erased or rectified
- kept in a form that permits identification of data subjects for no longer than is necessary for the purposes for which the data were collected or for which they are further processed

In addition, individuals have the right to access data collected about them, to correct inaccurate or incomplete data, and to have those corrections sent to those who have received the data. The report adds three more principles to the Fair Information Policies.

- *Special protection for sensitive data:* There should be greater restrictions on data collection and processing that involves "sensitive data." Under the E.U. data protection directive, information is sensitive if it involves "racial or ethnic origin, political opinions, religious beliefs, philosophical or ethical persuasion . . . [or] health or sexual life."
- *Data transfer:* This principle explicitly restricts authorized users of personal information from transferring that information to third parties without the permission of the data subject.
- *Independent oversight:* Entities that process personal data should not only be accountable but should also be subject to independent oversight. In the case of the government, this requires oversight by an office or department that is separate and independent from the unit engaged in the data processing. Under the data protection directive, the independent overseer must have the authority to audit data processing systems, investigate complaints brought by individuals, and enforce sanctions for noncompliance.

These requirements apply to governments, businesses, and other organizations that collect personal data. Since the 1995 directive, the European Union has extended coverage to telecommunications systems and made other changes to adapt to advances in technology. In addition to European

countries and the United States, other countries, such as Japan, Australia, and Canada, have passed laws protecting the privacy of personal data about individuals. Different laws in different jurisdictions will inevitably clash. Relations between the European Union and the United States have been strained over privacy because the E.U. law forbids sharing data with companies or governments in countries whose privacy laws are not as strong as those of the E.U.

14.3.5. Anonymity, Multiple Identities

One way to preserve privacy is to guard our identity. Not every context requires us to reveal our identity, so some people wear a form of electronic mask.

Anonymity

A person may want to do some things anonymously. For example, a rock star buying a beach house might want to avoid unwanted attention from neighbours, or someone posting to a dating list might want to view replies before making a date. Some people like the anonymity of the web because it reduces fears of discrimination. Fairness in housing, employment, and association are easier to ensure when the basis for potential discrimination is hidden. Also, people researching what they consider a private matter, such as a health issue or sexual orientation, are more likely to seek first information from what they consider an anonymous source, turning to a human when they have found out more about their situation.

Anonymity creates problems, too. How does an anonymous person pay for something? A trusted third party (for example, a real estate agent or a lawyer) can complete the sale and preserve anonymity. But then you need a third party and the third party knows who you are. Chaum studied this problem and devised a set of protocols by which such payments could occur without revealing the buyer to the seller.

Multiple Identities Linked or Not

Most people already have multiple identities. To your bank you might be the holder of account 123456, to your motor vehicles bureau you might be the holder of driver's license number 234567, and to your credit card company you might be the holder of card 345678. For their purposes, these numbers are your identity; the fact that each may (or may not) be held in your name is irrelevant. The name does become important if it is used as a way to link these records. How many people share your name? Can (or should) it serve as a key value to link these separate databases? We ignore the complication of misspellings and multiple valid forms (with and without middle initials, with full middle name, with one of two middle names if you have them, and so forth). Suppose you changed your name legally but never changed the name on your credit card; then your name could not be used as a key on which to link. Another possible link field is address. However, trying to use an address on which to link presents another risk: Perhaps a criminal lived in your house before you bought it. You should not have to defend your reputation because of a previous occupant. Now we need to match on date, too, so we connect only people who actually lived in a house at the same time. Then we need to address the problem of group houses or roommates of convenience, and so forth. As computer scientists, we know we can program all these possibilities, but that requires careful and time-consuming consideration of the potential problems before designing the solution. We can also see the potential for misuse and inaccuracy.

Linking identities correctly to create dossiers and break anonymity creates privacy risks but linking them incorrectly creates much more serious risks for the use of the data and the privacy of affected people. If we think carefully we can determine many of the ways such a system would fail, but that approach is potentially expensive and time consuming. The temptation to act quickly but inaccurately will also affect privacy.

Pseudonymity

Sometimes, full anonymity is not wanted. A person may want to order flower bulbs but not be placed on a dozen mailing lists for gardening supplies. But the person does want to be able to place similar orders again, asking for the same colour tulips as before. This situation calls for pseudonyms, unique identifiers that can be used to link records in a server's database but that cannot be used to trace back to a real identity. Multiple identities can also be convenient, for example, having a professional e-mail account and a social one. Similarly, disposable identities can be convenient. When you sign up for something and you know your e-mail address will be sold many times, you might get a new e-mail address to use until the spam and other unsolicited e-mail are oppressive, and then you discard the address. These uses are called pseudonymity. These ways protect our privacy because we do not have to divulge what we consider sensitive data. But they also show we need a form of privacy protection that is unavailable. The Swiss bank account was a classic example of a pseudonym. Each customer had only a number to access the account. Presumably anyone with that number could perform any transaction on the account. While such accounts were in use, Swiss banks had an outstanding reputation for maintaining the anonymity of the depositors. Some people register pseudonyms with e-mail providers so that they have anonymous drop boxes for e-mail. Others use pseudonyms in chat rooms or with online dating services.

14.3.6. Government and Privacy

The government gathers and stores data on citizens, residents, and visitors. Government facilitates and regulates commerce and other kinds of personal activities such as healthcare, employment, education, and banking. In those roles the government is both an enabler or regulator of privacy and a user of private data. Government use of private data should be controlled. In this section we consider some of the implications of government access to private data.

Authentication

Government plays a complex role in personal authentication. Many government agencies (such as the motor vehicles bureau) use identifiers to perform their work. Authentication documents (such as passports and insurance cards) often come from the government. The government may also regulate the businesses that use identification and authentication keys. And sometimes the government obtains data based on those keys from others (for example, the U.S. government planned to buy credit reports from private companies to help with screening airline passenger lists for terrorists). In these multiple roles, the government may misuse data and violate privacy rights.

Data Access Risks

Recognizing that there were risks in government access to personal data, the Secretary of Defense appointed a committee to investigate private data collection. The Technology and Privacy Advisory Committee, chaired by Newton Minow, former chair of the Federal Communications Commission, produced its report in 2004, they recognized risks when the government started to acquire data from other parties:

- *data errors:* ranging from transcription errors to incorrect analysis
- *inaccurate linking:* two or more correct data items but incorrectly linked on a presumed common element
- *difference of form and content:* precision, accuracy, format, and semantic errors
- *purposely wrong:* collected from a source that intentionally gives incorrect data, such as a forged identity card or a false address given to mislead
- *false positive:* an incorrect or out-of-date conclusion that the government does not have data to verify or reject, for example, delinquency in paying state taxes
- *mission creep:* data acquired for one purpose leading to a broader use because the data will

support that mission
- *poorly protected:* data of questionable integrity because of the way it has been managed and handled

These risks apply to all branches of government, and most of them apply to private collection and use of data.

Steps to Protect Against Privacy Loss

The committee recommended several steps the government can take to help safeguard private data.

- *Data minimization:* Obtain the least data necessary for the task. For example, if the goal is to study the spread of a disease, only the condition, date, and vague location (city or county) may suffice; the name or contact information of the patient may be unnecessary.
- *Data anonymization:* Where possible, replace identifying information with untraceable codes (such as a record number); but make sure those codes cannot be linked to another database that reveals sensitive data.
- *Audit trail:* Record who has accessed data and when, both to help identify responsible parties in the event of a breach and to document the extent of damage.
- *Security and controlled access:* Adequately protect and control access to sensitive data.
- *Training:* Ensure people accessing data understand what to protect and how to do so.
- *Quality:* Take into account the purpose for which data were collected, how they were stored, their age, and similar factors to determine the usefulness of the data.
- *Restricted usage:* Different from controlling access, review all proposed uses of the data to determine if those uses are consistent with the purpose for which the data were collected and the manner in which they were handled (validated, stored, controlled).
- *Data left in place:* If possible, leave data in place with the original owner. This step helps guard against possible misuses of the data from expanded mission just because the data are available.
- *Policy:* Establish a clear policy for data privacy. Do not encourage violation of privacy policies.

These steps would help significantly to ensure protection of privacy.

14.3.7. Identity Theft

As the name implies, identity theft is taking another person's identity. Use of another person's credit card is fraud; taking out a new credit card in that person's name is identity theft. Identity theft has risen as a problem from a relatively rare issue in the 1970s. In 2005, the U.S. Federal Trade Commission received over 250,000 complaints of identity theft. Most cases of identity theft become apparent in a month or two when fraudulent bills start coming in. By that time the thief has made a profit and has dropped this identity, moving on to a new victim. Having relatively few unique keys facilitates identity theft: A thief who gets one key can use that to get a second, and those two to get a third. Each key gives access to more data and resources. Few companies or agencies are set up to ask truly discriminating authentication questions (such as the grocery store at which you frequently shop or the city to which you recently bought an airplane ticket or third digit on line four of your last tax return). Because there are few authentication keys, we are often asked to give the same key (such as mother's maiden name) out to many people, some of whom might be part-time accomplices in identity theft.

14.4 Authentication and Privacy

In an earlier chapter we studied authentication, which we described as a means of proving or verifying a previously given identity. We also discussed various authentication technologies, which

are subject to false accept (false positive) and false reject (false negative) limitations. A social problem occurs when we confuse authentication with identification. We know that passwords are a poor discriminator. You would not expect all users of a system to have chosen different passwords. All we need is for the ID, password pair to be unique. On the other end of the spectrum, fingerprints and the blood vessel pattern in the retina of the eye are unique: given a fingerprint or retina pattern we expect to get but one identity that corresponds or to find no match in the database. That assumes we work with a good image. If the fingerprint is blurred or incomplete (not a complete contact or on a partly unsuitable surface), we might get several possible matches. If the possible matches are A, B, and C and the question is whether the print belongs to B, it is probably acceptable to allow the access on the grounds that the identity was among a small set of probable matches. Other authenticators are less sophisticated still. Hand geometry or the appearance of a face does not discriminate so well. Face recognition, in particular, is highly dependent on the quality of the facial image: Evaluating a photograph of one person staring directly into a camera is very different from trying to work with one face in the picture of a crowd. Two different purposes are at work here, although the two are sometimes confused. For authentication we have an identity and some authentication data, and we ask if the authentication data match the pattern for the given identity. For identification, we have only the authentication data and we ask which identity corresponds to the authenticator. The second is a much harder question to answer than the first. For the first, we can say the pattern matches some percentage of the characteristics of our stored template, and based on the percentage, we declare a match or no match. For the second question, we do not know if the subject is even in the database. So even if we find several potential matches at various percentages, we do not know if there might be an even better match with a template not in our database.

