



# Linux Experts

A Fully loaded blog for Linux Admins, Beginners, Linux Interview Questions, Linux Configurations..

[Home](#)[Linux Interview Questions](#)[Linux Basics](#)[Unix](#)

## Linux Interview Questions : L1 & L2

### 1) What are the process states in Unix?

As a process executes it changes state according to its circumstances. Unix processes have the following states:

- Running** : The process is either running or it is ready to run .
- Waiting** : The process is waiting for an event or for a resource.
- Stopped** : The process has been stopped, usually by receiving a signal.
- Zombie** : The process is dead but have not been removed from the process table.

### 2) What Happens when you execute a program?

When you execute a program on your UNIX system, the system creates a special environment for that program. This environment contains everything needed for the system to run the program as if no other program were running on the system. Each process has process context, which is everything that is unique about the state of the program you are currently running. Every time you execute a program the UNIX system does a fork, which performs a series of operations to create a process context and then execute your program


Total Pageviews



52656

Feedjit

#### Live Traffic Feed

A visitor from **Newark, New Jersey**   
view ed "[Linux Experts: Linux Interview Questions : L1 & L2](#)" 0 secs ago

A visitor from **Saint-maur, Ile-de-France**   
view ed "[Linux Experts: Linux Interview Questions : L1 & L2](#)" 1 min ago

A visitor from **Savigny-le-temple, Ile-de-France**   
view ed "[Linux Experts: Configuring A High Availability Cluster \(Heartbeat\) On CentOS](#)" 7 hrs 14 mins ago

in that context. The steps include the following: Allocate a slot in the process table, a list of currently running programs kept by UNIX. Assign a unique process identifier (PID) to the process. iCopy the context of the parent, the process that requested the spawning of the new process. Return the new PID to the parent process. This enables the parent process to examine or control the process directly. After the fork is complete, UNIX runs your program.

### 3) What Happens when you execute a command?

When you enter 'ls' command to look at the contents of your current working directory, UNIX does a series of things to create an environment for ls and then run it: The shell has UNIX perform a fork. This creates a new process that the shell will use to run the ls program. The shell has UNIX perform an exec of the ls program. This replaces the shell program and data with the program and data for ls and then starts running that new program. The ls program is loaded into the new process context, replacing the text and data of the shell. The ls program performs its task, listing the contents of the current directory.

### 4) What is a Daemon?

A daemon is a process that detaches itself from the terminal and runs, disconnected, in the background, waiting for requests and responding to them. It can also be defined as the background process that does not belong to a terminal session. Many system functions are commonly performed by daemons, including the sendmail daemon, which handles mail, and the NNTP daemon, which handles USENET news. Many other daemons may exist. Some of the most common daemons are: init: Takes over the basic running of the system when the kernel has finished the boot process. inetd: Responsible for starting network services that do not have their own stand-alone daemons. For example, inetd usually takes care of incoming rlogin, telnet, and ftp connections. cron: Responsible for running repetitive tasks on a regular schedule.

### 5) What is 'ps' command for?

The ps command prints the process status for some or all of the running processes. The information given are the process identification number (PID), the amount of time that the process has taken to execute so far etc.


### 6) How would you kill a process?

#### Maharashtra


view ed "Linux Experts: Linux Interview Questions : L1 &L2" 7 hrs 49 mins ago

A visitor from **Jaipur, Rajasthan** 


view ed "Linux Experts: Linux Interview Questions : L1 &L2" 8 hrs 47 mins ago

A visitor from **London, London, City of** 

view ed "Linux Experts: Linux Interview Questions : L1 &L2" 10 hrs 56 mins ago

A visitor from **Mumbai, Maharashtra** 

view ed "Linux Experts: Linux Interview Questions : L1 &L2" 19 hrs 23 mins ago

A visitor from **Mumbai, Maharashtra** 

view ed "Linux Experts: Linux Interview Questions" 1 day 5 hours ago

A visitor from **London, London, City of** 

view ed "Linux Experts: Linux Interview Questions : L1 &L2" 1 day 7 hours ago

A visitor from **Mumbai, Maharashtra** 

view ed "Linux Experts: Configuring A High Availability Cluster (Heartbeat) On CentOS" 1 day 7 hours ago

Real-time view · Get Feedjit

The kill command takes the PID as one argument; this identifies which process to terminate. The PID of a process can be got using 'ps' command.

