ASHOK IT

*Learn Here.. Lead Anywhere..!!*

# LINUX OS

## By Mr. Ashok

**Phone: +91 9985 39 66 77**

**Location: H.No - 7-1-413/2, Beside Sonabhai Temple, Ameerpet, Hyderabad - 500016**

## What is Operating System?

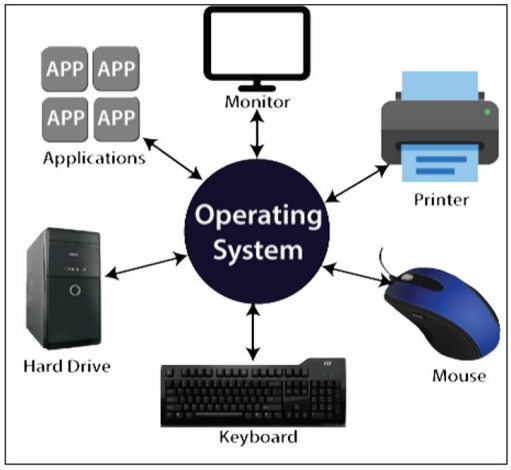
-> Operating System is a software which is used to interact with computer

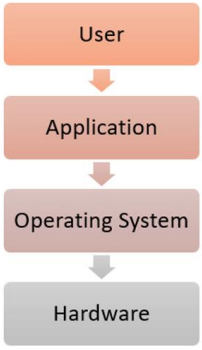
-> Operating System is acting as mediator between users and computer hardware components

-> Operating System is mandatory to use any computer

-> OS provides environment to run other applications

(Browser, Notepad, Paint, Calc)




-> The OS came into market in 1950

-> Microsoft released its first OS in 1981 (MS DOS)

## Windows OS

-> Developed by Microsoft Company

-> It is having GUI (Graphical user interface)

-> It is single user based Operating System (only one person can use this at a time)

-> It is commercial (paid)

-> Less Security

-> It is recommended for personal use

- Watch Movies
- Using Internet
- Playing Games
- Attending Online classes
- Store data

## Linux OS

-> Linux is Community Based OS

-> Linux is Free & Open-Source OS

-> Linux is Multi User Based OS

-> Linux provides High Security

-> Linux is recommended to use for Applications, Servers, Databases etc

Note: In real-time, we will use Linux OS only to setup infrastructure required to run our application

-> Linux OS is not only for Administrators, even developers and testers also will use Linux OS in real-time to monitor our application and application servers and application logs.

## History of Linux

-> In 1991, a student 'Linus Torvalds' developed this Linux OS

-> Linux Torvalds identified some challenges in UNIX OS and he suggested some changes for Unix OS but UNIX OS Team rejected 'Linux Torvalds 'suggestions

-> Linus Torvalds used Minux OS to develop Linux

Linus + Minux -

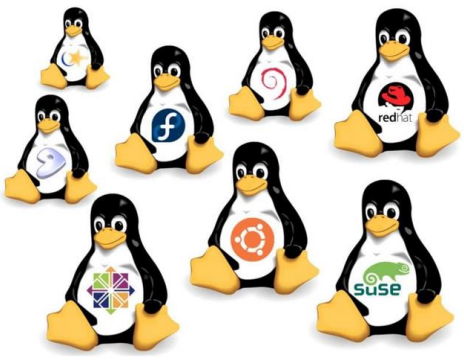
-> First Two letters from his name and last 3 letters from Minux OS

LI + NUX => LINUX

-> Linus Torvalds released LINUX OS with source code into market so that anybody can modify LINUX OS thats why it is called as Open-Source Operating System.

-> As Linux OS is open source, so many people and companies taken that Linux OS and modified according to their requirements and released into market with different names those are called as Linux Distributions.

- RED HAT
- Ubuntu OS
- Cent OS
- Fedora
- Open SUSE
- Kali Linux
- Debian
- Amazon Linux




Note: 200+ Linux Distributions are available in the market.

## Environment Setup

**We can setup Linux machine in multiple ways**

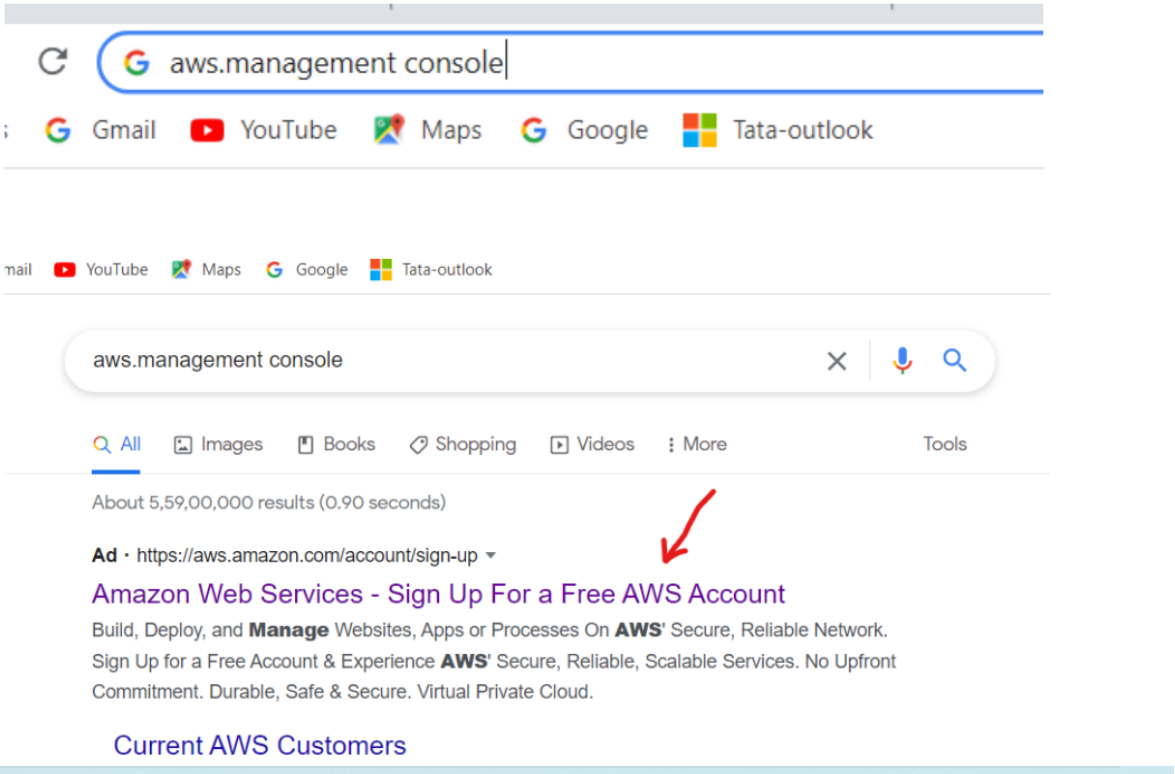
1) **We can install Linux OS directly in computer**
2) **We can setup Linux OS in windows machine using Virtual Box**
3) **We can setup Linux Virtual Machine Cloud Platform like AWS and Azure**

**Note: In this note we will see how to setup Linux VM in AWS Cloud**

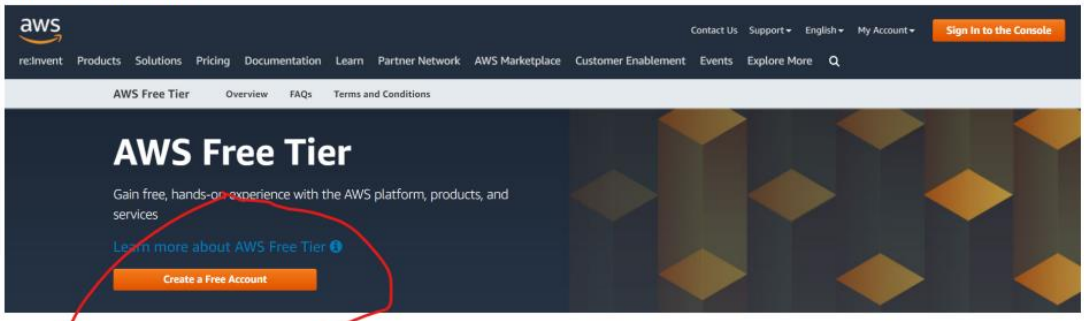
## AWS Account Creation

**-> Create account in AWS (it will ask debit/credit card) - selected cards are accepted**

AWS Account creation
Search on google aws.management console like snip



Click the first link




**Types of offers**

Explore more than 100 products and start building on AWS using the Free Tier. Three different types of free offers are available depending on the product used. Click icon below to explore our offers.

