**STARTED on 19 Jan 2020**

**STARTED on 19 Jan 2020**

**26 Jan 2020**

Vagrant init

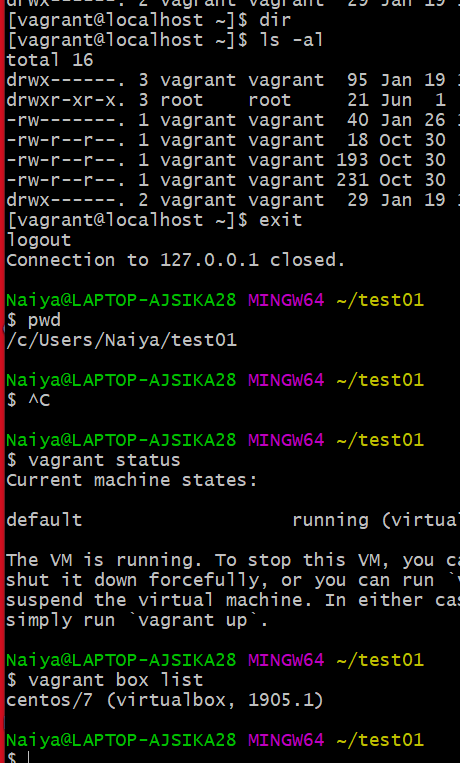
Vagrant status

Vagrant SSH

Sudo yum update -y

Vagrant init centos/7

Vagrant box list



C:\>cd Users/Naiya/test01

C:\Users\Naiya\test01>pwd

'pwd' is not recognized as an internal or external command,

operable program or batch file.

C:\Users\Naiya\test01>vagrant status

Current machine states:

default running (virtualbox)

The VM is running. To stop this VM, you can run `vagrant halt` to

shut it down forcefully, or you can run `vagrant suspend` to simply

suspend the virtual machine. In either case, to restart it again,

simply run `vagrant up`.

C:\Users\Naiya\test01>ls

'ls' is not recognized as an internal or external command,

operable program or batch file.

C:\Users\Naiya\test01>ls -a

'ls' is not recognized as an internal or external command,

operable program or batch file.

C:\Users\Naiya\test01>vagrant box list

centos/7 (virtualbox, 1905.1)

C:\Users\Naiya\test01>vagrant ssh

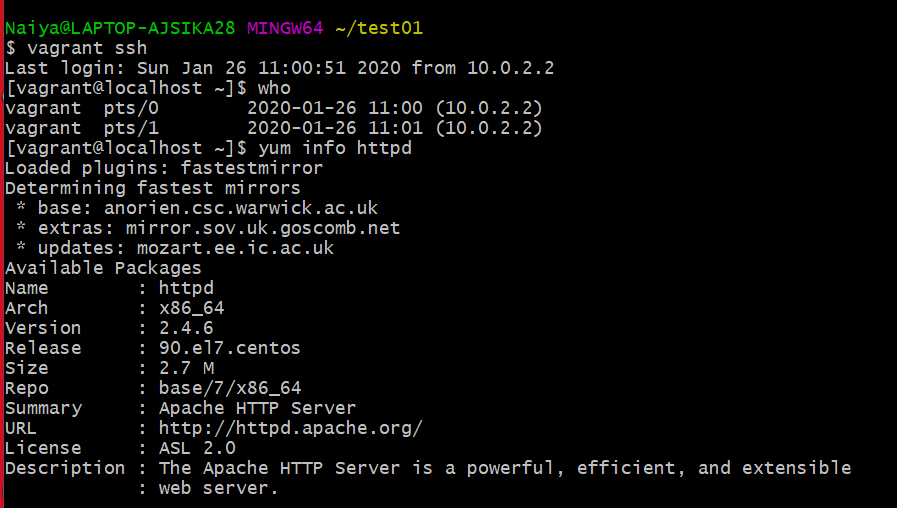
Last login: Sun Jan 26 10:35:31 2020 from 10.0.2.2

Last login: Sun Jan 26 10:35:31 2020 from 10.0.2.2

[vagrant@localhost ~]$ who

vagrant pts/0 2020-01-26 11:00 (10.0.2.2)

vagrant pts/1 2020-01-26 11:01 (10.0.2.2)



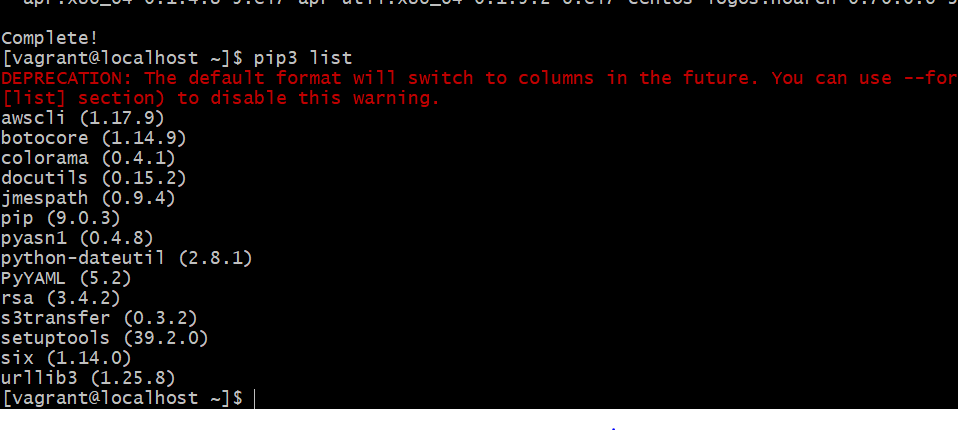
YUM : YELLOWDOG UPDATE MANAGER

Web servers: httpd (apache and nginx) , we can find information (yum info httpd)

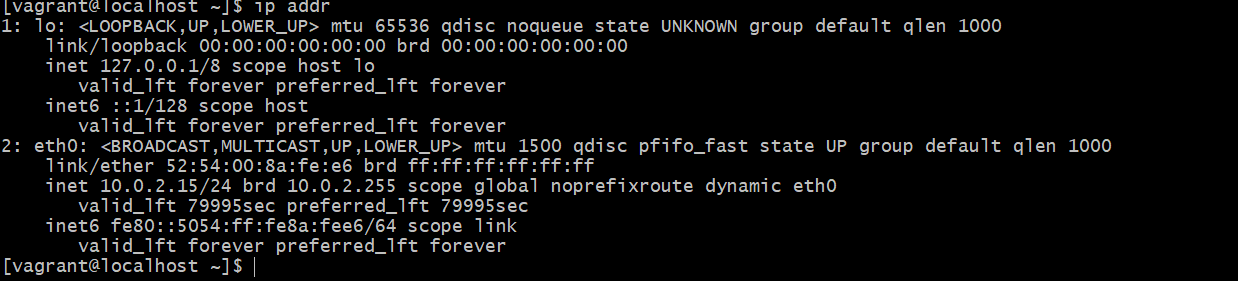
Repos: it is stored lib information: Source code information or binary files.

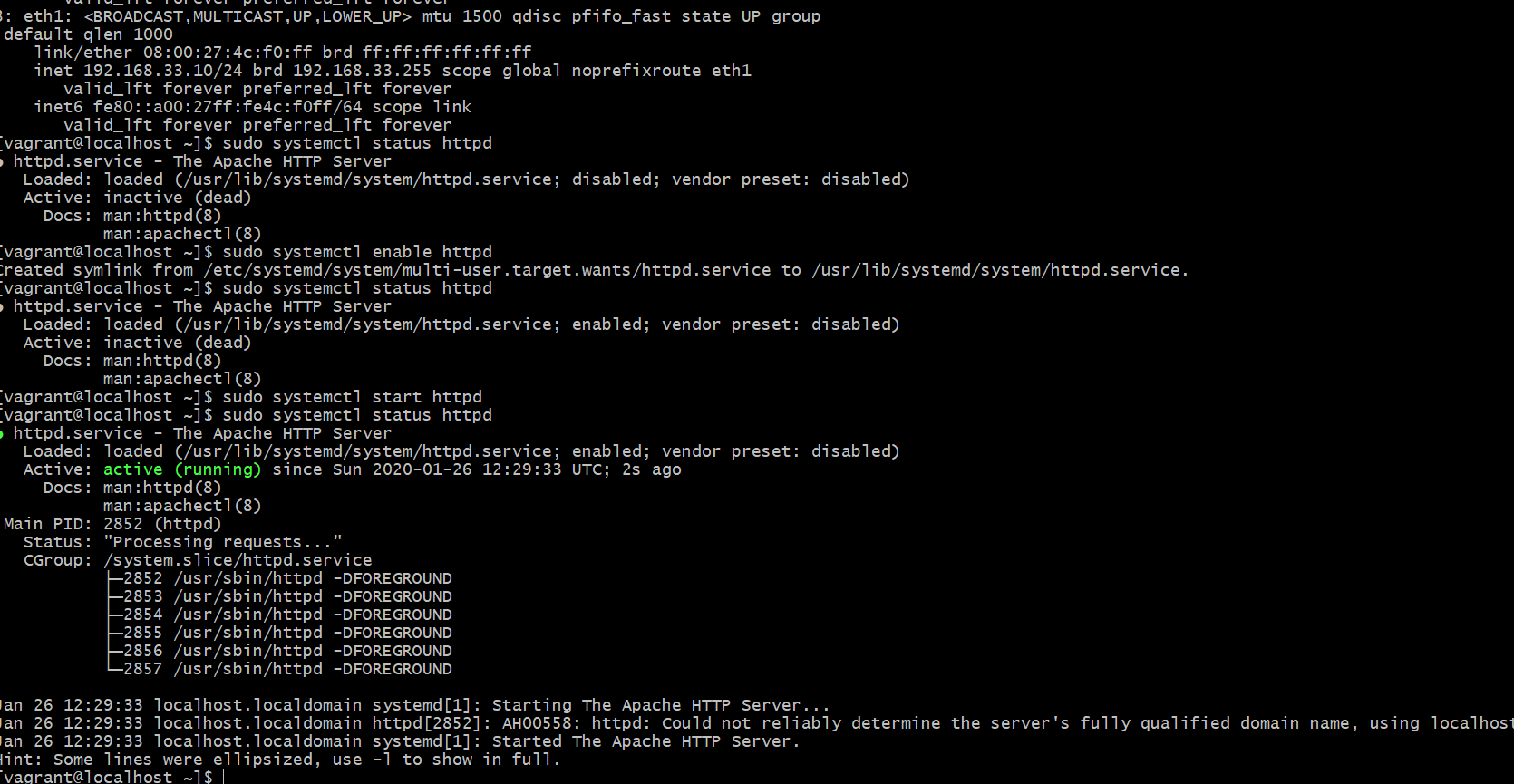
Sudo (super user do) to be a root user.

**Pip3 list** shows all the packages are here.



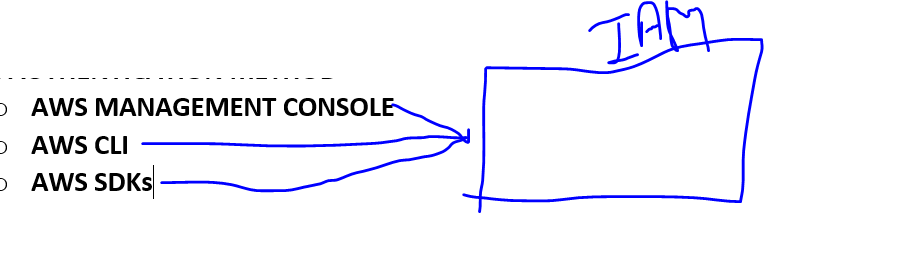
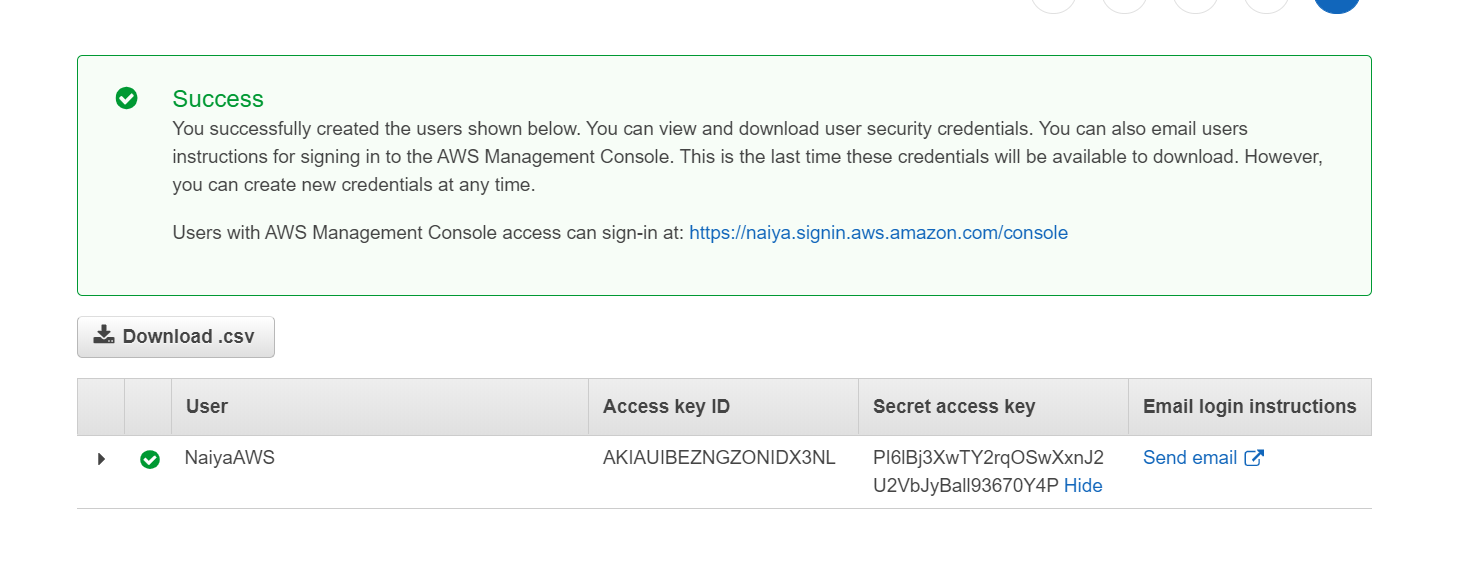
Ip address



* DNS – (IP to NAME) :> NAT (Network Address Translator
* Vagrant halt and then we have to an another Ip for the local network
* We open the vagrant file in visual code and made 2 uncommented and then it was made a real network availability.
* Then start vagrant up
* Vagrant ssh
* Vagrant status
* Ip addr
* **Linux session (26 jan) : slides**
* What is shell
* BIOS + MBR + GRUB+KERNAL + INIT(compare to SYSTEMD) + RUN LEVEL(TARGETS)
* SystemD is latest version of INIT and (replaces with Sysvinit (redhat) & upstart)
* Kernal/OS/SHELL/USER
* 
* If you don’t enable then every time you have to start the server/ but if you have to keep active the server then you have to do enable.
* <http://127.0.0.1:8000/>
* <http://192.168.33.10/>

