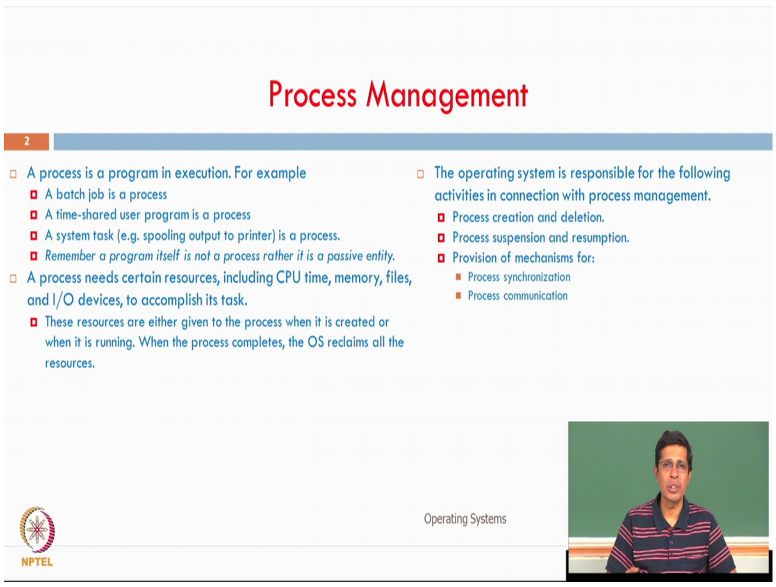
Information Security-3
Prof. V Kamakoti
Department of Computer science and Engineering
Indian Institute of Technology Madras
Basics of Unix and Network Administration
Operating Systems Introduction
Mod01, Lecture 03
Module 3: OS Management Services

Welcome to module 3, in this module we will be looking at OS management services. What is it that the OS need to manage? A OS is a layer which tries to help the CPU and the application software to work and get the required services. So OS is basically a service layer. It services the application software. It satisfies the needs of the application software by getting whatever the application software wants executed on the hardware and so what is it, what sort of services will the application layer need from the hardware and how does the operating system basically handles them. This is very crucial for us to understand, because the understanding of this will help us in understanding the information security part of building an operating system. Many of the vulnerabilities come, because these interfaces are not well understood either by the application software or by the operating system module and because of this lacuna or whatever a gap in understanding the functionalities, the vulnerabilities basically (())(1:25).

(Refer Slide Time: 1:26)


Process Management

2

- A process is a program in execution. For example
  - A batch job is a process
  - A time-shared user program is a process
  - A system task (e.g. spooling output to printer) is a process.
  - *Remember a program itself is not a process rather it is a passive entity.*
- A process needs certain resources, including CPU time, memory, files, and I/O devices, to accomplish its task.
  - These resources are either given to the process when it is created or when it is running. When the process completes, the OS reclaims all the resources.
- The operating system is responsible for the following activities in connection with process management.
  - Process creation and deletion.
  - Process suspension and resumption.
  - Provision of mechanisms for:
    - Process synchronization
    - Process communication

NPTEL

Operating Systems

Now let us understand the first most important functionality of an operating system is process management. What is a process; already I said a process is a program in execution. There can be variety of processes, what you see on a mobile phone is actually at an interactive process.

It is actually a timeshared process. A process can also be that I submit and sometime later it gets executed. This is called a batch job. So you submit it and tomorrow morning you come and see whether it is done. This the basically happens on large supercomputing, high performance computing environments where you do not talk to the system that is executing a program one to one like, how do you talk to your laptop or your mobile phone or your dextop, but you actually submit the program and over a period of time the program actually gets executed and say within some reasonable amount of time the results come up came.

So this is basically called a batch job. A timeshared job is something that we do here where there are many many processes executing on a system and your CPU is timeshared between multiple of them and then there are other type of processes, which we need to understand here is called a system task. A system task is different from a user task, when I execute a program then it is a user program, but when we go and do some important functionalities, for example, I go and say printf or scanf then it basically calls the system for help, because I as say for example I as a hallo world program, we will not know how to go and print on the screen, because I do not know what screen is there.

So I basically use that printf routine and that printf routine essentially calls the operating system, which in turn calls the graphics adapter and there is a sequence of calls that go before the hallo world appears on the screen, right. So this is very very important. So when we look at process, there could be timeshared interactive processes, there could be batch processes and more importantly, there is another category of process is called system task, which works at higher privilege level.

Now when we want to execute a process again, I have covered this in great detail in the information security 2 course. When I want to execute a process, I go and ask the operating system say, I am a process, I have say 1 megabyte of code and I need say, half megabyte of data, I have so many arrays, I have so many things. So give me this storage. So first the operating system has to give you some memory where the process can be loaded, the data etcetera can be loaded, after that we start executing this.