**Explore Free Tier products with a new AWS account.**

To learn more, visit aws.amazon.com/free.

## Sign up for AWS

Email address
You will use this email address to sign in to your new AWS account.

Password

Confirm password

AWS account name
Choose a name for your account. You can change this name in your account settings after you sign up.

Mannava Narasimharao

Continue (step 1 of 5)

Sign in to an existing AWS account

## Free Tier offers

All AWS accounts can explore 3 different types of free offers, depending on the product used.

**Always free**
Never expires

**12 months free**
Start from initial sign-up date

**Trials**
Start from service activation date

# Sign up for AWS

## Contact Information

How do you plan to use AWS?

- ○ Business - for your work, school, or organization
- ◉ Personal - for your own projects

Who should we contact about this account?

Full Name

[redacted]

Phone Number
Enter your country code and your phone number.

[redacted]

Country or Region

India

Address

vinukonda

*Apartment, suite, unit, building, floor, etc.*

City

vinukonda

## Secure verification

We will not charge for usage below AWS Free Tier limits. We temporarily hold INR 2 as a pending transaction for 3-5 days to verify your identity.

## Sign up for AWS

### Billing Information

Credit or Debit card number

[redacted]

VISA

AWS accepts all major credit and debit cards. To learn more about payment options, review our FAQ

Expiration date

[redacted]

Cardholder's name

[redacted]

CVV

•••

Billing address

- (•) Use my contact address
  vinukonda
  vinukonda Andhra Pradesh 522659
  IN
- ( ) Use a new address

Do you have a PAN?

Permanent Account Number (PAN) is a ten-digit alphanumeric number issued by the Indian Income Tax Department. This 10-digit number is printed on the front of your PAN card.

- ( ) Yes

VISA

kotak Kotak Mahindra Bank

**Merchant Details**

| | |
|---|---|
| Merchant Name | AMAZON |
| Date | Nov 29, 2021 |
| Card Number | 4594 XXXX XXXX 1755 |
| Amount | **INR2.00** |

**Authenticate Transaction**

OTP

Your OTP was successfully sent to your registered mobile number XXXXXX7418 and Email Address maXXXXXXXXXXXXXXXXXXX@gmail.com.
Not your contact details? Refresh Here

Enter OTP

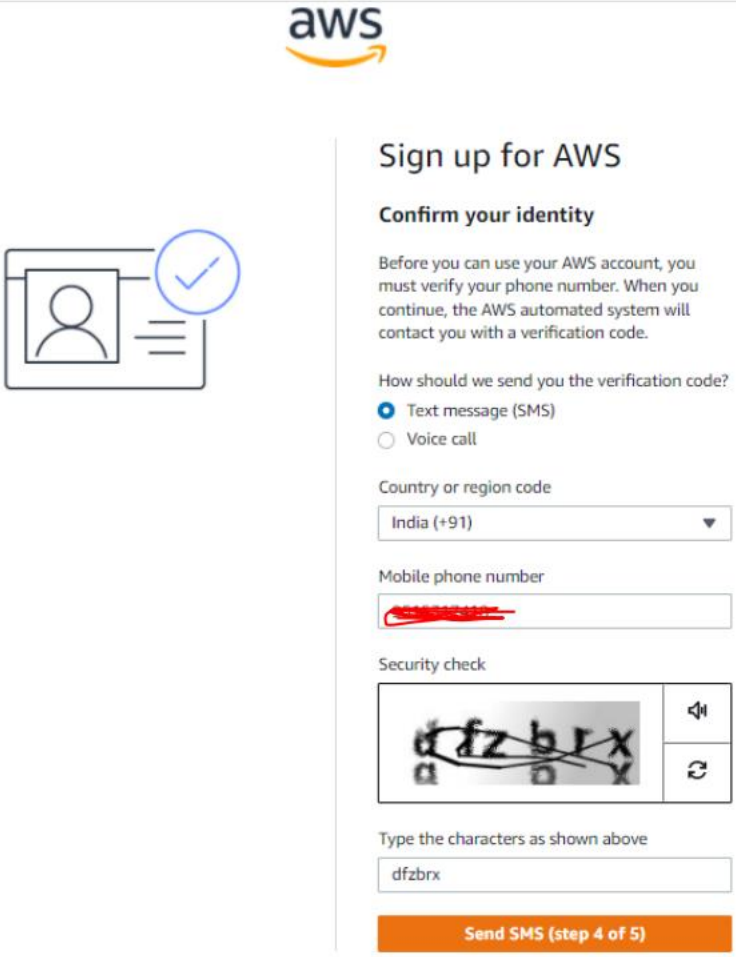
••••••

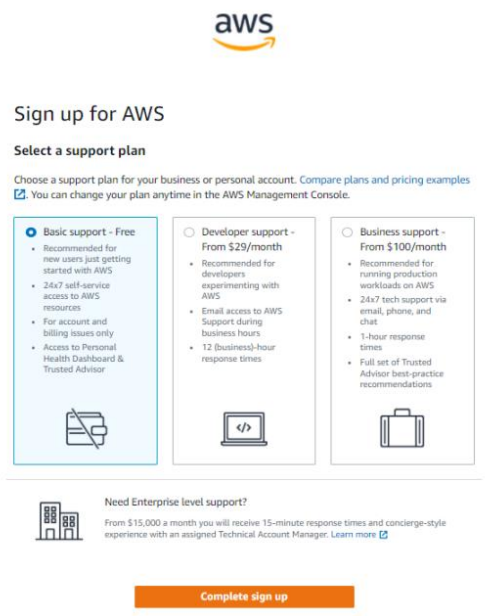
Resend OTP

Cancel | Submit

This screen will automatically time out after 2 minutes

Powered by


aws
Sign up for AWS
Confirm your identity
Before you can use your AWS account, you must verify your phone number. When you continue, the AWS automated system will contact you with a verification code.
How should we send you the verification code?
Text message (SMS)
Voice call
Country or region code
India (+91)
Mobile phone number
Security check
dfzbrx
Type the characters as shown above
dfzbrx
Send SMS (step 4 of 5)


aws
Sign up for AWS
Select a support plan
Choose a support plan for your business or personal account. Compare plans and pricing examples. You can change your plan anytime in the AWS Management Console.
Basic support - Free
Recommended for new users just getting started with AWS
24x7 self-service access to AWS resources
For account and billing issues only
Access to Personal Health Dashboard & Trusted Advisor
Developer support - From $29/month
Recommended for developers experimenting with AWS
Email access to AWS Support during business hours
12 (business)-hour response times
Business support - From $100/month
Recommended for running production workloads on AWS
24x7 tech support via email, phone, and chat
1-hour response times
Full set of Trusted Advisor best-practice recommendations
Need Enterprise level support?
From $15,000 a month you will receive 15-minute response times and concierge-style experience with an assigned Technical Account Manager. Learn more
Complete sign up

## Congratulations

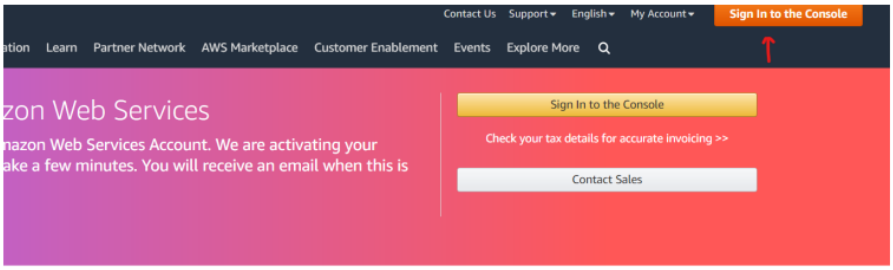
Thank you for signing up for AWS.

