MSc IT (S3) 03
Exam. Code:M3OPS

Operating System

SEMESTER-III

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY

BLOCK- 2

KRISHNA KANTA HANDIQUI STATE OPEN UNIVERSITY

Subject Experts

Prof. Anjana Kakati Mahanta, Gauhati University

Prof. (Retd.) Pranhari Talukdar, Gauhati University

Dr. Jyotiprokash Goswami, Assam Engineering College

Prof. Shikhar Kumar Sarma, Gauhati University

Programme Co-ordinator

Dr. Ridip Dev Choudhury, KKHSOU

Course Co-ordinator :

Dr. Ridip Dev Choudhury, KKHSOU

Dr. Tapashi Kashyap Das, KKHSOU

Dr. Sruti Sruba Bharali, KKHSOU

SLM Preparation Team

UNITS	CONTRIBUTORS
9	**Mr. Dipankar Dutta,** NERIM
10	**Dr. Swapnani Gogoi,** IDOL
11, 12	**Mr. Adarsh Pradhan,** GIMT
13, 14, 15	**Dr. Utpal Barman,** GIMT

Editorial Team

Content : **Dr. Pranab Das,** Assam Don Bosco University

Structure, Format & Graphics : Dr. Sruti Sruba Bharali, KKHSOU

January, 2022 **ISBN: .....**

This Self Learning Material (SLM) of the Krishna Kanta Handiqui State Open University is made available under a Creative Commons Attribution-Non Commercial-Share Alike 4.0 License (international): http://creativecommons.org/licenses/by-nc-sa/4.0/

Printed and published by Registrar on behalf of the Krishna Kanta Handiqui State Open University

Headquarters: Patgaon, Rani Gate, Guwahati-781017

City Centre: NH 37, Resham Nagar, Khanapara, Guwahati-781022; Web:www.kkhsou.in

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY

Operating System

Block-2

DETAILED SYLLABUS

		Pages
UNIT 9 :	**FILE SYSTEM** Definition of File, File naming, File types (directory, regular), Sequential access and Random access files, File attributes, Operations on file, Hierarchical directory structure, Path name (relative and absolute), Operation on directories, File system implementation techniques.	**7-25**
UNIT 10:	**I/O MANAGEMENT** Basic principles I/O Operations, I/O Devices, Device controllers, DMA, Principles of I/O Software, its goals, Interrupt Handlers, Device Drivers.	**26-47**
UNIT 11:	**PROTECTION** Needs for protection, domain of protection, Example in UNIX, access matrix, Implementation of Access matrix.	**48-65**
UNIT 12:	**SECURITY** The security problem, Authentication, Password, password vulnerabilities, Encrypted password, One time password(OTP), Biometrics and its types, Intrusion detection (definition only).	**66-88**
UNIT 13:	**MULTIPROCESSOR SYSTEMS** Types of Multiprocessor Operating Systems, Multiprocessor OS Functions and Requirements, Multiprocessor synchronization.	**89-98**

UNIT 14: DISTRIBUTED OPERATING SYSTEMS **99-112**

Distributed Processing, Coping with Failures Models of Distributed systems, Remote procedure calls, distributed Shared Memory, Distributed File Systems.

UNIT 15: INTRODUCTION TO MULTIPROGRAMMING SYSTEM **113-128**

Queue management, I/O supervisors, memory management. File system, Disk and Drum Scheduling.

BLOCK INTRODUCTION

This is the *second block* of the course '***Operating System***'. Concepts like file system, I/O management, protection, security are covered in this unit. The block also coveres introductory topics on areas of multiprocessor systems, distributed operating systems and multiprogramming system. The units of this block are provided with various examples that illustrate the basic concepts.

This block comprises the following seven units :

Unit - 9 gives detailed overview of the file system. Topics like sequential access, random access along with different file system implementation techniques are covered in this unit.

Unit - 10 is concerned with I/O management. The unit covers areas of I/O management like I/O operations, I/O devices, DMA, interrupt handlers and device drivers.

Unit - 11 deals with the concepts of protection. Need for protection, access matrix and implementation of access matrix are some of the topic covered in this unit.

Unit - 12 concentrates on security. Authentication, password protection, biometrics and its types are some of the topics covered in this unit.

Unit - 13 introduces concepts on multiprocessor system. Topics like types of multiprocessor systems, its functions and requirements, multiprocessor synchronization are covered in this unit.

Unit - 14 includes the basic concepts of distributed operating system. Distributed processing, failure models of distributed systems, remote procedure calls, shared memory and distributed file systems are some of the topics covered in this unit.

Unit - 15 introduces the concepts on multiprogramming system. Topics like queue management, I/O supervisors, file system, disk and drum scheduling are covered in this unit.

The structure of Block 2 is as follows :

UNIT 9 : FILE SYSTEM

UNIT 10 : I/O MANAGEMENT

UNIT 11 : PROTECTION

UNIT 12 : SECURITY

UNIT 13 : MULTIPROCESSOR SYSTEMS

UNIT 14 : DISTRIBUTED OPERATING SYSTEMS

UNIT 15 : INTRODUCTION TO MULTIPROGRAMMING SYSTEM

UNIT 9: FILE SYSTEM

UNIT STRUCTURE

9.1 Learning Objectives

9.2 Introduction

9.3 File System

9.4 File Naming

9.5 File Attributes

9.6 File Types

9.7 Operations on File

9.8 Access Method

- 9.8.1 Sequential Access
- 9.8.2 Direct Access Files
- 9.8.3 Other Access Method

9.9 File Structure

9.10 Directory and Disk Structure

- 9.10.1 Storage Structure
- 9.10.2 Directory Overview
- 9.10.3 Single Level Directory
- 9.10.4 Two-level Directory
- 9.10.5 Tree-structured Directory
- 9.10.6 Acyclic Graph Directory

9.11. File System Implementation Techniques

- 9.11.1 Implementing Files
- 9.11.2 Contiguous Allocation
- 9.11.3 Linked List Allocation

9.12 Lets Us Sum Up

9.13 Further Reading

9.14 Answers ToCheck Your Progress

9.15 Model Questions

1.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- learn about the concept of file system in operating system
- know the types of files and file attributes
- explain different file structures, their merits, and demerits
- explain different directory structures, their merits, and demerits
- discuss implementation techniques of afile system
- explain file allocation techniques-contiguous and linked list allocation

9.2 INTRODUCTION

This is the first unit of the block. In this unit you will be introduced to file system. The unit will cover topics like file naming, file attributes, file types and operations on file. The different types of access method and file structure are also covered in detail in this unit. Directory and disk structure along with different file system implementation techniques are also covered in this unit. In the next unit, the topics on I/O management will be covered. Let us now look into the definition of a file in the computer system.

9.3 FILE SYSTEM

The computer system has two different types of memories. Primary memory is used to store data or information temporarily and in secondary memory, data are stored permanently. In the case of temporary storage of data in primary memory, users should not concern about memory management and data transfer. The operating system deals with it. To store data in secondary devices such as magnetic disk, drum, floppy, USB, CD-Disk, optical –disk, magnetic tapes, etc. the operating system provides a logical platform. The operating system delivers a logical view of information storage such that the users can access and view the storage.A logical storage unit has been designed by the operating system, abstracting the physical properties of memory units is known as a ***file***.The mapping of files with physical devices is done by the operating system. Files are stored

permanently in secondary storage devices as they are non-volatile. Formally a file can be defined as a named collection of related information that is recorded in a permanent storage media. A user cannot access the secondary storage without intervening in the logical one. To write into or read from a storage device, the data has to be in a file. Normally a file can represent source programs, object codes, and data. Data files may be numeric, alphabetic, alphanumeric, binary, or text files, etc. In general, a file can be defined as a sequence of lines, bits, bytes, and records that may be a text file or formatted rigidly. The contents of a file are decided by the creator. Each file has its structure which depends on its type. A text file consists of a sequence of characters that are organized into lines and pages. A source file may contain a sequence of subroutines and functions. An object file contains a sequence of bytes organized into some blocks.

9.4 FILE NAMING

A file can be named as per user convenience and is referred to by a name. A name is usually a set of characters followed by a file extension such as *test.java.* The file extension defines the file type. In the given example test.java, the file contains code written in Java programming language. Some of the systems differentiate between uppercase and lower case letters in the file name. For example, Test.java and test.java may be considered as the same name by some of the systems, but for some systems both the files are different.The naming of a file follows some standard rules and regulations depending on the file type. Some of the common rules are – file name never starts with any numerals or any alphanumeric characters. Most of the file name does not allow any special character in between the name except underscore.

9.5 FILE ATTRIBUTES

File attributes can vary from one operating system to another. Some common file attributes of a file are:

- Name: The name of a file is the only symbolic information kept in files that can be read by a human.

- Type: The type of a file determines the structure of the file.
- Location: This information is a pointer to a device and the location of the file in the device.
- Size: The current size of the file in terms of bytes, words, or blocks and possibly the maximum size are included in this attribute.
- Protection: Access control information restricts the read, write, execution, and other operation on files to the users.
- Time, date, and user identification: This information regarding the time, date, and user details may be kept for creation, last modification, and last use of a file. This data can be useful for data protection, security, and monitoring.

All file information is used to store in a directory structure, which is also stored in the secondary storage. Typically a directory structure consists of the file name and its unique identifier.

9.6 FILE TYPES

An operating system does not support all types of files. Such as an executable file with a.exe extension is not supported in Linux-based operating systems. To access a file, it should be supported by the operating system. If the operating system recognizes the file type, only then it can work on it.

Usually, the type of a file included with a dot symbol, i.e. the name of a file is split into two parts – name of the file and type of the file. With this naming convention, it is easy to inform about the file type to the user and the operating system. Some of the common file types are: .doc, .pdf, .jpg, .c, .cpp, .xlxs, .mp4, .mp3 etc.

The type of a file is needed to indicate the type of the operations that can be performed on that file. Suppose the file type - .exe, .com and .bat are executable files .exe and .com are binary executable files, whereas.bat represents a batch file. Applications also use an extension to indicate file types. Suppose an 8085 microprocessor simulator IDE always expects a source code file with the extension .asm, and a user can write only .asm files using this application. Similarly in the MS Word application, we can create only .doc files and the application always expects .doc file type.

9.7 OPERATIONS ON FILE

Numerous operations can be performed on a file. Some of them are: create, write, read, reposition, delete, and truncate. These are the basic six operations we can perform on a file. Renaming a file is also an operation that can be performed on a file.

Creating a file: The creation of files involves two basic steps – first, search for a space to locate the file in the file system. Second, make an entry for the new file in the file directory.

Writing a file: For writing into a file it is required to make a system call specifying the file name and writing information to be written into the file. Using the file name the system call search for the file location in the directory. The pointer will be located from where the next write operation has to be taken. After completion of the write operation, the pointer location must be updated.

Reading a file: For reading a file, a system call has to specify the name of the file and the memory location from where the next block of the file is put. With the given name the system call search for the file in the file directory and the system needs a read pointer to execute the read operation. Once the read operation is completed, the read pointer is updated.

Repositioning within a file: The file directory is searched for asuitable entry. The pointer which refers to the currentfileposition isresetby a user value. Repositioning within a file does not require any actual input/ output operation. This repositioning within a file is also known as ***file seek.***

Deleting a file: To perform a delete operation on a file we have to search the file name in the file directory. After locating the searched file, release all the occupied spaces such that memory locations can be reused by other files. Then remove the file entry from the file directory.

Truncating a file: The user may want to delete all the file contents by keeping the attributes unchanged except the file length. That means, without deleting the file, users are allowed to re-write the file contents. During truncating a file, the file length may be changed.

These are the six basic file operations in an operating system. Other common operations are renaming a file and appending new information in an existing file. A user can copy a file or file contents to another file or I/O devices.

9.8 ACCESS METHOD

Files are used to store information. To access a file, the system provides different ways. But some of the systems provide only one access method for files.

9.8.1 Sequential Access

Sequential access is the simplest file access method. In this method, information is processed sequentially, i.e. one after another. In a system, sequential file access is most common. Editors and compilers usually access files sequentially. Read and write makes up bulk of the operation on a file. A read operation reads the next portion of the file andmoves the file pointer automatically.Similarly,the pointer is appended at the end of the file in a write operation. After writing a block into the file, the pointer is advanced to the next material which is going to be written.

9.8.2 Direct Access

The direct access method allows reading and writing file records rapidly without following any particular order. Thus a user can access block 13 and then block 54 without any order. To access data from a larger file, direct access files are found very useful and efficient. Database applications often use direct access file. When a query request is executed, only the concerned block of information is being accessed. To execute a selected query no need to load the complete database file into the memory. To implement the concept of the direct access method of file accessing, a block number has to be added as a parameter. So the read operation becomes *read n* to read the n^{th} block directly from the file.

All operating system does not support both the access method sequential and direct. Some system allows only sequential access and some of them allow direct access.

9.8.3 Other Access Methods

Other access methods are built on top of the direct access method. By creating an index for files, we can generate an efficient access method. In the index file, the address of the various blocks has been kept as a reference. To find a record in the file, first, search the index file and then use the pointer to access the file directly and access the required data or information from the file. The indexing concept can be related to the last page indexing of a book. For example, retail – price file might list thousands of products' names with their prices. To search the price of a particular product we can implement a binary search operation on the index fields. From this search, we can easily locate the block, where the desired contains are located, and accessed the data. This indexing structure helps us to locate a block in a large file doing a few I/O operations. For a large file, the index file also becomes a very large one. To optimize it, we can maintain another index for the index file. The primary index files will contain the pointers on the secondary index files from which we can have the actual data.



EXERCISE

1. What are the different types of files?
2. What is the difference between file and directory?
3. What are the fileaccess methods explain.
4. What are the operations we can perform on a file?

9.9 FILE STRUCTURE

To define the structure of a file the file type can be used. The file structure should be understood by the operating system. For example, the executable files should have a specific structure, such that the operating

system can understand where to load program codes, the first instruction, and what is the starting address. Some of the operating systems have built-in such a way that, they support more than one file structure. The operating systems which support multiple file structures may have a disadvantage. Some of the operating systems such as UNIX and MS-DOS impose a very minimal number of file structures. The Macintosh operating system also supports a minimal file structure. It expects two parts in a file – a *resource fork* and a *data fork*.The information of user interest such as the labels on a button is the contents of the resource fork. The system allows a user to modify this displayed information according to their interest. The data fork contains program code or data, i.e. the traditional file contents.

9.10 DIRECTORY AND DISK STRUCTURE

A computer system typically can have thousands, millions, and even billions of files. Now, the main concern is to store this huge number of files permanently. The storage device may be the hard disk drive, optical drive, SSD, etc. A storage device can be used in its entirety for a file system. A device also can be partitioned into different segments of the same or different sizes to hold files. Partitioning is useful for limiting the sizes of individual file systems, putting multiple file systems in the same device, or leaving part of the device available for other uses. An entity containing file systems is known as volume. A volume may be the subsets of a device. A volume allows a user to store multiple operating systems, boot them, and run them. Volume containing the information about the file system keeps the information about the file content. This information is kept in a device directory or volume table of contents. Information regarding a file system consists of – name, location, size, and type of the file.

9.10.1 Storage Structure

A computer system may have different types of file storage mediums. Each storage device will be sliced into several volumes to store files. A computer system may have zero or multiple numbers of files in it and the types of files can also vary. Some of the file systems in the Solaris system are:

tmpfs - temporary file system created in RAM. This type of file will be erased during reboot or system crash.

objfs – a virtual file system that gives debuggers access to kernel symbols.

ctfs – a virtual file system that maintains the contract information to manage the starting time of a process, booting time of the system, and must continue to run during operation.

lofs – a loopback file system that allows one file system to be accessed in place of another one.

procfs– a virtual file system that presents information on all processes as a file system.

ufs, zfs – general-purpose file system.

9.10.2 Directory Overview

A directory can be viewed as a table containing the file names. A file directory can be organized in many ways. In a directory, a user can create, view, search, and delete a file. The common operations that can be performed on a directory system are:

Search for a file – a directory is used to search for a required file. Since the directory keeps the names of all files, the user can match the required filename with the directory contents.

Create a file – user can create a new file and can add to directory.

Delete a file – Once the file is no longer required, it has to be removed from the directory. Users can delete unnecessary files from the directory.

List a directory – The contents of the directory should be able to view as a list of files.

Rename a file – Users are allowed to change the name of a file. Renaming a file can change the location of the file in the file directory.

Traverse the file system– Users may want to access each directory and every file within the directory structure. At regular interval, the contents and structure of the entire file system should be saved which makes the system more reliable. Often all the files

are copied and save into magnetic tape. In case of system failure, this technique delivers a backup copy. If a file is not used for a longer period, that may transfer permanently to tape such that the disk space can be utilized by another file.

9.10.3 Single Level Directory

The single-level directory is the simplest directory structure. Here all the files are located in the same directory, which is very easy to implement and understand. But it becomes inefficient if the number of users or numbers of files is increased. For more number of files, the file name should be unique. But if two different users kept the same name for their file, then this rule of uniqueness will be violated. If the numbers of files are very high, then remembering the file names for a single user also becomes very difficult. Keeping track of so many files is a daunting task.

9.10.4 Two-Level Directory

The drawback of a single-level directory structure for multiple users can be removed by introducing different directories for each user. Here each user will have their user file directory (UFD). Each user file directory consists of a set of files from one user only. When a user refers to a file, only his/her UFD will be searched. Thus different users can use the same file name, as long as the file names within the UFD are unique. To create a file the operating system searches only that user UFD to make sure that no other files with the same name exist in the directory. Similarly for deleting a file also, the operating system searches the file within the local UFD only. Thus the system prevents the deletion of files from another user's directory that has the same name. Figure 9.1 depicts a two-level directory structure.

The user's directory can be deleted by the users themselves. Though this two-level directory structure solves the problem of confliction of names, it still has disadvantages. Here all the users

will treat as an individual one and they cannot share their data among themselves. The isolation of users gives a platform to work independently. If the users want cooperation to perform some task and to access another's file, then this structure will not allow the users to do that. It is one of the disadvantages of the two-level directory.

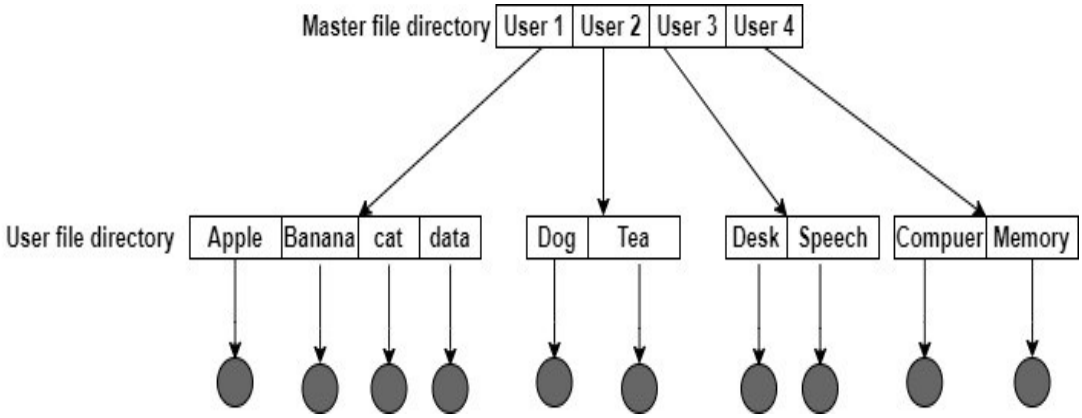


Figure 9.1: Two-level directory structure

9.10.5 Tree-Structured Directory

A tree is the most common directory structure. A tree has a root directory and each file has a unique pathname. A directory or a subdirectory contains a set of files or subdirectories as shown in figure 9.2. A directory is simply another file, but it is treated in another way. Each directory's internal structures are the same. One bit in each directory entry defines it as a file(0) or as a subdirectory(1). Generally, each process has a current directory. The current directory should contain most of the files that are of current interest to the process. When a file is requested, the current directory will be searched. If the required file is not available in the current directory, the user can search it in a different path or can change the current directory to another that holds the required file name. Thus a user can change the current directory according to their requirement or choice.

The file path can be categorized into two categories – *absolute* and *relative*. In the case of an absolute path, it starts from the root and goes down to the specified file. In relative path, it is defined from the current directory. A user can create his/her subdirectories in the system and thus can impose a file structure.