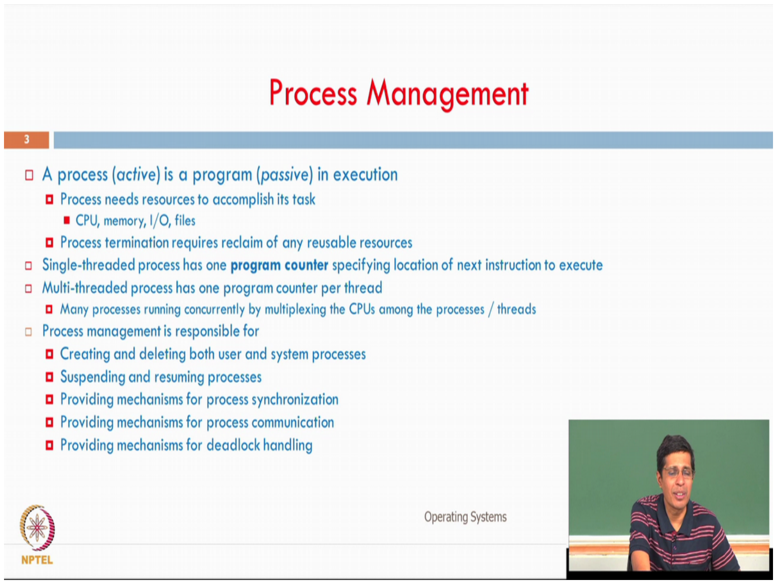
So a particular a every process will need certain resources in terms of CPU time, because I need to execute the program, I need some amount of memory then I will write to do some IO into (())(4:22) like printf and scanf from the keyboard and printf on the screen and I will also try and do lot of fprintf and fscanf, I am using C language to basically explain this , I hope all of you know C, if you do not know just get to know these commands. So I use fscanf and fprintf to read and write from files.

So as a process I need lot of resources including CPU time, memory files, IO devices to complete my task or accomplish my task and these resources are made available to the process through the operating system and whenever the process completes then it returns back the resources to the operating system. So the operating system one of the important aspects of process management is for the operating system to provide CPU time to provide memory files, IO devices etcetera.

The second important process management is to actually monitor and you know control the execution of a process when you say suppose we compile C program and say, dot slash a dot (())(5:28) press the enter. Now a process has to be created when process completes, it needs to be deleted. So there something called a process creation and deletion and we can also suspend a process and make it wait. So process suspension and once a suspended process I want to bring it back, I can again revoke it back or what you call as process resumption. So this is also operating system functionality while I mentioned earlier why should a process get suspended? A process actually get suspended, because it is waiting for an IO operation to complete and why should it get resume, after the (())(6:02) when will it get resumed after the IO completes, it can basically resume.

So there is a provision of mechanisms for process synchronization and process communication that is very very important. What you mean by process synchronization I want to perform a synchronous IO. This is an example for a process synchronization and one process wants to communicate to another process. So I could have inter-process communication and we even within a process, one sub-routine wants to pass information to another sub-routine; there is also intra-process communication that happens between different modules of a given program.

(Refer Slide Time: 6:39)



Process Management

3

- A process (*active*) is a program (*passive*) in execution
  - Process needs resources to accomplish its task
    - CPU, memory, I/O, files
  - Process termination requires reclaim of any reusable resources
- Single-threaded process has one **program counter** specifying location of next instruction to execute
- Multi-threaded process has one program counter per thread
  - Many processes running concurrently by multiplexing the CPUs among the processes / threads
- Process management is responsible for
  - Creating and deleting both user and system processes
  - Suspending and resuming processes
  - Providing mechanisms for process synchronization
  - Providing mechanisms for process communication
  - Providing mechanisms for deadlock handling

Operating Systems

NPTEL

Now we are also talking of shearing CPUs. What you mean by shearing a CPU? Today programs can also be multithreaded. Thread is nothing but unit of execution. So instead of one unit of execution running, I could have four units of execution running, right. So this is called a multi-threaded program where each thread can execute concurrently without depending on the another thread. So if I have some certain operations that could be concurrently executed, I go to a multi-threaded system and say suppose I have four different operations that could execute concurrently, I go to a multi-threaded system, which has four threads and give one operation each to one of these threads and all the four can happen concurrently. If we do not have a threaded architecture then these four operations has to happen one after another.