We are activating your account, which should only take a few minutes. You will receive an email when this is complete.

**Go to the AWS Management Console**

Sign up for another account or contact sales.

Click her to sign in your account




Personalize Your Experience

Select root user and enter your mail id and click next.

## Sign in

**Root user**
Account owner that performs tasks requiring unrestricted access. Learn more

**IAM user**
User within an account that performs daily tasks.
Learn more

**Root user email address**

[redacted]@gmail.com

**Next**

By continuing, you agree to the AWS Customer Agreement or other agreement for AWS services, and the Privacy Notice. This site uses essential cookies. See our Cookie Notice for more information.

## Root user sign in

Email: [redacted]@gmail.com

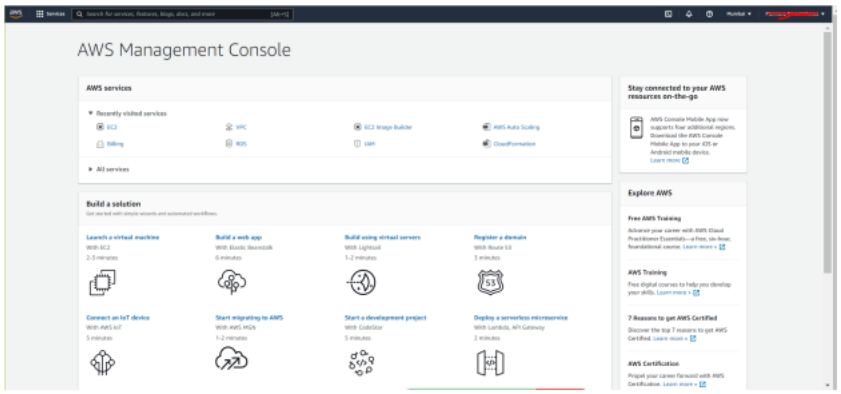
Password Forgot password?

••••••••••••

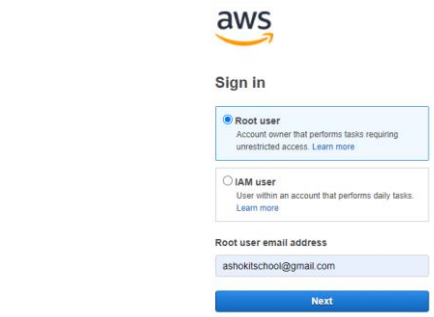
Sign in

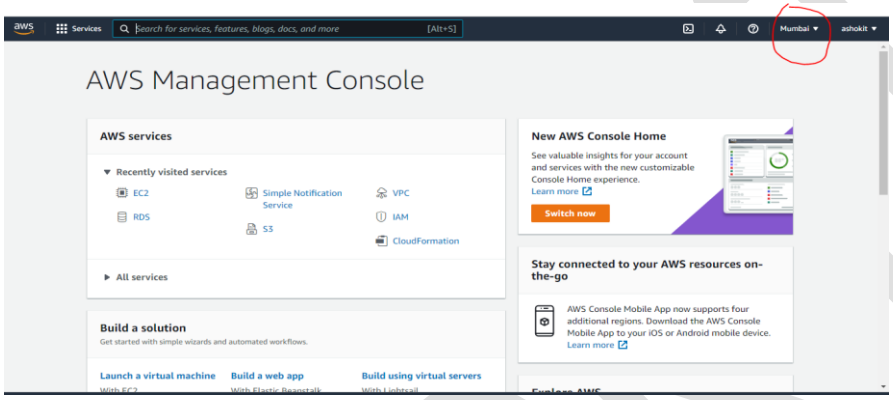
Sign in to a different account

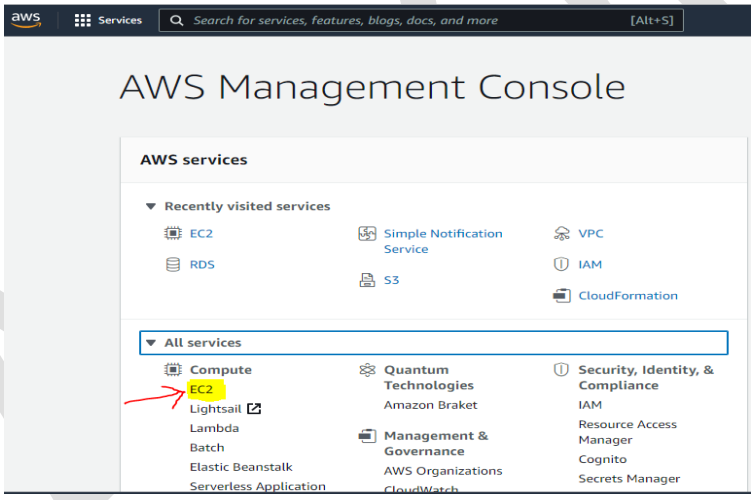
Create a new AWS account

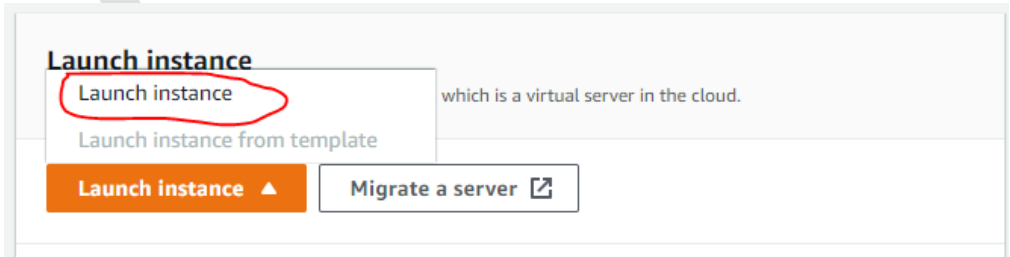


# Create Linux VM using AWS EC2 service



## Name and tags Info

Name

Linux-VM

Add additional tags

## Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Recents | My AMIs | Quick Start