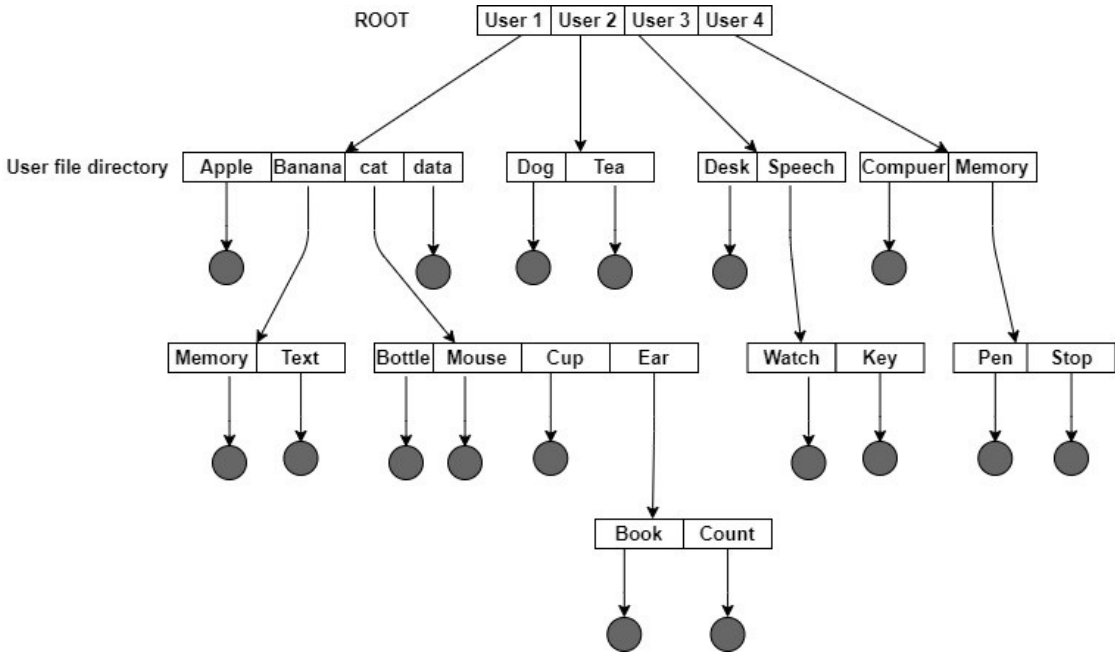


Figure 9.2: Tree-structured directory structures

To delete a directory from the tree structure directories can have two different approaches. If the directory is empty, then it can be deleted directly from the directory. If the directory is not empty or the directory has several files or subdirectories, then one of the two approaches can be followed. In some operating system, like MS-DOS does not allow deleting a non-empty file. To delete such a non-empty directory, first, we have to delete the files or the subdirectories. For subdirectories also, if it contains files or sub-sub directories, first delete all the internal files and sub-sub directories and then delete the subdirectories, and finally, we can delete the directory. Thus for sub directories, the process has to repeat recursively. This process can result in a substantial amount of work. An alternative approach is followed by the UNIX operating

system. By calling the ***rm –*** command, the complete directory including the files and sub-directories can be deleted. This approach is very easy to implement but at the same time, it is more dangerous also. If by mistake a user deletes a directory using the ***rm*** –command in UNIX the entire directory and files needs to be restored.

In the tree structure directory, one user is allowed to access the directories of others, i.e. the files of other users. The file path in the tree structure directory may be comparatively longer than the two-level directory structure.

9.10.6 Acyclic Graph Directories

If two users want to share the contents of a particular file from their directory, the tree structure does not permit sharing of file contents. For sharing a file, the common subdirectory should be shared. A shared directory or file will exist in the file system in more than one place simultaneously. Acyclic graph, i.e. a graph with no cycles allows directories to share subdirectories and files as shown in figure 9.3. Here one file can exist in two different places at the same time. Two copies of the file and the existence of a shared file in two different places are not the same. In the case of two copies of files in different places, the changes made in one file at one location do not reflect or appear in the others. But in the case of a shared directory, the changes made in one place are immediately reflected or appear in the other place also. Sharing is particularly important for subdirectories. If a user creates a new file in the shared sub-directory, immediately it will be visible in all the shared subdirectories for the other users.

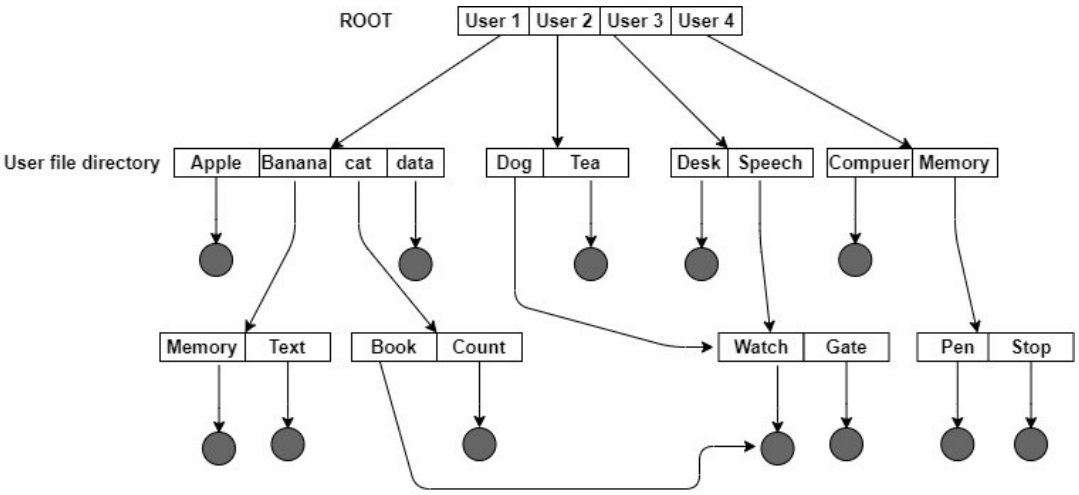


Figure 9.3: Acyclic graph directory structures

Acyclic-graph directory is more flexible than the tree structure directory. At the same time, it is more complex also. Here a file may have more than one absolute pathname. Consequently, distinct file names referred to the same file in the system.

The deletion of a file in the acyclic-graph directory may cause a dangling pointer. When a user deletes a file from the directory, may cause a dangling pointer for the non-existent files.



EXERCISE

5. What do you mean by directory?
6. Explain the operation that a user can perform on a directory.
7. What is the main drawback of single level directory over tree structured directory?
8. Differentiate between the tree structured directory and acyclic-graph directory.

9.11 FILE SYSTEM IMPLEMENTATION TECHNIQUES

File system implementation techniques give us the way to store the file and directories. It explains how to manage the disk space and make the whole process efficient and reliable. The following figure shows the layout

of a typical file system. Generally, the file systems are stored on a disk. In the figure 9.4:

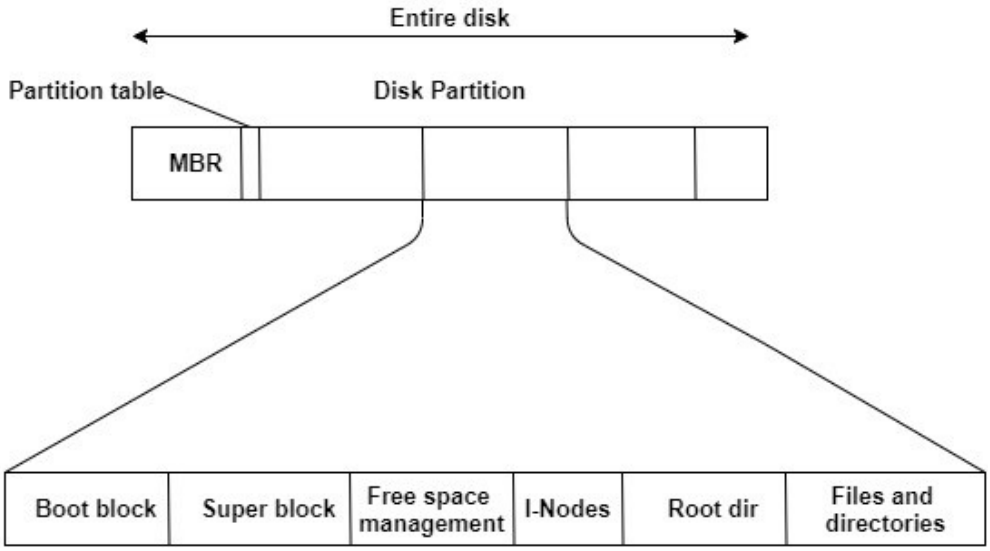


Figure 9.4: File system layouts

MBR: The contents of this portion are known as the master boot record which is used in the booting process.

Partition table: The MBR is followed by the partition table. It gives the starting and ending address of each of the partitions.

Boot block: during the booting of a computer, the BIOS reads in and executes the MBR. The operating system is loaded by the boot block. Every partition contains a boot block at the beginning though it does not have any operating system.

Super block: All the key parameters related to the file system are stored in the Superblock. When the computer is booted or the file system is accessed, all the key parameters are read into the memory.

9.11.1 Implementing Files

To implement the file structure, the memory has to allocate for the files or the directories. The memory allocation can be done in two different ways – contiguous allocation and linked list allocation.

9.11.2 Contiguous Allocations

In contiguous allocation, each file is stored in contiguous blocks in memory. Contiguous allocation is very easy to implement and the performance is also very good because the entire file can read from the disk in a single operation.

9.11.3 Linked List Allocation

In this approach, the files are stored in a disk as a linked list of blocks. The first word of each block is used as a pointer to the next one. The rest of the block is for data. Unlike contiguous allocation, no space is lost in disk fragmentation. The implementation of this process for random access is very slow.

- **Linked list allocation using a table in memory:**The demerits of the linked list allocation can be overcome by keeping all the pointer words from each bock in a separate table structure in the memory. Such a table in the memory is known as the FAT (File allocation table). Using FAT the random access can be made much easier. The main disadvantage of this technique is that all the time the FAT should have in the memory to make it work.
- **I-Nodes:** I-node is a data structure used to identify which blocks belong to which files. It contains the attributes and disk addresses of the file blocks. Unlike the in-memory table, the I-node needs to be in the memory only when the corresponding file is open.



EXERCISE

9. Explain the layout of a file structure?
10. What is the difference between contiguous allocation and linked list allocation?
11. What is the main disadvantage of the linked list allocation? And how to overcome from it ?



CHECK YOUR PROGRESS

Choose the correct answer from the given options:

Q1. A file is a sequence of?

i. Bits ii. Bytes

iii. Lines iv. All of the above

Q2. _____ is a sequence of bytes organized into blocks that are understandable by the machine.

i. Object file ii. source file

iii. text file iv. None of the above

Q3. What is true about Directory files?

i. These files represent physical device like disks, terminals, printers, networks, tape drive etc.

ii. These may have text, databases or executable program.

iii. These files contain list of file names and other information related to these files.

iv. All of the above

Q4. In Space Allocation, which of the following ways are correct to allocate disk space to files?

i. Contiguous Allocation ii. Linked Allocation

iii. Indexed Allocation iv. All of the above



9.12 LET US SUM UP

- The computer system has two different types of memories. Primary memories are used to store data or information temporarily and in secondary memory, data are stored permanently.
- Some of the secondary devices to store data are magnetic disk, drum, floppy, USB, CD-Disk, optical – disk, magnetic tapes, etc.
- The operating system abstracts the physical properties from the storage devices to define a logical storage unit, known as a ***file***.

- Files are stored permanently in secondary storage devices as they are non-volatile.
- A file can be named as per user convenience and is referred to by a filename.
- File attributes are name, size, type, location, protection, etc.
- Some of the common file types are: .doc, .pdf, .jpg, .c, .cpp, .xlxs, .mp4,. .mp3 etc.
- In sequential file access, information is processed sequentially, i.e. one after another.
- The direct access method allows reading and writing file records without following any particular order.
- A computer system may have different types of file storage mediums.
- The single-level directory is the simplest directory structure. Here all the files are located in the same directory.
- The file path can be categorized into two categories – *absolute* and *relative*.
- The deletion of a file in the acyclic-graph directory may cause a dangling pointer.



9.13 FURTHER READING

1. Galvin, P. B., Gagne, G., &Silberschatz, A. (2003). Operating system concepts (Vol. 10). John Wiley & Sons.
2. Deitel, H. M., Deitel, P. J., &Choffnes, D. R. (2004). Operating systems. Pearson/Prentice Hall.
3. Tanenbaum, A. (2009). *Modern operating systems*. Pearson Education, Inc.,.



9.14 ANSWERS TO THE CHECK YOUR PROGRESS

Ans to Q No 1: (iv) All of the above

Ans to Q No 2: (i) Object file

Ans to Q No 3: (ii)These may have text, databases or executable program.

Ans to Q No 4: (iv) All of the above



9.15 MODEL QUESTIONS

Q 1. Why do some systems keep track of the type of a file, while others leave it to the user and others simply do not implement multiple file types? Which system is better?

Q 2. Give an example of an application that could benefit from operating system support for random access to indexed files.

Q 3. Explain different file implementation techniques.

Q 4. Explain different file directory structures.

Q 5. Explain the operations that a user can perform on the file directory.

*** ***** ***

UNIT 10 : I/O MANAGEMENT

UNIT STRUCTURE

10.1 Learning Objectives

10.2 Introduction

10.3 I/O Hardware

- 10.3.1 I/O Devices
- 10.3.2 Device Controllers

10.4 Basic Principles of I/O Operations

- 10.4.1 I/O Operation Controlled By Program
- 10.4.2 I/O Operation Controlled By Interrupt
- 10.4.3 Direct Memory Access (DMA)

10.5 I/O Software

- 10.5.1 Objectives of the I/O Software

10.6 Device Driver

10.7 Interrupt and Interrupt Handlers

10.8 Let Us Sum Up

10.9 Further Reading

10.10 Answers to Check Your Progress

10.11 Model Questions

10.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- learn about basic principles of I/O operations
- learn about basic principles of I/O hardware
- learn about I/O devices
- know about device controllers
- know about the Direct Memory Access and its requirement
- learn principles of I/O software and its goals
- learn about interrupt handlers
- know about device drivers

10.2 INTRODUCTION

In the earlier units, we have already learnt about management of resources like CPU and computer memory by the operating system. File management and process management by operating system were also discussed. In this unit, we are going to discuss about Input/output management by operating system. Input/output is one of the very important functions of operating system. Input devices like keyboard, scanner, mouse, microphone etc. are used to input necessary information into the computer system. On the other hand output devices like monitor, printer, projector, sound card etc. are used to represent the result or output processed by the computer system.To improve the performance of a computer system, the input/output devices must be managed efficiently.The operating system is responsible for the management of I/O devices. In this unit, basic principles of I/O operations, I/O hardware and I/O software will be discussed. Device controller, interrupt handler and device driver will be explained. Direct Memory Access (DMA) and its requirement will be also discussed in this unit. In the next unit, we will explore the different issues related to protection.

10.3 I/O HARDWARE

The I/O hardware of computer system consists of I/O devices, device controllers and DMA controller. On the other hand,bus and port are important hardware used by I/O devices to perform their I/O operations. Port is used to connect the I/O devices to the computer system and bus is a collection of wires connected to the ports and device controllers. All signals for I/O instructions are sent through the bus.

10.3.1 I/O Devices

Input/Output (I/O) devices are the hardware used in the communication between computer system and users or other devices. The necessary information and instructions for these communications are sent or received through the I/O devices. The

responsibility of the operating system is to deliver a standard, efficient and reliable technique to access I/O devices.

In general, I/O devices can be categorized into four groups presented as follows:

1) Block devices
2) Character devices
3) Network devices
4) Devices that are neither block devices nor character devices

Block devices: The main characteristic of block devices is that these devices store data in fixed sized blocks.Data communication is performed in case of these devices by sending one or more complete blocks of data. Each block has its own unique address. In general, the block size of one block may be found in the range of 512 bytes to 32768 bytes. Each block can be read and written independently. Example of block device:Disks, Tap drive etc.

Character devices: The character devices do not have any block structure. These devices send or receive a stream of characters. One more important point regarding character devices is that these devices don't possess any seek operation as they are not addressable. Example of character devices:Printers, Keyboard etc.

Network Devices: Network devices are used in the communication of computer system over available computer networks. Network socket interface is available in these kinds of devices to transmit and receive data packets from remote locations or devices.Example of network I/O devices: Network adapters, Modems etc.

Devices that are neither block devices nor character devices: There are some I/O devices available which don't have the characteristics of both block and character devices.For example**:** clocks and timers don't have any block structure and they also don't transmit or receive any character stream. They are used by the operating system to implement interrupt mechanism. Operating

system can fix the CPU time period for a particular process using clocks and timers.

10.3.2 Device Controllers

The device controller is an electronic component which works as an interface between an I/O device and its device driver. In general, it is a printed circuit card. An I/O device is connected to the computer system through I/O bus with a plug and socket where the device controller of the device is connected with the socket. Through a device controller, the operating system manages or controls or communicates with the associated I/O device. The most important jobs of a device controller are to manage incoming and outgoing signals of the CPU and implement error correction in an I/O device. Multiple devices may be handled by a single device controller.

A device controller contains some local buffers termed as special purpose registers to store data temporarily that are received from the connected devices. Each device controller is associated with a compatible device driver. A device controller communicates its stored data to its associated device driver. The device driver is the software used by the device controller to communicate data with the operating system.

A device controller retains three types of device registers that are Status registers, Command registers and Data registers. These registers can be accessed by the CPU to perform I/O operations using special I/O commands or through memory mapping technique.

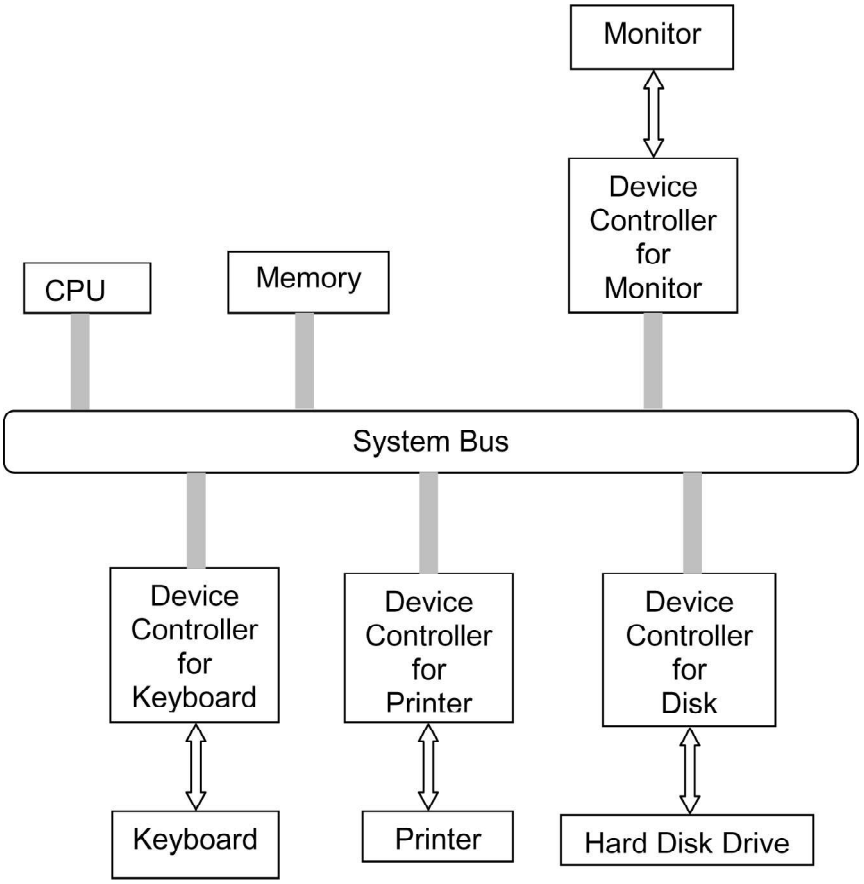


Figure 10.1: Representation of I/O Hardware Connected with the System Bus

10.4 BASIC PRINCIPLES OF I/O OPERATIONS

The I/O operations in a computer system are required in three different situations as mentioned below.

- When a user input necessary information to a computer system so that he/she can receive the desired result or output from the system.
- When necessary information comes as input from a particular device to the computer system so that the desired result can be processed by the system and it is received by the device.
- When computer systems are required to communicate over networks.

In the first situation, usually, a computer user input necessary information to a computer system by using a mouse or a keyboard. In case of both these input devices, only a few bytes of information are input to the

computer. But still in these communications, immediate response from the system is very necessary.As a result, in these kinds of I/O operations, an interrupt mechanismor a programmed data mode of operation is frequently required.

In the second situation, the communication between the computer and devices like disk and tapes is established. Data density in this type of communication is high. The data from these devices are transferred to the computer as blocks of information.Sometimes, in this type of data transfer, format control may be required to arrange the information so that it will be compatible to the device and data properties.The data rate of a particular device to deliver information may be significantly different from the data rate of the device which receives the information as input.In this scenario, the operating system is responsible for providing a uniform and simple I/O structure. In general, operating system performs this job by using and managing buffers.

In the third and final situation, a computer system requires I/O operations to communicate over the available computer networks.In this process, the network traffics must be managed and controlled by the existing network system.In this kind of communications, computer systems execute the I/O operations to access resources available in the connected networks. The internet resources are accessed by a system with these kinds of I/O operations so that the system can transfer files to other computers and support e-mail and web applications.Personal Digital Assistant (PDA) or mobile phones are also accessed with the help of infrared or wireless enabled communications. I/O operations based on Direct Memory Access (DMA) mechanism are provided in most network cards to communicate over the connected networks.The communication between a device and a computer system is implemented with the help of a data bus and a device controller. The communication of a computer over a network is different to the communication with the devices.

I/O operations can be implemented in three ways that are presented as follows:

- I/O operation controlled by program
- I/O operation controlled by Interrupt
- Direct Memory Access(DMA)

10.4.1 I/O Operation Controlled By Program

The I/O instructions in programs are responsible for the implementation of I/O operations in this approach. In general, data communication is performed between CPU registers and memory or peripheral devices with this way of I/O operations. In this case, CPU is also responsible for the communications between memory and I/O devices. I/O operations are performed by the CPU by maintaining the interaction with the device controller of an I/O device.In this process, the CPU is required to check the status register of the device repetitively so that when the device is available for data transfer then the CPU can initiate the I/O operations by executing instructions. Checking of the status register of a device constantly by CPU is termed as polling and it is the main problem of I/O operations controlled by programs. Due to polling, CPU time is wasted as the CPU is busy waiting unnecessarily till I/O devices become ready for communication.

10.4.2 I/O Operation Controlled By Interrupt

The solution to the problem associated with the I/O operations controlled by programs is the use of an interrupt mechanism so that CPU can be prevented from busy waiting. I/O operations can be controlled by an interrupt driven approach. In this approach, if I/O device is not available or ready for communication then the CPU is not required for busy waiting by constantly checking the status register of the device. When the I/O device is ready then interrupt can be initiated by the interrupt mechanism by sending an interrupt request signal to the CPU. After receiving the signal, CPU stops temporarily its current job and starts to execute the I/O operations for the I/O device. After completion of the I/O operation,

the CPU again starts performing its previous job from the point where it has stopped due to interrupt.

