Operating Systems

40201341

2023-2024 Summer

Dr. Sultan Al-Rushdan

Eng. Yazan Al-Shannik

By: Nasrullah Rami Hamad

Section (3)

Contents

[Task 1: 3](#_Toc176277688)

[Task 2: 6](#_Toc176277689)

[Task 3: 6](#_Toc176277690)

[Task 4: 7](#_Toc176277691)

[Task 5: 7](#_Toc176277692)

[Task 6: 9](#_Toc176277693)

[Task 7: 9](#_Toc176277694)

[Task 8: 10](#_Toc176277695)

[Task 9: 11](#_Toc176277696)

[Task 10: 12](#_Toc176277697)

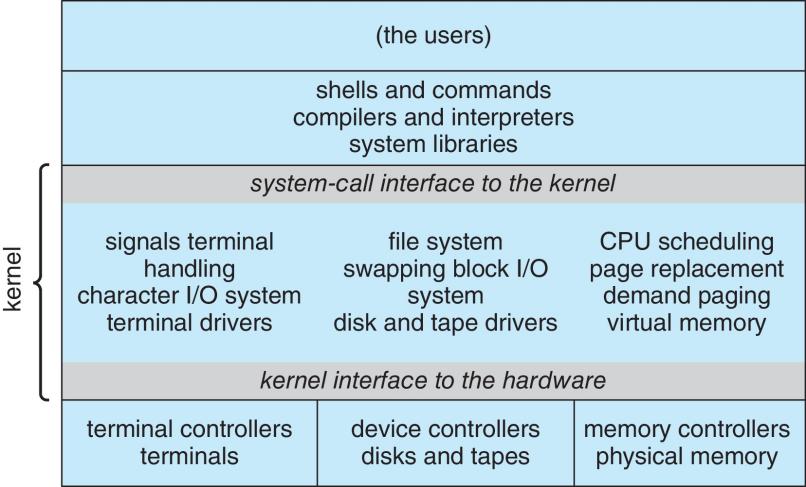
# Task 1:

An operating system (OS) is a software that helps manage resources (Hardware & Software). It acts as an intermediary between computer users and the computer hardware itself. Its goals are to execute programs and solve user problems faster, and make the systems convenient and easy to use for users. While making efficient use of the hardware.

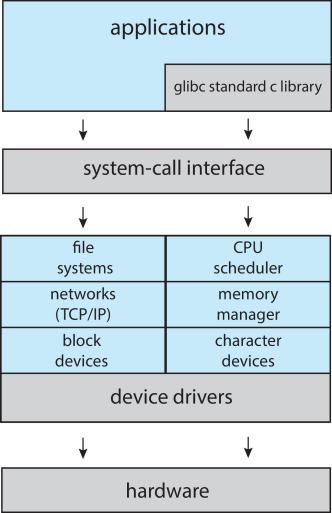
The structure of such operating systems varies based on its design, task and device used, to Monolithic, layered, Microkernel, Module, and hybrid structures. Each structure is used for different purposes and for different memory usage.

1. Monolithic structure (Orinigal UNIX).

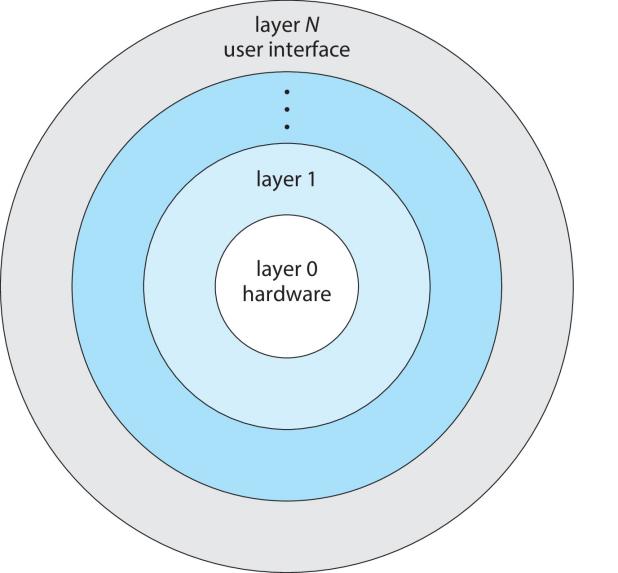
This structure is known for its limited functionality in hardware since the UNIX had a poor structure implementation. It consists of system programs and the kernel (Core of the OS). The kernel maintains and manages all core operations of an OS, such as handling I/O systems, swapping blocks, and CPU scheduling, and more.



