WARDIERE INC.

protection Mechanisme

Presented by : merkhoufi mohamed naoui

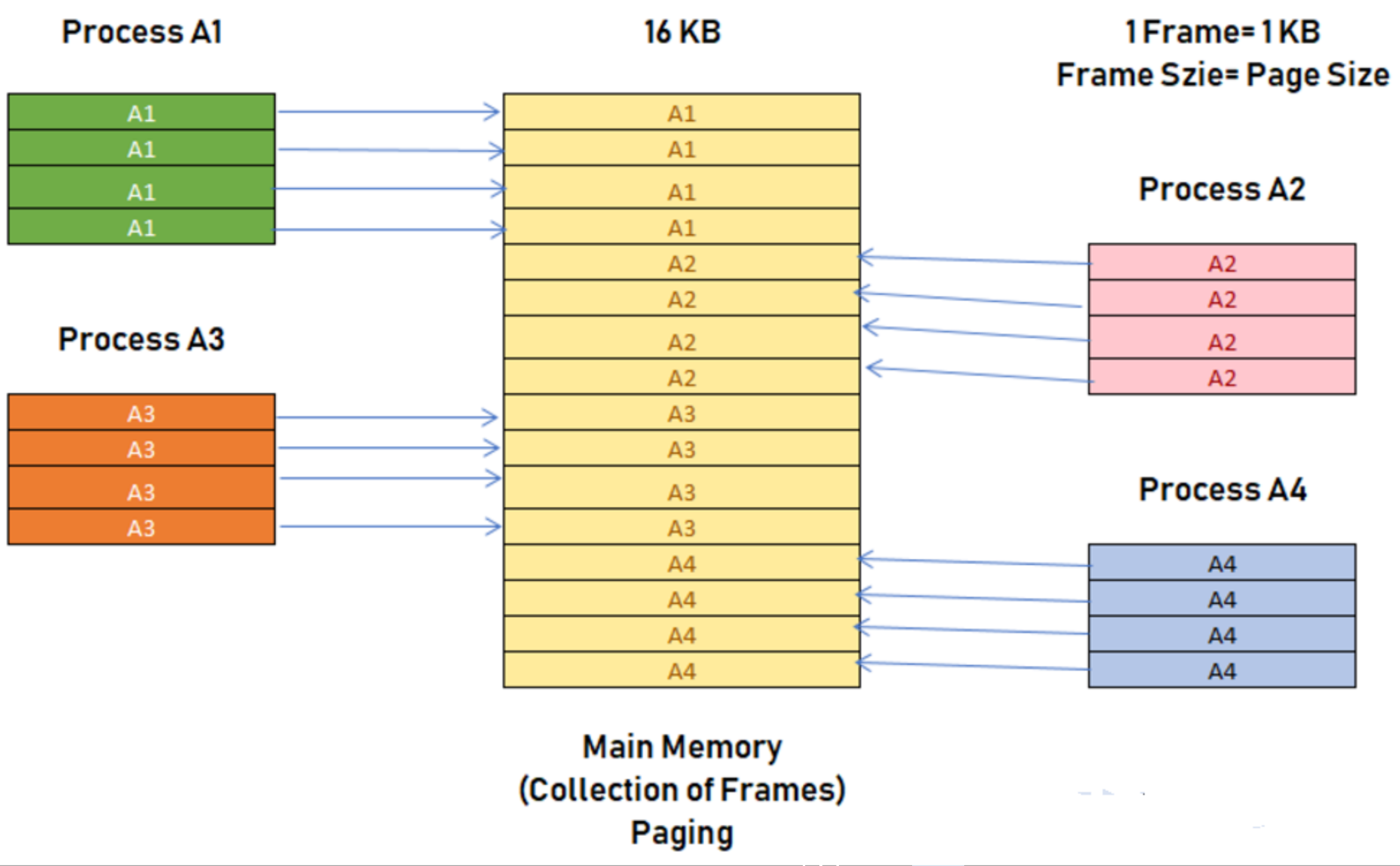
1) paging
2)memory protection
3)kernel memory allocation

1- paiging

Definition: Paging is a memory management scheme that eliminates the need for contiguous allocation of physical memory. It breaks physical memory into fixed-size blocks called "pages" and divides the virtual memory space used by processes into equal-sized blocks called "frames." The main goal of paging is to efficiently manage memory and allow processes to access memory without worrying about physical address contiguity.