Amazon Linux | macOS | Ubuntu | Windows | Red Hat | SUSE

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

## Instance type Info

Instance type

t2.micro — Free tier eligible

Family: t2 1 vCPU 1 GiB Memory

On-Demand Linux pricing: 0.0124 USD per Hour

On-Demand Windows pricing: 0.017 USD per Hour

Compare instance types

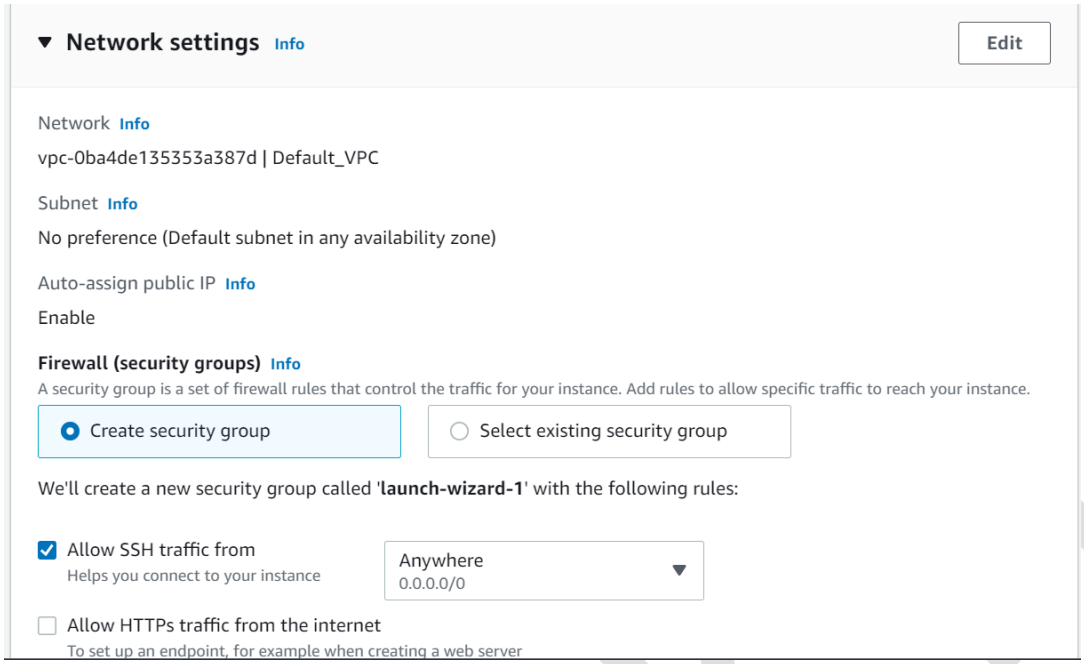
## Key pair (login) Info

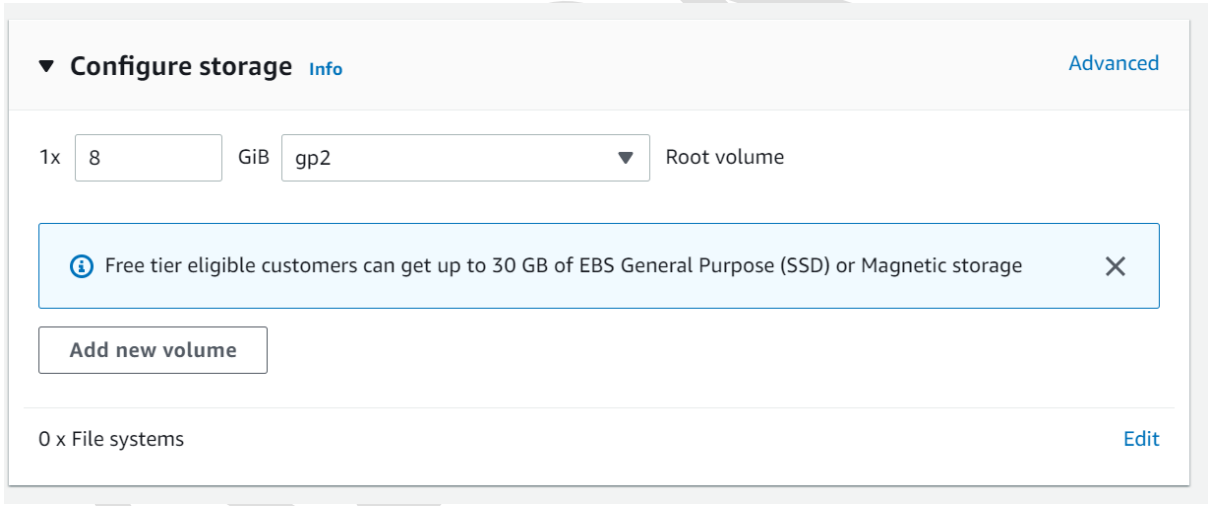
You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

