

Secure Operating Systems (CYB610)

Master of Science in Cybersecurity

Lecture 3 User Privileges and Permissions

Dr. Ibrahim Gomaa

*Faculty of Informatics
Midocean University*



Chapter

03

Topics

This chapter covers the following topics and concepts:

- **Operating system security attack types**
- **Principles of secure OS Design**
- **Protecting an OS from Malicious software**
- **Memory and address protection**
- **Linux Security**
- **Windows Security**

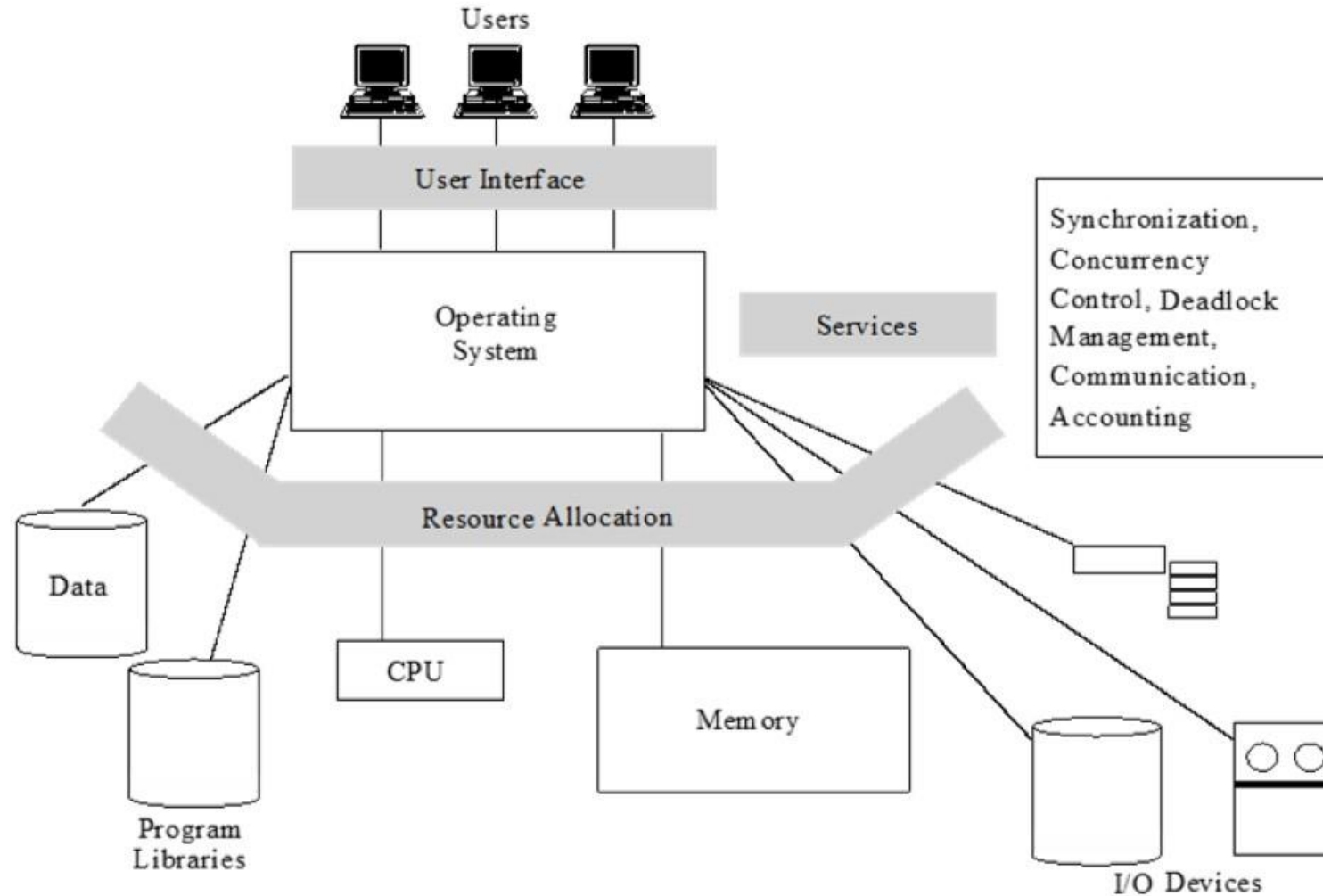
Chapter

03

Goals

When you complete this chapter, you will be able to:

- **Describe the protected objects**
- **Suggest how to secure your computer OS**
- **Explain the method of protecting an OS from malicious software**
- **Explain users, groups, and file permission**





- Single-user systems, no OS
- Multiprogramming OS
 - ✓ Multiple users
 - ✓ Multiple programs
 - ✓ Scheduling, sharing, concurrent use
- Personal computers
 - ✓ Single-user, multiple tasks
 - ✓ Multiple-user, multiple tasks



- **Control panel:** main component of operating system security environment
- Used to gain access to the OS and its features
- **Include**
 - ✓ User authentication
 - ✓ Remote access
 - ✓ Administration tasks
 - ✓ Password policies



❖ These are some of the common objects that need protection by and in OSs.

- Memory
- Networks
- Serially reusable I/O devices, such as printers
- **Sharable:**
 - programs and sub procedures
 - I/O devices, such as disks
 - data





❖ Malware Attack:

- A generic term for software that has a malicious purpose
- Is software that is intentionally included or inserted in a system for a harmful purpose.

❖ Different forms of malicious software (malware) Intended to

- Cause distress to a user
- Damage files or systems
- Disrupt ordinary computer and network functions

❖ Examples

- Viruses, worms
- Logic bomb
- Trojan horses
- Spy-wares
- New ones: Spam/scam, Slammer, Nimda, e-payment frauds, etc...



Name	Description
Virus	Attaches itself to a program and propagates copies of itself to other programs
Worm	Program that propagates copies of itself to other computers
Logic bomb	Triggers action when condition occurs
Trojan horse	Program that contains unexpected additional functionality
Backdoor (trapdoor)	is a secret entry point into a program that allows someone that is aware of the backdoor to gain access without going through the usual security access procedures.
Auto-rooter	Malicious hacker tools used to break into new machines remotely
Kit (virus generator)	Set of tools for generating new viruses automatically
Spammer programs	Used to send large volumes of unwanted e-mail
Flooders	Used to attack networked computer systems with a large volume of traffic to carry out a denial of service (DoS) attack
Keyloggers	Captures keystrokes on a compromised system
Zombie	Program activated on an infected machine that is activated to launch attacks on other machines
Rootkit	Set of hacker tools used after attacker has broken into a computer system and gained root-level access



- ❖ **Malicious software** can be divided into two categories:
 - i. **those that need a host program**
 - fragments of programs that cannot exist independently of some actual application program, utility, or system program.
 - Viruses and logic bombs are examples
 - ii. **those that are independent**
 - are self-contained programs that can be scheduled and run by the operating system.
 - Worms and zombie programs are examples.



❖ Viruses

- A malicious code that replicates and hides itself inside other programs usually without your knowledge.
- A virus is a piece of software that can "infect" programs by modifying them.
- Similar to biological virus: Replicates and Spreads
- Can do serious damage such as erasing file...

❖ Worms

- A worm is a program that can replicate itself and send copies from computer to computer across network connections.



During its lifetime, a typical virus goes through the following **four phases**:

- **Dormant phase:** The virus is idle.
 - The virus will eventually be activated by some event, such as a date, the presence of another program or file, or the capacity of the disk exceeding some limit.
- **Propagation phase:** The virus places an identical copy of itself into other programs or into certain system areas on the disk.
 - Each infected program will now contain a clone of the virus, which will itself enter a propagation phase.
- **Triggering phase:** The virus is activated to perform the function for which it was intended.
 - As with the dormant phase, the triggering phase can be caused by a variety of system events
- **Execution phase:** The function is performed.
 - The function may be harmless, such as a message on the screen, or
 - damaging, such as the destruction of programs and data files.



