

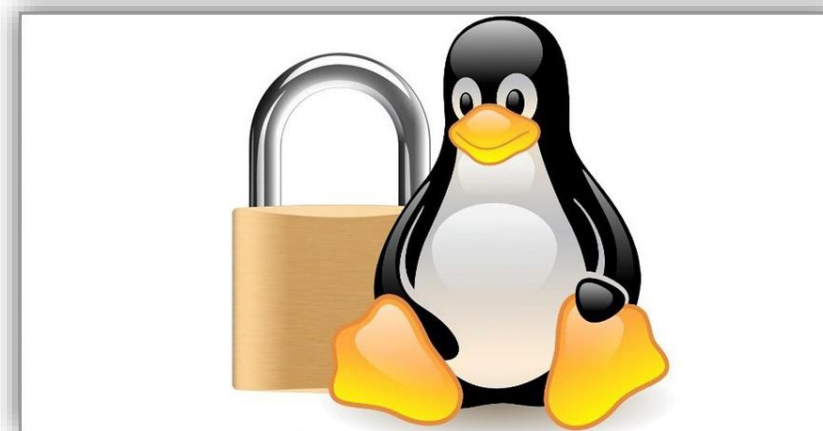


مجموعة جامعة
الجميع الذكية
للتعليم القابضة

MIDOCEAN
UNIVERSITY
BY EVERYONE'S SMART UNIVERSITY

Lecture: #02

Basic Components of OS Security.



Dr. Hesham Abou Elfetouh

*Cybersecurity Department
Faculty of Informatics
Midocean University*



Secure Operating Systems (CYB610)

www.midocean.ae



This chapter covers the following topics and concepts

What is the Vulnerabilities

**How the Methods
of Defense work**

What is the Software Controls

**What is the Protected Objects and Methods
of Protection**

**What is the
Protection Types**



In Previous Lecture you learned:

- Who are the attackers
- What are the vulnerabilities
- Security features in Operating System
- Operating Systems Hardening
- Security Maintenance
- The origins of Linux
- Open-source software security considerations
- The confidentiality, integrity, and availability (C-I-A) triad
- Linux distributions
- Linux security and security threats



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

- Describe the Security goals
- Describe the Vulnerabilities
- Explain the Principles of Adequate Protection
- Differentiate between the Methods of Securing the Operating Systems
- Explain the Protection Types



The purpose of computer security is to:

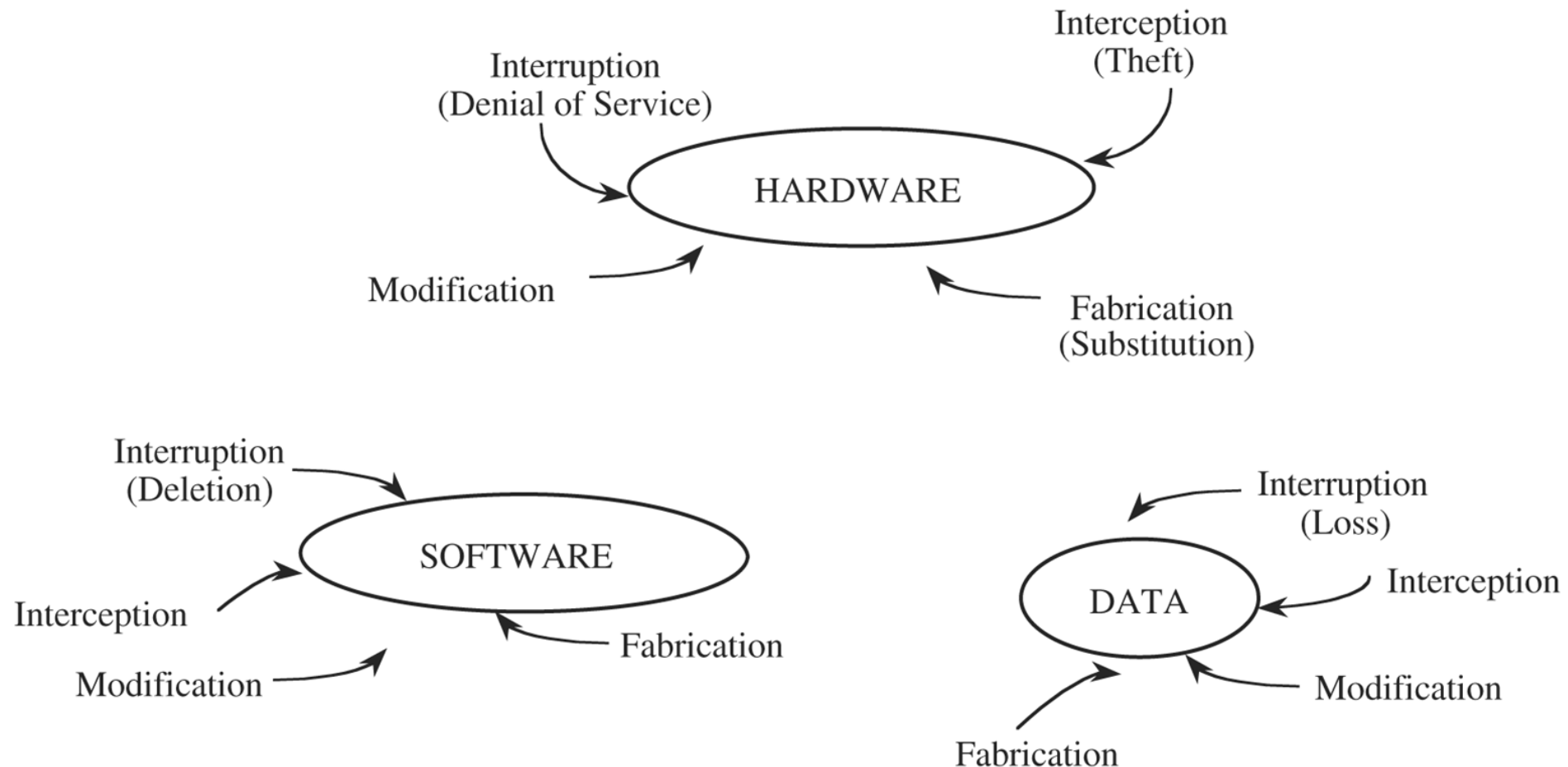
- Devise ways to prevent the weaknesses from being exploited.



- **Physical** vulnerabilities (Eg. Computer can be stolen)
- **Natural** vulnerabilities (Eg. Earthquake)
- **Hardware and Software** vulnerabilities (Eg. Failures)
- **Media** vulnerabilities (Eg. Hard disks can be stolen)
- **Communication** vulnerabilities (Ex. Wires can be tapped)
- **Human** vulnerabilities (Eg. Insiders)
- Poorly chosen passwords
- Software bugs (non reliability of software)
 - **buffer overflow attacks**



- When we prepare to **specify, design, code, or test** a secure system, we try to imagine the vulnerabilities that would prevent us from reaching one or more of our three security goals (**C-I-A**)
- **The three assets** (**hardware, software and data**) and the connections among them are all potential security weak points





- Software is surprisingly easy to delete and to copy
- Software is vulnerable to modifications that either cause it to fail or cause it to perform an unintended task



- ❖ **Logic bomb:** a program that has been maliciously modified to fail when certain conditions are met or when a certain date or time is reached
- ❖ **Trojan horse:** a program that overtly does one thing while covertly doing another
- ❖ **Virus:** a specific type of Trojan horse that can be used to spread its "infection" from one computer to another



- ❖ **Trapdoor:** a program that has a secret entry point
- ❖ **Information leaks in a program:** code that makes information accessible to unauthorized people or programs



- ❖ More people know how to use or interpret data
- ❖ Confidential data leaked to a competitor may narrow a competitive edge
- ❖ Data incorrectly modified can cost human lives
- ❖ Inadequate security may lead to financial liability if certain personal data are made public



- ❖ Computer items must be protected only until they lose their value
- ❖ They must be protected to a degree consistent with their value



- ❖ Harm occurs when a threat is realized against a vulnerability
- ❖ To protect against harm, we can neutralize the threat, close the vulnerability, or both
- ❖ The possibility for harm to occur is called risk



We can deal with harm in several ways.

