Operating Systems:

**Student Id:** 21110214

**Submitted to:** Dr. Samer Suleiman

**Submitted by:** Saif Kamal Salim Haddad

**Submitted on:** 29/1/2024.

Table of Contents

[Task 1: 3](#_Toc157734766)

[Task 2: 11](#_Toc157734767)

[Task 3: 14](#_Toc157734768)

[Task 4: 21](#_Toc157734769)

[Task 5: 24](#_Toc157734770)

[Task 6: 31](#_Toc157734771)

[Task 7: 35](#_Toc157734772)

[Task 8: 42](#_Toc157734773)

[References: 48](#_Toc157734774)

[**Task 1 references:** 48](#_Toc157734775)

[**Task 2 references:** 48](#_Toc157734776)

[**Task 5 references:** 49](#_Toc157734777)

[**Task 6 references:** 49](#_Toc157734778)

[**Task 7 references:** 49](#_Toc157734779)

[**Task 8 references:** 50](#_Toc157734780)

# **Task 1:**

**Concept of Virtual Memory:**

virtual memory allows a computer to move data between RAM and other storage, such as a hard drive or SSD, to fill physical memory shortages. By doing this, the computer might appear to have more RAM than is really installed by treating secondary storage as a part of the main memory.

(Abraham Silberschatz and Peter Galvin, 2011) (GeeksforGeeks, 2024) (Csc-Knu, 2017) (Gillis et al., 2021)

**How it Works:**

(Abraham Silberschatz and Peter Galvin, 2011) (GeeksforGeeks, 2024) (Csc-Knu, 2017) (Gillis et al., 2021)

**1-Memory Mapping:**

The operating system divides memory into units called "pages" or "segments." These pages may be kept on a drive or in RAM.

**2-Page File or Swap Space:**

The operating system transfers certain pages to a page file, often known as swap space, on the hard drive when the RAM is full.

**3-Address Translation:**

The OS converts the virtual addresses of programs into physical addresses in RAM when an application requires data. The data is put into RAM, frequently displacing existing data, after being retrieved from the disk if it is not already there.

**Role in Memory Management Process of operating systems:** (Abraham Silberschatz and Peter Galvin, 2011) (GeeksforGeeks, 2024) (Csc-Knu, 2017) (Gillis et al., 2021)

**1-Executing Big Programs:**

Even in situations where physical RAM is limited, virtual memory enables the simultaneous operation of big programs or many applications. In order to prevent crashes caused by memory shortages, programs whose memory requirements above available RAM temporarily store sections of their programs in secondary storage and bring them into RAM as needed.

**2-Memory Isolation:**

Processes are isolated from one another using virtual memory, which stops them from accessing or changing each other's memory. By isolating problem apps, the OS and other users are less likely to be impacted, improving system security and stability.

**3-Multitasking:**

It makes multitasking effective. Programs can execute at once, spreading their memory over RAM and disk storage.

**4-Address Abstraction:**

Processes operating on the computer are separated from the actual memory by virtual memory. The virtual address space that each process is given is usually bigger than the amount of actual RAM that is accessible. This simplifies memory management for applications.

**5-Address Translation:**

Virtual addresses are mapped to physical locations via a memory management unit (MMU). The MMU converts a virtual address that a program accesses to the matching physical address, ensuring that data is retrieved from the right place in RAM or secondary storage.

**6-Utilizing RAM Efficiently:**

By storing frequently used code and data in physical memory and transferring less often used content to secondary storage, the operating system maximizes the use of RAM. By giving priority access to RAM to essential and running programs, this improves system responsiveness.

**Virtual Memory Enhancement of Overall System Performance:**

(Abraham Silberschatz and Peter Galvin, 2011) (GeeksforGeeks, 2024) (Csc-Knu, 2017) (Gillis et al., 2021)

**1-Increased Memory Capacity:**

The system may use more memory than is physically accessible because of virtual memory, which allows huge programs and multitasking environments to run smoothly without crashing from memory limitations.

**2-Optimized Resource Usage:**

Virtual memory optimizes resource use and responsiveness by dynamically transferring data between RAM and secondary storage to give priority access to fast memory to the most important and running programs.

**3-Efficiency:**

Virtual memory aids in greater RAM use by only loading the pages that are actually needed into RAM, leaving the other pages on the disk.

**4-Reduction of Physical Memory Needs:**

Systems are able to operate with less physical memory, which lowers the cost of hardware.

**5-Reduces I/O Blocking:**

Reduced waiting times for I/O operations result from programs' ability to continue working even when parts of their code or data are being swapped out to disk.

**Limitations:**

(Abraham Silberschatz and Peter Galvin, 2011) (GeeksforGeeks, 2024) (Csc-Knu, 2017) (Gillis et al., 2021)

**Speed:**

A system may become slower if it relies too much on virtual memory since retrieving data from disk storage takes longer than retrieving data from RAM.

**Minimum Disk Space Needs:**

Hard disk space is needed for virtual memory, and it may not be enough.

**A concept in virtual memory:** (Abraham Silberschatz and Peter Galvin, 2011) (GeeksforGeeks, 2024) (Csc-Knu, 2017) (Gillis et al., 2021)

**Demand paging:**

**Definition:**

It is a method of managing virtual memory that lets the operating system load a page into physical memory only when it's needed for operation.

**Some contexts in demand paging in virtual memory systems:** (Abraham Silberschatz and Peter Galvin, 2011) (GeeksforGeeks, 2024) (Csc-Knu, 2017) (Gillis et al., 2021)

**1-Hit:**

Hit happens when a page request is already present in the RAM (physical memory). When a page is requested by a process and the page table indicates that the requested page is available in RAM.

When a page hit happens, it means that the system can directly give the page to the process without requiring it to be read from the disk. This results in quicker data access times and enhanced system efficiency.

**2-Miss (Page fault):**

Page fault happens when a process asks for a page that isn't in physical memory now.

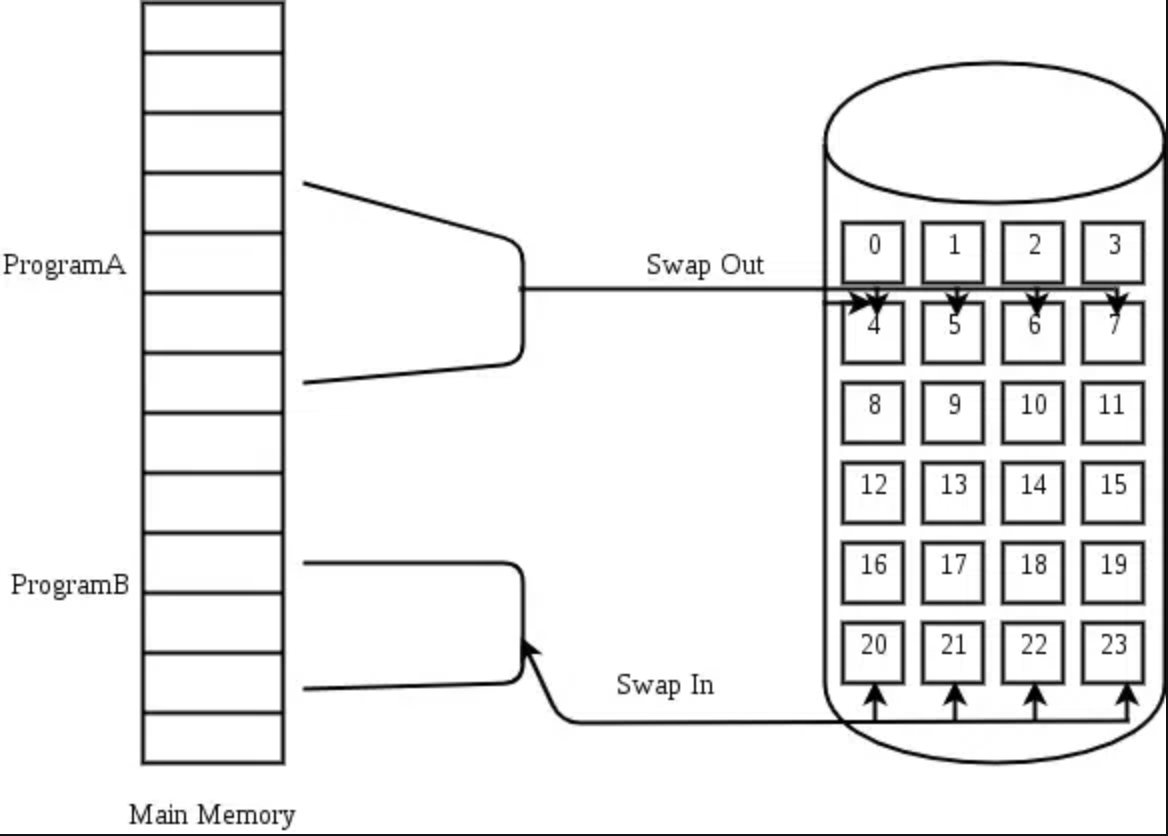
When a page fault happens, the page has to be found and loaded into RAM by the operating system from secondary storage, such as a hard drive or SSD. Because data access from the disk is slower than data access from RAM, this procedure takes longer than a page hit.

**Note:** As the number of page faults increases, much of the performance and efficiency of the demand page will be impacted.

**3-Swap:**

The swapping technique happens when a page fault happens and the physical main memory which is the RAM is full, so the operating system will manage this issue by making space for the new page, and this happens by swapping one of the already had pages from the RAM to the disk.

Information to consider that the decision of which page will be swapped with will be taken by a page replacement algorithm such as like Least Recently Used (LRU), First-In-First-Out (FIFO)



**Comparison between Hit and Miss and Swap and their impact on the performance:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Hit** | **Miss (Page Fault)** | **Swap** |
| **Performance Impact** | Excellent performance as a result of quick access. | Reduced efficiency since it takes longer to load pages from the disk. | Depending on the situation, performance can be maintained by efficient switching but significantly decreased by excessive swapping (thrashing). |
| **Access Time** | Since the data is already in RAM, it is quite quick. | slow; it reads information from the disk, which is quite a bit slower than RAM. | Depending on the device, switching can be done very quickly, but it's still slower than accessing RAM. |
| **Effect on System Performance:** | Beneficial effect; supports efficient and easy system operation. | Bad effect if repeated | may be harmful if done repeatedly. |
| **Usage of Resources:** | Efficient use of memory; no extra resource usage. | Increased use of disk I/O resources. | Uses both memory and disk resources. |
| **Impact on User Experience:** | Positive; leads to faster application response times. | Can negatively affect user experience due to delays. | Moderate to negative impact; noticeable effect on system responsiveness if frequent. |

(Abraham Silberschatz and Peter Galvin, 2011) (GeeksforGeeks, 2024) (Csc-Knu, 2017) (Gillis et al., 2021)

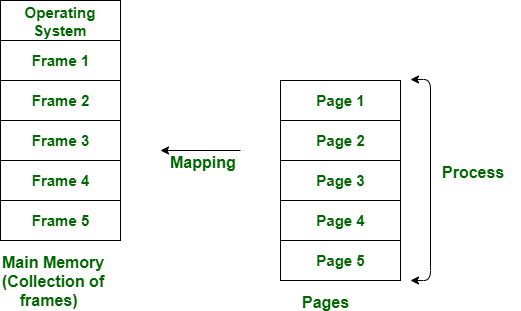
**Some types of virtual memory management:**

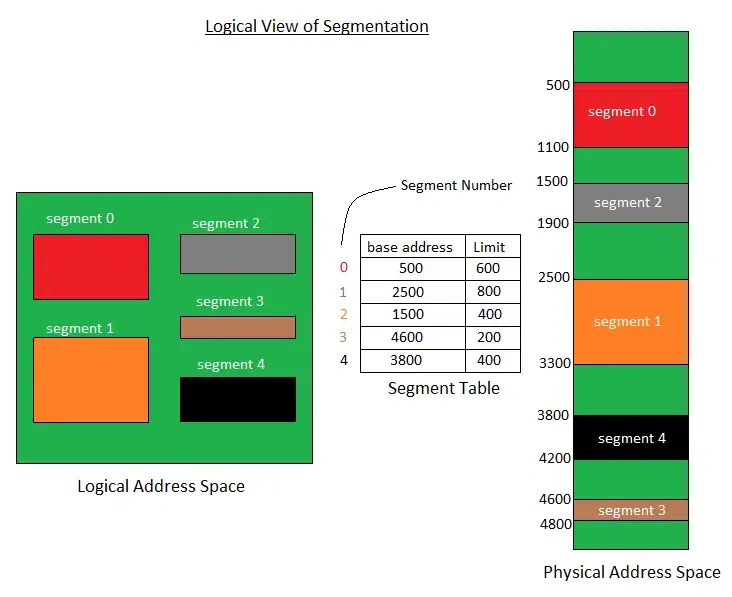
|  |  |
| --- | --- |
| **Type** | **Description** |
| **Paging** | It divides the memory into fixed-size pages and maps them using a page table. |
| **Segmentation** | It divides the memory into variable-sized segmentation and these segments will be tracked by a segmentation table. |
| **Paged Segmentation** | It is a combination of pages and segments, which means the memory will be divided into pages and segments. |
| **First-In-First-Out (FIFO) Page Replacement** | The oldest page in memory gets replaced first based on the page replacement policy. |
| **Optimal Page Replacement** | swaps out the page that will be unneeded for the longest period of time. |
| **Least Recently Used (LRU) Page Replacement** | swaps out the page that has been inactive for the longest. |

Performance could decrease as a result of frequent page failures and repeated swapping. The behavior is known as "thrashing", since the system spends more time switching between data and memory than executing tasks.