10.4.3 Direct Memory Access (DMA)

We have already discussed two approaches to implement I/O operations in the earlier sub-sections. In case of these approaches, it has been observed that CPU involvement is necessary to implement I/O operations. CPU must be required to execute several instructions for each I/O operations like Read and Write. So, in case of a computer system with large number of I/O operations, CPU will be unavailable to perform other important operations. As a result the system performance will be degraded. The solution of this problem is to use Direct Memory Access (DMA) technique to implement I/O operations.The Direct Memory Access (DMA) technique provides a mechanism to the computer system so that the CPU can be released from the implementation of I/O operations between I/O devices and memory. DMA allows data transfer directly to the memory without requiring constant involvement of the CPU.So, using DMA, computer systems is able to transfer data to and from devices with a little CPU overhead and it increases the speed of the execution of the read and write instructions.In this process, the CPU initiates a data transfer operation and then starts to perform other operations. At this moment, the DMA mechanism takes control of the data transfer operation and when data transfer is completed, an interrupt signal is sent by the DMA controller.DMA controller is the hardware part of the DMA mechanism. Multiple devices may be connected to the DMA controller. DMA controller is controlled by the CPU. Finally we can state that DMA mechanism provides concurrency between computation and data transfer. Two drawbacks are also associated with DMA. Firstly, the cost of the computer system increases with DMA mechanism as DMA controller is a hardware device. Secondly, due to DMA, cache coherency problems may occur in the system.

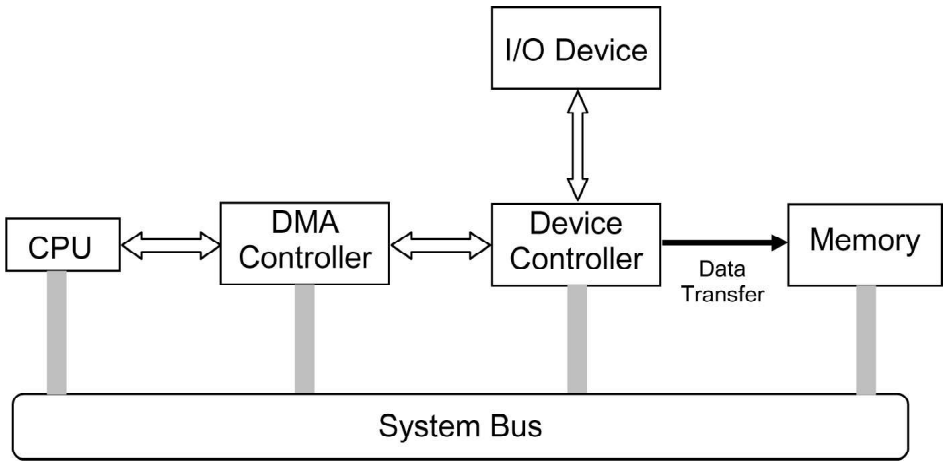


Figure 10.2: Representation of DMA Function

Working Principle of DMA

In case of the data transfer controlled and managed by the DMA facility of a computer system, a block of memory from one device is copied to another device without constant involvement of the CPU.In this process, at first,the CPU initiates an I/O operation by sending the following information to the DMA controller.

- What kind of instruction will be executed (Read or Write instruction)?
- The address of the I/O device for the I/O operation.
- Memory address from where data will be transferred or memory address where data will be written to.
- Number of bytes to be transferred.

After initiation of the I/O operation, CPU starts to perform other jobs. At the same moment, DMA controller starts to perform the data transfer and in this process the following steps are implemented.

- DMA facility starts control of the bus system from the CPU. Now, during the data transfer by the DMA if the CPU requires the bus system then there are two possibilities available. The first one is that the CPU must wait for the completion of the DMA job. The second is that the DMA may release the bus after transferring one word at a time and then again requests for the bus. As a result, the CPU can use the bus to perform its

necessary jobs. Then DMA again is able to start controlling the bus to transfer the next word.A third way to handle this problem is also available where the DMA is allowed to control the bus only when the CPU does not need it.

- DMA use the bus to transmit a read request to the device controller of the I/O device. The memory address where data will be transferred is also provided to the device controller.
- Then the device controller starts transferring data directly to the memory address provided by the DMA and acknowledges the DMA controller.
- After completion of the data transfer, the DMA sends an interrupt signal for the CPU to inform about the completion of the data transfer so that the CPU can take control of the bus from the DMA.



CHECK YOUR PROGRESS

Q1. Which of the following is not I/O hardware?

a) I/O device b) Device driver

c) Device controller d) None of the above

Q2. Which of the following is a block device?

a) Disk b) Printer

c) Clock d) None of the above

Q3. Which of the following is a character device?

a) Printer b) Keyboard

c) Clock d) Both a and b

Q4. _____ is an electronic component which works as an interface between an I/O device and its device driver.

a) Device driver b) Device controller

c) Interrupt d) None of the above

Q5. Constant involvement of CPU is not required in _____.

a) I/O operation controlled by program

b) I/O operation controlled by Interrupt

c) Direct Memory Access

d) None of the above

10.5 I/O SOFTWARE

We have already learnt about the I/O hardware in the earlier sections. In this section we are going to learn about the I/O software.Broadly, the I/O software can be divided into two parts where one part is device specific and other part is device independent. The device driver is the I/O software part which is device specific. It is used to control a specific type of I/O device. On the other hand, the device independent part of the I/O software is responsible for the following tasks.

1) The device independent I/O software provides identical interfaces between device drivers and other part of the operating system. Without identical interfaces, the operating system must be adjusted every time when a new device is added to the computer system.
2) The device independent I/O software provides a uniform naming structure for the I/O devices so that it can map a device name to its corresponding device driver. In UNIX, a unique i-node corresponding to a specific device name is available for a device file. Each i-node contains a major device number and a minor device number. Major device number is used to find the proper device driver. On the other hand minor device number is used as a parameter to the device driver to locate the particular I/O device.
3) Buffering is another important service that is provided by the device independent I/O software. Buffering refers to the operation to store data temporarily in a physical storage area for a small amount of time before transmitting it to the main memory or disk. This storage area is referred as buffer. Buffering is required to control the data flow between devices where the sender transmits data at speed different from the receiver's speed.Operating system may provide buffering in three ways that are explained as follows.

   Buffering using single buffer: Operating system provides one system buffer to store data temporarily when an I/O operation is requested. In this process, when the transfer of a complete block of data to the buffer is finished then the block of data is transferred

to the user space from the buffer. After completion of data transfer, the buffer will be available for the next block of data. The main drawback of this buffering approach is that the execution of user program may be blocked till the buffer becomes available after transferring a complete block of data.

Buffering using double buffer: The drawback of the single buffer system can be resolved by providing double buffer in the buffering operation.In this process, operating system provides two system buffers. When I/O operation is requested by a user program, data is transferred to the first buffer from I/O device. Then data from the first buffer is transferred to the second buffer from where the data will be finally transferred to the user space.So, in this approach, if the second buffer is busy with data transfer to the user space then at the same time, the first buffer is available for new data block.

Buffering using more than two buffers: Operating system may provide more than two buffers if large numbers of data blocks are continuously required to be transferred from I/O devices.The set of buffers assigned in this approach of buffering is termed as circular buffer. In this buffering, data is transferred to the first buffer from I/O device when I/O operation is requested. Then it is transferred from the first buffer to the second buffer and from second buffer to the third buffer. In this way, data is transmitted through each of the buffer and finally it is transferred from the last buffer to the user space.

4) Operating system is always responsible for handling errors that may occur in I/O operations. Operating system provides a device independent structure to handle I/O errors. Different types of I/O error may occur during I/O operations. Most of the I/O errors occurs due to the errors available in I/O devices. For example, I/O error will occur if some data is tried to be written in some physically damaged blocks of the disk. In such cases, the device driver is responsible to take care of the error. If the device driver is unable to handle it then the problem is forwarded to the device independent software. The

device independent software handles such an error depending upon the type and the current situation of the error. It may ignore the error or inform the user about the error. It may also terminate the I/O operation in worst case scenario.In some cases, I/O errors occur due to wrong instructions available in a program. For example, if any program try to read from an output device like printer. In such cases, the device independent structure reports an error code to the process.

5) The device independent software of operating system is responsible for the allocation and release of dedicated I/O devices.
6) The device independent software is also responsible for providing uniform block sizes for different disks by hiding the differences of the sector sizes in different disks.

A small part of the I/O software is also available separately from the operating system. This part is composed of the user programs, libraries and spooling system.The libraries are used to provide interface between user programs and the operating system or kernel by defining system calls.Spooling system is the mechanism to handle non-preemptable I/O devices like printer in multiprogramming environment. But spooling cannot be used to handle all types of non-preemptable I/O devices.

10.5.1 Objectives of the I/O Software

The I/O software are designed based on the following objectives.

- The I/O software should be device independent so that user programs are able to access I/O devices without requiring the details or specification of the devices.
- The I/O software should provide a uniform naming mechanism for I/O devices so that the name of the devices is independent of them. The name of a particular device may be a string or an integer number.
- The I/O software should provide an efficient buffering mechanism depending upon the number and types of the I/O requests available in a certain situation.

- The I/O software should provide interrupt driven data transfer so that the wastage of CPU time can be reduced. This type of data transfer is called asynchronous transfer. On the other hand, in some situation, synchronous transfer is more preferred than the asynchronous transfer. For example, in case of a read instruction available in a user program, the data transfer from I/O device to the buffer should be synchronous. So, operating system is responsible for selection of the type of data transfer for a particular I/O operation.
- We have already learnt in the earlier units that the preemptable I/O devices can be shared by multiple processes at the same time. But in case of non-preemptable I/O devices, it is not possible. Due to non-preemptable devices, different issues can occur in the system. For example: deadlock. The I/O software should take care of both preemptable and non-preemptable devices so that problems like deadlock can be avoided.
- If errors cannot be handled by the device controller of I/O devices then the I/O software should solve the problem.

10.6 DEVICE DRIVER

We have already learnt that each I/O device is connected with an electronic component termed as device controller. The structures of device controllers for different I/O devices may be different in terms of the number of their device registers.Similarly, the set of instructions associated with different I/O devices may also be different for each type of I/O devices. As a result, specific software is required to control a particular type of I/O device.This device specific part of the I/O software is termed as device driver.In general device driver is executed as a part of the operating system kernel. A particular device driver manages the device controllers of one kind of I/O devices or one group of devices which are closely related to each other. A proper device driver for a device is very necessary to perform its job efficiently. In general, device drivers are developed by the manufacturers of the devices.

The job of a device driver is to communicate with I/O device using computer bus system. It handles requests from the operating system kernel to access I/O device and to instruct the device for performing I/O operations**.** It works like an interface between the I/O device and the user program. When an I/O request comes from the user program, the device driver is responsible for checking all the relevant information regarding the requested I/O operation. If the requested I/O operation is found to be applicable then the device driver interacts with the device to perform the I/O operation.Device driver is also responsible for handling I/O errors if the errors cannot be handled at the hardware level.But if the device driver is unable to handle it then it is forwarded to the device independent I/O software**.** Due to device driver, installing a new device to a computer system becomes simpler.

In general, device drivers are categorized into four types depending upon the type of hardware devices which are controlled by them. These classifications of device drivers are discussed as follows.

1) **Block device drivers:** Block device drivers are the drivers which control block devices using buffer cache provided by the operating system. The kernel also provides standard interface for these drivers to perform required I/O operations. Block device drivers are utilized in case of the devices like Disks.
2) **Character device drivers:** Character device drivers are the drivers which control the character devices in which operating system can read or write data one byte at a time. Character device driver are useful in case of the devices like printer, keyboard etc. The device driver to manage terminal devices is also a character device driver.
3) **Network Device Driver:** Network device drivers control the network interface cards by managing the transmission and reception of network data packets over the network interface.
4) **Virtual Device Driver:** A virtual device is actually a form of software which can perform some operations just like a particular physical hardware device. Virtual device drivers are responsible for managing I/O operations by virtual devices.

10.7 INTERRUPT AND INTERRUPT HANDLERS

We have already discussed about I/O operation controlled by interrupt in an earlier section. In this section, we are going to discuss about interrupt and interrupt handlers.

Interrupt is actually a signal that is transmitted by I/O device or program to inform the CPU that an event has happened. For example, when an I/O device is ready for data transfer operation then it sends an interrupt signal to the CPU to inform about it so that the CPU can take necessary actions to initiate the operation. On the other hand, a user program may be suspended temporarily due to interrupt signals.

Interrupt handler is a set of instructions provided in the operating system to perform necessary operations after detection of an interrupt. Interrupt handler is also referred as Interrupt Service Routine (ISR). When an interrupt is detected then a particular interrupt handler is executed as a response to it. Interrupt handler is responsible for recognizing the reason of the interrupt and then performing necessary operations.

The device controller of I/O devices generates interrupt signal to perform I/O operations and it transmits the signal using system bus. The CPU can detect the interrupt signal through the wire that is referred as the interrupt request line.The CPU may possess two interrupt request lines which are termed as maskable interrupt line and nonmaskable interrupt line. The maskable interrupt line is used by I/O devices for transmitting the interrupt signals to the CPU. On the other hand, nonmaskable interrupt line is not used in case of transmitting normal interrupt signals. This interrupt request line is used only for transmitting special interrupt signals which cannot be postponned or ignored by the operating system.Now after detection of the interrupt, the CPU stops performing the current job and save the current states of that job. Then it starts executing the appropriate interrupt handler. The starting addresses of the ISR are stored in a table that is termed as interrupt vector table. A number from 0 to 255 is assigned to each of the possible interrupts. This number is used to access the starting address of a specific ISR from the interrupt vector table. After completion of

the necessary operations by the interrupt handler, the interrupt handler executes a return instruction so that the CPU can again start executing the earlier job from the saved position.But we have already learnt that in case of Direct Memory Access (DMA), after detection of interrupt signal, the CPU only initiates the I/O operation and then again starts performing its earlier job.

Source of interrupt signals can be hardware devices or may be software also. For example, hardware devices like keyboard, printer etc. generate interrupt signals for performing requested I/O operations. Interrupt generated from hardware devices are termed as hardware interrupts. In case of hardware interrupt, the devices are connected with the Interrupt Request Line. On the other hand interrupt signals initiated by programs are termed as software interrupts. A software interrupt is also referred as a trap or an exception.Software interrupts are handled by the operating system. Example of software interrupt: interrupt generated due to some errors that occur during execution of a program. For example,if any value is divided by zero in a program then software interrupt may be generated.

A computer system may be connected with many I/O devices. In that case, the computer system should be able to handle multiple interrupts and their interrupt handlers. We have already learnt that when an interrupt is sensed by the CPU then the starting address of the corresponding interrupt handler can be searched in the interrupt vector table. So in case of multiple interrupts, each of the interrupts is handled one by one by the operating system using the interrupt vector table. The operating system may also provide priority levels for the interrupts generated by different I/O devices to handle multiple interrupts. In this approach, higher priority interrupts are processed first by the CPU. It means that if a higher priority interrupt comes after a low priority interrupt then CPU postpone the low priority interrupt and start processing the higher priority interrupt. After handling the high priority interrupt, the CPU again start processing the low priority interrupt.



CHECK YOUR PROGRESS

Q6. Which of the following is an objective to design device independent I/O software?

a) Uniform naming structure to the I/O devices
b) Efficient buffering
c) Efficient allocation of dedicated devices
d) All of the above

Q7. How many buffers are used in case of circular buffer?

a) One b) Two
c) More than two d) None of the above

Q8. Which of the following is responsible for providing uniform block sizes for different disks?

a) Device independent I/O software
b) Device driver
c) User program
d) Both a and b

Q9. _____ communicates with I/O device using computer bus system.

a) Device driver
b) Device controller
c) Device independent I/O software
d) None of the above

Q10. Which type of device driver is used to handle a printer?

a) Block device driver
b) Character device driver
c) Network device driver
d) None of the above

Q11. _____is a set of instructions to perform necessary operations after detection of an interrupt.

a) Device driver b) DMA
c) Interrupt handler d) None of the above

Q12. The CPU can detect the interrupt signal through the wire that is called as _____.

a) Interrupt request line
b) Interrupt service routine
c) Interrupt handler
d) None of the above



10.8 LET US SUM UP

- The operating system is responsible for efficient management of Input output (I/O) devices.
- The I/O hardware of computer system consists of I/O devices, device controllers and DMA controller.
- In general, I/O devices can be divided into four groups that are Block devices, Character devices, Network devices and Devices that are neither block devices nor character devices.
- Block devices store data in fixed sized blocks.Character devices send or receive a stream of characters.Network devices are used in the communication of computer system over available computer networks.
- The device controller is an electronic component which works as an interface between an I/O device and its device driver. The Operating system manages or controls or communicates with an I/O device through its device controller.
- I/O operations can be implemented in three ways that are (a) I/O operation controlled by program, (b) I/O operation controlled by Interrupt and (c) Direct Memory Access (DMA).
- In case of I/O operation controlled by program, CPU is required to busy wait by constantly checking the status register of the device if the device is currently not available for I/O operation. This problem can be solved by using interrupt mechanism.

- The Direct Memory Access (DMA) technique provides a mechanism to allow data transfer directly to the memory without requiring constant involvement of the CPU.
- The I/O software can be divided into two parts where one part is device specific and other part is device independent.
- The device independent I/O software provides identical interfaces between device drivers, uniform naming structure for the I/O devices and uniform block sizes for different disks. It is also responsible for the allocation and release of dedicated I/O devices and buffering.
- A small part of the I/O software is placed separately from the operating system. This part is composed of the user programs, libraries and spooling system.
- Device driver is the device specific part of the I/O software. A particular device driver manages the device controllers of one kind of I/O devices or one group of devices which are closely related to each other. The job of a device driver is to communicate with I/O device using computer bus system.
- Normally, device drivers are divided into four types that are Block device drivers, Character device drivers, Network Device Drivers andVirtual Device Driver.
- Interrupt is a signal generated by I/O device or program to inform the CPU that an event has been happened.
- Interrupt handler is a set of instructions provided in the operating system to perform necessary operations after detection of an interrupt.The starting addresses of the interrupt handlers are stored in a table that is termed as interrupt vector table.
- The CPU can detect the interrupt signal through the wire that is referred as the Interrupt request line.
- Source of interrupt signals may be generated by both hardware devices and programs.
- The operating system can handle multiple interrupts using the interrupt vector table.

- Priority levels for I/O devices provided by the operating system can also be used to handle multiple interrupts. In this approach,higher priority interruptsare processed before the lower priority interrupts by the CPU.



10.9 FURTHER READING

1. Tanenbaum, A. (2009). *Modern operating systems*. Pearson Education, Inc.,.
2. Deitel, H. M., Deitel, P. J., &Choffnes, D. R. (2004). Operating systems. Pearson/Prentice Hall.
3. Jain, M. (2008). Operating Systems. BPB Publications
4. Stallings, W. (2008). Operating Systems: Internals and Design Principles. Pearson.
5. Galvin, P. B., Gagne, G., &Silberschatz, A. (2003). *Operating system concepts* (Vol. 10). John Wiley & Sons.



10.10 ANSWERS TO CHECK YOUR PROGRESS

Ans to Q No 1: b) Device driver

Ans to Q No 2 a) Disk

Ans to Q No 3: d) Both a and b

Ans to Q No 4: b) Device controller

Ans to Q No 5: c) Direct Memory Access

Ans to Q No 6: d) All of the above

Ans to Q No 7: c) More than two

Ans to Q No 8: a) Device independent I/O software

Ans to Q No 9: a) Device driver

Ans to Q No 10: b) Character device driver

Ans to Q No 11: c) Interrupt handler

Ans to Q No 12: a) Interrupt request line



10.11 MODEL QUESTIONS

Q 1. Explain the structure of I/O hardware in a computer system.

Q 2. Explain different objectives of device independent I/O software.

Q 3. Why DMA is required? Explain.

Q 4. What is device driver? Write down the jobs of a device driver.

Q 5. What is device controller? Write down the jobs of a device controller.

Q 6. What is interrupt and interrupt handler? Why interrupt is necessary in computer system?

*** ***** ***

UNIT 11: PROTECTION

UNIT STRUCTURE

11.1 Learning Objectives

11.2 Introduction

11.3 Needs for Protection

11.4 Principles of Protection

11.5 Domain of Protection

11.5.1 Domain Structure

11.6 Example in UNIX

11.7 Access Matrix

11.7.1 Domain Switching

11.7.2 Controlled Change Operations

11.8 Let Us Sum Up

11.9 Further Reading

11.10 Answers To Check Your Progress

11.11 Model Questions

11.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- learn about protecting the system from unauthorized users or programs.
- describe the protection mechanisms of operating systems
- learn about sharing of resources according to system policies set by system designersor by system administrators
- To explain how access matrix can be used to specify the resources a process may access.

11.2 INTRODUCTION

In the previous unit,we gained knowledge about operating system's I/O systems. We learned about the principles, complexities and performance aspects of I/O hardware and software. In this unit, we will understand about various protection mechanisms for the operating systems. We will learn about sharing resources according to system policies set by system designers or by

system administrators and see how to enable the user so that the least damage is caused. In the next unit, we will explore different security related issues.

11.3 NEEDS FOR PROTECTION

Computers have penetrated every nook and corner of our lives. It is used in almost all offices, organizations and companies; it has become public as well as a private commodity. With the exponential use of computers, the attacks on computer systems, their components, resources and operating systems have also become a real threat. One of the major threats is unauthorized access to system or system resources. There should be a predefined set of policies regarding the usage of the system and its resources. The policies can be fixed in the system itself by the designer of the operating system, or they can be set by the administrator of the system. Even single users of the system / resources can have some basic protection by setting their own policies.The applications that run on the operating system should also have their own set of policies to guard the system and its resources. Another method of employing strong protection is to have a process in place that detects and identifies errors or faults that may remain hidden or stay silently inside the system or in the applications that are installed in the operating system. If such dormant errors are not detected then a bad actor can exploit this frailty, making the system completely unusable.

