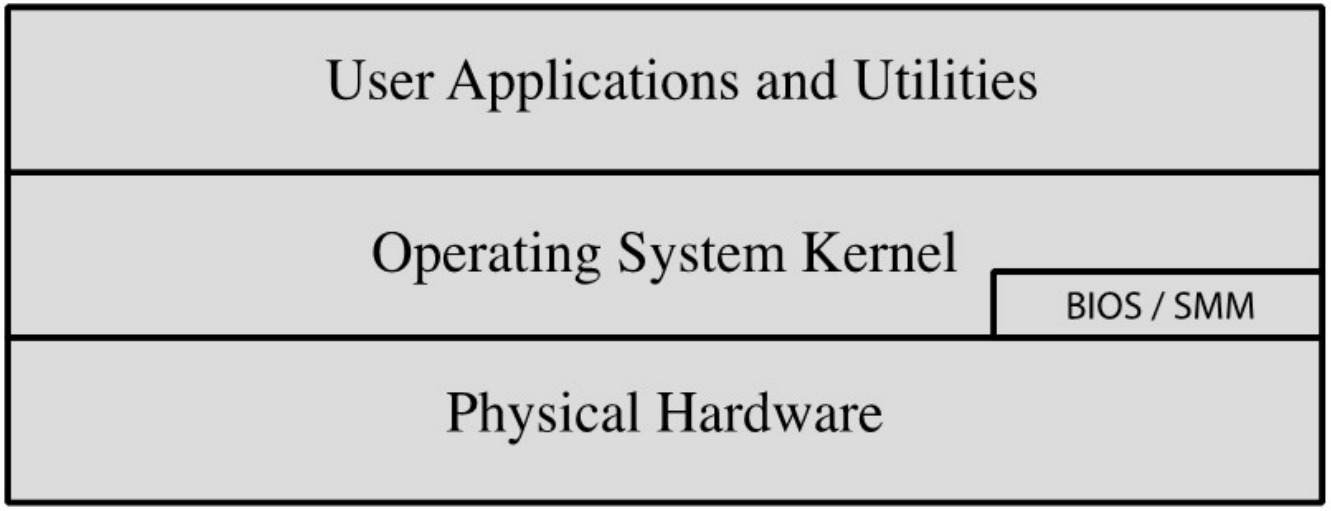
Security and Protection

LECTURE 10

Operating system security layers

- Each layer needs measures in place to provide appropriate security services
- each layer is vulnerable to attack from below if the lower layers are not secured appropriately



Security in Operating System

- Security refers to providing protection to computer system resources such as:
  - CPU, memory, disk
  - software programs and
  - most importantly data/information stored in the computer system.

- So a computer system must be protected against
  - unauthorized access by users and
  - malicious access to the system including viruses, worms, etc.…

What are the vulnerabilities?

- Physical vulnerabilities (Computer/ Hard disks can be stolen)
- Natural vulnerabilities (Power Outages: Power failures can cause interruptions in the functioning of an OS)
- Hardware and Software vulnerabilities (Failures, Software bugs, etc)
- Communication vulnerabilities (Lack of Encryption, Weak Authentication, etc)
- Human vulnerabilities (Eg. Insiders)
- Poorly chosen passwords

Protection

- Protection refers to a mechanism for controlling the access of programs, processes, or users to the resources defined by a computer system
- Modern protection concepts have evolved to increase the reliability of any complex system that makes use of shared resources
- Protection is the need to ensure that each program component active in a system uses system resources only in ways consistent with stated policies
- A protection-oriented system provides means to distinguish between authorized and unauthorized usage.