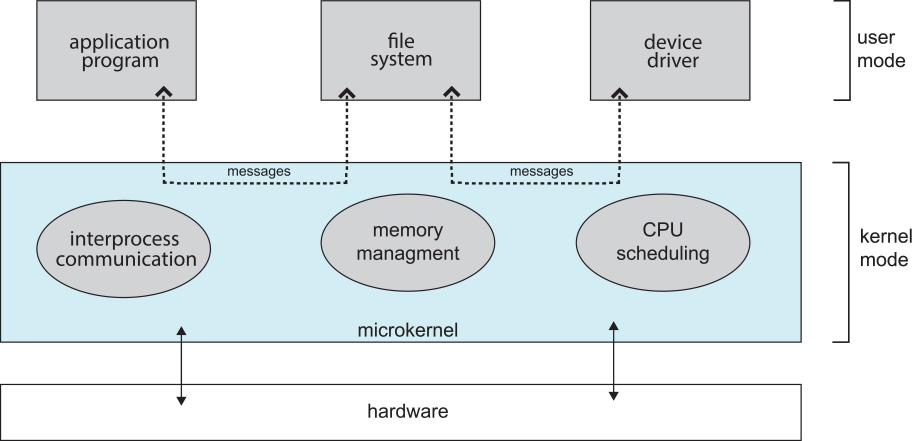
The Linux OS uses a modular monolithic structure.



1. Layered approach, is another OS structure that divides the OS into number of layers or levels, each level is built on top of the lower levels. The bottom layer is the hardware structure and the top layer is the UI. Each layer has functionality that supports the execution of the operating system and its effciency.



1. Microkernels are another operating systems structure, it differs from other structures because only essential processes are run from the kernel into the user space, it uses message passing as a way to communicate between users. Its known for its efficiency and scalability to extend, and its reliable and secure. But it can suffer from performance overhead and decreased performance because of context switching.



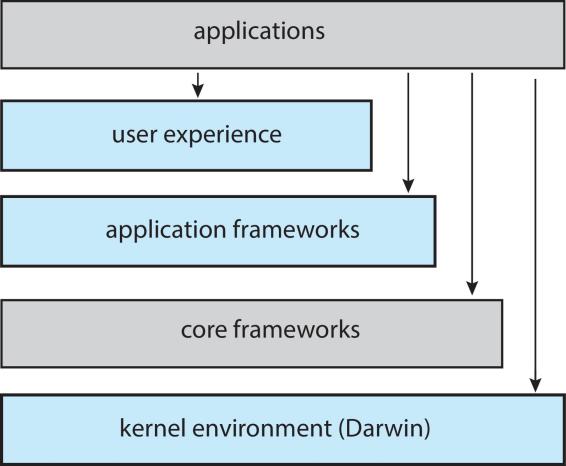
1. Modules, or LoadabLe Kernel Modules (LKMs) are one of the five OS structures, this type is known for its flexibility and its OO use. In thIs structure, each core component Is separate and their communication is done over known interfaces. It is similar to the layered approach but is more flexible. Linux and Solaris are few OS examples that use this structure.
2. Hybrid systems are thelast OS structure, and it is what most modern operating systems are made with. It combines several approahes and structures to counter performance overhead issues, and to increase security and usabiliy needs.It usually combines monolihic and microkernels in structure, such as the Windows operating system. It has a balanced performance and it contains dynamic loading and modularity. But it can be complex and costly to implement.

Examples of current operating systems are:

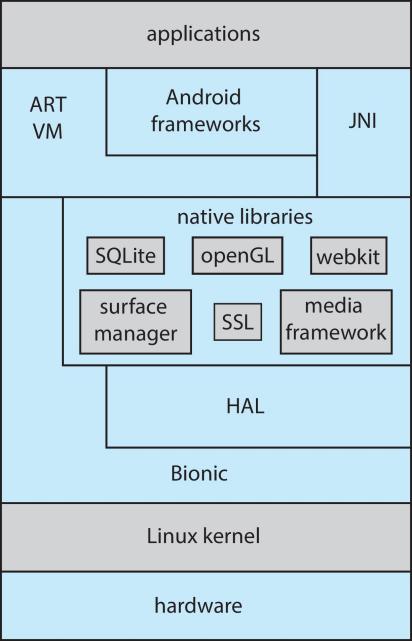
- Windows: which uses a hybrid approach for its devices, it provides a comfortable UI and contains levels of high security within the operating system. It uses monolithic structures and sometimes microkernels for some subsystems.