11.4 PRINCIPLES OF PROTECTION

One of the best methods of providing protection is called as the *principle of least privilege*. It is based on the principle that each user or program should be given just the access, usage or execution capacity that is required to perform the designated task. If an operating system follows the principle of least privilege in its programs, system calls, and data structures, then in the event of a program or any of its components being compromised, the extent of damage will be limited to that component only. For example, a buffer overflow may result in the failure of the daemon process, but if the operating system implements the principle of least

privilege, then failure will be bounded within the daemon process itself. Otherwise, someone can exploit the problem of buffer overflow, and cause the failure to propagate to other parts of the system and somehow gain access privilege to the entire system. The operating system should provide system calls and services that allow applications to write in-depth and well-defined access controls. It should have mechanisms to enable or disable privileges as and when required. Another method is the use of audit trails so that the activities on the system can be tracked back to the person who accessed the system.

11.5 DOMAIN OF PROTECTION

A computer system has hardware and software objects. Hardware objects consist of items such as the CPU, memory segments, printers, disks, and tape drives. Software objects consist of files, programs, semaphores, etc. Objects are represented by abstract data types. Each object has a unique identifying name and can be accessed only through well-defined and meaningful operations. Various operations can be performed on the object,for example, on a CPU, we can only execute. ACD-ROM or DVD-ROM can only be read whereas memory segments can be read and written.When it comes to data files, it can be created, opened, read, written, closed, and deleted; whereas program files can be read, written, executed, and deleted. For proper handling of our resources, we should allow a process to access only those resources for which it has authorization and provide only those resources that it currently requires. This is known as the *need-to-know* principle, and like the *principle of least privilege*, it also restricts the damage from propagating to other parts of the system.If a process p invokes procedure A(), then the procedure should be allowed to access only its own variables and the formal parameters passed to it; it should not be able to access all the variables of process p. Similarly, the process p too should be allowed to access only the resources of procedure A() that it currently requires to execute, and should be restricted to access any other files or processes that the procedure internally uses to perform its own work.

11.5.1 Domain Structure

To ensure protection, a *protection domain* is maintained, which tells which resources are assigned to which processes. It also specifies the operations(like read, write, delete, execute, etc.) that are allowed on each object. When a process is able to perform an operation on an object, then it is said to have the *access right*. A collection of such access rights *d <object-name, rights-set>*, is called as a *domain*. Consider the following example where there three domains D_1, D_2 and D_3 are depicted in Figure 11.1:

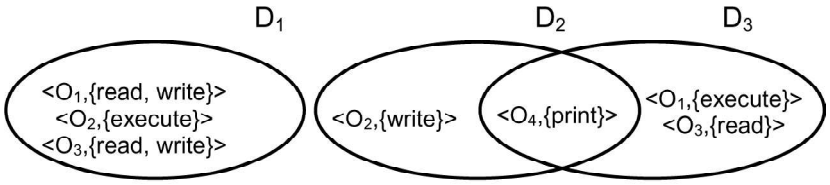


Figure 11.1: Protection domain D_1, D_2 and D_3 with their access rights

Here O_1, O_2 and O_3 can be objects like files, programs, memory segments, printers, etc. If a process is executing in domain D_1 then the operations allowed on object O_1 are only read and write. If a process wants to execute object O_1 then it has to be in domain D_3.

Another scenario we see here is sharing of access rights between D_2 and D_3, which means that a process executing in either of D_2 or D_3 can print the object O_4.

The way processes are assigned to domain can be either static or dynamic. In case of static association a process is associated to a particular domain and the set of resources available to the process will be fixed throughout the process's lifetime. Here, we have to ensure that the need-to-know principle is upheld. For example, in the first phase of execution, say a process p needs to read a file object F_1, in the second phase it needs to obtain permission to write in $F_1$and finally in the third phase it needs to execute F_1. Now if the association between process and domain is static,

meaning we cannot change it later, then we might make an error of including all the read, write and execute operations in the same domain and assign it to process p. This will infringe our notion of need-to-know principle, granting the process all the access rights in the same domain.

If case of dynamic association, a processis not only able to switch from one domain to another but the content of a domain can also be changed. If there are certain constraints that do not allow changing of the domain content, then we can easily create a new domain, include the required operations and switch to the new domain.



EXERCISE

1. Give some basic steps to protect the system and its components.
2. Give an example how the principle of least privilege can mitigate the extent of damage on a system.
3. Why do you think the operating system should provide system calls and services that allow applications to write in-depth and well-defined access controls?
4. What are the hardware and software objects of a computer system?
5. Why do you think we should allow a process to access only those resources for which it has authorization and provide only those resources that it currently requires?
6. If case of dynamic association, if there are certain constraints that do not allow changing of domain content then what actions can be taken?



CHECK YOUR PROGRESS

Q1. How to protect a system and its resources from unauthorized access?

Q2. What is the principle of least privilege?

Q3. Give example of computer objects and the operations that can be performed.

Q4. What is the need-to-know principle and why it can mitigate the extent of damage on a system?

Q5. A collection of access rights d <object-name, rights-set>, is called as a _____.

Q6. Consider two domains D_1 and D_2. D_1 has the access right <file F_1,{read,write}> and <file F_2,{read}>; D_2 has the access right <file F_1,{read,execute}> and <file F_2,{print}>. Answer True or False for the following statements:

i) A process executing in D_1 can read file F_1 and file F_2 both.

ii) A process executing in D_1 can write file F_1 and file F_2 both.

iii) A process executing in D_2 can execute file F_1 and file F_2 both.

iv) A process executing in D_2 can execute file F_1 and print file F_2.

11.6 EXAMPLE IN UNIX

In UNIX operating system, each file has owner identification and a domain bit, called as *setuid bit*. The setuid bit can be set *on* or *off* by the owner of the file. If the setuid bit is *off* then, whenever a user (say A) executes the file, the user ID of the process is also A. In this case, the user A can run the file with the limited privileges that has already been designated to A. But there might be some cases where a normal user might require special permission or privilege to perform a certain task. Take for example the need to change our login password. In UNIX,when a user wants to change their password, they run the *passwd* command. The *passwd* program does not belong to a normal user, but to the *root* account. So, if a user wants to change the password, it is not possible unless the user has the permission to do so. In such cases, the setuid bit of the *passwd* program is set *on* so

that the user is temporarily granted root access for that limited purpose. Thus, when the setuid bit is set *on*, the user ID of the process is not set to the user ID of the normal user (say A) but it remains that of the owner of the file (say *root*), granting the user, all the privileges of the owner of the file. But we should be careful while setting the setuid bit *on*, because if an unauthorized person has full privilege to the system then it might open door for possible security attacks. Then the question is why would we want to set the setuid bit *on* when we know the underlying security risk? The answer is, sometimes it might be necessary to allow trusted / authorized users or trusted programs (like the *ping* or *passwd* command) to execute the file with full privilege so that the required task can be completed.

There is another problem with *setuid* programs where hackers create and hide their programs in obscure file or directory names and use it or activate it when required. To avoid this threat, there is another alternative where we place our privileged or important files and programs in a special directory, such that only those who can access the special directory, can execute the privileged files or programs. When the owner of the file or any authorized user enters the special directory, the user ID is set to the appropriate user ID of the owner and he or she can perform the desired operation. But if a hacker or an unauthorized person tries to execute, modify or delete the privilege files or programs, he or she will not be able to do so because they do not have access to the special directory and the user ID will not be set to the user ID of the file owner.



EXERCISE

7. In UNIX operating system what kind of privilege is granted to the user if the *setuid* bit of a program is *on* or *off*.
8. If a special directory is maintained and all the privileged programs are kept in the special directory with privilege access, then can an unauthorized user still execute the privilege program? Why? (Hint: User ID)

11.7 ACCESS MATRIX

The model of protection can be defined using a two-dimensional matrix which is called an access matrix. The access matrix is divided into rows and columns, where the domains are placed in the rows whereas the objects are placed in the column sections; and each entry in the matrix contains the set of access rights. For clarification look in the access matrix as shown in Figure 11.2. As seen in the example, each *entry (i,j)* inside the matrix table, defines the set of operations that a process executing in domain D_i can invoke on object O_j. Here '*i*' is the row and represents the domain and '*j*' is the column and represents the object.

Object → / Domain ↓	F_1	F_2	F_3	Printer
D_1	read		read	
D_2				print
D_3		read	execute	
D_4	read write		read write	

Figure 11.2: Example of access matrix

If we look at the access matrix table given in Figure 11.2 we will find that there are four domains – D_1, D_2, D_3 and D_4 and four objects – F_1, F_2, F_3 which are files and the last object is a printer. A process executing in domain D_1 can only read files F_1 and F_3 whereas a process executing in domain D_4 can not only read files F_1 and F_3, it also has the access right to write onto files F_1 and F_3. Similarly, only a process executing in D_2 can operate the printer. The operating system, application programs and the system administrator must strictly adhere to the policies stated in the access-matrix table, such that a process executing in domain D_i can access only those objects specified in row '*i*', as allowed by the access-matrix entries.

11.7.1 Domain Switching

When the processes have to change the association between the domains, a *switch* operation is required to be executed between the processes and the domains. To perform domain

switching, the domains are included as objects on the column side of the access matrix table. If we consider the example given in Figure 11.3, a process executing in domain D_2 can switch to domain D_3 or domain D_4. Similarly, a process in domain D_1 can switch to D_2, and a process in domain D_4 can switch to D_1.

Object → / Domain ↓	F_1	F_2	F_3	Printer	D_1	D_2	D_3	D_4
D_1	read		read			switch		
D_2				print			switch	switch
D_3		read	execute					
D_4	read write		read write		switch			

Figure 11.3: Access matrix with domain switching operation

11.7.2 Controlled Change Operations

To allow changes in the access matrix in a controlled manner, there are three main operations– *copy*, *owner* and *control*. First let us discuss how the *copy* operation is performed. If an access right can be copied from one domain in the access matrix table to another domain in the access matrix table, then that operation is marked by an asterisk (*) symbol. The copy of access rights can be done only within the column, i.e., within the same object. More specifically, an access right from the entry *access(i, j)* can be copied only to *access(k, j)* and not to *access(i,k)*. For example, if we notice in Figure 11.4 and Figure 11.5, the access rights *read** for file F_2 and the access right *write** for file F_3 can be copied only within the column itself, i.e., within the respective files.

Object → / Domain ↓	F_1	F_2	F_3
D_1	read		write*
D_2		read*	execute
D_3	write	execute	
D_4	read		

Figure 11.4: Access matrix with copy (*) operation

Object → / Domain ↓	F_1	F_2	F_3
D_1	read		
D_2		read*	execute
D_3	write	execute	
D_4	read	read	write*

Figure 11.5: Access matrix after copy (*) operation

As seen in Figure 11.4 and Figure 11.5, the *read* operation for file F_2 has been copied from domain D_2 to domain D_4. One thing we must observe here is that the *read* operation that is copied from domain D_2 to domain D_4 does not have the asterisk (*) symbol. This means that the *read* operation in domain D_4 can no longer be copied further. This is called a *limited copy* where the access right, after being copied, cannot be propagated further. Also, if we see the *write** operation in Figure 11.4 is copied entirely from domain D_1 to domain D_4 in Figure 11.5. This action is called the *transfer* of a right from one domain to another.

The second change operation that can be used is the *owner*, which gives us the mechanism to add or delete access rights from any domain. Here, if *access(i, j)* includes the owner right, then a process executing in domain D_i can add or remove any right in any entry in column *j*. For example, in Figure 11.6, domain D_1 is the owner of F_1 and domain D_2 is the owner of F_2 and F_3. Thus they can add and remove any valid rights within the respective columns. Thus, in Figure 11.7, domain D_1 being the owner of file F_1has removed the operation *execute* from domain D_3, whereas it has added the operation *write* to domain D2. Also, since D_2 is the owner of files F_2 and F_3, it has the ability to add and delete some access rights from some domains, as can be seen in Figure 11.7.

Object → / Domain ↓	F_1	F_2	F_3
D_1	owner execute		write
D_2		read* owner	read* owner write
D_3	execute		

Figure 11.6: Access matrix withowner right

Object → / Domain ↓	F_1	F_2	F_3
D_1	owner execute		
D_2	write	read* write* owner	read* owner write
D_3		write	write

Figure 11.7: Adding and deleting the access rights by the owner

The last access control mechanism is the *control* right which can be applied only to domain objects and is used to change the access rights along the rows. If *access(i, j)* includes the *control* right, then a process executing in domain D_i can remove any access right from row *j*. For example, consider the access matrix in Figure 11.8 and Figure 11.9.

Object → / Domain ↓	F_1	F_2	Printer	D_1	D_2	D_3
D_1	read				switch	
D_2			print			switch control
D_3	execute read	read write				

Figure 11.8: Access matrix with control right

Here, domain D_2 has the *control* right over D_3. So, it can remove some access right from domain D_3, as seen in Figure 11.9.

Object → / Domain ↓	F_1	F_2	Printer	D_1	D_2	D_3
D_1	read				switch	
D_2			print			switch control
D_3	execute	read				

Figure 11.9: Access matrix after the removal of access rights



EXERCISE

9. The access matrix is a three dimensional matrix. (True or False)
10. The access matrix has row, column and height. (True or False)
11. The column represents the domain and the row represents the object. (True or False)
12. Each *entry (i,j)* inside the access matrix, defines the set of operations that a process executing in domain *Di* can invoke on object *Oj*. (True or False)
13. A *switch* operation is required when the processes have to change the association between the domains. (True or False)
14. To perform domain switching, _____are included as objects on the _____ of the access matrix table.
15. What are the three main operations with the help of which we can change the access rights in the access matrix in a controlled manner?
16. Control right can be applied only to _____ and is used to change the access rights along the _____.



CHECK YOUR PROGRESS

Q7. Let us consider the two access matrix given below follow the limited copy mechanism. The first matrix is the original matrix and the second matrix is the access matrix after the limited copy operation has been performed. Find the error in the second matrix.

Object Domain ↓ →	F_1	F_2
D_1	read*	
D_2		write*

Object Domain ↓ →	F_1	F_2
D_1	read*	write*
D_2	read	write*

Q8. What is an access matrix and what is the provision for protection using the access matrix?

Q9. How is it possible for a process to make a shift from one domain to another?

Q10. What does it mean if an operation in the access matrix is marked by an asterisk (*) symbol?

Q11. Explain the control access mechanism in the access matrix.



11.8 LET US SUM UP

- One of the major threats is unauthorized access to system or system resources.
- There should be a predefined set of policies regarding the usage of the system and its resources.

- A method of employing strong protection is to have a process in place that detects and identifies errors or faults that may remain hidden or stay silently inside the system or in the applications that are installed in the operating system.
- The principle of least privilege is based on the principle that each user or program should be given only just the access, usage or execution capacity that is required to perform the designated task, and not more.
- The principle of least privilege is based on the principle that we should allow a process to access only those resources for which it has authorization and provide only those resources that it currently requires.
- Access right is the ability of a process to perform an operation on an object.
- A collection of such access rights e <object-name, rights-set>, is called as a domain.
- In UNIX operating system, each file has owner identification and a domain bit, called as setuid bit which can be set on or off by the owner of the file.
- We should be careful while setting the setuid bit on, because if an unauthorized person has full privilege to the system then it might compromise the system.
- The setuid bit is sometimes set on because it might be necessary to allow trusted/authorized users or trusted programs (like the ping or passwd command) to execute the file with full privilege so that the required task can be completed.
- The model of protection can be defined using a two-dimensional matrix which is called an access matrix, where the rows represent the domains, the columns represent the objects and each entry in the matrix contains the set of access rights.
- To allow changes in the access matrix in a controlled manner, there are three main operations – copy, owner and control.



11.9 FURTHER READING

1. Stallings, W. (2008). Operating Systems: Internals and Design Principles. Pearson.
2. Deitel, H. M., Deitel, P. J., & Choffnes, D. R. (2004). Operating systems. Pearson/Prentice Hall.



11.10 ANSWERS TO CHECK YOUR PROGRESS

Ans to Q No 1: To protect a system and its resources from unauthorized access, there should be a predefined set of policies regarding usage of system and its resources. The policies can be fixed in the system itself by the designer of the operating system, or it can be set by the administrator of the system. Even single users of the system / resources can have some basic protection by setting their own policies. The applications that run on the operating system should also have their own set of policies to guard the system and its resources. Another method of employing strong protection is to have a process in place that detects and identifies errors or faults that may remain hidden or stay silently inside the system or in the applications that are installed in the operating system. If such dormant errors are not detected then a bad actor can exploit this frailty, making the complete system unusable.

Ans to Q No 2: It is the principle in which each user or program is given only just the access, usage or execution capacity that is required to perform the designated task, and not more. If an operating system follows the principle of least privilege in its programs, system calls, and data structures, then in the event of a program or any of its components being compromised, the extent of damage will be limited to that component only.

Ans to Q No 3: Example of hardware objects: CPU, memory segments, printers, disks, and tape drives. Example of software objects: files, programs, semaphore, etc. Example of operations that can be performed on the object: on a CPU, we can only execute, on a CD-ROM or DVD-ROM we can only be read whereas memory segments can be read and written. When it comes to data files, it can be created, opened, read, written, closed, and deleted; whereas program files can be read, written, executed, and deleted.

Ans to Q No 4: The need-to-know principle is the principle where we allow a process to access only those resources for which it has authorization and provides only those resources that it currently requires. This will help in stopping the damage to spread to other components of the system. For example, if a process p invokes procedure A(), then the procedure should be allowed to access only its own variables and the formal parameters passed to it; it should not be able to access all the variables of process p. Similarly, the process p too should be allowed to access only the resources of procedure A() that it currently requires to execute, and should be restricted to access any other files or processes that the procedure internally uses to perform its own work.

Ans to Q No 5: A collection of access rights d<object-name, rights-set>, is called as a domain.

Ans to Q No 6:

i) True

ii) False

iii) False

iv) True

Ans to Q No 7:The model of protection can be defined using a two-dimensional matrix which is called an access matrix. The access matrix is divided into rows and columns, where the domains are placed in the rows whereas the objects are placed in the column sections; and each entry in the matrix contains the set of access rights. Each *entry (i,j)* inside the matrix table, defines the set of

operations that a process executing in domain D_i can invoke on object O_j.Here '*i*' is the row and represents the domain and '*j*' is the column and represents the object.

The operating system, application programs and the system administrator must strictly adhere to the policies stated in the access-matrix table, such that a process executing in domain Di can access only those objects specified in row 'i', as allowed by the access-matrix entries.

Ans to Q No. 8: When the processes have to change the association between the domains, a *switch* operation is required to be executed between the processes and the domains. To perform domain switching, the domains are included as objects on the column side of the access matrix table.

Ans to Q No. 9: If an operation in the access matrix is marked by an asterisk (*) symbol then that access right can be copied from one domain to another domain in the access matrix table.

Ans to Q No. 10: In a limited copy operation, the access right, after being copied, cannot be propagated further, i.e., the copied access right cannot again have the asterisk(*) symbol. So, in the second matrix after the write operation has been copied from domain D_2 to domain D_1, it cannot again have the asterisk(*) symbol.

Ans to Q No. 11: The *control* right can be applied only to domain objects and is used to change the access rights along the rows. If *access (i, j)* includes the control right, then a process executing in domain D_i can remove any access right from row '*j*'.



11.11 MODEL QUESTIONS

Q1. Explain the principle of least privilege and the principle of need-to-know with proper examples.

Q2. With the help of a proper example explain the protection domain structure.

Q3. Explain what *setuid* is and how it can be used to elevate the privilege for normal users. What are the pros and cons of doing it?

Q4. Explain all the change operations that are allowed in the access matrix with suitable examples.

Q5. Explain the copy operation that is allowed in the access matrix along with the different types of copy operations.

Q6. Explain the difference between the two copy operations: limited copy and transfer copy?

Q7. Explain the purpose of owner access right in the access matrix.

*** ***** ***

UNIT 12: SECURITY

UNIT STRUCTURE

12.1 Learning Objectives

12.2 Introduction

12.3 The Security Problem

12.4 Program Threats

- 12.4.1 Trojan Horse
- 12.4.2 Trap Door
- 12.4.3 Logic Bomb
- 12.4.4 Stack and Buffer Overflow
- 12.4.5 Viruses

12.5 System and Network Threats

- 12.5.1 Worms
- 12.5.2 Port Scanning
- 12.5.3 Denial of Service

12.6 User Authentication

12.7 Password

12.8 Password Vulnerabilities

12.9 Encrypted Password

12.10 One Time Password (OTP)

12.11 Biometrics

12.12 Intrusion Detection

12.13 Let Us Sum Up

12.14 Further Reading

12.15 Answers To Check Your Progress

12.16 Model Questions

12.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- know about the various security threats and attacks.
- elaborate the different kinds of program threats that can harm a computer system.

- understand the various threats that can be used to abuse the services of network connections.
- describe various counter measures to security attacks

12.2 INTRODUCTION

In the previous unit, we learned how unauthorized users can harm the system and the different protection and resource sharing mechanisms. We also learned how to control the system access using an access matrix.In this unit, we will know about the attacks which are intended by individuals to steal information, damage information, or cause any kind of destruction; and measures to protect the system from such attacks. In the next unit, we will explore the topic of multiprocessor systems in detail.

12.3 THE SECURITY PROBLEM

It takes significant effort and hard work to provide and ensure the security of the computer system. Systems that contain sensitive company information like payroll, intellectual property, trade secrets, investments, clients information, or other important financial data may become the target of unscrupulous and clandestine criminal organizations. Any theft or loss of such data, by accident or by fraud, can cause huge losses to a company. In the previous chapter, we learned about the methods that the operating system can provide to protect the system. But it works only if the user follows the instructions and utilizes the resources as intended. So, we need security mechanisms that make it even harder for unscrupulous and unauthorized users to breach the systems. The following list includes several forms of accidental and malicious security violations:

- **Breach of confidentiality:** When an unauthorized person can read (or steal) sensitive and confidential data, then it is called a breach of confidentiality. It is one of the main aim of an intruder.
- **Breach of integrity:** When an attacker does the unauthorized modification of data then it is called a breach of integrity because the integrity of data has been violated.