**FEB 2nd, 2020 Sunday Session**

**Cloud practitioners and consultant**

* **Cloud computing : http: AWS /cloud computing**
* **“CC” is flexible and low cost IT**
* **HA/ Disaster recovery/ Fault Tolerance**
* **Advantages: Go Global, Capacity, increase speed & Agility, maintain & Running datacentres with less cost.(Trade Capital expense for Variable expense)**
* **Scale up / Down**
* **EC2**
* **Types of CC: IAAS, PAAS, SAAS**
* **IAAS: Network+ storage + servers + virtual + O/S+ Middleware + Runtime+ data + APPS**
* **PAAS: data + APPS**
* **SAAS: like a google(as a user)**
* **What is AWS? (FORMULA 1, SAP) (Top cloud prov)**
* **AWS Global Infrastructure: Regions + AZs.**
* **Regions provide multiple, physically separated and isolated AZ which are connected with LOW LATENCY, HIGH THROUGHPUT and HIGHLY REDUNDANT NETWORK.**
* **AZ offers an easier way to design & operate app and databases, making them more HA, FT, and scalable as compare to single/multiple datacentres**
* **69 = AZ**
* **22 = Geo Regions**
* **1 = Local Region**
* **EDGE LOCATIONS (Single or multiple locations**
* **REGIONAL EDGE CACHES**
* **CDN : CONTENT DELIVERY NETWORK**
* **13 = AZ and 4 Regions = coming soon**
* **Shared Responsibility Model**
* **AWS GLOBAL Infra : Region + az = edge location**
* **Services : COMPUTE , STORAGE, DATABASE, Networking**
  + **Client side data Encryption & Data Integrity Authentication**
  + **Server Side Encryption (file system &/ data)**
  + **Network Traffic Protection(Encryption/Integrity/ Identity)**
* **(NOTE : physical server side people will not have a logical access and logical people will not have an access to physical )**
* **AWS is responsible OF the Cloud Security)**
* **OS, Network & Firewall Config**
* **Platform , APP, Identity & Access Management**
  + **Authenticate & Authorization**
  + **Authenticate -🡪 (Signed in)**
  + **Authorization 🡪 (Has Permissions)**
* **Customer data**
* **Console login**
* **Root**
* **MFA ( GOOGLE APP : authenticator)**
* **Password Policy**
* **Permission & Perveledges**
  + **USER ARE REPACED BY RBAC: ROLE BASED ACCESS CONTROL SYSTEM (user based access,**
* **IAM USE :** 
  + **USERS**
  + **GROUPS**
  + **ACCESS POLICIES**
  + **ROLES**
  + **USER CREDENTIALS**
  + **USER PASSWORD POLICIES**
  + **MFA**
  + **API KEY FOR PRAGRAMMATIC ACCESS :**
* **IAM AUTHENTICATION METHOD : they must need access keys** 
  + **AWS MANAGEMENT CONSOLE**
  + **AWS CLI**
  + **AWS SDKs**
  + **iam users and groups**
  + **FOR SDKs we do need an API Keys ()**
* **[vagrant@localhost ~]$ pip3 list**
  + - * + **DEPRECATION: The default format will switch to columns in the future. You can use --format=(legacy|columns) (or define a format=(legacy|columns) in your pip.conf under the [list] section) to disable this warning.**
        + **awscli (1.17.9)**
        + **botocore (1.14.9)**
        + **colorama (0.4.1)**
        + **docutils (0.15.2)**
        + **jmespath (0.9.4)**
        + **pip (9.0.3)**
        + **pyasn1 (0.4.8)**
        + **python-dateutil (2.8.1)**
        + **PyYAML (5.2)**
        + **rsa (3.4.2)**
        + **s3transfer (0.3.2)**
        + **setuptools (39.2.0)**
        + **six (1.14.0)**
        + **urllib3 (1.25.8)**
* **AWS(IAM ) Authorization :**
  + **Full Access -> S3 Bucket**
  + **Read Only Access ->EC2 Instance**
* **Root user and Root Privilege user** 
  + **Create user with no permissions**
  + 

**Copy the console link into the google incognito window and check the privileges**

* **Go back to the root user screen and give some permissions**
* <https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#Instances:sort=instanceId> (IAM USER – NaiyaAWS)
* <https://console.aws.amazon.com/iam/home?region=us-east-1#/users/NaiyaAWS> (Root User)

**09 Feb 2020 Sunday (Devops)**

* **Amazon Global services : IAM and Route 53**
* **NOTE: Don’t use the root user for everyday tasks.**
* **AWS IAM is web service**
* **List privileged policy, which deny everything**
* **Orange box policies are default policies.**
* **Region: 🡪 AZ (min 2 and max – 6) in each region**
* **AWS – VPC –AZ--Subnet**
* **CIDR : Classic Inter-domain Routing**

AWS

REGION

VPC

**We are using Cloud platform to run the Server**

* **Continue with in Cognito window (IAM User (Not the root user)**
* **EC2 instance**
* Step 1: Choose an Amazon Machine Image (AMI)**Cancel and Exit**
  + - An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.
* **Amazon Linux 2 AMI (HVM), SSD Volume Type** - ami-02ccb28830b645a41 (64-bit x86) / ami-0adab12dcddc251ea (64-bit Arm)
  + Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.
  + Root device type: ebs Virtualization type: hvm
  + ENA Enabled: Yes
* **Hardware Virtualization Machine (\*HVM)**
* Step 2: Choose an Instance Type : Next: Configure Instance Details
* Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](https://aws.amazon.com/ec2/instance-types/) about instance types and how they can meet your computing needs.
* **Type of t2micro**
* **T =**
* **C = CPU**
* **R = RAM**
* **M =**
* **I =**
* **EBS : Elastic Block Storage :**
* **Instance : Pay as you go**
* **Subnets are per region**
* **VPC spans in the AZ**
* **Cloud watch -> monitoring be default -> 5 mins (free) or 6 sec( paid)**
* **Tenancy -> Shared/ Dedicated/ dedicated host (read from websites)**
* **General purpose / provisioned Iops / 🡪**
* **Magnetic(ephemeral) -> will lost the data**
* Step 5: Add Tags
* A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.  
  A copy of a tag can be applied to volumes, instances or both.  
  Tags will be applied to all instances and volumes. [Learn more](https://docs.aws.amazon.com/console/ec2/tags) about tagging your Amazon EC2 resources.
* Step 6: Configure Security Group
* A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](https://docs.aws.amazon.com/console/ec2/security-groups) about Amazon EC2 security groups.
* Step 7: Review Instance Launch
* Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.
* 3.15.6.81
* Connecting to the Linux Virual Machine and go to the same directory and RUN the following :
* **ls \*.pem**

**Then run the following command**

* **$ ssh -i testec2.pem** [**ec2-user@3.15.6.81**](mailto:ec2-user@3.15.6.81)

**26 Jan 2020**

Vagrant init

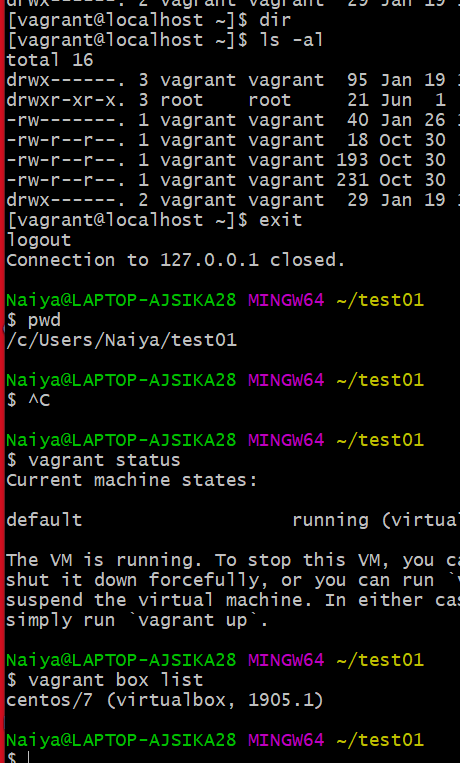
Vagrant status

Vagrant SSH

Sudo yum update -y

Vagrant init centos/7

Vagrant box list



C:\>cd Users/Naiya/test01

C:\Users\Naiya\test01>pwd

'pwd' is not recognized as an internal or external command,

operable program or batch file.

C:\Users\Naiya\test01>vagrant status

Current machine states:

default running (virtualbox)

The VM is running. To stop this VM, you can run `vagrant halt` to

shut it down forcefully, or you can run `vagrant suspend` to simply

suspend the virtual machine. In either case, to restart it again,

simply run `vagrant up`.

C:\Users\Naiya\test01>ls

'ls' is not recognized as an internal or external command,

operable program or batch file.

C:\Users\Naiya\test01>ls -a

'ls' is not recognized as an internal or external command,

operable program or batch file.

C:\Users\Naiya\test01>vagrant box list

centos/7 (virtualbox, 1905.1)

C:\Users\Naiya\test01>vagrant ssh

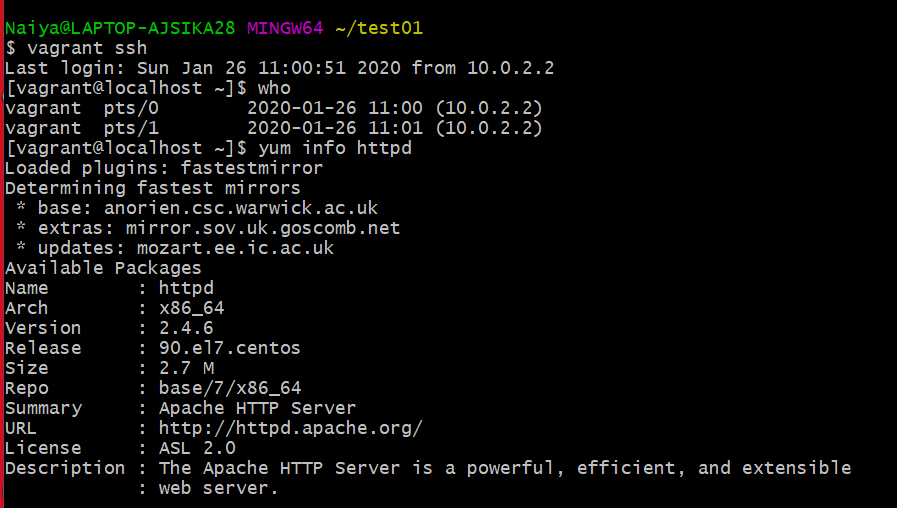
Last login: Sun Jan 26 10:35:31 2020 from 10.0.2.2

Last login: Sun Jan 26 10:35:31 2020 from 10.0.2.2

[vagrant@localhost ~]$ who

vagrant pts/0 2020-01-26 11:00 (10.0.2.2)

vagrant pts/1 2020-01-26 11:01 (10.0.2.2)



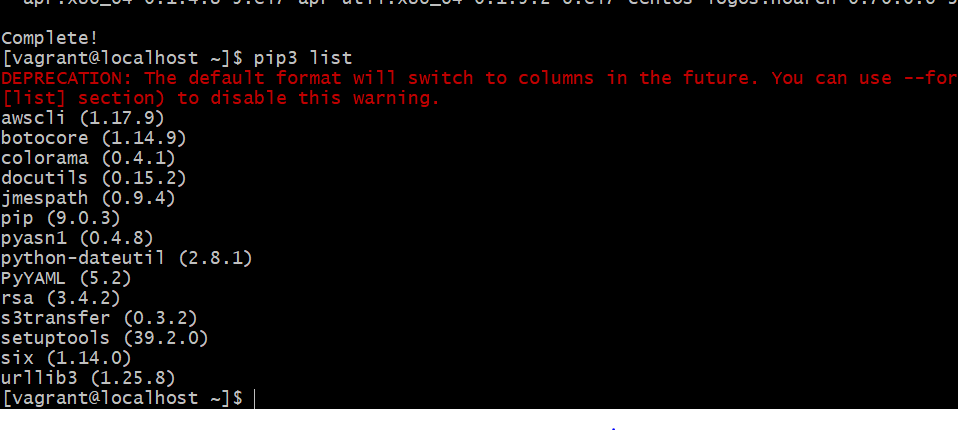
YUM : YELLOWDOG UPDATE MANAGER

Web servers: httpd (apache and nginx) , we can find information (yum info httpd)

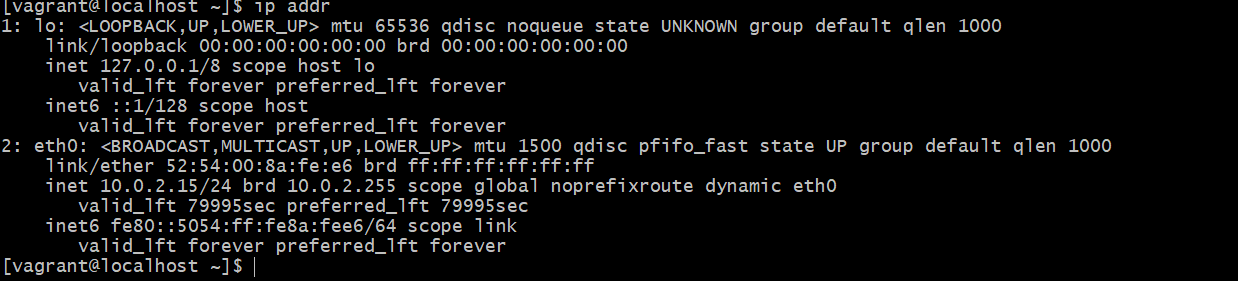
Repos: it is stored lib information: Source code information or binary files.

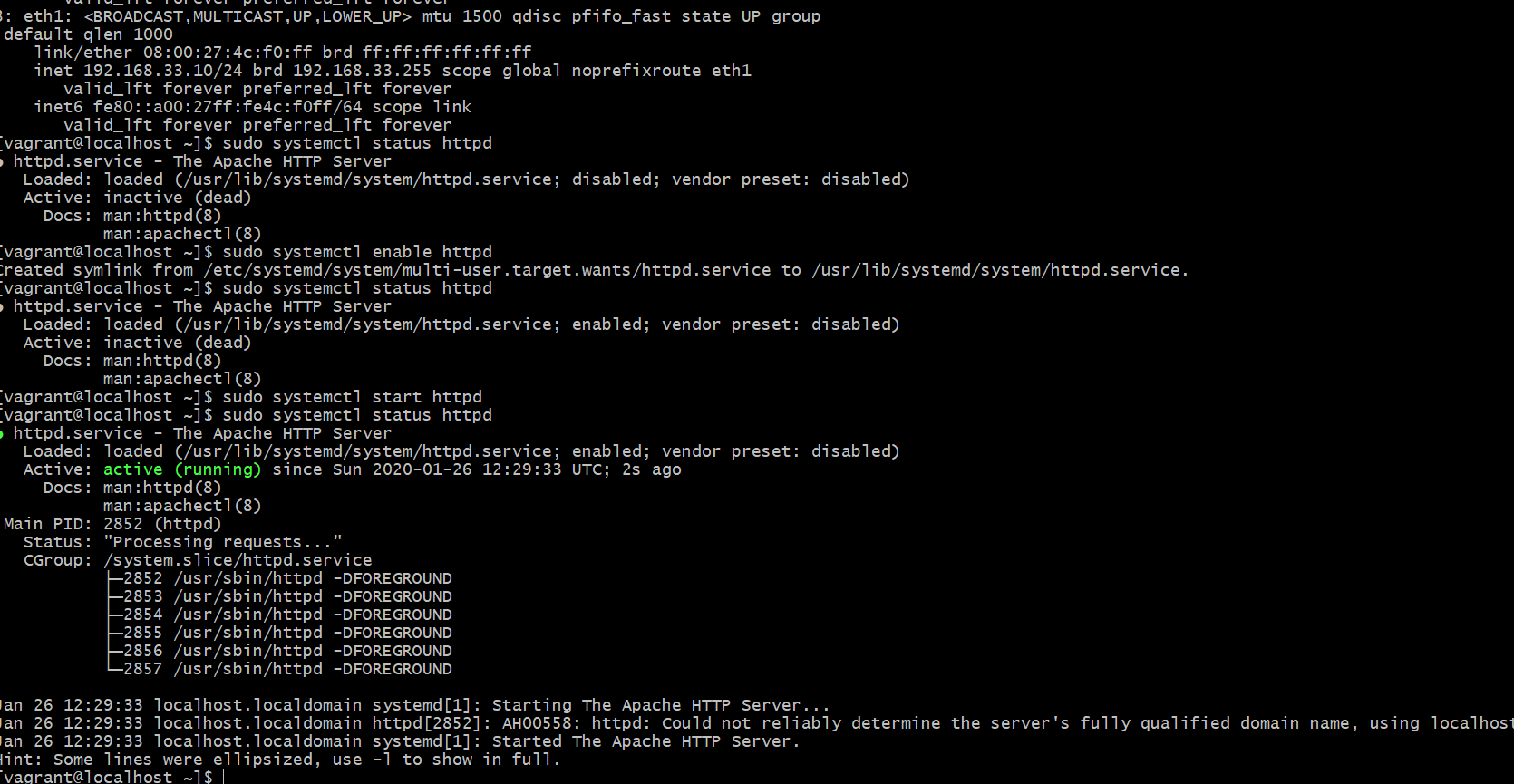
Sudo (super user do) to be a root user.

