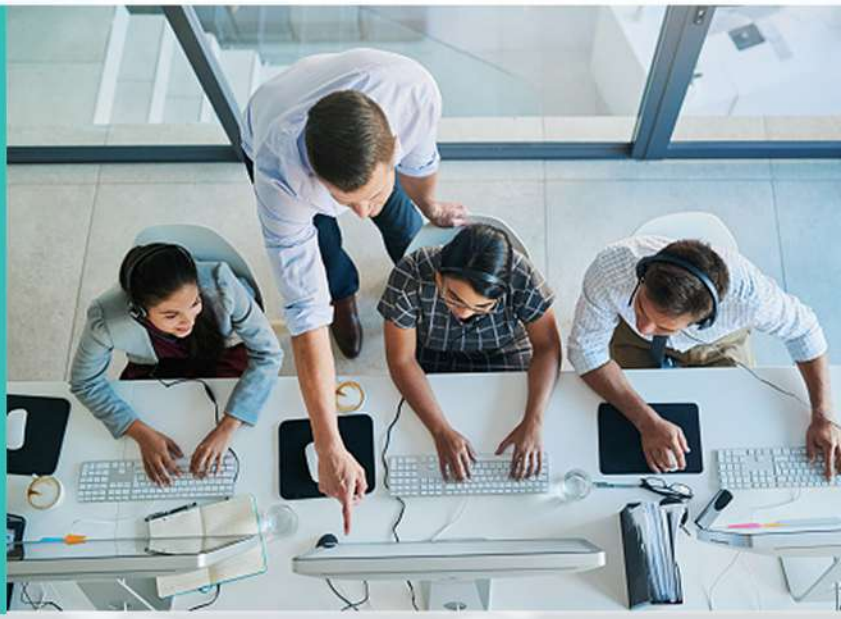
Andhra Pradesh State Skill Development Corporation

Skill AP
APSSDC

eSkill AP
Learn Anytime Anywhere
Andhra Pradesh State Skill Development Corporation



AWS CLOUD COMPUTING

LINUX - 1

Basic Linux Commands Part-1

Basic Linux Commands

What is UNIX?

The UNIX operating system is a set of programs that act as a link between the computer and the user. The computer program that allocates the system resources and coordinates all the details of the computer's internal is called the **operating system** or the **kernel.**

Users communicate with the kernel through a program known as the shell. The **shell** is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel.

- UNIX was originally developed in 1969 by a group of AT&T employees Ken Thompson, Dennis Ritchie, Douglas McIlroy, and Joe Ossanna at Bell Labs.
- There are various UNIX variants available in the market. Solaris UNIX, AIX, HP UNIX and BSD are a few examples. Linux is also a flavour of UNIX which is freely available.
- Several people can use a UNIX computer at the same time; hence UNIX is called a multiuser system.
- A user can also run multiple programs at the same time; hence UNIX is a multitasking environment.

The main concept that unites all the versions of Unix is the following four basics:

- **Kernel** − The kernel is the heart of the operating system. It interacts with the hardware and most of the tasks like memory management, task scheduling and file management.
- **Shell** − The shell is the utility that processes your requests. When you type in a command at your terminal, the shell interprets the command and calls the program that you want. The shell uses standard syntax for all commands. C Shell, Bourne Shell and Korn Shell are the most famous shells which are available with most of the UNIX variants.
- **Commands and Utilities** − There are various commands and utilities which you can make use of in your day to day activities. cp, mv, cat and grep, etc. are few examples of commands and utilities. There are over 250 standard commands plus numerous others provided through 3rd party software. All the commands come along with various options.
- **Files and Directories** − All the data of Unix is organized into files. All files are then organized into directories. These directories are further organized into a tree-like structure called the file system.

System Boot up:

If you have a computer which has the Unix operating system installed in it, then you simply need to turn on the system to make it live.

As soon as you turn on the system, it starts booting up and finally it prompts you to log into the system, which is an activity to log into the system and use it for your day-to-day activities.

Login UNIX:

When you first connect to a Unix system, you usually see a prompt such as the following

Login:

To log in

- Have your user id (user identification) and password ready. Contact your system administrator if you don't have these yet.
- Type your user id at the login prompt, and then press ENTER. Your userID is case-sensitive, so be sure you type it exactly as your system administrator has instructed.
- Type your password at the password prompt, and then press ENTER. Your password is also case-sensitive.
- If you provide the correct userID and password, then you will be allowed to enter into the system.