Principles of Protection

- A key, the time-tested guiding principle for protection is the **principle of least privilege**
- It dictates that programs, users, and even systems be given just enough privileges to perform their tasks
- For example, if a process halts, it should not halt the whole system
- Managing users with the principle of least privilege entails creating a separate account for each user, with just the privileges that the user needs

Domain of Protection

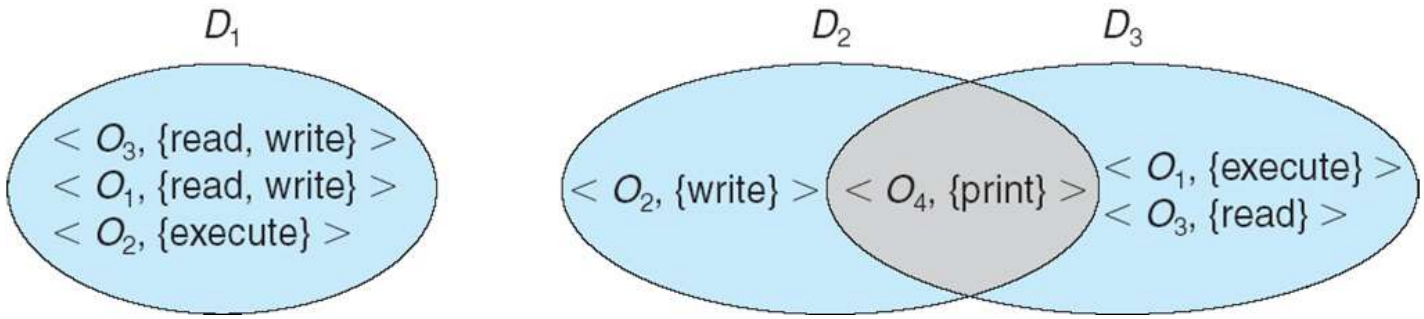
- A computer system is a collection of processes and objects
- Each object has a unique name that differentiates it from all other objects in the system, and each can be accessed only through well-defined and meaningful operations
- The operations that are possible may depend on the object. For example,
  - On a CPU, we can only execute.
  - Memory segments can be read and written

Domain of Protection

- A process should be able to access only those resources that it currently requires to complete its task
- It is useful in limiting the amount of damage a faulty process can cause in the system
- For example, when process **p** invokes procedure **A()**,
  - the procedure should be allowed to access only its own variables and the formal parameters passed to it
  - it should not be able to access all the variables of process p

Domain Structure

- Access-right = <object-name, rights-set>
  where rights-set is a subset of all valid operations that can be performed on the object
- Domain = set of access-rights



Domain Structure

- A domain can be realized in a variety of ways:
  - **Each user may be a domain**. In this case, the set of objects that can be accessed depends on the identity of the user.
  - **Each process may be a domain**. In this case, the set of objects that can be accessed depends on the identity of the process.
  - **Each procedure may be a domain**. In this case, the set of objects that can be accessed corresponds to the local variables defined within the procedure.

Access Matrix

- View protection as a matrix (access matrix)
- Rows represent domains
  - Domains are set of access rights
- Columns represent objects
- `Access(i, j)` is the set of operations that a process executing in $\texttt{Domain}_i$ can invoke on $\texttt{Object}_j$

Access Matrix

domain \ object	F_1	F_2	F_3	printer
D_1	read		read	
D_2				print
D_3		read	execute	
D_4	read write		read write	

Access Matrix

- Processes should be able to switch from one domain to another
- Switching from domain Di to domain Dj is allowed if and only if the access right `switch` ∈ `access(i, j)`

domain \ object	F_1	F_2	F_3	laser printer	D_1	D_2	D_3	D_4
D_1	read		read			switch		
D_2				print			switch	switch
D_3		read	execute					
D_4	read write		read write		switch			

Class Activity

- Given an access matrix as below. Please answer following questions.

object / domain	F_1	F_2	F_3	laser printer	D_1	D_2	D_3	D_4
D_1	read		read			switch		
D_2				print			switch	switch
D_3		read	execute					
D_4	read write		read write		switch			

If a user is in D2 and wants to read both F2 and F3, whether the user is able to do that? Please explain the reason.

Linux Permissions

- Linux determines whether a user or group has access to files, programs, or other resources on a system by checking the overall effective permissions on the resource
- The traditional permissions model in Linux is simple
- It is based on four access types or rules. The following access types are possible:
  - (r) Read permission
  - (w) Write permission
  - (x) Execute permission
  - (-) No permission or no access
- These permissions are applied to: Owner, Group and Everyone

Linux Permissions

Classes of File/Directory Users

Class	Definition
Owner	A file/directory has an owner. The owner is the user who created it.
Group	A file or directory has a group, which is group of users. The owner can use the chgrp command to switch the file's team to any other group of which the owner is a member.
Others	Others are everyone else!