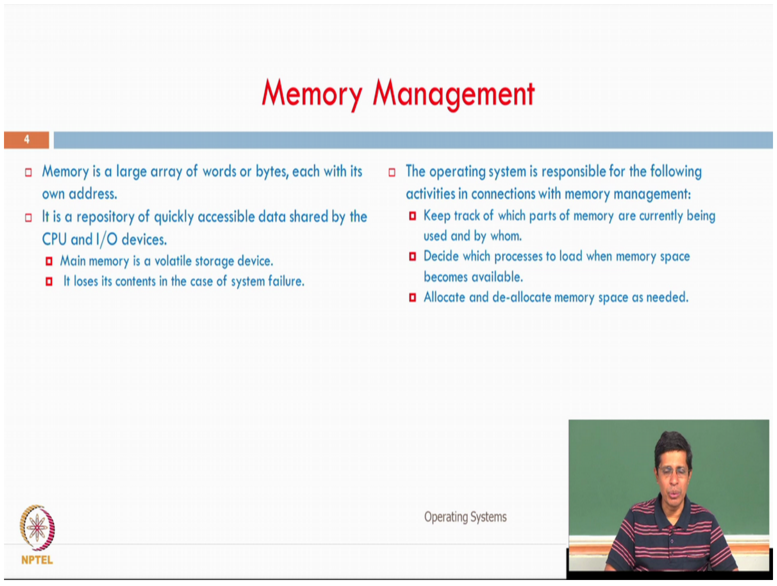
So a multi-threaded architecture gives us very very good speed up or throughput. So if I have a single threaded process in which I am using only one thread then there will be one program counter, which basically specifies a location of the next instruction, but if I have multi-threaded process, each of the multi-threaded process has one program counter each thread has a program counter. So now when I look at a multi-threaded architecture. An architecture that could execute more than one thread at a time. Now the operating systems responsibility is to see that these program counters are handle properly and you know the processes are moving in a dought (())(8:08) properly. So there is a lot of difference between having a single threaded system verses a multi-threaded system and the process management accordingly changes.

Now many of the security vulnerabilities come by the way why different threads actually interact with each other and that needs to also be taken care of. So when you look at the information security 2 course, there we have introduce the notion of what we called as call gate (())(8:35) towards the end of these those lectures, which basically talks about multi-layer security. So and many of these vulnerabilities actually happen when we switch from one mode or one unit of execution to another unit of execution. It can be from one program to another program and when we do this switching, the isolation between these two programs is not so perfect and because of that lack of isolation some information of program one when it switches to program when the CPU switches from program one to program two, some information of program one can be leaked on to program two and that forms a very very important aspect of information security.

So to finally conclude process management is responsible for creating and deleting both user and system processes. So now we have making a distinction between what is a user process, for example, the hallo world program is a user process, but the printf and the subsequent execution that is happening important a system process. So that is how we can distinguish very quickly between the user and system processes and the operating system is also responsible for suspending and resuming processes, it also provides mechanisms for process synchronization. It will also provide mechanisms for process communication and it also use useful for handling what we called as deadlock. What is a dead lock?

Deadlock is nothing but a process P1 is waiting for a resource R1 which is currently held by process P2, which in turn is waiting for a resource R2, which is held by process P1. So P1 wants R1 which is held by P2 which in needs R2 which is held by P1. So for P1 to complete it needs R1, it has R2 with its and for P2 to complete it needs R2 and it has R1 with it. So each of them are competing and so this is basically what we called as say deadlock and there are the process (())(10:37) operating system should handle deadlocks.

(Refer Slide Time: 10:40)



Memory Management

4

- Memory is a large array of words or bytes, each with its own address.
- It is a repository of quickly accessible data shared by the CPU and I/O devices.
  - Main memory is a volatile storage device.
  - It loses its contents in the case of system failure.
- The operating system is responsible for the following activities in connections with memory management:
  - Keep track of which parts of memory are currently being used and by whom.
  - Decide which processes to load when memory space becomes available.
  - Allocate and de-allocate memory space as needed.

Operating Systems

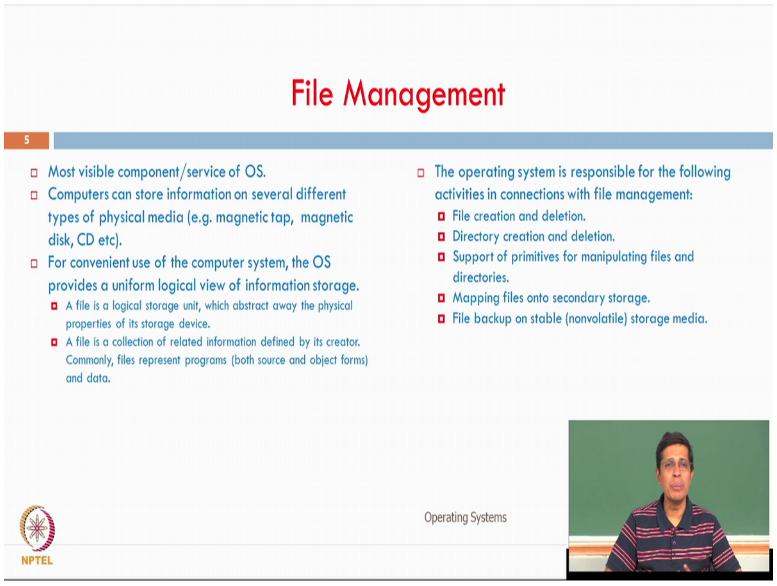
The next important aspect of an operating system is to basically handle memory. Memory is a very large array of words each with its own address and what do we store in memory? We store lot of data that are basically sheared within data used by the CPU and also, basically that are sheared between the CPU and the IO devices. We already mention the Notion of what we called as direct memory access, wherein a peripheral can write directly into a memory location and then the CPU can later read from that memory location.