14.4.1. What authentication means

We use the term authentication to mean three different things: We authenticate an individual, identity, or attribute. An individual is a unique person. Authenticating an individual is what we do when we allow a person to enter a controlled room: We want only that human being to be allowed to enter. An identity is a character string or similar descriptor, but it does not necessarily correspond to a single person, nor does each person have only one name. We authenticate an *identity* when we acknowledge that whoever (or whatever) is trying to log in as *admin* has presented an authenticator valid for that account. Similarly, authenticating an identity in a chat room as SuzyQ does not say anything about the person using that identifier: It might be a 16-year-old girl or a pair of middle-aged male police detectives, who at other times use the identity FrereJacques. Finally, we authenticate an *attribute* if we verify that a person has that attribute. An attribute is a characteristic. Here's an example of authenticating an attribute. Some places require one to be 21 or older in order to drink alcohol. A club's doorkeeper verifies a person's age and stamps the person's hand to show that the patron is over 21. Note that to decide, the doorkeeper may have looked at an identity card listing the person's birth date, so the doorkeeper knew the person's exact age to be 24 years, 6 months, 3 days, or the doorkeeper might be authorized to look at someone's face and decide if the person looks so far beyond 21 that there is no need to verify. The stamp authenticator signifies only that the person possesses the attribute of being 21 or over.

In computing applications we frequently authenticate individuals, identities, and attributes. Privacy issues arise when we confuse these different authentications and what they mean. For example, the U.S. social security number was never intended to be an identifier, but now it often serves as an identifier, an authenticator, a database key, or all of these. When one data value serves two or more uses, a person acquiring it for one purpose can use it for another.

Relating an identity to a person is tricky. In an earlier chapter, we tell the story of rootkits, malicious software by which an unauthorized person can acquire supervisory control of a computer. Suppose the police arrest Ionut for chewing gum in public and seize his computer. By examining the computer the police find evidence connecting that computer to an espionage case. The police show

incriminating e-mail messages from Ionut on Ionut's computer and charge him. In his defense, Ionut points to a rootkit on his computer. He acknowledges that his computer may have been used in the espionage, but he denies that he was personally involved. The police have, he says, drawn an unjustifiable connection between Ionut's identity in the e-mail and Ionut the person. The rootkit is a plausible explanation for how some other person acted under the identity of Ionut. This example shows why we must carefully distinguish individual, identity, and attribute authentication.

Individual Authentication

There are relatively few ways of identifying an individual. When we are born, for most of us our birth is registered at a government records office, and we (probably our parents) receive a birth certificate. A few years later our parents enrol us in school, and they have to present the birth certificate, which then may lead to receiving a school identity card. We submit the birth certificate and a photo to get a passport or a national identity card. We receive many other authentication numbers and cards throughout life. The whole process starts with a birth certificate issued to (the parents of) a baby, whose physical description (height, weight, even hair colour) will change significantly in just months. Birth certificates may contain the baby's fingerprints but matching a poorly taken fingerprint of a new born baby to that of an adult is challenging at best. Fortunately, in most settings it is acceptable to settle for weak authentication for individuals: A friend who has known you since childhood, a schoolteacher, neighbours, and co-workers can support a claim of identity.

Identity Authentication

We all use many different identities. When you buy something with a credit card, you do so under the identity of the credit card holder. You check into a hotel and get a magnetic stripe card instead of a key, and the door to your room authenticates you as a valid resident for the next three nights. If you think about your day, you will probably find 10 to 20 different ways some identity of you has been authenticated.

From a privacy standpoint, there may or may not be ways to connect all these different identities. A credit card links to the name and address of the card payer, who may be you, your spouse, or anyone else willing to pay your expenses. Your auto toll device links to the name and perhaps address of whoever is paying the tolls: you, the car's owner, or an employer. When you make a telephone call, there is an authentication to the account holder of the telephone, and so forth. Sometimes we do not want an action associated with an identity. For example, an anonymous tip or "whistle-blower's" telephone line is a means of providing anonymous tips of illegal or inappropriate activity. If you know your boss is cheating the company, confronting your boss might not be a good career-enhancing move. You probably don't even want there to be a record that would allow your boss to determine who reported the fraud. So, you report it anonymously. You might take the precaution of calling from a public phone so there would be no way to trace the person who called. In that case, you are purposely taking steps so that no common identifier could link you to the report.

Because of the accumulation of data, however, linking may be possible. As you leave your office to go to a public phone, there is a record of the badge you swiped at the door. A surveillance camera shows you standing at the public phone. The record of the coffee shop has a timestamp showing when you bought your coffee (using your customer loyalty card) before returning to your office. The time of these details matches the time of the anonymous tip by telephone. In the abstract these data items do not stand out from millions of others. But someone probing a few minutes around the time of the tip can construct those links. In this example, linking would be done by hand. Ever-improving technology permits more parallels like these to be drawn by computers from seemingly unrelated and uninteresting datapoints. Therefore, to preserve our privacy we may thwart attempts to link records. A friend gives a fictitious name when signing up for customer loyalty cards at stores. Another friend makes dinner reservations under a pseudonym. In one store they always ask

for my telephone number when I buy something, even if I pay cash. Records clerks do not make the rules, so it is futile asking them why they need my number. If all they want is a number, I gladly give them one; it just doesn't happen to correspond to me.

Anonymized Records

Part of privacy is linkages: Some person is named Erin, some person has the medical condition diabetes; neither of those facts is sensitive. The linkage that Erin has diabetes becomes sensitive. Medical researchers want to study populations to determine incidence of diseases, common factors, trends, and patterns. To preserve privacy, researchers often deal with anonymized records, records from which identifying information has been removed. If those records can be reconnected to the identifying information, privacy suffers. If, for example, names have been removed from records but telephone numbers remain, a researcher can use a different database of telephone numbers to determine the patient, or at least the name assigned to the telephone. Removing enough information to prevent identification is difficult and can also limit the research possibilities.

14.5 Summary

- To summarize, some points about privacy:
  - Privacy is controlled disclosure: The subject chooses what personal data to give out and to whom.
  - After disclosing something, a subject relinquishes much control to the receiver.
  - What data are sensitive is at the discretion of the subject; people consider different things sensitive.
  - Why a person considers something sensitive is less important than that it is.
  - Individuals, informal groups, and formal organizations all have things they consider private.
  - Privacy has a cost; choosing not to give out certain data may limit other benefits.
- The first step in establishing privacy is the same as the other areas of computer security: We must first define a privacy *policy* that documents what privacy we require. The early work by Ware's committee laid out very important fundamental principles of information privacy.
- Identification and authentication are two different activities that are easy to confuse. Part of the confusion arises because people do not clearly distinguish the underlying concepts. The confusion is also the result of using one data item for more than one purpose.
- Authentication depends on something that confirms a property. In life few sound authenticators exist, so we tend to overuse those we do have: an identification number, birth date, or family name. But, as we described, those authenticators are also used as database keys, with negative consequences to privacy.
- We have also studied cases in which we do not want to be identified. Anonymity and pseudonymity are useful in certain contexts. But data collection and correlation, on a scale made possible only with computers, can defeat anonymity and pseudonymity. As we computer professionals introduce new computer capabilities, we need to encourage a public debate on the related privacy issues.

14.6 Review Questions

a) What are the aspects of information privacy?
b) Write a short note on computer related privacy problems.
c) Explain fair information policies in detail.
d) What are the controls placed on commercial web sites? Explain with suitable examples.
e) Explain anonymity and pseudonymity.

f) What are data access risks?
g) What are the steps to protect against privacy loss?
h) What does authentication mean?
i) Explain individual authentication and identity authentication.

14.7 Bibliography, References and Further Reading

- *Security in Computing by* C. P. Pfleeger, and S. L. Pfleeger, Pearson Education.
- *Computer Security: Art and Science by* Matt Bishop, Pearson Education.
- *Cryptography And Network Security: Principles and practice by* Stallings
- *Network Security by* Kaufman, Perlman, Speciner
- *Network Security : A Beginner's Guide by* Eric Maiwald, TMH
- *Java Network Security by* Macro Pistoia, Pearson Education
- *Principles of information security by* Whitman, Mattord, Thomson

IDOL

Chapter 15

Legal and Ethical Issues in Computer Security

IDOL

15.0 Objectives

At the end of this chapter, you will understand

- Program and data protection by patents, copyrights, and trademarks
- Computer crime
- Ethical analysis of computer security situations
- Codes of professional ethics

15.1 Introduction

In this chapter we study human controls applicable to computer security: the legal system and ethics. The legal system has adapted quite well to computer technology by reusing some old forms of legal protection (copyrights and patents) and creating laws where no adequate ones existed (malicious access). Still, the courts are not a perfect form of protection for computer resources, for two reasons. First, the courts tend to be reactive instead of proactive. That is, we have to wait for a transgression to occur and then adjudicate it, rather than try to prevent it in the first place. Second, fixing a problem through the courts can be time consuming (sometimes taking years) and expensive; the latter characteristic prevents all but the wealthy from addressing most security issues.

15.2 Protecting Programs and Data

Copyrights, patents, and trade secrets are legal devices that can protect computers, programs, and data. However, in some cases, precise steps must be taken to protect the work before anyone else is allowed access to it. In this section, we explain how each of these forms of protection was originally designed to be used and how each is currently used in computing. We focus primarily on U.S. law, to provide examples of intent and consequence.

15.2.1. Copyrights

In the United States, the basis of copyright protection is presented in the U.S. Constitution. The body of legislation supporting constitutional provisions contains laws that elaborate on or expand the constitutional protections. Relevant statutes include the U.S. copyright law of 1978, which was updated in 1998 as the Digital Millennium Copyright Act (DMCA) specifically to deal with computers and other electronic media such as digital video and music. The 1998 changes brought U.S. copyright law into general conformance with the World Intellectual Property Organization treaty of 1996, an international copyright standard to which 95 countries adhere.

Copyrights are designed to protect the expression of ideas. Thus, a copyright applies to a creative

work, such as a story, photograph, song, or pencil sketch. The right to copy an *expression* of an idea is protected by a copyright. Ideas themselves, the law alleges, are free; anyone with a bright mind can think up anything anyone else can, at least in theory. The intention of a copyright is to allow regular and free exchange of ideas. The law protects an individual's right to earn a living, while recognizing that exchanging ideas supports the intellectual growth of society. The copyright says that a particular *way* of expressing an idea belongs to the author. For example, in music, there may be two or three copyrights related to a single creation: A composer can copyright a song, an arranger can copyright an arrangement of that song, and an artist can copyright a specific performance of that arrangement of that song. The price you pay for a ticket to a concert includes compensation for all three creative expressions. Copyright gives the author the *exclusive* right to make copies of the expression and sell them to the public. That is, only the author (or booksellers or others working as the author's agents) can sell copies of the author's book.

Definition of Intellectual Property

The U.S. copyright law (§102) states that a copyright can be registered for "original works of authorship fixed in any tangible medium of expression, ... from which they can be perceived, reproduced, or otherwise communicated, either directly or with the aid of a machine or device". Only the originator of the expression is entitled to copyright; if an expression has no determinable originator, copyright cannot be granted. Certain works are considered to be in the public domain, owned by the public, by no one in particular. Works of the U.S. government and many other governments are considered to be in the public domain and therefore not subject to copyright. Finally, copyright lasts for only a limited period of time, so certain very old works, such as the plays of Shakespeare, are in the public domain, their possibility of copyright having expired. The copyrighted expression must also be in some tangible medium. A story or art work must be written, printed, painted, recorded (on a physical medium such as a plastic record), stored on a magnetic medium (such as a disk or tape), or fixed in some other way. Furthermore, the purpose of the copyright is to promote distribution of the work; therefore, the work must be distributed, even if a fee is charged for a copy.