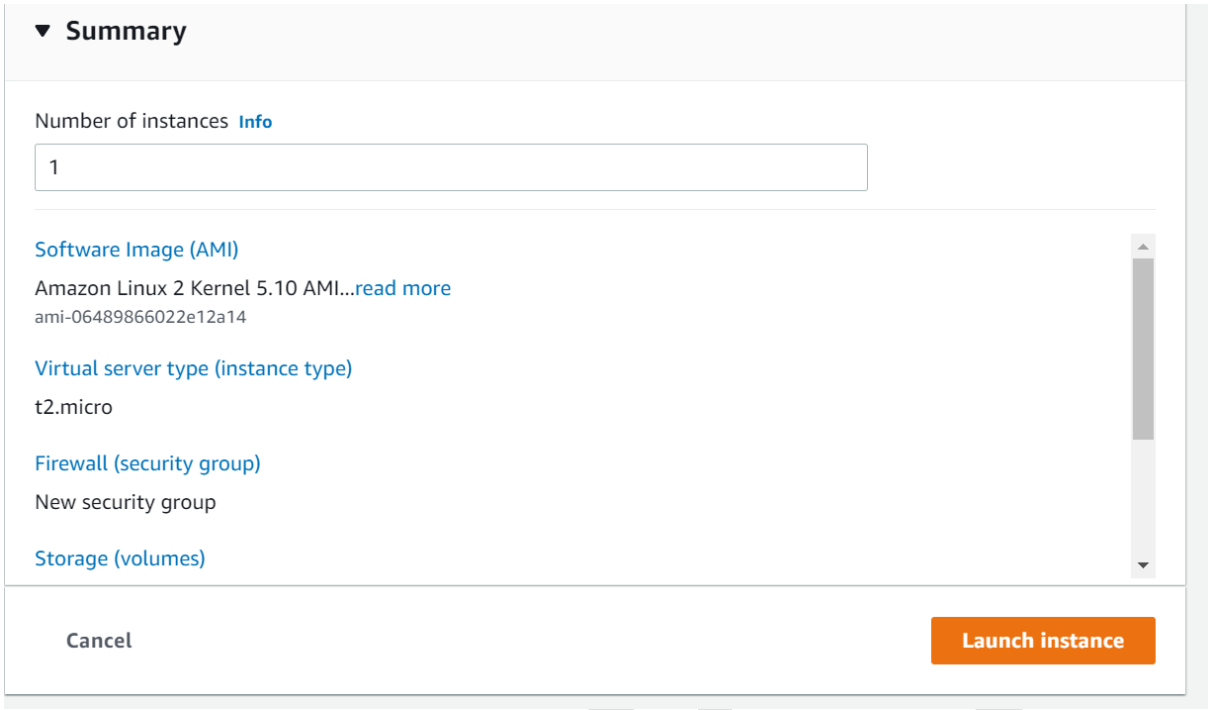
Key pair name - *required*

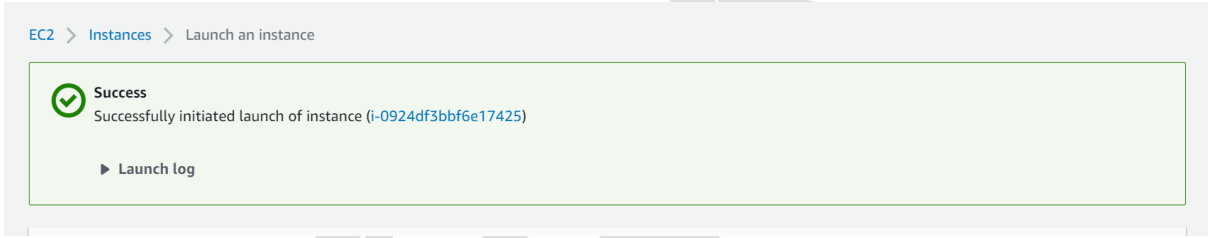
linux

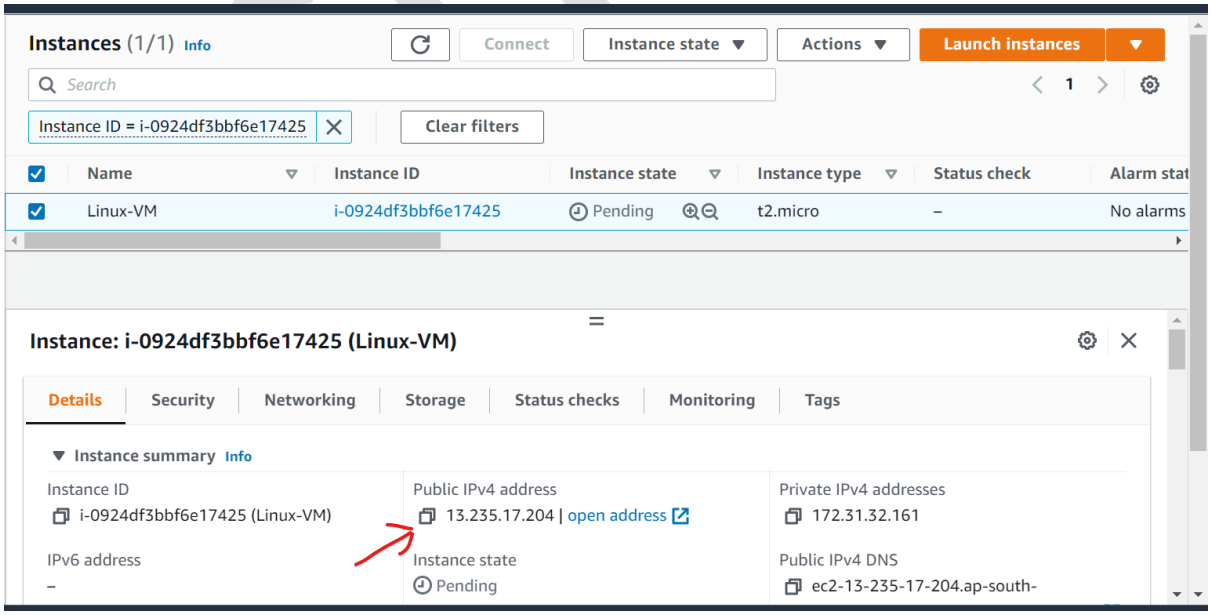
Create new key pair


Network settings Info
Edit
Network Info
vpc-0ba4de135353a387d | Default_VPC
Subnet Info
No preference (Default subnet in any availability zone)
Auto-assign public IP Info
Enable
Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
Create security group
Select existing security group
We'll create a new security group called 'launch-wizard-1' with the following rules:
Allow SSH traffic from
Helps you connect to your instance
Anywhere
0.0.0.0/0
Allow HTTPs traffic from the internet
To set up an endpoint, for example when creating a web server

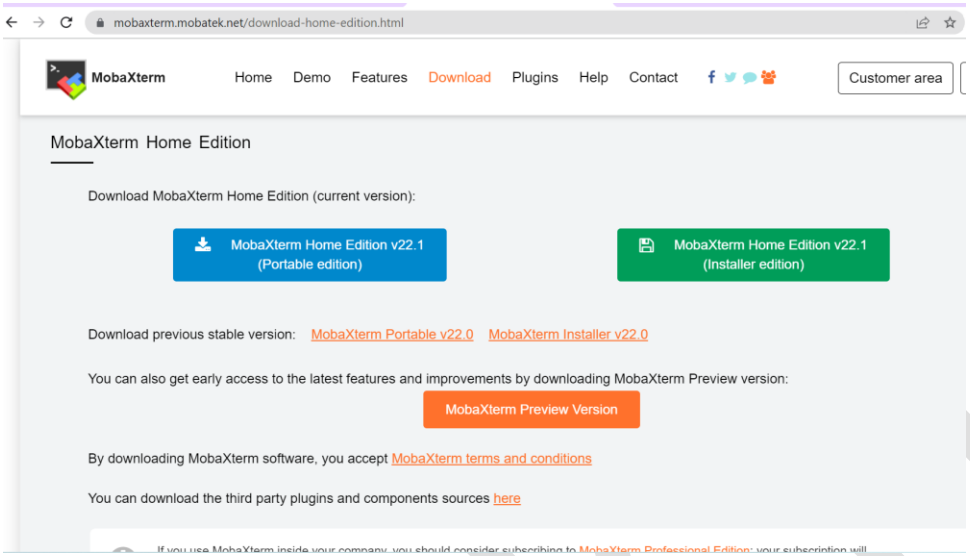

Configure storage Info
Advanced
1x
8
GiB
gp2
Root volume
Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage
Add new volume
0 x File systems
Edit


Summary
Number of instances Info
1
Software Image (AMI)
Amazon Linux 2 Kernel 5.10 AMI...read more
ami-06489866022e12a14
Virtual server type (instance type)
t2.micro
Firewall (security group)
New security group
Storage (volumes)
Cancel
Launch instance


EC2 > Instances > Launch an instance
Success
Successfully initiated launch of instance (i-0924df3bbf6e17425)
Launch log


Instances (1/1) Info
Connect
Instance state
Actions
Launch instances
Search
1
Instance ID = i-0924df3bbf6e17425
Clear filters
Name
Instance ID
Instance state
Instance type
Status check
Alarm stat
Linux-VM
i-0924df3bbf6e17425
Pending
t2.micro
–
No alarms
Instance: i-0924df3bbf6e17425 (Linux-VM)
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Instance summary Info
Instance ID
i-0924df3bbf6e17425 (Linux-VM)
Public IPv4 address
13.235.17.204 | open address
Private IPv4 addresses
172.31.32.161
IPv6 address
–
Instance state
Pending
Public IPv4 DNS
ec2-13-235-17-204.ap-south-

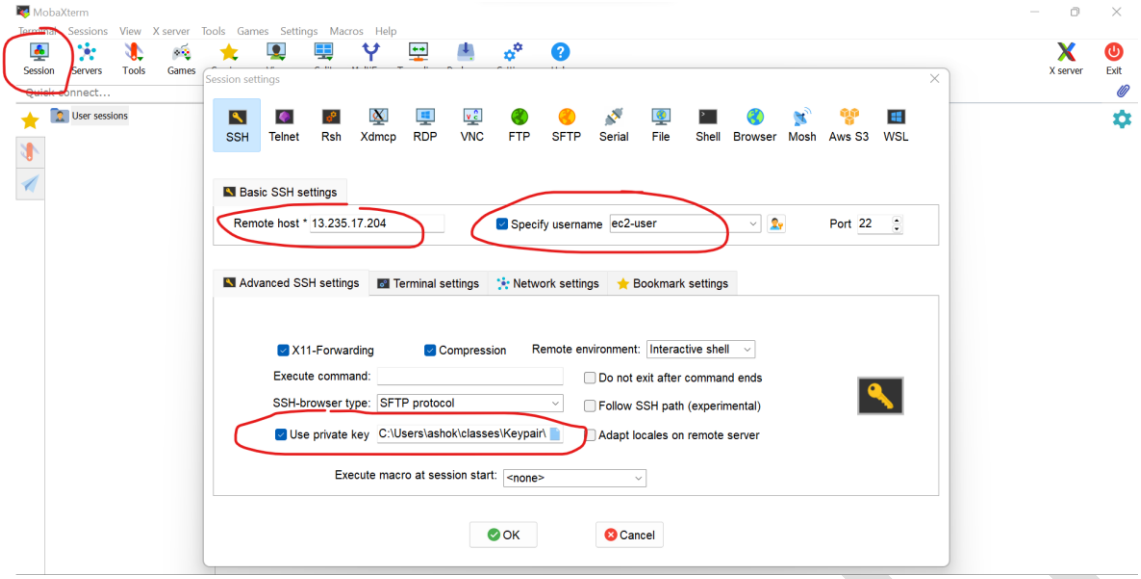
## Download Mobaxterm software to connect with Linux VM



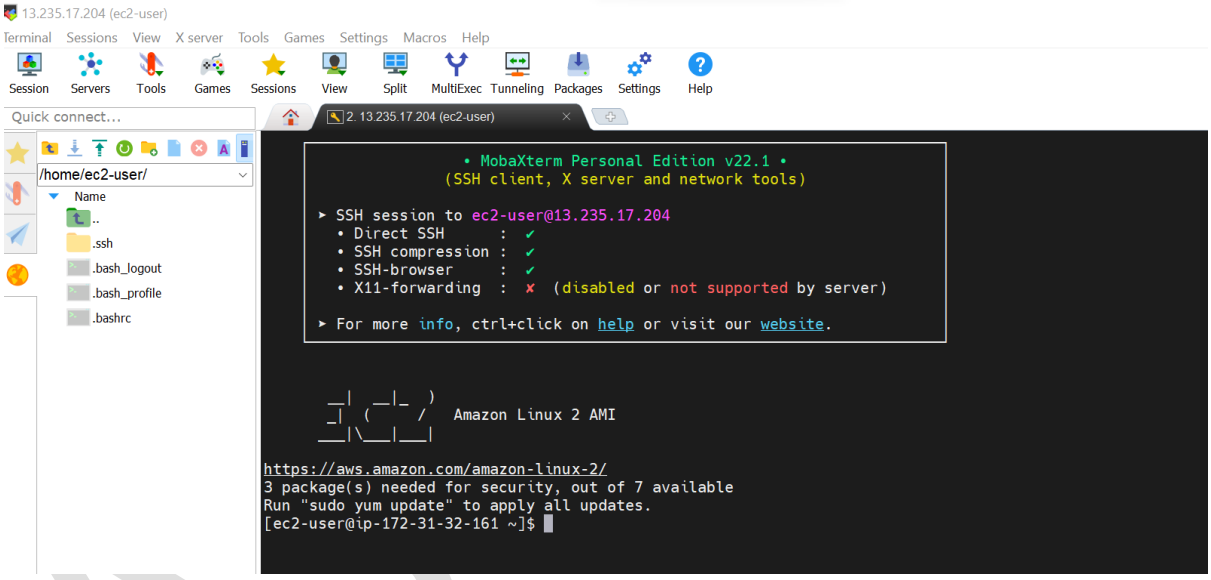

## Connect with Linux VM using Public IP and pem file

**Note: Once work is completed then stop your Linux VM in EC2 to avoid billing**


➔ **Open Mobaxterm software and connect with Linux VM**




➔ **Once we connect with EC2 instance using Mobaxterm we can see screen like below**