**Pip3 list** shows all the packages are here.



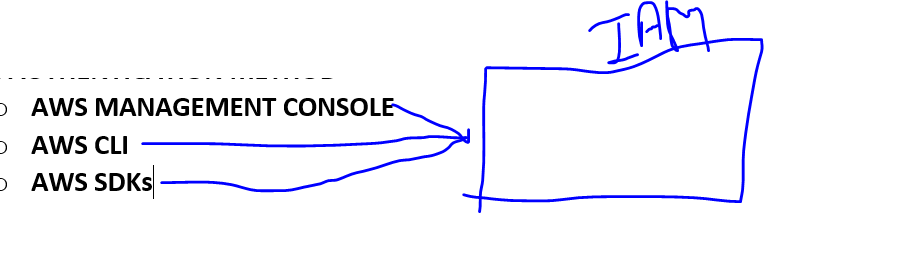
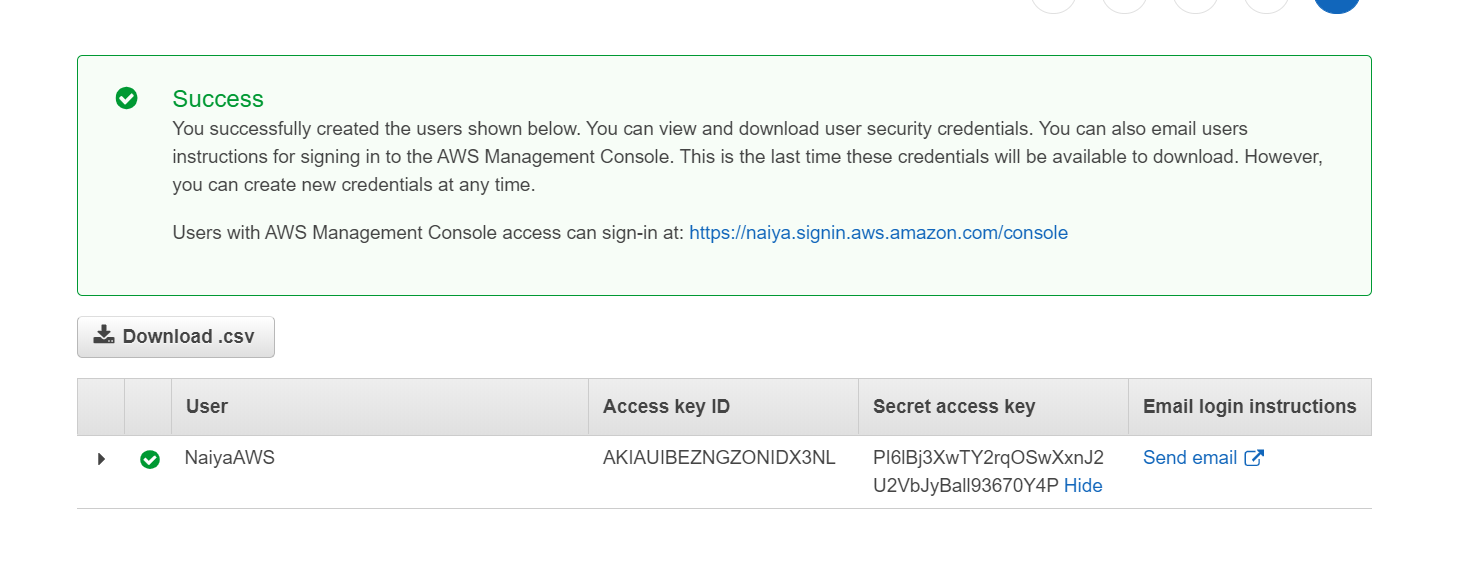
Ip address



* DNS – (IP to NAME) :> NAT (Network Address Translator
* Vagrant halt and then we have to an another Ip for the local network
* We open the vagrant file in visual code and made 2 uncommented and then it was made a real network availability.
* Then start vagrant up
* Vagrant ssh
* Vagrant status
* Ip addr
* **Linux session (26 jan) : slides**
* What is shell
* BIOS + MBR + GRUB+KERNAL + INIT(compare to SYSTEMD) + RUN LEVEL(TARGETS)
* SystemD is latest version of INIT and (replaces with Sysvinit (redhat) & upstart)
* Kernal/OS/SHELL/USER
* 
* If you don’t enable then every time you have to start the server/ but if you have to keep active the server then you have to do enable.
* <http://127.0.0.1:8000/>
* <http://192.168.33.10/>

**FEB 2nd, 2020 Sunday Session**

**Cloud practitioners and consultant**

* **Cloud computing : http: AWS /cloud computing**
* **“CC” is flexible and low cost IT**
* **HA/ Disaster recovery/ Fault Tolerance**
* **Advantages: Go Global, Capacity, increase speed & Agility, maintain & Running datacentres with less cost.(Trade Capital expense for Variable expense)**
* **Scale up / Down**
* **EC2**
* **Types of CC: IAAS, PAAS, SAAS**
* **IAAS: Network+ storage + servers + virtual + O/S+ Middleware + Runtime+ data + APPS**
* **PAAS: data + APPS**
* **SAAS: like a google(as a user)**
* **What is AWS? (FORMULA 1, SAP) (Top cloud prov)**
* **AWS Global Infrastructure: Regions + AZs.**
* **Regions provide multiple, physically separated and isolated AZ which are connected with LOW LATENCY, HIGH THROUGHPUT and HIGHLY REDUNDANT NETWORK.**
* **AZ offers an easier way to design & operate app and databases, making them more HA, FT, and scalable as compare to single/multiple datacentres**
* **69 = AZ**
* **22 = Geo Regions**
* **1 = Local Region**
* **EDGE LOCATIONS (Single or multiple locations**
* **REGIONAL EDGE CACHES**
* **CDN : CONTENT DELIVERY NETWORK**
* **13 = AZ and 4 Regions = coming soon**
* **Shared Responsibility Model**
* **AWS GLOBAL Infra : Region + az = edge location**
* **Services : COMPUTE , STORAGE, DATABASE, Networking**
  + **Client side data Encryption & Data Integrity Authentication**
  + **Server Side Encryption (file system &/ data)**
  + **Network Traffic Protection(Encryption/Integrity/ Identity)**
* **(NOTE : physical server side people will not have a logical access and logical people will not have an access to physical )**
* **AWS is responsible OF the Cloud Security)**
* **OS, Network & Firewall Config**
* **Platform , APP, Identity & Access Management**
  + **Authenticate & Authorization**
  + **Authenticate -🡪 (Signed in)**
  + **Authorization 🡪 (Has Permissions)**
* **Customer data**
* **Console login**
* **Root**
* **MFA ( GOOGLE APP : authenticator)**
* **Password Policy**
* **Permission & Perveledges**
  + **USER ARE REPACED BY RBAC: ROLE BASED ACCESS CONTROL SYSTEM (user based access,**
* **IAM USE :** 
  + **USERS**
  + **GROUPS**
  + **ACCESS POLICIES**
  + **ROLES**
  + **USER CREDENTIALS**
  + **USER PASSWORD POLICIES**
  + **MFA**
  + **API KEY FOR PRAGRAMMATIC ACCESS :**
* **IAM AUTHENTICATION METHOD : they must need access keys** 
  + **AWS MANAGEMENT CONSOLE**
  + **AWS CLI**
  + **AWS SDKs**
  + **iam users and groups**
  + **FOR SDKs we do need an API Keys ()**
* **[vagrant@localhost ~]$ pip3 list**
  + - * + **DEPRECATION: The default format will switch to columns in the future. You can use --format=(legacy|columns) (or define a format=(legacy|columns) in your pip.conf under the [list] section) to disable this warning.**
        + **awscli (1.17.9)**
        + **botocore (1.14.9)**
        + **colorama (0.4.1)**
        + **docutils (0.15.2)**
        + **jmespath (0.9.4)**
        + **pip (9.0.3)**
        + **pyasn1 (0.4.8)**
        + **python-dateutil (2.8.1)**
        + **PyYAML (5.2)**
        + **rsa (3.4.2)**
        + **s3transfer (0.3.2)**
        + **setuptools (39.2.0)**
        + **six (1.14.0)**
        + **urllib3 (1.25.8)**
* **AWS(IAM ) Authorization :**
  + **Full Access -> S3 Bucket**
  + **Read Only Access ->EC2 Instance**
* **Root user and Root Privilege user** 
  + **Create user with no permissions**
  + 

**Copy the console link into the google incognito window and check the privileges**

* **Go back to the root user screen and give some permissions**
* <https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#Instances:sort=instanceId> (IAM USER – NaiyaAWS)
* <https://console.aws.amazon.com/iam/home?region=us-east-1#/users/NaiyaAWS> (Root User)

**09 Feb 2020 Sunday (Devops)**

* **Amazon Global services : IAM and Route 53**
* **NOTE: Don’t use the root user for everyday tasks.**
* **AWS IAM is web service**
* **List privileged policy, which deny everything**
* **Orange box policies are default policies.**
* **Region: 🡪 AZ (min 2 and max – 6) in each region**
* **AWS – VPC –AZ--Subnet**
* **CIDR : Classic Inter-domain Routing**

AWS

REGION

VPC

**We are using Cloud platform to run the Server**

* **Continue with in Cognito window (IAM User (Not the root user)**
* **EC2 instance**
* Step 1: Choose an Amazon Machine Image (AMI)**Cancel and Exit**
  + - An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.
* **Amazon Linux 2 AMI (HVM), SSD Volume Type** - ami-02ccb28830b645a41 (64-bit x86) / ami-0adab12dcddc251ea (64-bit Arm)
  + Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.
  + Root device type: ebs Virtualization type: hvm
  + ENA Enabled: Yes
* **Hardware Virtualization Machine (\*HVM)**
* Step 2: Choose an Instance Type : Next: Configure Instance Details
* Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](https://aws.amazon.com/ec2/instance-types/) about instance types and how they can meet your computing needs.
* **Type of t2micro**
* **T =**
* **C = CPU**
* **R = RAM**
* **M =**
* **I =**
* **EBS : Elastic Block Storage :**
* **Instance : Pay as you go**
* **Subnets are per region**
* **VPC spans in the AZ**
* **Cloud watch -> monitoring be default -> 5 mins (free) or 6 sec( paid)**
* **Tenancy -> Shared/ Dedicated/ dedicated host (read from websites)**
* **General purpose / provisioned Iops / 🡪**
* **Magnetic(ephemeral) -> will lost the data**
* Step 5: Add Tags
* A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.  
  A copy of a tag can be applied to volumes, instances or both.  
  Tags will be applied to all instances and volumes. [Learn more](https://docs.aws.amazon.com/console/ec2/tags) about tagging your Amazon EC2 resources.
* Step 6: Configure Security Group
* A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](https://docs.aws.amazon.com/console/ec2/security-groups) about Amazon EC2 security groups.
* Step 7: Review Instance Launch
* Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.
* 3.15.6.81
* Connecting to the Linux Virual Machine and go to the same directory and RUN the following :
* **ls \*.pem**

**Then run the following command**

* **$ ssh -i testec2.pem** [**ec2-user@3.15.6.81**](mailto:ec2-user@3.15.6.81)

**26 Jan 2020**

Vagrant init

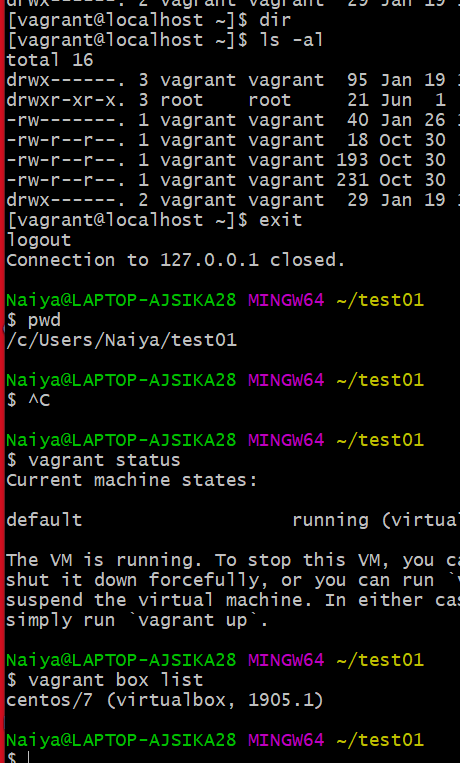
Vagrant status

Vagrant SSH

Sudo yum update -y

Vagrant init centos/7

Vagrant box list



C:\>cd Users/Naiya/test01

C:\Users\Naiya\test01>pwd

'pwd' is not recognized as an internal or external command,

operable program or batch file.

C:\Users\Naiya\test01>vagrant status

Current machine states:

default running (virtualbox)

The VM is running. To stop this VM, you can run `vagrant halt` to

shut it down forcefully, or you can run `vagrant suspend` to simply

suspend the virtual machine. In either case, to restart it again,

simply run `vagrant up`.

C:\Users\Naiya\test01>ls

'ls' is not recognized as an internal or external command,

operable program or batch file.

C:\Users\Naiya\test01>ls -a

'ls' is not recognized as an internal or external command,

operable program or batch file.

C:\Users\Naiya\test01>vagrant box list

centos/7 (virtualbox, 1905.1)

C:\Users\Naiya\test01>vagrant ssh

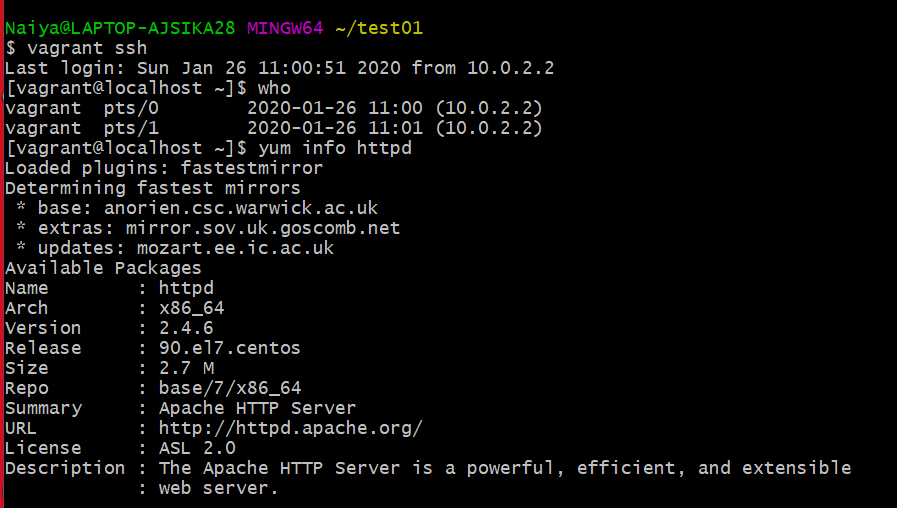
Last login: Sun Jan 26 10:35:31 2020 from 10.0.2.2

Last login: Sun Jan 26 10:35:31 2020 from 10.0.2.2

[vagrant@localhost ~]$ who

vagrant pts/0 2020-01-26 11:00 (10.0.2.2)

vagrant pts/1 2020-01-26 11:01 (10.0.2.2)



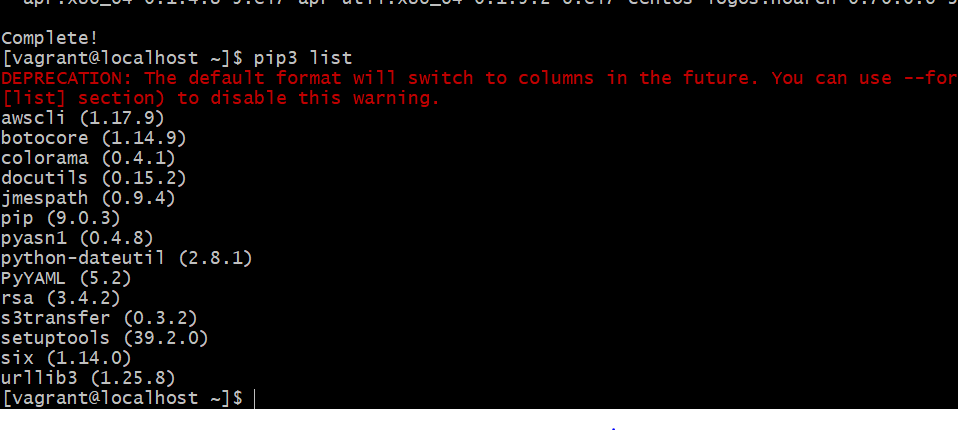
YUM : YELLOWDOG UPDATE MANAGER

Web servers: httpd (apache and nginx) , we can find information (yum info httpd)

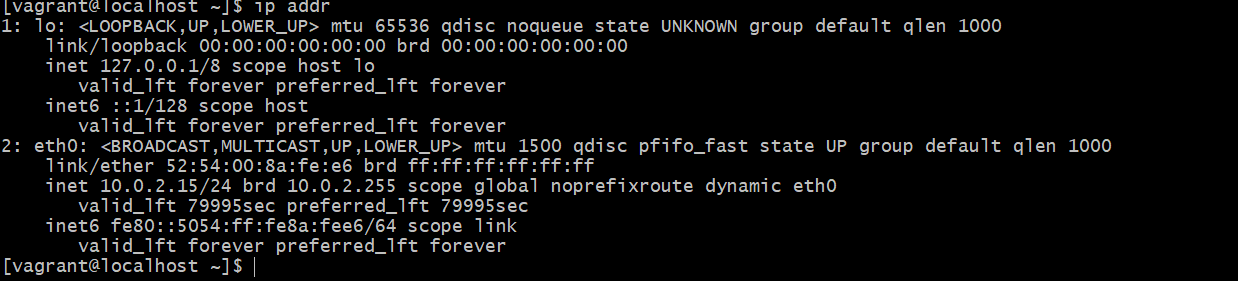
Repos: it is stored lib information: Source code information or binary files.