- **Breach of availability:** The aim of some attackers is not to withhold the data for ransom, but to simply destroy the data so that no one can access it. This is called a breach of availability. Web-site defacement is a common example of this type of security breach.
- **Theft of service:** When an intruder has unauthorized use of resources, it is called theft of service.
- **Denial of service:** Denial of service is typically accomplished by flooding the targeted machine or resource with redundant requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled. This violation involves preventing legitimate use of the system.

To breach the security of a system, the intruder many times pretend to be someone else. This is called as *masquerading*, and by doing so, the intruder can breach the authentication method of a system and gain special access and privilege to the system. Another kind of attack is called a *replay attack*, in which an attacker eavesdrops on messages relayed through network communication, fraudulently captures it, and then resends (replays) the message with unauthorized modifications to misdirect the receiver into doing what the hacker wants. There is another kind of attack called the *man-in-the-middle attack*, in which an attacker intercepts the data flow between a legitimate sender and a receiver, then starts masquerading (pretending) as both sender and receiver.

In the next section, we discuss the different types of security threats that exist and the ways to guard our system.



EXERCISE

1. Explain the difference between breach of confidentiality and breach of integrity.
2. Explain the difference between breach of availability and denial of service.
3. How *masquerading* is done in *man-in-the-middle* attack?

12.4 PROGRAM THREATS

Computer programs are used to perform various work on the computer. So, it is natural for hackers and attackers to create and write malicious programs to harm the system or steal important data from the system. Here we describe some common methods by which programs cause security breaches in a system.

12.4.1 Trojan Horse

A *Trojan horse* is a malicious program that looks legitimate or harmless or very enticing, but its actual intent may be to damage, disrupt, or steal the data. An example of the Trojan horse is an attachment that is received in an e-mailor on some advertisement on social media that seems to be from an authentic source,but on clicking the attachment a malicious code is executed which spreads to other files and damages the computer.*Spyware* is another kind of Trojan horse, which is automatically installed when we install another software. Sometimes, we think we are installing freeware, shareware, or any commercial software, but unknown to us, the original software also installs spyware, which might be used to display ads on the user's computer or it might even capture user's information and send it back to the parent site.

12.4.2 Trap Door

Sometimes the programmer might leave a *trap door* or a secret entry point in software that lets him surpass the security procedure and gain access to the system. Programmers sometimes leave a trap door for genuine testing and debugging purpose, or to give service-technicians emergency access. But it might become a grievous threat if an intruder or a dishonest programmer uses it to gain unhindered access to the entire system. Another back-door program is a *keystroke logger*, which records everything entered on the keyboard (including passwords and credit card numbers).

12.4.3 Logic Bomb

A *logic bomb* is a malicious code that is intentionally inserted by a dishonest programmer and gets activated when certain conditions, as set by the programmer, are met. For example, a programmer might write a code to detect whether he or she was still employed; and if that check failed, the code could be executed to cause damage to the system.

12.4.4 Stack and Buffer Overflow

The *stack-buffer-overflow* attack is one of the most common ways to attack and gain unauthorized access to the target system. The attacker exploits a bug in a program that could be the result of poor programming, like when the programmer neglects to bound check on an input field. An attacker can run a malicious program that writes more data to a buffer located on the stack than what is allocated for that buffer. This almost always results in corruption of adjacent data on the stack, and will often cause the program to crash or operate incorrectly. If the affected program is running with special privileges or accepts data from untrusted network hosts (e.g. a webserver) then the bug is a potential security vulnerability. If the stack buffer is filled with data supplied from an untrusted user then that user can corrupt the stack in such a way as to inject executable code into the running program and take control of the process.

12.4.5 Viruses

A *virus* is a piece of vicious code that is hidden in a program or a file that might seem legitimate and genuine. Once a virus reaches a target machine, a Trojan horse known as a *virus dropper* inserts the virus into the system. Once the virus is installed and activated, it replicates itself multiple times and wreaks havoc by damaging files and causing system crashes and malfunctions. Viruses usually spread through the internet via e-mail, or when users

download viral programs or files from the internet file-sharing services or exchange infected files/disks.Though there are thousands of viruses, they broadly fall into several categories as given below:

- ***File*:** A standard file virus infects a system by appending itself to a file. It changes the start of the program so that execution jumps to its code. After it executes, it returns control to the program so that its execution is not noticed. File viruses are sometimes known as parasitic viruses, as they leave no full files behind and leave the host program still functional.
- ***Boot*:** A boot virus infects the boot sector of the system, executing every time the system is booted and before the operating system is loaded. It watches for other bootable media (that is, floppy disks) and infects them. These viruses are also known as memory viruses, because they do not appear in the file system.
- ***Macro*:** Most viruses are written in a low-level language, such as assembly or C. Macro viruses are written in a high-level language, such as Visual Basic. These viruses are triggered when a program capable of executing the macro is run. For example, a macro virus could be contained in a spreadsheet file.
- ***Source code*:** A source code virus looks for source code and modifies it to include the virus and to help spread the virus.
- ***Polymorphic*:** A polymorphic virus changes each time it is installed to avoid detection by antivirus software. The changes do not affect the virus's functionality but rather change the virus's signature. A virus signature is a pattern that can be used to identify a virus, typically a series of bytes that make up the virus code.
- ***Encrypted*:** An encrypted virus includes decryption code along with the encrypted virus, again to avoid detection. The virus first decrypts and then executes.

- ***Stealth*:** This tricky virus attempts to avoid detection by modifying parts of the system that could be used to detect it. For example, it could modify the read system call so that if the file it has modified is read, the original form of the code is returned rather than the infected code.
- ***Tunneling*:** This virus attempts to bypass detection by an antivirus scanner by installing itself in the interrupt-handler chain. Similar viruses install themselves in device drivers.
- ***Multipartite*:** A virus of this type can infect multiple parts of a system, including boot sectors, memory, and files. This makes it difficult to detect and contain.
- ***Armored*:** An armored virus is coded to make it hard for antivirus researchers to unravel and understand. It can also be compressed to avoid detection and disinfection. In addition, virus droppers and other full files that are part of a virus infestation are frequently hidden via file attributes or unviewable file names.



EXERCISE

4. Give an example of Trojan horse.
5. Why do crackers use back-door program like keystroke logger into the system?
6. In stack-buffer-overflow attack why does the attacker exploits a bug in a program that could be the result of poor programming, like when the programmer neglects to bound check on an input field?
7. What malicious program is used by attacker to insert the virus



CHECK YOUR PROGRESS

Q1. What is a Trojan horse? Give an example.

Q2. Why do programmers sometimes leave a trap door in the program?

Q3. What will happen if an attacker runs a malicious program that writes more data to a buffer located on the stack?

Q4. What is a computer virus?

12.5 SYSTEM AND NETWORK THREATS

In system and network threats the attackers violate and abuse the services of network connections. For example, initially the operating system Solaris 10 had services like FTP, telnet, etc., enabled by default but in the newer model, almost all services are disabled at installation time and must specifically be enabled by system administrators. This significantly reduced the number of attacks on the operating system. Some examples of system and network threats are worms, port scanning, and denial-of-service attacks, which we discuss below:

12.5.1 Worms

Worms are little malign destructive programs whose main task is to duplicate and distribute themselves in the entire network, consuming bandwidth and blocking system resources, thus rendering the entire network insecure and inoperative. Once a worm reaches a new computer system, it will use this system as a host to scan and infect other computers, and this process will be repeated again and again. Worms can be more infectious than traditional viruses as they not only infect local computers, but also all servers and clients connected to the network. Unlike a computer virus that generally requires a host program, a worm does not need a host program. A computer virus has to write its code into the host program and when the host program is executed, the virus is activated. But a worm does not need a host program, as it is an independent program and can run independently and actively carry out attacks.

One example of a worm is "*W32.Sobig.F@mm*" which became the fastest-spreading worm and infected hundreds of thousands of computers. It clogged e-mail inboxes, slowed networks, and took a huge number of hours to clean up. Another

example is the "*Sobig.F*" worm which was disguised as a photo. It targeted Microsoft Windows operating systems and tried to modify the Windows registry. It used its own SMTP engine to e-mail itself to all the addresses found on an infected system. Sobig.F included an attachment for the target e-mail reader to click on, and once clicked, it stored a program called "*WINPPR32.EXE*" in the default Windows directory, along with a text file.

12.5.2 Port Scanning

Port scanning is a mechanism used by crackers to find any system vulnerability which can be later exploited to attack the system. In port scanning, a TCP/IP connection is created to a specific port or a range of ports. For example, consider there is a vulnerability (or a bug) in '*sendmail*' service. Then the cracker could launch a port scanner to try to connect, say, to port 25, of one or many systems. If the connection was successful and if the service answered back, the cracker could try to exploit the bug using buffer overflows. Once the stack buffer is flooded with data then that the cracker can corrupt the stack in such a way as to inject executable code like Trojan horses, back-door programs, and so on.

12.5.3 Denial of Service

Denial-of-service (DOS) attacks are generally network-based attacks that are aimed not at gaining information or stealing resources but rather at disrupting the legitimate use of a system or facility. These attacks result from abuse of some of the fundamental functionality of TCP/IP. One example of DOS attack is an incomplete TCP connection. For instance, an attacker might send a signal that says "*I want to start a TCP connection*" but does not reply back to the server saying "*The connection is now complete*", making the server expect and wait for the acknowledgment. If enough of these sessions are launched, they can eat up all the network resources of the system, disabling any further legitimate TCP connections. Even more dangerous and severe attack is the distributed denial-

of-service attacks (DDOS). These attacks are launched toward a common target from multiple systems which have already been compromised or infected, called zombie systems. DDOS attacks have become more common and are sometimes associated with blackmail attempts.



EXERCISE

8. Distinguish between worn and virus.
9. Give examples of worm and virus.
10. Port scanning is not a worm or virus but it is a mechanism to scan the system for vulnerabilities and weaknesses. (True or False)
11. What are zombie systems?



CHECK YOUR PROGRESS

Q5. How are worms more infectious than traditional viruses?

Q6. Give an example of a worm.

Q7. In port scanning, why does the attacker scan the system for vulnerabilities and weaknesses?

Q8. Why is distributed denial-of-service attack (DDOS) even more dangerous and severe?

12.6 USER AUTHNTICATION

User authentication is one of the major security goals of a system. The aim here is to validate whether the request is coming from a valid user or an unauthorized person. Some of the ways to validate the user are by using *Passwords* or *Biometrics*.

12.7 PASSWORD

Password is a secret word or phrase that is composed of a combination of letters, digits and special characters. A user has to use the

password to gain access and use the services of the system. The use of a password protection scheme is one of the easiest and simplest forms of security mechanism.The idea here is that before any user requests for any service or resource, the user has to first enter the password assigned to it. The system will verify whether the password entered by the user is correct. If the password matches, then the user gains access. Generation of a password is either entrusted to the user itself or in some cases the password is auto-generated by the system.

12.8 PASSWORD VULNERABILITIES

Though *password* is one of the most common methods of validating an authorized user, it has many flaws. For example, it can be guessed, accidentally exposed, sniffed, or illegally transferred from one user to another. Very often people set their passwords with some commonly known information like their own birth dates or their relatives, or the names of their spouses, parents, or children. Such passwords can be easily guessed by a stalking intruder. Another way to guess the password is to use brute force, i.e., try all possible combinations of valid password characters (letters, numbers, and punctuation on some systems)—until the password is found. Short passwords are especially vulnerable to this method. Brute force is less successful where systems allow longer passwords that include both uppercase and lowercase letters, along with numbers and special characters.

In addition to being guessed, passwords are vulnerable to visual or electronic monitoring. An intruder can look over the shoulder of a user (*shoulder surfing*) when the user is logging in and can learn the password easily by watching the keyboard. Alternatively, anyone with access to the network on which a computer resides can seamlessly add a network monitor, allowing him to watch all data being transferred on the network (*sniffing*), including user IDs and passwords. Encrypting the data stream containing the password solves this problem. Even such a system could have passwords stolen, however. For example, if a file is used to contain the passwords, it could be copied for off-system analysis. Or consider a

Trojan-horse program installed on the system that captures every keystroke before sending it on to the application.

Finally, another way of how a password can be compromised is by illegal transfer. Sometimes, innocent users tend to help friends and share the password. This behavior can result in a system being compromised and eventually being accessed by unauthorized users.

For security purposes, simple measures for better password schemes can be used. For example, longer passwords should be preferred that include both uppercase and lowercase letters, along with numbers and special characters. A good technique is to generate the password by using the first letter of each word of an easily remembered phrase that includes upper case – lower case characters, numbers as well as special characters. For example, for the phrase “My mother’s name is Meena Kumari and her date of birth is 12 December 1962” the password might look something like this “Mmni.MK_ahdob.12D62”. This password is hard to crack but easy for the user to remember. The password should also be changed more frequently. For even tighter security, the password can be changed from session to session. A new password is selected (either by the system or by the user) at the end of each session, and that password must be used for the next session. In such a case, even if a password is misused, it can be used only once.

12.9 ENCRYPTED PASSWORD

As the password entered by the user will be matched with the password stored in the system, a stronger protection mechanism is required to protect the list of passwords that are already stored in the system. If this list of passwords stored in system is stolen by some means then the system will be completely compromised. One such password protection mechanism is encryption. In this case, the actual password is first encrypted by using an encryption algorithm. The encryption algorithm will give us an encrypted or encoded form of the actual password. For example, if ‘*x*’ is the password and if ‘*f()*’ is the encryption algorithm, then ‘*y*’ is the encrypted password. Now, this encrypted password is stored in the system and not

the actual password. The strongest point about the encryption method is that even if '*f()*' and '*y*' are known, it is extremely difficult to regenerate or decrypt '*x*' from '*y*'. Now when a user will enter the password, the system will first encrypt the entered password using the same encryption algorithm, and the result is matched with the encrypted password that is already stored in the system. Thus if an attacker somehow steals the encrypted password, he will still be unable to access the system. However, this mechanism will fail if the attacker steals the actual password from an authorized user.

Another problem is that an attacker might be able to crack the password by running the encryption algorithm on a powerful computer and trying the brute force method with many compilations and combinations of password guesses. For this reason, it is always advisable to change the password periodically. New versions of UNIX operating system store the encrypted password entries in a file readable only by the *superuser*. The operating system also includes a "salt," or recorded random number, in the encryption algorithm. The salt is added to the password to ensure that if two plaintext passwords are the same, they result in different encrypted texts.

12.10 ONE TIME PASSWORD (OTP)

A mechanism of paired passwords can be used to tackle the problem of password sniffing and shoulder surfing. In this technique, a password challenge is given to the user wherein the system will generate one part of the password pair, and the user must answer the other part correctly. In this scheme, first, both the system and the user will have to share a secret code that will never be transmitted. The secret code is also called a *personal identification number* or *PIN*. Next, the system will use an algorithm to generate a random integer and this integer is sent to the user. This random integer is called as a *seed*. Now, the user applies both the secret code as well as the random integer to a function and the result is sent back to the system. When the system receives the result, it also applies the secret code as well as the integer to the same function and checks whether the result is correct or not. Every time an authorized user wants to access the

system, the system will generate a new random number and send it to the user. Thus even if an intruder is able to sniff it, he will not be able to generate the answer because the secret will not be transmitted. As the random number is generated for each session, its reuse is also not possible. This kind of *one-time password* scheme is one of the best methods to prevent unauthorized access due to password exposure.

12.11 BIOMETRICS

Biometric information like a fingerprint, retinal scan, facial recognition, etc., can also be used as a password to authenticate a user. A more secure and accurate mechanism will be to implement a two-factor authentication scheme that requires both a password as well as biometric information likea fingerprint scan. In addition to this, if this information is encrypted before transmission, the system can be very resistant to spoofing or replay attacks.

12.12 INTRUSION DETECTION

An intrusion detection system continuously scans the network traffic for any malicious activity or security threats. An intrusion detection system (IDS) can be used on a single host computer called host-based intrusion detection systems (HIDS), or it can be applied on a large network called network-based intrusion detection systems (NIDS). A HIDS reads all the network packets coming to and from the individual devices and scans for any indication of abnormality by comparing them with the signatures of normal packets. Whereas a NIDS is installed at certain points in the network from where the entire network traffic can be monitored. NIDS also performs the packet comparing mechanism. Modern intrusion detection systems use Artificial Neural Network (ANN) to analyze huge volumes of network data. One characteristic of ANN is that it can self-organize and learn from previous errors, and thus can predict and foresee attacks.



EXERCISE

12. Password is a secret word or phrase that is composed of a combination of _____.
13. Generation of a password is either entrusted to the user itself or in some cases the password is auto-generated by the system. (True or False)
14. What are some of the risks of using the password?
15. What are some of the common mistakes people do while setting their passwords.
16. How illegal transfer of passwords can compromise our entire system?
17. A stronger protection mechanism is not required to protect the list of passwords that are already stored in the system. (True or False)
18. What is a personal identification number (PIN) and a seed?



CHECK YOUR PROGRESS

Q9. How do cracker use brute force attack to crack password?

Q10. What measures to take to protect oneself from the abuse of brute force attacksused to crack passwords?

Q11. What is shoulder surfing and sniffing?

Q12. Why a stronger protection mechanism is required to protect the list of passwords that are already stored in the system?

Q13. Even if an attacker somehow steals the encrypted password, why he will still be unable to access the system?

Q14. In one time password generation scheme, even if an intruder is able to sniff the auto-generated random number, why he will still not be able to generate the password?



12.13 LET US SUM UP

- Any theft or loss of such data, by accident or by fraud, can cause huge losses to a company.
- The following list includes several forms of accidental and malicious security violations: breach of confidentiality, breach of integrity, breach of availability, theft of service and denial of service.
- Three different attacks used by intruders and crackers are: masquerading attack, replay attack and man-in-the-middle attack.
- A Trojan horse is a malicious program that looks legitimate or harmless or very enticing, but its actual intent may be to damage, disrupt, or steal the data.
- Spyware is another kind of Trojan horse, which is automatically installed when we install another software.
- Sometimes the programmer might leave a trap door or a secret entry point in software that lets him surpass the security procedure and gain access to the system.
- Keystroke logger is a back-door program, which records everything entered on the keyboard (including passwords and credit card numbers).
- A logic bomb is a malicious code that is intentionally inserted by an unscrupulous programmer and gets activated when certain conditions are met.
- In stack-buffer-overflow attack, an attacker can run a malicious program that writes more data to a buffer located on the stack than what is allocated for that buffer.
- A virus is a piece of vicious code that is hidden in a program or a file that might seem legitimate and genuine.
- Worms are little malign destructive programs whose main task is to duplicate and distribute themselves in the entire network.
- Port scanning is a mechanism used by crackers to find any system vulnerability which can be later exploited to attack the system.

- Denial-of-service (DOS) attacks are generally network-based attacks that are aimed not at gaining information or stealing resources but rather at disrupting the legitimate use of a system or facility.
- Denial-of-service (DOS) attacks result from abuse of some of the fundamental functionality of TCP/IP.
- Distributed denial-of-service attacks (DDOS) are launched toward a common target from multiple systems which have already been compromised or infected, called zombie systems.
- Password is a secret word or phrase that is composed of a combination of letters, digits and special characters.
- A user has to use the password to gain access and use the services of the system.
- Generation of a password is either entrusted to the user itself or in some cases the password is auto-generated by the system.
- Password has many flaws: it can be guessed, accidentally exposed, sniffed, or illegally transferred from one user to another.
- Brute force on password is less successful where systems allow longer passwords that include both uppercase and lowercase letters, along with numbers and special characters.
- Another good technique for user to generate the password is by using the first letter of each word of an easily remembered phrase that includes upper case – lower case characters, numbers as well as special characters.
- Encrypting the data stream containing the password solves the problem of sniffing.
- Illegal transfer of password is when innocent users tend to help friends and share the password.
- Encryption provides a stronger protection mechanism to protect the list of passwords that are already stored in the system.
- If ‘x’ is the password, ‘f()’ is the encryption algorithm, and ‘y’ is the encrypted password, then the strongest point about the encryption method is that even if ‘f()’ and ‘y’ are known, it is extremely difficult to regenerate or decrypt ‘x’ from ‘y’.

- The mechanism of paired passwords can be used to tackle the problem of password sniffing and shoulder surfing.
- In this scheme, first, both the system and the user will have to share a common secret code that will not be transmitted and the system will generate a random integer which is only sent to the user. Both the system and the user will generate the result using the secret code and the random integer. Access is granted only if the result is matched.
- Even if an intruder is able to sniff the one-time generated random number, he will not be able to generate the answer because the secret will not be transmitted. As the random number is generated for each session, its reuse is also not possible.
- One-time password scheme is one of the best methods to prevent unauthorized access due to password exposure.
- A more secure and accurate mechanism will be to implement a two-factor authentication scheme that requires both a password as well as biometric information like a fingerprint scan.
- An intrusion detection system continuously scans the network traffic for any malicious activity or security threats.
- An intrusion detection system (IDS) can be used on a single host computer called host-based intrusion detection systems (HIDS), or it can be applied on a large network called network-based intrusion detection systems (NIDS).
- An intrusion detection system reads the network packets and scans for any indication of abnormality by comparing them with the signatures of normal packets.
- Modern intrusion detection systems use Artificial Neural Network (ANN) to analyze huge volumes of network data.



12.14 FURTHER READING

1. Stallings W.(2005),Operating Systems: Internals and Design Principles,Pearson/Prentice Hall.
2. Deitel H., Deitel P., Choffnes D. R. (2005),Operating Systems,Pearson



12.15 ANSWERS TO CHECK YOUR PROGRESS

Ans to Q No 1: A Trojan horse is a malicious program that looks legitimate or harmless or very enticing, but its actual intent may be to damage, disrupt, or steal the data. An example of the Trojan horse is an attachment that is received in an e-mail or on some advertisement on social media that seems to be from an authentic source, but on clicking the attachment a malicious code is executed which spreads to other files and damages the computer. Spyware is another kind of Trojan horse, which is automatically installed when we install another software.