Types of viruses

- **Parasitic virus:** The traditional and still most common form of virus.
 - A parasitic virus attaches itself to executable files and replicates
- **Memory-resident virus:** resides in main memory as part of a resident system program.
 - From that point on, the virus infects every program that executes.
- **Boot sector virus:** Infects a master boot record or boot record and spreads when a system is booted from the disk containing the virus.
- **Stealth virus:** A form of virus explicitly designed to **hide** itself from detection by antivirus software.
 - a virus that uses compression so that the infected program is exactly the same length as an uninfected version.



Types of viruses

- **Polymorphic virus:** A virus that **mutates** with every infection, making detection by the "signature" of the virus impossible.
- **Metamorphic virus:** As with a polymorphic virus, a metamorphic virus mutates with every infection.
 - **The difference is that:**
 - A metamorphic **virus rewrites itself** completely at each iteration, increasing the difficulty of detection.
 - Metamorphic viruses may change their **behavior** as well as their **appearance**.



- **Logic bomb**

- The logic bomb is code embedded in some legitimate program that is set to "explode" when certain conditions are met.
- Examples of conditions that can be used as triggers for a logic bomb are:
 - The presence or absence of certain files,
 - A particular day of the week or date, or
 - A particular user running the application.
- Once triggered, a bomb may:
 - Alter or delete data or entire files,
 - Cause a machine halt, or
 - Do some other damage



- **Trojan Horse**

- Any malicious program which misinterprets itself as useful, or interesting in order to convince a victim to install it.
- The program claims to do one thing
 - (it may claim to be a game) but instead does damage when you run it (it may erase your Hard Disk).
- Trojan horse programs do not replicate themselves like a virus,
- Such program traps user login credentials and stores them to send to malicious user





- ❖ A rootkit is a malicious software package that attains and takes advantage of root status or effectively becomes part of the OS
- ❖ Rootkits often go to great length to avoid being discovered or, if discovered and partially removed, to reestablish themselves
 - ✓ This can include intercepting or modifying basic OS functions



- Software placed on a computer
 - typically without user's knowledge
 - reports back information about user's activities
- Some operate through monitoring cookies
- A software that literally spies on what you do on your computer.
 - Example: Simple Cookies, mobile codes , web crawlers, Xerox
 - Types of information that is gathered includes the Websites visited, browser and system information, and your computer IP address.



- Filling e-mail inboxes with unwanted junk mail.
- Anyone using e-mail is essentially guaranteed to receive spam
- How spammers get your mail.
 - Web search
 - Sending test emails
 - Exchange or buy from other spammers



- Never reply junk emails
- Do not open any files or executable attachments
- Immediately **DELETE** the malicious email
- Don't post your actual email address in the website.
- Norton, McAfee, and many more include spam as one of the threats that they protect against.
- Can use spam filtering applications



- **Infection mechanisms**

- First, the virus should search for and detect objects to infect
- Installation into the infectable object
 - Writing on the boot sector
 - Add some code to executable programs
 - Add some code to initialization/auto-executable programs

- **Trigger mechanism**

- Date
- Number of infections
- First use

- **Effects: It can be anything**

- A message
- Deleting files
- Formatting disk
- Overloading processor/memory
- Etc.



- Check for suspicious patterns of activity
 - i.e. several incorrect password attempts may signal password guessing.
- Audit log – records the time, user, and type of all accesses to an object; useful for recovery from a violation and developing better security measures.
- Scan the system periodically for security holes; done when the computer is relatively unused.



Check for:

- Short or easy-to-guess passwords
- Unauthorized set-uid programs
- Unauthorized programs in system directories
- Unexpected long-running processes
- Improper directory protections
- Improper protections on system data files
- Dangerous entries in the program search path (Trojan horse)
- Changes to system programs: monitor checksum values



- Use anti-virus software.
- Depending on the vendor, the antivirus software may also contain anti-spyware tools, anti-spam filtering, a personal firewall, and more.
- Update your computer regularly.
- Be careful with the email attachments
 - Safe: .jpg .bmp .pdf .txt
 - Unsafe: .exe .doc .xls .ppt ...
- Use firewall to protect you from malware attack.
- Use IDS...