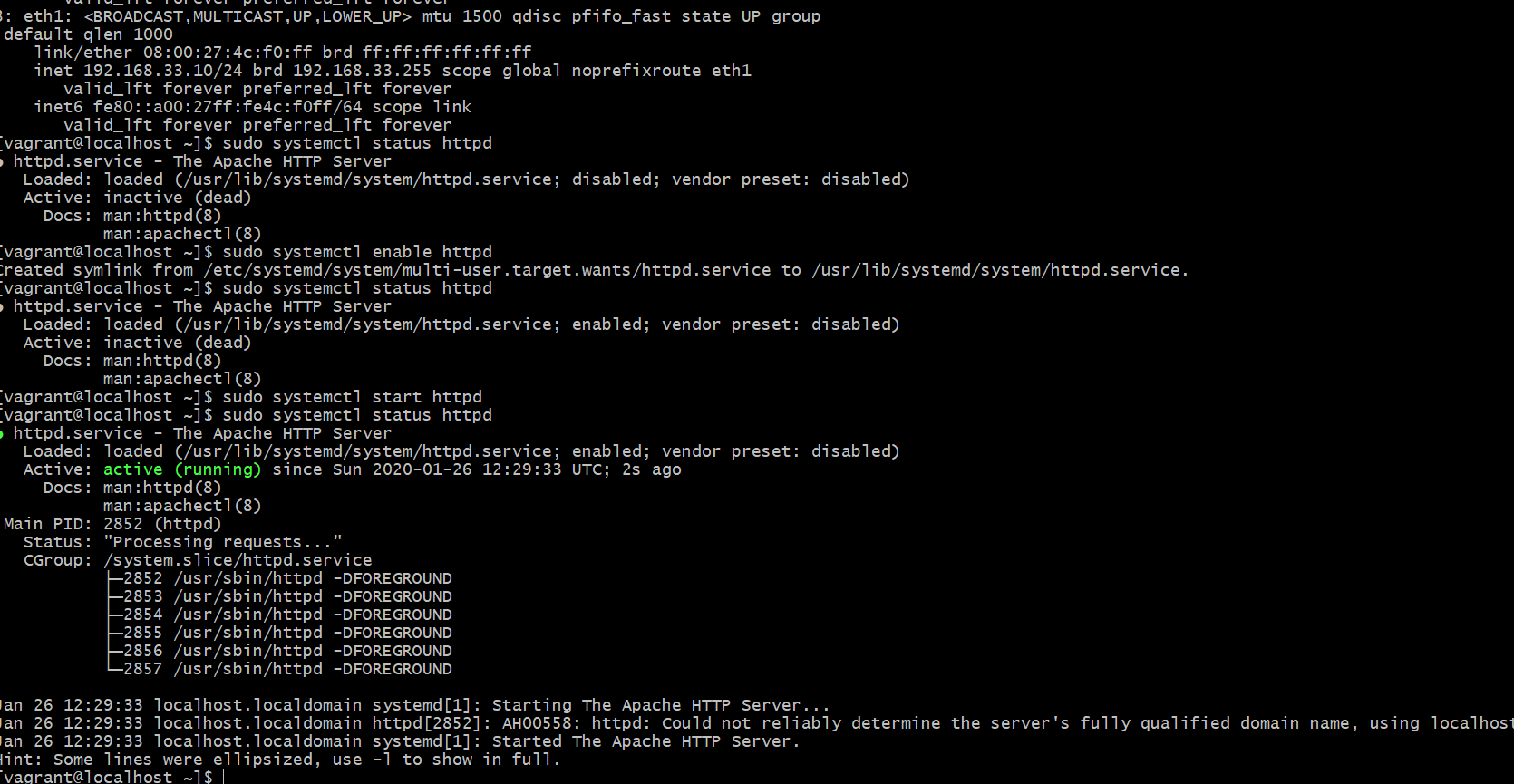
Sudo (super user do) to be a root user.

**Pip3 list** shows all the packages are here.



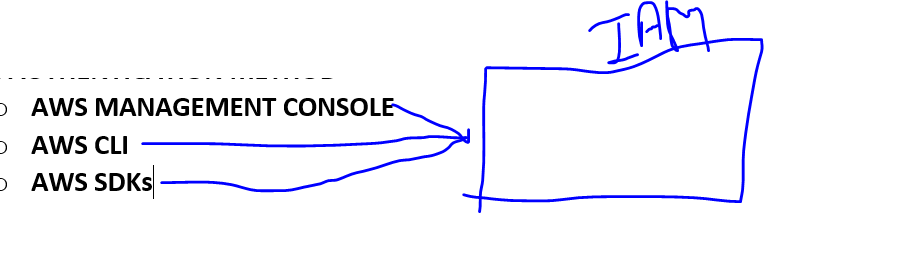
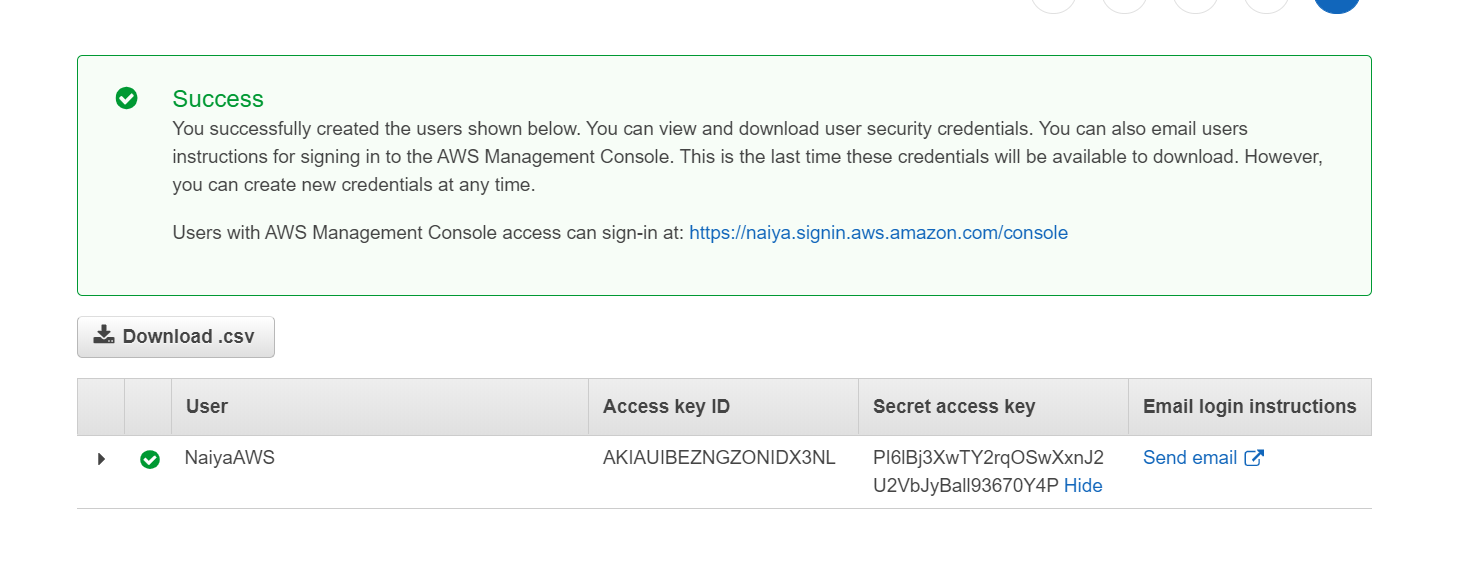
Ip address



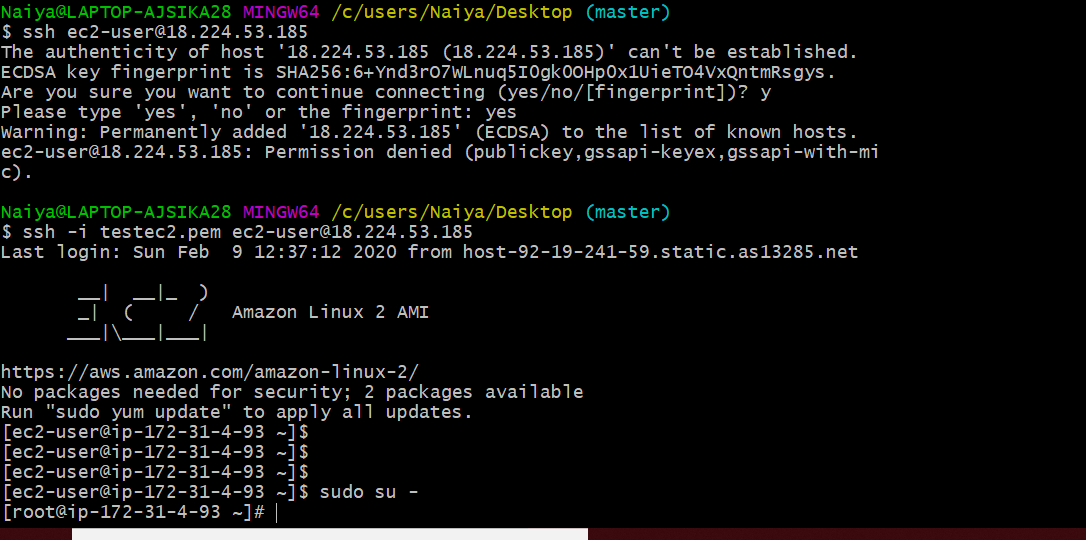
* DNS – (IP to NAME) :> NAT (Network Address Translator
* Vagrant halt and then we have to an another Ip for the local network
* We open the vagrant file in visual code and made 2 uncommented and then it was made a real network availability.
* Then start vagrant up
* Vagrant ssh
* Vagrant status
* Ip addr
* **Linux session (26 jan) : slides**
* What is shell
* BIOS + MBR + GRUB+KERNAL + INIT(compare to SYSTEMD) + RUN LEVEL(TARGETS)
* SystemD is latest version of INIT and (replaces with Sysvinit (redhat) & upstart)
* Kernal/OS/SHELL/USER
* 
* If you don’t enable then every time you have to start the server/ but if you have to keep active the server then you have to do enable.
* <http://127.0.0.1:8000/>
* <http://192.168.33.10/>

**FEB 2nd, 2020 Sunday Session**

**Cloud practitioners and consultant**

* **Cloud computing : http: AWS /cloud computing**
* **“CC” is flexible and low cost IT**
* **HA/ Disaster recovery/ Fault Tolerance**
* **Advantages: Go Global, Capacity, increase speed & Agility, maintain & Running datacentres with less cost.(Trade Capital expense for Variable expense)**
* **Scale up / Down**
* **EC2**
* **Types of CC: IAAS, PAAS, SAAS**
* **IAAS: Network+ storage + servers + virtual + O/S+ Middleware + Runtime+ data + APPS**
* **PAAS: data + APPS**
* **SAAS: like a google(as a user)**
* **What is AWS? (FORMULA 1, SAP) (Top cloud prov)**
* **AWS Global Infrastructure: Regions + AZs.**
* **Regions provide multiple, physically separated and isolated AZ which are connected with LOW LATENCY, HIGH THROUGHPUT and HIGHLY REDUNDANT NETWORK.**
* **AZ offers an easier way to design & operate app and databases, making them more HA, FT, and scalable as compare to single/multiple datacentres**
* **69 = AZ**
* **22 = Geo Regions**
* **1 = Local Region**
* **EDGE LOCATIONS (Single or multiple locations**
* **REGIONAL EDGE CACHES**
* **CDN : CONTENT DELIVERY NETWORK**
* **13 = AZ and 4 Regions = coming soon**
* **Shared Responsibility Model**
* **AWS GLOBAL Infra : Region + az = edge location**
* **Services : COMPUTE , STORAGE, DATABASE, Networking**
  + **Client side data Encryption & Data Integrity Authentication**
  + **Server Side Encryption (file system &/ data)**
  + **Network Traffic Protection(Encryption/Integrity/ Identity)**
* **(NOTE : physical server side people will not have a logical access and logical people will not have an access to physical )**
* **AWS is responsible OF the Cloud Security)**
* **OS, Network & Firewall Config**
* **Platform , APP, Identity & Access Management**
  + **Authenticate & Authorization**
  + **Authenticate -🡪 (Signed in)**
  + **Authorization 🡪 (Has Permissions)**
* **Customer data**
* **Console login**
* **Root**
* **MFA ( GOOGLE APP : authenticator)**
* **Password Policy**
* **Permission & Perveledges**
  + **USER ARE REPACED BY RBAC: ROLE BASED ACCESS CONTROL SYSTEM (user based access,**
* **IAM USE :** 
  + **USERS**
  + **GROUPS**
  + **ACCESS POLICIES**
  + **ROLES**
  + **USER CREDENTIALS**
  + **USER PASSWORD POLICIES**
  + **MFA**
  + **API KEY FOR PRAGRAMMATIC ACCESS**
* **IAM AUTHENTICATION METHOD**
  + **AWS MANAGEMENT CONSOLE**
  + **AWS CLI**
  + **AWS SDKs**
  + **iam users and groups**
  + **FOR SDKs we do need an API Keys ()**
* **[vagrant@localhost ~]$ pip3 list**
  + - * + **DEPRECATION: The default format will switch to columns in the future. You can use --format=(legacy|columns) (or define a format=(legacy|columns) in your pip.conf under the [list] section) to disable this warning.**
        + **awscli (1.17.9)**
        + **botocore (1.14.9)**
        + **colorama (0.4.1)**
        + **docutils (0.15.2)**
        + **jmespath (0.9.4)**
        + **pip (9.0.3)**
        + **pyasn1 (0.4.8)**
        + **python-dateutil (2.8.1)**
        + **PyYAML (5.2)**
        + **rsa (3.4.2)**
        + **s3transfer (0.3.2)**
        + **setuptools (39.2.0)**
        + **six (1.14.0)**
        + **urllib3 (1.25.8)**
* **AWS(IAM ) Authorization :**
  + **Full Access -> S3 Bucket**
  + **Read Only Access ->EC2 Instance**
* **Root user and Root Privilege user** 
  + **Create user with no permissions**
  + 