Ans to Q No 2: Programmers sometimes leave a trap door for genuine testing and debugging purpose, or to give service technicians emergency access. But it might become a grievous threat if an intruder or a dishonest programmer uses it to gain unhindered access to the entire system.

Ans to Q No 3: If an attacker can run a malicious program that writes more data to a buffer located on the stack than what is allocated for that buffer, then it might result in corruption of adjacent data on the stack, and will often cause the program to crash or operate incorrectly.

Ans to Q No 4: A virus is a piece of vicious code that is hidden in a program or a file that might seem legitimate and genuine. Once a virus reaches a target machine, a Trojan horse known as a virus dropper inserts the virus into the system. Once the virus is installed and activated, it replicates itself multiple times and wreaks havoc by damaging files

and causing system crashes and malfunctions. Viruses usually spread through the internet via e-mail, or when users download viral programs or files from the internet file-sharing services or exchange infected files/disks.

Ans to Q No 5: Worms can be more infectious than traditional viruses as they not only infect local computers, but also all servers and clients connected to the network. Once a worm reaches a new computer system, it will use this system as a host to scan and infect other computers, and this process will be repeated again and again. Moreover, a worm does not need a host program. A computer virus has to write its code into the host program and when the host program is executed, the virus is activated. But a worm does not need a host program, as it is an independent program and can run independently and actively carry out attacks.

Ans to Q No 6: example is the "Sobig.F" worm which was disguised as a photo. It targeted Microsoft Windows operating systems and tried to modify the Windows registry. It used its own SMTP engine to e-mail itself to all the addresses found on an infected system. Sobig.F included an attachment for the target e-mail reader to click on, and once clicked, it stored a program called "WINPPR32.EXE" in the default Windows directory, along with a text file.

Ans to Q No 7: In port scanning, the attacker scans the system for vulnerabilities and weaknesses because once the vulnerability is found, it can be later exploited to launch a severe attack.

Ans to Q No 8: Distributed denial-of-service attacks (DDOS) are even more dangerous and severe because these attacks are launched toward a common target from multiple systems which have already been compromised or infected, called zombie systems. DDOS attacks have become more common and are sometimes associated with blackmail attempts.

Ans to Q No 9: A way to guess the password is by using brute force, i.e., try all possible combinations of valid password characters (letters,

numbers, and punctuation on some systems)—until the password is found. Short passwords are especially vulnerable to this method.

Ans to Q No 10: To protect oneself from the abuse of brute force attack to crack password, simple but effective measures can be taken. For example, longer passwords should be preferred that include both uppercase and lowercase letters, along with numbers and special characters. A good technique is to generate the password by using the first letter of each word of an easily remembered phrase that includes upper case – lower case characters, numbers as well as special characters. For example, for the phrase "My mother's name is MeenaKumari and her date of birth is 12 December 1962" the password might look something like this "Mmni.MK_ahdob.12D62". This password is hard to crack but easy for the user to remember. The password should also be changed more frequently.

Ans to Q No 11: An intruder can look over the shoulder of a user (shoulder surfing) when the user is logging in and can learn the password easily by watching the keyboard. Alternatively, anyone with access to the network on which a computer resides can seamlessly add a network monitor, allowing him to watch all data being transferred on the network (sniffing), including user IDs and passwords.

Ans to Q No 12: As the password entered by the user will be matched with the password stored in the system, a stronger protection mechanism is required to protect the list of passwords that are already stored in the system. If this list of passwords stored in system is stolen by some means then the system will be completely compromised.

Ans to Q No 13: In encryption mechanism, the actual password is first encrypted by using an encryption algorithm. The encryption algorithm will give us an encrypted or encoded form of the actual password. For example, if 'x' is the password and if 'f()' is the encryption algorithm, then 'y' is the encrypted password. Now, this encrypted password is stored in the system and not the actual password. The strongest point about the encryption method is that even if 'f()' and 'y' are known, it is extremely difficult to regenerate or decrypt 'x' from 'y'. Now when

a user will enter the password, the system will first encrypt the entered password using the same encryption algorithm, and the result is matched with the encrypted password that is already stored in the system. Thus if an attacker somehow steals the encrypted password, he will still be unable to access the system.

Ans to Q No 14: In one time password generation scheme, the answer can only be generated by using both the auto-generated random number (seed) and the secret code called a personal identification number (PIN). Only the random number will be sent to the user, the secret code (PIN) is not transmitted. The secret code will already be there with both the user and the system. So, even if an intruder is able to sniff the auto-generated random number, he will still not be able to generate the password. Also, as the random number is generated for each session, its reuse is also not possible.



12.16 MODEL QUESTIONS

Q1. What are different forms of accidental and malicious security violations?

Q2. Explain each of the different attacks: masquerading attack, replay attack and man-in-the-middle attack.

Q3. Explain Trojan horse, Trap door and Logic bomb with suitable examples.

Q4. How the stack-buffer-overflow attack is used to gain unauthorized access to the target system?

Q5. Explain briefly all the several categories of viruses.

Q6. What is a Worm and how it disrupts the network systems? Give examples.

Q7. Explain the difference between a worm and a virus and explain how a worm can be even more dangerous than a virus.

Q8. Explain Port scanning with a suitable example.

Q9. Explain the Denial-of-service attack with a suitable example.

Q10. Explain in detail all the password protection mechanisms.

Q11. How passwords become vulnerable to visual or electronic monitoring? What measure to take to prevent such attacks?

Q12. Why a stronger protection mechanism is required to protect the list of passwords that are already stored in the system and how encryption safeguards the password?

Q13. Explain the one time password scheme in detail.

Q14. Write short notes on Biometric and Intrusion Detection System.

*** ***** ***

UNIT 13: MULTIPROCESSOR SYSTEMS

UNIT STRUCTURE

13.1 Learning Objectives

13.2 Introduction

13.3 Multiprocessor Systems

13.4 Types of Multiprocessor Operating Systems

13.5 Multiprocessor OS Functions and Requirements

13.6 Multiprocessor Synchronization

13.7 Let Us Sum Up

13.8 Further Reading

13.9 Answer to Check Your Progress

13.10 Model Questions

13.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- define the multiprocessors operating systems
- describe types of multiprocessors operating systems
- explain functions of multiprocessors operating systems
- explain synchronization on multiprocessors operating systems

13.2 INTRODUCTION

In the previous unit, we learned about attacks which are intended by individuals to steal information, damage information, or cause any kind of destruction; and measures to protect the system from such attacks. In this unit, we will learn about multiprocessor systems. We will also learn about the types of multiprocessor systems, its functions and requirements and about synchronization of multiprocessor systems. In the next unit, we will explore the topic of distributed operating system in detail.

13.3 MULTIPROCESSORS SYSTEMS

A single processor operating system (OS) contains only one Central Processing Unit (CPU). This single process utilizes the computer memory,

clock, bus, I/O, peripheral devices, etc. A multiprocessor operating system contains more than one CPU inside a single computer system and it has more demand nowadays. The processors work in parallel and they share computer memory, clock, bus, I/O, peripheral devices, etc. among them. For example: to handle the weather forecasting data multiprocessor systems are used. Like other operating systems, multiprocessors systems also have four components such as CPU, I/O devices, I/O processors, and memory (Figure. 13.1).

The multiprocessors system has the following system advantages.

i) **More throughput**: As more than one processors work in parallel, the throughput of the overall multiprocessor system is more as compared to other uniprocessor systems.

ii) **More Reliable**: Due to more number of CPU in a computer system, the overall reliability of the multiprocessor OS is increased because the failure of one CPU cannot affect the system. The other CPU handles the system.

iii) **Cost-Effective**: Due to the sharing of computer memory, clock, bus, I/O, peripheral devices, etc. among the CPUs, the multiprocessor OS is more cost-effective.

iv) **Parallel Processing**: Parallel processing is more in multiprocessor OS because more than one CPU works at the same time.

The multiprocessors system has the following disadvantages.

i) The system is complex.

ii) The multiprocessor system may have more traffic due to the parallel execution of CPU functions.

iii) The throughput may be reduced sometimes in the shared system when one CPU waits for the same I/O that is used by another CPU.

iv) CPU synchronization is complex because more than one CPU work at a single instance of time.

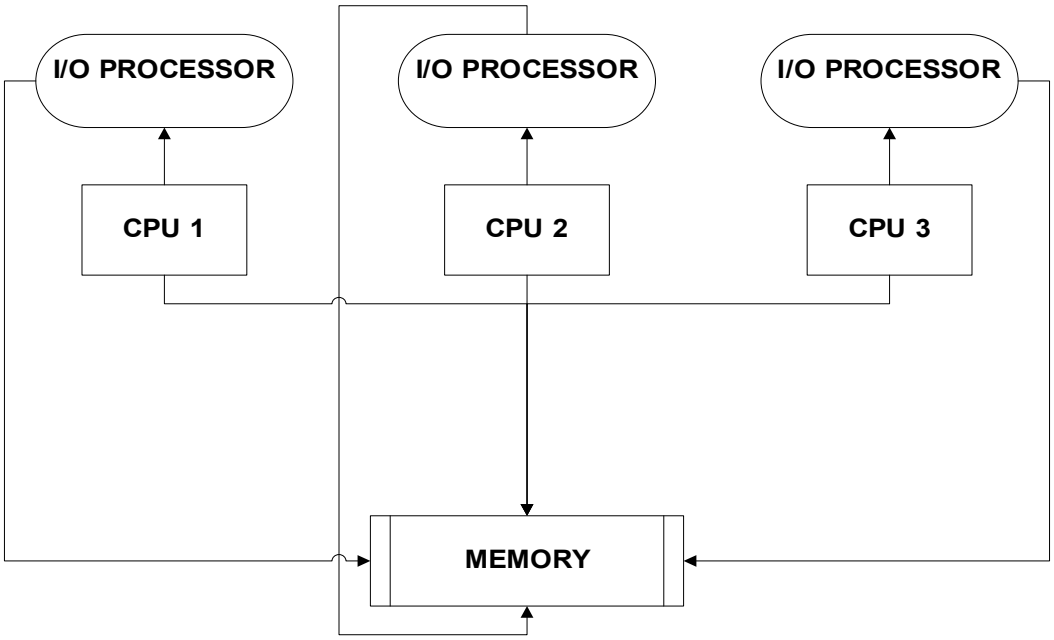


Figure 13.1: Block Diagram of Multiprocessor Operating System

13.4 TYPES OF MULTIPROCESSOR OPERATING SYSTEM

There are mainly 6 multiprocessors systems and they are described below.

i) **Symmetric Multiprocessor OS:**

In symmetric multiprocessor system, each processor keeps identical copy of the OS and thus, they communicate between themselves like a peer to peer structure. In this process each process has its own OS and user space. The system maintains a global queue to consider all the ready queue. The processes are allocated to the processor from the global queue at the time of execution. Sometimes, two processes may get the same processoror one processor may get more processes than other processor. Sometimes, some processors may be idle. The locking technique and CPU balancing scheduler are used to overcome the above mentioned issues.

ii) **Asymmetric Multiprocessor OS:**

The asymmetric multiprocessor system works on the master slave relationship. The master processor gives instructions to all the slaves

processor to perform the predefined works. The master processor maintains the ready queue. With the help of a scheduler, the master processor allocate the processes to the other processors.

iii) **Shared Memory Multiprocessor OS:**

In the shared memory multiprocessor system, the processors share their memory and have direct access to the main memory. The processor permits to access the data or information of other processor's.

iv) **Distributed Memory Multiprocessor OS:**

In the distributed memory multiprocessor system, the processors have their own memory module and this memory can be accessed only by its processors. A message passing (MP) technique is used to allow a processor to access other memory modules associated with other processors.

v) **UMA Multiprocessor OS:**

UMA Multiprocessor system means **"Uniform Memory Access Multiprocessor"** where all the processor access the main memory with an uniform speed and uniform time. It is part of shared multiprocessor system where a common physical memory is present for all the processors.

vi) **NUMA Multiprocessor OS:**

NUMA Multiprocessor system means **"Non Uniform Memory Access Multiprocessor".** In this system, the memory access time is depended on the processor's memory location. So multiple controllers are present at NUMA.



CHECK YOUR PROGRESS

Q1. What do you mean by a multiprocessor system?

Q2. The multiprocessor system is too complex. (True or False)

Q3. UMA means Uniform Memory Access Multiprocessor (True or False)

Q4. NUMA means Non Uniform Memory Access Multiprocessor (True or False)

13.5 MULTIPROCESSOR OS FUNCTIONS AND REQUIREMENTS

A program can be designed for a problem. A multiprocessor system has the following requirements

i) It must offers the functionality of a multiprogramming along with the features that supports a multiprocessor system.

ii) All the concurrent process, threads, kernel should be synchronized in a multiprocessor system.

iii) The synchronization of a multiprocessor system can be done with the help of a locks.

iv) For effective use of multiprocessors system, the process scheduling is necessary. The processor should be allocated in such a manner that design objectives of the system remain consistent. This technique help to increase the throughput of the system.

The efficient use of processor and its allocation always effects the speed of the multiprocessor system.



EXERCISE

1. Discuss About the Multiprocessor OS requirements.
2. State the difference between Symmetric and Asymmetric multiprocessor system.
3. What is UMA and NUMA multiprocessor system?
4. What is issues of multiprocessor system?

13.6 MULTIPROCESSOR SYNCHRONIZATION

A multiprocessor system supports parallel program execution and sharing of data that demands process synchronization. A reliable and correct mechanism is essential to serialize the processor and its access and this mechanism is known as synchronization. For synchronization, following methods are considered in multiprocessor systems.

i) Test and Set

ii) Compare and Swap

iii) Fetch and Add.

I) Test and Set: The test and set is an instruction. It automatically reads the contents of a memory and modifies the contents of a memory location in one memory cycle. It is as follows:

```
Function Test_and_Set (var m: Boolean);
    begin
        Test_and_Set:=m;
        m:= true
    end;
```

The Test_and_set instruction defined above works atomically i.e. as one uninterruptible unit. If two different CPU execute the Test_and_Set operation simultaneously, they will be executed sequentially in and any arbitrary order.

ii) Compare and Swap: The compare and swap instruction is used in the synchronization of concurrent updates to a memory location. This instruction is defined as follows. In this instructions, r1 and r2 are the registers of a processor and m is a memory location

```
Function Compare-and-Swap(var m: Boolean); Boolean;
var X: integer;
begin
    X:=m;
    if X = r1
        then {m:= r2;
        z:=1}
    else
        [r1:=X; z:=0}
end;
```

Initially, the memory location value is stored in a variable X. When the contents of r1(register) and X(memory location) are same, then the contents of the register 2 is transferred to m and sets z to 1. Otherwise, the contents of X is again transferred to r1 and set z to 0. Variable z is a flag that indicates the success of the execution. This instruction can be used to synchronize concurrent access to a shared variable.

iii) Fetch and Add: The fetch and add instruction is a multiple operation memory access instruction which automatically adds a constant to a memory location and returns the previous contents of the memory location. This instruction is defined as follows:

```
Function Fetch-and-add (m: integer; c: integer);
Var temp: integer;
Begin
     Previous:= m;
     M:= m + c;
     Return (Previous)
end;
```

When several processors concurrently execute a fetch-and-add instruction on the same memory location, these instructions are combined in the network and are executed by the network in the following way:

i) A single increment, which is the sum of the increments of all these instructions, is added to the memory location.

ii) A single value is returned by the network to each of the processors, which is an arbitrary serialization of the execution of the individual instructions.

iii) If a number of processors simultaneously perform fetch-and-add instructions on the same memory location, the net result is as if these instructions were executed serially in some arbitrary order. The fetch-and-add instruction is powerful and it allows the implementation of P and V operations on a general semaphore.



CHECK YOUR PROGRESS

Q5. What issymmetric multiprocessor system?

Q6. What is Test and Set?

Q7. What is Fetch and Add?

Q8. What is Compare and swap?

Q9. Why does synchronization is important in multiprocessor system?



13.7 LET US SUM UP

- A multiprocessor operating system contains more than one CPU inside a single computer system and it has more demand nowadays. The processors work in parallel and they share computer memory, clock, bus, I/O, peripheral devices, etc. among them.
- The multiprocessors system has the following system advantages: More throughput, More Reliable, Cost-Effective, Parallel Processing.
- The multiprocessor system may have more traffic due to the parallel execution of CPU functions.
- The throughput may be reduced sometimes in the shared system when one CPU waits for the same I/O that is used by another CPU.
- CPU synchronization is more complex because more than one CPU work at a single instance of time.
- There are mainly 6 multiprocessors systems and they are: Symmetric Multiprocessor OS, Asymmetric Multiprocessor OS, Shared Memory Multiprocessor OS, Distributed Memory Multiprocessor OS, NUMA Multiprocessor OS, UMA Multiprocessor OS.
- A multiprocessor system must offers the functionality of a multiprogramming along with the features that supports a multiprocessor system. All the concurrent process, threads, kernel should be synchronized in a multiprocessor system.The synchronization of a multiprocessor system can be done with the help of a locks.
- A multiprocessor system supports parallel program execution and sharing of data that demands process synchronization. A reliable and correct mechanism is essential to serialize the processor and its access and this mechanism is known as synchronization. For synchronization, following methods are considered in multiprocessor systems :Test and Set, Compare and swap, Fetch and add.



13.8 FURTHER READING

1. Jerraya, A., & Wolf, W. (2004).Multiprocessor systems-on-chips.Elsevier.



13.9 ANSWERS TO CHECK YOUR PROGRESS

Ans to Q No 1: A multiprocessor operating system contains more than one CPU inside a single computer system and it has more demand nowadays. The processors work in parallel and they share computer memory, clock, bus, I/O, peripheral devices, etc. among them.

Ans to Q No 2: True

Ans to Q No 3: True

Ans to Q No 4: True

Ans to Q No 5: In symmetric multiprocessor system, each processor keep a copy of the OS and using this identical OS, they communicate in between them like a peer to peer structure. In this process each process has its own OS and user space. The system maintains a global queue to consider all the ready queue. The processes are allocated to the processor from the global queue at the time of need. Sometimes, two processes may get the same process or sometimes, one processor may get the more processes than other one. Sometimes, some processors may idle. The locking technique and CPU balancing scheduler are used to overcome the mentioned issues.

Ans to Q No 6: The test and set is an instruction to handle synchronization. It automatically reads the contents of a memory and modifies the contents of a memory location in one memory cycle. It is as follows:

```
Function Test_and_Set (var m: Boolean);
begin
     Test_and_Set:=m;
     m:= true
end;
```

Ans to Q No 7: The fetch and add instruction (synchronization handle) is a multiple operation memory access instruction which automatically adds a constant to a memory location and returns the previous contents of the memory location.

Ans to Q No 8: The compare and swap instruction is used in the optimistic synchronization of concurrent updates to a memory location.

Ans to Q No 9: A multiprocessor system supports parallel program execution and sharing of data that demands process synchronization. A reliable and correct mechanism is essential to serialize the processor and its access and this mechanism is known as synchronization.



13.10 MODEL QUESTIONS

Q1. What is a multiprocessor system?

Q2. What are the different advantage and disadvantages of the multiprocessor system?

Q3. What are the types multiprocessor system?

Q4. Explain the different types of multiprocessor system.

Q5. Difference between UMA and NUMA multiprocessor system.

Q6. Explain the Multiprocessor OS requirements.

Q7. Explain multiprocessor synchronization techniques.

*** ***** ***

UNIT 14: DISTRIBUTED OPERATING SYSTEMS

UNIT STRUCTURE

14.1 Learning Objectives

14.2 Introduction

14.3 Distributed Operating Systems

14.4 Distributed Processing

14.5 Failures Models of Distributed Systems

14.6 Remote Procedure Calls

14.7 Distributed Shared Memory

14.8 Let Us Sum Up

14.9 Further Reading

14.10 Answer To Check Your Progress

14.11 Model Questions

14.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- define distributed operating systems
- explain processing distributed systems
- describe about the failure models of distributed systems
- explain about RPC and shared memory

14.2 INTRODUCTION

In the previous unit, we learned about multiprocessor systems. We also learned about the types of multiprocessor systems, its functions and requirements and synchronization of multiprocessor systems. In this unit, we will learn about distributed operating systems and distributed processing. We will also learn about RPC, shared memory and the different failure models of distributed systems. In the next unit, we will explore the topic of multiprogramming system in detail.

14.3 DISTRIBUTED OPERATING SYSTEMS

Distributed Operating System (DOS) is another type of operating system (OS). Generally, OS is software that manages the hardware resources and other software efficiently. In the case of DOS, different systems are interconnected and linked through a communication bus or cable, etc. DOS handles multiple real-time applications for multiple users. This system is also known as a loosely coupled system because each system is located remotely and they have their CPU, memory, and all. These loosely coupled systems have no hardware connections at the CPU level but they have their external interfaces of the software.

The DOS has the following advantages:

i) One system present at one location can access the resources of the other system located at other locations with the help of the resource sharing facility of the DOS.

ii) Failure of one system in DOS doesn't affect the other system in the DOS.

iii) As the system and workload are distributed, the overall load of the host system is reduced in the DOS.

iv) Due to the interconnection of the remote system, there will be a reduction in the data processing in DOS.