## Linux Commands

**$ whoami : It will display currently logged in username**

**$ pwd : present working directory**

**$ date : To display current date**

**$ cal : To display calendar**

## Linux File System

-> In Linux everything will be represented as a file

-> We have 3 types of files in Linux

1) Ordinary File / Normal File (it will start with -)

2) Directory File (it will start with d)

3) Link File (it will start with l)

-> The file which contains data is called as ordinary file

-> Directory file is equal to folder (it can contain files and folders)

-> The file which is having linking is called as Link File

-> touch : it is used to create empty file

```
$ touch f1.txt
$ touch f2.txt
$ touch f3.txt f4.txt
```

-> To display files we will use 'ls' command

```
$ ls
```

-> To create a file with data we will use 'cat' command

```
$ cat > hello.txt
//write data
press CTRL + d (to save and exit)
$ cat hello.txt   (To display file data)
$ cat >> hello.txt (To append data in the file)
//write data
press CTRL + d (to save and exit)
```

-> To create directory we will use 'mkdir' command

```
$ mkdir dirname
```

-> To remove the file we will use 'rm' command

```
$ rm filename
```

-> To remove empty directory we will use 'rmdir' command

```
$ rmdir dirname
```

-> 'ls' is used to list out all files & directories available in the given directory

Note: we can pass several options for 'ls' commands

-> ls : It will display all files in alphabetical order (a to z)

-> ls -r : It will display all files in reverse of alphabetical order (z to a)

-> ls -l : It will display long listing of files

-> ls -t : It will display all files based on last modified date and time. Most recent file will be display at top and old files will display at bottom.

-> ls -rt : It will display all files based on reverse of last modified date and time. Old files will display at top and recent files will display bottom.

-> ls -a : It will display all files including hidden files (hidden files will start with .)

-> ls -li : It will display files with inode numbers.

-> ls -lR : It will display all files and directories along with sub directoris content

Note: -R represents recursive

Note: We can use several options for ls command at a time. When we are using multiple options order of the options is not important

$ ls -ltr

$ ls -tlr

$ ls -l -t -r

$ ls -trl

Note: All the above commands will give same output

-> To display content of given directory we can execute like below

$ ls -l <dirname>

-> To delete a file we will use 'rm' command

$ rm <filename>

-> To delete empty directory we will use 'rmdir' command

$ rmdir <dirname>

-> To delete non-empty directory we will use 'rm' command like below

$ rm -r dirname

-> To display file content we will use 'cat' command

$ cat filename

-> To display file content with line numbers we will use '-n' option

$ cat -n filename

-> To display multiple files content at a time execute command like below

$ cat file1 file2 file3

-> Copy one file data into another file using 'cat' command

$ cat f1.txt > f8.txt

-> Copy more than one file data into another file

$ cat f1.txt f2.txt > f9.txt

-> 'tac' command is used to reverse file content

$ tac filename

-> 'rev' command is used to reverse each line content of the file

$ rev filename

-> head command is used to display file data from top (default 10 lines)

$ head filename

$ head -n 5 data.log (first 5 lines data)

$ head -n 20 data.log (first 20 lines data)

-> tail command is used to display file data from bottom (default 10 lines)

$ tail filename (last 10 lines data)

$ tail -n 100 filename (last 100 lines data)

$ tail +25 filename (it will display data from 25th line to bottom)

Note: To see on-growing logs we can use '-f' option

$ tail -f data.log (Live log message we can see)

-> It is used to count no.of lines, no.of words and no.of characters in the file

$ wc f1.txt

-> To copy the data from one file to another file

$ cp one.txt two.txt ( or ) $ cat one.txt > two.txt

$ cp f1.txt f2.txt f3.txt (invalid syntax)

-> We can't copy morethan one file data using 'cp' command. To copy multiple files data we should go for 'cat' command

**$ cat f1.txt f2.txt > f3.txt**

**-> To rename files we will use 'mv' command**

**$ mv f1.txt f1111.txt**

**$ mv  dirname dirnewname**

**Note: We can use 'mv' command for renaming and moving files**

**-> To compare files we can use below commands**

**$ cmp f1.txt f2.txt**

**$ diff f1.txt f2.txt**

**-> cmp command will display only first difference in given 2 two files**

**-> diff command will display all the differences in the content**

**-> 'grep' stands for global regular expression print.**

**-> 'grep' command will process the text line by line and prints any lines which matches given pattern.**

**Ex: I want to print all lines which contains 'NullPointerException'**

**$ grep -i 'NullPointerException'  ***

**Note: We can install grep using below command**

**$ sudo yum install grep**

**//search for the lines which contains given word in the given filename**

**$ grep 'word' filename**

**//search for the lines which are having exception keyword in server.log file**

**$ grep -i 'exception' server.log**

**//search for the given text in present directory and in sub-directories also**

**$ grep -R 'exception'**

**=> We can pass several options for 'grep' command**

**-c : This prints only the count of files that matches give pattern**

**-i : ignore case-sensitivity**

**-n : Display the matched lines and their line numbers**

**-l : Displays only file names that matches the pattern**

**-h : Displays matched lines without file names**

## Working with Text Editors in Linux

-> The default editor that comes with the LNIX operating system is called vi (visual editor).

-> Using vi editor, we can edit an existing file or we create a new file from scratch

-> We can also use this vi editor to just read a text file.

Modes of Operation in vi editor

There are three modes of operations in vi:

1) command mode

2) insert mode

3) escape mode

### Command Mode:

When vi starts up, it is in Command Mode. This mode is where vi interprets any characters we type as commands and thus does not display them in the window

This mode allows us to move through a file, and to delete, copy, or paste a piece of text.

To enter into Command Mode from any other mode, it requires pressing the [Esc] key. If we press [Esc] when we are already in Command Mode, then vi will beep or flash the screen.

### Insert mode:

This mode enables you to insert text into the file.

Everything that's typed in this mode is interpreted as input and finally, it is put in the file.

The vi always starts in command mode. To enter text, you must be in insert mode.

To come in insert mode you simply type i.

To get out of insert mode, press the Esc key, which will put you back into command mode.

### Last Line Mode (Escape Mode):

Line Mode is invoked by typing a colon [:], while vi is in Command Mode.

The cursor will jump to the last line of the screen and vi will wait for a command.

This mode enables you to perform tasks such as saving files, executing commands.

There are following way you can start using vi editor :

Commands and their Description:

vi filename: Creates a new file if it already not exist, otherwise opens existing file.

vi -R filename : Opens an existing file in read only mode.