You will be provided with a command prompt (sometimes called the $ prompt) where you type all your commands. For example, to check calendar, you need to type the **cal** command as follows –

$ cal

```
    June 2009
Su Mo Tu We Th Fr Sa
       1  2   3  4  5  6
 7  8  9  10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30
$
```

Change Password:

All Unix systems require passwords to help ensure that your files and data remain your own and that the system itself is secure from hackers and crackers. Following are the steps to change your password −

Step 1 − To start, type password at the command prompt as shown below.

Step 2 − Enter your old password, the one you're currently using.

Step 3 − Type in your new password. Always keep your password complex enough so that nobody can guess it. But make sure you remember it.

Step 4 − You must verify the password by typing it again.

$ passwd

Changing password for amrood

(current) Unix password : ******

New UNIX password : *******

Retype new UNIX password : *******

Passwd : all authentication tokens updated successfully

$

Note: We have added asterisk (*) here just to show the location where you need to enter the current and new passwords otherwise at your system. It does not show you any character when you type.

Listing Directories and Files:

All data in Unix is organized into files. All files are organized into directories. These directories are organized into a tree-like structure called the filesystem.

You can use the **ls** command to list out all the files or directories available in a directory. Following is the example of using **ls** command with **-l** option.

$ ls -l

total 19621

drwxrwxr-x 2 amrood amrood 4096 Dec 25 09:59 uml

-rw-rw-r-- 1 amrood amrood 5341 Dec 25 08:38 uml.jpg

drwxr-xr-x 2 amrood amrood 4096 Feb 15 2006 univ

drwxr-xr-x 2 root root 4096 Dec 9 2007 urlspedia

-rw-r--r-- 1 root root 276480 Dec 9 2007 urlspedia.tar

drwxr-xr-x 8 root root 4096 Nov 25 2007 usr

-rwxr-xr-x 1 root root 3192 Nov 25 2007 webthumb.php

-rw-rw-r-- 1 amrood amrood 20480 Nov 25 2007 webthumb.tar

-rw-rw-r-- 1 amrood amrood 5654 Aug 9 2007 yourfile.mid

-rw-rw-r-- 1 amrood amrood 166255 Aug 9 2007 yourfile.swf

$

Here entries starting with d..... represent directories. For example, uml, univ and urlspedia are directories and the rest of the entries are files.

Who Are You?

While you're logged into the system, you might be willing to know: Who am I?

The easiest way to find out "who you are" is to enter the whoami command –

$ whoami

amrood

$

Try it on your system. This command lists the account name associated with the current login. You can try **who am i** command as well to get information about yourself.

Who is Logged in?

Sometime you might be interested to know who is logged in to the computer at the same time.

There are three commands available to get you this information, based on how much you wish to know about the other users: **users, who,** and **w.**

$ users

amrood bablu qadir

$ who

amrood ttyp0 Oct 8 14:10 (limbo)

bablu ttyp2 Oct 4 09:08 (calliope)

qadir ttyp4 Oct 8 12:09 (dent)

$

Try the w command on your system to check the output. This lists down information associated with the users logged in the system.

Logging Out:

When you finish your session, you need to log out of the system. This is to ensure that nobody else accesses your files.

To log out

- Just type the **logout** command at the command prompt, and the system will clean up everything and break the connection.

System Shutdown:

The most consistent way to shut down a Unix system properly via the command line is to use one of the following commands –

S.no	Command & Description
1	**Halt** Brings the system down immediately
2	**init 0** Powers off the system using predefined scripts to synchronize and clean up the system prior to shutting down
3	**init 6** Reboots the system by shutting it down completely and then restarting it
4	**Poweroff** Shuts down the system by powering off
5	**Reboot** Reboots the system
6	**Shutdown** Shuts down the system

You typically need to be the super user or root (the most privileged account on a Unix system) to shut down the system.