**Copy the console link into the google incognito window and check the privileges**

* **Go back to the root user screen and give some permissions**
* <https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#Instances:sort=instanceId> (IAM USER – NaiyaAWS)
* <https://console.aws.amazon.com/iam/home?region=us-east-1#/users/NaiyaAWS> (Root User)
* **HOME WORK DONE**
* 

**16 feb 2020 classroom session**

**Asymmetric keys : Encryption or decryption**

**What is SSH: Secure Shell or secure socket shell**

**Network Protocol**

**Secure shell is a cryptographic protocol for operating network services over the nw.**

**Open SSH is the free open source premier connectivity tool for the remote login within the SSH protocol. It encrypts the traffic between client and server.**

* **SSH🡪 Client**
* **SSHD🡪 Server (default port number is 22) D – Daemon.**
* **Key based Authentication (.pem keys)**
* **Advantages: Key based authentication : full control of the keys its unique (longer key) and tunnel is secure.**

**Vagrant SSH ; vagrant up perform and forward to port 22**

* Adapter 1: NAT
* Adapter 2: Hostonly
* Ssh add: 127.0.0.1:2222
* Ssh user name : vagrant
* Ss auth method : private key
* **Ls .vagrant/machines/default/virtualbox**
* **Vagrant ssh-config**
* Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)
* $ cd test01
* Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)
* $ ls -a
* ./ ../ .vagrant/ Vagrantfile
* Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)
* $ vagrant up
* Bringing machine 'default' up with 'virtualbox' provider...
* ==> default: Checking if box 'centos/7' version '1905.1' is up to date...
* ==> default: Machine already provisioned. Run `vagrant provision` or use the `--provision`
* ==> default: flag to force provisioning. Provisioners marked to run always will still run.
* Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)
* $ vagrant status
* Current machine states:
* default running (virtualbox)
* The VM is running. To stop this VM, you can run `vagrant halt` to
* shut it down forcefully, or you can run `vagrant suspend` to simply
* suspend the virtual machine. In either case, to restart it again,
* simply run `vagrant up`.
* Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)
* $ ls -a
* ./ ../ .vagrant/ Vagrantfile
* Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)
* $ vagrant ssh-config
* Host default
* HostName 127.0.0.1
* User vagrant
* Port 2222
* UserKnownHostsFile /dev/null
* StrictHostKeyChecking no
* PasswordAuthentication no
* IdentityFile C:/Users/Naiya/test01/.vagrant/machines/default/virtualbox/private\_key
* IdentitiesOnly yes
* LogLevel FATAL
* Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)
* $ vagrant
* Usage: vagrant [options] <command> [<args>]
* -v, --version Print the version and exit.
* -h, --help Print this help.
* Common commands:
* box manages boxes: installation, removal, etc.
* cloud manages everything related to Vagrant Cloud
* destroy stops and deletes all traces of the vagrant machine
* global-status outputs status Vagrant environments for this user
* halt stops the vagrant machine
* help shows the help for a subcommand
* init initializes a new Vagrant environment by creating a Vagrantfile
* login
* package packages a running vagrant environment into a box
* plugin manages plugins: install, uninstall, update, etc.
* port displays information about guest port mappings
* powershell connects to machine via powershell remoting
* provision provisions the vagrant machine
* push deploys code in this environment to a configured destination
* rdp connects to machine via RDP
* reload restarts vagrant machine, loads new Vagrantfile configuration
* resume resume a suspended vagrant machine
* snapshot manages snapshots: saving, restoring, etc.
* ssh connects to machine via SSH
* ssh-config outputs OpenSSH valid configuration to connect to the machine
* status outputs status of the vagrant machine
* suspend suspends the machine
* up starts and provisions the vagrant environment
* upload upload to machine via communicator
* validate validates the Vagrantfile
* version prints current and latest Vagrant version
* winrm executes commands on a machine via WinRM
* winrm-config outputs WinRM configuration to connect to the machine
* For help on any individual command run `vagrant COMMAND -h`
* Additional subcommands are available, but are either more advanced
* or not commonly used. To see all subcommands, run the command
* `vagrant list-commands`.
* Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)
* $ vagrant cloud
* Usage: vagrant cloud <subcommand> [<args>]
* The cloud command can be used for taking actions against
* Vagrant Cloud like searching or uploading a Vagrant Box
* Available subcommands:
* auth For various authorization operations on Vagrant Cloud
* box For managing a Vagrant box entry on Vagrant Cloud
* provider For managing a Vagrant box's provider options
* publish A complete solution for creating or updating a new box on Vagrant Cloud
* search Search Vagrant Cloud for available boxes
* version For managing a Vagrant box's versions
* For help on any individual subcommand run `vagrant cloud <subcommand> -h`
* Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)
* $ ip addr
* bash: ip: command not found
* Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)
* $ ip add
* bash: ip: command not found
* Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)
* $ vagrant ssh-config
* Host default
* HostName 127.0.0.1
* User vagrant
* Port 2222
* UserKnownHostsFile /dev/null
* StrictHostKeyChecking no
* PasswordAuthentication no
* IdentityFile C:/Users/Naiya/test01/.vagrant/machines/default/virtualbox/private\_key
* IdentitiesOnly yes
* LogLevel FATAL
* Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)
* $ vagrant ssh
* Last login: Sun Feb 16 06:30:37 2020 from 10.0.2.2
* [vagrant@localhost ~]$ ip addr
* 1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
* link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
* inet 127.0.0.1/8 scope host lo
* valid\_lft forever preferred\_lft forever
* inet6 ::1/128 scope host
* valid\_lft forever preferred\_lft forever
* 2: eth0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP group default qlen 1000
* link/ether 52:54:00:8a:fe:e6 brd ff:ff:ff:ff:ff:ff
* inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic eth0
* valid\_lft 83090sec preferred\_lft 83090sec
* inet6 fe80::5054:ff:fe8a:fee6/64 scope link
* valid\_lft forever preferred\_lft forever
* 3: eth1: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP group default qlen 1000
* link/ether 08:00:27:4c:f0:ff brd ff:ff:ff:ff:ff:ff
* inet 192.168.33.10/24 brd 192.168.33.255 scope global noprefixroute eth1
* valid\_lft forever preferred\_lft forever
* inet6 fe80::a00:27ff:fe4c:f0ff/64 scope link
* valid\_lft forever preferred\_lft forever

**ssh –I .vagrant/machines/default//virtualbox/private\_key vagrant@** 192.168.33.10/24

**###### Generate own user key : RUN**

**Ssh-keygen –b 4096 –t rsa (practical)**

**-b = bit**

**-t type of an algo**

**Rsa -> is an algoritham ( dsa)**

Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ pwd

/

Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ ssh-keygen -b 4096 -t rsa

Generating public/private rsa key pair.

Enter file in which to save the key (/c/Users/Naiya/.ssh/id\_rsa):

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /c/Users/Naiya/.ssh/id\_rsa.

Your public key has been saved in /c/Users/Naiya/.ssh/id\_rsa.pub.

The key fingerprint is:

SHA256:akfK+ap6PQhqSgtGxUZVx9yPwePtqg0WaBxWdrmiqQU Naiya@LAPTOP-AJSIKA28

The key's randomart image is:

+---[RSA 4096]----+

| .....o+o.. |

| o +o.\* |

| + o . B |

| o Eo o. + o |

| . .+S.. . |

|. . ..O . . |

|.+ . oO .o . |

|=.. oooo. o. |

|+..o...o.... |

+----[SHA256]-----+

**To check the user -> id or whoami**

***Ls –a .ssh -> to verify the***

**If above don’t works go to the ~ then you rin the command.**

Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ cd

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ pwd

/c/Users/Naiya

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ls -a .ssh

./ ../ id\_rsa id\_rsa.pub known\_hosts

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$

**To verify the type of file= File .ssh/id\_rsa**

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ file .ssh/id\_rsa

.ssh/id\_rsa: OpenSSH private key

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ^C

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ file .ssh/id\_rsa.pub

.ssh/id\_rsa.pub: OpenSSH RSA public key

**Example with password based authentication**

**We have a private key in our home directory, which is not available publically, so permission is denied on vagrant ssh ( ssh have nothing in it)**

**Now we have to copy**

**We exit from the virtual machine and then run**

**Pwd**

**Ls –a .ssh/**

[root@localhost ~]# exit

logout

[vagrant@localhost ~]$ pwd

/home/vagrant

[vagrant@localhost ~]$ ls -a .ssh/

. .. authorized\_keys

[vagrant@localhost ~]$

* To check the content of the file : Cat .ssh/authorized\_keys

Steps to remember:--

Vag init -> to build a machine

Up -> start a machine

Ssh -> to communicate

**Vi editor and copy the public file in it**

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ pwd

/c/Users/Naiya

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ id

uid=197609(Naiya) gid=197121 groups=197121

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ls -a .ssh /

.ssh:

./ ../ id\_rsa id\_rsa.pub known\_hosts

/:

./ ../ bin/ cmd/ dev/ etc/ git-bash.exe\* git-cmd.exe\* LICENSE.txt mingw64/ proc/ ReleaseNotes.html tmp/ unins001.dat unins001.exe\* unins001.msg usr/

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ls -a .ssh/ls –a

./ ../ id\_rsa id\_rsa.pub known\_hosts

vi

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

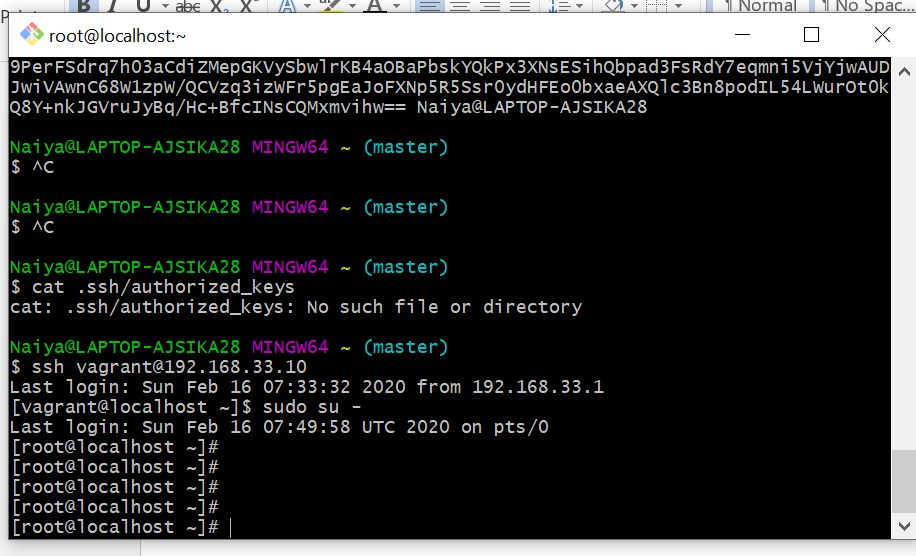
$

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ cat ~/.ssh/id\_rsa.pub

ssh-rsa  Naiya@LAPTOP-AJSIKA28

**Copy the public keys into vi editor and save. :wq!**



Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ ssh-keygen –b 4096 –t rsa

Too many arguments.

usage: ssh-keygen [-q] [-b bits] [-C comment] [-f output\_keyfile] [-m format]

[-N new\_passphrase] [-t dsa | ecdsa | ed25519 | rsa]

ssh-keygen -p [-f keyfile] [-m format] [-N new\_passphrase]

[-P old\_passphrase]

ssh-keygen -i [-f input\_keyfile] [-m key\_format]

ssh-keygen -e [-f input\_keyfile] [-m key\_format]

ssh-keygen -y [-f input\_keyfile]

ssh-keygen -c [-C comment] [-f keyfile] [-P passphrase]

ssh-keygen -l [-v] [-E fingerprint\_hash] [-f input\_keyfile]

ssh-keygen -B [-f input\_keyfile]

ssh-keygen -D pkcs11

ssh-keygen -F hostname [-lv] [-f known\_hosts\_file]

ssh-keygen -H [-f known\_hosts\_file]

ssh-keygen -R hostname [-f known\_hosts\_file]

ssh-keygen -r hostname [-g] [-f input\_keyfile]

ssh-keygen -G output\_file [-v] [-b bits] [-M memory] [-S start\_point]

ssh-keygen -f input\_file -T output\_file [-v] [-a rounds] [-J num\_lines]

[-j start\_line] [-K checkpt] [-W generator]

ssh-keygen -I certificate\_identity -s ca\_key [-hU] [-D pkcs11\_provider]

[-n principals] [-O option] [-V validity\_interval]

[-z serial\_number] file ...

ssh-keygen -L [-f input\_keyfile]

ssh-keygen -A [-f prefix\_path]

ssh-keygen -k -f krl\_file [-u] [-s ca\_public] [-z version\_number]

file ...

ssh-keygen -Q -f krl\_file file ...

ssh-keygen -Y check-novalidate -n namespace -s signature\_file

ssh-keygen -Y sign -f key\_file -n namespace file ...

ssh-keygen -Y verify -f allowed\_signers\_file -I signer\_identity

-n namespace -s signature\_file [-r revocation\_file]

Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ pwd

/

Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ ssh-keygen -b 4096 -t rsa

Generating public/private rsa key pair.

Enter file in which to save the key (/c/Users/Naiya/.ssh/id\_rsa):

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /c/Users/Naiya/.ssh/id\_rsa.

Your public key has been saved in /c/Users/Naiya/.ssh/id\_rsa.pub.

The key fingerprint is:

SHA256:akfK+ap6PQhqSgtGxUZVx9yPwePtqg0WaBxWdrmiqQU Naiya@LAPTOP-AJSIKA28

The key's randomart image is:

+---[RSA 4096]----+

| .....o+o.. |

| o +o.\* |

| + o . B |

| o Eo o. + o |

| . .+S.. . |

|. . ..O . . |

|.+ . oO .o . |

|=.. oooo. o. |

|+..o...o.... |

+----[SHA256]-----+

Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ ^C

Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ id

uid=197609(Naiya) gid=197121 groups=197121

Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ whoami

Naiya

Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ ls -a .ssh

ls: cannot access '.ssh': No such file or directory

Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ ls -a .ssh

ls: cannot access '.ssh': No such file or directory

Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ cd test01

bash: cd: test01: No such file or directory

Naiya@LAPTOP-AJSIKA28 MINGW64 /

$ cd

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ pwd

/c/Users/Naiya

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ls -a .ssh

./ ../ id\_rsa id\_rsa.pub known\_hosts

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ^C

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ file .ssh/id\_rsa

.ssh/id\_rsa: OpenSSH private key

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ^C

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ file .ssh/id\_rsa.pub

.ssh/id\_rsa.pub: OpenSSH RSA public key

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ^C

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ pwd

/c/Users/Naiya

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ id

uid=197609(Naiya) gid=197121 groups=197121

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ls -a .ssh /

.ssh:

./ ../ id\_rsa id\_rsa.pub known\_hosts

/:

./ ../ bin/ cmd/ dev/ etc/ git-bash.exe\* git-cmd.exe\* LICENSE.txt mingw

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ls -a .ssh/

./ ../ id\_rsa id\_rsa.pub known\_hosts

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ cat ~/.ssh/id\_rsa.pub

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQC9LoI9LaiwpfyL1JoLTDR4vbQFZH+dGT9Ejf3T9aXi+CZovEzjVcpkybM4tAbUveQ8PqCX0haHQhktUonJcpJvB+/hveUljzqEB99jJYh1RPGnGBoC4bHpJKOr9PerFSdrq7h03aCdiZMepGKVySbwlrKB4aOBaPbskYQkPx3XNsESihQbpad3FsRdY7eqmni5VjYjwAUDJwiVAwnC68W1zpW/QCVzq3izWFr5pgEaJoFXNp5R5Ssr0ydHFEo0bxaeAXQlc3Bn8podIL54LWurOt0kQ8Y+nkJGVruJyBq/Hc+BfcINsCQMxmvihw== Naiya@LAPTOP-AJSIKA28

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ^C

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ^C

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ cat .ssh/authorized\_keys

cat: .ssh/authorized\_keys: No such file or directory

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ ssh vagrant@192.168.33.10

Last login: Sun Feb 16 07:33:32 2020 from 192.168.33.1

[vagrant@localhost ~]$ sudo su -

Last login: Sun Feb 16 07:49:58 UTC 2020 on pts/0

[root@localhost ~]#

[root@localhost ~]#

[root@localhost ~]#

[root@localhost ~]#

Naiya@LAPTOP-AJSIKA28 MINGW64 ~ (master)

$ cd test01

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ ls -a

./ ../ .vagrant/ Vagrantfile

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ vagrant up

Bringing machine 'default' up with 'virtualbox' provider...