➤ Simplicity of design

- OSs are inherently complex, and any unnecessary complexity only makes them harder to understand and secure

➤ Layered design

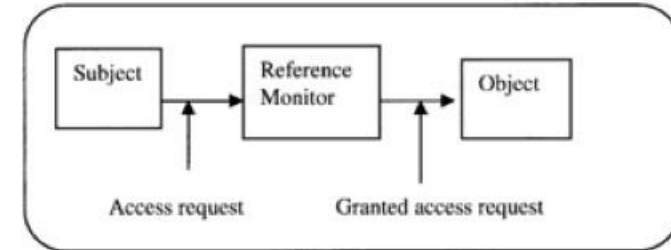
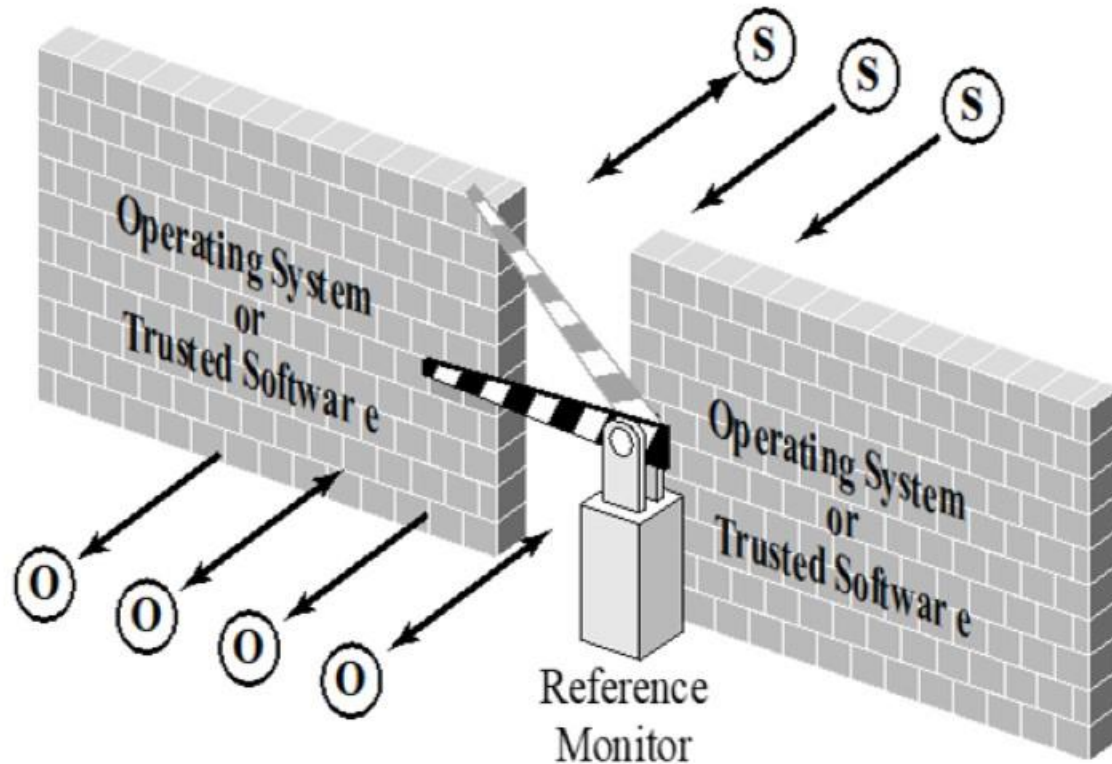
- Enables layered trust

➤ Layered trust

- Layering is both a way to keep a design logical and understandable and a way to limit risk
- Example: very tight access controls on critical OS functions, fewer access controls on important noncritical functions, and few if any access controls on functions that aren't important to the OS



- A kernel is the part of the OS that performs the lowest-level functions
 - Synchronization
 - Inter-process communication
 - Message passing
 - Interrupt handling
- A security kernel is responsible for enforcing the security mechanisms of the entire OS
 - Typically contained within the kernel



- **The subject**, for example a user, initiates an access request for a specified system resource, such as a Web resource.
- **The request** goes to the reference monitor.
- **The job of the reference monitor** is to check on the hierarchy of rules that specify certain restrictions.
- A set of such rules is called an **access control list** (ACL).
- The access control hierarchy is based on the URL path for a Web access, or file path for a file access such as in a directory.
- When a request for access is made, the monitor or server goes in turn through each ACL rule, continuing until it encounters a rule that prevents it from continuing and results in a request rejection or comes to the last rule for that resource, resulting into access right being it granted.