So when we look at the operating systems role in memory management, it has to provide the necessary memory that is needed by a particular process and in the case where we have a notion virtual memory, the operating system can go and it has to go and setup certain paging related or whatever the virtual memory related stuff and then it has to basically give those memory to the respective process. So memory management is also covered a specially the virtual memory setting up page tables etcetera and a segmentation. These are all covered, there are very interesting lab exercises that have been given as part of information security 2 course, I advise you to go and look at that in greater detail.

So the operating system is responsible for giving memory to a process and it is also responsible for getting back that memory and releasing it properly so that it can be used by another process at some other later point of time. So space or memory management is vary very important and there are two aspects of memory management, one is memory allocation and the release of memory and handling, which part of the memory is allocated, which is not allocated etcetera. The second part of memory and management is also to look at provision a virtual memory. So every program assumes that it has 4GB RAM, but I could have only 4GB RAM in the system and there could be 100s of processes that are shearing this.

So a notion virtual memory basically helps you in realizing that all the memory is given to a particular process in terms of virtually. So there will be 100 process, but each process thinks that the entire 4GB address spaces for it, which will not necessarily be the case, but the operating system will give you that virtual feeling and importantly when a process completes or when its resource is released, it has to be properly documented or it has to be properly maintained so that it ((())(13:08) and it could be reused at a later stage.

(Refer Slide Time: 13:12)



File Management

5

- Most visible component/service of OS.
- Computers can store information on several different types of physical media (e.g. magnetic tap, magnetic disk, CD etc).
- For convenient use of the computer system, the OS provides a uniform logical view of information storage.
  - A file is a logical storage unit, which abstract away the physical properties of its storage device.
  - A file is a collection of related information defined by its creator. Commonly, files represent programs (both source and object forms) and data.
- The operating system is responsible for the following activities in connections with file management:
  - File creation and deletion.
  - Directory creation and deletion.
  - Support of primitives for manipulating files and directories.
  - Mapping files onto secondary storage.
  - File backup on stable (nonvolatile) storage media.

Operating Systems

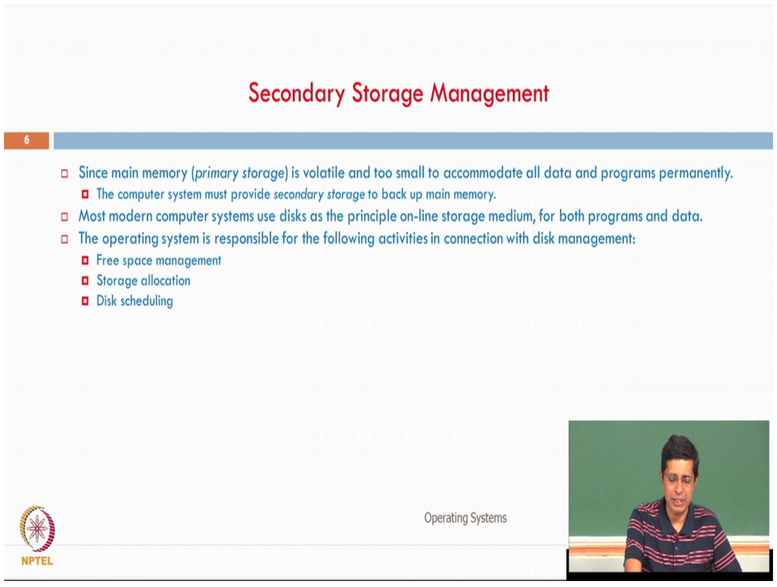
NPTEL

Note that a file is the most visible component of operating system, anything that I store the basic unit of information is called file and is a actually a logical storage unit and file gives you an abstraction. So today I created this ppt file in another dextop then I sent it through an email, somebody copied it and they have stored it in a dextop and now they have brought it here. So this file is called module 3 dot ppt, right, but is the same module 3 dot ppt was there, in my disc then it was there on my mail server then it came to the dextop of Benjamin here. Benjamin is the staff, who is helping me in recording this lecture and then from there he transform throughput (())(13:56) into this laptop.

So but the file has remain the same, but the way it is getting stored, it got stored in my dextop, which is a Mack apple system and then the way it got stored in the mail server we do not know how the mail server is configured (())(14:14) some other disc, the way it has got stored in the studio and the way it is stored on the laptop, there all different discs, different formats, but a file remains as a file from the user perspective, I just see module 3 dot ppt everywhere right.

So that is very very important and who takes care of this uniform view for the user though, the way it is stored is different, the operating system uses that. So that is very very important here and the operating system is responsible for the following activities in connection with file management, creating a file, deleting a file, creating a directory, deleting a directory and support of primitives, who are manipulating files and directories right, deleting a files changing some permissions etcetera and then the most important thing is we need to map the files on to the secondary storage and then we have to do file backup on stable non-volatile storage media so that for our cable (())(15:11) purpose and retrieval purpose in case there is a power break down etcetera. So these are all very important aspects of file management.