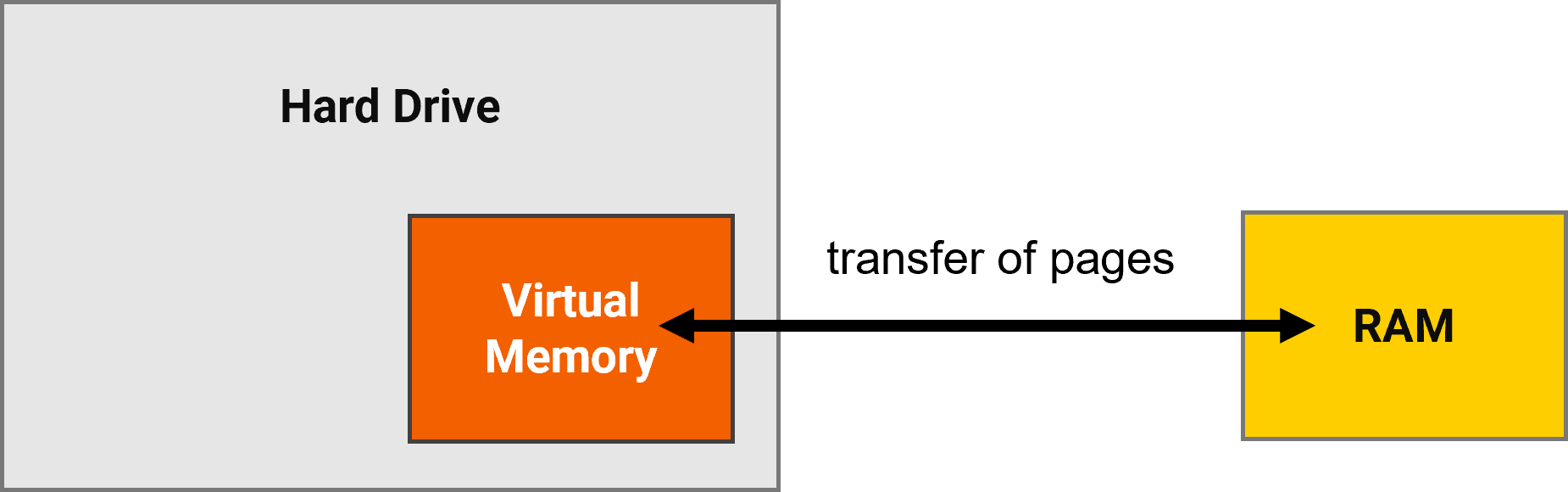
(Abraham Silberschatz and Peter Galvin, 2011) (GeeksforGeeks, 2024) (Csc-Knu, 2017) (Gillis et al., 2021)

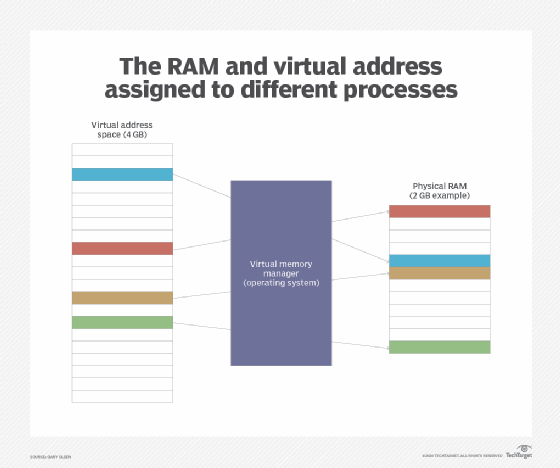
Paging





Virtual Memory





# **Task 2:**

**Given:**

Address Space = 44-bit

Page Size = 16 KB = 214 Bytes

Page Table Entry Size = 16 Bytes=24

**Step-by-Step Analysis:**

**1-** **Size of the Process:**

The process has a 44-bit address space, meaning it can address up to 244 bytes of memory.

Size of the Process = 244

**2-** **Size of a Page:**

Each page in this paging system is 16 KB, translating to 214 bytes per page.

**3- Total number of pages:**

The total memory addressed is divided by the page size to determine the number of pages required.

244 / 214 = 230 Pages.

**4-** **Entry Size of Each Entry Page Table:**

Each entry in the page table that points to a page is 16 bytes.

16 Bytes = 24 Bytes.

**5-** **Size of the first Page Table:**

The number of entries in the outer page table (equivalent to the number of pages in the inner table) multiplied by the size of each entry gives the total size of the outer page table.

230 \* 24 = 234 Bytes.

234 does not equal 214

So, we need a second-page table:

**Pages for second-page table**

234/214=220

The size of it is.

220 \* 24=224

224 does not equal to 214

So we need a third-page table:

**pages for the third page table**

224/214=210

The size of it is

210 \* 24=214 Pages for the third-page table

214=214

In summary, we will need three-page tables for this process, one outer page table and 2 inner page tables.

In which each table contains 210 pages since each level is decreasing by 210.

**Part 2:**

Coming to the distribution of 44 bit among the page table indices and offset

Log base 2(214)= 14 bit for the offset which is log base 2 for the page size

Coming to the page table indices

We must know the number of pages in each page table to get this we need to divide the page size/page entry size which is:

214/24=210 pages per table

So taking the log base 2(number of pages in the required page table)

Log base 2(210)=10 bit

Since all the page tables have the same number of pages 10 bit for all page table indices.

|  |  |  |  |
| --- | --- | --- | --- |
| Outer page table (10 bit) | Second Inner Page table (10 bit) | First Inner page table (10 bit) | Offset (14-bit) |

In this calculation we would obtain how the 44-bit address space is distributed among the offset and page table indices

**Part 3:**

Since we are reflecting on the logical address in a multi-level paging system, the logical address is typically divided into several parts, a page number that is divided into parts in which each part corresponds to an index within a level of the page table hierarchy, also we do have the offset.

Page table indices: Page table indices are parts of the logical address that are used to index page tables at a multi-level paging system.

When a CPU generates a logical address, the bits allocated for the outermost page table are used to index into this table, the entries found at this index, point to the location of the second inner-level page table in memory. However, the next set of bits in the logical address are used to index are used to index the first inner page table which contains the physical frame number of the memory page that contains the data.

Offset: It is used to pinpoint the exact byte within the physical page frame so the data would reside in this exact byte in the physical memory so it would operate.

Since the MMU which is memory management unit used the base address to locate the physical page frame in memory then it will use the base address once again, adding it to the offset so it would give us the complete physical address in which where the data will reside.

**Deferent between Linux and Mac iOS:  
both use page tables:**

**Linux:**

* **Paging Implementation:**

Linux employs a virtual memory concept in conjunction with a demand paging system. This implies that memory pages are not loaded into physical memory at the beginning of a program's execution, instead they are loaded into memory only when needed.

* **Page Replacement Algorithm:**

The "page replacement algorithm," a modified form of the Least Recently Used (LRU) algorithm, is the main method used by Linux. Its purpose is to optimize the way the system uses swap space.

* **Page Size:**

Linux typically uses a page size of 4 KB on x86 architectures, but it can support larger page sizes as well (like 2 MB and 1 GB) using HugePages.

* **Kernel Space and User Space:**

the kernel area has its own set of page tables, stability and security are ensured.

**MAC IOS:**

* **Paging Implementation:**

Demand paging is also used, especially since the Mach kernel was added to its design. With cutting-edge features like compressed memory to increase efficiency, this system is highly complex.

* **Page Replacement Algorithm:**

For page replacement, macOS employs an instance of the LRU algorithm; however, extra heuristics are frequently added to improve efficiency, particularly in multimedia apps.

* **Page Size:**

The normal page size in macOS is 4 KB, although like Linux, it may support greater page sizes (up to 2 MB) for certain needs.

* **Kernel Space and User Space:**

In order to make effective use of the RAM that is available, macOS employs a unified buffer cache system that combines the buffer cache (used for file I/O) with the page cache (used for executable code).

**Deferent between Windows and Mac Linux:  
both use page tables:**

**Windows**

* **Multi-Level Page Table Structure:**

Windows uses a multi-level page table structure that includes the Page Map Level 4 (PML4), Page Directory Pointer Table (PDPT), Page Directory (PD), and Page Table (PT) in 64-bit architectures.

* **Page Sizes:**

Standard page sizes in Windows are 4 KB, and it also supports large pages of 2 MB or larger. This is particularly useful for applications requiring extensive memory access.

* **Swapping:**

Windows uses a page file (pagefile.sys) for swapping. This file is used to store pages that have been moved from RAM to disk.

**Linux**

* **Adaptable Multi-Level Page Table Structure**:

Linux also uses a multi-level page table, but its structure is highly adaptable depending on the processor architecture. For x86\_64, it similarly uses a four-level structure.

* **Page Sizes:**

Linux supports standard 4 KB pages and huge pages (2 MB or 1 GB on x86\_64). It offers greater flexibility with huge pages, which can be beneficial for performance in certain scenarios.

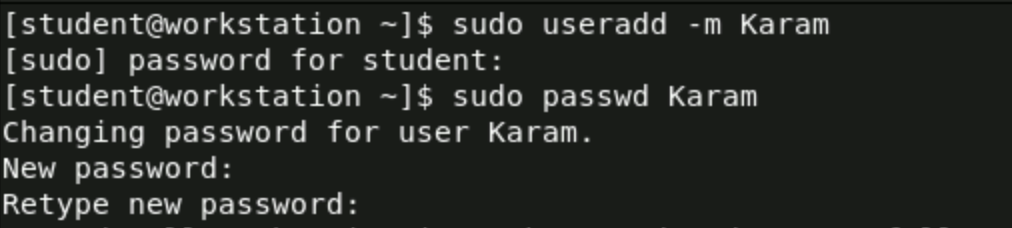
* **Swapping:**

Linux uses a swap partition and supports multiple swap files. This approach allows for more flexibility in managing swap space compared to a single swap file.

# **Task 3:**

Part 1:

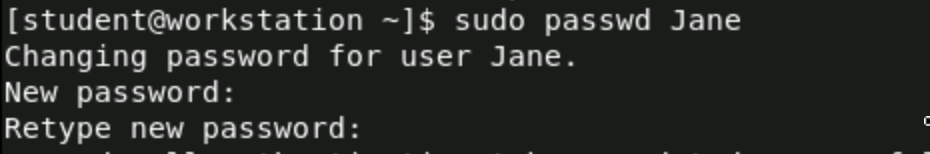
Create a user for Karam and set a password for him:



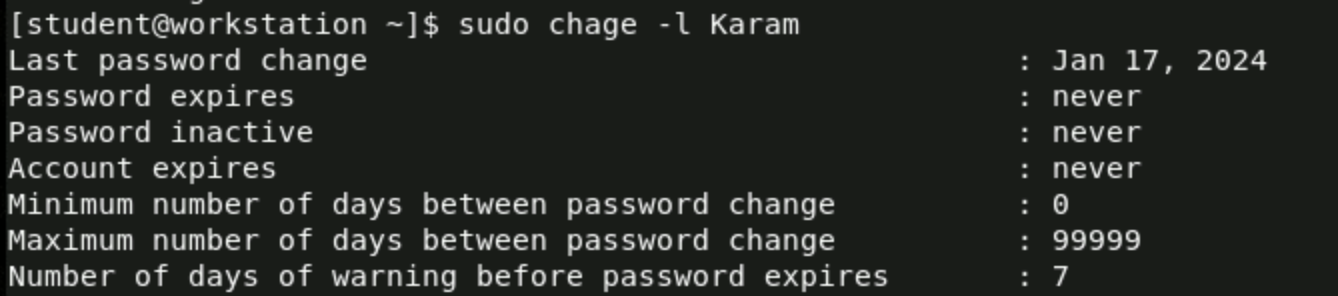
-m so to make a home directory for the user personal files and settings

Create a user for Jane and set a password for her:

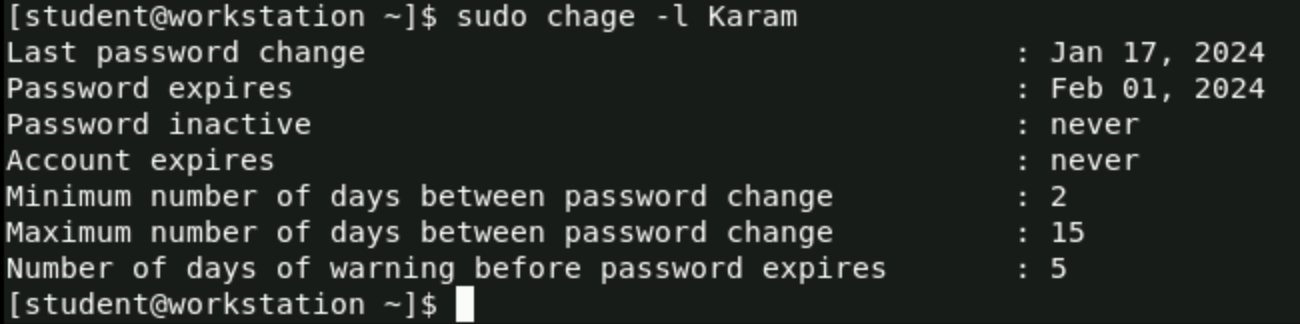




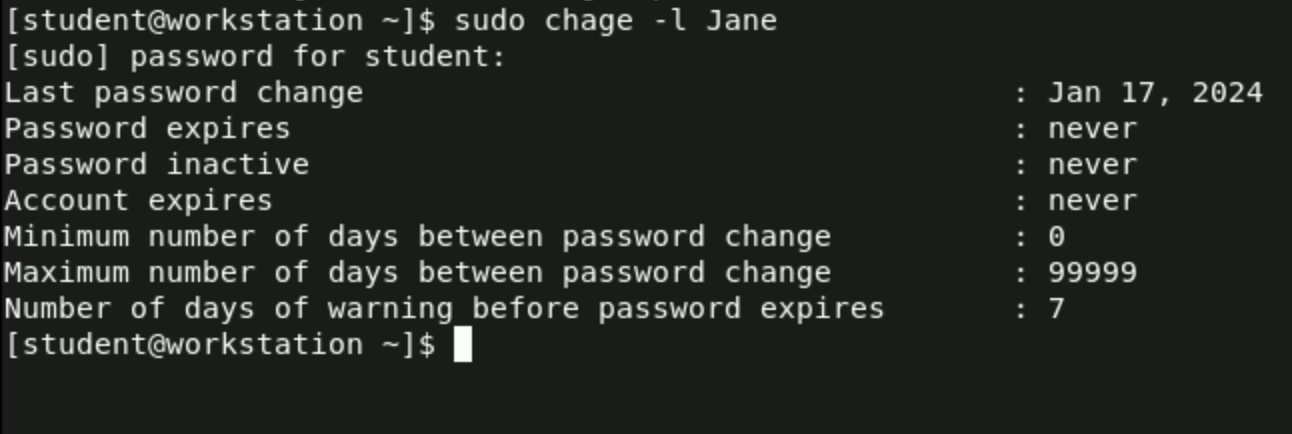
Editing the date when the password will expire and edit the minimum number of days before which the password cannot be changed, edit the maximum number of days the password is valid, and edit the number of days before password expiration that the user is warned for Karam.



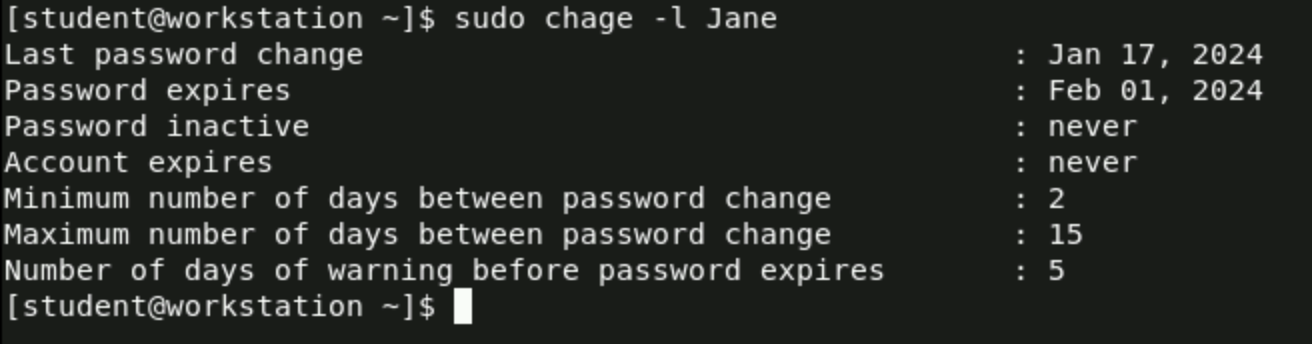




Editing the date when the password will expire and editing the minimum number of days before which the password cannot be changed, editing the maximum number of days the password is valid, and editing the number of days before password expiration that the user is warned for Jane.







**Part 2:**

Creating a group called "TechTeam":



-g 60 cat /etc/group

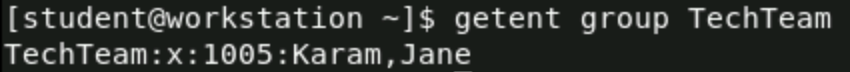
Add Users to the "TechTeam" Group:





1. G- these so add the user to a group without removing him from another group that he is already in

Checking if Karam and Jane are in the group:



**Part 3:**

Create the Project Directory:



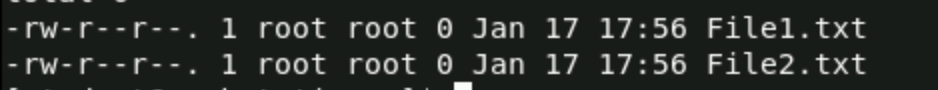
Create Two Files in the Directory:





Check if the files are in the directory:

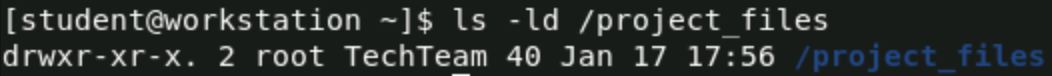
Ls -l /project\_files

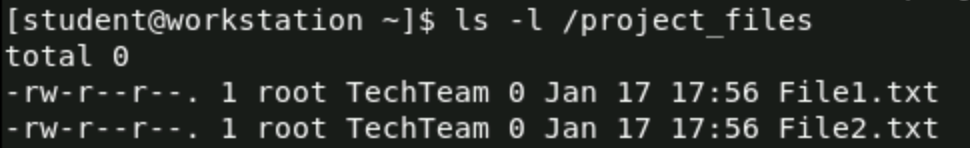


**Part 4:**

Change the ownership of the directory to the "TechTeam" group:







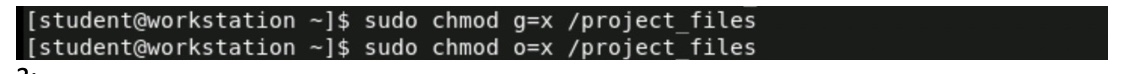
-R is recursive which means it will apply to the directory and everything in the directory.

**Part 5:**

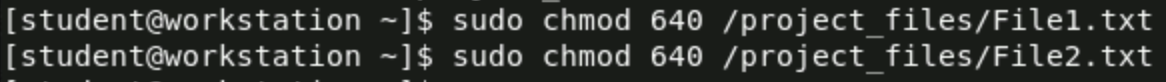
Set the directory and file permissions so that the user can read, write, and execute:



Give permissions to group and others just to execute the directory only:

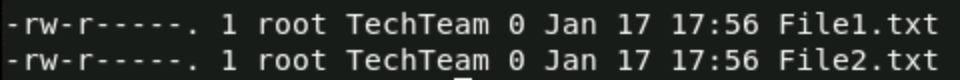


Give permission to the user to read and write files only and give permission for the group to read-only files:



Check the privileges for the users and groups and others for the files:





Check the privileges for the users and groups and others for the directory:

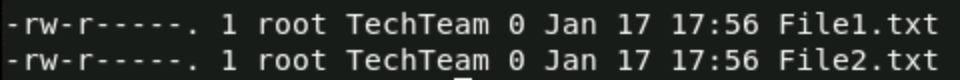




**Part 6:**

Check the privileges for the users and groups and others for the files:



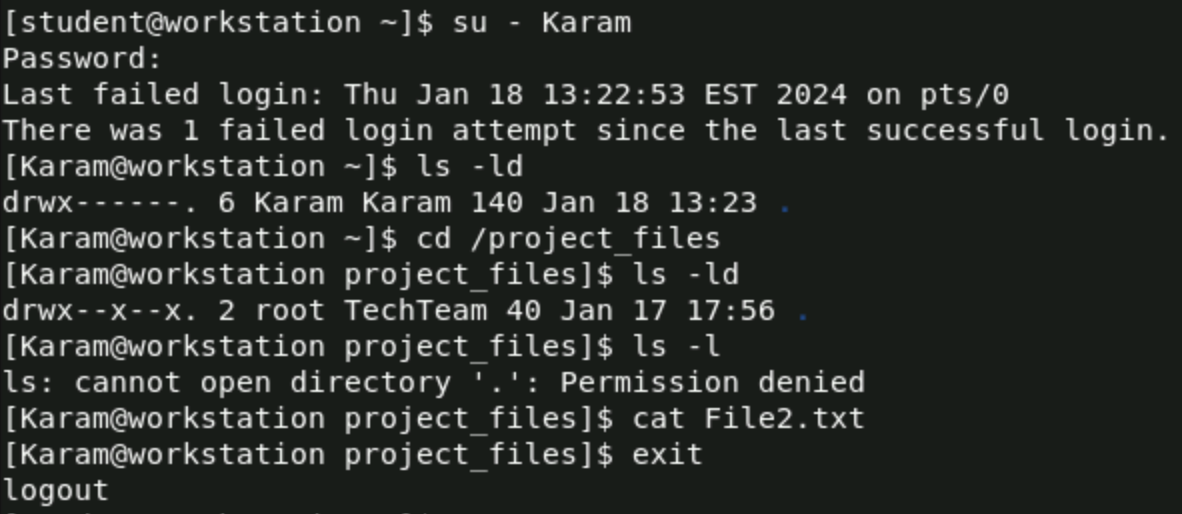


Check the privileges for the users and groups and others for the directory:

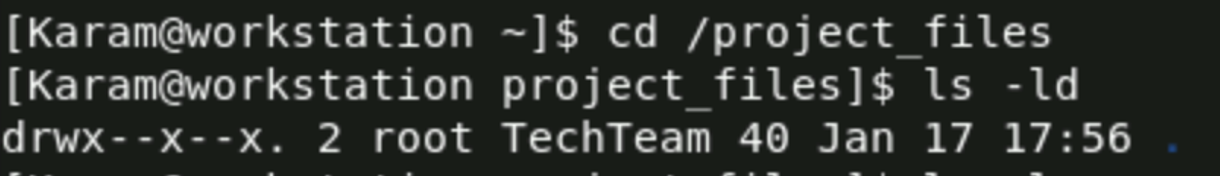




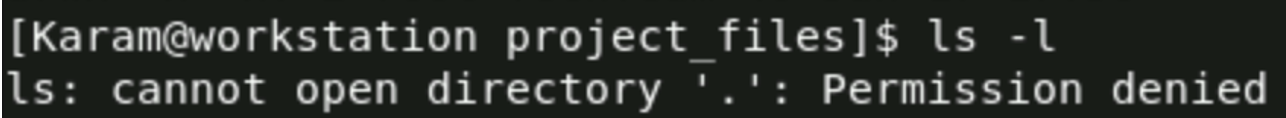
**Part 7:**



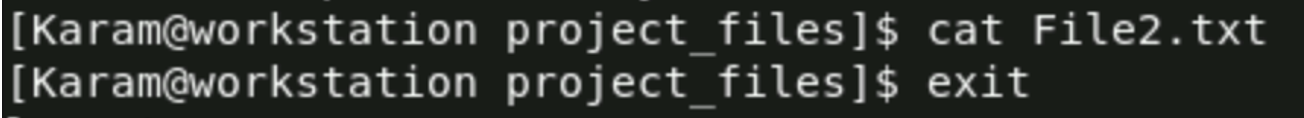
This mean that Karam can execute the directory:



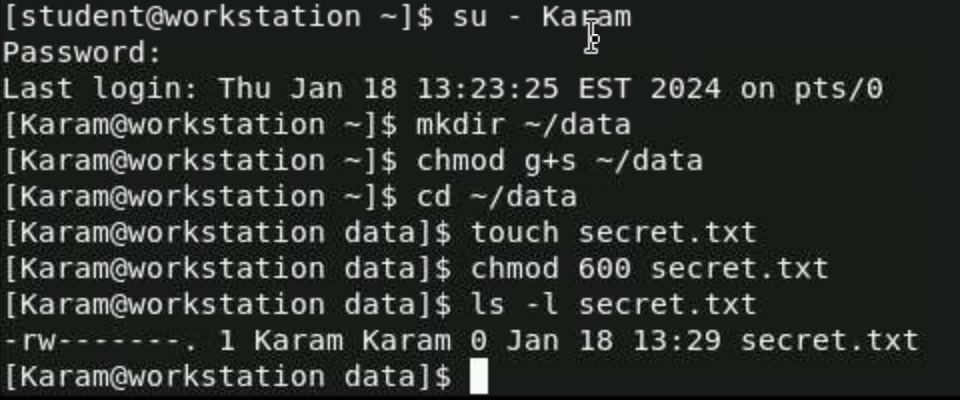
This means that Karam cant execute the files:



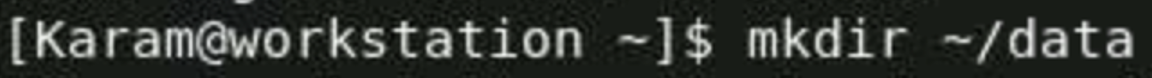
This means that Karam can read the files:



**Part 8:**



Create a directory called data:

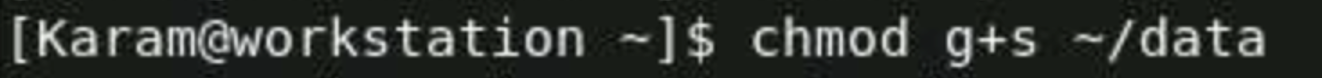


modify the permissions of the data directory:

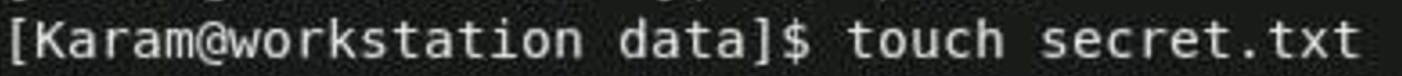
g: Refers to the group ownership of the file or directory.