- A trusted system is one that has been shown to warrant some degree of trust that it will perform certain activities faithfully
- **Characteristics of a trusted system:**
 - A defined policy that details what security qualities it enforces
 - Appropriate measures and mechanisms by which it can enforce security adequately
 - Independent scrutiny or evaluation to ensure that the mechanisms have been selected and implemented properly



➤ Secure startup

- System startup is a tricky time for security, as most systems load basic I/O functionality before being able to load security functions

➤ Trusted path

- An unforgeable connection by which the user can be confident of communicating directly with the OS

➤ Object reuse control

- OS clears memory before reassigning it to ensure that leftover data doesn't become compromised

➤ Audit

- Trusted systems track security-relevant changes, such as the installation of new programs or OS modification
- Audit logs must be protected against tampering and deletion



- **Install updates**
- Use malicious software scanners
- Back up systems and create repair disks
- Create and implement organizational policies



- **Update**
 - Provides access to patches that are regularly issued/released
- **Service packs**
 - Address security issues and problems affecting stability, performance, or operation of features included with the OS
- **Patch** This fixes something small and is usually quick to download and install.
- **Rollup** This might include a group of patches for a program.
- **Update** Updates might add or fix features in your program or fix an earlier patch.



- Install updates
- **Use malicious software scanners**
- Back up systems and create repair disks
- Create and implement organizational policies



- Effective way to protect operating system
- Scan systems for virus, worms, and Trojan horses
- Often Called Virus Scanners
- Functions of anti-viruses
 - Identification of known viruses
 - Detection of suspected viruses
 - Blocking of possible viruses
 - Disinfection of infected objects
 - Deletion and overwriting of infected objects



- The most obvious problem of multiprogramming is preventing one program from affecting the data and programs in the memory space of other users.



- Fence
- Relocation
- Base/Bounds Registers
- Tagged Architecture
- Segmentation
- Paging
- Combined Paging with Segmentation