==> default: Checking if box 'centos/7' version '1905.1' is up to date...

==> default: Machine already provisioned. Run `vagrant provision` or use the `--provision`

==> default: flag to force provisioning. Provisioners marked to run always will still run.

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ vagrant status

Current machine states:

default running (virtualbox)

The VM is running. To stop this VM, you can run `vagrant halt` to

shut it down forcefully, or you can run `vagrant suspend` to simply

suspend the virtual machine. In either case, to restart it again,

simply run `vagrant up`.

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ ls -a

./ ../ .vagrant/ Vagrantfile

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ vagrant ssh-config

Host default

HostName 127.0.0.1

User vagrant

Port 2222

UserKnownHostsFile /dev/null

StrictHostKeyChecking no

PasswordAuthentication no

IdentityFile C:/Users/Naiya/test01/.vagrant/machines/default/virtualbox/private\_key

IdentitiesOnly yes

LogLevel FATAL

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ vagrant

Usage: vagrant [options] <command> [<args>]

-v, --version Print the version and exit.

-h, --help Print this help.

Common commands:

box manages boxes: installation, removal, etc.

cloud manages everything related to Vagrant Cloud

destroy stops and deletes all traces of the vagrant machine

global-status outputs status Vagrant environments for this user

halt stops the vagrant machine

help shows the help for a subcommand

init initializes a new Vagrant environment by creating a Vagrantfile

login

package packages a running vagrant environment into a box

plugin manages plugins: install, uninstall, update, etc.

port displays information about guest port mappings

powershell connects to machine via powershell remoting

provision provisions the vagrant machine

push deploys code in this environment to a configured destination

rdp connects to machine via RDP

reload restarts vagrant machine, loads new Vagrantfile configuration

resume resume a suspended vagrant machine

snapshot manages snapshots: saving, restoring, etc.

ssh connects to machine via SSH

ssh-config outputs OpenSSH valid configuration to connect to the machine

status outputs status of the vagrant machine

suspend suspends the machine

up starts and provisions the vagrant environment

upload upload to machine via communicator

validate validates the Vagrantfile

version prints current and latest Vagrant version

winrm executes commands on a machine via WinRM

winrm-config outputs WinRM configuration to connect to the machine

For help on any individual command run `vagrant COMMAND -h`

Additional subcommands are available, but are either more advanced

or not commonly used. To see all subcommands, run the command

`vagrant list-commands`.

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ vagrant cloud

Usage: vagrant cloud <subcommand> [<args>]

The cloud command can be used for taking actions against

Vagrant Cloud like searching or uploading a Vagrant Box

Available subcommands:

auth For various authorization operations on Vagrant Cloud

box For managing a Vagrant box entry on Vagrant Cloud

provider For managing a Vagrant box's provider options

publish A complete solution for creating or updating a new box on Vagrant Cloud

search Search Vagrant Cloud for available boxes

version For managing a Vagrant box's versions

For help on any individual subcommand run `vagrant cloud <subcommand> -h`

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ ip addr

bash: ip: command not found

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ ip add

bash: ip: command not found

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ vagrant ssh-config

Host default

HostName 127.0.0.1

User vagrant

Port 2222

UserKnownHostsFile /dev/null

StrictHostKeyChecking no

PasswordAuthentication no

IdentityFile C:/Users/Naiya/test01/.vagrant/machines/default/virtualbox/private\_key

IdentitiesOnly yes

LogLevel FATAL

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ vagrant ssh

Last login: Sun Feb 16 06:30:37 2020 from 10.0.2.2

[vagrant@localhost ~]$ ip addr

1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

inet 127.0.0.1/8 scope host lo

valid\_lft forever preferred\_lft forever

inet6 ::1/128 scope host

valid\_lft forever preferred\_lft forever

2: eth0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP group default qlen 1000

link/ether 52:54:00:8a:fe:e6 brd ff:ff:ff:ff:ff:ff

inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic eth0

valid\_lft 83090sec preferred\_lft 83090sec

inet6 fe80::5054:ff:fe8a:fee6/64 scope link

valid\_lft forever preferred\_lft forever

3: eth1: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP group default qlen 1000

link/ether 08:00:27:4c:f0:ff brd ff:ff:ff:ff:ff:ff

inet 192.168.33.10/24 brd 192.168.33.255 scope global noprefixroute eth1

valid\_lft forever preferred\_lft forever

inet6 fe80::a00:27ff:fe4c:f0ff/64 scope link

valid\_lft forever preferred\_lft forever

[vagrant@localhost ~]$ ^C

[vagrant@localhost ~]$ exit

logout

Connection to 127.0.0.1 closed.

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ ^Mssh –I .vagrant/machines/default//virtualbox/private\_key vagrant@ 192.168.33.10/24

ssh: Could not resolve hostname \342\200\223i: Name or service not known

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ ssh –I .vagrant/machines/default//virtualbox/private\_key vagrant@ 192.168.33.10/24

ssh: Could not resolve hostname \342\200\223i: Name or service not known

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ ssh –i .vagrant/machines/default//virtualbox/private\_key vagrant@192.168.33.10/24

ssh: Could not resolve hostname \342\200\223i: Name or service not known

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ ssh –i .vagrant/machines/default//virtualbox/private\_key vagrant@192.168.33.10

ssh: Could not resolve hostname \342\200\223i: Name or service not known

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ vagrant ssh

Last login: Sun Feb 16 07:21:28 2020 from 10.0.2.2

[vagrant@localhost ~]$ ip a

1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

inet 127.0.0.1/8 scope host lo

valid\_lft forever preferred\_lft forever

inet6 ::1/128 scope host

valid\_lft forever preferred\_lft forever

2: eth0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP group default qlen 1000

link/ether 52:54:00:8a:fe:e6 brd ff:ff:ff:ff:ff:ff

inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic eth0

valid\_lft 82608sec preferred\_lft 82608sec

inet6 fe80::5054:ff:fe8a:fee6/64 scope link

valid\_lft forever preferred\_lft forever

3: eth1: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP group default qlen 1000

link/ether 08:00:27:4c:f0:ff brd ff:ff:ff:ff:ff:ff

inet 192.168.33.10/24 brd 192.168.33.255 scope global noprefixroute eth1

valid\_lft forever preferred\_lft forever

inet6 fe80::a00:27ff:fe4c:f0ff/64 scope link

valid\_lft forever preferred\_lft forever

[vagrant@localhost ~]$ netstat -a

-bash: netstat: command not found

[vagrant@localhost ~]$ nestat -ntap

-bash: nestat: command not found

[vagrant@localhost ~]$ netstat

-bash: netstat: command not found

[vagrant@localhost ~]$ netstat

-bash: netstat: command not found

[vagrant@localhost ~]$ ip a

1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

inet 127.0.0.1/8 scope host lo

valid\_lft forever preferred\_lft forever

inet6 ::1/128 scope host

valid\_lft forever preferred\_lft forever

2: eth0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP group default qlen 1000

link/ether 52:54:00:8a:fe:e6 brd ff:ff:ff:ff:ff:ff

inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic eth0

valid\_lft 82572sec preferred\_lft 82572sec

inet6 fe80::5054:ff:fe8a:fee6/64 scope link

valid\_lft forever preferred\_lft forever

3: eth1: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP group default qlen 1000

link/ether 08:00:27:4c:f0:ff brd ff:ff:ff:ff:ff:ff

inet 192.168.33.10/24 brd 192.168.33.255 scope global noprefixroute eth1

valid\_lft forever preferred\_lft forever

inet6 fe80::a00:27ff:fe4c:f0ff/64 scope link

valid\_lft forever preferred\_lft forever

[vagrant@localhost ~]$ exit

logout

Connection to 127.0.0.1 closed.

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ ssh –i .vagrant/machines/default//virtualbox/private\_key vagrant@192.168.33.10

ssh: Could not resolve hostname \342\200\223i: Name or service not known

Naiya@LAPTOP-AJSIKA28 MINGW64 ~/test01 (master)

$ ssh -i .vagrant/machines/default/virtualbox/private\_key vagrant@192.168.33.10

The authenticity of host '192.168.33.10 (192.168.33.10)' can't be established.

ECDSA key fingerprint is SHA256:ay6NsBrDBqMfM4v6epmQyx0mvvEcEASce35BIB9noPY.

Are you sure you want to continue connecting (yes/no/[fingerprint])? yes

Warning: Permanently added '192.168.33.10' (ECDSA) to the list of known hosts.

Last login: Sun Feb 16 07:29:36 2020 from 10.0.2.2

[vagrant@localhost ~]$ sudo su -

[root@localhost ~]# systemctl status sshd

● sshd.service - OpenSSH server daemon

Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)

Active: active (running) since Sun 2020-02-16 06:26:12 UTC; 1h 25min ago

Docs: man:sshd(8)

man:sshd\_config(5)

Main PID: 697 (sshd)

CGroup: /system.slice/sshd.service

└─697 /usr/sbin/sshd -D -u0

Feb 16 06:26:11 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...

Feb 16 06:26:12 localhost.localdomain sshd[697]: Server listening on 0.0.0.0 port 22.

Feb 16 06:26:12 localhost.localdomain sshd[697]: Server listening on :: port 22.

Feb 16 06:26:12 localhost.localdomain systemd[1]: Started OpenSSH server daemon.

Feb 16 06:26:12 localhost.localdomain sshd[713]: Accepted publickey for vagrant from 10.0.2.2 port 59332 ssh2: RSA SHA256:7Uq50BhF5sDcdfAYdGvEfBnNs6WXfk2PsKwD+Y/hjvU

Feb 16 06:26:31 localhost.localdomain sshd[2806]: Accepted publickey for vagrant from 10.0.2.2 port 59337 ssh2: RSA SHA256:7Uq50BhF5sDcdfAYdGvEfBnNs6WXfk2PsKwD+Y/hjvU

Feb 16 06:30:37 localhost.localdomain sshd[2932]: Accepted publickey for vagrant from 10.0.2.2 port 59381 ssh2: RSA SHA256:7Uq50BhF5sDcdfAYdGvEfBnNs6WXfk2PsKwD+Y/hjvU

Feb 16 07:21:27 localhost.localdomain sshd[2983]: Accepted publickey for vagrant from 10.0.2.2 port 59922 ssh2: RSA SHA256:7Uq50BhF5sDcdfAYdGvEfBnNs6WXfk2PsKwD+Y/hjvU

Feb 16 07:29:35 localhost.localdomain sshd[3015]: Accepted publickey for vagrant from 10.0.2.2 port 59959 ssh2: RSA SHA256:7Uq50BhF5sDcdfAYdGvEfBnNs6WXfk2PsKwD+Y/hjvU

Feb 16 07:33:31 localhost.localdomain sshd[3083]: Accepted publickey for vagrant from 192.168.33.1 port 59969 ssh2: RSA SHA256:7Uq50BhF5sDcdfAYdGvEfBnNs6WXfk2PsKwD+Y/hjvU

[root@localhost ~]# ssh vagrant@192.168.33.10

The authenticity of host '192.168.33.10 (192.168.33.10)' can't be established.

ECDSA key fingerprint is SHA256:ay6NsBrDBqMfM4v6epmQyx0mvvEcEASce35BIB9noPY.

ECDSA key fingerprint is MD5:97:22:2c:fa:5a:8d:63:0e:b8:50:bb:17:ca:5b:52:05.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added '192.168.33.10' (ECDSA) to the list of known hosts.

Permission denied (publickey,gssapi-keyex,gssapi-with-mic).

[root@localhost ~]# exit

logout

[vagrant@localhost ~]$ pwd

/home/vagrant

[vagrant@localhost ~]$ ls -a .ssh/

. .. authorized\_keys

[vagrant@localhost ~]$ ^C

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$

[vagrant@localhost ~]$ cat .ssh/authorized\_keys

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDB+vFWLWtZgHRjGpADZZEpkbWFX56E3qP6P5AQUZHDkn4+2RCLo0G3SJckTMKple3//kRFfRryNAzee10mmPNxGIrAZ5JEmsjB+sAe1NmQ1x2FU3urYOSb4RVyJK4pZ+wTOjeYkOK45iBPSuITIHcFsnHm5+4/5zD6lBklOexx6sTIXKtzOP+jX+dNIjpVy33iCwGg0fLZOJ5hc7NKLqPi8HW/tQMcQVyxRpumRCmUZv/jQ+FNpvTLBHuxEUbHjDdQB/NdppnGttHdlp+Ae6llGRaLcFINqzOxhnDMDZsVYz4B8eZ1iT3kTPt1fryL0pEkoyA05RFIoVPfOjW7XVVR vagrant

[vagrant@localhost ~]$ vi .ssh/authorized\_keys

[vagrant@localhost ~]$ vi .ssh/authorized\_keys

[vagrant@localhost ~]$ cat .ssh/authorized\_keys

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDB+vFWLWtZgHRjGpADZZEpkbWFX56E3qP6P5AQUZHDkn4+2RCLo0G3SJckTMKple3//kRFfRryNAzee10mmPNxGIrAZ5JEmsjB+sAe1NmQ1x2FU3urYOSb4RVyJK4pZ+wTOjeYkOK45iBPSuITIHcFsnHm5+4/5zD6lBklOexx6sTIXKtzOP+jX+dNIjpVy33iCwGg0fLZOJ5hc7NKLqPi8HW/tQMcQVyxRpumRCmUZv/jQ+FNpvTLBHuxEUbHjDdQB/NdppnGttHdlp+Ae6llGRaLcFINqzOxhnDMDZsVYz4B8eZ1iT3kTPt1fryL0pEkoyA05RFIoVPfOjW7XVVR vagrant

[vagrant@localhost ~]$ cat .ssh/authorized\_keys

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDB+vFWLWtZgHRjGpADZZEpkbWFX56E3qP6P5AQUZHDkn4+2RCLo0G3SJckTMKple3//kRFfRryNAzee10mmPNxGIrAZ5JEmsjB+sAe1NmQ1x2FU3urYOSb4RVyJK4pZ+wTOjeYkOK45iBPSuITIHcFsnHm5+4/5zD6lBklOexx6sTIXKtzOP+jX+dNIjpVy33iCwGg0fLZOJ5hc7NKLqPi8HW/tQMcQVyxRpumRCmUZv/jQ+FNpvTLBHuxEUbHjDdQB/NdppnGttHdlp+Ae6llGRaLcFINqzOxhnDMDZsVYz4B8eZ1iT3kTPt1fryL0pEkoyA05RFIoVPfOjW7XVVR vagrant