Linux Permissions

File permissions

Class	Definition
Read	If a file has read permissions for its owner, its group, or others, then its owner, its group, or others can examine the contents of a file.
Write	If a file has write permissions for its owner, its group, or others, then its owner, its group, or others can change the contents of that file
Execute	If a file has execute permissions for its owner, its group, or others, then its owner, its group, or others can execute that file as a command. It makes sense to give a file execute permissions if and only if it contains executable code: executable binary code, a Bash shell script, a Python script, etc.

Linux Permissions

Directory permissions

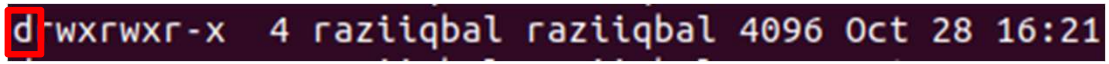
Permissions are interpreted differently for directories

Class	Definition
Read	It allows listing names of files in directory, but not their properties like size and permissions
Write	It allows creating and deleting files within the directory
Execute	it allows entering the directory and getting properties of files in the directory (**command ls -l**)

Linux Permissions

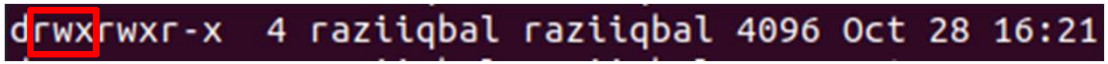
```
drwxrwxr-x  4 raziiqbal raziiqbal 4096 Oct 28 16:21
```

Linux Permissions



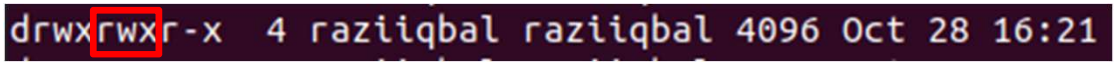
File / Directory

Linux Permissions



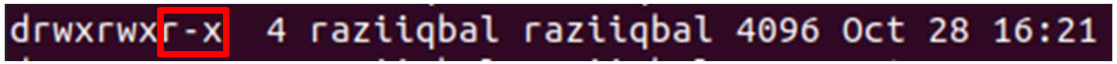
Owner Permissions

Linux Permissions



Group Permissions

Linux Permissions



Other / Everyone else Permissions

Linux Permissions

In order to give permissions, we set the flag to 1, else 0

Owner Permissions	Group Permissions	Others Permissions
r w x	r w x	r w x

Linux Permissions

Permissions Examples (Regular Files)

-rw-r--r--	read/write for owner, read-only for everyone else
-rw-r-----	read/write for owner, read-only for group, forbidden to others
-rwx------	read/write/execute for owner, forbidden to everyone else
-r--r--r--	read-only to everyone, including owner
-rwxrwxrwx	read/write/execute to everyone

Linux Permissions

Permissions Examples (Directories)

drwxr-xr-x	all can enter and list the directory, only the owner can add/delete files
drwxrwx---	full access to owner and group, forbidden to others
drwx--x---	full access to owner, group can access the directory and get properties of files in the directory , forbidden to others
-r--r--r--	read-only to everyone, including owner
-rwxrwxrwx	full access to everyone

Linux Permissions

- Change Permissions
  - Use the chmod command to change the permission

```
Usage: chmod [OPTION]... MODE[,MODE]... FILE...
```

- Change ownership
  - Use the chown command to change the ownership

```
Usage: chown [OPTION]... [OWNER][:[GROUP]] FILE...
```

Linux Permissions

- There are two ways to set permissions when using the chmod command:

Symbolic mode: testfile has permissions of -r--r--r–

		u g o
$ chmod g+x testfile	==>	-r--r-xr--
$ chmod u+wx testfile	==>	-rwxr-xr--
$ chmod ug-x testfile	==>	-rw-r--r--

u=user, g=group, o=other

Linux Permissions

We use the values represented like this:

Letter	Permission	Value
R	Read	4
W	Write	2
X	Execute	1
-	None	0

For each column, User, Group or Other you can set values from 0 to 7. Here is what each means