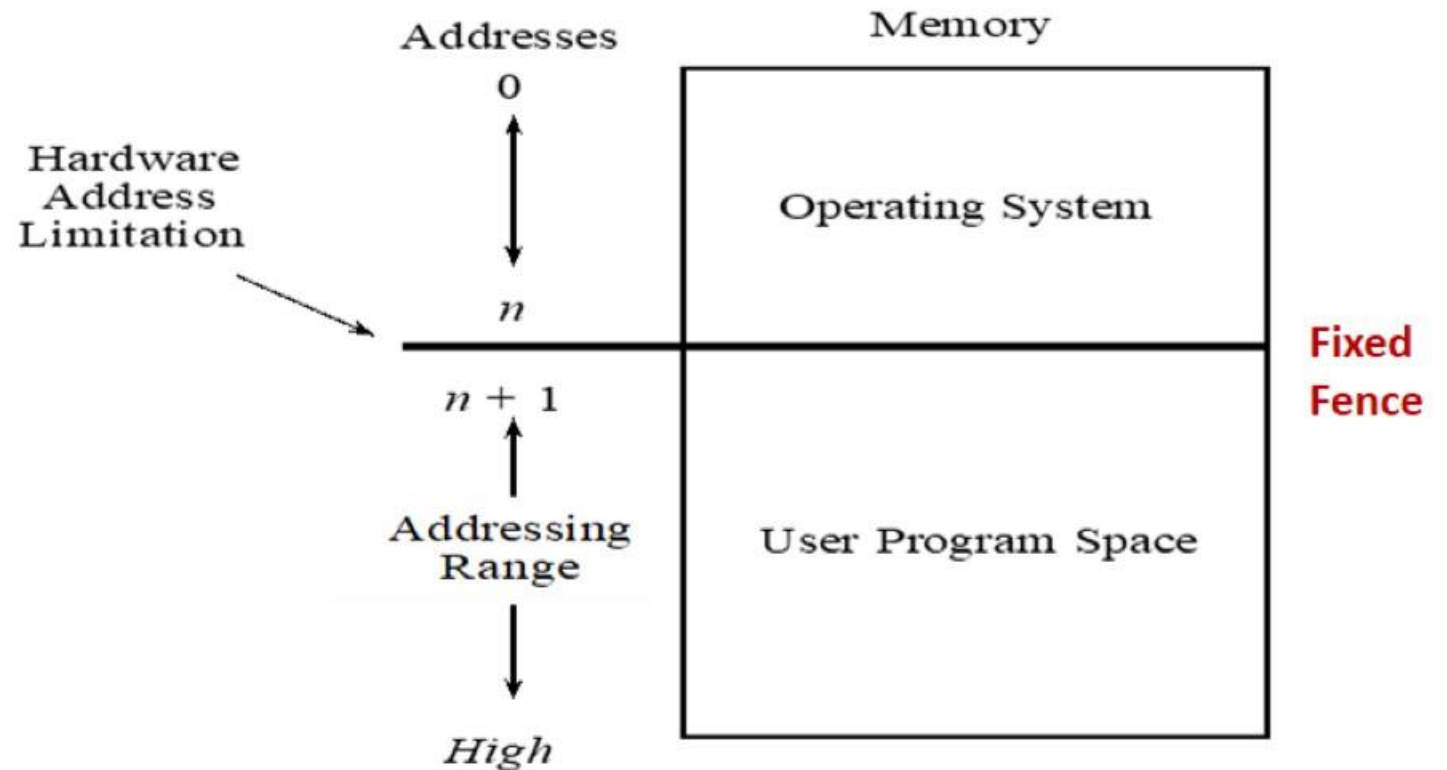


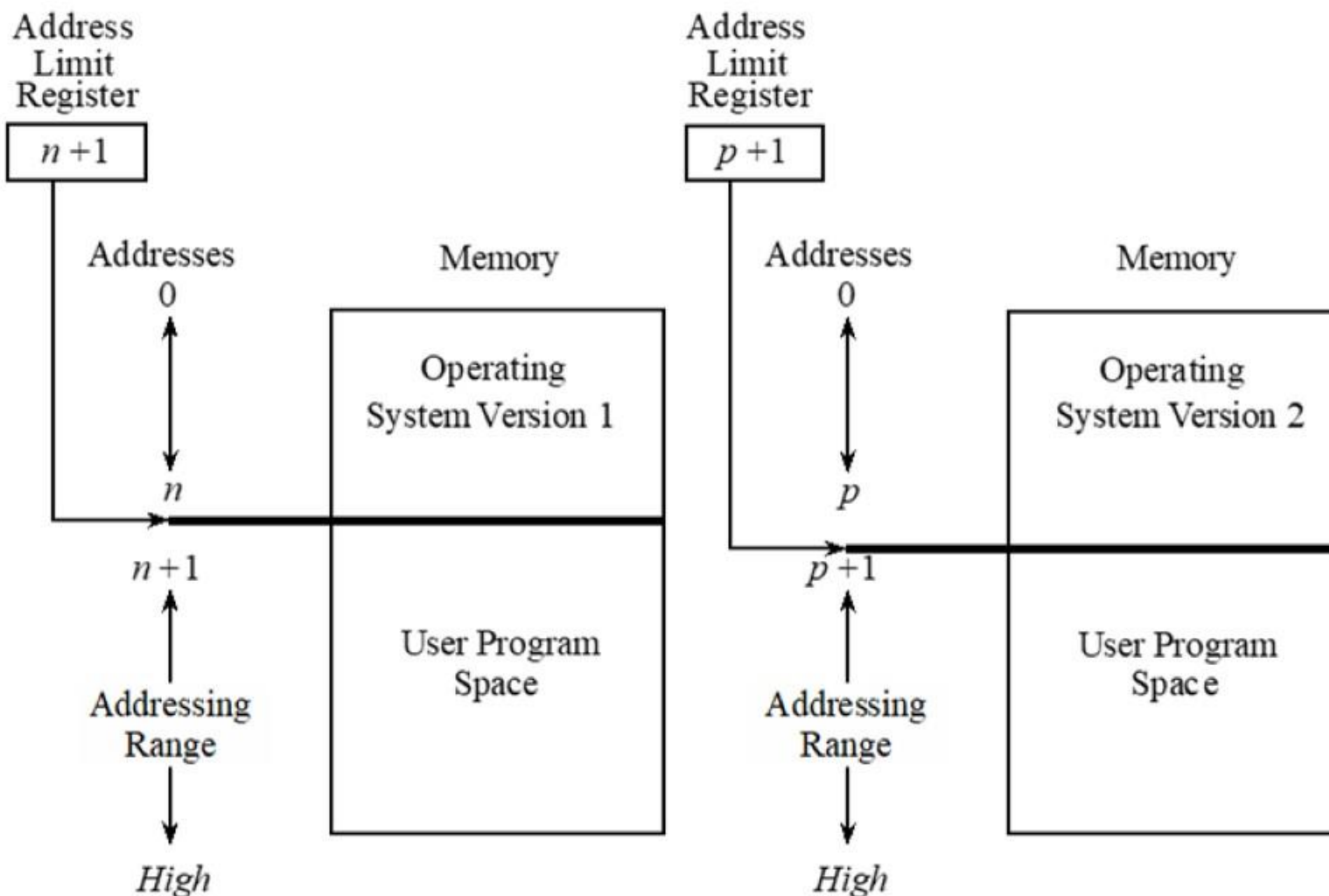
- The simplest form of memory protection was introduced in single-user operating systems to prevent a faulty user program from destroying part of the resident portion of the operating system.
- A fence is a method to confine users to one side of a boundary.