- Linux: this OS is an open-source operating system, it consists of a hybrid approach and is used in servers, laptops and some virtual machines. It is known for its strength in running programs and codes, and its flexibility.

- MacOS: this OS is a Unix-based hybrid operating system, designed for Apple device, and it is used in most Mac devices. It focuses mostly on integration with hardware devices and user experence.



- Android: Android is an open-source mobile operating system, it is similar to iOS and is based on a modified version of Linux, that provides process, memory, and driver management. Its runtime environment includes core sets of different libraries, some are developed in Java and Android API.



# Task 2:

**Paging** is a way that enhances how memory is stored inside the OS, without using the contiguous allocation (Base, limit Registers). It divides the physical memory into blocks of memory called frames (Their size is power of 2, commonly between 512B and 16MB), and the logical memory into pages. It contains a page table which maintains the translation from a logical address to a physical address, it also keeps track of the frames and their corresponding pages.

The process requests memory access, then the logical address is divided into page numbers and an offset depending on the page size and the process size. The page number can be used as an index to know which frame is connected to it on the page table, and the offset is added to get the real physical address space.

The paging method works on solving two problems that the previous methods suffered from, external fragmentation and the varying sized memory chunks given randomly. It is also considered a method of high security since it isolates the memory spaces of processes. Because each process has its own page table/s. Here’s how:

- External fragmentation: Paging avoids external fragmentation by using fixed-sized frames, which are used to efficiently utilize all the process size memory locations inside the frames. Unlike the previous method which contained an underutilized memory space.

- Since frames and pages have fixed sizes, they can be efficiently used without the varying sized chunks of memory that were given to different processes in previous methods, it gives a better utilization of memory and a simple memory management process.

- Security is one of paging’s powerful characteristics, since it enhances processes’ memory allocations by each having their isolated page table without memory sharing unless allowed. This mitigates the risk of malicious intent and overuse of memory.

# Task 3:

1. System calls are a programming interface for the OS services, they’re commonly written in high-level languages such as C or C++. They allow the processes to request services from the OS which operates on low-level language so that they can run smoothly and effectively. They provide readings of files, creation of parent and child processes, and more.

System calls are usually followed by a number, this number can be indexed inside the system-call interface, system calls are implemented by processes (User-mode) requesting a specific service from the OS (Kernel mode), the OS then checks the number of the needed system call and executes the system call, then returns the status of the system call (Success or Fail) and any return values.

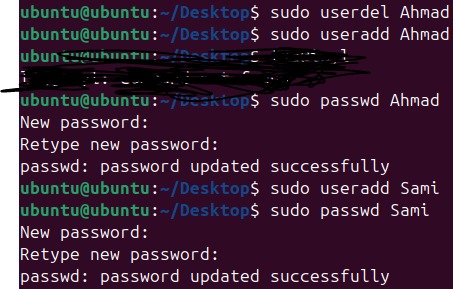
1. System calls are accessed by programs through calling the system calls (functions) while obeying the API and understanding what values will be returned by the OS, for example: the user wants to create a child process, so they call the fork () function. This will trigger a request to the kernel (OS) and execute it, while returning a child process to the user/program.
2. Parameters are passed to system calls by three methods:
   * Using registers to pass the parameters, which can be read and executed by the OS. Sometimes, parameters may exceed the number of registers.
   * Storing the parameters in blocks, tables, memory, combined with the address of the block passed as a parameter (Pointer to the block), this method is used by Linux.
   * They can also be placed in stacks by the program and then out of the stacks by the OS to execute them. This method doesn’t limit the number of parameters passed.
3. There are many types and examples of system calls used in modern operating systems, such as.
   * Process control system calls (wait, create process, abort).
   * File management system calls (create, delete, open, read).
   * Process communication system calls (send, receive, attach).