Originality of Work

The work being copyrighted must be original to the author. As noted previously, some expressions in the public domain are not subject to copyright. A work can be copyrighted even if it contains some public domain material, as long as there is some originality, too. The author does not even have to identify what is public and what is original. For example, a music historian could copyright a collection of folksongs even if some are in the public domain. To be subject to copyright, something in or *about* the collection has to be original. The historian might argue that collecting the songs, selecting which ones to include, and putting them in order was the original part. In this case, the copyright law would not protect the folksongs (which would be in the public domain) but would instead protect that specific selection and organization.

Fair Use of Material

The copyright law indicates that the copyrighted object is subject to fair use. A purchaser has the right to use the product in the manner for which it was intended and in a way that does not interfere with the author's rights. Specifically, the law allows "fair use of a copyrighted work, including such use by reproduction in copies… for purposes such as criticism, comment, news reporting, teaching (including multiple copies for classroom use), scholarship or research." The purpose and effect of the use on the potential market for or value of the work affect the decision of what constitutes fair use. For example, fair use allows making a backup copy of copyrighted software you acquired legally: Your backup copy protects your use against system failures, but it doesn't affect the author because you have no need for nor do you want use of two copies at once. The copyright law usually upholds the author's right to a fair return for the work, while encouraging others to use the

underlying ideas. Unfair use of a copyrighted item is called piracy.
The copyright law also has the concept of a first sale: after having bought a copyrighted object, the new owner can give away or resell the object. That is, the copyright owner is entitled to control the first sale of the object. This concept works fine for books: An author is compensated when a bookstore sells a book, but the author earns no additional revenue if the book is later resold at a second-hand store.

Requirements for Registering a Copyright
The copyright is easy to obtain, and mistakes in securing a copyright can be corrected. The first step of registration is notice. Any potential user must be made aware that the work is copyrighted. Each copy must be marked with the copyright symbol ©, the word *Copyright*, the year, and the author's name. Each copy distributed must be so marked, although the law will forgive failure to mark copies if a reasonable attempt is made to recall and mark any ones distributed without a mark. The copyright must also be officially filed. In the United States a form is completed and submitted to the Copyright Office, along with a nominal fee and a copy of the work. A U.S. copyright now lasts for 70 years beyond the death of the last surviving author or, if the item was copyrighted by a company or organization, for 95 years after the date of publication. The international standard is 50 years after the death of the last author or 50 years from publication.

Copyright Infringement
The holder of the copyright must go to court to prove that someone has infringed on the copyright. The infringement must be substantial, and it must be copying, not independent work. In theory, two people might write identically the same song independently, neither knowing the other. These two people would *both* be entitled to copyright protection for their work. Neither would have infringed on the other, and both would have the right to distribute their work for a fee. Again, copyright is most easily understood for written works of fiction because it is extremely unlikely that two people would express an idea with the same or similar wording. However, it is far less likely that two textbook authors would have the same pattern of presentation and the same examples from beginning to end.

Copyrights for Computer Software
The original copyright law envisioned protection for things such as books, songs, and photographs. People can rather easily detect when these items are copied. The separation between public domain and creativity is fairly clear. And the distinction between an idea (feeling, emotion) and its expression is pretty obvious. Works of nonfiction understandably have less leeway for independent expression. Because of programming language constraints and speed and size efficiency, computer programs have less leeway still. Can a computer program be copyrighted? Yes. The 1976 copyright law was amended in 1980 to include an explicit definition of computer software. However, copyright protection may not be an especially desirable form of protection for computer works.
To see why, consider the algorithm used in a given program. The algorithm is the idea, and the statements of the programming language are the expression of the idea. Therefore, protection is allowed for the program statements themselves, but not for the algorithmic concept: copying the code intact is prohibited, but reimplementing the algorithm is permitted. Remember that one purpose of copyright is to promote the dissemination of ideas. The algorithm, which is the idea embodied in the computer program, is to be shared. A second problem with copyright protection for computer works is the requirement that the work be published. A program may be published by distribution of copies of its object code, for example, on a disk. However, if the source code is not distributed, it has not been published. An alleged infringer cannot have violated a copyright on source code if the source code was never published.

Copyrights for Digital Objects

The Digital Millennium Copyright Act (DMCA) of 1998 clarified some issues of digital objects (such as music files, graphics images, data in a database, and also computer programs), but it left others unclear. Among the provisions of the DMCA are these:

- Digital objects *can be* subject to copyright.
- It is a crime to circumvent or disable antipiracy functionality built into an object.
- It is a crime to manufacture, sell, or distribute devices that disable antipiracy functionality or that copy digital objects.
- However, these devices can be used (and manufactured, sold, or distributed) for research and educational purposes.
- It is acceptable to make a backup copy of a digital object as a protection against hardware or software failure or to store copies in an archive.
- Libraries can make up to three copies of a digital object for lending to other libraries.

So, a user can make reasonable copies of an object in the normal course of its use and as a protection against system failures. If a system is regularly backed up and so a digital object (such as a software program) is copied onto many backups, that is not a violation of copyright.
The uncertainty comes in deciding what is considered to be a device to counter piracy. A disassembler or decompiler could support piracy or could be used to study and enhance a program. Someone who decompiles an executable program, studies it to infer its method, and then modifies, compiles, and sells the result is misusing the decompiler. But the distinction is hard to enforce, in part because the usage depends on intent and context. Reaction to the Digital Millennium Copyright Act has not been uniformly favourable. Some say it limits computer security research. Worse, others point out it can be used to prevent exactly the free interchange of ideas that copyright was intended to promote. In 2001 a Princeton University professor, Edward Felten, and students presented a paper on cryptanalysis of the digital watermarking techniques used to protect digital music files from being copied. They had been pressured not to present in the preceding April by music industry groups who threatened legal action under the DMCA. Digital objects are more problematic than paper ones because they can be copied exactly. Unlike fifth-generation photocopies, each digital copy of a digital object can be identical to the original. An emerging principle is that software, like music, is acquired in a style more like rental than purchase. You purchase not a piece of software, but the right to use it. Clarifying this position, the U.S. No Electronic Theft (NET) Act of 1997 makes it a criminal offense to reproduce or distribute copyrighted works, such as software or digital recordings, even without charge. The area of copyright protection applied to computer works continues to evolve and is subject to much interpretation by the courts. Therefore, it is not certain what aspects of a computer work are subject to copyright. Although copyright protection can be applied to computer works, the copyright concept was conceived before the electronic age, and thus the protection may be less than what we desire. Copyrights do not address all the critical computing system elements that require protection.

15.2.2. Patents

Patents are unlike copyrights in that they protect inventions, tangible objects, or ways to make them, not works of the mind. The distinction between patents and copyrights is that patents were intended to apply to the results of science, technology, and engineering, whereas copyrights were meant to cover works in the arts, literature, and written scholarship. A patent can protect a “new and useful process, machine, manufacture, or composition of matter”. The U.S. law excludes “newly discovered laws of nature… [and] mental processes”. A patent is designed to protect the device or process for *carrying out* an idea, not the idea itself.

Requirement of Novelty

If two composers happen to compose the same song independently at different times, copyright law

would allow both of them to have copyright. If two inventors devise the same invention, the patent goes to the person who invented it first, regardless of who first filed the patent. A patent can be valid only for something that is truly novel or unique, so there can be only one patent for a given invention. An object patented must also be nonobvious. If an invention would be obvious to a person ordinarily skilled in the field, it cannot be patented. The law states that a patent *cannot* be obtained "if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains".

Procedure for Registering a Patent

One registers a copyright by filing a brief form, marking a copyright notice on the creative work, and distributing the work. The whole process takes less than an hour.

To obtain a patent, an inventor must convince the U.S. Patent and Trademark Office that the invention deserves a patent. For a fee, a patent attorney will research the patents already issued for similar inventions. This search accomplishes two things. First, it determines that the invention to be patented has not already been patented (and, presumably, has not been previously invented). Second, the search can help identify similar things that have been patented. These similarities can be useful when describing the unique features of the invention that make it worthy of patent protection. The Patent Office compares an application to those of all other similar patented inventions and decides whether the application covers something truly novel and nonobvious. If the office decides the invention is novel, a patent is granted.

Typically, an inventor writes a patent application listing many claims of originality, from very general to very specific. The Patent Office may disallow some of the more general claims while upholding some of the more specific ones. The patent is valid for all the upheld claims. The patent applicant reveals what is novel about the invention in sufficient detail to allow the Patent Office and the courts to judge novelty; that degree of detail may also tell the world how the invention works, thereby opening the possibility of infringement. The patent owner uses the patented invention by producing products or by licensing others to produce them. Patented objects are sometimes marked with a patent number to warn others that the technology is patented. The patent holder hopes this warning will prevent others from infringing.

Patent Infringement

A patent holder *must* oppose all infringement. With a copyright, the holder can choose which cases to prosecute, ignoring small infringements and waiting for serious infractions where the infringement is great enough to ensure success in court or to justify the cost of the court case. However, failing to sue a patent infringement even a small one or one the patent holder does not know about can mean losing the patent rights entirely. But, unlike copyright infringement, a patent holder does not have to prove that the infringer copied the invention; a patent infringement occurs even if someone independently invents the same thing, without knowledge of the patented invention. Every infringement must be prosecuted. Prosecution is expensive and time consuming, but even worse, suing for patent infringement could cause the patent *holder* to lose the patent. Someone charged with infringement can argue all of the following points as a defense against the charge of infringement.

- *This isn't infringement:* The alleged infringer will claim that the two inventions are sufficiently different that no infringement occurred.
- *The patent is invalid:* If a prior infringement was not opposed, the patent rights may no longer be valid.
- *The invention is not novel:* In this case, the supposed infringer will try to persuade the judge that the Patent Office acted incorrectly in granting a patent and that the invention is nothing worthy of patent.
- *The infringer invented the object first:* If so, the accused infringer, and not the original patent

holder, is entitled to the patent.

The first defense does not damage a patent, although it can limit the novelty of the invention. However, the other three defenses can destroy patent rights. Worse, all four defenses can be used every time a patent holder sues someone for infringement. Finally, obtaining and defending a patent can incur substantial legal fees. Patent protection is most appropriate for large companies with substantial research and development (and legal) staffs.

Applicability of Patents to Computer Objects

The Patent Office has not encouraged patents of computer software. For a long time, computer programs were seen as the representation of an algorithm, and an algorithm was a fact of nature, which is not subject to patent. An early software patent case, *Gottschalk v. Benson*, involved a request to patent a process for converting decimal numbers into binary. The Supreme Court rejected the claim, saying it seemed to attempt to patent an abstract idea, in short, an algorithm. But the underlying algorithm is precisely what most software developers would like to protect.