- The fence was a **predefined memory address**, enabling the operating system to reside on **one side** and the user to stay on **the other**.







- Containing the address of the end of the operating system.
- The location of the fence could be changed.
- Each time a user program generated an address for data modification
 - If the address was greater than the fence address the instruction was executed
 - if it was less than the fence address, an error condition was raised
- ❖ **The fence cannot protect one user from another user.**



- Like other general-purpose operating systems, Linux's wide range of features presents a broad attack surface.
- Even so, by leveraging native Linux security controls, carefully configuring Linux applications, and deploying certain add-on security packages, you can create highly secure Linux systems.



➤ Patch management

- Keeping security patches up to date is a widely recognized and critical control for maintaining security

➤ Application and service configuration

- Most commonly implemented using separate text files for each application and service
- Generally located either in the /etc directory or in the installation tree for a specific application
- Individual user configurations that can override the system defaults are located in hidden “dot” files in each user’s home directory
- Most important changes needed to improve system security are to disable services and applications that are not required



➤ Users, groups, and permissions

- Access is specified as granting read, write, and execute permissions to each of owner, group, and others for each resource
- Guides recommend changing the access permissions for critical directories and files
- Local exploit - Software vulnerability that can be exploited by an attacker to gain elevated privileges
- Remote exploit - Software vulnerability in a network server that could be triggered by a remote attacker

➤ chroot jail

- Restricts the server's view of the file system to just a specified portion
- Uses chroot system call to confine a process by mapping the root of the filesystem to some other directory
- File directories outside the chroot jail aren't visible or reachable
- Main disadvantage is added complexity



- There are two things on a Linux/UNIX system that aren't represented by files:
 - User accounts
 - Group accounts
- **User account**
 - Represents someone or something capable of using files
- **Group account**
 - A list of user accounts
 - Each user account is defined with a main group membership, but may belong to as many groups as you need it to



- Each file on a UNIX system has **two owners (a user & a group)**
- Each user and group has its own set of permissions that specify what the user or group may do with the file (read it, write to it, delete it, execute it)
- Other
 - User accounts that don't own the file or belong to the group that owns it
- Permissions are listed in the order “**user permissions, group permissions, other permissions**”.
- Listing below shows a long file-listing for the file
/home/maestro/baton_dealers.txt

```
-rw-rw-r-- 1 maestro conductors 35414 Mar 25 01:38 baton_dealers.txt
```

File-Listing Showing Permissions



- Operating System Functions
- Protected Objects
- Operating system Security attack types
- Malware Attacks
- Principles of Secure OS Design
- Memory and Address Protection
- Linux/Unix Security



Thank You