



MACHAKOS UNIVERSITY

SCHOOL OF ENGINEERING & TECHNOLOGY

DEPARTMENT OF COMPUTING & INFORMATION TECHNOLOGY

Course Programme:
BSc Computer Science and Mathematics

Course Unit:
SIT 121 Introduction to Operating Systems

Semester & Academic Year
First Year Second Semester 2020/2021

Lecturer:
Mr. Raphael Kaibiru

Copyright © Machakos University, 2021

All Rights Reserved

MONTH, YEAR

MARCH, 2021

LECTURE 8: SECURITY

Introduction

Computer resources must be guarded against unauthorized access, malicious destruction or alteration, and accidental introduction of inconsistency. These resources include information stored in the system (both data and code), as well as the CPU, memory, disks, tapes, and networking that are the computer.

In this lecture, we start by examining ways in which resources may be accidentally or purposely misused. We then explore a key security enabler —cryptography. Finally, we look at mechanisms to guard against or detect attacks.

The Security Problem

In many applications, ensuring the security of the computer system is worth considerable effort. Large commercial systems containing payroll or other financial data are inviting targets to thieves. Systems that contain data pertaining to corporate operations may be of interest to unscrupulous competitors.

Furthermore, loss of such data, whether by accident or fraud, can seriously impair the ability of the corporation to function. We say that a system is **secure** if its resources are used and accessed as intended under all circumstances. Unfortunately, total security cannot be achieved. Nonetheless, we must have mechanisms to make security breaches a rare occurrence, rather than the norm.

Security violations (or misuse) of the system can be categorized as intentional (malicious) or accidental. It is easier to protect against accidental misuse than against malicious misuse. For the most part, protection mechanisms are the core of protection from accidents. The following list includes several forms of accidental and malicious security violations.

We should note that in our discussion of security, we use the terms **intruder** and **cracker** for those attempting to breach security. In addition, a **threat** is the potential for a security violation, such as the discovery of a vulnerability, whereas an **attack** is the attempt to break security.

1. **Breach of confidentiality.** This type of violation involves unauthorized reading of data (or theft of information). Typically, a breach of confidentiality is the goal of an intruder. Capturing secret data from a system or a data stream, such as credit-card information or identity information for identity theft, can result directly in money for the intruder.

2. **Breach of integrity.** This violation involves unauthorized modification of data. Such attacks can, for example, result in passing of liability to an innocent party or modification of the source code of an important commercial application.
3. **Breach of availability.** This violation involves unauthorized destruction of data. Some crackers would rather wreak havoc and gain status or bragging rights than gain financially. Website defacement is a common example of this type of security breach.
4. **Theft of service.** This violation involves unauthorized use of resources. For example, an intruder (or intrusion program) may install a daemon on a system that acts as a file server.
5. **Denial of service.** This violation involves preventing legitimate use of the system. **Denial-of-service (DOS)** attacks are sometimes accidental. The original Internet worm turned into a DOS attack when a bug failed to delay its rapid spread.

Attackers use several standard methods in their attempts to breach security. The most common is **masquerading**, in which one participant in a communication pretends to be someone else (another host or another person). By masquerading, attackers breach **authentication**, the correctness of identification.

Yet another kind of attack is the **man-in-the-middle attack**, in which an attacker sits in the data flow of a communication, masquerading as the sender to the receiver, and vice versa. In a network communication, a man-in-the-middle attack may be preceded by a **session hijacking**, in which an active communication session is intercepted.