In 1981, two cases (*Diamond v. Bradley* and *Diamond v. Diehr*) won patents for a process that used computer software, a well-known algorithm, temperature sensors, and a computer to calculate the time to cure rubber seals. The court upheld the right to a patent because the claim was not for the software or the algorithm alone, but for the process that happened to use the software as one of its steps. An unfortunate inference is that using the software without using the other patented steps of the process would not be infringement.

Since 1981 the patent law has expanded to include computer software, recognizing that algorithms, like processes and formulas, are inventions. The Patent Office has issued thousands of software patents since these cases. But because of the time and expense involved in obtaining and maintaining a patent, this form of protection may be unacceptable for a small-scale software writer.

15.2.3. Trade Secrets

A trade secret is unlike a patent or copyright in that it must be kept a *secret*. The information has value only as a secret, and an infringer is one who divulges the secret. Once divulged, the information usually cannot be made secret again.

Characteristics of Trade Secrets

A trade secret is information that gives one company a competitive edge over others. For example, the formula for a soft drink is a trade secret, as is a mailing list of customers or information about a product due to be announced in a few months. The distinguishing characteristic of a trade secret is that it must always be kept secret. Employees and outsiders who have access to the secret must be required not to divulge the secret. The owner must take precautions to protect the secret, such as storing it in a safe, encrypting it in a computer file, or making employees sign a statement that they will not disclose the secret.

If someone obtains a trade secret improperly and profits from it, the owner can recover profits, damages, lost revenues, and legal costs. The court will do whatever it can to return the holder to the same competitive position it had while the information was secret and may award damages to compensate for lost sales. However, trade secret protection evaporates in case of independent discovery. If someone else happens to discover the secret independently, there is no infringement and trade secret rights are gone.

Reverse Engineering

Another way trade secret protection can vanish is by reverse engineering. Suppose a secret is the way to pack tissues in a cardboard box to make one pop up as another is pulled out. Anyone can cut open the box and study the process. Therefore, the trade secret is easily discovered. In reverse

IDOL

engineering, one studies a finished object to determine how it is manufactured or how it works. Through reverse engineering someone might discover how a telephone is built; the design of the telephone is obvious from the components and how they are connected. Therefore, a patent is the appropriate way to protect an invention such as a telephone. However, something like a soft drink is not just the combination of its ingredients. Making a soft drink may involve time, temperature, presence of oxygen or other gases, and similar factors that could not be learned from a straight chemical decomposition of the product. The recipe of a soft drink is a closely guarded trade secret. Trade secret protection works best when the secret is not apparent in the product.

Applicability to Computer Objects

Trade secret protection applies very well to computer software. The underlying algorithm of a computer program is novel, but its novelty depends on nobody else's knowing it. Trade secret protection allows distribution of the *result* of a secret (the executable program) while still keeping the program design hidden. Trade secret protection does not cover copying a product (specifically a computer program), so it cannot protect against a pirate who sells copies of someone else's program without permission. However, trade secret protection makes it illegal to steal a secret algorithm and use it in another product. The difficulty with computer programs is that reverse engineering works. Decompiler and disassembler programs can produce a source version of an executable program. Of course, this source does not contain the descriptive variable names or the comments to explain the code, but it is an accurate version that someone else can study, reuse, or extend.

Difficulty of Enforcement

Trade secret protection is of no help when someone infers a program's design by studying its output or, worse yet, decoding the object code. Both of these are legitimate (that is, legal) activities, and both cause trade secret protection to disappear. The confidentiality of a trade secret must be ensured with adequate safeguards. If source code is distributed loosely or if the owner fails to impress on people (such as employees) the importance of keeping the secret, any prosecution of infringement will be weakened. Employment contracts typically include a clause stating that the employee will not divulge any trade secrets received from the company, even after leaving a job. Additional protection, such as marking copies of sensitive documents or controlling access to computer files of secret information, may be necessary to impress people with the importance of secrecy.

15.2.4. Protection for Computer Objects

The previous sections have described three forms of protection: the copyright, patent, and trade secret laws. Each of these provides a different form of protection to sensitive things. In this section we consider different kinds of computer objects and describe which forms of protection are most appropriate for each kind. Table 15-1 shows how these three forms of protection compare in several significant ways.

Table 15-1 Comparing Copyright, Patent, and Trade Secret Protection

IDOL

	Copyright	Patent	Trade Secret
Protects	Expression of idea, not idea itself	Invention—the way something works	A secret, competitive advantage
Protected object made public	Yes; intention is to promote publication	Design filed at Patent Office	No
Requirement to distribute	Yes	No	No
Ease of filing	Very easy, do-it-yourself	Very complicated; specialist lawyer suggested	No filing
Duration	Varies by country; approximately 75–100 years is typical	19 years	Indefinite
Legal protection	Sue if unauthorized copy sold	Sue if invention copied	Sue if secret improperly obtained

Computer artifacts are new and constantly changing, and they are not yet fully appreciated by the legal system based on centuries of precedent. Perhaps in a few years the issue of what protection is most appropriate for a given computer object will be more clear-cut. Possibly a new form of protection or a new use of an old form will apply specifically to computer objects. Until the law provides protection that truly fits computer goods, here are some guidelines for using the law to protect computer objects.

Protecting Hardware

Hardware, such as chips, disk drives, or floppy disk media, can be patented. The medium itself can be patented, and someone who invents a new process for manufacturing it can obtain a second patent.

Protecting Firmware

The situation is a little less clear with regard to microcode. Certainly, the physical devices on which microcode is stored can be patented. Also, a special-purpose chip that can do only one specific task can probably be patented. However, the data (instructions, algorithms, microcode, programs) contained in the devices are probably not patentable. Can they be copyrighted? Are these the expression of an idea in a form that promotes dissemination of the idea? Probably not. And assuming that these devices were copyrighted, what would be the definition of a copy that infringed on the copyright? Worse, would the manufacturer really want to register a copy of the internal algorithm with the Copyright Office? Copyright protection is probably inappropriate for computer firmware.

Trade secret protection seems appropriate for the code embedded in a chip. Given enough time, we can reverse-engineer and infer the code from the behaviour of the chip. The behaviour of the chip does not reveal what algorithm is used to produce that behaviour. The original algorithm may have better (or worse) performance (speed, size, fault tolerance) that would not be obvious from reverse engineering. The courts have affirmed that computer software *is* an appropriate subject for copyright protection and that protection should be no less valid when the software is in a chip rather than in a conventional program.

Protecting Object Code Software

Object code is usually copied so that it can be distributed for profit. The code is a work of creativity,

and most people agree that object code distribution is an acceptable medium of publication. Thus, copyright protection seems appropriate. A copyright application is usually accompanied by a copy of the object being protected. With a book or piece of music (printed or recorded), it is easy to provide a copy. The Copyright Office has not yet decided what is an appropriate medium in which to accept object code. A binary listing of the object code will be taken, but the Copyright Office does so without acknowledging the listing to be acceptable or sufficient. The Office will accept a source code listing. Some people argue that a source code listing is not equivalent to an object code listing, in the same way that a French translation of a novel is different from its original language version. It is not clear *in the courts* that registering a source code version provides copyright protection to object code. However, someone should not be able to take the object code of a system, rearrange the order of the individual routines, and say that the result is a new system. Without the original source listings, it would be very difficult to compare two binary files and determine that one was the functional equivalent of the other simply through rearrangement.

Protecting Source Code Software

Software developers selling to the mass market are reticent to distribute their source code. The code can be treated as a trade secret, although some lawyers also encourage that it be copyrighted. A copyright protects the right to distribute copies of the *expression* of an idea, not the idea itself. Therefore, a copyright does not prevent someone from reimplementing an algorithm, expressed through a copyrighted computer program. As just described, source code may be the most appropriate form in which to register a copyright for a program distributed in object form. It is difficult to register source code with the Copyright Office while still ensuring its secrecy.

Protecting Documentation

If we think of documentation as a written work of nonfiction (or, perhaps, fiction), copyright protection is effective and appropriate for it. Notice that the documentation is distinct from the program. A program and its documentation must be copyrighted separately. Furthermore, copyright protection of the documentation may win a judgment against someone who illegally copies both a program and its documentation. In cases where a written law is unclear or is not obviously applicable to a situation, the results of court cases serve to clarify or even extend the words of the law. As more unfair acts involving computer works are perpetrated, lawyers will argue for expanded interpretations of the law. Thus, the meaning and use of the law will continue to evolve through judges' rulings. In a sense, computer technology has advanced much faster than the law has been able to.

Protecting Web Content

Content on the web is media, much the same as a book or photograph, so the most appropriate protection for it is copyright. This copyright would also protect software you write to animate or otherwise affect the display of your web page. And, in theory, if your web page contains malicious code, your copyright covers that, too. As we discussed earlier, a copyrighted work does not have to be exclusively new; it can be a mixture of new work to which you claim copyright and old things to which you do not. You may purchase or use with permission a piece of web art, a widget (such as an applet that shows a spinning globe), or some music. Copyright protects your original works. Protecting Domain Names and URLs Domain names, URLs, company names, product names, and commercial symbols are protected by a trademark, which gives exclusive rights of use to the owner of such identifying marks.

15.3 Information and the Law

Source code, object code, and even the "look and feel" of a computer screen are recognizable, if not tangible, objects. The law deals reasonably well, although somewhat belatedly, with these things.

But computing is in transition to a new class of object, with new legal protection requirements. Electronic commerce, electronic publishing, electronic voting, electronic banking, these are the new challenges to the legal system. In this section we consider some of these new security requirements.

15.3.1. Information as an Object

The shopkeeper used to stock "things" in the store, such as buttons, automobiles, and pounds of sugar. The buyers were customers. When a thing was sold to a customer, the shopkeeper's stock of that thing was reduced by one, and the customer paid for and left with a thing. Sometimes the customer could resell the thing to someone else, for more or less than the customer originally paid. Other kinds of shops provided services that could be identified as things, for example, a haircut, root canal, or defense for a trial. The value of a service in a free economy was somehow related to its desirability to the buyer and the seller. But today we must consider a third category for sale: information. No one would argue against the proposition that information is valuable. Students are tempted to pay others for answers during examinations, and businesses pay for credit reports, client lists, and marketing advice. But information does not fit the familiar commercial paradigms with which we have dealt for many years. Let us examine why information is different from other commercial things.

Information Is Not Depletable

Unlike tangible things and services, information can be sold again and again without depleting stock or diminishing quality. For example, a credit bureau can sell the same credit report on an individual to an unlimited number of requesting clients. Each client pays for the information in the report. The report may be delivered on some tangible medium, such as paper, but it is the *information*, not the medium, that has the value.

Information Can Be Replicated

The value of information is what the buyer will pay the seller. But after having bought the information, the buyer can then become a seller and can potentially deprive the original seller of further sales. Because information is not depletable, the buyer can enjoy or use the information and can also sell it many times over, perhaps even making a profit.