+s it specifically refers to setgid (set group ID).

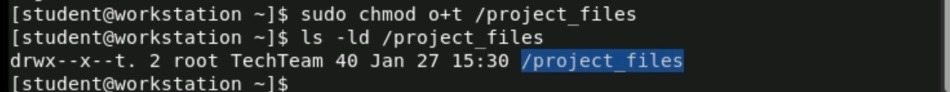
~ represents the home directory of the current user.



Create a file called secret.txt in the data directory:

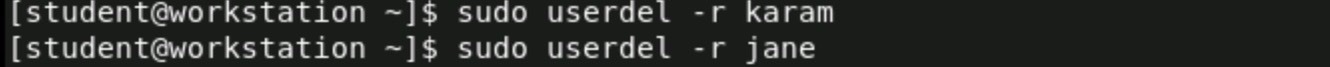


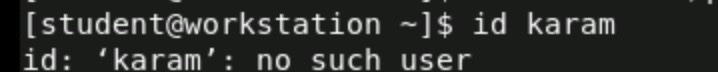
**Part 9:**



**Part 10:**

deleting users and checking:



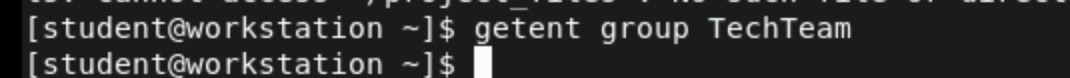


A close up of a black background

Description automatically generated

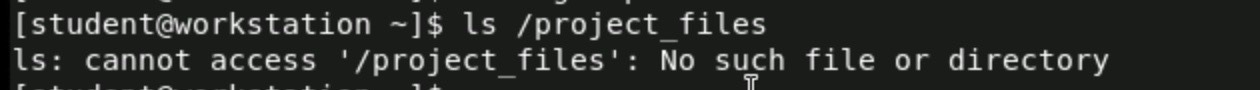
deleting techteam group and checking





deleting project file directory and checking



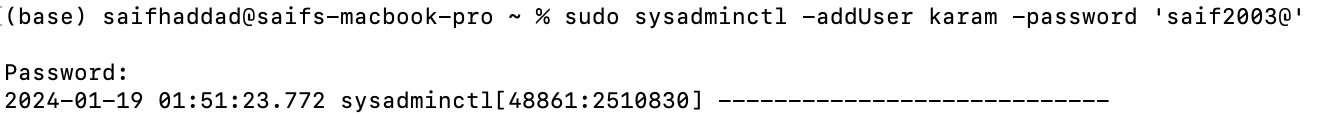


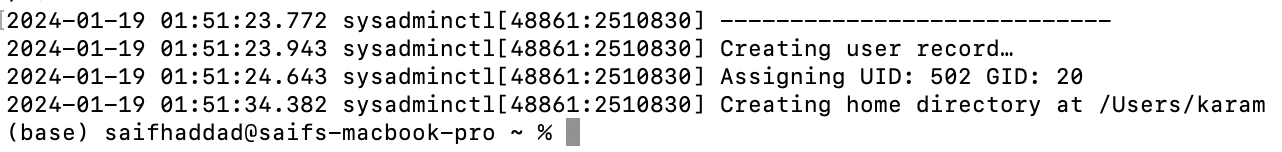
# **Task 4:**

**Part 1:**

**Creating Karam user and setting a password for Karam:**

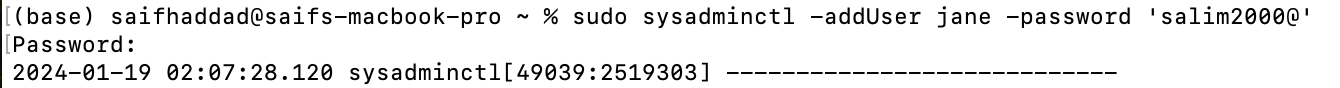
**creates a new "Karam" user account and gives it a password. To run programs with administrative access, I used sudo. Sysadminctl is a macOS tool for user management.**

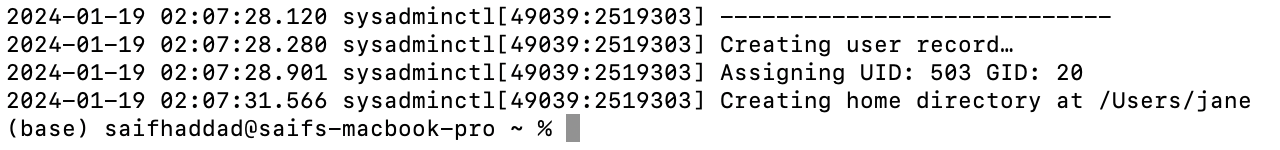




**Creating Karam user and setting a password for Jane:**

**creates a new "Jane" user account and gives it a password. To run programs with administrative access, I used sudo. Sysadminctl is a macOS tool for user management.**





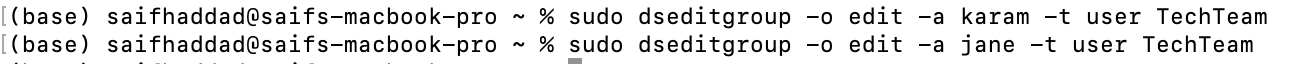
**Note: Compared to Linux, macOS does not offer as much command-line control over password policies. Usually, advanced setups include dealing with system settings or utilizing other tools.**

**Part 2:**

**Creates a new group named "TechTeam". dscl is the Directory Service command line utility for managing users and groups.**

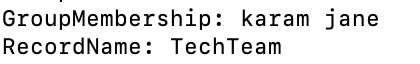


**Adds the user "karam and Jane" to the "TechTeam" group. dseditgroup is used for editing group membership.**



**Checking the group and the user exists:**





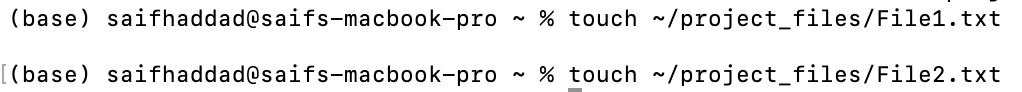


**Part 3:**

**Creating project\_files as a directory:**



**Create two files in the directory:**



**Checking if the files are in the directory:**



**Part 4:**

**Setting TechTeam as a ownership for the project\_failes directory:**



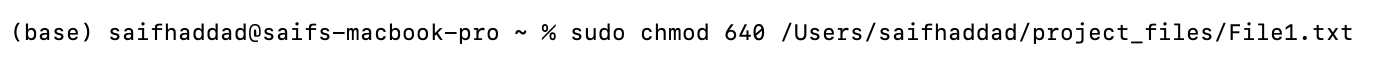
**Part 5:**

**Set the directory and file permissions so that the user can read, write, and execute:**

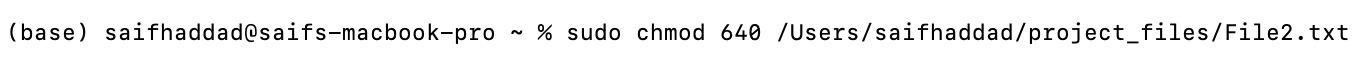


**Give permissions to group and others just to execute the directory only:**

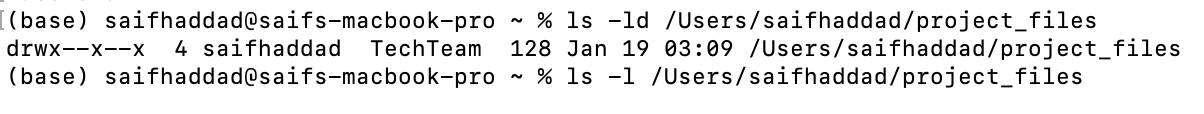




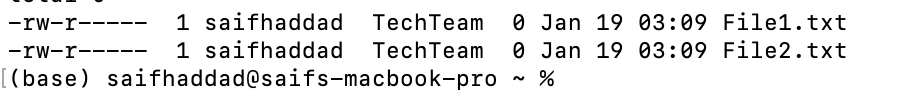
**Give permission to the user to read and write files only and give permission for the group to read-only files:**



**Check the privileges for the users and groups and others for the directory:**



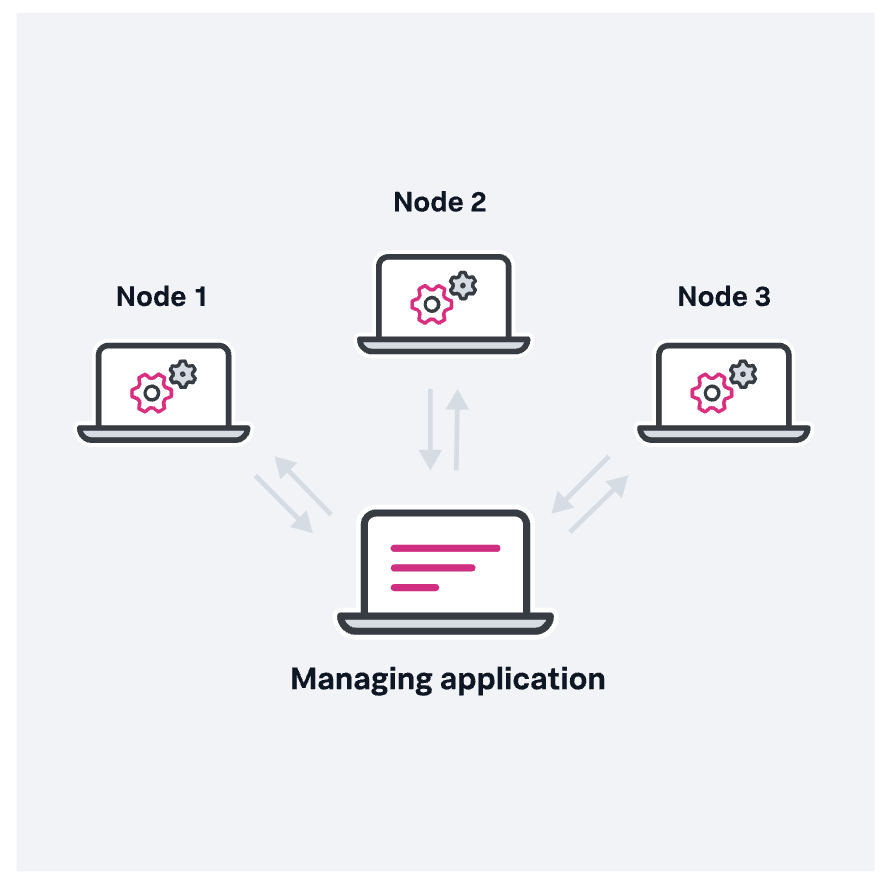
**Check the privileges for the users and groups and others for the files:**



# **Task 5:**

**What is a distributed system in general:** (Splunk, 2023)

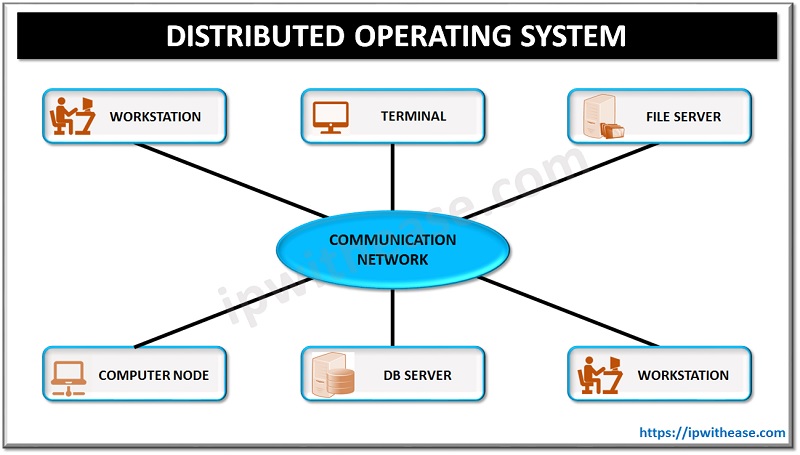
This system allows a network of independent computers to deal with each other and work with each other to do a task, which appears to the user as a single user.