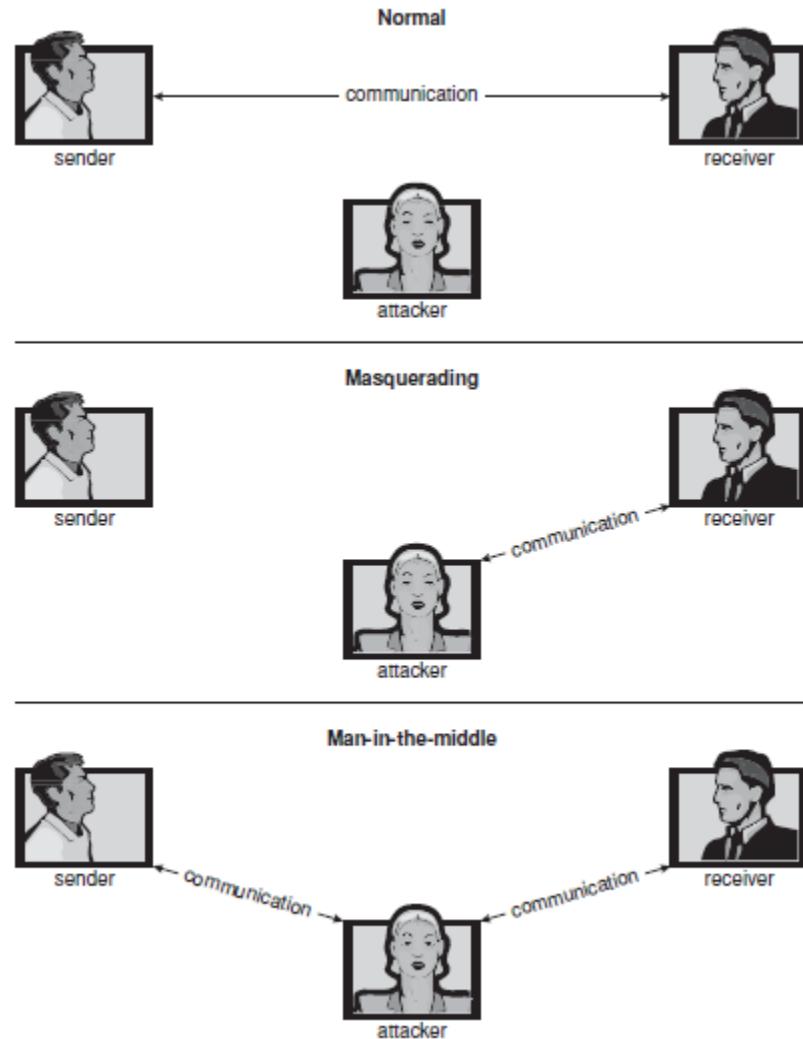


Figure 1: Standard Security Attack

To protect a system, we must take security measures at four levels:

1. **Physical.** The site or sites containing the computer systems must be physically secured against armed or surreptitious entry by intruders. Both the machine rooms and the terminals or workstations that have access to the machines must be secured.
2. **Human.** Authorization must be done carefully to assure that only appropriate users have access to the system. Even authorized users, however, may be “encouraged” to let others use their access (in exchange for a bribe, for example). They may also be tricked into allowing access via **social engineering**. One type of social-engineering attack is **phishing**. Here, a legitimate-looking e-mail or web page misleads a user into entering confidential information. Another technique is **dumpster diving**, a general term for attempting to

gather information in order to gain unauthorized access to the computer (by looking through trash, finding phone books, or finding notes containing passwords, for example).

These security problems are management and personnel issues, not problems pertaining to operating systems.

3. **Operating system.** The system must protect itself from accidental or purposeful security breaches. A runaway process could constitute an accidental denial-of-service attack. A query to a service could reveal passwords. A stack overflow could allow the launching of an unauthorized process. The list of possible breaches is almost endless.
4. **Network.** Much computer data in modern systems travels over private leased lines, shared lines like the Internet, wireless connections, or dial-up lines. Intercepting these data could be just as harmful as breaking into a computer, and interruption of communications could constitute a remote denial-of-service attack, diminishing users' use of and trust in the system.

Security at the first two levels must be maintained if operating-system security is to be ensured.

Program Threats

Processes, along with the kernel, are the only means of accomplishing work on a computer. Therefore, writing a program that creates a breach of security, or causing a normal process to change its behavior and create a breach, is a common goal of crackers. Note that there is considerable variation in the naming conventions for security holes and that we use the most common or descriptive terms.

1. Trojan horse

Many systems have mechanisms for allowing programs written by users to be executed by other users. If these programs are executed in a domain that provides the access rights of the executing user, the other users may misuse these rights. A text-editor program, for example, may include code to search the file to be edited for certain keywords. If any are found, the entire file may be copied to a special area accessible to the creator of the text editor. A code segment that misuses its environment is called a **Trojan horse**.

A variation of the Trojan horse is a program that emulates a login program. An unsuspecting user starts to log in at a terminal and notices that he has apparently mistyped his password. He tries again and is successful. What has happened is that his authentication key and password have been stolen by the login emulator, which was left running on the terminal by the thief.