(Refer Slide Time: 15:20)



Secondary Storage Management

6

- Since main memory (*primary storage*) is volatile and too small to accommodate all data and programs permanently.
  - The computer system must provide *secondary storage* to back up main memory.
- Most modern computer systems use disks as the principle on-line storage medium, for both programs and data.
- The operating system is responsible for the following activities in connection with disk management:
  - Free space management
  - Storage allocation
  - Disk scheduling

Operating Systems

NPTEL

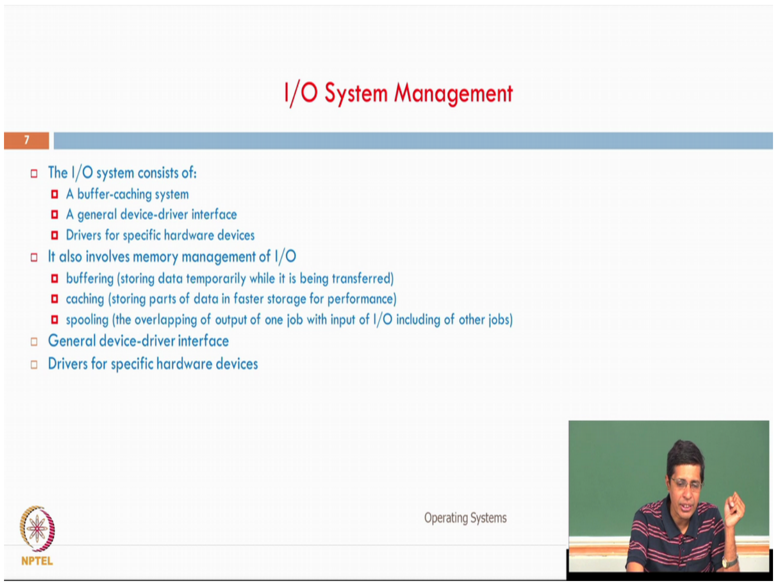
In addition there is something call the secondary storage management. What is a secondary storage? The main storage is call or whatever called as a primary storage is basically the main memory. A secondary storage is a system is the hard disc for example or it can be any even online storage medium today, I could go drop box etcetera. So these are all secondary storage for us. So one of the important thing with respective a disc, for example, the operating system is responsible for the following activities in connection with managing a disc for example, an hard disc.

Now the operating system has to maintain in a (())(15:11) completing about the free space that is available there. So I want to store a file, I should know which space is free in the disc.

So this is what we call as free space management and then I have to there are so many free space I have to allocate a particular space that is called storage allocation and then every time I read or write from the disc.

Now this today we are looking at SSDs or solid state devices, but in olden days when they look at you know mechanical discs, right then the head will be at some point and if there are 10 files, I would like to take that file, which is very close to the head (())(16:32) so that quickly I can write it there. So I will decide on which file to be written at which point of time, I in the sense the operating system. So the disc scheduling also becomes a very very important part of secondary storage management by the operating system.

(Refer Slide Time: 16:48)



I/O System Management

7

- The I/O system consists of:
  - A buffer-caching system
  - A general device-driver interface
  - Drivers for specific hardware devices
- It also involves memory management of I/O
  - buffering (storing data temporarily while it is being transferred)
  - caching (storing parts of data in faster storage for performance)
  - spooling (the overlapping of output of one job with input of I/O including of other jobs)
- General device-driver interface
- Drivers for specific hardware devices

NPTEL

Operating Systems

Now there something called IO system management, IO system management is that I have an IO device that is trying to communicate with the CPU and how do we go about doing this. So what is it for me to read from an IO device, first thing is I need to understand how I can talk with the IO device, I means the CPU. Now there are 100s of IO devices in the market and all of them are compatible with the current you know the windows or the Unix operating system. They are compatible with it.

Now does this compatibility come? How are you able to access n number of peripherals here, the reason is that there is a driver a software, which talks to the corresponding device controller told you in the previous module that every peripheral will come with a peripheral controller, disc comes with a controller, graphics card comes with a graphics card controller.

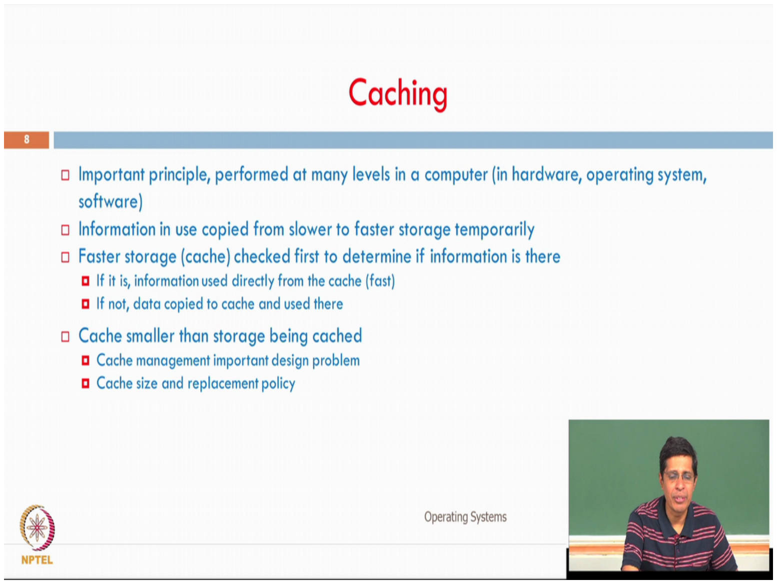
Now there is a peripheral and there is a hardware, which controls that peripheral and that hardware is interface to the CPU through the bus.