How it Works:

- The virtual memory used by programs is split into small blocks (pages), typically 4 KB or 8 KB in size
- Similarly, physical memory is divided into frame-sized blocks, which correspond to the size of a page
- Page tables are used to keep track of where the pages of a process are stored in physical memory. When a program accesses a virtual address, the operating system (OS) uses the page table to map the virtual address to the correct physical address
- Memory Mapping: When a program requests memory, it uses virtual addresses that the system maps to physical addresses. This translation is handled by the Memory Management Unit (MMU), which stores the mapping in the page tables


Process A1
16 KB
1 Frame= 1 KB
Frame Szie= Page Size
A1
A1
A1
A1
A1
A1
A1
A1
Process A2
A2
A2
A2
A2
A2
A2
A2
A2
Process A3
A3
A3
A3
A3
A3
A3
A3
A3
Process A4
A4
A4
A4
A4
A4
A4
A4
A4
Main Memory
(Collection of Frames)
Paging

Advantages of Paging:

- Eliminates the problem of fragmentation (physical memory doesn't need to be contiguous).
- Allows for efficient memory use and virtual memory management, as processes can be loaded into any available physical memory location.
- Enables swapping, allowing the OS to move pages in and out of disk storage (paging to disk when the physical memory is full).

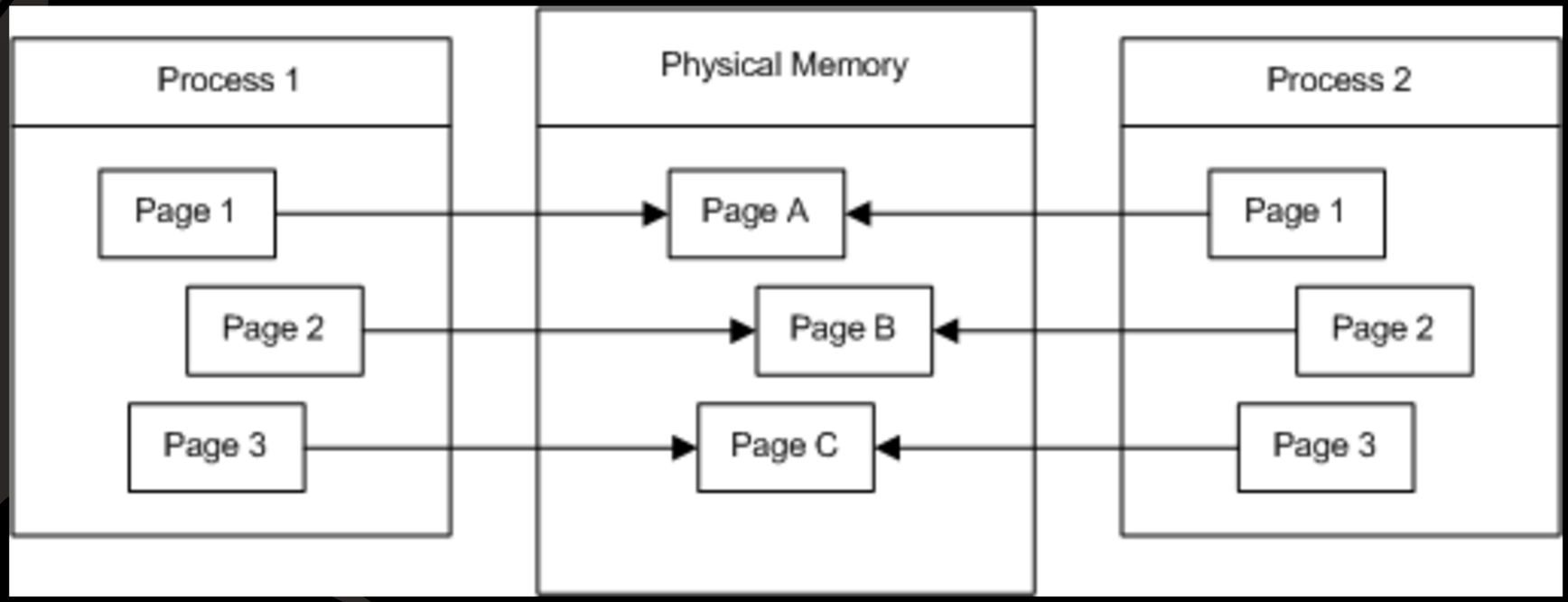
Protection in Paging:

The MMU ensures that memory access checks are enforced, preventing a program from accessing memory that is not allocated to it

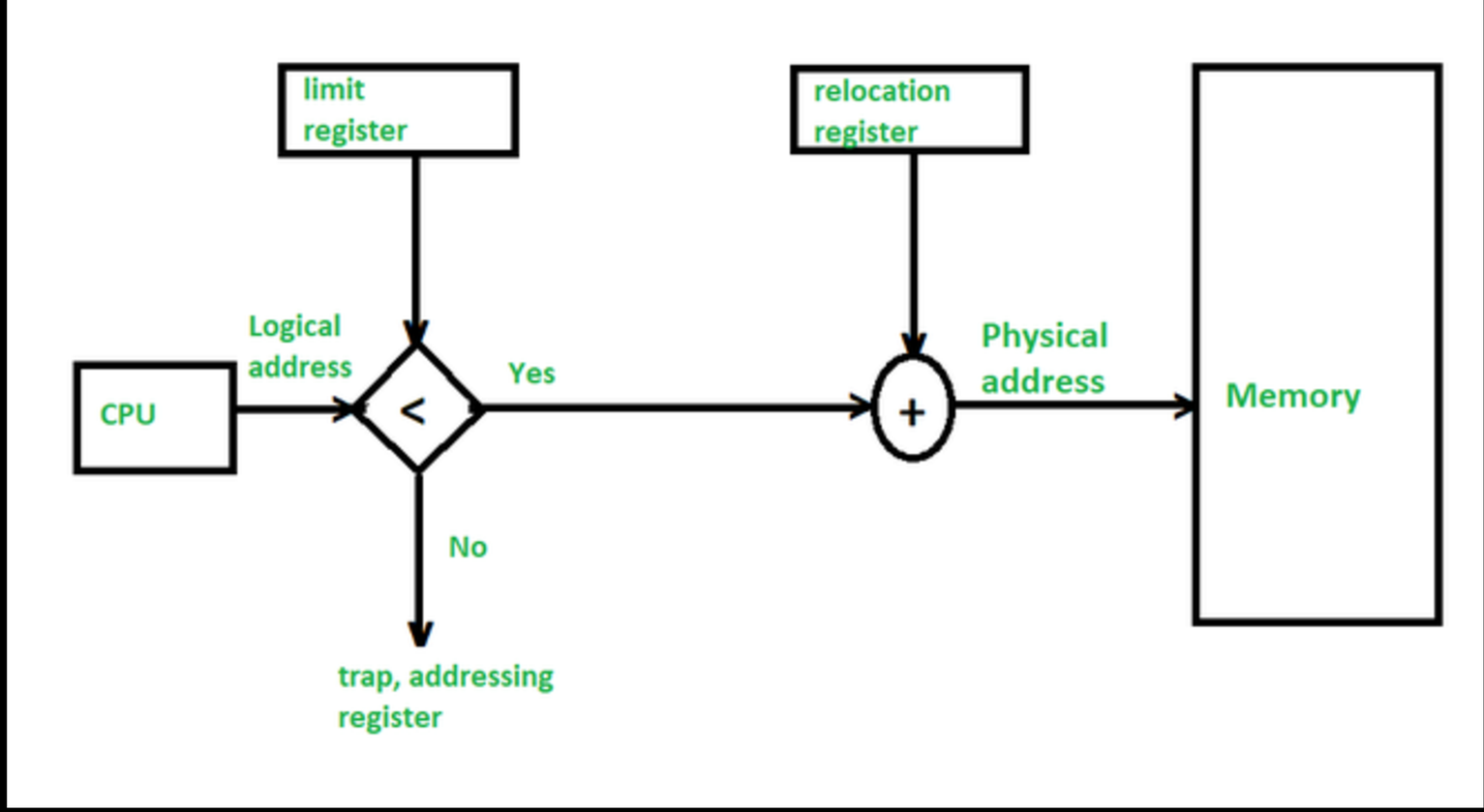
The OS can assign permissions to pages (e.g., read-only or read-write), ensuring that data in memory is not tampered with or modified by unauthorized .processes

2. Memory Protection

Definition: Memory protection is a set of mechanisms that ensure processes or programs cannot access or modify memory that they are not authorized to access. This protection ensures the stability and security of the system by preventing bugs or malicious programs from damaging system resources or other programs' .data



In the above diagram, when the scheduler selects a process for the execution process, the dispatcher, on the other hand, is responsible for loading the relocation and limit registers with the correct values as part of the context switch as every address generated by the CPU is checked against these 2 registers, and we may protect the operating system, programs, and the data of the users from being altered by this running process.