The emulator stored away the password, printed out a login error message, and exited; the user was then provided with a genuine login prompt. This type of attack can be defeated by having the operating system print a usage message at the end of an interactive session or by a non-trappable key sequence, such as the control-alt-delete combination used by all modern Windows operating systems.

Another variation on the Trojan horse is **spyware**. Spyware sometimes accompanies a program that the user has chosen to install. Most frequently, it comes along with freeware or shareware programs, but sometimes it is included with commercial software. The goal of spyware is to download ads to display on the user's system, create pop-up browser windows when certain sites are visited, or capture information from the user's system and return it to a central site.

2. Trap door

The designer of a program or system might leave a hole in the software that only she is capable of using. This type of security breach (or **trap door**) was shown in the movie *War Games*. For instance, the code might check for a specific user ID or password, and it might circumvent normal security procedures. Programmers have been arrested for embezzling from banks by including rounding errors in their code and having the occasional half-cent credited to their accounts.

This account crediting can add up to a large amount of money, considering the number of transactions that a large bank executes. Trap doors pose a difficult problem because, to detect them, we have to analyze all the source code for all components of a system. Given that software systems may consist of millions of lines of code, this analysis is not done frequently, and frequently it is not done at all!

3. Logic bomb

Consider a program that initiates a security incident only under certain circumstances. It would be hard to detect because under normal operations, there would be no security hole. However, when a predefined set of parameters was met, the security hole would be created. This scenario is known as a **logic bomb**. A programmer, for example, might write code to detect whether he was still employed; if that check failed, a daemon could be spawned to allow remote access, or code could be launched to cause damage to the site.

4. Viruses

Another form of program threat is a **virus**. A virus is a fragment of code embedded in a legitimate program. Viruses are self-replicating and are designed to “infect” other programs. They can wreak havoc in a system by modifying or destroying files and causing system crashes and program malfunctions. As with most penetration attacks, viruses are very specific to architectures, operating systems, and applications. Viruses are a particular problem for users of PCs.

System and Network Threat

Program threats typically use a breakdown in the protection mechanisms of a system to attack programs. In contrast, system and network threats involve the abuse of services and network connections. System and network threats create a situation in which operating-system resources and user files are misused. Sometimes, a system and network attack is used to launch a program attack, and vice versa.

a) Worms

A **worm** is a process that uses the **spawn** mechanism to duplicate itself. The worm spawns copies of itself, using up system resources and perhaps locking out all other processes. On computer networks, worms are particularly potent, since they may reproduce themselves among systems and thus shut down an entire network.

b) Port Scanning

Port scanning is not an attack but rather a means for a cracker to detect a system’s vulnerabilities to attack. Port scanning typically is automated, involving a tool that attempts to create a TCP/IP connection to a specific port or a range of ports. Because port scans are detectable (Section 15.6.3), they frequently are launched from **zombie systems**. Such systems are previously compromised, independent systems that are serving their owners while being used for nefarious purposes, including denial-of-service attacks and spam relay. Zombies make crackers particularly difficult to prosecute because determining the source of the attack and the person that launched it is challenging. This is one of many reasons for securing “inconsequential” systems, not just systems containing “valuable” information or services.

c) Denial of Service

As mentioned earlier, denial-of-service attacks are aimed not at gaining information or stealing resources but rather at disrupting legitimate use of a system or facility. Most such attacks involve

systems that the attacker has not penetrated. Launching an attack that prevents legitimate use is frequently easier than breaking into a machine or facility.

Denial-of-service attacks are generally network based. They fall into two categories. Attacks in the first category use so many facility resources that, in essence, no useful work can be done.

Generally, it is impossible to prevent denial-of-service attacks. The attacks use the same mechanisms as normal operation. Even more difficult to prevent and resolve are **distributed denial-of-service (DDOS)** attacks. These attacks are launched from multiple sites at once, toward a common target, typically by zombies.

User Authentication

Thus, a major security problem for operating systems is **user authentication**. Generally, user authentication is based on one or more of three things: the user's possession of something (a key or card), the user's knowledge of something (a user identifier and password), or an attribute of the user (fingerprint, retina pattern, or signature).

1) Password

The most common approach to authenticating a user identity is the use of **passwords**. When the user identifies herself by user ID or account name, she is asked for a password. If the user-supplied password matches the password stored in the system, the system assumes that the account is being accessed by the owner of that account.