[vagrant@localhost ~]$ vim .ssh/authorized\_keys

-bash: vim: command not found

[vagrant@localhost ~]$ vi .ssh/authorized\_keys

[vagrant@localhost ~]$ cat .ssh/authorized\_keys

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDB+vFWLWtZgHRjGpADZZEpkbWFX56E3qP6P5AQUZHDkn4+2RCLo0G3SJckTMKple3//kRFfRryNAzee10mmPNxGIrAZ5JEmsjB+sAe1NmQ1x2FU3urYOSb4RVyJK4pZ+wTOjeYkOK45iBPSuITIHcFsnHm5+4/5zD6lBklOexx6sTIXKtzOP+jX+dNIjpVy33iCwGg0fLZOJ5hc7NKLqPi8HW/tQMcQVyxRpumRCmUZv/jQ+FNpvTLBHuxEUbHjDdQB/NdppnGttHdlp+Ae6llGRaLcFINqzOxhnDMDZsVYz4B8eZ1iT3kTPt1fryL0pEkoyA05RFIoVPfOjW7XVVR vagrant

ssh-rsa  Naiya@LAPTOP-AJSIKA28

[vagrant@localhost ~]$

**23 FEB 2020**

**Storage : S3 EBS**

**Block storage (EBS and EFS) and Object Storage (S3)**

**S3: Simple Storage Solution (WORM):Write Once Read Many times**

**Scalable**

**Not good for OS or DATABASE**

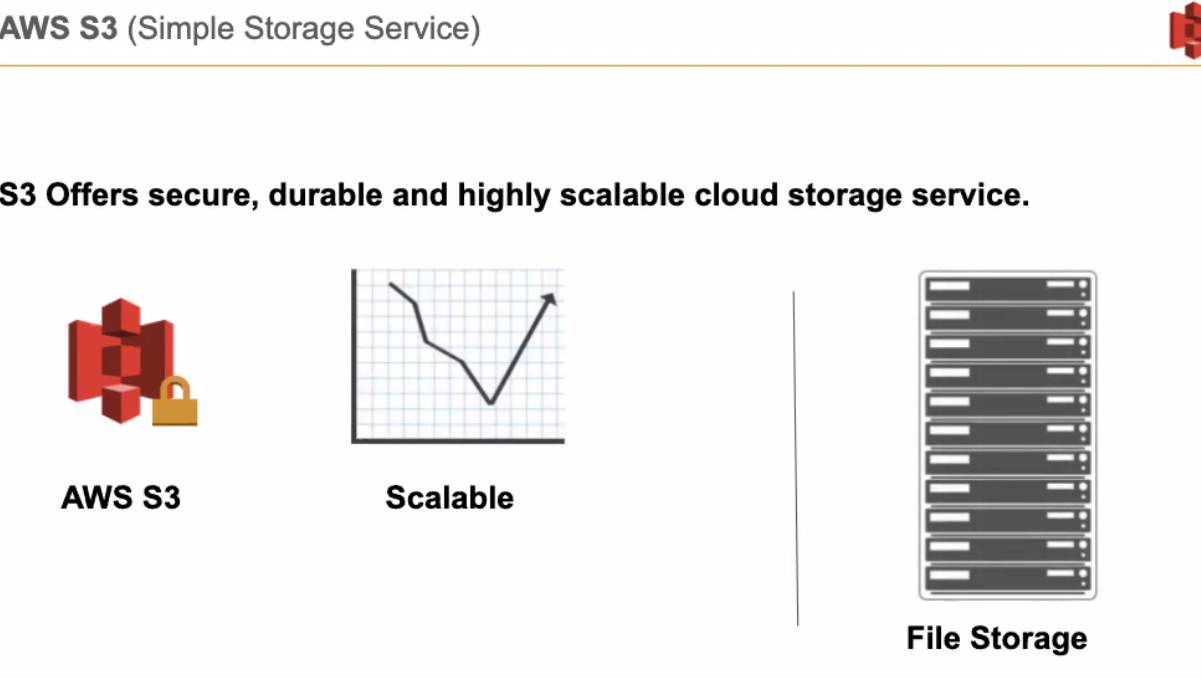
**EBS: Server disks**

**Persistent: more valuable**

**Ebs to one ec2 instance.**

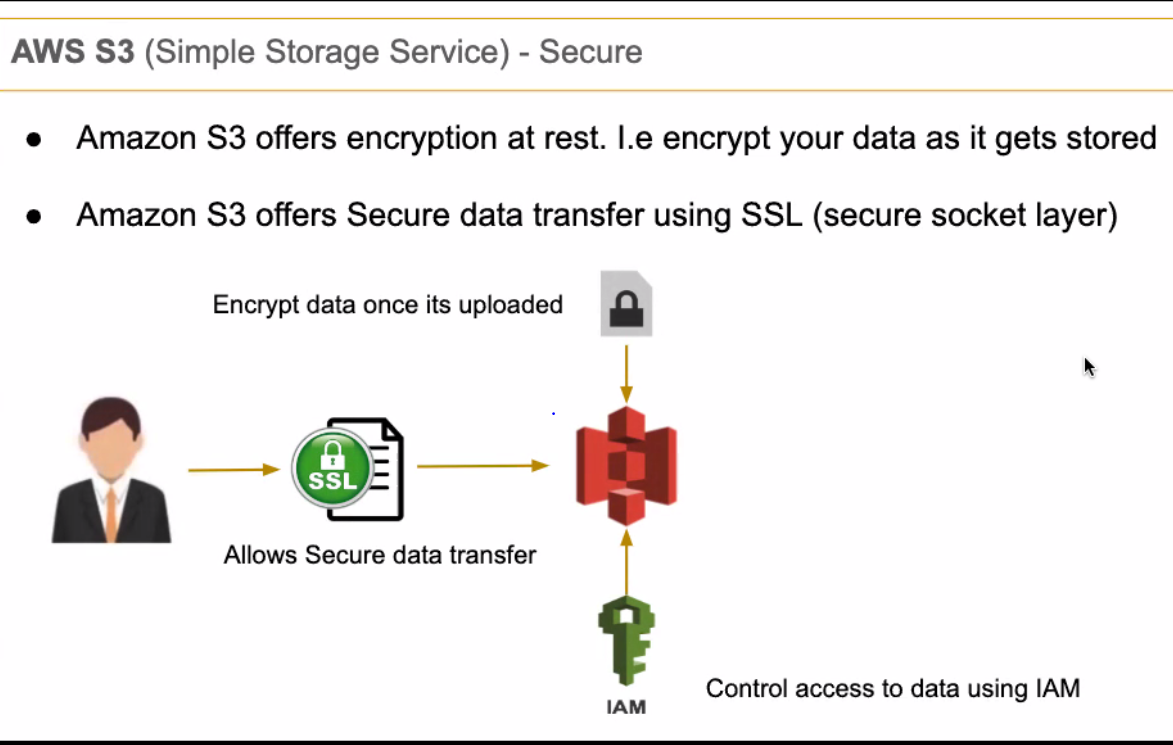
**AWS (Elastic File System)->EFS**

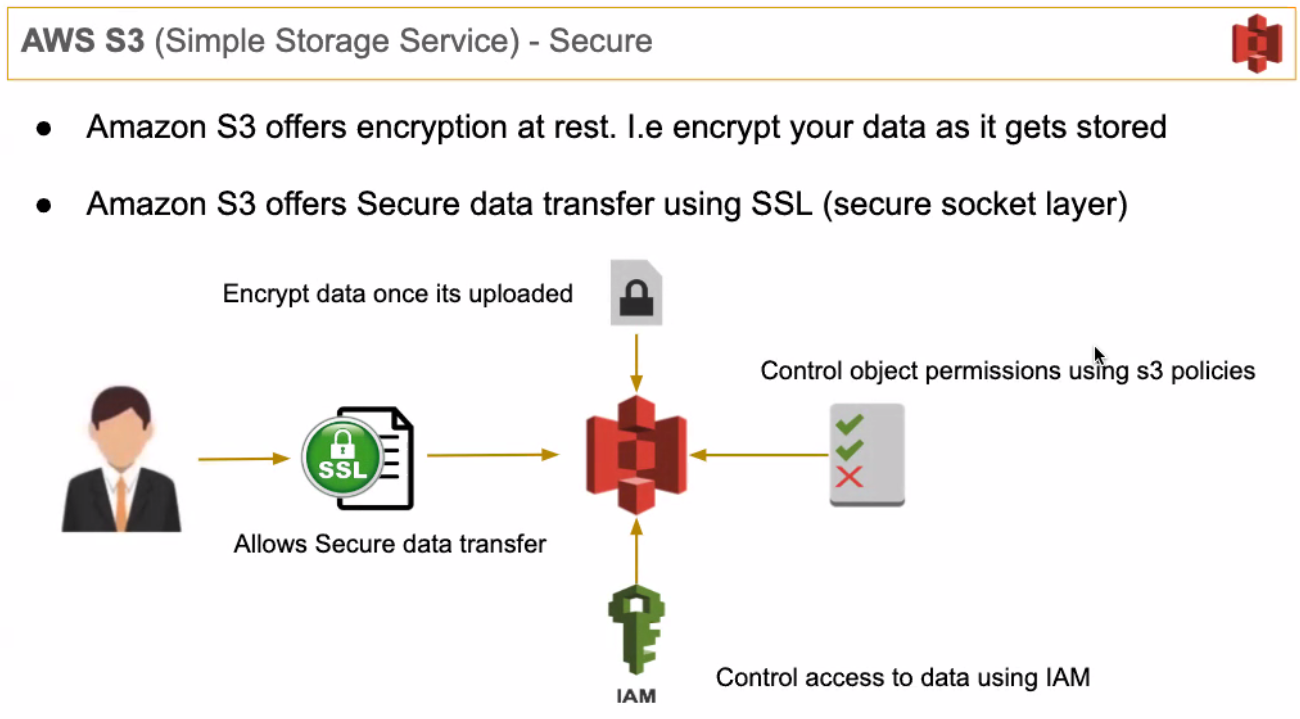
**NFS : Network File System (NFSv4) Protocol**



