Exam: Final

Semester: 3rd

Subject: Operating System

ID: 15499

Amir Abbas

Teacher: M. Dawood

Q1:

Ans.1: **DEADLOCKS**:

Deadlocks possible when resources are shared between concurrent process.

- System can be depicted using a resource allocation graph.
- Resource allocation graph may be used to check for deadlocks.

Deadlocks prevention:

Ensure that at least one of the necessary conditions does not hold.

- Mutual exclusion.
  - Not required for sharable resources (Read Only files)
  - Must hold for non sharable resources.
  - Cannot prevent deadlock by denying mutual-exclusion condition there are non sharable resources.

- Hold and unit.
    - Must guarantee that whenever a process request a resources, it does not hold any other resources.
    - Requires process to request and be allocated all its resources before begins execution.
    - System call requesting resources proceed all others system calls.
    - Always process to request resources only when the process has none.
    - Law resource utilization, starvation possible.

Deadlock avoidance:

Requires that the system has some additional a prior information available.

- Simplest and most useful model requires that each process declare the maximum number of resources of each type dated may need.
- Resource – allocation state is defined by the number of available in allocated resources and the maximum demands of the process.

Safe state:

- When process requests and unvariable resource system must decide if immediate allocation leaves the system in a safe state.
- System is a safe state if there exist a sequence $< P_1 , P_2 \ldots P_n >$ of all the process in the systems such that for each P_i , resources that P_i can still requests can be satisfied by currently available resources + resources held by all the P_i
- That is

- If P_i resources need or not immediately available, then P_i can wait until all P_i have finished.
- When P_i is finished, P_i can obtain needed resources, execute return allocated resources and terminate.
- When P_i terminates $P_i + 1$ can obtain its needed resources, and so on.

Basic facts:

- If a system is in a safe state no deadlocks.
- If a system is in unsafe state ➔ possibility of deadlock.
- Avoidance ➔ ensure that a system will never enter an unsafe state.

Q.2:

Ans.2: **Dynamic loading:**

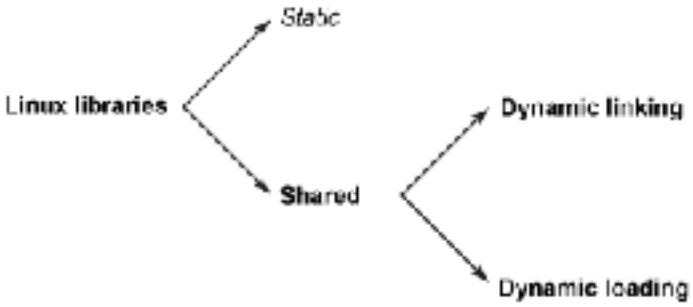
Dynamic loading means loading the library (or any other binary for that matter) into the memory during load or run-time.

Dynamic loading can be imagined to be similar to plugins, that is an exe can actually execute before the dynamic loading happens (The dynamic loading for example can be created using LoadLibrary call in C or C++)

Dynamic linking:

Dynamic linking refers to the linking that is done during load or run-time and not when the exe is created.

In case of dynamic linking the linker while creating the exe does minimal work. For the dynamic linker to work it actually has to load the libraries too. Hence it's also called linking loader.



This answer will suppose you know basic Linux command.

In Linux, there are two types of libraries: static or shared.

In order to call functions in a static library you need to statically link the library into your executable, resulting in a static binary.

While to call functions in a shared library, you have two options.

First option is dynamic linking, which is commonly used - when compiling your executable you must specify the shared library your program uses, otherwise it won't even compile. When your program starts it's the system's job to open these libraries, which can be listed using the `ldd` command.

The other option is dynamic loading - when your program runs, it's the program's job to open that library. Such programs are usually linked with libdl, which provides the ability to open a shared library.

Q.3:

Ans.3: The most suitable component of an operating system of an operating system that suited to ensure fair, secure, orderly and efficient use of memory is memory management system.

The task of memory management includes keeping track of used and free memory space at well all when where and how much memory to allocate and deallocate.

It is also responsible for swapping process in and out of main memory. The purpose of a memory management is to ensure fair secure orderly and efficient use of a memory.

Q.4:

Ans.4:

Difference between Symmetric and Asymmetric Encryption:

While communicating on an unsecured medium like the internet, you have to be careful about the confidentiality of the information you are sharing with other. There are two techniques use to preserve the confidentiality of your message, Symmetric and Asymmetric Encryption. The fundamental difference that distinguishes symmetric and asymmetric encryption is that **symmetric encryption** allows encryption and decryption of the message with the same key.

On the other hand, **asymmetric encryption** uses the public key for the encryption, and a private key is used for decryption. To acknowledge some more differences between symmetric and asymmetric encryption have a look at the comparison chart shown below.

Comparison Chart

Basis for Comparison	Symmetric Encryption	Asymmetric Encryption
Basic	Symmetric encryption uses a single key for both encryption and Decryption.	Asymmetric encryption uses a different key for encryption and decryption.