In practice, most systems require only one password for a user to gain full rights. Although more passwords theoretically would be more secure, such systems tend not to be implemented due to the classic trade-off between security and convenience. If security makes something inconvenient, then the security is frequently bypassed or otherwise circumvented.

2) Password Vulnerability

Passwords are extremely common because they are easy to understand and use. Unfortunately, passwords can often be guessed, accidentally exposed, sniffed (read by an eavesdropper), or illegally transferred from an authorized user to an unauthorized one.

There are two common ways to guess a password. One way is for the intruder (either human or program) to know the user or to have information about the user. All too frequently, people use obvious information (such as the names of their cats or spouses) as their passwords. The other way is to use brute force, trying enumeration—or all possible combinations of valid password characters (letters, numbers, and punctuation on some systems)—until the password is found.

3) Securing Password

A good technique is to generate your password by using the first letter of each word of an easily remembered phrase using both upper and lower characters with a number or punctuation mark thrown in for good measure. For example, the phrase “My mother’s name is Katherine” might yield the password “Mmn.isK!”. The password is hard to crack but easy for the user to remember. A more secure system would allow more characters in its passwords. Indeed, a system might also allow passwords to include the space character, so that a user could create a **passphrase**.

4) One time Password

To avoid the problems of password sniffing and shoulder surfing, a system can use a set of **paired passwords**. When a session begins, the system randomly selects and presents one part of a password pair; the user must supply the other part. In this system, the user is **challenged** and must **respond** with the correct answer to that challenge.

5) Biometric

Yet another variation on the use of passwords for authentication involves the use of biometric measures. Palm- or hand-readers are commonly used to secure physical access—for example, access to a data center. These readers match stored parameters against what is being read from hand-reader pads. The parameters can include a temperature map, as well as finger length, finger width, and line patterns. These devices are currently too large and expensive to be used for normal computer authentication.

Implementing Security Defenses

Just as there are myriad threats to system and network security, there are many security solutions. The solutions range from improved user education, through technology, to writing bug-free software. Most security professionals subscribe to the theory of **defense in depth**, which states that more layers of defense are better than fewer layers. Of course, this theory applies to any kind of security.

Consider the security of a house without a door lock, with a door lock, and with a lock and an alarm. In this section, we look at the major methods, tools, and techniques that can be used to improve resistance to threats.

Security Policy

The first step toward improving the security of any aspect of computing is to have a **security policy**. Policies vary widely but generally include a statement of what is being secured. For

example, a policy might state that all outside accessible applications must have a code review before being deployed, or that users should not share their passwords, or that all connection points between a company and the outside must have port scans run every six months. Without a policy in place, it is impossible for users and administrators to know what is permissible, what is required, and what is not allowed. The policy is a road map to security, and if a site is trying to move from less secure to more secure, it needs a map to know how to get there.

Once the security policy is in place, the people it affects should know it well. It should be their guide. The policy should also be a **living document** that is reviewed and updated periodically to ensure that it is still pertinent and still followed.

Vulnerability Assessment

The best way is to execute a vulnerability assessment. Such assessments can cover broad ground, from social engineering through risk assessment to port scans. **Risk assessment**, for example, attempts to value the assets of the entity in question (a program, a management team, a system, or a facility) and determine the odds that a security incident will affect the entity and decrease its value. When the odds of suffering a loss and the amount of the potential loss are known, a value can be placed on trying to secure the entity.

Intrusion Detection

Securing systems and facilities is intimately linked to intrusion detection. **Intrusion detection**, as its name suggests, strives to detect attempted or successful intrusions into computer systems and to initiate appropriate responses to the intrusions. Intrusion detection encompasses a wide array of techniques that vary on a number of axes, including the following:

- The time at which detection occurs. Detection can occur in real time (while the intrusion is occurring) or after the fact.
- The types of inputs examined to detect intrusive activity. These may include user-shell commands, process system calls, and network packet headers or contents. Some forms of intrusion might be detected only by correlating information from several such sources.
- The range of response capabilities. Simple forms of response include alerting an administrator to the potential intrusion or somehow halting the potentially intrusive activity—for example, killing a process engaged in such activity. In a sophisticated form of response, a system might transparently divert an intruder's activity to a **honeypot**—a false resource exposed to the attacker.