**What is Distributed Operating Systems and Undistributed (Centralized) Operating Systems:** (Splunk, 2023) (javatpoint, 2022) (GeeksforGeeks, 2023)

**Distributed Operating Systems:**

These are systems that allow several computers, connected by a network to be managed and coordinated such that they function as a single, one unit. They allow the sharing of resources between several machines, such as memory and computing power (independent computers).



**Undistributed (Centralized) Operating Systems:**

One computer's resources are managed by these systems. Without the need for networked cooperation with other computers, all processing and resource management operations are performed on a single system.

**Advantages of Distributed Operating Systems:** (Splunk, 2023) (javatpoint, 2022) (GeeksforGeeks, 2023)

**1-** **Concurrency of Components**:

Different components of your system can operate together at the same time in DOS, but they are located on separate computers. For instance, one computer may be managing network requests, another may be storing files, and yet another may be analyzing data. While they each work alone, they all collaborate and communicate with each other.

**2-** **Independent Failure of Components:**

Every computer in a distributed system has the potential for failure without impacting the whole system. For example, the other computers can continue to function even if one crashes unlike a single computer system, which may stop all activities in the event of an issue. DOS is built to handle these errors.

**3-** **Distributed Resource Sharing:**

Among several computers, DOS allows for the sharing of resources including files, printers, and internet connections. This enables many computers to share a printer or, for instance, view a file saved on one computer that is shared with another computer on the network. In order to improve accessibility and efficiency, because resources can be distributed depending on demand and unused resources may be used somewhere else, it allows for more effective task handling and may result in cost savings.

**Disadvantages of Distributed Operating Systems:** (Splunk, 2023) (javatpoint, 2022) (GeeksforGeeks, 2023)

**1-Complexity:**

In comparison to managing a centralized system, managing a distributed system is much more complicated.

**2-** **Security:**

In distributed systems, security management may provide additional difficulties. Because the system operates over several computers, often in different places

**3-Network dependency:**

DOS is mostly dependent on network connectivity. The system's performance may be significantly impacted if the network is weak or slowing down.

**4-Difficulty in maintenance:**

Finding and solving problems is so hard in distributed operating systems because it could be in the network or one of the hardware computers has an issue.

**The difference between Distributed Operating Systems and Centralized Operating Systems:** (Splunk, 2023) (javatpoint, 2022) (GeeksforGeeks, 2023)

**1-Security:**

**Centralized:** Weak as a single server contains all information. Illegal access may result in lost or altered data.

**Distributed:** Provides strong security since data is spread among network nodes. All nodes in the system reflect changes, and because it is self-monitoring it is resistant to attacks and unexpected changes in data.

**2-** **Availability:**

**Centralized:** Commonly failures when there are a lot of requests, which makes the system unavailable.

**Distributed:** Able to handle heavy network traffic because requests and data are spread among several nodes, increasing the overall availability of the network.

**3-** **Accessibility:**

**Centralized:** If there are problems with the central server, accessibility challenges happen because standardized procedures may not meet different user requirements.

**Distributed:** Distributed networks are more accessible under pressure because they provide faster reaction times and more safety.

**4-** **Data Transfer Rates:**

**Centralized:** Connection and transmission speeds can be affected by distance to the server

**Distributed:** Gives customers the option to select the closest or best node, improving the effectiveness of data transport.

**5-** **Scalability:**

**Centralized:** The limitations of server capacity make scaling difficult. The server may get stressed as the number of customers rises, particularly during peak hours.

**Distributed:** Distributed models don't have this issue since several machines share the load.

|  |  |  |
| --- | --- | --- |
|  | **Centralized** | **Distributed** |
| Data Storage and Management | One server for the management and storage of all data | Data is distributed over several servers or nodes, usually in different places. |
| Resource Allocation and Task Execution | Problems may result from all tasks being handled by the central server. | Resources and tasks are spread among nodes, allowing for parallel processing. |
| Scalability and Flexibility | Often, scaling up requires updating the central server, which is more expensive and difficult. | Easier to grow by adding additional nodes; which is more adaptive to increasing needs. |

(Splunk, 2023) (javatpoint, 2022) (GeeksforGeeks, 2023)

**A real-life example of the benefits of Distributed Operating System:** (Splunk, 2023) (javatpoint, 2022) (GeeksforGeeks, 2023)

**Google search system:** Hundreds of computers process each request, searching the web and returning relevant data. Although it seems to the user that Google is a single system, in reality, it is a collection of computers collaborating to do a single task: returning the search query's results.

**How middleware is used in Google search system:**

Middleware enables effective communication between servers in Google's distributed system, ensuring that a user's query is handled quickly and accurately across several computers."

**The benefits:** (javatpoint, 2022)

**1-** **Speed and Efficiency in Processing Queries:**

Due to its distributed architecture, Google is able to divide up search requests into smaller jobs that are handled concurrently by several computers. Google can deliver search results in just a few seconds because to its lightning-fast information retrieval and analysis powered by parallel computing.

**2-** **Handling Massive Volumes of Data:**

Google's distributed system is capable of dynamic scaling to handle this massive, continuously expanding data volume. It can index and process billions of web pages, something that a centralized system would not be able to do.

**3-** **High Availability and Reliability:**

Google makes sure its search engine is always online by spreading the workload over a network of machines. Service interruptions may be avoided if one computer, or even an entire data center, has problems since other machines can take over.

**4-** **Load Balancing:**

By effectively distributing the load across its servers, Google's distributed architecture makes sure that no server is overloaded with requests. The system's ideal performance levels are maintained throughout because to this balanced load distribution.

**5-** **Cost-Effectiveness and Energy Efficiency:**

Google can make better use of its processing resources because to a distributed architecture. It may make use of a greater number of more affordable computers as instead of depending just on a small number of powerful (and expensive) servers. The distribution of the workload can lower the total power usage, which improves energy efficiency.

**A real-life example of distributed systems in general:**

An example is the video editor on the client’s computer splits the tasks into small pieces, these tasks will be given to a lot of nodes or computers to complete the needed task. Once the task is done the managing application gives a new task to the node that completes the task and will continue in this process until all the tasks are done, then these pieces of tasks will put to gather again, this process might be distributed over hundreds of nodes. This helps to execute a task in one minute rather than it would take days to complete on a single computer

**Some concepts that are related to Distributed Operating Systems:** (Splunk, 2023) (javatpoint, 2022) (GeeksforGeeks, 2023)

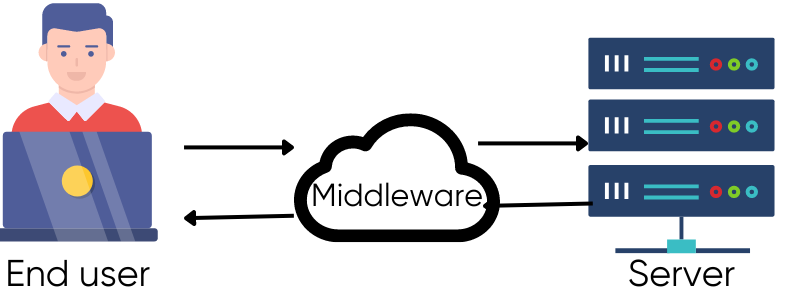
**1-** **Middleware:**

**Definition:**

a kind of software that serves as a bridge to allow collaboration and communication between several machines connected to a network. Consider it as a translator or mediator that enables successful communication between several computers—which may have different operating systems and apps.

**Functionality:**

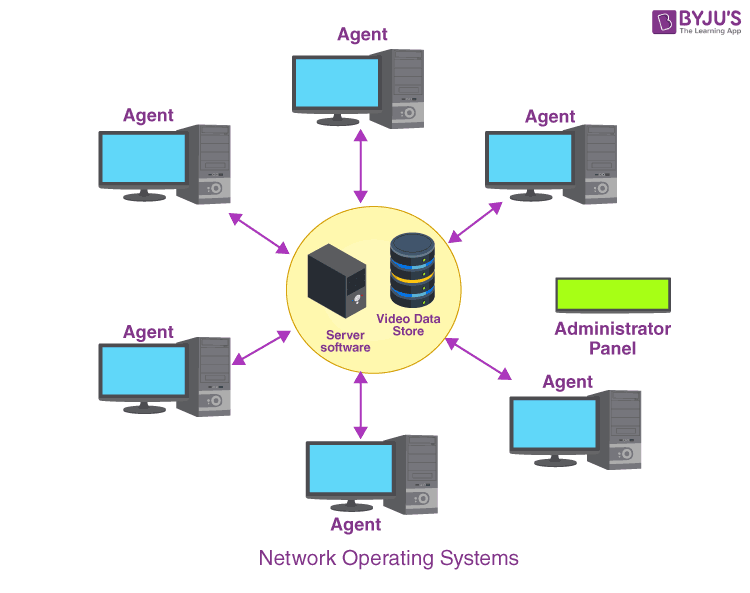
Middleware connects disparate systems by standardizing communication and data exchange, thereby gluing them together so they can work together seamlessly." This is essential in a DOS as many machines may be running different applications or operating systems.



**2-** **Network Operating Systems:**

**Definition:**

Network Operating Systems are a type of distributed system designed for network data sharing, it is a software program that enables resource sharing and communication between several computers linked to a network. network operating systems (NOS) are made especially for running networked settings. They offer the tools and protocols required for efficient resource management and communication across multiple devices connected to the network.



**Note: The difference between NOS and DOS:**

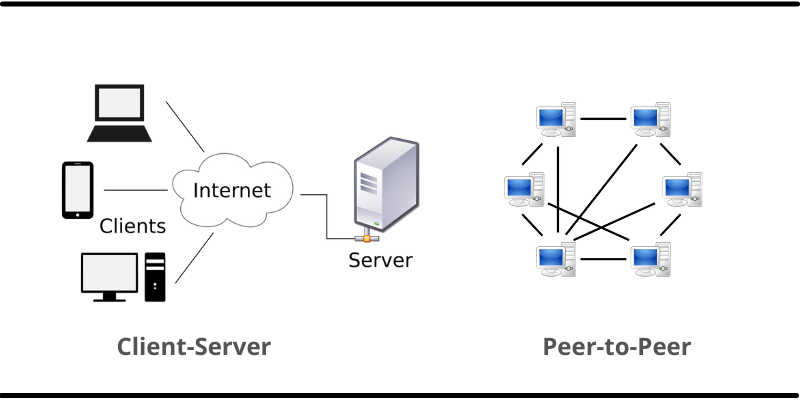
Network Operating Systems (NOS) link many computers together so they may communicate and share resources like printers and data. Every computer still functions independently, which means that NOS treats each computer as an individual node of the network. Distributed operating systems, allow several computers to share resources and operate together as if they were a single, large computer, which means that a task can be distributed on a lot of nodes that are connected to the network.

**3-** **Peer-to-Peer Networks:**

Peer-to-peer networks are a type of distributed system, in which every peer of computers shares resources and tasks and communicates with each other’s by acting as both a client and a server.

**Example of peer-to-peer networks:**

**BitTorrent:** this application uses peer-to-peer communication protocol. It divides files into smaller parts and spreads them all around a user network. Since users can download and upload parts of files to and from several peers at once, this approach enables efficient and scalable downloading. This reduces server load and enhances download speeds. This application is usually used for legitimate purposes.



# **Task 6:**

**Definition of Concurrency in Operating Systems?** (javatpoin, 2022) (Libretexts, 2021) (Tutorialspoint, 2022)

In operating systems, concurrency refers to the idea of running several processes or threads together on a single CPU system. An operating system may manage more than one job in a given amount of time by managing and carrying out many activities or processes at the same time. By making better use of resources like CPU and memory, concurrency aims to improve system speed and efficiency. On single-processor systems.

Concurrency is the ability to manage several processes at once in a way that enhances the system's responsiveness and efficiency, whether through time-sharing or real parallelism. It relates to the system's capacity to handle several jobs, even if they may be fast switching on a single processor and making them appear to be happening at the same time.

**Concurrency can be achieved in:** (javatpoin, 2022) (Libretexts, 2021) (Tutorialspoint, 2022)

**1-** **Real parallelism:**

Multiple processes or threads execute at the same time on separate processors or cores in multi-core or multi-processor systems.

**2-Time sharing (Time Slicing):**

Time-slicing is used in single-core CPUs to achieve concurrency. In order to give the impression of simultaneous execution, the operating system provides just a single slice to each process in turn. The CPU switches between processes fast enough to give the impression that they are operating concurrently, but in actuality, it is switching between them quickly.

**Principles of Concurrency:** (Libretexts, 2021) (Tutorialspoint, 2022)

**1-** **Executing Multiple Tasks at Once:**An operating system can handle several tasks at once because of concurrency. While this is accomplished by fast switching, in single-core CPUs, different cores in multi-core CPUs may actually handle multiple tasks at the same time.

**2-** **Effective Use of Resources:**

Concurrency improves system use of resources by enabling multiple tasks to run concurrently in time, including CPU, memory, and input/output. For example, one job might use the CPU while another waits for an I/O operation.

**3- Increasing the throughput of the system:**

Higher system throughput is typically the result of concurrent process execution. It is particularly essential in multi-tasking contexts to be able to finish more activities in less time.

**Weakness in Concurrency:** (javatpoin, 2022) (Tutorialspoint, 2022)

**1-** **Locating the programming errors:**

Programming errors are hard to identify since common components have different states every time the code runs.