Now what is a next step here? Now, the next step here is that somebody has to talk to that controller and who is that somebody? It is the operating system, which part of the operating system that is called a device driver. So the moment you inserts some device then immediately into your system either say, windows or Linux then immediately says we are trying to find out and install a device driver. So the device driver talks to the device controller and then intern it will go and access the device controller will access the device and do the necessary job. So and please note the IO system is much much much slower than the actual CPU and so what we do is that as an when we read from that we buffers the inputs and the outputs if we are going to write into a location. So we do that buffering to actually match the speed right.

So when we look at IO systems one of the most important things is to have a buffer caching system. Now a buffer caching system basically brings from disc some data and it is store in the memory and it keeps as a cache. Now improper handling of the security of the buffer cache can essentially lead to a information leakage, right as a process (())(19:13) would have some confidential data on the disc, which is basically taken up and put it in the buffer cache and then the process I completes, but the buffer cache is not erased, it is just an free less (()) (19:24) and so the next process can come and look at this buffer cache and try to get lot more information about the password etcetera or anything there.

So handling this buffer caching system is very important from a security perspective and there is also something called memory management of IO and there looking at you know the caching looking at, what you call as pulling which is nothing but the overlapping of output of one job with the input of IO including of others job. So this is also very very important. So the memory management of IO, the IO system itself is very very crucial for the operating system and so what the operating system today provides for handling IO system, it is general device driver interface and I already explain what a device driver is and we also need these drivers for multiple type of hardware’s that we use. So an operating system why we need a generic device driver interface, because we have 1000s of peripherals that we could attach to the different systems and so the drivers need to have a generic device driver interface.

(Refer Slide Time: 20:35)



Caching

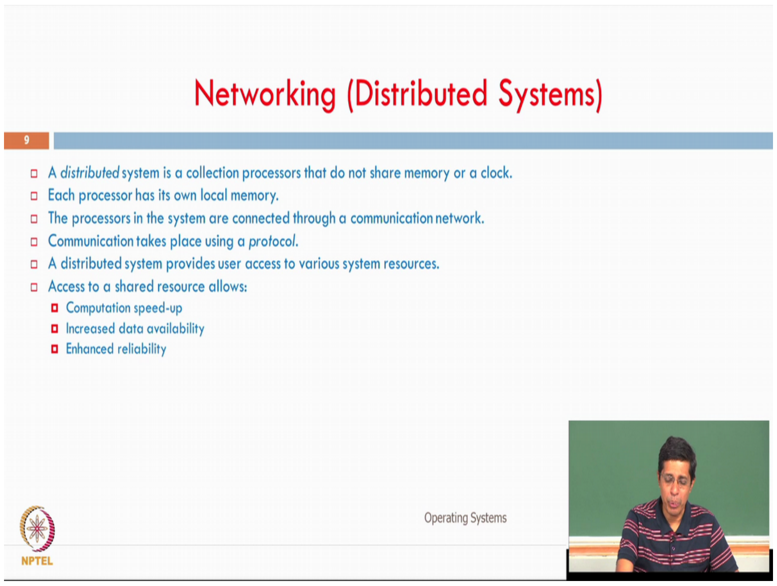
- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storage temporarily
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly from the cache (fast)
  - If not, data copied to cache and used there
- Cache smaller than storage being cached
  - Cache management important design problem
  - Cache size and replacement policy

Now we have already explained what is caching, but I will just quickly touch here, caching is nothing but between two levels of storage in the storage hierarchy, the slower device when it tries to interface to the faster device as caching mechanism built in the faster device so that every time, I want to go and access a data I need not go to the slower device, but I could have it from the faster device. So this is the notion of what we called as cache and the notion of cache actually works, because programmers when they write a program implicitly follow, it is not intentional, but it implicitly follows two important things called locality reference, one is called special locality another is called temporary locality.

What is special locality? if I address data if I go and access data or instruction at address A within the next few milliseconds, I will access something at the path (())(21:33) A plus delta A. So when I am fetching address A, I will also fetch all the things around it so that next time I need not go to the disc, but I can take it from the cache. Bring it means bring it where bring and store it in the cache and I could access the cache at a later time. The another locality of reference call temporary locality, which basically talks about I access a data at time t, the probability that I will be accessing the data at time t plus delta t is very high, for example, I initialize a variable before I use it.