### 7) What is an advantage of executing a process in background?

The most common reason to put a process in the background is to allow you to do something else interactively without waiting for the process to complete. At the end of the command you add the special background symbol, &. This symbol tells your shell to execute the given command in the background. Example: `cp *.* ../backup&` (cp is for copy)

### 8) How do you execute one program from within another?

The system calls used for low-level process creation are `execlp()` and `execvp()`. The `execlp` call overlays the existing program with the new one, runs that and exits. The original program gets back control only when an error occurs. `execlp(path, file_name, arguments...);` //last argument must be NULL. A variant of `execlp` called `execvp` is used when the number of arguments is not known in advance. `execvp(path, argument_array);` //argument array should be terminated by NULL

### 9) What is IPC? What are the various schemes available?

The term IPC (Inter-Process Communication) describes various ways by which different processes running on some operating system communicate between each other. Various schemes available are as follows:

- Pipes:** One-way communication scheme through which different processes can communicate. The problem is that the two processes should have a common ancestor (parent-child relationship). However, this problem was fixed with the introduction of named-pipes (FIFO).
- Message Queues:** Message queues can be used between related and unrelated processes running on a machine.
- Shared Memory:** This is the fastest of all IPC schemes. The memory to be shared is mapped into the address space of the processes (that are sharing). The speed achieved is attributed to the fact that there is no kernel involvement. But this scheme needs synchronization. Various forms of synchronisation are mutexes, condition-variables, read-write locks, record-locks, and semaphores.

### 10) What is the difference between Swapping and Paging?

**Swapping** : Whole process is moved from the swap device to the main memory for execution. Process size must be less than or equal to the available main memory. It is easier to implementation and overhead to the system. Swapping systems does not handle the memory more flexibly as compared to the paging systems.

**Paging** : Only the required memory pages are moved to main memory from the swap device for execution. Process size does not matter. Gives the concept of the virtual memory. It provides greater flexibility in mapping the virtual address space into the physical memory of the machine. Allows more number of processes to fit in the main memory simultaneously. Allows the greater process size than the available physical memory. Demand paging systems handle the memory more flexibly.

#### 11) What is major difference between the Historic Unix and the new BSD release of Unix System V in terms of Memory Management?

**Historic Unix uses Swapping** – entire process is transferred to the main memory from the swap device, whereas the

**Unix System V uses Demand Paging** – only the part of the process is moved to the main memory. Historic Unix uses one Swap Device and Unix System V allow multiple Swap Devices.

#### 12) What is the main goal of the Memory Management?

It decides which process should reside in the main memory, Manages the parts of the virtual address space of a process which is non-core resident, Monitors the available main memory and periodically write the processes into the swap device to provide more processes fit in the main memory simultaneously.

#### 13) What is a Map?

A Map is an Array, which contains the addresses of the free space in the swap device that are allocatable resources, and the number of the resource units available there. This allows First-Fit allocation of contiguous blocks of a resource. Initially the Map contains one entry – address (block offset from the starting of the swap area) and the total number of resources. Kernel treats each unit of Map as a group of disk blocks. On the allocation and freeing of the resources Kernel updates the Map for accurate information.

**14) What scheme does the Kernel in Unix System V follow while choosing a swap device among the multiple swap devices?**

Kernel follows Round Robin scheme choosing a swap device among the multiple swap devices in Unix System V.

**15) What is a Region?**

A Region is a continuous area of a process's address space (such as text, data and stack). The kernel in a 'Region Table' that is local to the process maintains region. Regions are sharable among the process.