Information Has a Minimal Marginal Cost

The marginal cost of an item is the cost to produce another one after having produced some already. If a newspaper sold only one copy on a particular day, that one issue would be prohibitively expensive because it would have to cover the day's cost (salary and benefits) of all the writers, editors, and production staff, as well as a share of the cost of all equipment for its production. These are fixed costs needed to produce a first copy. With this model, the cost of the second and subsequent copies is minuscule, representing basically just the cost of paper and ink to print them. Fortunately, newspapers have very large press runs and daily sales, so the fixed costs are spread evenly across a large number of copies printed. The cost of information similarly depends on fixed costs plus costs to reproduce. Typically, the fixed costs are large whereas the cost to reproduce is extremely small, even less than for a newspaper because there is no cost for the raw materials of paper and ink. However, unlike a newspaper, information is far more feasible for a buyer to resell. A copy of digital information can be perfect, indistinguishable from the original, the same being true for copies of copies of copies of copies.

The Value of Information Is Often Time Dependent

If you knew for certain what the trading price of a share of Microsoft stock would be next week, that information would be extremely valuable because you could make an enormous profit on the

stock market. Of course, that price cannot be known today. But suppose you knew that Microsoft was certain to announce something next week that would cause the price to rise or fall. That information would be almost as valuable as knowing the exact price, and it could be known in advance. However, knowing *yesterday's* price for Microsoft stock or knowing that *yesterday* Microsoft announced something that caused the stock price to plummet is almost worthless because it is printed in every major financial newspaper. Thus, the value of information may depend on when you know it.

Information Is Often Transferred Intangibly

A newspaper is a printed artifact. The news agent hands it to a customer, who walks away with it. Both the seller and the buyer realize and acknowledge that something has been acquired. Furthermore, it is evident if the newspaper is seriously damaged; if a serious production flaw appears in the middle, the defect is easy to point out. But times are changing. Increasingly, information is being delivered as bits across a network instead of being printed on paper. If the bits are visibly flawed (that is, if an error detecting code indicates a transmission error), demonstrating that flaw is easy. However, if the copy of the information is accurate but the underlying information is incorrect, useless, or not as expected, it is difficult to justify a claim that the information is flawed.

15.3.2. Legal Issues Relating to Information

These characteristics of information significantly affect its legal treatment. If we want to understand how information relates to copyright, patent, and trademark laws, we must understand these attributes. We can note first that information has some, limited legal basis for the protection. For example, information can be related to trade secrets, in that information is the stock in trade of the information seller. While the seller has the information, trade secret protection applies naturally to the seller's legitimate ability to profit from information. Thus, the courts recognize that information has value. Other forms of protection are offered by copyrights and patents. As we have seen earlier, neither of these applies perfectly to computer hardware or software, and they apply even less well to information. The pace of change in the legal system is slow, helping to ensure that the changes that do occur are fair and well considered. The deliberate pace of change in the legal system is about to be hit by the supersonic rate of change in the information technology industry. Laws do not, and cannot, control all cyber threats. Let us look at several examples of situations in which information needs are about to place significant demands on the legal system.

Information Commerce

Information is unlike most other goods traded, even though it has value and is the basis of some forms of commerce. The market for information is still young, and so far, the legal community has experienced few problems. Nevertheless, several key issues must be resolved. For example, we have seen that software piracy involves copying information without offering adequate payment to those who deserve to be paid. Several approaches have been tried to ensure that the software developer or publisher receives just compensation for use of the software: copy protection, freeware, and controlled distribution. More recently, software is being delivered as mobile code or applets, supplied electronically as needed. The applet approach gives the author and distributor more control. Each applet can potentially be tracked and charged for, and each applet can destroy itself after use so that nothing remains to be passed for free to someone else. But this scheme requires a great deal of accounting and tracking, increasing the costs of what might otherwise be reasonably priced.

Electronic Publishing

Many newspapers and magazines post a version of their content on the Internet, as do wire services and television news organizations. For example, the British Broadcasting Company (BBC) and the Reuters news services have a significant web presence. We should expect that some news and information will eventually be published and distributed exclusively on the Internet. Indeed, encyclopedias such as the Britannica and Expedia are mainly web-based services now, rather than being delivered as the large number of book volumes they used to occupy. Here again the publisher has a problem ensuring that it receives fair compensation for the work. Cryptography-based technical solutions are under development to address this problem. However, these technical solutions must be supported by a legal structure to enforce their use.

Protecting Data in a Database

Databases are a particular form of software that has posed significant problems for legal interpretation. The courts have had difficulty deciding which protection laws apply to databases. How does one determine that a set of data came from a particular database (so that the database owner can claim some compensation)? Who even owns the data in a database if it is public data, such as names and addresses?

Electronic Commerce

Laws related to trade in goods have evolved literally over centuries. Adequate legal protections exist to cover defective goods, fraudulent payment, and failure to deliver when the goods are tangible and are bought through traditional outlets such as stores and catalogues. However, the situation becomes less clear when the goods are traded electronically. If you order goods electronically, digital signatures and other cryptographic protocols can provide a technical protection for your "money." However, suppose the information you order is not suitable for use or never arrives or arrives damaged or arrives too late to use. How do you prove conditions of the delivery? For catalogue sales, you often have receipts or some paper form of acknowledgment of time, date, and location. But for digital sales, such verification may not exist or can be easily modified. These legal issues must be resolved as we move into an age of electronic commerce.

15.3.3. Protecting Information

Clearly, current laws are inadequate for protecting the information itself and for protecting electronically based forms of commerce. So how is information to be protected legally? As described, copyrights, patents, and trade secrets cover some, but not all, issues related to information. Nevertheless, the legal system does not allow free traffic in information; some mechanisms can be useful.

Criminal and Civil Law

Statutes are laws that state explicitly that certain actions are illegal. A statute is the result of a legislative process by which a governing body declares that the new law will be in force after a designated time. Often, a violation of a statute will result in a criminal trial, in which the government argues for punishment because an illegal act has harmed the desired nature of society. The goal of a criminal case is to punish the criminal, usually by depriving him or her of rights in some way (such as putting the criminal in prison or assessing a fine).

Civil law is a different type of law, not requiring such a high standard of proof of guilt. In a civil case, an individual, organization, company, or group claims it has been harmed. The goal of a civil case is restitution: to make the victim "whole" again by repairing the harm.

Tort Law

Special legal language describes the wrongs treated in a civil case. The language reflects whether a

case is based on breaking a law or on violating precedents of behaviour that have evolved over time. In other words, sometimes judges may make determinations based on what is reasonable and what has come before, rather than on what is written in legislation. A tort is harm not occurring from violation of a statute or from breach of a contract but instead from being counter to the accumulated body of precedents. Thus, statute law is written by legislators and is interpreted by the courts; tort law is unwritten but evolves through court decisions that become precedents for cases that follow. The basic test of a tort is what a reasonable person would do. Computer information is perfectly suited to tort law. The court merely has to decide what is reasonable behaviour, not whether a statute covers the activity. Because tort law is written only as a series of court decisions that evolve constantly, prosecution of a tort case can be difficult. If you are involved in a case based on tort law, you and your lawyer are likely to try two approaches: First, you might argue that your case is a clear violation of the norms of society, that it is not what a fair, prudent person would do. This approach could establish a new tort. Second, you might argue that your case is similar to one or more precedents, perhaps drawing a parallel between a computer program and a work of art. The judge or jury would have to decide whether the comparison was apt. In both of these ways, law can evolve to cover new objects.

Contract Law

A third form of protection for computer objects is contracts. A contract is an agreement between two parties. A contract must involve three things:

- an offer
- an acceptance
- a consideration

A contract must include consideration of money or other valuables. The basic idea is that two parties exchange things of value, such as time traded for money or technical knowledge for marketing skills. A written contract can involve hundreds of pages of terms and conditions qualifying the offer and the consideration. One final aspect of a contract is its freedom: the two parties have to enter into the contract voluntarily. A contract signed under duress or with fraudulent action is not binding. A contract does not have to be fair, in the sense of equivalent consideration for both parties, as long as both parties freely accept the conditions.

Information is often exchanged under contract. Contracts are ideal for protecting the transfer of information because they can specify any conditions. "You have the right to use but not modify this information," "you have the right to use but not resell this information," or "you have the right to view this information yourself but not allow others to view it" are three potential contract conditions that could protect the commercial interests of an owner of information. Computer contracts typically involve the development and use of software and computerized data. As with tort law, the most common legal remedy in contract law is money.

15.4 Rights of Employees and Employers

Employers hire employees to generate ideas and make products. The protection offered by copyrights, patents, and trade secrets appeals to employers because it applies to the ideas and products. However, the issue of who owns the ideas and products is complex. Ownership is a computer security concern because it relates to the rights of an employer to protect the secrecy and integrity of works produced by the employees. In this section we study the respective rights of employers and employees to their computer products.

15.4.1. Ownership of Products

There are many different situations and interpreting the laws of ownership is difficult. Let us

consider each type of protection in turn.

Ownership of a Patent
The person who owns a work under patent or copyright law is the inventor. Under patent law, it is important to know who files the patent application. If an employee lets an employer patent an invention, the employer is deemed to own the patent and therefore the rights to the invention. The employer also has the right to patent if the employee's job functions included inventing the product. For instance, in a large company a scientist may be hired to do research and development, and the results of this inventive work become the property of the employer. Even if an employee patents something, the employer can argue for a right to use the invention if the employer contributed some resources in developing the invention.

Ownership of a Copyright
Owning a copyright is similar to owning a patent. The author (programmer) is the presumed owner of the work, and the owner has all rights to an object. However, a special situation known as *work for hire* applies to many copyrights for developing software or other products.

Work for Hire
In a work for hire situation, the employer, *not* the employee, is considered the author of a work. Work for hire is not easy to identify and depends in part on the laws of the state in which the employment occurs. The relationship between an employee and employer is considered a work for hire if some or all of the following conditions are true:

- The employer has a supervisory relationship, overseeing the manner in which the creative work is done.
- The employer has the right to fire the employee.
- The employer arranges for the work to be done before the work was created.
- A written contract between the employer and employee states that the employer has hired the employee to do certain work.

Licenses
An alternative to a work for hire arrangement is licensed software. In this situation, the programmer develops and retains full ownership of the software. In return for a fee, the programmer grants to a company a license to use the program. The license can be granted for a definite or unlimited period of time, for one copy or for an unlimited number, to use at one location or many, to use on one machine or all, at specified or unlimited times. This arrangement is highly advantageous to the programmer, just as a work for hire arrangement is highly advantageous to the employer. The choice between work for hire and license is largely what the two parties will agree to.

Trade Secret Protection
A trade secret is different from either a patent or a copyright in that there is no registered inventor or author; there is no registration office for trade secrets. In the event a trade secret is revealed, the owner can prosecute the revealer for damages suffered. But first, ownership must be established because only the owner can be harmed. A company owns the trade secrets of its business-confidential data. As soon as a secret is developed, the company becomes the owner. As with copyrights, an employer may argue about having contributed to the development of trade secrets. If your trade secret is an improved sorting algorithm and part of your job involves investigating and testing sorting algorithms, your employer will probably claim at least partial ownership of the algorithm you try to market.