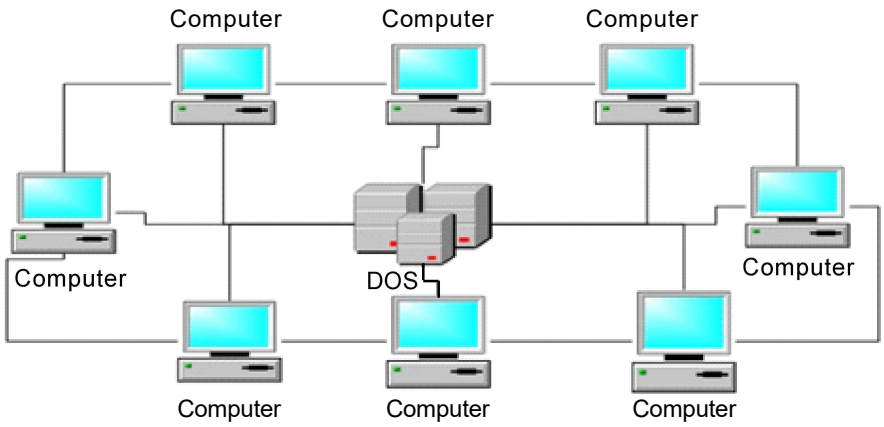


Figure 14.1: Typical View of DOS

In the distributed system, the user does not know, on which machine the required programs are running and where their files are stored. So it

has some disadvantages also. The disadvantages of the distributed operating system are:

i) Failure of the primary network will entirely stop communication in DOS.

ii) The DOS is very expensive.

iii) The underlying software of the DOS is highly complex.

14.4 DISTRIBUTED PROCESSING

In DOS all the individual systems are located remotely. So, Distributed Processing (DP) refers to the processing of doing one task by more than one processor. It refers to the use of more than one processor to run a task. To understand this concept, let's take the example of Oracle client-server architecture. In Oracle, database system has two parts as part A and part B (Fig.14.2). In PART A, the client and server are interconnected via some network cabling. The client and server are in different locations. In PART B, a single system has more than one processor. These processors execute the different tasks of the client. In Oracle, the client takes the query from the user and takes the data from the oracle server, and presents this information to the user. The client systems are not location-dependent. If data is removed or moved to another oracle server, the client continues its function with a little modification. So, this process is known as distributed processing where the client or processor or systems are responsible to perform the task without giving any information to the user.

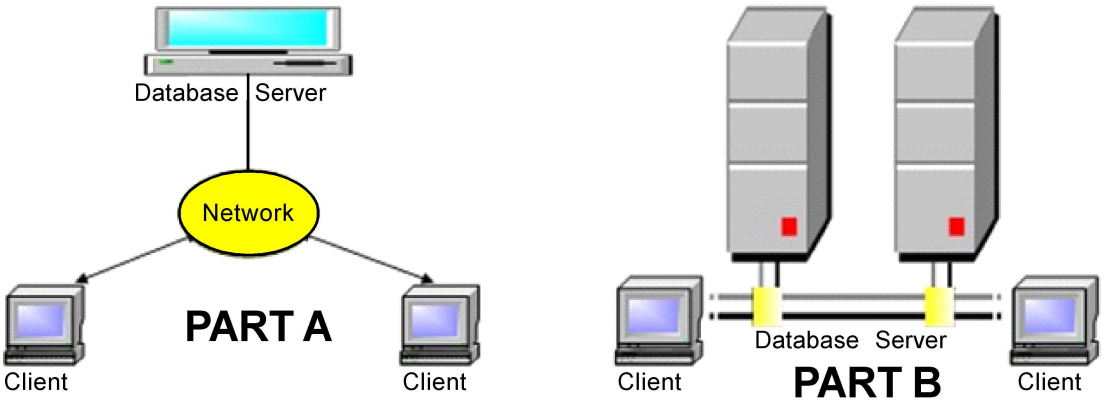


Figure 14.2: Oracle Database system

During the time of distributed processing, some common characteristics are always followed.

i) **Resource Sharing**: Resource sharing is the process of sharing information between the systems processors. The resource manager of the DOS controls the access, naming scheme, and concurrency. The resource manager follows some resource-sharing models to describe how the resources are shared, used, and interact.

ii) **Openness**: A DOS systems are always in openness mode so that a new component can join in the existing DOS. It will help to increase the processing of the system.

iii) **Concurrency:** Distributed tasks are executed in concurrent processes. Sharing of resources and components updating in DOS is always in a concurrent manner and it is well coordinated. If the coordination is not well, then it may lead to inconsistency.

iv) **Fault Tolerance:** DOS must assure the availability during the time of failure.

v) **Transparency:** A DOS is always transparent**.** The transparency of the DOS is perceived by the user of the DOS or sometimes by the application program. Some of the transparency of the DOS is access transparency, location transparency, concurrency transparency, replication transparency, failure transparency, scaling transparency, and performance transparency.

Overall the distributed processing is a term that refers to the use of more than one computer system to run a single process. This refers the parallel processing where a single computer uses more than a CPU for a given task. Sometimes it refers to a LAN where a single program can run at different systems. Another example of distributed processing is distributed database where more than one computer can use the data which are stored in a different location.

EXERCISE

1. Explain the design issues of DOS.
2. What is the difference between transparency and concurrency in DOS?

14.5 FAILURES MODELS OF DISTRIBUTED SYSTEMS

Before explaining the failure models of a Distributed system(DS), let's understand the basic layer architecture of the DOS. In DOS (Fig.14.3) architecture, the application software layer provides the services to the user. The DOS is responsible for shared resources, remote procedures. The shared resources layer is responsible for shared resources and can allocate the resources for use. The resources management layer manages the resources locally and makes them available for share. A kernel is a process that handles the scheduling task, and IPC. The hardware layer is responsible for hardware devices of the DOS. The communication system provides transporting systems services and application messaging services.

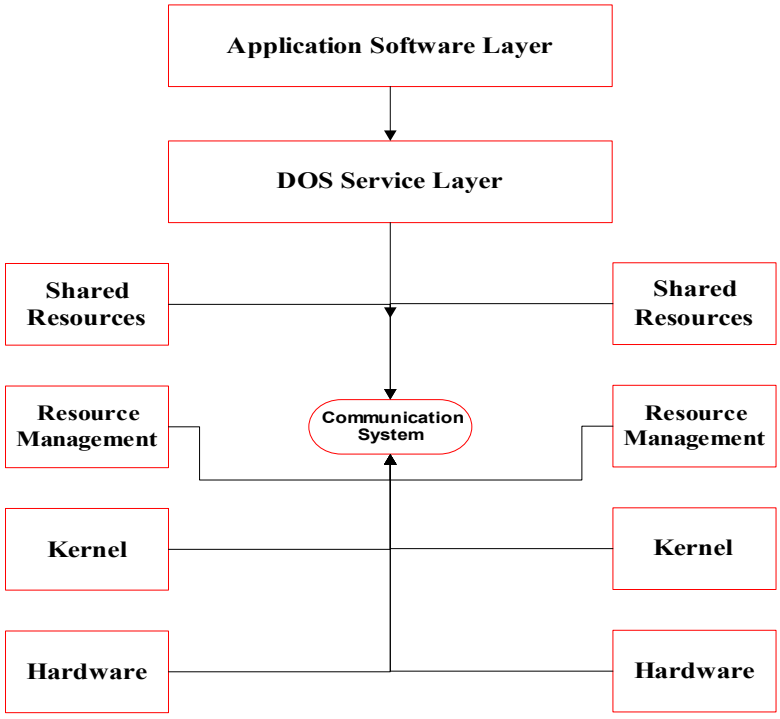


Figure 14.3: DOS Architecture

Based on the architecture of the DOS, the general failure model of the DOS can be explained as shown in Figure 14.4.

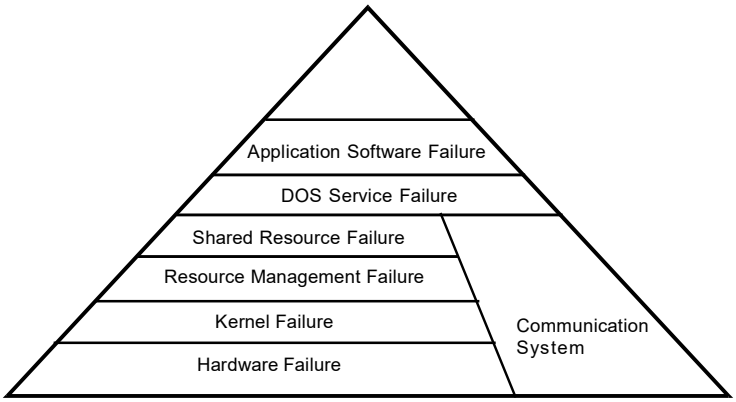


Figure 14.4: General Failure Model of DOS

According to Figure.14.4, the failures in DS may occur at the different layers. All the failures are correlated. The application layer failure occurs when the application layer is unable to provide service to the user. It happens when DOS service failure occurs. Again the DOS service failure depends on the failure of the shared resources. If the resource management unit of DS is unable to manage the resources and halting occurs during the resources sharing then failure may occur in the shared resource layer. The hardware of one device or CPU fails then the kernel of the CPU may affect and it may fail. Overall the failure of the DS is dependent on hardware. If it fails, it will affect the entire layer. The communication system failure is one of the causes of the failure of DS.

Depending on the cause of the failure, the failures in DS are categorized in a different section.

i) **Method Failure:**

In method failure, the DS is halted and unable to execute a specific task. It is due to the application software service and DOS service failure.

ii) **System Failure:**

In system failure, due to computer error, hardware issue, or bus failure, the DS fails to perform a specific task and is halted. It is due to the hardware and kernel layer failure.

iii) Secondary Storage Device or Shared Resource Failure:

In this type of failure, the secondary storage of the system may not be accessed by the other system in the DS and due to this issue, the resources may not be shared among the systems. It is related to share resource failure and resource management unit.

iv) Communication Medium Failure:

This type of failure happens when one system of DS cannot communicate with the other system of the DS within a network. It is due to the link failure of the DS.

A DS has different modes of failure on the server-side and these are as follows.

i) Byzantine or Arbitrary Failures:

It is an arbitrary failure that is caused across the servers of DS. Due to these failures, the server of the DS behaves arbitrarily and it also responds arbitrarily. So, there will be a chance of getting inappropriate output from the server and also chances of getting the malicious procedures and duplicate posts from the server to the client.

ii) Omission Failures:

It occurs on the server-side due to the lack of reply from the server across the DS. The main reason for this failure is that the server is not listening properly.

iii) Timing failures:

In timing failures, the response time of the server is more than the expected range. In this situation, the client may not wait for the server response.

iv) Crash Failures:

Crash failures may occur in the DS due to the crashing of the OS. If these failures occur, then the operations of the server are halted for some time.



CHECK YOUR PROGRESS

Q 1. What is the Distributed operating system?

Q 2. State two advantages of DOS.

Q 3. What are the types of distributed operating systems?

Q 4. In distributed system, each processor has its own _____

Q 5. What is not a major reason for building distributed systems?

14.6 REMOTE PROCEDURES CALLS

Remote Procedure Calls (RPC) is a technique or protocol which is used by one program to put a request for a service from another program on a network without having the network details. In this situation, the process of the communication devices does not need to understand the network background and details, because the RPC takes care of the entire process. In many aspects, it is similar to Inter-Process Communication (IPC), a message-based communication scheme is required to provide the remote service because we are dealing with such an environment where processes are running in different environments. As the processes are running in different systems via the network in RPC, a shared memory system with a message passing technique may not work in RPC. That is the reason of using a message-based communication scheme. Due to passing of information over a network, an RPC message should be well structured.The communication of RPC in a client-server mode is based on the following steps. The client has a request message, sends to the server. The server receives the request; sends an acknowledgment to the sender again. The client is blocked while the server is processing the call and only resumed execution after the server is finished.

- The client sends the message to the server by client stub and system call by putting all the parameters.
- The message is sent from the client to the server by the client's operating system.
- The message is passed to the server stubby the server operating system.

- The client's parameters are removed from the message by the server stub.
- Then, the server procedure is called by the server stub.

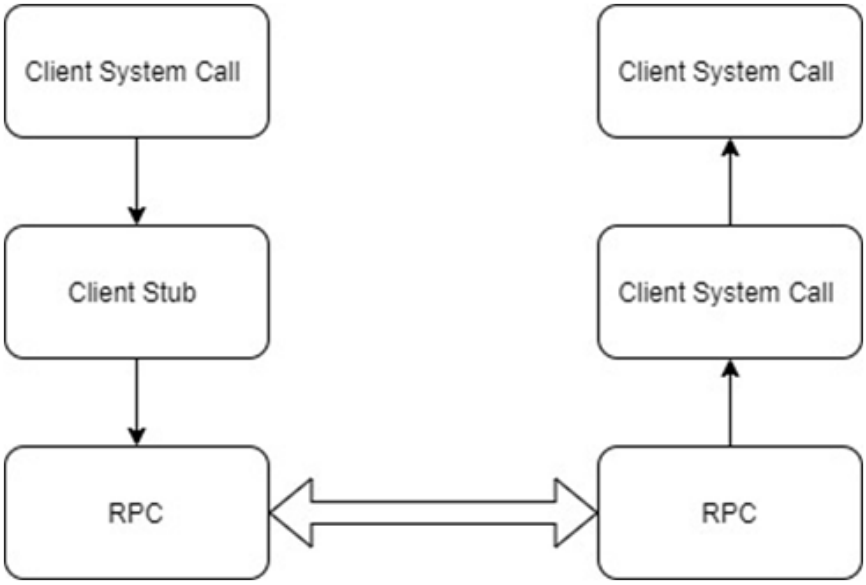


Figure 14.5: RPC communication

The advantages of RPC are given below.

- The process-oriented and thread-oriented models are supported by the RPC.
- The users are unaware of the internal message passing mechanism of RPC.
- The effort to re-write and re-develop the code is minimum in remote procedure calls.
- It can be used in a distributed environment as well as the local environment.

The disadvantages of RPC are presented below.

- The implementation of RPC is not standard due to its diverse implementation.
- The system costs increases due to RPC.

EXERCISE

3. Explain the RPC communication state its difference between IPC.
4. What is the need of RPC?

14.7 DISTRIBUTED SHARED MEMORY

In a distributed system, the distributed shared memory (DSM) acts as a resource management technique that does not have any physical memory. It creates virtual space that is shared among the nodes of a distributed system.

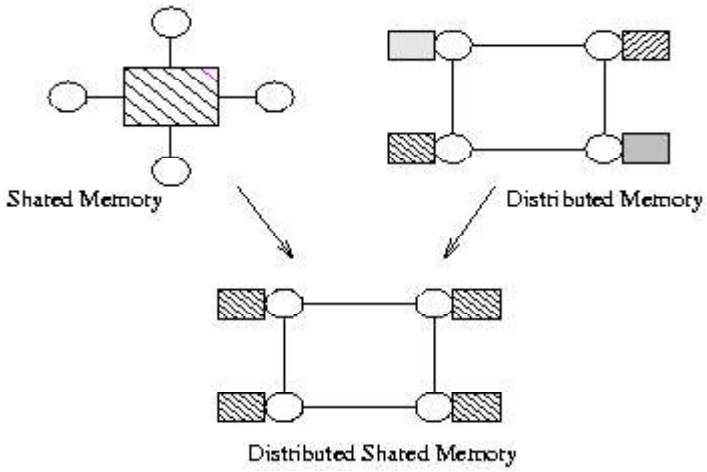


Figure 14.6: Architecture of DSM

A distributed shared memory allows end-users processes to access shared data without using inter-process communications. In other words, to make IPC transparent the distributed shared memory is used. It is different with message passing.

Message passing	Distributed shared memory
Variables are not shared directly	Variables are shared directly
The cost of communication is understandable	The cost of communication is not understandable.
Processes should execute at the same time	Processes should execute at a different time within a period.

Table 14.1: Difference between Message Passing and DSM

The advantages of DSM are as follows.

a. The DSM is scalable, i.e., the new system can be adopted under the DSM.
b. The DSM hides the message passing and can also handle a complex data system without replication.
c. The architecture of DSM is cheaper than a multiprocessor system with large virtual memory space.
d. The DSM system can improve performance by speeding up data access.

The disadvantages of DSM are presented below.

a. The DSM may be the cause of system performance.
b. Should provide for protection against simultaneous access to shared data such as lock.



CHECK YOUR PROGRESS

Q 6. What is RPC?

Q 7. What is IPC?

Q 8. What are the advantages of RPC?

Q 9. An RPC (remote procedure call) is initiated by the _____

Q10. A remote procedure call is IPC (True or False)

Q11. A remote procedure is uniquely identified by ____.



14.8 LET US SUM UP

- Distributed operating system (DOS) is another type of operating system (OS). Generally, OS is a software that manages the hardware resources and other software efficiently.
- These loosely coupled systems have no hardware connections at the CPU level but they have their external interfaces of the software.
- In DOS all the individual systems are located remotely. So, Distributed Processing (DP) refers to the processing of doing one task by more than one processor.

- Resource sharing is the process of sharing information in-between the systems processors. The resource manager of the DOS controls the access, naming scheme, and concurrency.
- A DOS systems are always in openness mode so that a new component can join in the existing DOS.
- The failures in DS are categorized in different sections are: Method Failure, System Failure, Secondary Storage Device or Shared resource Failure, Communication medium failure
- A DS has different modes of failure on the server-side and these are: Byzantine or arbitrary failure, Omission failures, Timing failures, Crash failures .
- A Remote Procedure Calls (RPC) is a technique or protocol which is used by one program to put a request for a service from another program on a network without having the network details.
- In a distributed system, the distributed shared memory (DSM) acts as a resource management technique that does not have any physical memory.
- It creates virtual space that is shared among the nodes of a distributed system.



14.9 FURTHER READING

1. Tanenbaum, A. S. (1995). Distributed operating systems. Pearson Education India.



14.10 ANSWERS TO CHECK YOUR PROGRESS

Ans to Q No 1: Distributed operating system (DOS) is another type of operating system (OS). Generally, OS is a software that manages the hardware resources and other software efficiently.

Ans to Q No 2: The DOS has the following advantages:

a. One system present at one location can access the resources of the other system located at other locations with the help of the resource sharing facility of the DOS.

b. Failure of one system in DOS doesn't affect the other system in the DOS.

Ans to Q No 3: Network Operating system.

Ans to Q No 4: Both local memory and clock

Ans to Q No 5: Simplicity is the major reason for building DOS.

Ans to Q No 6: A Remote Procedure Calls (RPC) is a technique or protocol which is used by one program to put a request for a service from another program on a network without having the network details.

Ans to Q No 7: Inter-process communication (IPC) is a mechanism that allows processes to communicate with each other and synchronize their actions.

Ans to Q No 8: The advantages of RPC are given below.

a. The process-oriented and thread-oriented models are supported by the RPC.

b. The users are unaware of the internal message passing mechanism of RPC.

c. The effort to re-write and re-develop the code is minimum in remote procedure calls.

d. It can be used in a distributed environment as well as the local environment.

Ans to Q No 9: Client

Ans to Q No 10: True

Ans to Q No 11: A remote procedure is uniquely identified by program number, version number, procedure number



14.11 MODEL QUESTIONS

Q 1. What is a distributed operating system?

Q 2. What are the advantages of DOS?

Q 3. What are the disadvantages of DOS?

Q 4. Explain the distributed processing.

Q 5. What do you mean by openness andconcurrency in DOS?

Q 6. What RPC? Explain its advantage and disadvantages.

Q 7. What is DSM? Explain its advantages and disadvantages.

Q 8. Explain the RPC Communication.

*** ***** ***

UNIT 15: INTRODUCTION TO MULTIPROGRAMMING SYSTEM

UNIT STRUCTURE

15.1 Learning Objectives

15.2 Introduction

15.3 Multiprogramming System

15.4 I/O Supervisors

15.5 Memory Management

15.6 File System

15.7 Disk Scheduling

15.8 Let Us Sum Up

15.9 Further Reading

15.10 Answer To Check Your Progress

15.11 Model Questions

15.1 LEARNING OBJECTIVES

After going through this unit, you will be able to –

- define the multiprogramming system.
- describe queue management in multiprogramming system
- explain about I/O Supervisors in multiprogramming system
- describe file system in multiprogramming system.
- describe disk scheduling in multiprogramming system.

15.2 INTRODUCTION

In the previous unit, we learned about about distributed operating systems and distributed processing. We also learned about RPC, shared memory and the different failure models of distributed systems. In this unit, we will learn about multiprogramming system. We will also learn about I/O supervisors, disk scheduling and file system in multiprogramming system. This is the last unit of the block and this course.

15.3 INTRODUCTION TO MULTIPROGRAMMING SYSTEM

A multiprogramming system may run more than one program in a single processor. Different programs run in the processor simultaneously. Though the user may think that the programs are executing at the same time, sharing of the processor is done in a multiprogramming system. The multiprogramming technique increases the utilization of the CPU by scheduling the execution of programs one by one. The figure 15.1 represents the block diagram of the multiprogramming system along with its memory layout.

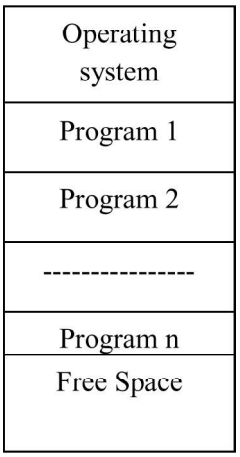


Figure 15.1: Block diagram of multiprogramming system

The following are the features of a multiprogramming system.

- The multiprogramming operating system contains many jobs or programs in memory at a time.
- These set of jobs in a multiprogramming system is a subset of the jobs kept in the job pool.
- The multiprogramming operating system picks and execute one by one job in the memory.
- Multiprogramming operating systems monitor the state of all active programs and system resources.
- The multiprogramming OS never keeps the CPU idle unless there are no jobs to process.

The following are the advantages of a multiprogramming system.

- The CPU utilization of a multi-programming system is very high and efficient.
- Though the user may think that the programs are executing at the same time, sharing of the processor is done in a multiprogramming system.

The disadvantages of multiprogramming system are present below.

- The CPU scheduling is required.
- The I/O supervisor and memory management are very important in multiprogramming.

15.4 I/O SUPERVISORS

The input-output supervisor (IOS) is a control program of IBM mainframe OS/360 that issues the I/O instructions and supervises the I/O interrupts of a job. It has two purposes:-

- To handle I/O requests, that are requests for the execution of channel programs
- To handle I/O interruptions, which result from the execution of channel programs and operator intervention

To handle the above two purposes, the IOS has two primary supervisors.

i) Channel Program supervisor
ii) I/O interrupt supervisor.