**view filename : Opens an existing file in read only mode.**

**$ vi f1.txt**

**=> After making changes if we want to save those changes then execute :wq**

**=> After making changes if we don't want to save those changes then execute :q!**

**Moving within a File(Navigation):**

**To move around within a file without affecting text must be in command mode (press Esc twice). Here are some of the commands can be used to move around one character at a time.**

**Commands and their Description:**

**k : Moves the cursor up one line.**

**j : Moves the cursor down one line.**

**h : Moves the cursor to the left one character position.**

**l : Moves the cursor to the right one character position.**

**0 or | : Positions cursor at beginning of line.**

**$ : Positions cursor at end of line.**

**W : Positions cursor to the next word.**

**B : Positions cursor to previous word.**

**( : Positions cursor to beginning of current sentence.**

**) : Positions cursor to beginning of next sentence.**

**H : Move to top of screen.**

**nH : Moves to nth line from the top of the screen.**

**M : Move to middle of screen.**

**L : Move to bottom of screen.**

**nL : Moves to nth line from the bottom of the screen.**

**colon along with x : Colon followed by a number would position the cursor on line number represented by x.**

**Usecase:**

**-> We will use 'vi' editor to perform below activities**

**a) To edit config files of our application**

**b) To edit shell script files**

## SED command:

**SED command in LNIX stands for stream editor and it can perform lots of functions on file like searching, find and replace, insertion or deletion.**

**Though most common use of SED command in LNIX is for substitution or for find and replace.**

**By using SED you can edit files even without opening them, which is much quicker way to find and replace something in file, than first opening that file in VI Editor and then changing it.**

**SED is a powerful text stream editor. Can do insertion, deletion, search and replace (substitution).**

**SED command in lnix supports regular expression which allows it perform complex pattern matching.**

**Example:**

**$ cat > myfile.txt**

**unix is great os. unix is opensource. unix is free os.**

**learn operating system.**

**unix linux which one you choose.**

**unix is easy to learn.unix is a multiuser os. Learn unix. unix is a powerful.**

**Replacing or substituting string :**

**Sed command is mostly used to replace the text in a file. The below simple sed command replaces the word “unix” with “linux” in the file**

**$ sed 's/unix/linux/' myfile.txt**

**By default, the sed command replaces the first occurrence of the pattern in each line and it won’t replace the second, third…occurrence in the line.**

**Replacing the nth occurrence of a pattern in a line :**

**Use the /1, /2 etc flags to replace the first, second occurrence of a pattern in a line. The below command replaces the second occurrence of the word “unix” with “linux” in a line.**

**$sed 's/unix/linux/2' geekfile.txt**

**Replacing all the occurrence of the pattern in a line :**

**The substitute flag /g (global replacement) specifies the sed command to replace all the occurrences of the string in the line.**

**$ sed 's/unix/linux/g' myfile.txt**

**Deleting lines from a particular file**

**SED command can also be used for deleting lines from a particular file. SED command is used for performing deletion operation without even opening the file**

**To Delete a particular line say n in this example**

**$ sed '5d' myfile.txt**

**-> To Delete a last line : $ sed '$d' myfile.txt**

**-> To Delete from nth to last line : $ sed '12,$d' myfile.txt**

**Note: By default SED command changes will not store in file.**

**=> To make SED command changes to file permanently we will use '-i' option.**

**$ sed -i 's/unix/linux/g' myfile.txt**

**Note: With above command 'unix' keyword will be replaced with 'linux' keyword in the file permanently.**

## Linux File permissions:

**To create a secure environment in Linux, you need to learn about user groups and permissions. For example, if you work in a company and you want the finance department to read a file but not make any modification to it, then you need to use permissions in Linux. It is a must for every programmer working with Linux nowadays.**

**Let's start by talking about the ownership of Linux files.**

**User: the owner of the file (person who created the file).**

**Group: the group can contain multiple users.**

**Therefore, all users in that group will have the same permissions. It makes things easier than assign permission for every user you want.**

**Other: any person has access to that file, that person has neither created the file, nor are they in any group which has access to that file.**

**We will work with this part "-rw-r–r–".**

**The characters mean:**

**'r' = read.**

**'w' = write.**

**'x' = execute.**

**'-' = no permission.**

**-rw-r--r--**

**-: It represents file**

**rw: User**

**r: Group**

**r: Other**

**As we see above, the empty first part means that it is a file. If it were a directory then it will be the letter "d" instead.**

**The second part means that the user "Home" has read and write permissions but he does not have the execute one.**

**The group and others have only the read permission.**

**Let's change the permissions using the chmod command.**

**$ chmod o+w section.txt**

**This command will add the write permission for other users to my text file "section.txt".**

**Now if you try to execute ls -l then you will see -rw-r--rw-**

**"o" refers to others, "g" for the group, "u" for the user, and "a" for all.**

**Now let's add the execute permission to the user with:**

**$ chmod u+x section.txt**

**The permissions will be -rwxr--rw-**

**If you want to remove the permission, you can use the same method but with "-" instead of "+".**

**For example, let's remove the execute permission from the user by:**

**$ chmod u-x section.txt**

**And the permissions now are: -rw-r--rw-**

**Also, you can use Symbolic Mode to modify permissions like the following:**

| Number | Permission |
|---|---|
| 0 | No permission |
| 1 | Execute |
| 2 | Write |
| 3 | Execute and Write |
| 4 | Read |
| 5 | Read and Execute |
| 6 | Read and Write |
| 7 | Read, Write and Execute |

**For example, let's give every permission for all with:**

**$ chmod 777 section.txt**

**Then the permissions will be: -rwxrwxrwx.**

**Let's remove the execute from the group and the write from other by:**

**$ chmod 765 section.txt**

**Then the permission will be : -rwxrw-r-x**

## Working with User Accounts

**-> Linux is multi user based operating systems**

**-> Within one Linux machine we can create multiple user accounts**

**-> Multiple users can access single linux machine and can perform multi-tasking**

**-> When we launch EC2 instance with Amazon Linux OS, we got by default 'ec2-user' user account**

**Create a user:**

**$ sudo useradd <uname>**

**-> After creating user account we can verify user account details using 'id' command**

**$ id <uname>**

**-> 'id' command will display user account information**

**-> Check the files available in home directory**

**$ ls -l**

**(We can see ec2-user folder and newly created user folder)**

**(That means we have 2 user accounts in our machine)**

**-> After creating a new user and setting a password to it, you can log in from the terminal**

**$ su - <uname>**

**Delete a user:**

**$ sudo userdel <uname>**

**If you try that command, you will notice that the user directory has not been deleted and you need to delete it by yourself.**

**You can use this automated command to do everything for you:**

**$ sudo userdel <uname>**

## User groups:

-> A group is a collection of users.

-> The primary purpose of the groups is to define a set of privileges like read, write, or execute permission for a given resource that can be shared among the users within the group.

Create a group:

You can see all of the groups you have by opening the following file:

$ cat /etc/group

$ sudo groupadd <groupname>

Add user to a group:

$ sudo usermod -aG <group-name> <username>

-> verify user groups using command

$ id username

Delete user from a group:

$ sudo gpasswd -d <username> <groupname>

Delete a group:

$ sudo groupdel <groupname>

## Working with 'find' and 'locate' commands

-> find and locate commands are used to search files in linux machines

locate command:

The locate Command find will search for data in local db

$ sudo yum install mlocate

$ locate apache

$ locate -c apache

$ locate -c *.txt

$ locate -S (to see locate database)

Note: when we create new files it will take some time to update those files in mlocate db

find command:

-> find command will search for the files in entire linux file system.

-> find command providing advanced searching technique

-> Using find command, we can search for the files based on name and type also.