Employment Contracts
An employment contract often spells out rights of ownership. But sometimes the software developer

and possible employer have no contract. Having a contract is desirable both for employees and employers so that both will understand their rights and responsibilities. Typically, an employment contract specifies that the employee be hired to work as a programmer exclusively for the benefit of the company. The company states that this is a work for hire situation. The company claims all rights to any programs developed, including all copyright rights and the right to market. The contract may further state that the employee is receiving access to certain trade secrets as a part of employment, and the employee agrees not to reveal those secrets to anyone. An agreement not to compete is sometimes included in a contract. The employee states that simply having worked for one employer will make the employee very valuable to a competitor. The employee agrees not to compete by working in the same field for a set period of time after termination. Agreements not to compete are not always enforceable in law; in some states the employee's right to earn a living takes precedence over the employer's rights.

15.5 Redress for Software Failures

So far, we have considered programs, algorithms, and data as objects of ownership. But these objects vary in quality, and some of the legal issues involved with them concern the degree to which they function properly or well. In fact, people have legitimate differences of opinion on what constitutes "fair," "good," and "prudent" as these terms relate to computer software and programmers and vendors. The law applies most easily when there is broad consensus.

Program development is a human process of design, creation, and testing, involving a great deal of communication and interaction. For these reasons, there will always be errors in the software we produce. We sometimes expect perfect consumer products, such as automobiles or lawn mowers. At other times, we expect products to be "good enough" for use, in that most instances will be acceptable. But the situation with software is very different. To be fair, an operating system is a great deal more complex than many consumer products, and more opportunities for failure exist. For this reason, this section addresses three questions:

- What are the legal issues in selling correct and usable software?
- What are the moral or ethical issues in producing correct and usable software?
- What are the moral or ethical issues in finding, reporting, publicizing, and fixing flaws?

In some ways, the legal issues are evolving. Everyone acknowledges that all vendors *should* produce good software, but that does not always happen. The more difficult concerns arise in the development and maintenance communities about what to do when faults are discovered.

15.5.1. Selling Correct Software

Software is a product. It is built with a purpose and an audience in mind, and it is purchased by a consumer with an intended use in an expected context. And the consumer has some expectations of a reasonable level of quality and function. In that sense, buying software is like buying a radio. If you buy a faulty radio, you have certain legal rights relating to your purchase and you can enforce them in court if necessary. You may have three reactions if you find something wrong with the radio: You want your money back, you want a different (not faulty) radio, or you want someone to fix your radio. With software you have the same three possibilities, and we consider each one in turn. To consider our alternatives with software, we must first investigate the nature of the faulty code. Why was the software bad? One possibility is that it was presented on a defective medium. The second possibility is that the software worked properly, but you don't like it when you try it out. It may not do all it was advertised to do. Or you don't like the "look and feel," or it is slower than you expected it to be. The bottom line is that there is some attribute of the software that disappoints you, and you do not want this software. The final possibility is that the software malfunctions, so you cannot use it with your computer system. Here, too, you do not want the software and hope to

return it.

I Want a Refund

If the item were a radio, you would have the opportunity to look at it and listen to it in the shop, to assess its sound quality, measure its size (if it is to fit in a particular space), and inspect it for flaws. Do you have that opportunity with a program? Probably not. The U.S. Uniform Commercial Code (UCC) governs transactions between buyers and sellers in the United States. Section 2-601 says that "if the goods or the tender of delivery fail in any respect to conform to the contract, the buyer may reject them." You may have had no opportunity to try out the software before purchase, particularly on your computer. Your inspection often could not occur in the store. So, you take home the software, only to find that it is free from flaws but does not fit your needs. You are entitled to a reasonable period to inspect the software, long enough to try out its features. If you decide within a reasonably short period of time that the product is not for you, you can cite UCC §2-601 to obtain a refund. More often, though, the reason you want to return the software is because it simply is not of high enough quality. Unfortunately, correctness of software is more difficult to enforce legally.

I Want It to Be Good

Quality demands for mass market software are usually outside the range of legal enforcement for several reasons:

- Mass-market software is seldom totally bad. Certain features may not work, and faults may prevent some features from working as specified or as advertised. But the software works for most of its many users or works most of the time for all of its users.
- The manufacturer has "deep pockets." An individual suing a major manufacturer could find that the manufacturer has a permanent legal staff of dozens of full-time attorneys. The cost to the individual of bringing a suit is prohibitive.
- Legal remedies typically result in monetary awards for damages, not a mandate to fix the faulty software.
- The manufacturer has little incentive to fix small problems. Unless a problem will seriously damage a manufacturer's image or possibly leave the manufacturer open to large damage amounts, there is little justification to fix problems that affect only a small number of users or that do not render the product unfit for general use.

The "fit for use" provision of the UCC dictates that the product must be usable for its intended purpose; software that doesn't work is clearly not usable. Some manufacturers are very attentive to their customers. When flaws are discovered, the manufacturers promptly investigate the problems and fix serious ones immediately, perhaps holding smaller corrections for a later release. These companies are motivated more by public image or moral obligation than by legal requirement.

15.5.2. Reporting Software Flaws

Who should publicize flaws the user or the manufacturer? A user might want the recognition of finding a flaw; delaying the release might let someone else get that credit. A manufacturer might want to ignore a problem or fail to credit the user. And either could say the other was wrong. And how should these flaws be reported? Several different viewpoints exist.

What You Don't Know Can Hurt You

The several variants of Code Red in 2001 sparked a debate about whether we should allow full disclosure of the mechanisms that allow malicious code to enter and thrive in our systems. For example, the first variant of Code Red was relatively benign, but the third and fourth variants were powerful. When the first Code Red variant appeared, it was studied by many security analysts, including those at eEye Digital Security in Aliso Viejo, California. In an effort to pressure vendors

and software managers to take seriously the threats they represent, eEye practices full disclosure of what it knows about security flaws. However, some observers claim that such open sharing of information is precisely what enables hackers to learn about vulnerabilities and then exploit them. And many security analysts encourage users and managers to apply patches right away, closing security holes before they can be exploited. But the patches require resources and may introduce other problems while fixing the initial one. Each software-using organization must analyze and balance the risks and cost of not acting with the risks and costs of acting right away.

The Vendor's Interests

Microsoft argues that producing one patch for each discovered vulnerability is inefficient both for the vendor and the user. The vendor might prefer to bundle several patches into a single service pack or, for noncritical vulnerabilities, to hold them until the next version. So, Microsoft would like to control if or when the report of a vulnerability goes public. Scott Culp argued that "a vendor's responsibility is to its customers, not to a self-described security community." He opposed what he called "information anarchy, … the practice of deliberately publishing explicit, step-by-step instructions for exploiting security vulnerabilities without regard for how the information may be used." But he also acknowledged that the process of developing, distributing, and applying patches is imperfect, and his own company "need[s] to make it easier for users to keep their systems secure."

Users' Interests

David Litchfield, a security researcher noted for locating flaws in vendors' programs, announced in May 2002 that he would no longer automatically wait for a vendor's patch before going public with a vulnerability announcement. Citing "lethargy and an unwillingness to patch security problems as and when they are found," Litchfield criticized the approach of holding fixes of several vulnerabilities until enough had accumulated to warrant a single service pack. He makes the point that publicized or not, the vulnerabilities still exist. If one reporter has found the problem, so too could any number of malicious attackers. For a vendor to fail to provide timely patches to vulnerabilities of which the vendor is aware leaves the users wide open to attacks of which the user may be unaware. Litchfield's solution is to put pressure on the vendor. He announced he would give vendors one week's notice of a vulnerability before publicizing the vulnerability but not the details of how to exploit it to the world.

"Responsible" Vulnerability Reporting

Clearly the conflicting interests of vendors and users must meet at some compromise position. Christey and Wysopal have proposed a vulnerability reporting process that meets constraints of timeliness, fair play, and responsibility. They call the user reporting a suspected vulnerability a "reporter" and the manufacturer the "vendor." A third party such as a computer emergency response center called a "coordinator" could also play a role when a conflict or power issue arises between reporter and vendor. Basically, the process requires reporter and vendor to do the following:

- The vendor must acknowledge a vulnerability report confidentially to the reporter.
- The vendor must agree that the vulnerability exists (or argue otherwise) confidentially to the reporter.
- The vendor must inform users of the vulnerability and any available countermeasures within 30 days or request additional time from the reporter as needed.
- After informing users, the vendor may request from the reporter a 30-day quiet period to allow users time to install patches.
- At the end of the quiet period the vendor and reporter should agree upon a date at which time the vulnerability information may be released to the general public.
- The vendor should credit the reporter with having located the vulnerability.
- If the vendor does not follow these steps, the reporter should work with a coordinator to

determine a responsible way to publicize the vulnerability.

Such a proposal can only have the status of a commonly agreed-on process, since there is no authority that can enforce adherence on either users or vendors.

Quality Software

Boris Beizer, a consultant, has said, "Software should be shipped with bugs. The zero-defect notion is mythological and theoretically unachievable. That doesn't mean shipping ill-behaved or useless software; it means being open with users about the bugs we find, sending notices or including the bug list, publishing the workarounds when we have them, and being honest and open about what we have and haven't yet tested and when we do and don't plan to test in the near future." The whole debate over how and when to disclose vulnerabilities avoids the real issue. The world does not need faster patches, it needs better software with fewer vulnerabilities after delivery to the user. The issue is not how promptly a vulnerability is patched or how much detail is released with a vulnerability announcement. The issue is that, as the Anderson report noted over three decades ago, "penetrate and patch" is a fatally flawed concept: after a flaw was patched, the penetrators always found other old flaws or new flaws introduced because of or in the patch. The issue is technical, psychological, sociological, managerial, and economic. Until we produce consistently solid software, our entire computing infrastructure is seriously at risk.

15.6 Computer Crime

The law related to contracts and employment is difficult, but at least employees, objects, contracts, and owners are fairly standard entities for which legal precedents have been developed over centuries. The definitions in copyright and patent law are strained when applied to computing because old forms must be made to fit new objects; for these situations, however, cases being decided now are establishing legal precedents. But crimes involving computers are an area of the law that is even less clear than the other areas.

15.6.1. Why a Separate Category for Computer Crime is needed

Crimes can be organized into certain recognized categories, including *murder*, *robbery*, and *littering*. We do not separate crime into categories for different weapons, such as *gun crime* or *knife crime*, but we separate crime victims into categories, depending on whether they are *people* or *other objects*. Nevertheless, driving into your neighbour's picture window can be as bad as driving into his evergreen tree or pet sheep. Let us look at an example to see why these categories are not sufficient and why we need special laws relating to computers as subjects and objects of crime.

Rules of Property

The legal system has explicit rules about what constitutes property. Generally, property is tangible, unlike magnetic impulses. For example, unauthorized use of a neighbour's lawn mower constitutes theft, even if the lawn mower was returned in essentially the same condition as it was when taken. To a computer professional, taking a copy of a software package without permission is clear-cut theft. Fortunately, laws evolve to fit the times, and this interpretation from the 1980s has been refined so that bits are now recognized items of property. A similar problem arises with computer services. We would generally agree that unauthorized access to a computing system is a crime. For example, if a stranger enters your garden and walks around, even if nothing is touched or damaged, the act is considered trespassing. However, because access by computer does not involve a physical object, not all courts punish it as a serious crime.