Listing Files:

To list the files and directories stored in the current directory, use the following command - $ls Here is the sample output of the above command –

$ ls

```
bin       hosts  lib     res.03
ch07      hw1    pub     test_results
ch07.bak  hw2    res.01  users
docs      hw3    res.02  work
```

The command **ls** supports the **-l** option which would help you to get more information about the listed files

$ ls -l

```
total 1962188
drwxrwxr-x  2 amrood amrood      4096 Dec 25 09:59 uml
-rw-rw-r--  1 amrood amrood      5341 Dec 25 08:38 uml.jpg
drwxr-xr-x  2 amrood amrood      4096 Feb 15  2006 univ
drwxr-xr-x  2 root   root        4096 Dec  9  2007 urlspedia
-rw-r--r--  1 root   root      276480 Dec  9  2007 urlspedia.tar
drwxr-xr-x  8 root   root        4096 Nov 25  2007 usr
drwxr-xr-x  2    200    300      4096 Nov 25  2007 webthumb-1.01
-rwxr-xr-x  1 root   root        3192 Nov 25  2007 webthumb.php
-rw-rw-r--  1 amrood amrood     20480 Nov 25  2007 webthumb.tar
-rw-rw-r--  1 amrood amrood      5654 Aug  9  2007 yourfile.mid
-rw-rw-r--  1 amrood amrood    166255 Aug  9  2007 yourfile.swf
drwxr-xr-x 11 amrood amrood      4096 May 29  2007 zlib-1.2.3
$
```

Here is the information about all the listed columns −

- **First Column** - Represents the file type and the permission given on the file. Below is the description of all types of files.
- **Second Column** - Represents the number of memory blocks taken by the file or directory.
- **Third Column** - Represents the owner of the file. This is the Unix user who created this file.
- **Fourth Column** - Represents the group of the owner. Every Unix user will have an associated group.

- **Fifth Column** - Represents the file size in bytes.
- **Sixth Column** - Represents the date and the time when this file was created or modified for the last time.
- **Seventh Column** - Represents the file or the directory name.

In the ls -l listing example, every file line begins with a **d**, **-**, or **l**. These characters indicate the type of the file that's listed.

S.No	Prefix & Description
1	**-** Regular file, such as an ASCII text file, binary executable, or hard link.
2	**B** Block special file. Block input/output device file such as a physical hard drive.
3	**C** Character special file. Raw input/output device file such as a physical hard drive.
4	**d** Directory file that contains a listing of other files and directories.
5	**l** Symbolic link file. Links on any regular file.

6	**P** Named pipe. A mechanism for interprocess communications.
7	**s** Socket used for interprocess communication.

Hidden Files:

An invisible file is one, the first character of which is the dot or the period character (.). Unix programs (including the shell) use most of these files to store configuration information.

Some common examples of the hidden files include the files −

- .profile − The Bourne shell (sh) initialization script
- .kshrc − The Korn shell (ksh) initialization script
- .cshrc − The C shell (csh) initialization script
- .rhosts − The remote shell configuration file

To list the invisible files, specify the **-a** option to **ls** –

$ ls -a

```
.       .profile      docs    lib    test_results
..      .rhosts       hosts   pub    users
.emacs  bin           hw1     res.01 work
.exrc   ch07          hw2     res.02
.kshrc  ch07.bak      hw3     res.03
$
```

- Single dot (.) − This represents the current directory.
- Double dot (..) − This represents the parent directory.

Creating Files:

You can use the vi editor to create ordinary files on any Unix system. You simply need to give the following command –

$ vi filename

The above command will open a file with the given filename. Now, press the key **i** to come into edit mode. Once you are in the edit mode, you can start writing your content in the file as in the following program –

This is unix file....I created it for the first time.....

I'm going to save this content in this file.

Once you are done with the program, follow these steps −

- Press the key esc to come out of the edit mode.
- Press two keys Shift + ZZ together to come out of the file completely.

You will now have a file created with filename in the current directory.

$ vi filename

$

Display Content of a File:

You can use the **cat** command to see the content of a file. Following is a simple example to see the content of the above created file −

$ cat filename

This is unix file....I created it for the first time.....

I'm going to save this content in this file.

$

You can display the line numbers by using the **-b** option along with the **cat** command as follows −

$ cat -b filename

1 This is unix file....I created it for the first time.....

2 I'm going to save this content in this file.

$

Counting Words in a File:

You can use the **wc** command to get a count of the total number of lines, words, and characters contained in a file. Following is a simple example to see the information about the file created above −

$ wc filename

2 19 103 filename

$

Here is the detail of all the four columns −

- **First Column** - Represents the total number of lines in the file.
- **Second Column** - Represents the total number of words in the file.
- **Third Column** - Represents the total number of bytes in the file. This is the actual size of the file.
- **Fourth Column** - Represents the file name.