**2-** **Sharing Global Resources**

Conflicts in outcomes are possible when several operating system processes attempt to utilize and modify the same data (such as a variable) at the same time when multitasking.

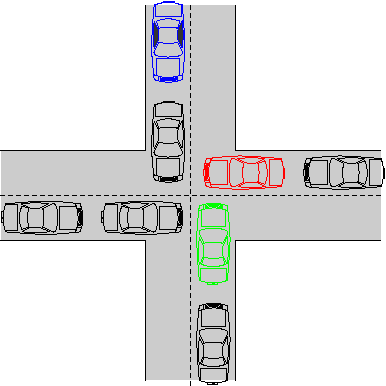
**3-** **Locking the channel:**

When a resource in an operating system is locked, only one process is able to access it at once, which could cause other programs to wait their turn and slow down system performance.

**4-** **Synchronization:**

A need for synchronization is to avoid conflicts when several jobs are using common resources. Ensuring the integrity and consistency of data is crucial.

**5-** **Deadlocks:**

Multiple processes becoming stopped while waiting for resources that other processes are holding might result from concurrency.

**Where concurrency is used in real-life examples:** (javatpoin, 2022)

**1-** **Web Servers:**

They manage several client requests at once by using concurrency. The server is able to serve numerous clients independently because each request is processed by a different thread or process.

**2-** **Operating Systems:**

Concurrency is a feature of modern operating systems like Windows, macOS, and Linux that allows many system processes and user programs to operate concurrently, improving system speed and user experience.

**3-** **Database Systems:**

Database concurrency is essential because it enables several transactions to take place concurrently, improving the system's throughput and performance.

**How does concurrency Maximize or Minimize Computer Performance?** (javatpoin, 2022) (Libretexts, 2021) (Tutorialspoint, 2022)

**Benefits:**

**1-** **Efficient Resource Utilization:**

An operating system that supports concurrency can utilize system resources more effectively. For example, the CPU can handle other tasks while one process waits for input/output operations to finish.

**2-** **Flexibility:**

In the background, some applications can be used while the user interacts with others.

**Drawback:**

**1-** **Complexity in Management:**

Needing complex algorithms for efficient task scheduling.

**2-Deadlocks:**

When many processes are waiting for one another to release resources, it might result in deadlocks caused by concurrency.

**Example of techniques that is used to achieve concurrency.** (javatpoin, 2022) (Libretexts, 2021) (Tutorialspoint, 2022)

**1-** **Multi-Threading (Just with Multi-Core):**

A multi-threaded process is divided into smaller components known as threads. Although they share process resources like memory and file handles, each thread represents an individual route of execution. This enables the concurrent execution of many threads inside a single process.

**Advantage:**

Effective use of CPU time, particularly in situations when a thread is awaiting an I/O operation. It enables the CPU to be used by additional threads. This results in increased throughput and improved system responsiveness.

**Example:**

Web browsers have the ability of multi-threading, which allows tabs to run as independent threads within a single process.

**2-** **Process Scheduling:**

The process by which an operating system selects which of the available processes the processor will run is called process scheduling. Processes are given CPU time using techniques including round-robin, priority scheduling, and first-come, first-served.

**Advantage:**

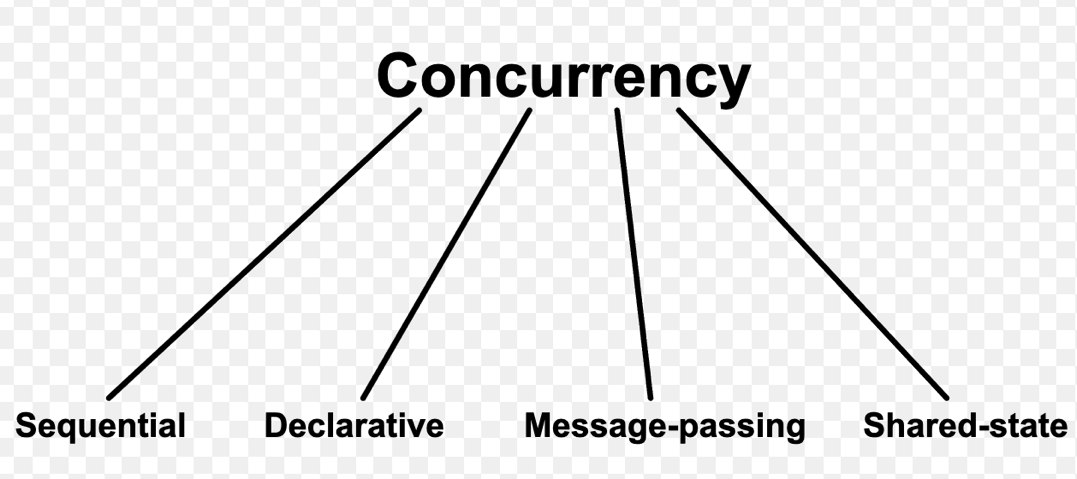
Increases performance and ensures equal CPU time for all processes.

**Example:**

The scheduler of a multitasking operating system, such as Linux, distributes CPU time slices to different programs so they can operate concurrently.

**3-Fork:** when any process uses a Fork to create a child process. This means it allows for parallel execution, parallelism is concurrency which means executing multitasks. So because the processes are working independently we can say that Fork is practical application of concurrency.

**4-** **Multitasking:** a particular kind of concurrency in operating systems, multitasking allows several processes to be controlled and executed quickly to make them appear to be done at the same time by switching between them.



# **Task 7:**

**What are Remote Procedure Calls (RPC) and Remote Method Invocation in distributed operating systems:**

**Remote Procedure Calls (RPC):** (GeeksforGeeks, 2022) (: Lawrence Williams & Williams, 2023) (Neha, 2018)

A protocol that enables a computer program to call a function, or procedure, in a different address space (usually on another computer connected to a shared network), seeming like a call to a local function without the programmer having to manually code the information for the remote interaction.

The RPC is usually called by value.

**How Remote Procedure Calls (RPC) works:** (GeeksforGeeks, 2022) (: Lawrence Williams & Williams, 2023)

Any required arguments are passed together with a request to run a specified operation from the calling software (client) to a distant system (server).

The client receives the results from the distant system when the server completes the operation.

**Step by Step:** (GeeksforGeeks, 2022) (: Lawrence Williams & Williams, 2023) (Neha, 2018)

**1-** **Client Procedure Call:**

When a client application calls a procedure (function), the process begins. However, the client calls a unique function called a stub rather than a local process. The purpose of this stub function is to give the client the impression that it is the original remote operation.

**2-** **Parameter Marshalling:**

The procedure call is ready for transmission when the client-side stub is created. This calls for marshaling, which is the act of transforming the procedure parameters into a format that is suitable for sending over a network.

**3- Request Sending:**

The procedure name and parameters (the marshaled data) are then transmitted to the server via the network.

**4-** **Server Receives Request:**

Incoming requests are heard by a server-side procedure. The data is transferred to the server stub after receiving the call from the client.

**5- Server-Side Unmarshalling:**

The received data is unmarshalled by the server stub. In other words, it converts the byte stream back into its original parameters or data structure.

**6-** **Procedure Execution:**

Using these unmarshalled arguments, the server's stub then calls the server's actual process.

**7-** **Result Marshalling:**

The server stub marshals the outcome (results) once the server has completed the process.

**8- Result Sent Back to Client:**

The marshaled result is returned to the client across the network by the server stub.

**9-** **Client-Side Unmarshalling:**

When the client stub receives the answer, it unmarshalls the outcome and puts it back in a manner that the client application can utilize.

**Remote Method Invocation (RMI):** (.javatpoint, 2022) (Awati, 2023) (Neha, 2018)

RMI, which is frequently connected to Java, expands on the RPC paradigm to help with object-oriented programming. It enables a program to call methods on an object on a different networked computer.

The RMI is usually called by reference.

**How Remote Method Invocation (RMI) works:** (.javatpoint, 2022) (Awati, 2023) (Neha, 2018)

works by enabling a client application to invoke methods on a distant server object's object. After receiving a request from the client including the method name and parameters, the server uses the remote object to execute the method and returns the results to the client.

**Step by Step:** (.javatpoint, 2022) (Awati, 2023) (Neha, 2018)

**1-** **Invoking a Method on a Stub:**

A method on the stub object, which serves as a local stand-in for the distant object, is called by the client.

**2-** **Establish Connection:**

A network connection is made between the stub and the server hosting where the actual remote item (object) is.

**3-** **Marshal Parameters:**

The method arguments are marshalled by the stub and sent to the server in a manner appropriate for network transmission.

**4-** **Unmarshal on the server side:**

The marshaled arguments are received by the server, which then Umarshaled (turns them back into objects) and sends them to the real remote object.

**5-** **Method Execution:**

Using the parameters provided, the remote object runs and executes the needed method.

**6- Result Marshal:**

This result is Marshal by the remote object and sent back to the client-side stub.

**7-** **Unmarshal Return Value (client side):**

After receiving the marshalled result, the stub re-creates the original return value or exception unmarshaling it.

|  |  |  |
| --- | --- | --- |
| **The Difference between RPC and RMI** | | |
| **Feature** | **RPC** | **RMI** |
| **Paradigm** | Procedural | Object-oriented |
| **Language Specificity** | works with many languages | Java-specific |
| **Interface Definition** | Uses Interface Definition Language | Uses Java interfaces |
| **Marshalling** | Manual marshalling of arguments and results | Automatic serialization of objects |
| **Transport Protocol** | Various protocols (TCP/IP, HTTP, etc.) | Java Remote Method Protocol (JRMP) |
| **Usage** | Suitable for a wide range of applications | Designed for Java applications |

(.javatpoint, 2022) (Awati, 2023)

(GeeksforGeeks, 2022) (: Lawrence Williams & Williams, 2023)

(Neha, 2018)

(: Lawrence Williams & Williams, 2023) (Neha, 2018)

**A diagram of a server

Description automatically generatedRPC has five components:** (GeeksforGeeks, 2022)

1-Client

2-Client Stub

3-RPC Runtime

4-Server Stub

5-Server

**RPC Real-life example:** (GeeksforGeeks, 2022) (: Lawrence Williams & Williams, 2023) (Neha, 2018)

**Distributed Weather Service:** RPC can be used by a national weather service to provide weather information to different websites and apps. Applications use the service to retrieve location-specific RPC queries, and the service returns weather information, including wind speed, temperature, and forecast. By centralizing the service, redundancy is decreased and data consistency across many platforms is secured.

**1-** **Procedure Calls Across Network:**

Client apps send network calls to a distant server to get weather information.

**2-** **Centralized Service:**

The weather service receives requests from many customers (websites, apps, etc.) and handles them centrally. Certain operations that can be invoked remotely are exposed by this server.

**3-** **Request and Response:**

Using the RPC protocol, clients submit requests with parameters (such as location), which are then processed by the server, which collects the relevant weather data for the requested location and returns the result.

**4-** **Platform-Agnostic:**Since the RPC technique is generally platform- and language-independent, it enables diverse applications developed in different programming languages to communicate with the weather service.

**5-** **Consistency and Efficiency:**

The service ensures that all clients receive consistent weather data by centralizing the obtaining and distribution of weather information using RPC calls. This eliminates redundancy by removing the requirement for each client to fetch or store the data individually.

**RMI has five components:** (.javatpoint, 2022) (Awati, 2023) (Neha, 2018)

**1-** **Remote Interface:** Specifies the methods that have remote invoke abilities.

**2-** **Remote Object Implementation:** The actual class that puts the remote interface's functions and methods into practice.

**3-** **Client Program:** Searches for distant objects and calls their methods.

**4-** **RMI Registry:** A naming service that helps clients locate objects located remotely.

**5-** **Stub:** An on-the-client side proxy that stands in for the distant object.

**Real-life example of RMI:** (.javatpoint, 2022) (Awati, 2023) (Neha, 2018)

**Distributed Stock Trading Platform:** Consider a stock trading website where customers may purchase and sell stocks from multiple locations. RMI might be used to implement the essential features, such as handling orders and pricing changes, on a central server. Users may place orders, get estimates, and manage their accounts by using client programs on their devices to call methods on the server objects. This distribution transfers processing from individual devices to the server, which increases scalability and performance.

**1-Remote Interfaces:**

The server will define remote interfaces with trade placement, stock price checking, and account management capabilities. Client programs can connect to these interfaces over a network.

**2-** **Remote Object Implementation:**

Specific classes would implement these interfaces on the server. When a client calls a method, these server-side classes take care of the actual logic and data processing.

**3-** **Client Interaction with Server Objects:**

Client applications of users would use the naming or directory service provided by RMI to locate the distant objects. The client can call methods on a stub it has obtained for a distant object in the same way that it would call local methods.