# Task 4:

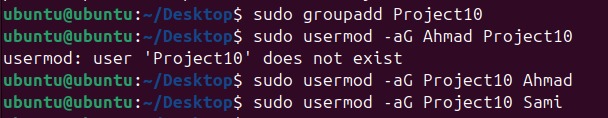
***The C program was done in Ubuntu (Linux) and sent via E-learning***

# Task 5:

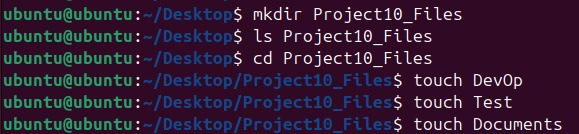
1. Create the employees "Ahmed" and "Sami" accounts with a strong password and with heightened security configurations.



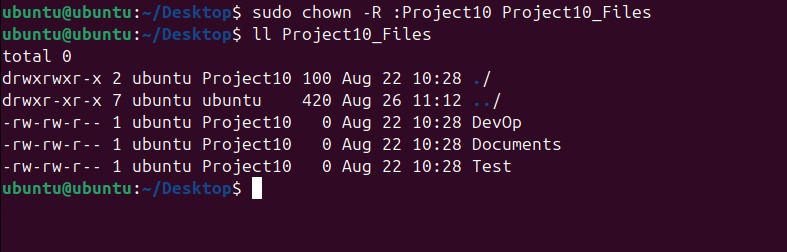
1. Form a new project team named "Project10" containing "Ahmed" and "Sami"



1. Prepare a dedicated directory called "Project10\_Files" for the project, and it must include three files: 'DevOp', 'Test', and 'Documents' within the directory.



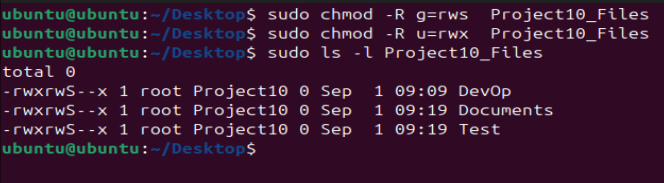
1. Verify and set correct group ownership for the "Project10\_Files" directory and its contents



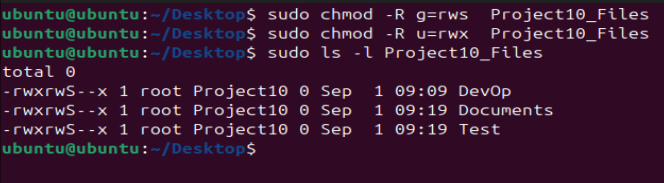
1. Ensure the "Project10" group has necessary access for seamless collaboration while maintaining security standards.



1. Apply the following security policies:
2. Grant comprehensive permissions to "Ahmed" and "Sami" for full read and write capabilities on files and directories.
3. for groups and others, implement a restrictive access, allowing only execute permission.



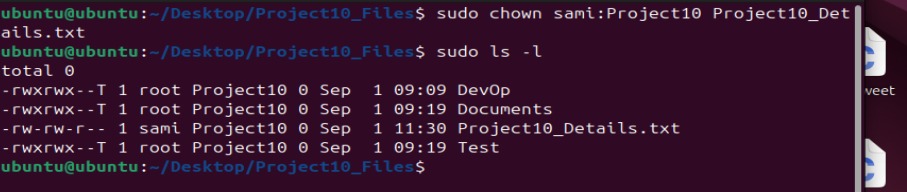
1. Validate implemented permissions to ensure alignment with team's operational requirements.



1. As "Ahmed" verify access to files in the "Project10\_Files" directory according to established permissions.



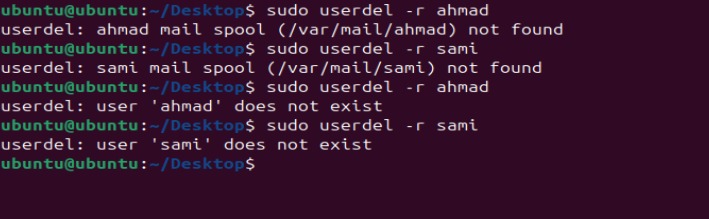
1. Ensure "Sami" has access to a restricted file named "Project\_Details.txt" in the "Project10\_Files" directory without changing its access rights or revealing its contents to the group owner



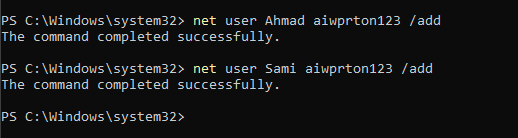
1. Ensure employees can create and modify their own files within the " Project10\_Files " directory while preventing them from deleting files owned by other employees.