0= --- 1= --x 2= -w- 3= -wx

4= r-- 5= r-x 6= rw- 7= rwx

Linux Permissions

Example index.html file with typical permission values:

```
$ chmod 755 index.html
$ ls -l index.html
        -rwxr-xr-x    root   wheel  0  May 24 06:20  index.html
                      ----------------------------
$ chmod 644 index.html
$ ls -l index.html
        -rw-r--r--   root       wheel      0      May        24 06:20         index.html
```

- **-rwxr-xr-x**: This part represents the permissions. Each character represents a specific permission for a user category
- **root:** This indicates the owner of the file. In this case, the owner is the root user.
- **wheel:** This indicates the group assigned to the file. In this case, the group is "wheel".
- **0:** This represents the file size in bytes.
- **May 24 06:20**: This represents the date and time the file was last modified.
- **index.html:** This is the name of the file.

Operating System Security

- System is considered secure if resources are used and accessed as intended under all circumstances
- Threat is a potential security violation
- Attack is an attempt to breach security
- Attack can be accidental or malicious

Security Violations

- Breach of confidentiality
  - Unauthorized reading of data
- Breach of integrity
  - Unauthorized modification of data
- Breach of availability
  - Unauthorized destruction of data
- Theft of service
  - Unauthorized use of resources
- Denial of service (DOS)
  - Prevention of legitimate use

Security Violations Methods

- Masquerading (breach authentication)
  - Pretending to be an authorized user to escalate privileges
- Replay attack
  - As is or with message modification
- Man-in-the-middle attack
  - Intruder sits in the data flow, masquerading as sender to receiver and vice versa
- Session hijacking
  - Intercept an already-established session to bypass authentication

Security Measure Levels

- Security must occur at four levels to be effective:
  - Physical
    - Data centres, servers, connected terminals
  - Human
    - Avoid social engineering, phishing
  - Operating System
    - Protection mechanisms
  - Network
    - Intercepted communications, interruption, DOS

Program Threats

- **Trojan Horse**
  - Code segment that misuses its environment
  - they rely on social engineering techniques to deceive users
  - Perform various malicious actions, such as stealing sensitive information, damaging files, or enabling other types of malware to be installed.

Program Threats

- Viruses
  - Code fragment embedded in a legitimate program
  - Self-replicating, designed to infect other computers
  - Very specific to CPU architecture, operating system, applications
  - Usually borne via email or as a macro

Program Threats

- Logic Bomb
  - Program that initiates a security incident under certain circumstances
- Stack and Buffer Overflow
  - Exploits a bug in a program (overflow either the stack or memory buffers)
  - An attacker intentionally writes more data into a buffer than its allocated size, causing the excess data to overflow into adjacent memory locations.
  - Failure to check bounds on inputs, arguments
  - Unauthorized user or privilege escalation

Program Threats

- Attacks still common, still occurring
- Attacks moved over time from science experiments to tools of organized crime
  - Targeting specific companies
  - Creating botnets to use as a tool for spam and DDOS delivery
  - Keystroke logger to grab passwords, credit card numbers, or other confidential data entered by users
- Why is Windows the target for most attacks?
  - Most common
  - Attackers often have a better understanding of Windows systems

System and Network Threats

- Network threats are harder to detect, prevent
  - Protection systems weaker
  - More difficult to have a shared secret on which to base access
  - No physical limits once the system is attached to the internet

System and Network Threats

- **Worms**
  - Differ from viruses in that they do not attach to a host file, but are self-contained programs that propagate across networks and computers.
  - Worms are commonly spread through email attachments; opening the attachment activates the worm program.
  - A typical worm exploit involves the worm sending a copy of itself to every contact in an infected computer's email address
  - In addition to conducting malicious activities, a worm spreading across the internet and overloading email servers can result in denial-of-service attacks against nodes on the network.