These two services are present inside the main memory and support the normal execution of the I/O request and interrupt. The IOS is designed using the multiprogramming concept that manages the I/O channels, control unit, and devices that are executing concurrently. The IOS, in multiprogramming, is a hyper-vising OS built on top of the OS. The hypervisor is responsible for the performing the above purposes.

15.5 MEMORY MANAGEMENT

Memory management in multiprogramming system refers to the management of primary memory of the system. A primary memory or the

main memory contains a large array of bytes that has its address. A multiprogramming operating system is responsible for the following activities for primary memory.

i) Tracking of primary memory. It means to identify the used part and unused part of the memory.
ii) The OS selects the process that will get the memory and how much time it will avail.
iii) Allocate the memory space to the process, upon its request.
iv) De allocates the memory space if the process is no longer present in the main memory or it is terminated.

The process of handling the different processes in multiprogramming OS is known as the degree of multiprogramming. The maximum number of processes that a single processor or OS handles is known as the degree of multiprogramming. To handle or accommodate the process efficiently, memory should be available. The degree of multiprogramming is concerned with the following activities:-

i) In multiprogramming, the CPU is not connected with the secondary memory due to its low speed.
ii) Program or jobs are present in secondary memory and executing the same in CPU one by one after loading in RAM.
iii) In multiprogramming, more processes are placed in CPU from secondary memory to increase the utilization of CPU and also to increase the degree of multiprogramming

The two memory management techniques can be applied in operating systems:-

i) Uni-programming
ii) Multiprogramming

In multiprogramming, especially two methods are present,

i) Contiguous Memory Allocation
ii) Non- Contiguous Memory Allocation

Under the Contiguous Memory Allocation, the following two techniques are available

i) Single Contiguous Memory Allocation
ii) Partition Contiguous Memory Allocation

Under the Non-Contiguous Memory Allocation, the following two techniques are available

i) Page Memory Management
ii) Segmented Memory Management

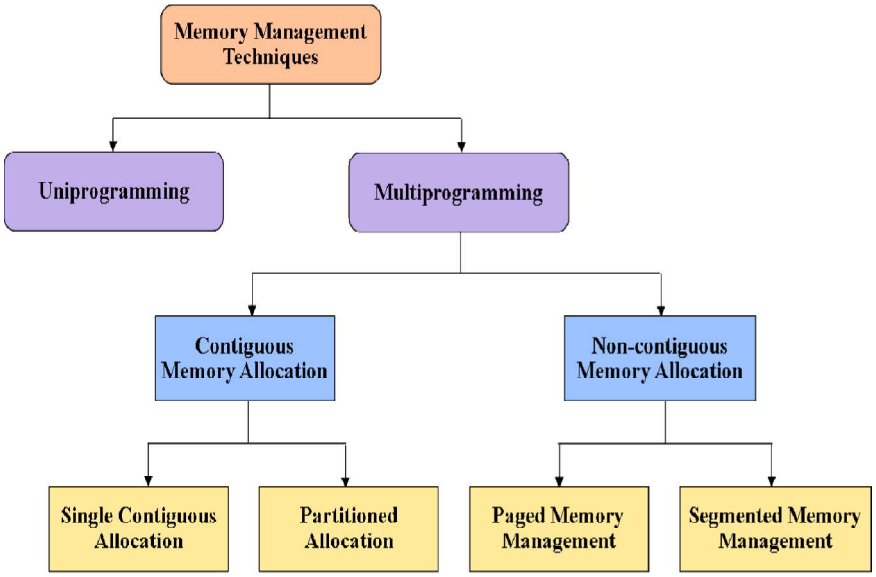


Figure 15.2: Memory Management in Multiprogramming

i) **Contiguous Memory Allocation:** In multiprogramming, the memory spaces are clubbed together in one spot. Here, a single contiguous memory is allocated to the program, for its execution. The OS and the programs reside inside the primary memory. One segment of the primary memory, used for the program, and the other part is used for program operation. The contiguous memory allocation has the following advantages.

- It allocates consecutive blocks of primary memory to a program.
- It is very easy to handle the memory space for OS due to its faster execution.
- Internal fragmentation and external fragmentation are present in contiguous memory allocation method.

- Wastages of memory is there in contiguous memory allocation

The types of contiguous memory allocation are explained below:

a) **Single Contiguous Memory Allocation**: It is a fixed-size segment where memory is divided into fixed-size partions. The sizes of the partition are same. The first partition contains the OS, other contains the programs. Overlapping of partition is not present in the single contiguous memory allocation. Some of the partitions are unused and wasted due to the bigger size of the process or program that is known as internal fragmentation. In fixed partitioning, the size of the partition is fixed, and we cannot vary it according to the process size; therefore, in fixed partitioning, the degree of multiprogramming is less and fixed.

b) **Partitioned Allocation:**It is a variable size partition that is used to overcome the issues present in the fixed-size partition. The size of the partition is known or declared at the time of partition loading. Here, the process size and the partition size are the same, and that reduces the internal fragmentation issues of partition. The problem of internal fragmentation is also reduced by varying the size of the partition based on the needs of the process. In variable size partition, the sizes of the process or program are not fixed, and it can be changed by varying the size of the partition. In variable partition, there is no internal fragmentation and unused space, which can load more number of processes in the memory. So, the degree of multiprogramming is more in variable size partition.

ii) **Non-Contiguous Memory Allocation:** In multiprogramming, the non contiguous memory allocation allocates separate blocks of memory to a file/process. Non-Contiguous memory allocation allocates the memory space present in different locations to the process as per its requirements. The non contiguous memory space is in distributed pattern and scattered. Hence, distribution of memory space to the process as per its requirements. This technique of memory allocation helps to reduce the wastage of memory. The types of non contiguous memory allocation are presented below:-

a) Paged Memory Management:

In the page memory management technique, the primary memories are divided into fixed-size units which are known as page frames. Paging is a memory management technique in which process address space is broken into blocks of the same size called pages. The main memory is divided into fixed-sized blocks, called frames and the size of a frame is kept the same as that of a page to have optimum utilization of the main memory and to avoid external fragmentation. The hardware memory management unit maps pages into frames. It is free of external fragmentation. Here, a computer can address more memory space than a physical memory that is known as virtual memory. The page address is called logical address and is represented by page number and the offset. The frame address is called physical address and is represented by a frame number and the offset.

Logical Address = Page number + page offset

Physical Address = Frame number + page offset

b) Segmented Memory Management: In the segmented memory management technique, each job is divided into several segments of different sizes. Each segment has a different logical address space of the program. During process execution, the corresponding segment is loaded into the non-contiguous memory.



CHECK YOUR PROGRESS

Q 1. What do you mean by multiprogramming OS?

Q 2. State two features of a multiprogramming system.

Q 3. Write two advantages of a multiprogramming system.

Q 4. What are the two primary I/O supervisors?

Q 5. What is the difference between contiguous and non-contiguous memory allocation?

Q 6. What is a logical and physical address?

15.6 FILE SYSTEM

A file system in multiprogramming OS is the collection of related information, that is persistent. A file represents the data and program. A file has the following attributes:-

i) **Name:** A file name is associated with a file.
ii) **Identifier:** A unique number is used to represent the file in the file system.
iii) **Type:** A file type is required to denote the different types of files.
iv) **Location:** A pointer is used to locate the location of a file.
v) **Size:** It is the size of a file.
vi) **Protection:** It defines the access protection of the file.
vii) **Time and Date:** It defines the time and date of file creation and modification information of the file.

The OS issues system calls to create, read, write, and delete the file.

i) Creating a file in the file system needs space for a file in the file system and an entry to the directory of the file system.
ii) Reading a file needs a read pointer to locate the file in the file system and the next read has taken place and the pointer is updated.
iii) Writing to the file needs a write pointer to locate the file and the next write has taken place.
iv) Deleting a file requires locating the directory of the file and then the deletion has taken place.

The file is structured in several ways but three of them are very important.

i) The first important structuring technique is byte sequence. Here, OS takes care of the byte rather than the contents of a file. This structure provides maximum flexibility to the user to write anything in the file and name them accordingly.
ii) The second most important file structure is record sequence. In this technique, the file record is returned in a read operation and append the same in the write operation.

iii) The tree is the third most important file structure technique. A file is structured as a tree with different types of records of varying lengths and key fields. The tree is stored on the key field for searching a particular key.

15.7 DISK SCHEDULING

Disk scheduling is a process where the operating system schedules the I/O requests. Disk scheduling is important because, to manage the multiple I/O requests which may arrive from a different processes. But only one I/O request can be served at a time by the disk controller. In this situation, other I/O request needs to wait in the waiting queue.

The types of disk scheduling algorithms are:

i) First Come First Serve (FCFS) Algorithm
ii) Shortest Seek Time First Scheduling
iii) SCAN Scheduling Algorithm
iv) C-Scan Scheduling Algorithm

Before discussing disk scheduling, you should learn the following parameters.

- **Seek Time:** It is time to locate the disk arm to read or write data.
- **Rotational Latency:** It is the time taken by the sector to rotate itself into a position to access the read and write heads.
- **Transfer Time:** It is the time to transfer the information depending on the speed of the disk.
- **Disk Access Time:** It is the combination of seek time + rotational latency + transfer time.
- **Disk Response Time:** It is the average request waiting time to perform its I/O operation

The **First Come First Serve (FCFS)** is the easiest disk scheduling among all. The first request is always served first in the disk queue. There is no starvation in the algorithm. Let understand the FCFS disk scheduling algorithm with the following examples. Let us consider a disk queue with

request for I/O to blocks on cylinders. Request = {70,3, 2, 40,4,6,90} and the position of the read and write head is 5 and the total number of the track is 100.

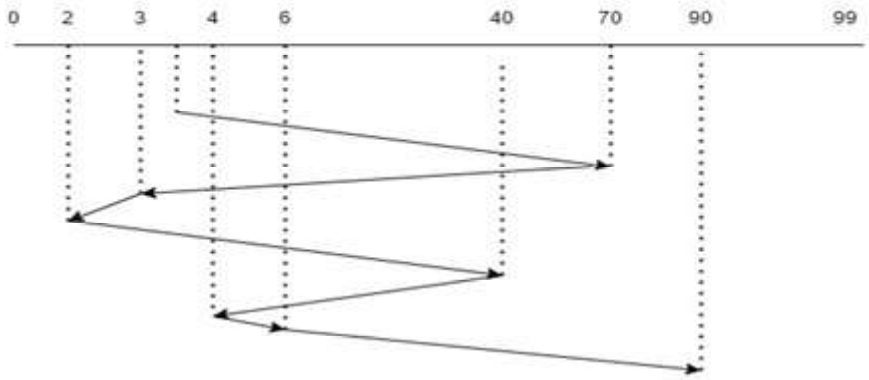


Figure 15.3: FCFS Disk Scheduling

The current position of the read and write head is 5. Starting from 5, the algorithm moves toward 70, then 3, 2, 40, 4, 6, and finally 90. So, the total cylinder move (seek time) is
(70-5) + (70-3) + (3-2) + (40-2) + (40-4) + (6-4) + (90-6) = 303.

The Shortest Seek Time First (SSTF) algorithm considers the shortest seek time request first. So, the average response time is decreased in SSTF. It increases the throughput of the scheduling. There is a chance of starvation in SSTF. Let's understand the algorithm by taking the following examples. Let's you have a disk that contains 100 tracks (0-99). The requests are with the cylinder numbers 70, 3, 2, 40, 4, 90, respectively. Starting from 5, the heads start as follows.

i) If it will from 5 to 4, this will give the shortest seek time among all the requests and that is (5-4) = 1.
ii) Then from 4, the head will move 3.
iii) Then it will move to 2. From 2, it will move to 40, 70, and finally 90.

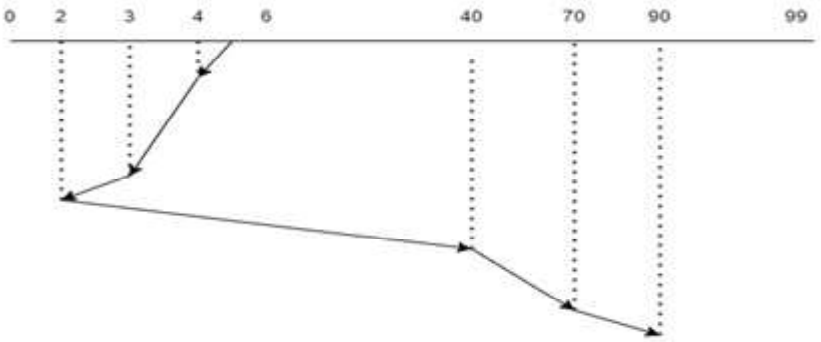


Figure. 15.4: SSTF Disk Scheduling

So, seek time = (5-2) + (90-2) = 91.

In the **SCAN** disk scheduling algorithm, the head starts at one end of the disk and moves towards other end servicing requests. When the disk arm reaches to the other end, it will move in reverse and all the requests are addressed. Consider a disk that contains 100 tracks (0-99). The requests are with the cylinder numbers 70, 3, 2, 40, 4, 90, respectively. The current position of the read/write head is 5. The head will move towards the right end and it will follow.

i) The head movement is towards the right direction from position 5. From 5 it will move to 40, 70, 90 and then 99.

ii) From 99, it will move to 4,3, and finally to 2.

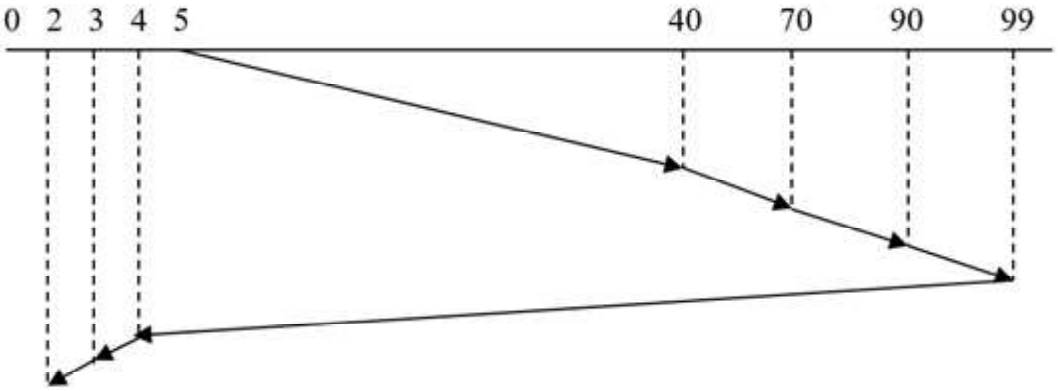


Figure. 15.5: SCAN Disk Scheduling

So, seek time = (99-5) + (99-2) = 191.

In the **C-SCAN algorithm**, the disk moves in a particular direction serving the requests to the end of the disk and then come back to the reverse direction until the end. From that end, the arm starts moving with serving the remaining requests. Consider a disk that contains 100 tracks (0-99). The requests are with the track numbers 70, 3, 2, 40, 4, 90, respectively. The current position of the read/write head is 5. Let's the head will move towards the larger end and it will execute as follows.

i) Let's the arms move in the right direction. So, its first move from 5 to 40, 70, the, 90 and finally it will go to 99.

ii) From 99, it will directly come to the other end, i.e., to the left side end 0.

iii) From 0 it will go to 2, 3, and 4.

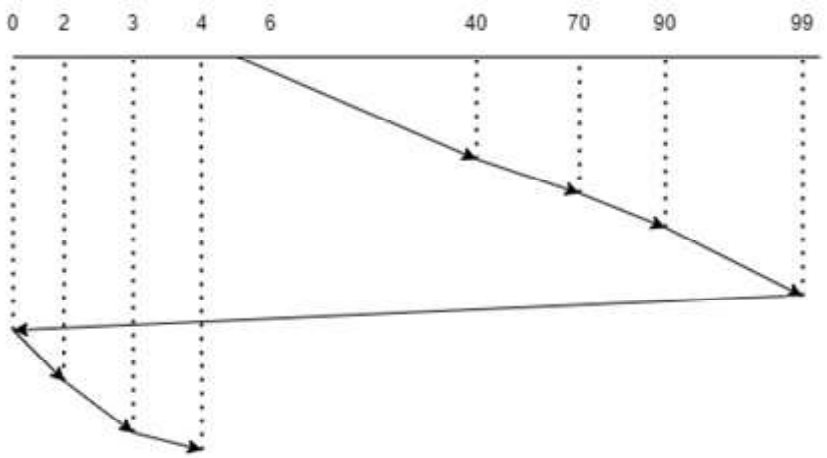


Figure15.6: C – SCAN Disk Scheduling

So, seek time = (99-5) + (99-0) + (4-0) = 19



CHECK YOUR PROGRESS

Q 7. What is a file system?

Q 8. What are the three attributes of the file system?

Q 9. What are the types of disk scheduling algorithms present?

Q10. Write the difference between C-SCAN and SCAN algorithm.

Q11. SSTF increases the throughput (True and False)



15.8 LET US SUM UP

- A multiprogramming system means more than one program in a single processor. Different programs run in the processors imultaneously.
- The multiprogramming operating system picks and executeone by one job in the memory.
- Multiprogramming operating systems monitor the state of all active programs and system resources.
- The CPU utilization of a multi-programming system is very high and efficient.
- Though the user may think that the programs are executing at the same time, sharing of the processor is done in a multiprogramming system.
- The IOS has two primary supervisors: Channel Program supervisor and I/O interrupt supervisor.
- Memory management in multiprogramming system refers to the management of primary memory of the system. A primary memory or the main memory contains a large array of bytes that has its address.
- The two memory management techniques can be applied in operating systems: Uni-programming, Multiprogramming.
- In multiprogramming, especially two methods are present: Contiguous Memory Allocation and Non- Contiguous Memory Allocation.
- Under the Contiguous Memory Allocation, the following two techniques are available: single contiguous memory allocation and partition contiguous memory allocation.
- Under the Non-Contiguous Memory Allocation, the following two techniques are available: page memory management and segmented memory management.

- Logical Address = Page number + page offset
- Physical Address = Frame number + page offset
- A file system in multiprogramming OS is the collection of related information, that is persistent.
- The types of disk scheduling algorithms are:First Come First Serve (FCFS) Algorithm, Shortest Seek Time First Scheduling, SCAN Scheduling Algorithm, C-Scan Scheduling Algorithm.



15.9 FURTHER READING

1. Galvin, P. B., Gagne, G., &Silberschatz, A. (2003). Operating system concepts (Vol. 10). John Wiley & Sons.
2. Hamacher, V. C., Vranesic, Z. G., &Zaky, S. G. (1984). Computer organization. McGraw-Hill, Inc..



15.10 ANSWERS TO CHECK YOUR PROGRESS

Ans to Q No 1: A multiprogramming system means more than one program in a single processor.

Ans to Q No 2: The following are the two features of a multiprogramming system.

i) The multiprogramming operating system contains many jobs or programs in memory at a time.

ii) This set of jobs in a multiprogramming system is a subset of the jobs kept in the job pool.

Ans to Q No 3: The following are the advantages of a multiprogramming system.

i. The CPU utilization of a multi-programming system is very high and efficient.

ii. Though the user may think that the programs are executing at the same time, sharing of the processor is done in a multiprogramming system.

Ans to Q No 4: The two primary IOS supervisors are: Channel Program supervisor and I/O interrupt supervisor

Ans to Q No 5: Contiguous Memory Allocation: In multiprogramming, the memory spaces are clubbed together in one spot. Here, a single contiguous memory is allocated to the program, for its execution. The OS and the programs reside inside the primary memory. One segment of the primary memory, used for the program, and the other part is used for program operation.

Non-Contiguous Memory Allocation: In multiprogramming, the non-Contiguous memory allocation allocates separate blocks of memory to a file/process. Non-Contiguous memory allocation allocates the memory space present in different locations to the process as per its requirements. The non-Contiguous memory space is in distributed pattern and scattered. Hence, distribution of memory space to the process as per its requirements. This technique of memory allocation helps to reduce the wastage of memory.

Ans to Q No 6:

Logical Address = Page number + page offset

Physical Address = Frame number + page offset

Ans to Q No 7: A file system in multiprogramming OS is the collection of related information, that is persistent. A file represents the data and program.

Ans to Q No 8: The three attributes of the file system are

i. Name: A file name is associated with a file.
ii. Identifier: A unique number is used to represent the file in the file system.
iii. Type: A file type is required to denote the different types of files

Ans to Q No 9: The types of disk scheduling algorithms are:

i. First Come First Serve (FCFS) Algorithm
ii. Shortest Seek Time First Scheduling
iii. SCAN Scheduling Algorithm
iv. C-Scan Scheduling Algorithm

Ans to Q No 10: In the **SCAN** disk scheduling algorithm, we move the head either left or right direction. In the moving path, each request is addressed. When the disk arm reaches to end, it will move towards the reverse direction and all the requests are addressed.

In the **C-SCAN algorithm**, the disk moves in a particular direction serving the requests to the end of the direction and then come back to the reverse direction until the end. From that end, the arm starts moving with serving the remaining requests

Ans to Q No 11: True



15.11 MODEL QUESTIONS

Q 1. Q1. What is a multiprogramming system?

Q 2. What are the features of a multiprogramming system?

Q 3. Write the advantage and disadvantages of multiprogramming system.

Q 4. Explain about the I/O supervisor.

Q 5. What is the difference between contiguous and non-contiguous memory allocation?

Q 6. Explain the memory management in a multiprogramming system.

Q 7. What are the attributes of a file system?

Q 8. What is a virtual memory?

Q 9. Explain the First Come First Serve (FCFS) Algorithm with an example.

Q 10. Explain the Shortest Seek Time First Scheduling algorithm.

Q 11. Explain the SCAN Scheduling Algorithm.

Q 12. Explain the C-Scan Scheduling Algorithm.

*** ***** ***