Performance	Symmetric encryption is fast in execution.	Asymmetric Encryption is slow in execution due to the high computational burden.
Algorithms	DES, 3DES, AES, and RC4.	Diffie-Hellman, RSA.
Purpose	The symmetric encryption is used for bulk data transmission.	The asymmetric encryption is often used for securely exchanging secret keys.

Q.5:

Ans.5: **Difference between Internal Fragmentation and External Fragmentation:**

As processes are loaded and removed from memory, the free memory space is broken into little pieces. It happens after sometimes that processes cannot be allocated to memory blocks considering their small size and memory blocks remains unused. This problem is known as Fragmentation.

Internal Fragmentation

Memory block assigned to process is bigger. Some portion of memory is left unused, as it cannot be used by another process. The internal fragmentation can be reduced by effectively assigning the smallest partition but large enough for the process.

External Fragmentation

Total memory space is enough to satisfy a request or to reside a process in it, but it is not contiguous, so it cannot be used. External fragmentation can be reduced by compaction or shuffle memory contents to place all free memory together in one large block. To make compaction feasible, relocation should be dynamic.

Following are the important differences between Internal Fragmentation and External Fragmentation.

Sr. No.	Key	Internal Fragmentation	External Fragmentation
1	Definition	When there is a difference between required memory space vs allotted memory space, problem is termed as Internal Fragmentation.	When there are small and non-contiguous memory blocks which cannot be assigned to any process, the problem is termed as External Fragmentation.
2	Memory Block Size	Internal Fragmentation occurs when allotted memory blocks are of fixed size.	External Fragmentation occurs when allotted memory blocks are of varying size.
3	Occurrence	Internal Fragmentation occurs when a process needs more space than the size of allotted memory block or use less space.	External Fragmentation occurs when a process is removed from the main memory.

4	Solution	Best Fit Block Search is the solution for internal fragmentation.	Compaction is the solution for external fragmentation.
5	Process	Internal Fragmentation occurs when Paging is employed.	External Fragmentation occurs when Segmentation is employed.

Q.6:

Ans.6:

Memory Allocation List

1. First Fit
2. Best fit
3. Worst fit
4. Buddy's system

1. First Fit

In the first fit approach is to allocate the first free partition or hole large enough which can accommodate the process. It finishes after finding the first suitable free partition.

Advantage

Fastest algorithm because it searches as little as possible.

Disadvantage

The remaining unused memory areas left after allocation become waste if it is too smaller. Thus request for larger memory requirement cannot be accomplished.

2. Best Fit

The best fit deals with allocating the smallest free partition which meets the requirement of the requesting process. This algorithm first searches the entire list of free partitions and considers the smallest hole that is adequate. It then tries to find a hole which is close to actual process size needed.

Advantage

Memory utilization is much better than first fit as it searches the smallest free partition first available.

Disadvantage

It is slower and may even tend to fill up memory with tiny useless holes.

3. Worst fit

In worst fit approach is to locate largest available free portion so that the portion left will be big enough to be useful. It is the reverse of best fit.

Advantage

Reduces the rate of production of small gaps.

Disadvantage

If a process requiring larger memory arrives at a later stage then it cannot be accommodated as the largest hole is already split and occupied.

4.Buddy's System

In buddy system, sizes of free blocks are in form of integral power of 2. E.g. 2, 4, 8, 16 etc. Up to the size of memory. When a free block of size 2k is requested, a free block from the list of free blocks of size 2k is allocated. If no free block of size 2k is available, the block of next larger size, 2k+1 is split in two halves called buddies to satisfy the request.

Example

Let total memory size be 512KB and let a process P1, requires 70KB to be swapped in. As the hole lists are only for powers of 2, 128KB will be big enough. Initially no 128KB is there, nor are blocks 256KB. Thus 512KB block is split into two buddies of 256KB each, one is further split into two 128KB blocks and one of them is allocated to the process. Next P2 requires 35KB. Rounding 35KB up to a power of 2, a 64KB block is required.

So when 128KB block is split into two 64KB buddies. Again a process P3(130KB) will be adjusted in the whole 256KB. After satisfying the request in this way when such block is free, the two blocks/buddies can be recombined to form the twice larger original block when it is second half buddy is also free.

Advantage

Buddy system is faster. When a block of size 2k is freed, a hole of 2k memory size is searched to check if a merge is possible, whereas in other algorithms all the hole list must be searched.

Disadvantage

It is often become inefficient in terms of memory utilization. As all requests must be rounded up to a power of 2, a 35KB process is allocated to 64KB, thus wasting extra 29KB causing internal fragmentation. There may be holes between the buddies causing external fragmentation.

Q.7:

Ans.7: User Level Threads

In this case, the thread management kernel is not aware of the existence of threads. The thread library contains code for creating and destroying threads, for passing message and data between threads, for scheduling thread execution and for saving and restoring thread contexts. The application starts with a single thread.

Advantages

- Thread switching does not require Kernel mode privileges.
- User level thread can run on any operating system.
- Scheduling can be application specific in the user level thread.
- User level threads are fast to create and manage.

Disadvantages

- In a typical operating system, most system calls are blocking.
- Multithreaded application cannot take advantage of multiprocessing.