System and Network Threats

- **Port scanning**
  - Automated attempt to connect to a range of ports on one or a range of IP addresses
  - Detection of answering service protocol
  - Detection of OS and version running on system
  - nmap scans all ports in a given IP range for a response
  - Frequently launched from zombie systems
  - To decrease trace-ability

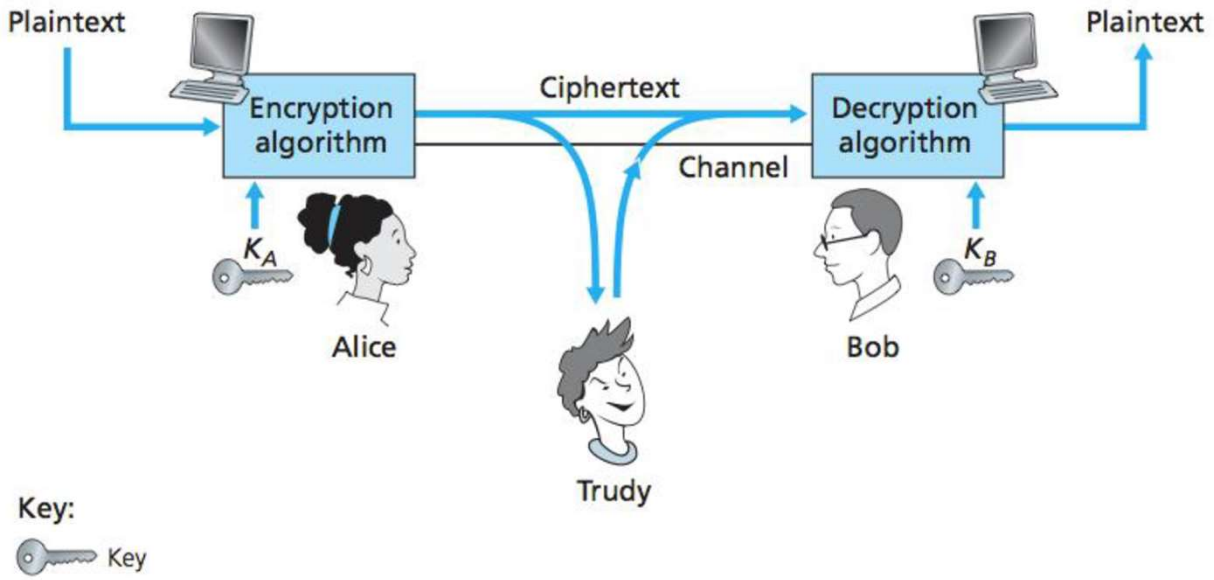
System and Network Threats

- **Denial of Service**
  - Overload the targeted computer preventing it from doing any useful work
  - Distributed denial-of-service (DDOS) comes from multiple sites at once
  - Accidental – CS students writing bad fork() code
  - Purposeful – extortion, punishment

Cryptography

- Cryptography techniques allow a sender to disguise his data from an intruder.
- Even if an intruder captures that data, it is encrypted.
- The receiver after receiving that data will use cryptographic techniques to decrypt that data.
- This highly encrypted data using cryptography is considered very secure and is used in many of modern network security systems.

Cryptography



Cryptography

- Encryption Algorithm
  - An algorithm used at sender to encrypt the data.
- Decryption Algorithm
  - An algorithm used at receiver to decrypt the data.
- Cipher text
  - An encrypted form of a plain text message using encryption algorithm is called Cipher Text.
- Key
  - Inputs provided to encryption and Decryption algorithm.

Cryptography

- Two popular cryptography techniques are
  - Symmetric Key Encryption
    - These keys are identical and secret.
  - Public Key Encryption
    - Two keys are used
    - One key is known
    - One key is unknown

Cryptography

Symmetric Key Encryption

- In this techniques each computer has a secret key (code) that it can use to encrypt a packet of information before it is sent over the network to another computer.
- In this techniques, normally sending and receiving computers know each other.
- Symmetric key encryption is essentially the same as a secret code that each of the two computers must know in order to decode the information.

Cryptography

Asymmetric Key Encryption

- It is also known as public key encryption.
- In this technique two keys are used; public and private key.
- Sender uses the public key of a receiver and uses it to encrypt the data.
- Receiver after receiving this encrypted text, uses his private key to decrypt the data.

Firewalls

- A firewall is a hardware device or software program that inspects packets going into or out of a network or computer, and then discards or forwards these packets based on a set of rules.
- A software firewall is installed in an OS and inspects all packets coming into or leaving the computer.
- Based on predefined rules, the packets are discarded or forwarded for further processing.