Need of Memory protection:

Memory protection prevents a process from accessing unallocated memory in OS as it stops the software from seizing control of an excessive amount of memory and may cause damage that will impact other software which is currently being used or may create a loss of saved data. These resources of memory protection also help in detecting malicious or harmful applications, that may after .damaged the processes of the operating system

How it Works

Isolation: Each process runs in its own virtual address space, ensuring that one process cannot directly access the memory of another process. This isolation prevents one program from overwriting the memory of another program or the OS itself

Access Control: The OS assigns different access rights to different regions of memory. These can include

Read-only: Memory pages that cannot be modified

Read-write: Memory pages that can be modified

No access: Memory pages that cannot be accesse

Memory Segmentation: Along with paging, some systems also use segmentation, which divides memory into logical segments (e.g., code, data, stack). Each segment can have different protection levels

Key Features of Memory Protection:

Virtual Memory: Virtual memory allows each process to have a "private" address space. The OS uses the MMU to translate virtual addresses to physical addresses. If a program tries to access memory it shouldn’t, the MMU raises a segmentation fault or page .fault

Access Control Lists (ACLs): The OS can use access control lists to define who has access to what memory. This is typically used in the kernel to protect sensitive system data

Privilege Levels: The OS uses privilege levels (rings) to enforce access restrictions. Ring 0 (kernel mode) has full access to the system, while Ring 3 (user mode) is restricted to limited .access

Protection Mechanisms

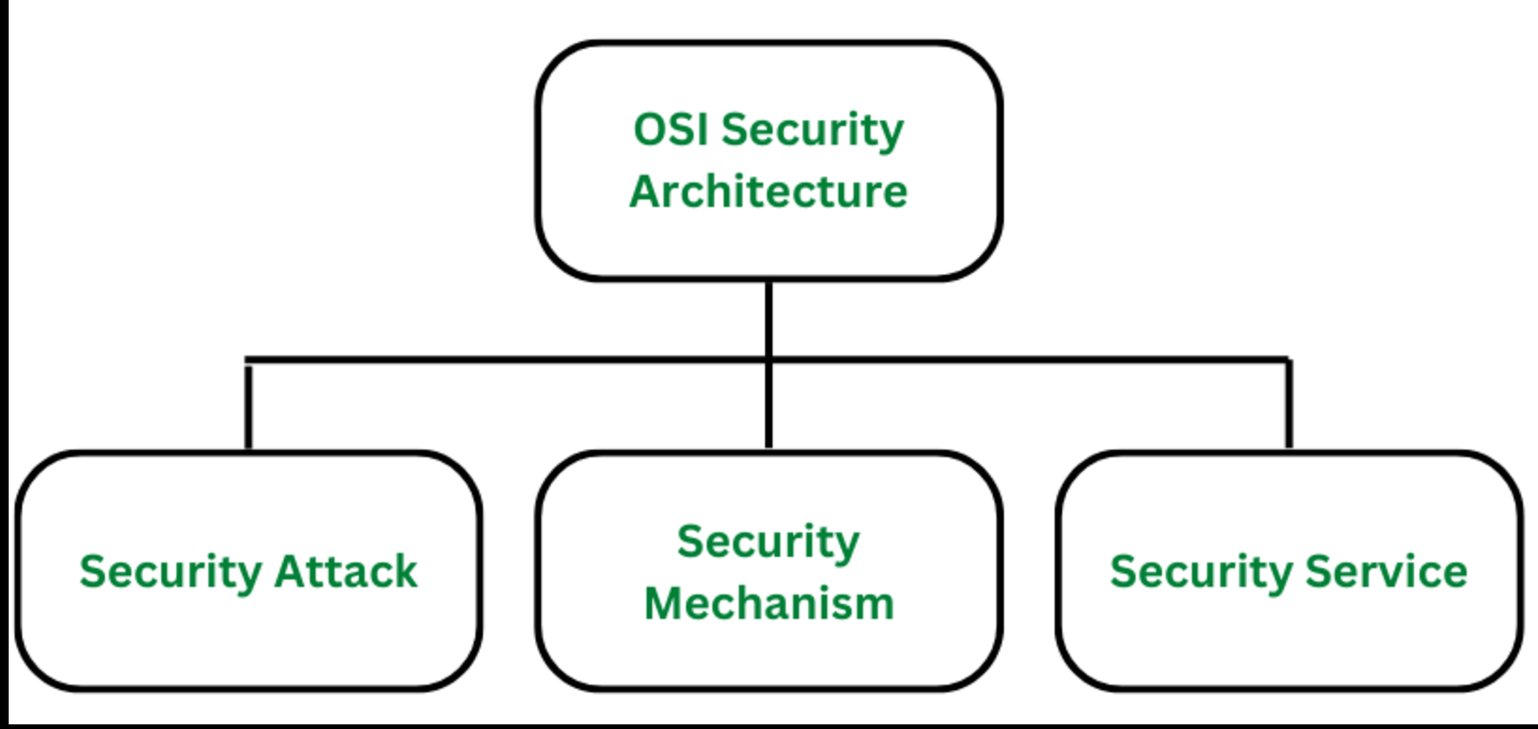
Hardware Support: Most modern CPUs have built-in memory protection mechanisms (like NX-bit for preventing execution in data regions) to prevent malicious programs from executing code in specific .memory areas