So initialization is one access to the variable and immediately after that I start using that variable I write into the variable or read from that variable. So this is an example so the way a programmers mind work automatically resolves into what we called as the temporary (()) (22:17) and special locality of reference. So when I access an address A at time t probability that I will access A at time t plus delta t is very high. The probability that I will access an address very close to A in the next delta t time is also a very high and this makes caching very very important aspect.

(Refer Slide Time: 22:37)



Networking (Distributed Systems)

- A *distributed* system is a collection processors that do not share memory or a clock.
- Each processor has its own local memory.
- The processors in the system are connected through a communication network.
- Communication takes place using a *protocol.*
- A distributed system provides user access to various system resources.
- Access to a shared resource allows:
  - Computation speed-up
  - Increased data availability
  - Enhanced reliability

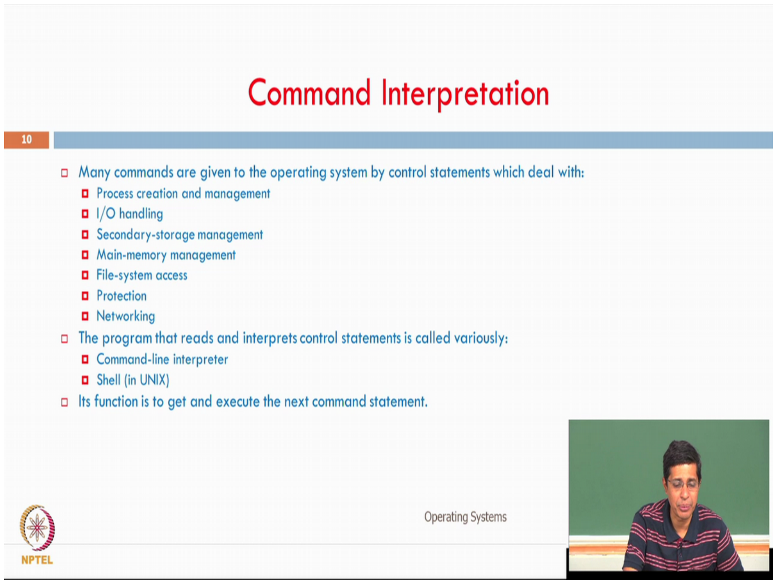
Operating Systems

The next is next aspect of operating system that we need to talk about is the distributed system. What do you mean by distributed system? You have multiple computers connected there is nothing that they shear except that they are connected through a cable or what we call as say, communication network. Now we need to program this. So let me say, I have 1000s of such computers networked and now I have a job, which has thousand parts and each of these parts can execute concurrently.

Now I need a mechanism by which I can take my job and split it up into those 1000s small parts and allocate one part each to one of these process, they execute and then they communicate and then they solve the problem. So this is basically called a distributed system and the most crucial thing for a distributed system is to have a very high speed interconnection network by which one system can talk to another.

So networking actually comes in here, in with the great detail (())(23:34). So networking is not to connect two remote systems, but today networking plays a crucial role in building up many of the super computers, wherein one processor gets connected to another processor to a network stack. So the notion of distributed systems is very very important.

(Refer Slide Time: 23:51)



Command Interpretation

10

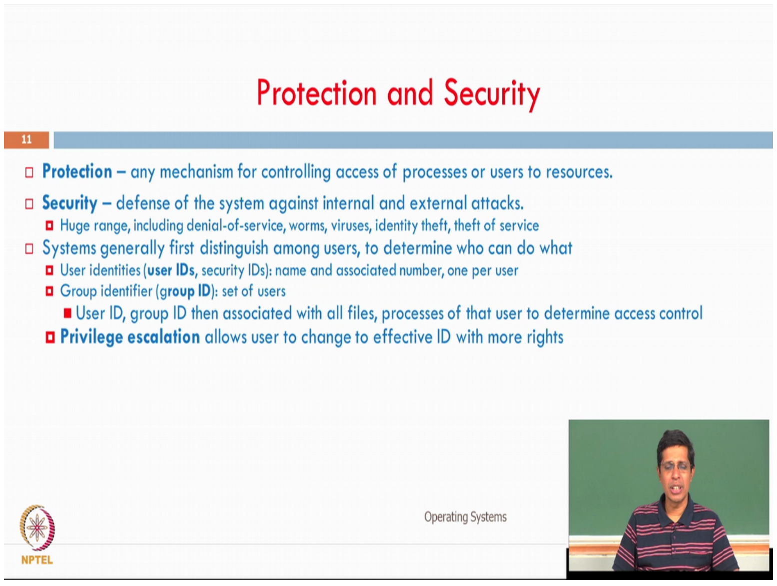
- Many commands are given to the operating system by control statements which deal with:
  - Process creation and management
  - I/O handling
  - Secondary-storage management
  - Main-memory management
  - File-system access
  - Protection
  - Networking
- The program that reads and interprets control statements is called variously:
  - Command-line interpreter
  - Shell (in UNIX)
- Its function is to get and execute the next command statement.