We can seek to

- ❖ Prevent it, by blocking the attack or closing the vulnerability
- ❖ Deter it, by making the attack harder
- ❖ Deflect it, by making another target more attractive
- ❖ Detect it, either as it happens or some time after the fact
- ❖ Recover from its effects



- ❑ **Internal program controls:** parts of the program that enforce security restrictions, such as access limitations
- ❑ **Operating system and network system controls:** limitations enforced by the operating system or network to protect each user from all other users



- ❑ **Independent control programs:** application programs, such as password checkers, intrusion detection utilities, or virus scanners, that protect against certain types of vulnerabilities
- ❑ **Development controls:** quality standards under which a program is designed, coded, tested, and maintained, to prevent software faults from becoming exploitable vulnerabilities



❑ Principle of Effectiveness:

- Controls must be used—and used properly—to be effective.
- They must be efficient, easy to use, and appropriate



- An **operating system** is the software on a computer that **manages** the way different programs use their hardware, and regulates the ways that a user **controls** the computer.
- **Operating systems** are found on almost any device that contains a computer with multiple programs—from cellular phones and video game consoles to supercomputers and web servers.
- Some popular **modern operating systems** for personal computers include Microsoft Windows, Mac OS X, and Linux.



Operating system functions can be categorized as:

- access control.
- identity and credential management.
- information flow.
- audit and integrity protection.
- Each of these activities has security implications.



what kinds of things operating systems can protect ?

what methods are available for protecting them?

- Once upon a time, there were no operating systems: Users entered their programs directly into the machine in binary.
- The first operating systems were simple utilities, designed to assist individual programmers and to smooth the transition from one user to another.



Operating systems took on a much broader role as the notion of multiprogramming was implemented

- two users could interleave access to the resources of a single computing system.
- researchers developed concepts such as scheduling, sharing, and parallel use.
- protecting one user's programs and data from other users' programs became an important issue in multi-programmed operating systems.



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



- An operating system **manages** and **controls** access to hardware components
- **Older** operating systems focused on ensuring data confidentiality
- **Modern** operating systems support four basic functions
 - **Positively identify a user**
 - **Restrict access to authorized resources**
 - **Record user activity**
 - **Ensure proper communications with other computers and devices (sending and receiving data)**



- **Authentication of users**
 - password comparison
- **Protection of memory**
 - user space, paging, segmentations
- **File and I/O device access control**
 - access control matrix
- **Allocation & access control to general objects**



- **Enforcement of sharing resources**
 - To preserve integrity and consistency (critical section)
- **Fair service**
 - no starvation and deadlock
- **Inter-process communication & synchronization**
 - Shared variable (e.g, using semaphores)
- **Protection of data**
 - encryption, isolation



- Identification and Authentication
- Mandatory (enforce multilevel security by classifying the data and users into various security classes) and
- Discretionary Access Control (grant privileges to users)
- Object use and reuse protection (Subject and object)
- Anti-virus scan
- Accountability and Audit (security log)
- Firewall
- Intrusion detection (patterns of normal system usages, anomalies)



- Do not protect
- Isolate
- Share all or share nothing
- Share via access limitation
- Share by capabilities
- Limit use of an object



- **Do not protect.** Operating systems with no protection are appropriate when sensitive procedures are being run at separate times.
- **Isolate.** When an operating system provides isolation, different processes running concurrently are unaware of the presence of each other. Each process has its own address space, files, and other objects



- **Share all or share nothing.** With this form of protection, the owner of an object declares it to be public or private.
- **Share via access limitation.** With protection by access limitation, the operating system checks the allow ability of each user's potential access to an object.