Rules of Evidence

Computer printouts have been used as evidence in many successful prosecutions. Frequently-used are computer records generated in the ordinary course of operation, such as system audit logs. Under the rules of evidence, courts prefer an original source document to a copy, under the assumption that the copy may be inaccurate or may have been modified in the copying process. The biggest difficulty with computer-based evidence in court is being able to demonstrate the authenticity of the evidence. Law enforcement officials operate under a chain of custody requirement: From the moment a piece of evidence is taken until it is presented in court, they track clearly and completely the order and identities of the people who had personal custody of that object. With computer-based evidence, it can be difficult to establish a chain of custody. If a crime occurred on Monday but was not discovered until Wednesday, who can verify that the log file was not altered?

Threats to Integrity and Confidentiality

The integrity and secrecy of data are also issues in many court cases. Parker and Nycom describe a case in which a trespasser gained remote access to a computing system. The computing system contained confidential records about people, and the integrity of the data was important. The prosecution of this case had to be phrased in terms of theft of computer time and valued as such, even though that was insignificant compared with loss of privacy and integrity. Why? Because the law as written recognized theft of computer time as a loss, but not loss of privacy or destruction of data. Now, however, several federal and state laws recognize the privacy of data about individuals. For example, disclosing grades or financial information without permission is a crime, and tort law would recognize other cases of computer abuse.

Value of Data

In another computer crime, a person was found guilty of having stolen a substantial amount of data from a computer data bank. However, the court determined that the "value" of that data was the cost of the paper on which it was printed, which was only a few dollars. Because of that valuation, this crime was classified as a misdemeanor and considered to be a minor crime. Fortunately, the courts have since determined that information and other intangibles can have significant value. The concept of what we value and how we determine its value is key to understanding the problems with computer-based law. Over time, the legal system will find ways to place a value on data that is representative of its value to those who use it. Although these methods of valuation are accepted in civil suits, they have not yet been widely accepted in criminal prosecution.

Acceptance of Computer Terminology

The law is also lagging behind technology in its acceptance of definitions of computing terms. For example, according to a federal statute, it is unlawful to commit arson within a federal enclave (18 USC 81). Part of that act relates to "machinery or building material or supplies" in the enclave, but court decisions have ruled that a motor vehicle located within a federal enclave at the time of the burning was not included under this statute. Because of that ruling, it is not clear whether computer hardware constitutes "machinery" in this context; "supplies" almost certainly does not include software. Computers and their software, media, and data must be understood and accepted by the legal system.

15.6.2. Why Computer Crime is hard to define

From these examples, it is clear that the legal community has not accommodated advances in computers as rapidly as has the rest of society. Some people in the legal process do not understand computers and computing, so crimes involving computers are not always treated properly. Creating and changing laws are slow processes, intended to involve substantial thought about the effects of

proposed changes. This deliberate process is very much out of pace with a technology that is progressing as fast as computing. Adding to the problem of a rapidly changing technology, a computer can perform many roles in a crime. A particular computer can be the subject, object, or medium of a crime. A computer can be attacked (attempted unauthorized access), used to attack (impersonating a legitimate node on a network), and used as a means to commit crime (Trojan horse or fake login). Computer crime statutes must address all of these evils.

15.6.3. Why Computer Crime is hard to prosecute

Even when everyone acknowledges that a computer crime has been committed, computer crime is hard to prosecute for the following reasons:

- *Lack of understanding:* Courts, lawyers, police agents, or jurors do not necessarily understand computers. Many judges began practicing law before the invention of computers, and most began before the widespread use of the personal computer. Fortunately, computer literacy in the courts is improving as judges, lawyers, and police officers use computers in their daily activities.
- *Lack of physical evidence:* Police and courts have for years depended on tangible evidence, such as fingerprints. As readers of Sherlock Holmes know, seemingly minuscule clues can lead to solutions to the most complicated crimes. But with many computer crimes there simply are no fingerprints and no physical clues of any sort.
- *Lack of recognition of assets:* We know what cash is, or diamonds, or even negotiable securities. But are twenty invisible magnetic spots really equivalent to a million dollars? Is computer time an asset? What is the value of stolen computer time if the system would have been idle during the time of the theft?
- *Lack of political impact*: Solving and obtaining a conviction for a murder or robbery is popular with the public, and so it gets high priority with prosecutors and police chiefs. Solving and obtaining a conviction for an obscure high-tech crime, especially one not involving obvious and significant loss, may get less attention. However, as computing becomes more pervasive, the visibility and impact of computer crime will increase.
- *Complexity of case:* Basic crimes that everyone understands, such as murder, kidnapping, or auto theft, can be easy to prosecute. A complex money-laundering or tax fraud case may be more difficult to present to a jury because jurors have a hard time following a circuitous accounting trail. But the hardest crime to present may be a high-tech crime, described, for example, as root access by a buffer overflow in which memory was overwritten by other instructions, which allowed the attacker to copy and execute code at will and then delete the code, eliminating all traces of entry.
- *Age of defendant:* Many computer crimes are committed by juveniles. Society understands immaturity and disregards even very serious crimes by juveniles because the juveniles did not understand the impact of their actions. A more serious, related problem is that many adults see juvenile computer crimes as childhood pranks, the modern equivalent of tipping over an outhouse.

Even when there is clear evidence of a crime, the victim may not want to prosecute because of possible negative publicity. Banks, insurance companies, investment firms, the government, and healthcare groups think their trust by the public will be diminished if a computer vulnerability is exposed. Also, they may fear repetition of the same crime by others: so-called copycat crimes. For all of these reasons, computer crimes are often not prosecuted.

15.6.4. Why Computer Criminals are hard to catch

As if computer crime laws and prosecution were not enough, it is also difficult for law enforcement agencies to catch computer criminals. There are two major reasons for this. First, computer crime is a multinational activity that must usually be pursued on a national or local level. There are no international laws on computer crime. Even though the major industrial nations cooperate very effectively on tracking computer criminals, criminals know there are "safe havens" from which they cannot be caught. Often, the trail of a criminal stops cold at the boundary of a country. Riptech Inc. studies Internet attack trends by many factors. For the period January-June 2002 the United States led the world in source of Internet attacks (40 percent) followed by Germany (7 percent). But when you normalize these data for number of users, a very different pattern emerges. Per Internet user, Israel and Hong Kong lead among those nations with more than 1 million users, and Kuwait and Iran top the list among nations with fewer than 1 million users. Nations all over the globe appear on these lists, which demonstrates that attackers can and do operate from many different countries.
Complexity is an even more significant factor than country of origin. As we have stated throughout this book, networked attacks are hard to trace and investigate because they can involve so many steps. A smart attacker will "bounce" an attack through many places to obscure the trail. Each step along the way makes the investigator complete more legal steps. If the trail leads from server A to B to C, the law enforcement investigators need a search warrant for data at A, and others for B and C. Even after obtaining the search warrants, the investigator has to find the right administrator and serve the warrants to begin obtaining data. In the time the investigator has to get and serve warrants, not to mention follow leads and correlate findings, the attacker has carefully erased the digital evidence.

15.6.5. What Computer Crime does not address

Even with the definitions included in the statutes, the courts must interpret what a computer is. Legislators cannot define precisely what a computer is because computer technology is used in many other devices, such as robots, calculators, watches, automobiles, microwave ovens, and medical instruments. More importantly, we cannot predict what kinds of devices may be invented ten or fifty years from now. Therefore, the language in each of these laws indicates the kinds of devices the legislature seeks to include as computers and leaves it up to the court to rule on a specific case. Unfortunately, it takes a while for courts to build up a pattern of cases, and different courts may rule differently in similar situations. The interpretation of each of these terms will be unsettled for some time to come. Value presents a similar problem. As noted in some of the cases presented, the courts have trouble separating the intrinsic value of an object (such as a sheet of paper with writing on it) from its cost to reproduce. The courts now recognize that a Van Gogh painting is worth more than the cost of the canvas and paint. But the courts have not agreed on the value of printed computer output. The cost of a blank diskette is miniscule, but it may have taken thousands of hours of data gathering and machine time to produce the data encoded on a diskette. The courts are still striving to determine the fair value of computer objects. Both the value of a person's privacy and the confidentiality of data about a person are even less settled.

15.7 Ethical Issues in Computer Society

This final section helps clarify thinking about the ethical issues involved in computer security. We list and explain some ethical principles. The primary purpose of this section is to explore some of the ethical issues associated with computer security and to show how ethics functions as a control.

NOT

15.7.1. Differences between the Law and Ethics

As we noted earlier, law is not always the appropriate way to deal with issues of human behaviour. It is difficult to define a law to preclude only the events we want it to. For example, a law that restricts animals from public places must be refined to *permit* guide dogs for the blind. Lawmakers, who are not computer professionals, are hard pressed to think of all the exceptions when they draft a law concerning computer affairs. Even when a law is well conceived and well written, its enforcement may be difficult. The courts are overburdened and prosecuting relatively minor infractions may be excessively time consuming relative to the benefit. Thus, it is impossible or impractical to develop laws to describe and enforce all forms of behaviour acceptable to society. Instead, society relies on ethics or morals to prescribe generally accepted standards of proper behaviour.

An ethic is an objectively defined standard of right and wrong. Ethical standards are often idealistic principles because they focus on one objective. In a given situation, however, several moral objectives may be involved, so people have to determine an action that is appropriate considering all the objectives. Even though religious groups and professional organizations promote certain standards of ethical behaviour, ultimately each person is responsible for deciding what to do in a specific situation. Therefore, through our choices, each of us defines a personal set of ethical practices. A set of ethical principles is called an ethical system.

An ethic is different from a law in several important ways. First, laws apply to everyone: One may disagree with the intent or the meaning of a law, but that is not an excuse for disobeying the law. Second, the courts have a regular process for determining which law supersedes which if two laws conflict. Third, the laws and the courts identify certain actions as right and others as wrong. From a legal standpoint, anything that is not illegal is right. Finally, laws can be enforced to rectify wrongs done by unlawful behaviour.

By contrast, ethics are personal: two people may have different frameworks for making moral judgments. What one person deems perfectly justifiable, another would never consider doing. Second, ethical positions can and often do come into conflict. As an example, the value of a human life is very important in most ethical systems. Most people would not cause the sacrifice of one life, but in the right context some would approve of sacrificing one person to save another, or one to save many others. The value of one life cannot be readily measured against the value of others, and many ethical decisions must be founded on precisely this ambiguity. Yet, there is no arbiter of ethical positions: when two ethical goals collide, each person must choose which goal is dominant. Third, two people may assess ethical values differently; no universal standard of right and wrong exists in ethical judgments. Nor can one person simply look to what another has done as guidance for choosing the right thing to do. Finally, there is no enforcement for ethical choices. These differences are summarized in Table 11-3.

Table 11-3. Contrast of Law vs. Ethics.