Operating Systems

The other thing is that I need to manage the operating system by giving command. So lot of people would have known command prompt is as a part of this course as we proceed, we will also talk much more about commands, the terminal where many commands will also be taught as a part of this course. Now, what is a command? A command is basically that to create a file, to delete a file, to do IO or you know to access main memory, to access secondary storage system, to create some poor protection in environment, I can also do certain basically execute certain network commands through the command interpretation. So there is a command line interpreter both for Unix which is called a shell a batch shell etcetera and for windows, there is a command always.

So this command is basically an interpretation. What is an interpretation? So I have say 10 commands, an interpreter will take one command execute it then take the next command the next command and execute it. A compiler will compile all the 10 commands and execute it in one short. So there is a difference between an interpreter and a compiler. So the command is always interpreter, in sense one command is taken, it is executed, then the next command then the next command that why we call it as command interpretation here. We strongly argue (())(25:02) to go and look at the command in a windows or the Linux system, terminal in the Linux system and understand some of the popular commands or very effective or very heavily used commands in the respective operating systems.

(Refer Slide Time: 25:17)



Protection and Security

11

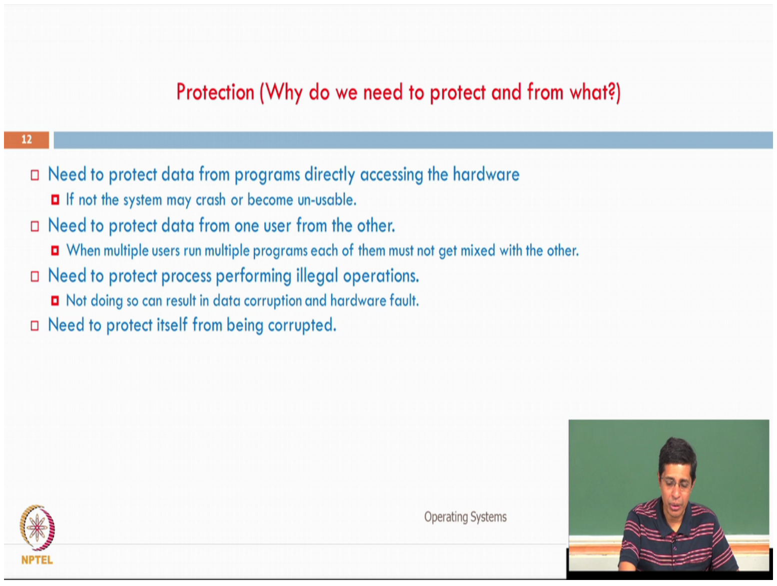
- **Protection –** any mechanism for controlling access of processes or users to resources.
- **Security –** defense of the system against internal and external attacks.
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
  - User identities (**user IDs**, security IDs): name and associated number, one per user
  - Group identifier (**group ID**): set of users
    - User ID, group ID then associated with all files, processes of that user to determine access control
  - **Privilege escalation** allows user to change to effective ID with more rights

Operating Systems

Now we will also define what do you mean by protection and what you mean by security? Protection is something, which is very very internal to the system. We will talk about protection in more detail as we go into the next module, but as far as this module is concern please, understand that within a system I need to protect lot of things. What are the things we will see in the next module but some sort of ensuring security within the system is called protection. Protection mechanism involved to protect a system from outside that is basically call security. Security is the defense of the system against internal and external attacks, okay. So the internal attack is somebody comes and hacks a password internally and then you can (())(25:58) and an external attack would be something like denial-of-service, can be worms, viruses etcetera.

So the protection and security is basically employed at three different levels, first if we take the operating system the Unix operating system, it has two levels of protection. First there is an user ID then there is an user ID and the group ID put together will give you another protection. So for a particular file there be a owner who has some permissions for it then that owner can become belong to a group and that any member of that group will have some permission on it and then you will have anybody else. So for every entity there is a way by which I could create certain access rights for certain people or certain users by which I can ensure both protection and security. So protection is internal threads between the modules and security is external thread to the system.

(Refer Slide Time: 26:55)



Protection (Why do we need to protect and from what?)

12

- Need to protect data from programs directly accessing the hardware
  - If not the system may crash or become un-usable.
- Need to protect data from one user from the other.
  - When multiple users run multiple programs each of them must not get mixed with the other.
- Need to protect process performing illegal operations.
  - Not doing so can result in data corruption and hardware fault.
- Need to protect itself from being corrupted.

Operating Systems

NPTEL

Why do we need to protect? I need to protect, because today let us take a mail server as big as (())(27:00) say, Google server gmail server. Now if I go and do some fault on that and the entire server stops, it essentially means there is denial-of-service to the entire mail users, say millions or billions of users of Gmail, right. So **so** that is very very important. So one of the in thing is that I need to protect my data from other programs which are trying to access it. I need to protect processes from doing some illegal operation which will go and create (()) (27:31) for the other processes.

So there is a need to protect the program itself and there is also a need to protect this program from going and doing something, which it should not do in the contest of another process. So there is a intra-process protection and then inter-process protection that we need to take around and this is very very important from an information security point of view. So we will continue about security measures (())(27:56) in the next module. Thank you.