**16) What are the events done by the Kernel after a process is being swapped out from the main memory?**

When Kernel swaps the process out of the primary memory, it performs the following: Kernel decrements the Reference Count of each region of the process. If the reference count becomes zero, swaps the region out of the main memory, Kernel allocates the space for the swapping process in the swap device, Kernel locks the other swapping process while the current swapping operation is going on, The Kernel saves the swap address of the region in the region table.

**17) Is the Process before and after the swap are the same? Give reason.**

Process before swapping is residing in the primary memory in its original form. The regions (text, data and stack) may not be occupied fully by the process, there may be few empty slots in any of the regions and while swapping Kernel do not bother about the empty slots while swapping the process out. After swapping the process resides in the swap (secondary memory) device. The regions swapped out will be present but only the occupied region slots but not the empty slots that were present before assigning. While swapping the process once again into the main memory, the Kernel referring to the Process Memory Map, it assigns the main memory accordingly taking care of the empty slots in the regions.

**18) What do you mean by u-area (user area) or u-block?**



This contains the private data that is manipulated only by the Kernel. This is local to the Process, i.e. each process is allocated a u-area.

### **19) What are the entities that are swapped out of the main memory while swapping the process out of the main memory?**

All memory space occupied by the process, process's u-area, and Kernel stack are swapped out, theoretically. Practically, if the process's u-area contains the Address Translation Tables for the process then Kernel implementations do not swap the u-area.

### **20) What is Fork swap?**

fork() is a system call to create a child process. When the parent process calls fork() system call, the child process is created and if there is short of memory then the child process is sent to the read-to-run state in the swap device, and return to the user state without swapping the parent process. When the memory will be available the child process will be swapped into the main memory.

### **21) What is Expansion swap?**

At the time when any process requires more memory than it is currently allocated, the Kernel performs Expansion swap. To do this Kernel reserves enough space in the swap device. Then the address translation mapping is adjusted for the new virtual address space but the physical memory is not allocated. At last Kernel swaps the process into the assigned space in the swap device. Later when the Kernel swaps the process into the main memory this assigns memory according to the new address translation mapping.

### **22) How the Swapper works?**

The swapper is the only process that swaps the processes. The Swapper operates only in the Kernel mode and it does not uses System calls instead it uses internal Kernel functions for swapping. It is the archetype of all kernel process.

### **23) Who owns the data dictionary?**

The SYS user owns the data dictionary. The SYS and SYSTEM users are created

when the database is created.

**24) You routinely compress old log files. You now need to examine a log from two months ago. In order to view its contents without first having to decompress it ?**

zcat -> The zcat utility allows you to examine the contents of a compressed file much the same way that cat displays a file.

**25) You suspect that you have two commands with the same name as the command is not producing the expected results. What command can you use to determine the location of the command being run?**

which -> The which command searches your path until it finds a command that matches the command you are looking for and displays its full path.

**26) You locate a command in the /bin directory but do not know what it does. What command can you use to determine its purpose.**

whatis ->The whatis command displays a summary line from the man page for the specified command.

**27) When you issue the command ls -l, the first character of the resulting display represents the file's?**

type ->The first character of the permission block designates the type of file that is being displayed.

**28) What utility can you use to show a dynamic listing of running processes?**

top -> The top utility shows a listing of all running processes that is dynamically updated.

**29) Where is standard output usually directed?**

To the screen or display.

### 30) What daemon is responsible for tracking events on your system?

syslogd ->The syslogd daemon is responsible for tracking system information and saving it to specified log files.

Posted by [Linux Don](#) at 7:27 PM      +2 Recommend this on Google

Location: [Hyderabad, Andhra Pradesh, India](#)

## No comments:

## Post a Comment

Enter your comment...

Comment as:

**Publish**

Preview



[Newer Post](#)

[Home](#)

[Older Post](#)

Subscribe to: [Post Comments \(Atom\)](#)

All Rights for this blog.. Template images by [Airyelf](#). Powered by [Blogger](#).