The resource appears real to the attacker and enables the system to monitor and gain information about the attack.

Virus Protection

As we have seen, viruses can and do wreak havoc on systems. Protection from viruses thus is an important security concern. Antivirus programs are often used to provide this protection. Some of these programs are effective against only particular known viruses. They work by searching all the programs on a system for the specific pattern of instructions known to make up the virus.

When they find a known pattern, they remove the instructions, **disinfecting** the program. Antivirus programs may have catalogs of thousands of viruses for which they search.

Firewalling to Protect Systems and Networks

We turn next to the question of how a trusted computer can be connected safely to an untrustworthy network. One solution is the use of a firewall to separate trusted and untrusted systems. A **firewall** is a computer, appliance, or router that sits between the trusted and the untrusted. A network firewall limits network access between the two **security domains** and monitors and logs all connections. It can also limit connections based on source or destination address, source or destination port, or direction of the connection. For instance, web servers use HTTP to communicate with web browsers. A firewall therefore may allow only HTTP to pass from all hosts outside the firewall to the web server within the firewall. In fact, a network firewall can separate a network into multiple domains.

A common implementation has the Internet as the untrusted domain; a semitrusted and semisecure network, called the **demilitarized zone (DMZ)**, as another domain; and a company's computers as a third domain. Connections are allowed from the Internet to the DMZ computers and from the company computers to the Internet but are not allowed from the Internet or DMZ computers to the company computers. Optionally, controlled communications maybe allowed between the DMZ and one company computer or more. For instance, a web server on the DMZ may need to query a database server on the corporate network. With a firewall, however, access is contained, and any DMZ systems that are broken into still are unable to access the company computers.

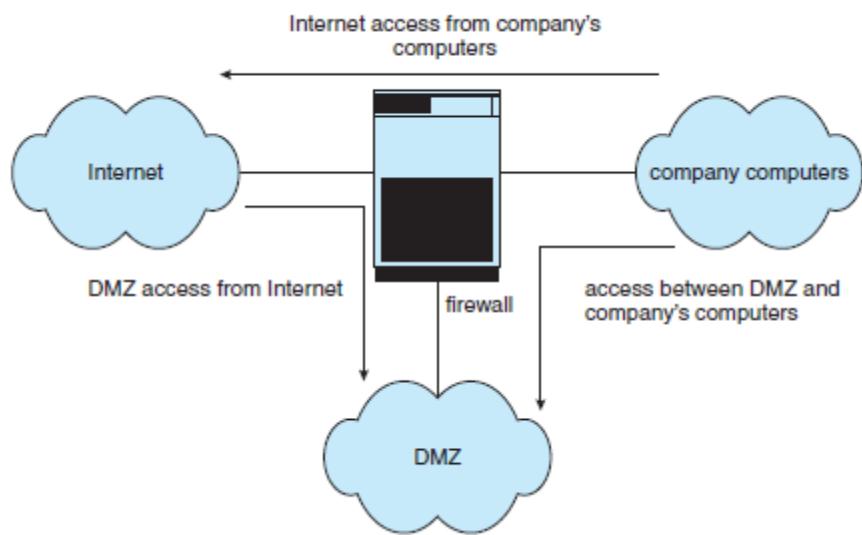


Figure 2: Domain Separation in Firewall

Summary

Protection is an internal problem. Security, in contrast, must consider both the computer system and the environment—people, buildings, businesses, valuable objects, and threats—within which the system is used.

The data stored in the computer system must be protected from unauthorized access, malicious destruction or alteration, and accidental introduction of inconsistency. It is easier to protect against accidental loss of data consistency than to protect against malicious access to the data. Absolute protection of the information stored in a computer system from malicious abuse is not possible; but the cost to the perpetrator can be made sufficiently high to deter most, if not all, attempts to access that information without proper authority.

User authentication methods are used to identify legitimate users of a system. In addition to standard user-name and password protection, several authentication methods are used. One-time passwords, for example, change from session to session to avoid replay attacks. Two-factor authentication requires two forms of authentication, such as a hardware calculator with an activation PIN. Multifactor authentication uses three or more forms. These methods greatly decrease the chance of authentication forgery.

Methods of preventing or detecting security incidents include intrusion detection systems, antivirus software, auditing and logging of system events, monitoring of system software changes, system-call monitoring, and firewalls.