Copying Files:

To make a copy of a file use the cp command. The basic syntax of the command is −

$ cp source_file destination_file

Following is the example to create a copy of the existing file filename.

$ cp filename copyfile

$

You will now find one more file copyfile in your current directory. This file will exactly be the same as the original file filename.

Renaming Files:

To change the name of a file, use the mv command. Following is the basic syntax −

$ mv old_file new_file

The following program will rename the existing file filename to newfile.

$ mv filename newfile

$

The mv command will move the existing file completely into the new file. In this case, you will find only new file in your current directory.

Creating multiple files:

To create multiple files, use touch command. Following is the basic syntax

$ touch file1 file2 file3

$

Deleting Files:

To delete an existing file, use the rm command. Following is the basic syntax −

$ rm filename

Caution - A file may contain useful information. It is always recommended to be careful while using this Delete command. It is better to use the -i option along with rm command.

Following is the example which shows how to completely remove the existing file filename.

$ rm filename

$

You can remove multiple files at a time with the command given below −

$ rm filename1 filename2 filename3

$

Home Directory:

The directory in which you find yourself when you first login is called your home directory.

You will be doing much of your work in your home directory and subdirectories that you'll be creating to organize your files.

You can go in your home directory anytime using the following command −

$ cd ~

$

Here ~ indicates the home directory. Suppose you have to go in any other user's home directory, use the following command −

$ cd ~username

$

To go in your last directory, you can use the following command −

$ cd -

$

Listing Directories:

To list the files in a directory, you can use the following syntax −

$ ls dirname

Following is the example to list all the files contained in /usr/local directory −

$ ls /usr/local

X11 bin gimp jikes sbin

ace doc include lib share

atalk etc info man ami

Creating Directories:

We will now understand how to create directories. Directories are created by the following command −

$ mkdir dirname

Here, directory is the absolute or relative pathname of the directory you want to create. For example, the command −

$ mkdir mydir

$

Creates the directory mydir in the current directory. Here is another example −

$ mkdir /tmp/test-dir

$

This command creates the directory test-dir in the /tmp directory. The mkdir command produces no output if it successfully creates the requested directory.

If you give more than one directory on the command line, mkdir creates each of the directories. For example, −

$ mkdir docs pub

$

Creates the directories docs and pub under the current directory.

Removing Directories:

Directories can be deleted using the rmdir command as follows −

$ rmdir dirname

$

Note: To remove a directory, make sure it is empty which means there should not be any file or sub-directory inside this directory.

You can remove multiple directories at a time as follows −

$ rmdir dirname1 dirname2 dirname3

$

The above command removes the directories dirname1, dirname2, and dirname3, if they are empty. The rmdir command produces no output if it is successful.

Changing Directories

You can use the cd command to do more than just change to a home directory. You can use it to change to any directory by specifying a valid absolute or relative path. The syntax is as given below −

$ cd dirname

$

Here, dirname is the name of the directory that you want to change to. For example, the command −

$ cd /usr/local/bin

$

Changes to the directory /usr/local/bin. From this directory, you can cd to the directory /usr/home/amrood using the following relative path −

$ cd ../../home/amrood

$

Renaming Directories:

The **mv** (move) command can also be used to rename a directory. The syntax is as follows −

$mv olddir newdir

$

You can rename a directory mydir to yourdir as follows −

$ mv mydir yourdir

$

The directories . (dot) and .. (dot dot):

The filename . (dot) represents the current working directory; and the filename .. (dot dot) represents the directory one level above the current working directory, often referred to as the parent directory.

If we enter the command to show a listing of the current working directories/files and use the **-a** option to list all the files and the **-l** option to provide the long listing, we will receive the following result.

$ ls -la

drwxrwxr-x 4 teacher class 2048 Jul 16 17.56 .

```
drwxr-xr-x    60   root              1536  Jul 13 14:18 ..
----------    1    teacher   class   4210  May 1 08:27 .profile
-rwxr-xr-x    1    teacher   class   1948  May 12 13:42 memo
$
```