INDU

Law	Ethics
Described by formal, written documents	Described by unwritten principles
Interpreted by courts	Interpreted by each individual
Established by legislatures representing all people	Presented by philosophers, religions, professional groups
Applied to everyone	Chosen personally
Priority determined by courts if two laws conflict	Priority determined by an individual if two principles conflict
"Right" arbitrated finally by court	Not arbitrated externally
Enforced by police and courts	Enforced by intangibles such as principles and beliefs

15.7.2. Studying Ethics

The study of ethics is not easy because the issues are complex. Sometimes people confuse ethics with religion because many religions supply a framework in which to make ethical choices. However, ethics can be studied apart from any religious connection. Difficult choices would be easier to make if there were a set of universal ethical principles to which everyone agreed. But the variety of social, cultural, and religious beliefs makes the identification of such a set of universal principles impossible. In this section we explore some of these problems and then consider how understanding ethics can help in dealing with issues of computer security.

Ethics and Religion

Ethics is a set of principles or norms for justifying what is right or wrong in a given situation. To understand what ethics *is* we may start by trying to understand what it *is not.* Ethical principles are different from religious beliefs. Religion is based on personal notions about the creation of the world and the existence of controlling forces or beings. Many moral principles are embodied in the major religions, and the basis of a personal morality is a matter of belief and conviction, much the same as for religions. However, two people with different religious backgrounds may develop the same ethical philosophy, while two exponents of the same religion might reach opposite ethical conclusions in a particular situation. Finally, we can analyze a situation from an ethical perspective and reach ethical conclusions without appealing to any particular religion or religious framework. Thus, it is important to distinguish ethics from religion.

Ethical Principles Are Not Universal

Ethical values vary by society, and from person to person within a society. For example, the concept of privacy is important in Western cultures. But in Eastern cultures, privacy is not desirable because people associate privacy with having something to hide. Not only is a Westerner's desire for privacy not understood but in fact it has a negative connotation. Therefore, the attitudes of people may be affected by culture or background. Also, an individual's standards of behaviour may be influenced by past events in life. A person who grew up in a large family may place greater emphasis on personal control and ownership of possessions than would an only child who seldom had to share. Major events or close contact with others can also shape one's ethical position. Despite these

differences, the underlying principles of how to make moral judgment are the same. Although these aspects of ethics are quite reasonable and understandable, they lead people to distrust ethics because it is not founded on basic principles all can accept. Also, people from a scientific or technical background expect precision and universality.

Ethics Does Not Provide Answers

Ethical pluralism is recognizing or admitting that more than one position may be ethically justifiable even equally so in a given situation. Pluralism is another way of noting that two people may legitimately disagree on issues of ethics. We expect and accept disagreement in such areas as politics and religion. However, in the scientific and technical fields, people expect to find unique, unambiguous, and unequivocal answers. In science, one answer must be correct or demonstrable in some sense. Science has provided life with fundamental explanations. Ethics is rejected or misunderstood by some scientists because it is "soft," meaning that it has no underlying framework or it does not depend on fundamental truths. Scientists are uncomfortable with ethics because ethics does not provide these clean distinctions. Worse, there is no higher authority of ethical truth. Two people may disagree on their opinion of the ethics of a situation, but there is no one to whom to appeal for a final determination of who is "right." Conflicting answers do not deter one from considering ethical issues in computer security. Nor do they excuse us from making and defending ethical choices.

15.7.3. Ethical Reasoning

Most people make ethical judgments often, perhaps daily. (Is it better to buy from a hometown merchant or from a nationwide chain? Should I spend time with a volunteer organization or with my friends? Is it acceptable to release sensitive data to someone who might not have justification for access to that data?) Because we all engage in ethical choice, we should clarify how we do this so that we can learn to apply the principles of ethics in professional situations, as we do in private life. Study of ethics can yield two positive results. First, in situations in which we already know what is right and what is wrong, ethics should help us justify our choice. Second, if we do not know the ethical action to take in a situation, ethics can help us identify the issues involved so that we can make reasoned judgments.

Examining a Case for Ethical Issues

How, then, can issues of ethical choice in computer security be approached? Here are several steps to making and justifying an ethical choice:

- *Understand the situation:* Learn the facts of the situation. Ask questions of interpretation or clarification. Attempt to find out whether any relevant forces have not been considered.
- *Know several theories of ethical reasoning:* To make an ethical choice, you have to know how those choices can be justified.
- *List the ethical principles involved:* What different philosophies could be applied in this case? Do any of these include others?
- *Determine which principles outweigh others:* This is a subjective evaluation. It often involves extending a principle to a logical conclusion or determining cases in which one principle clearly supersedes another.

Examples of Ethical Principles

There are two different schools of ethical reasoning: one based on the good that results from actions and one based on certain prima facie duties of people.

Consequence-Based Principles

The teleological theory of ethics focuses on the consequences of an action. *Teleology* is the general name applied to many theories of behaviour, all of which focus on the goal, outcome, or consequence of the action. There are two important forms of teleology. Egoism is the form that says a moral judgment is based on the positive benefits to the person taking the action. An egoist weighs the outcomes of all possible acts and chooses the one that produces the most personal good for him or her with the least negative consequence. The effects on other people are not relevant. The principle of utilitarianism is also an assessment of good and bad results, but the reference group is the entire universe. The utilitarian chooses that action that will bring the greatest collective good for all people with the least possible negative for all. In this situation, the utilitarian would assess personal good and bad, good and bad for the company, good and bad for the customer, and, perhaps, good and bad for society at large. For example, a developer designing software to monitor smokestack emissions would need to assess its effects on everyone breathing. The utilitarian might perceive greater good to everyone by taking the time to write high-quality code, despite the negative personal consequence of displeasing management.

Rule-Based Principles

Another ethical theory is deontology, which is founded in a sense of duty. This ethical principle states that certain things are good in and of themselves. These things that are naturally good are good rules or acts, which require no higher justification. Something just *is* good; it does not have to be judged for its effect. Examples of intrinsically good things are:

- truth, knowledge, and true opinion of various kinds; understanding, wisdom
- just distribution of good and evil; justice
- pleasure, satisfaction; happiness; life, consciousness
- peace, security, freedom
- good reputation, honour, esteem; mutual affection, love, friendship, cooperation; morally good dispositions or virtues
- beauty, aesthetic experience

Rule-deontology is the school of ethical reasoning that believes certain universal, self-evident, natural rules specify our proper conduct. Certain basic moral principles are adhered to because of our responsibilities to one another; these principles are often stated as rights: the right to know, the right to privacy, the right to fair compensation for work. Sir David Ross lists various duties incumbent on all human beings:

- *fidelity*, or truthfulness
- *reparation*, the duty to recompense for a previous wrongful act
- *gratitude*, thankfulness for previous services or kind acts
- *justice*, distribution of happiness in accordance with merit
- *beneficence*, the obligation to help other people or to make their lives better
- *nonmaleficence*, not harming others
- *self-improvement*, to become continually better, both in a mental sense and in a moral sense

Another school of reasoning is based on rules derived by each individual. Religion, teaching, experience, and reflection lead each person to a set of personal moral principles. The answer to an ethical question is found by weighing values in terms of what a person believes to be right behaviour.

15.8 Summary

- Contracts help fill the voids among criminal, civil, and tort law. That is, in the absence of relevant statutes, we first see common tort law develop. But people then enhance these laws by writing contracts with the specific protections they want.

- Enforcement of civil law, torts or contracts, can be expensive because it requires one party to sue the other. The legal system is informally weighted by money. It is attractive to sue a wealthy party who could pay a hefty judgment. And a big company that can afford dozens of top-quality lawyers will more likely prevail in a suit than an average individual.
- There are four aspects of the relationship between computing and the law.
    - First is the legal mechanisms of copyright, patent, and trade secret as means to protect the secrecy of computer hardware, software, and data. These mechanisms were designed before the invention of the computer, so their applicability to computing needs is somewhat limited. However, program protection is especially desired, and software companies are pressing the courts to extend the interpretation of these means of protection to include computers.
    - Second is the relationship between employers and employees, in the context of writers of software. Well-established laws and precedents control the acceptable access an employee has to software written for a company.
    - Third, is the legal side of software vulnerabilities: Who is liable for errors in software, and how is that liability enforced?
    - Fourth, we noted some of the difficulties in prosecuting computer crime. Several examples showed how breaches of computer security are treated by the courts. In general, the courts have not yet granted computers, software, and data appropriate status, considering value of assets and seriousness of crime. The legal system is moving cautiously in its acceptance of computers.
- In this study of ethics, we have tried not to decide right and wrong, or even to brand certain acts as ethical or unethical. The purpose is to the stimulate thinking about ethical issues concerned with confidentiality, integrity, and availability of data and computations.
    - The important first step in acting ethically in a situation is to obtain the facts, ask about any uncertainties, and acquire any additional information needed. In other words, first we must understand the situation.
    - The second step is to identify the ethical principles involved. Honesty, fair play, proper compensation, and respect for privacy are all ethical principles. Sometimes these conflict, and then we must determine which principles are more important than others. This analysis may not lead to one principle that obviously overshadows all others. Still, a ranking to identify the major principles involved is needed.
    - The third step is choosing an action that meets these ethical principles. Making a decision and taking action are difficult, especially if the action has evident negative consequences. However, taking action based on a *personal* ranking of principles is necessary. The fact that other equally sensible people may choose a different action does not excuse us from taking some action. Decisions may vary, based on fine differences between two situations. Or a person's views can change over time in response to experience and changing context. Learning to reason about ethical situations is not quite the same as learning "right" from "wrong." Terms such as *right* and *wrong* or *good* and *bad* imply a universal set of values. Yet we know that even widely accepted principles are overridden by some people in some situations. Therefore, our purpose in introducing this material has been to stimulate you to recognize and think about ethical principles involved in cases related to computer security. Only by recognizing and analyzing principles can you act consistently, thoughtfully, and responsibly.

15.9 Review Questions

a) Write a short note on copyrights
b) Write a short note on patents

c) Write a short note on trade secrets
d) Compare copyright, patent and trade secret.
e) Why is information different from other commercial things?
f) Write a short note on tort law.
g) Explain the contract law.
h) How can software failures be redressed?
i) Explain the various viewpoints for reporting software flaws.
j) Why a separate category for computer crime is needed?
k) Why is computer crime hard to prosecute?
l) Differentiate between law and ethics.
m) How can issues of ethical choice in computer security be approached?
n) Explain consequence-based and rule-based principles of ethical reasoning.

15.10 Bibliography, References and Further Reading

- *Security in Computing by* C. P. Pfleeger, and S. L. Pfleeger, Pearson Education.
- *Computer Security: Art and Science by* Matt Bishop, Pearson Education.
- *Cryptography And Network Security: Principles and practice by* Stallings
- *Network Security by* Kaufman, Perlman, Speciner
- *Network Security : A Beginner's Guide by* Eric Maiwald, TMH
- *Java Network Security by* Macro Pistoia, Pearson Education
- *Principles of information security by* Whitman, Mattord, Thomson

IDOL