- **Share by capabilities.** An extension of limited access sharing, this form of protection allows dynamic creation of sharing rights for objects. The degree of sharing can depend on the owner or the subject, on the context of the computation, or on the object itself.



- **Limit use of an object.** This form of protection limits not just the access to an object but the use made of that object after it has been accessed. **For example**, a user may be allowed to view a sensitive document, but not to print a copy of it.



The basis of protection is separation

separation: keeping one user's objects separate from other users

- physical separation
- temporal separation
- logical separation
- cryptographic separation



- **Physical separation**, in which different processes use different physical objects, such as separate printers for output requiring different levels of security.
- **Temporal separation**, in which processes having different security requirements are executed at different times



- **Logical separation**, in which users operate under the illusion that no other processes exist, as when an operating system constrains a program's accesses so that the program cannot access objects outside its permitted domain.
- **Cryptographic separation**, in which processes conceal their data and computations in such a way that they are unintelligible to outside processes



- The first two approaches are very stringent and can lead to **poor resource utilization**.
- Therefore, we would like to shift the burden of protection to the operating system to allow concurrent execution



- We want to **separate users and their objects**, but we also want to be able to provide **sharing** for some of those **objects**.
- An operating system can support separation and sharing in several ways, offering protection at any of several levels.



➤ The kernel

- The core of the operating system
- Includes a monolithic core and modular components

➤ The distribution

- A unified collection of applications, services, drivers, and libraries configured with the kernel
- Managed by either a company or a group of volunteers



➤ Kernel philosophy

- Kernel is robust
- Includes a core and supplemental modular components
- Monolithic part of kernel contains drivers and options essential to the kernel boot process
- Modular part of kernel adds everything else needed for smooth operation

➤ Basic kernels

- Development work is managed by the Kernel Organization
- Source code for the kernel can be acquired at <https://kernel.org>
- Building a custom kernel requires source code



➤ Distribution-specific kernels

- Distribution-specific kernel has been compiled and built with the intended hardware in mind
- Name of the kernel file generally includes the version number as well as the hardware architecture
 - **Example:** Debian Linux has a kernel named `vmlinuz-5.10.0-8-amd64`

➤ Custom kernels

- Kernel is highly customizable
- Different configuration interfaces exist, including a text-based one using the `ncurses` library



- **Unified Extensible Firmware Interface (UEFI)**
 - Can introduce capabilities for attackers to insert themselves into the boot process
 - Allows for secure boot
- **Physical security**
 - Limit access to live CDs
- **The threat of the live CD**
 - Malicious users can boot functional version with full root administrative access
 - Examples: Knoppix, Ubuntu, CentOS



➤ Boot process security

- Grand Unified Boot (GRUB) loader
 - Unless password protected, the GRUB boot loader can be used to boot a system into single-user mode, with full administrative privileges
- Use nonstandard port numbers for key services
 - Virtual physical security
 - Anyone who gains access to a virtual machine can more easily change virtual physical components



- **Security issues with the graphical user interface (GUI)**
 - Programs written for GUIs have a larger set of libraries that get added during the build; more code that can be broken
 - Reduce the number of installed packages to reduce the potential for vulnerabilities



➤ **O.S security updates for:**

- Regular users
- Home hobbyists
- Power users
- administrators
- Update administration



- **This chapter explained :**
 - Kinds of the Vulnerabilities
 - How the Methods of Defense work
 - The Software Controls
 - The meaning of Protected Objects and Methods of Protection
 - The Protection Types



مجموعة جامعة
الجميع الذكية
للتعليم القابضة

MIDOCEAN
UNIVERSITY



Thank You