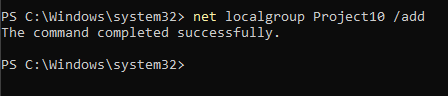


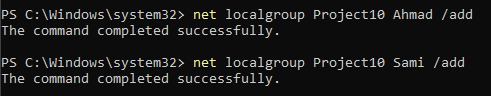
1. Remove the accounts of "Ahmed" and "Sami" along with their associated directories upon completion of the project.

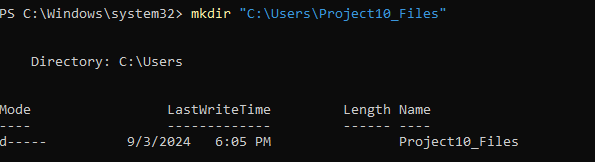


# Task 6:

****

1. 

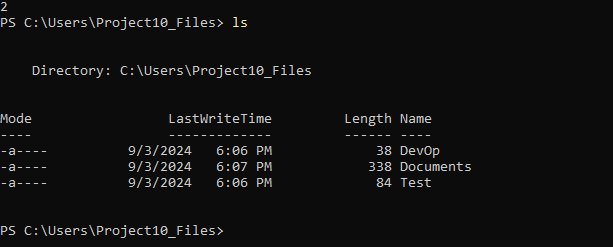


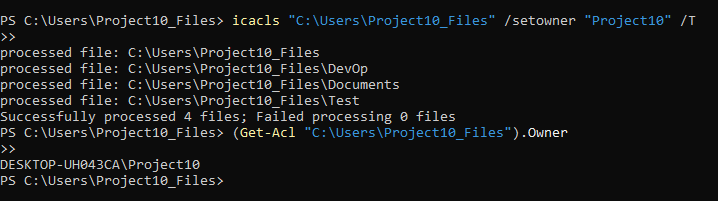












# Task 7:

**Distributed Operating Systems:**

The distributed operating systems (DOS) are a type of operating systems that run on a number of devices that are usually independent, but they are used by one user or system. Each device has its own operating system, when merged together, they can provide a high performance for tasks and projects users need them for.

Its structure is simple, where there is an operating system installed on each device, and the distributed services such as printers are transparent and accessible on the network by other devices.

In this structure, the resources and processes are global to use by the other operating systems or devices inside the network, the communication between such devices can be done through message passing or shared memory.

**Network Operating Systems:**

The network operating system (NOS) is another type of operating systems that is used for independent work, where there are independent devices, each with its operating system, but they can communicate with each other by allowing the network to enable communication and resource sharing between the independent computers and operating systems.

The structure of such operating systems is simple, each device has its own operating system, which can be different from other operating systems that other devices own (Linux, Windows). The operating systems provide some applications and features such as FTP, and HTTP services.

In NOS, the resources are allocated by the same device, where its independent from other resources. The communication may be through file passing.

**Middleware**:

Middleware can be located between both the operating system and other applications ran by the user. It maintains high transparency that allow the operating systems on other devices to not know about other devices. It also allows distributed applications to be run on different operating systems

The structure of middleware provides many services that run on top of the operating systems on each device, such as communication services, information system services, control services and more.

Middleware’s model of operation works by hiding the differences between operating systems from others, so that applications can be created and operate across different operating systems.

# Task 8:

Concurrency in operating systems is a method that allows multiple processes or threads to run at the same time, or to make it seem like it is. It maximizes CPU usage and provides efficient management for modern operating systems.

It is achieved by three approaches

- Multithreading, which allows multiple threads to be created within a process, and they are executed concurrently while sharing the same memory space but are doing other jobs. Windows and Linux use multithreading.

- Multicore programming is also a method that takes advantage of multicore processors to run the threads in parallel which provides parallelism for the tasks, it improves the performance of the operating systems.

- Last approach is the synchronization, which is a way to reduce race conditions and prevent them from happening. OSes use mutexes, semaphores, and locks to perform synchronization between processes. They ensure that one thread is running at a time. For data protection and.

It can impact the operating systems by allowing better utilization of CPU, and faster performance for tasks, while improving the responses of applications and other processes. But it can also lead to excessive context switching, and deadlocks. Which can cause the operating systems to crash and lose memory.

Concurrency in distributed systems can involve many processes or threads to execute at once across different devices inside a network. This can be complex to deal with, as the processes are in need of communication and synchronization for their connections and tasks.