**2. Find Files Under Home Directory**

**# find /home -name f1.txt**

**7. Find Files With 777 Permissions**

**# find . -type f -perm 0777 -print**

**19. Find all Empty Files**

**# find /home -type f -empty**

**20. Find all Empty Directories**

**# find /home -type d -empty**

**Note: As find command is scanning entire file system, it will take more time to give search results.**

## Chown command:

**-> The chown command changes user ownership of a file & directory in Linux**

**-> Every file is associated with an owning user or group**

**-> We can check the ownership of a file using 'ls -l'**

**# changing owner of a file**

**$ chown NewUser FILE**

**Note: we can change owner using userid also**

**$ chown 1001 filename**

**# changing group of a file**

**$ chown :NewGroup FILE**

**# We can change group using group id also**

**$ chown :1003 sample**

**# Change Owner and the Group**

**$ chown NewUser:NewGroup FILE**

**-> 'man' command is like a help command. It is used to understand command syntax and options.**

**$ man cat**

**-> To see ip address we will use 'ifconfig' command**

**$ ifconfig**

**-> 'ping' command is used to check connectivity**

**$ ping <ip>**

**-> 'curl' command is used to get response from the server**

**$ curl <url>**

**-> 'wget' command is used to download resources from internet**

**$ wget <url>**

## How to install Software in Linux Machine:

**-> In linux we have package manager to install a software**

**Note: Amazon Linux / CentOS / Red Hat : yum is the package manager**

**Note : Ubuntu : apt-get is the package manager**

**# update existing packages in linux**

**$ sudo yum update -y**

**# install git client**

**$ sudo yum install git**

**# check git version**

**$ git --version**

**# install maven**

**$ sudo yum install maven**

**# check maven version**

**$ mvn –version**

**# install java software**

**$ sudo yum install java**

**# install Java 1.8 version**

**$ sudo yum install java-1.8.0-openjdk**

## Installing Web Server in Linux VM (Amazon Linux):

**1) Connect to EC2 instance**

**2) Execute below commands to install Apache server (httpd)**

**$ sudo yum update -y**

**$ sudo yum install httpd**

**$ service httpd start**

**Note: Check accessing apache server from browser using EC2 VM public IP**

**-> If server is not accessible then, enable HTTP port in Security Group of our EC-2 VM**

**-> After adding security group try accessing EC2 instance using IP**

**Ex: http://52.66.101.3/**

**-> You can modify html page content**

**$ cd /var/www/html**

**$ sudo vi index.html**

**<h1> Welcome to Ashok IT - Website</h1>**

**-> Save the content and close the file ( press Esc -> :wq)**

**-> Now access public IP in browser**

## Shell Scripting

### What is shell?

**-> Shell is responsible for reading commands given by user**

**-> Shell will verify command and will give instructions to kernel to process that command**

**-> If command is invalid shell will give error**

**-> Kernel will execute our command with System Hard Components**

**-> Shell acts as mediator between User and Kernel**

### What is Scripting?

**-> Scripting means set of commands mentioned in a file for execution**

**-> Scripting is used to automate our routine work**

**-> For example i want to execute below commands every day as a linux user**

**$ date**

**$ cal**

**$ whoami**

**$ pwd**

**$ ls -l**

**-> Instead of executing all these commands manually we can keep them in a file and we can execute that file.**

**-> The file which contains set of commands for execution is called as 'Script file'**

## What is Shell Scripting?

**-> Shell Scripting is used to execute set of commands using a script file**

**-> When we execute script file then shell will read those commands and will verify commands syntax**

**-> Shell will give instructions to 'Kernel'**

**-> Kernel will give instructions to hardware components to perform actual operation**

### Types of Shells

**-> There are several shells available in linux OS**

**1) Bourne Shell**

**2) Bash Shell**

**3) Korn Shell**

**4) CShell**

**5) TShell**

**6) ZShell**

**# Display all the shells of our Linux machines**

**$ cat /etc/shells**

**# Display the default shell of our Linux machine**

**$ echo $SHELL**

**Note: The most commonly used shell in Linux in is 'BASH SHELL'.**

**Note: Shell Script files will have .sh extension**

## Working with First Shell Script Program

**# Create a script file**

**$ vi task.sh**

**whoami**

**pwd**

**date**

**-> Save the file and close it (ESC + :wq)**

**# Run the shell script (Option-1)**

**$ sh task.sh**

**Note: If we get permission denied then we should provide 'execute' permission using 'chmod' command**

**# Run the shell script (Option-2)**

**$ ./task**

## What is sha-bang in shell script?

**-> Sha-bang is used to specify which shell should be used to process our script**

**Syntax :**

**#! /bin/bash**

*****************Shell Script - 2***************

```
#! /bin/bash
echo "Welcome to Scripting"
echo "Scripting is used to automate regular work"
echo "Scripting requires lot of practise"
```

******************Shell Script - 3 ************************

```
#! /bin/bash
echo "Enter your name:"
read name
echo "Good Morning $name"
```

********** Shell Script - 4 **********

```
#! /bin/bash
a=10
b=20
c=$(($a + $b))
echo "Sum of $a and $b is =  $c"
```

********** Shell Script - 5 **********

```
#! /bin/bash
echo "Enter First Number"
read a
echo "Enter Second Number"
read b
c=$(($a + $b))
echo "Sum of $a and $b is =  $c"
END
```

## Variables

-> Variables are place-holders to store the value

-> Varibles are key-value pairs

-> In Shell Scripting there is no concept of Data Type.

-> Every value will be treated as text/string

Ex:

```
name=ashok
age=30
email=ashokitschool@gmail.com
phno=1234
```

-> Variables are divided into 2 types

1) Environment Variables or System variables

2) User Defined Variables

-> The variables which are already defined and using by our system are called as Environment/System variables

**Ex:**

**$ echo $USER**

**$ echo $SHELL**

**-> Based on our requirement we can define our own variables those are called as user defined variables**

**Ex:**

**name=ashok**

**age=30**

**$echo $name $ age**

## Variable Rules

**-> We should not use special symbols like -, @, # etc....**

**-> Variable name should not start with digit**

**Note: It is recommended to use uppercase characters for variable name**

**-> we can use 'readonly' for variable so that variable value modification will not be allowed**

## Command Line Arguments

**-> The arguments which will pass to script file at the time of execution**

**-> Cmd args are used to supply the values dynamically to the script file**

**Ex:**

**$ sh demo.sh ashokit 30**

**-> We can access cmd args in script file like below**

**$# - no.of args**

**$0 - script file name**

**$1 - First Cmd Arg**

**$2 - Second Cmd Arg**

**$3 - Third Cmd Arg**

**$* - All Cmd Args**

**-> To comment any single line we can use '#'**

**echo 'hi'**

**#echo 'hello'**

-> We can comment multiple lines also in script file like below

```
<<COMMENT
        .....................
COMMENT
```

-> We can hold script execution for some time using 'sleep' command

```
#! /bin/bash
echo $#
echo $0
echo $1
sleep 30s
echo $2
#echo $*
```

## Conditional Statements

-> Conditional statements are used to execute commands based on condition

Syntax:

```
if [ conition ]
then
        stmts
else
      stmts
fi
```

-> If given condition satisfied then if statments will be executed otherwise else statements will be executed

```
if [ condition ]
then
        stmts
elif [ condition ]
then
        stmts
```

```
else
        stmts
fi
```

Ex:

```
#!/bin/bash
echo "Enter Your Favorite Color"
read COLOR
if [ $COLOR == 'red' ]
then
    echo "Your are cheerful"
elif [ $COLOR == 'blue' ]
then
    echo "You are joyful"
else
    echo "You are lucky"
fi
```

## Working with loops

-> Loops are used to execute stmts multiple times

-> We can use 2 types of loops

1) Range based loop ( ex: for loop )

2) Conditional based loop ( ex: while loop )

## For Loop Example

```
#! /bin/bash
for ((i=1; i<=10; i++))
do
echo "$i"
done
```

## While loop Example

```
#! /bin/bash
i=10
while [ $i -ge 0 ]
do
echo "$i"
let i--;
done
```

## Infinite Loop

```
#! /bin/bash
while true
do
echo "This is my loop stmt"
done
```

**Note: To stop infinite loop we will use 'ctrl + c'**

## Functions

-> The big task can be divided into smaller tasks using functions

-> Function is used to perform an action / task

-> Using functions we can divide our tasks logically

-> Functions are re-usable

Syntax:

```
function functionaName( ) {
    // commands to execute
}
```

### Writing welcome function

```
#! /bin/bash
function welcome(){
 echo "Welcome to functions...";
 echo "I am learning Shell Scripting";
 echo "Shell Scripting is used to automate our regular work";
}
# call the function
welcome
```

### Function with Parameters

```
#! /bin/bash
function welcome ( ) {
    echo "$1";
}
# call function
welcome Linux
welcome AWS
welcome DevOps
```