User-space and Kernel-space Separation: Kernel code operates in privileged mode (ring 0) and has unrestricted access to all resources. User-space code runs in user mode (ring 3) and is restricted from .accessing kernel memory

The mechanism that is built to identify any breach of security or attack on the organization, is called a security mechanism. Security Mechanisms are also responsible for protecting a system, network, or device against unauthorized access, tampering, or other security threats.

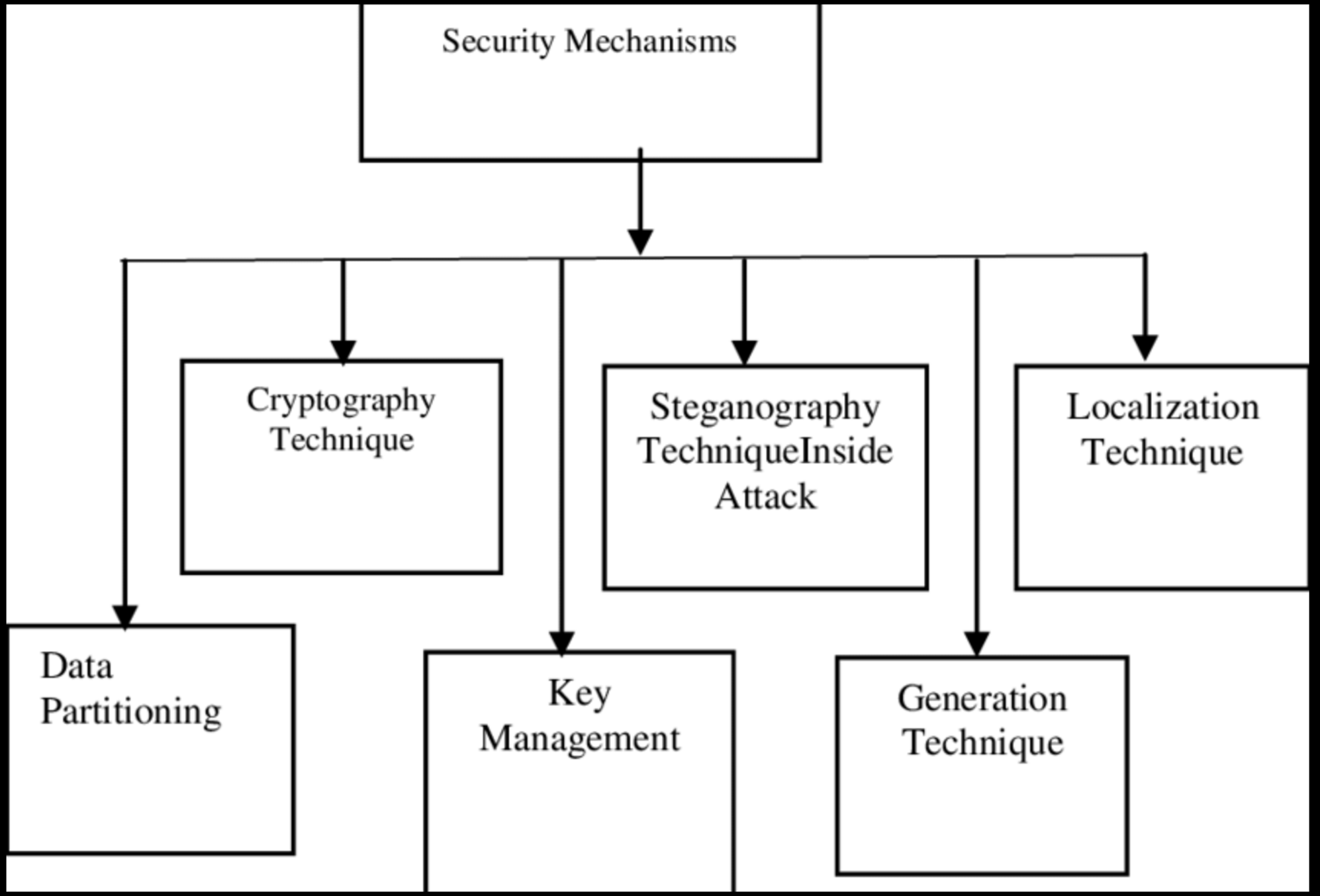
:Encipherment (Encryption)