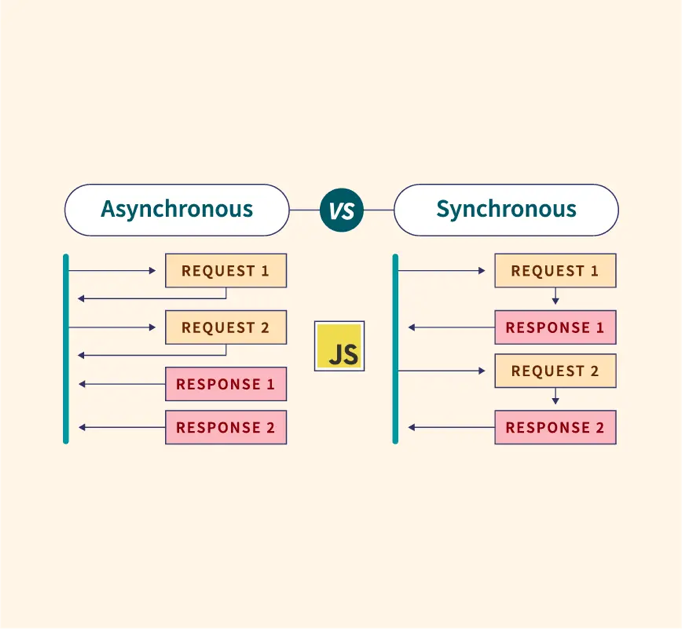
**4-** **Method Invocation Process:**

A method on the stub, like placeOrder, is called by the client when it wishes to put an order. The method call is sent over the network, the method arguments are serialized (marshalled) and then deserialized (unmarshalled) on the server side, the method is invoked on the real remote object, and any results are marshalled and sent back to the client by the RMI system.

**5-** **Scalability and Performance:**

The system can handle more users and orders than it could if every client device had to process all of the data locally. This is achieved by centralizing the basic functionality on a server and utilizing RMI to spread the processing. The client devices just send and receive data, whereas the server may be a robust device that manages intensive processing.

**What is the difference between Asynchronous and Synchronous:**



**Asynchronous:** (GeeksforGeeks, 2023) (BYJUS, 2023)

Procedures or tasks that don't happen together. It refers to the situation in computing where software sends out a request and then runs on without waiting for a response.

**Characteristics of Asynchronous:** (GeeksforGeeks, 2023) (BYJUS, 2023)

**1-Independent Operations:** Processes run separately from one another (independent operation).

**2-Non-Blocking:** Thecall does not block the program from running forward.

**3-Delayed Response:** Request responses might not come back immediately.

**Advantages of Asynchronous:** (GeeksforGeeks, 2023) (BYJUS, 2023)

**1-Can do multitasking:** Allows systems to handle numerous tasks at the same time.

**Synchronous:** (GeeksforGeeks, 2023) (BYJUS, 2023)

Procedures or tasks that happen together or one after the other. In computers, it indicates that a program pauses after submitting a request to await a response.

**Characteristics of Synchronous:** (GeeksforGeeks, 2023) (BYJUS, 2023)

**1-** **Synced Operations:** The way that processes work is coordinated and depends on time.

**2-Blocking:** The call blocks further advancement till it's finished.

**3-** **Immediate Response:** Responses happen right away, or nearly.

**Advantages of Synchronous:** (GeeksforGeeks, 2023) (BYJUS, 2023)

**1- Immediate Response:** Responses happen right away, or nearly.

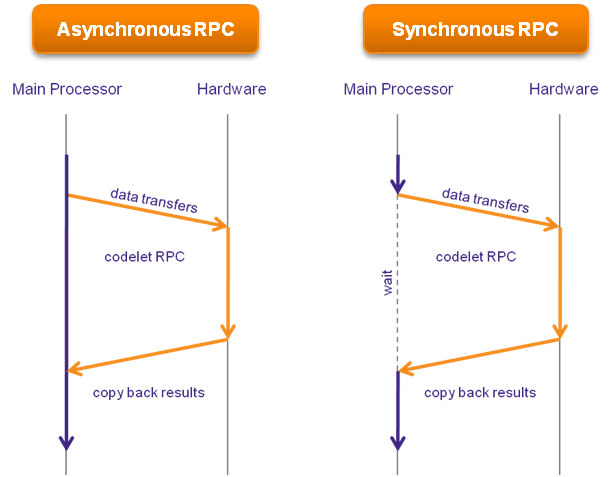
**Synchronous Communication in RPC and RMI:** (GeeksforGeeks, 2023) (BYJUS, 2023)

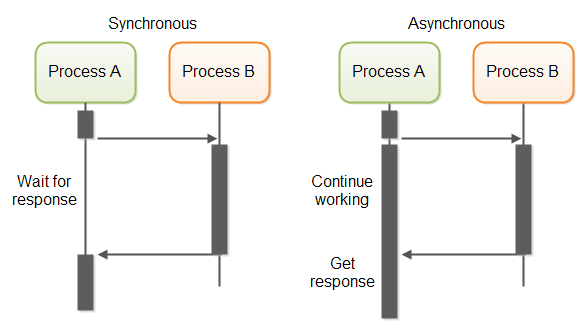
RMI and RPC are both synchronous. A client that calls a remote procedure or invokes a remote method does not move on until it receives a response. The client is blocked until the procedure or method is fully executed by the server and the result is returned.

**Asynchronous Communication in RPC and RMI:** (GeeksforGeeks, 2023) (BYJUS, 2023)

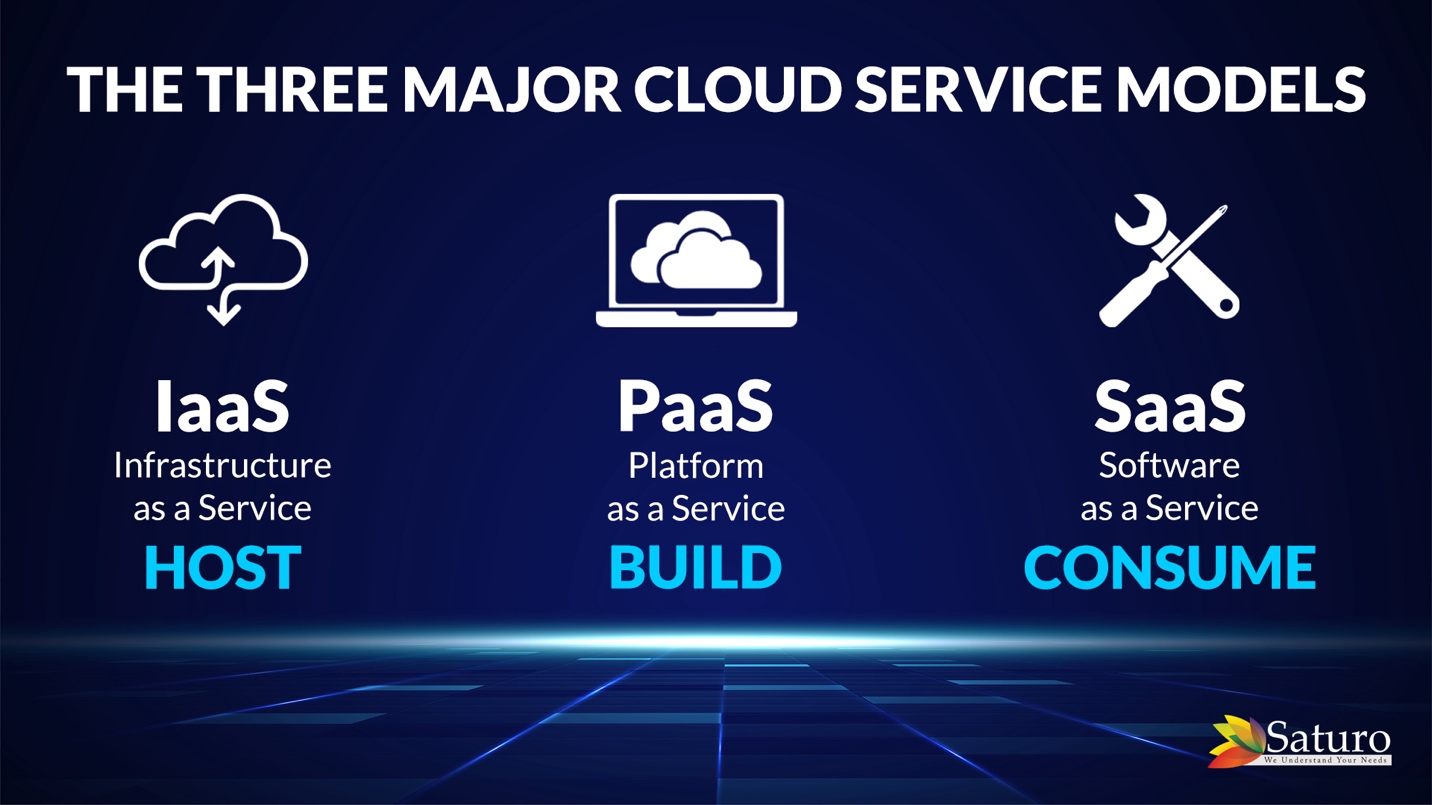
RPC and RMI provide asynchronous types where the client doesn't have to wait for the remote process or method to finish. Rather, it keeps running and deals with the outcome as soon as it's obtained.

**Note:** it depends on the situation





# **Task 8:**



**Infrastructure as a Service (IaaS):** (Red Hat, 2022) (javatpoint, 2022) (IBM, 2022) (Saratchandran, 2023) (Rosencrance, 2021)

**Definition:**

IaaS provides virtualized computing resources over the internet. It offers the foundational infrastructure components such as virtual machines, storage, and networks. Because of its scalability and flexibility, Infrastructure as a Service (IaaS) is an attractive choice for organizations with changing demands since it lets them quickly modify their resources without having to invest in actual hardware which means users can rent these resources based on their requirements, also IaaS allows fully isolated and independent virtual machines with their own operating systems.

**IaaS in Distributed OS and Modern OS Functionality:** (Red Hat, 2022) (javatpoint, 2022) (Saratchandran, 2023) (IBM, 2022) (Rosencrance, 2021)

**1-** **Resource Scaling:**

IaaS makes it possible for resources to scale dynamically, which is essential to distributed and modern operating systems. Pay-as-you-go models are supported by this feature, allowing companies to pay for the resources they utilize.

**2-** **Storage and Networking:**

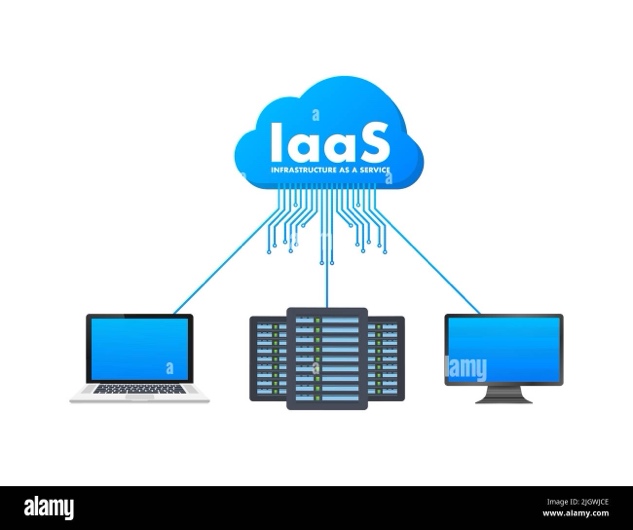
IaaS offers networking features including virtual private networks (VPNs) and bandwidth on demand in addition to scalable and secure storage solutions.

**3-** **Virtualization:**

IaaS is a virtualization technique that makes it possible to build virtual computers. Multiple virtualized environments may be included into a single physical configuration, which is a crucial feature of modern operating systems.

**Examples:**

As examples of well-known IaaS systems, Amazon Web Services (AWS) and Microsoft Azure provide a wide range of cloud computing tools.



**Platform as a Service (PaaS):** (Red Hat, 2022) (javatpoint, 2022) (IBM, 2022) (Saratchandran, 2023) (Rosencrance, 2021)

**Definition:**

Is a cloud computing service that offers a platform to the users so they can build, execute, and manage applications easily. It helps in managing the complex infrastructure, such as servers and databases, that is normally required for app development, it simplifies the procedure. PaaS refers to an additional layer above fundamental computing services, providing extra tools and services to facilitate the development and deployment of programs by developers.

**PaaS in Distributed OS and Modern OS Functionality:** (Red Hat, 2022) (javatpoint, 2022) (Saratchandran, 2023) (IBM, 2022) (Rosencrance, 2021)

**1-** **Development Tools:**

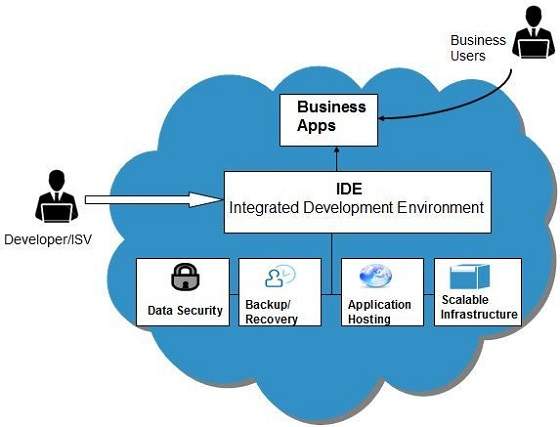
PaaS provides a set of environments and development tools.

**2-** **Middleware Integration:**

PaaS involves several middleware components which help in load balancing, application management, and application scaling.