Concurrency in distributed systems can be achieved by three approaches:

- Message passing, which is often used with distributed systems to facilitate communication with processes on devices. This can achieve concurrency on the specific network.

- Dividing the tasks and processes on different nodes and devices can lead to effective concurrency, since it can enable improved performance, yet, it may require continuous management to keep the concurrency going.

- Middleware can help provide concurrency management, by allowing the middleware distributed systems to handle communication, synchronization and fault tolerance.

It can impact the distributed systems positively by improving the scalability and fault tolerance of such systems, since they can distribute and divide tasks on other devices and OSes. But it can lead to synchronization overhead, and may lead to deadlocks or race conditions that are harder to detect in a distributed environment.

# Task 9:

RPCs (Remote Procedure Call) is used in distributed operating systems and enables the execution of programs that have procedures in them. It allows programmers to build distributed systems without worrying about complex communication.

Its way of working is when the client writes a procedure on a system, it would hide the details of the communication for the network. Which would make the remote procedure appear as a call to the programmer.

It can be facilitated to work under distributed systems by allowing them to manage their complex communications in different ways:

- It enables transparency for the network communication which allows processes from different operating systems and devices to communicate safely.

- It simplifies the communication between the procedure and the programmer, by since it allows to make procedure calls to handle procedures instead of message passing mechanisms, which allows lower errors and increases straightforwardness for applications.

- It can facilitate different operating systems to communicate and execute procedures that are on different operating systems, if they are on the same RPC protocol. This allows an application on Unix to be called on a Windows device or server.

Parameter passing in RPCs involves multiple steps that ensure data is not lost between the client and the server, since they may run on different platforms with different formats of data.

1. Marshalling is conducted to the parameters so that they can be transmitted over the network safely by transforming them into a data format easier to handle (Sequence of bytes).
2. Transmission is the method where after marshalling the parameters, they are sent over to the server, the RPC handles the details of this communication while it reaches its destination.
3. After reaching the destination, Unmarshalling is done to return the data back to its original format so that it can be read by the server. Then, the execution of the procedure is done as if it was a call.
4. Returning to the client is done by the same steps, marshalling, transmission, and unmarshalling. This ensures that the procedure was called and executed safely and effectively and back to the client (Programmer).

# Task 10:

Hypervisor is one of the major operating systems used continuously in data centers, it acts as a monitor for the virtual machines, which allows multiple operating systems to run on a single device, while managing the resources that each operating system needs and takes (CPU, storage, and memory). Using hypervisor ensures safety and isolation for the operating systems, and improves their efficiency.

It provides many services that help signify the virtualization method such as:

1. Resource management: since it monitors the resources needed for each virtual machine to work, hypervisor can manage the resources for the whole operating system and the virtual operating systems added, where each operating system takes the necessary resources to run, no more.
2. Isolation for the operating systems, since each virtual machine executes independently, so that any activity or task done in an OS cannot disturb the work of other virtual machines. Which ensures stability and secures the tasks inside each virtual machine.
3. It allows multiple virtual machines to execute on the same physical device, while optimizing the tasks and resource allocations for each virtual machine, this provides stability and efficiency for the virtual machines.
4. Security is provided for all virtual machines, since it restricts different virtual machines to see and use different data from unauthorized sources, this mitigates the problems and issues of security concerns and unauthorized access for the data, and provides data integrity.

As for its advantages for the datacenters and cloud computing:

1. Hypervisor works on resource optimization, since it allows different virtual machines to run on a single device, it maximizes the use of the resources, which would lower the need for hardware servers, this can result in power saving and cooling for the servers.
2. The environments monitored by hypervisors can be used in their full-potential, so that they are available always for tasks to come. It allows consistency during maintenance or errors. It also allows backups and checkpoints to avoid disaster loss of data and progress.
3. Hypervisor allows portability of virtual machines and ease of moving them across devices, which enables better resource allocation and better load balancing for the virtual machines. This ensures flexibility in datacenter operations.
4. Since hypervisors lower the need for physical servers, it can encourage adoption of cloud computing servers and virtualized datacenters for better security and resource management and speed.

In conclusion, hypervisor is a software that encourages virtualization for organizations by allowing them to handle multiple operating systems to run on a single physical device. Which provides increased resource allocation for the virtual machines, and better security and scalability since they become efficient and easy to use for modern cloud computing and datacenter operations. Hypervisor can help organizations monitor and optimize their infrastructure and need for IT and allows them to reduce the costs for the hardware servers.