Encryption involves the use of algorithms to transform data into a form that can only be read by someone with the appropriate decryption key. Encryption can be used to protect data it is transmitted over a network, or to protect data when it is stored on a device



Traffic padding: Traffic Padding is a technique used to add extra data to a network traffic stream in an attempt to obscure the true content of the traffic and make it more difficult to analyze.

Routing control: Routing Control allows the selection of specific physically secure routes for specific data transmission and enables routing changes, particularly when a gap in security is


Security Mechanisms
Cryptography Technique
Steganography TechniqueInside Attack
Localization Technique
Data Partitioning
Key Management
Generation Technique

3. Kernel Memory Allocation

Definition: Kernel memory allocation refers to how the operating system kernel allocates and manages memory for processes running in kernel space. The kernel is responsible for managing system resources, including memory, and allocating space to processes as needed.

How it Works:

The kernel maintains its own memory pool, which is separate from user space memory. It allocates memory for its own use and for user processes during their execution.

- **Dynamic Allocation:** The kernel dynamically allocates memory as needed. This includes both static and dynamic memory allocations: Static Memory Allocation: Fixed size memory areas that are allocated at compile time and remain the same throughout the program's execution
- **Dynamic Memory Allocation:** Memory that is allocated at runtime using mechanisms such as the heap and stack.
- **Heap Memory:** This is used for dynamic memory allocation (e.g., when objects are created during runtime). It grows and shrinks as needed, and the OS must manage it to avoid fragmentation.
- **Stack Memory:** The kernel also manages the stack, which stores local variables and function call information. Each thread has its own stack, which is used for function calls and storing local data.

Kernel Memory Allocation Management:

Slab Allocator: The kernel uses a memory management system like the slab allocator to manage memory more efficiently. This method groups memory into slabs (chunks of fixed-size memory) and allocates them to objects of the same type, minimizing fragmentation and improving memory usage

Buddy System: Another technique used in kernel memory allocation is the buddy system, where memory is divided into blocks of different sizes, and adjacent blocks are .merged to reduce fragmentation

Page Allocation: The kernel manages memory at the page level, allocating memory in blocks that correspond to pages in physical memory. The kernel keeps track of these pages and ensures that memory is not leaked or overwritten

Security Considerations:

- **Kernel Memory Protection:** Kernel memory is generally protected from direct access by user processes to prevent them from tampering with sensitive system data
- **Memory Leaks:** The kernel must also ensure that allocated memory is properly freed when it is no longer needed to prevent memory leaks and maintain system stability
- **Race Conditions and Synchronization:** Kernel memory allocation must also handle race conditions—situations where multiple processes attempt to allocate or free memory simultaneously. Proper synchronization mechanisms (like locks and semaphores) are required to ensure memory is allocated and freed correctly.

Conclusion

These three mechanisms—Paging, Memory Protection, and Kernel Memory Allocation—are essential for ensuring that memory is used efficiently and securely. Paging helps manage memory allocation, memory protection ensures processes don’t interfere with each other, and kernel memory allocation ensures the operating system has the resources it needs to function while maintaining system .integrity

THANK YOU