**3-** **Automated Backups and Updates:**

The PaaS model takes care of the automatic processing of updates, patches, and backups—all crucial components of operating system management.



**3-** **Software as a Service (SaaS):** (Red Hat, 2022) (javatpoint, 2022) (IBM, 2022) (Saratchandran, 2023) (Rosencrance, 2021)

**Definition:**

Is a method of using software over the Internet that often requires a membership. You don't need to worry about the technical complexities because the software provider handles everything for you, including security and upgrades. The program is simply used online whenever needed.

**SaaS in Distributed OS and Modern OS Functionality:** (Red Hat, 2022) (IBM, 2022) (Saratchandran, 2023) (Rosencrance, 2021)

**1-** **Software Management:**

SaaS improves on the standard operating system's notion of software management to a distributed cloud setting.

**2-** **Accessibility and Scalability:**

Customers may use SaaS apps from several devices through the Internet, providing a degree of accessibility similar to what current operating systems aim for in a dispersed setting. Scalability is controlled by the supplier and permits on-demand access to a greater or lesser number of services or features.

**3-** **Centralized Management and Updating:**

With the SaaS approach, the provider takes care of security, updates, and performance improvements, centralizing software management. This works in a distributed, cloud-based environment and is similar to how an operating system maintains apps.

**The relationship between operating systems (OS) and cloud computing systems:** (Red Hat, 2022) (javatpoint, 2022) (IBM, 2022) (Saratchandran, 2023) (Rosencrance, 2021)

Operating systems act as the essential interface between the hardware and the software:

**Functionalities and Operations of Modern OS in Cloud Computing Service Models:** (Red Hat, 2022) (javatpoint, 2022) (IBM, 2022) (Saratchandran, 2023) (Rosencrance, 2021)

**Resource Management:**

Operating systems in very helpful because the OS help in allocating hardware resources among various application

**Virtualization Support:**

The operating system in cloud environments, mainly in Infrastructure as a Service (IaaS) models, is in charge of generating and overseeing these virtual resources.

**Automation and Simplification:**

The OS mainly in PaaS and SaaS provides tools that help the deployment to be automated, such as scaling, and management of applications.

Performance Monitoring and Optimization:

One of the important roles of OS in cloud computing is to watch out the performance of applications.

**Functionalities and Operations of Distributed OS in Cloud Computing Service Models:** (Red Hat, 2022) (javatpoint, 2022) (IBM, 2022) (Saratchandran, 2023) (Rosencrance, 2021)

**Resource Distribution and Management**: The Distributed OS distributes resources across multiple physical and virtual machines to ensure good performance.

**Concurrency Control:** Manage multiple processes to ensure efficient execution.

**Network Services and Communication:** ensure efficient communication and data transfer between different nodes on the cloud.

**Manage problems and ensure high availability:** by putting methods like redundancy and backup mechanisms.

**Self-Healing:** Automatically detecting and recovering from system failures.

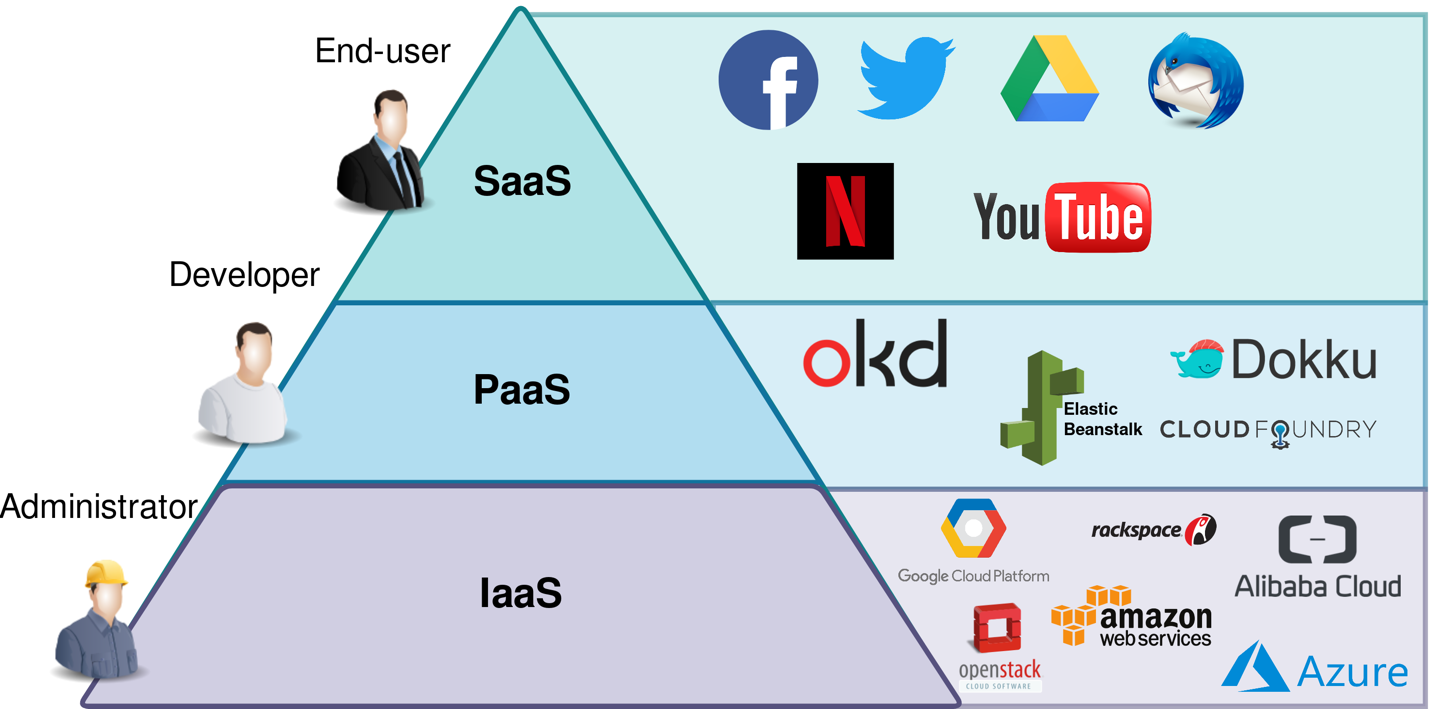
Examples of DOS:

Xen

Apache CloudStack

(Red Hat, 2022) (javatpoint, 2022) (IBM, 2022) (Saratchandran, 2023) (Rosencrance, 2021)

|  |  |  |  |
| --- | --- | --- | --- |
|  | **IaaS (Infrastructure as a Service)** | **PaaS (Platform as a Service)** | **SaaS (Software as a Service)** |
| **What It Is** | Providing virtual computers and storage over the internet. | An online place to create and run apps. | Using apps over the internet. |
| **Scaling** | You can easily get more or less virtual computers needed. | Makes it easy to make your app bigger or smaller as needed. | The app can serve more or fewer users easily |
| **Security** | Focuses on keeping the computers and networks safe. | Keeps the app-making tools and processes safe (middleware). | Ensures data security at the application level. |
| **Cost** | Can save money on buying computers and storage. | Can be cheaper than old ways of making apps. | Paying regularly (like a subscription) can save money for users. |
| **How It Uses Operating Systems** | Manages resources across multiple virtual machines. | Supports network services and communication for distributed app development. | Focuses on accessibility and software management in a distributed environment. |
| **Compared to Old Systems** | converting from a large physical computer collection to virtual ones as needed. | Move from traditional software development to cloud-based platforms. | transition from software installed on individual machines to cloud-based access. |



# References:

## **Task 1 references:**

Abraham Silberschatz and Peter Galvin (2011) *Operating System Concepts Essentials, 2nd Edition - pub.ro*, *Essentials, 2nd Edition*. Available at: https://ocw.cs.pub.ro/courses/\_media/poo-is/laboratoare/operating\_system\_concepts\_essentials\_2nd\_-\_abraham\_silberschatz.pdf (Accessed: 19 January 2024).

GeeksforGeeks (2024) *Virtual memory in operating system*, *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/virtual-memory-in-operating-system/ (Accessed: 19 January 2024).

Csc-Knu (2017) *CSC-KNU/Sys-prog: KNU Course on building language processors*, *GitHub*. Available at: https://github.com/csc-knu/sys-prog (Accessed: 19 January 2024).

Gillis, A.S., Peterson, S. and Lelii, S. (2021) *What is virtual memory?*, *Storage*. Available at: https://www.techtarget.com/searchstorage/definition/virtual-memory#:~:text=Virtual%20memory%20uses%20both%20hardware,though%20it%20were%20main%20memory. (Accessed: 19 January 2024).

## **Task 2 references:**

www.javatpoint.com. (2011). *OS Page Table - javatpoint*. [online] Available at: https://www.javatpoint.com/os-page-table.

‌

## **Task 5 references:**

Splunk (2023) *Distributed Systems explained*, *Splunk*. Available at: https://www.splunk.com/en\_us/blog/learn/distributed-systems.html#:~:text=A%20distributed%20system%20is%20simply,a%20single%20device%20ran%20it. (Accessed: 27 January 2024).

javatpoint (2022) *Distributed operating system - javatpoint*, *www.javatpoint.com*. Available at: https://www.javatpoint.com/distributed-operating-system (Accessed: 27 January 2024).

GeeksforGeeks (2023a) *Comparison - centralized, decentralized and distributed systems*, *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/comparison-centralized-decentralized-and-distributed-systems/ (Accessed: 27 January 2024).

## **Task 6 references:**

javatpoin (2022) *Concurrency in operating system - javatpoint*, *www.javatpoint.com*. Available at: https://www.javatpoint.com/concurrency-in-operating-system#:~:text=What%20is%20Concurrency%3F,shared%20memory%20or%20message%20passing. (Accessed: 27 January 2024).

Libretexts (2021) *5.1: Introduction to concurrency*, *Engineering LibreTexts*. Available at: https://eng.libretexts.org/Courses/Delta\_College/Operating\_System%3A\_The\_Basics/05%3A\_Process\_Synchronization/5.1%3A\_Introduction\_to\_Concurrency (Accessed: 27 January 2024).

Tutorialspoint (2022) *Concurrency in operating system*, *Tutorialspoint*. Available at: https://www.tutorialspoint.com/concurrency-in-operating-system (Accessed: 27 January 2024).

## **Task 7 references:**

.javatpoint (2022) *Remote method invocation (RMI) - javatpoint*, *www.javatpoint.com*. Available at: https://www.javatpoint.com/RMI (Accessed: 27 January 2024).

Awati, R. (2023) *What is remote method invocation (RMI)?: Definition from TechTarget*, *TheServerSide.com*. Available at: https://www.theserverside.com/definition/Remote-Method-Invocation-RMI#:~:text=Remote%20Method%20Invocation%20(RMI)%20is,JVM%20in%20a%20distributed%20network. (Accessed: 27 January 2024).

GeeksforGeeks (2022) *What is RPC mechanism in distributed system?*, *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/what-is-rpc-mechanism-in-distributed-system/ (Accessed: 27 January 2024).

Lawrence Williams and Williams, L. (2023) *Remote procedure call (RPC) protocol in Distributed System*, *Guru99*. Available at: https://www.guru99.com/remote-procedure-call-rpc.html (Accessed: 27 January 2024).

Neha, N. (2018) *Difference between RPC and RMI (with comparison chart)*, *Tech Differences*. Available at: https://techdifferences.com/difference-between-rpc-and-rmi.html (Accessed: 27 January 2024).

GeeksforGeeks (2023) *Difference between synchronous and asynchronous transmission*, *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/difference-between-synchronous-and-asynchronous-transmission/ (Accessed: 27 January 2024).

BYJUS (2023) *Difference between synchronous and asynchronous transmission*, *BYJUS*. Available at: https://byjus.com/gate/difference-between-synchronous-and-asynchronous-transmission/ (Accessed: 27 January 2024).

## **Task 8 references:**

Red Hat (no date) *IaaS vs. paas vs. SAAS*, *Red Hat - We make open source technologies for the enterprise*. Available at: https://www.redhat.com/en/topics/cloud-computing/iaas-vs-paas-vs-saas (Accessed: 28 January 2024).

javatpoint (2022) *Cloud service models - javatpoint*, *www.javatpoint.com*. Available at: https://www.javatpoint.com/cloud-service-models (Accessed: 28 January 2024).

IBM (2022) *IaaS vs. paas vs. SAAS*, *IBM*. Available at: https://www.ibm.com/topics/iaas-paas-saas (Accessed: 28 January 2024).

Saratchandran, V. (2023) *Cloud service models SAAS, iaas, paas - choose the right one for your business*, *Fingent*. Available at: https://www.fingent.com/blog/cloud-service-models-saas-iaas-paas-choose-the-right-one-for-your-business/ (Accessed: 28 January 2024).

Rosencrance, L. (2021) *SAAS vs. iaas vs. paas: Differences, Pros, cons and examples*, *WhatIs*. Available at: https://www.techtarget.com/whatis/SaaS-IaaS-PaaS-Comparing-Cloud-Service-Models#:~:text=There%20are%20three%20major%20cloud,use%20and%20subscription%20pricing%20models. (Accessed: 28 January 2024).