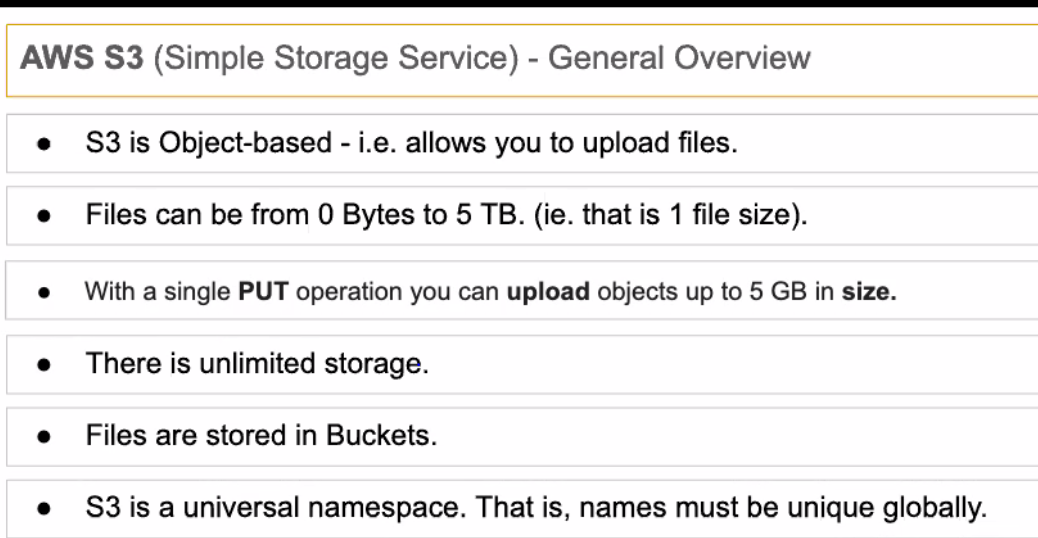
**Resilient:**

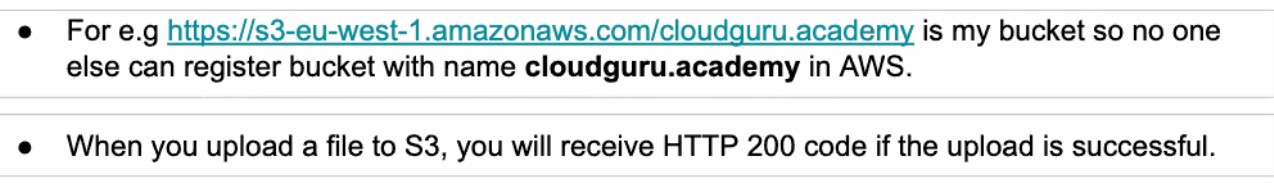
**Secure: S3 encrypted +ssl**

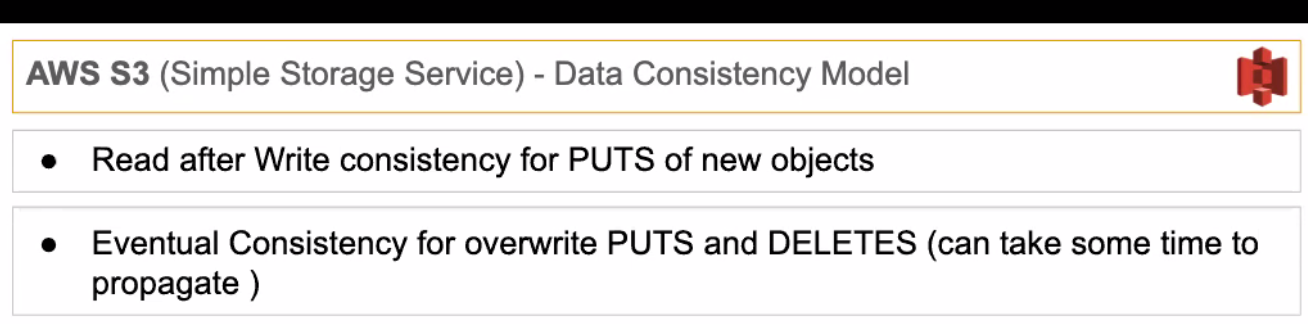


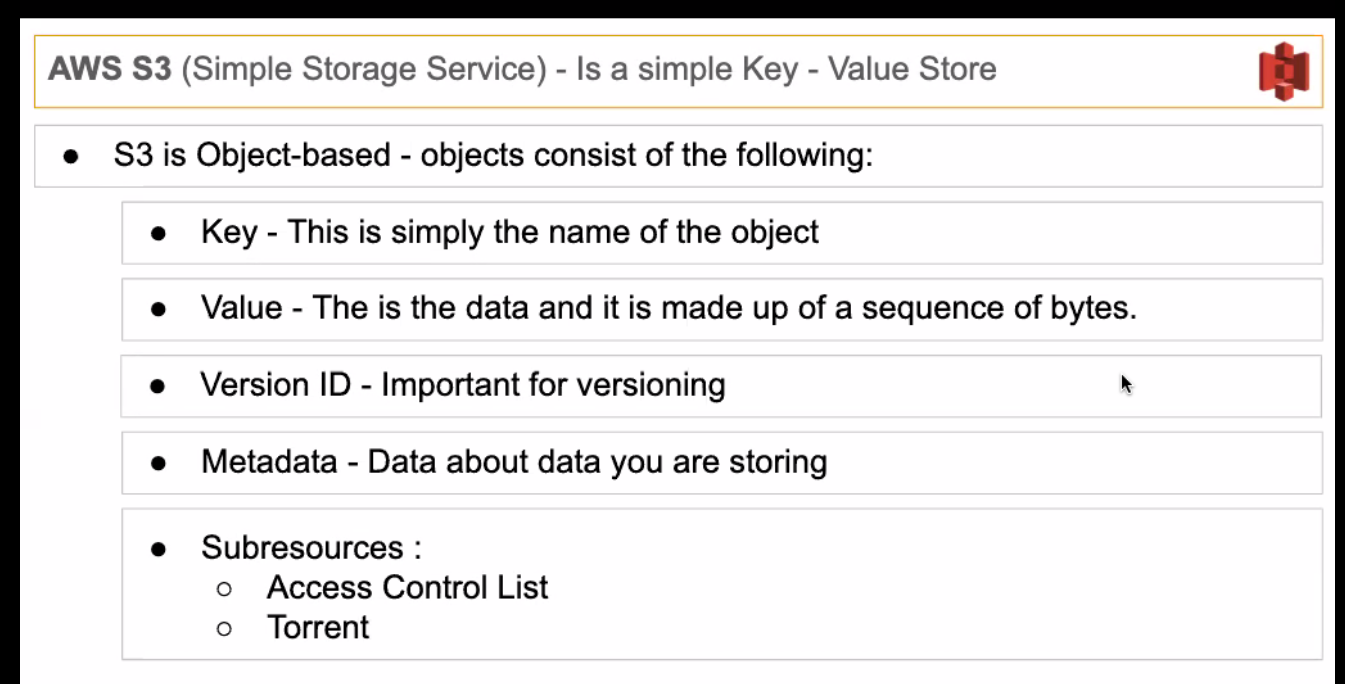


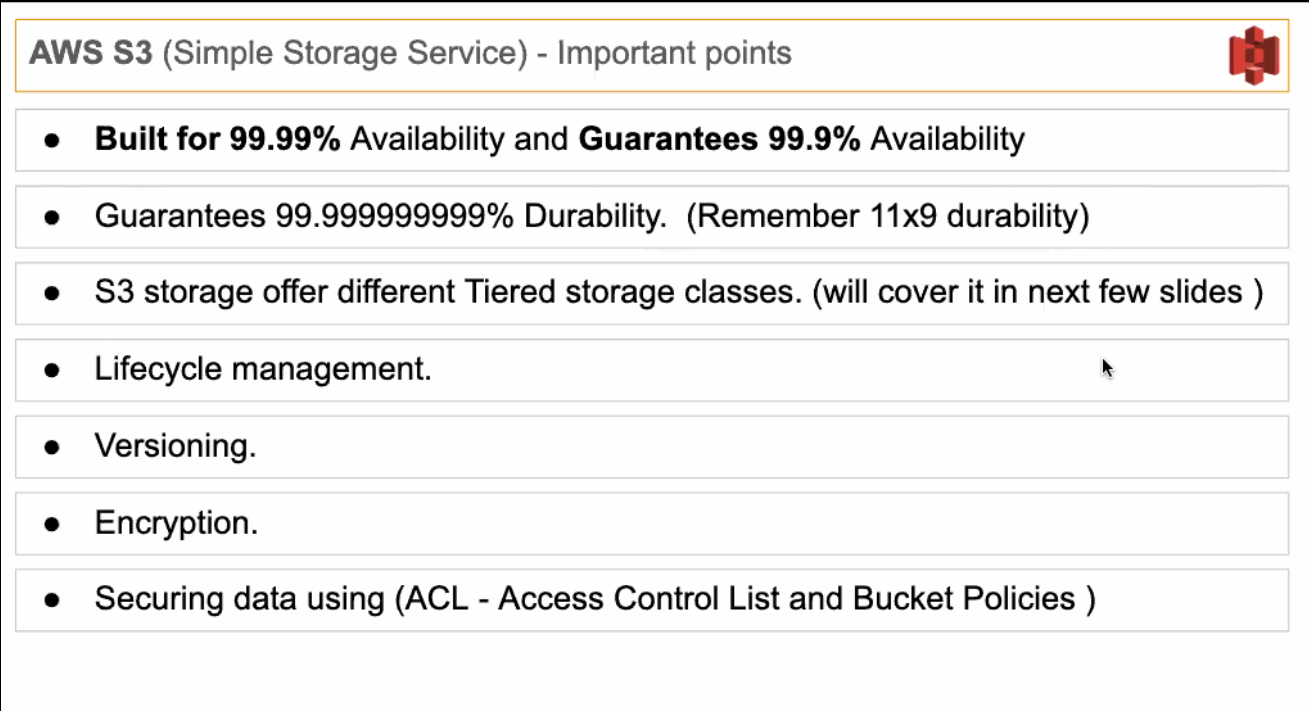
**Bucket ( data stored in it)**

****









**15 march 2020**

**Route 53**

**22 march**

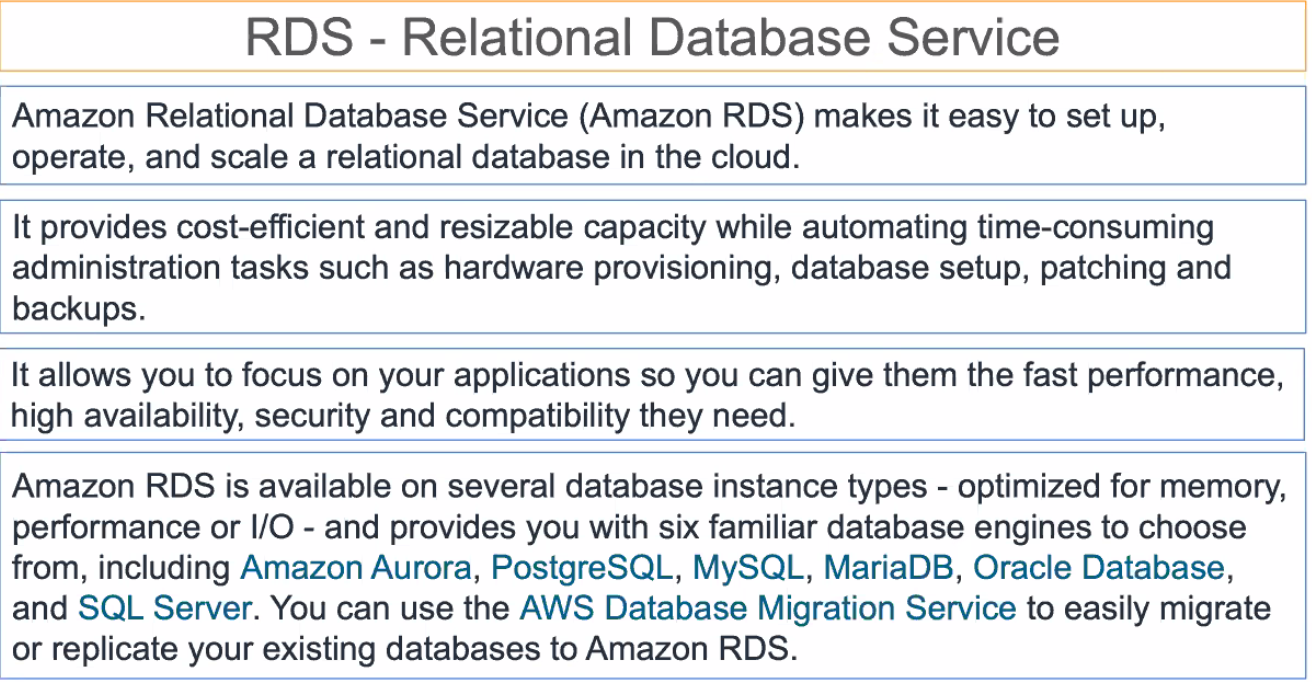
**29 march**

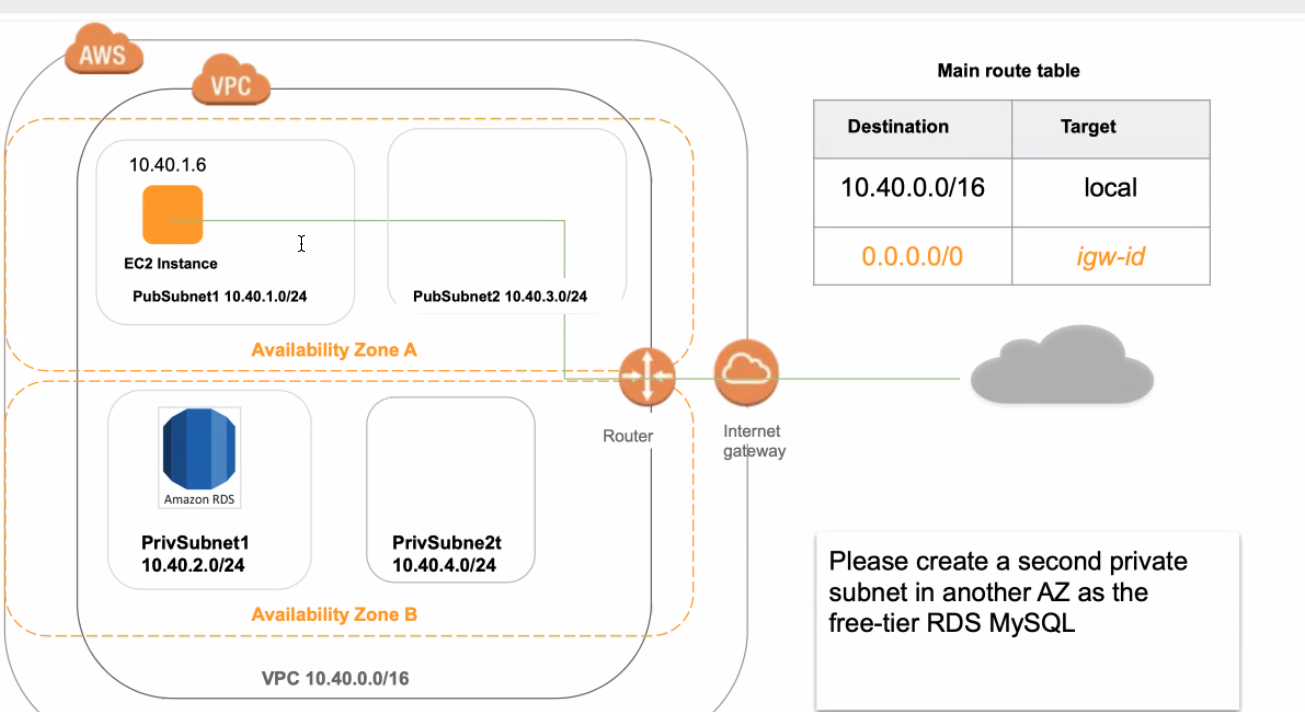
**5 april**

**12 april**

**19 april cancelled session**

**26 April: AWS- RDS-** Ashish





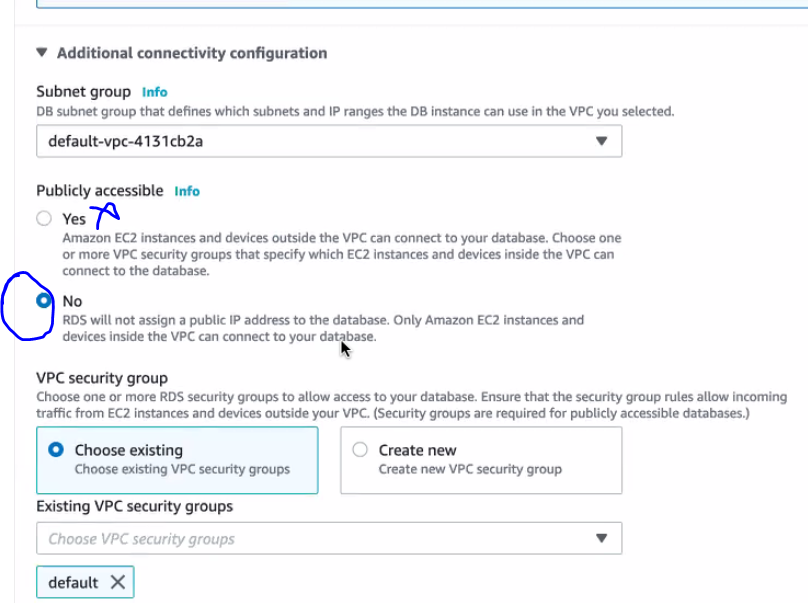
**We require 2 minimum private subnet (in RDS) because of the design and if one subnet fail, it is used by second subnet.**

**###Create a new Subnet; always check the route table and do edit Associations**

**While creating subnet group, add subnets, add only private subnets. Now to launch new RDS instance:**

**Amazon aurora is perfect but expensive.**

**###RDS is more expensive than EC2 instance**



**Important note for the subnet to choose it NO PUBLU+IC ACCESS.**



**###AUTOMATIC BACK UP AND RETENSION TIME.**

**MONITORING IN RDS**

**RDS TAKES 10-15 MINS**

**THEN CREATE EC2 AND WE WILL COMMUNICATE WITH RDS THROUGH EC2.**

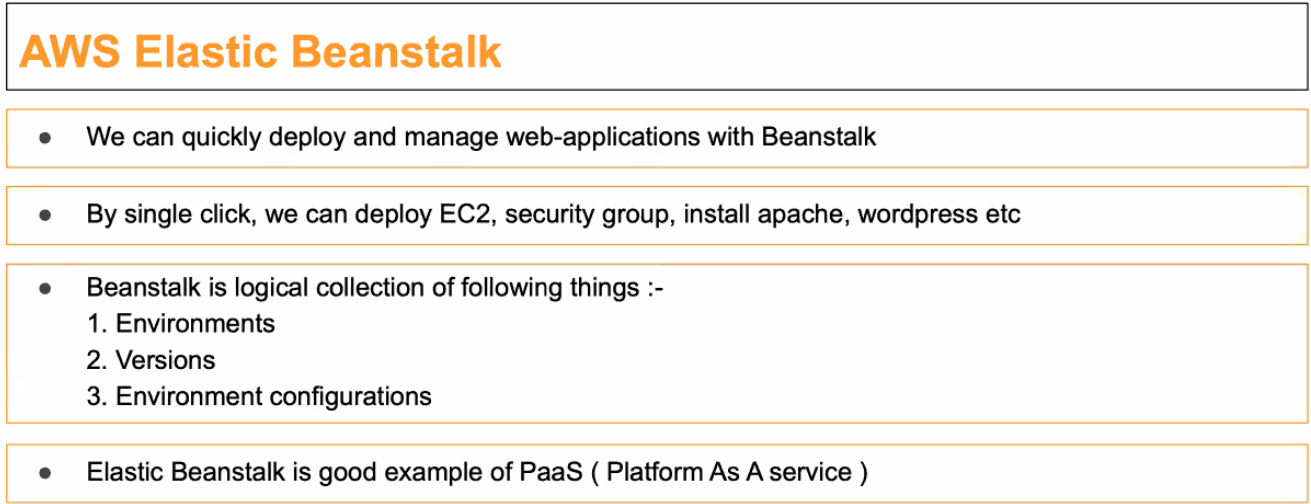
**SSH –I .PEM EC2-USER@IP**

**SUDO YUM INSTALL MARIDB –Y**

**MYSQL –U (USER) ADMIN –H(HOST)COPY THE ENDPOINT -P**

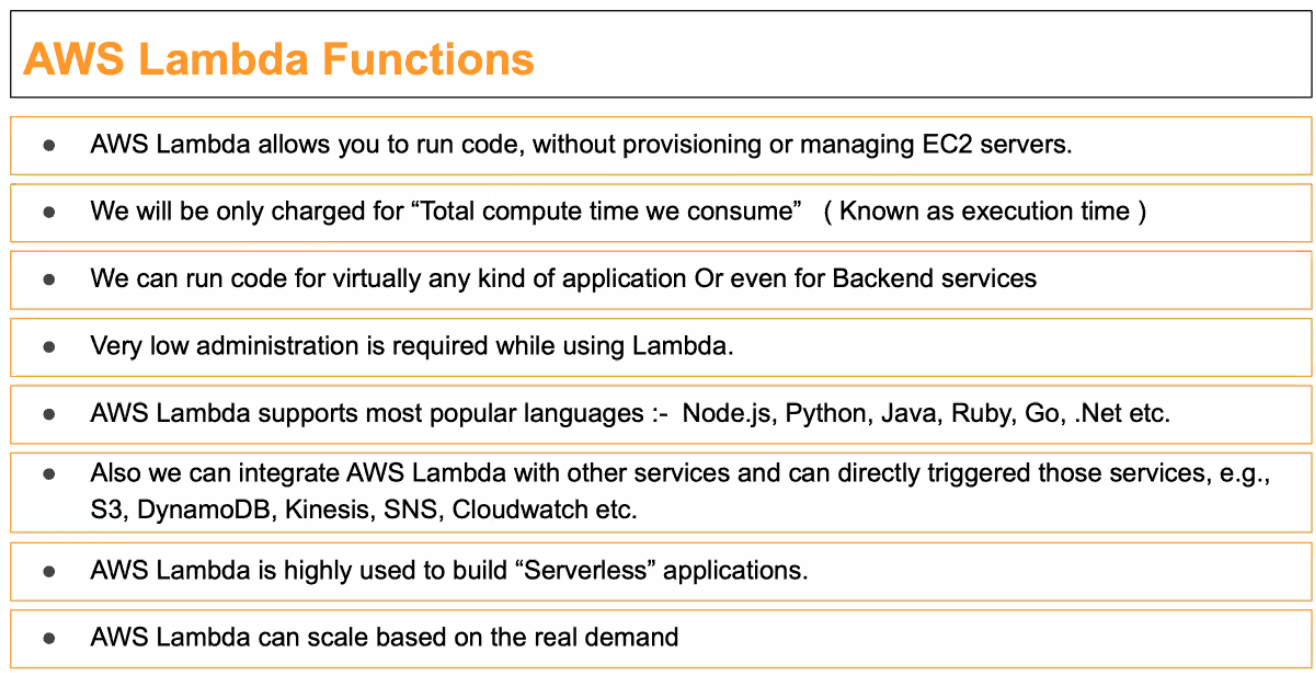
**SHOW DATABASES;**

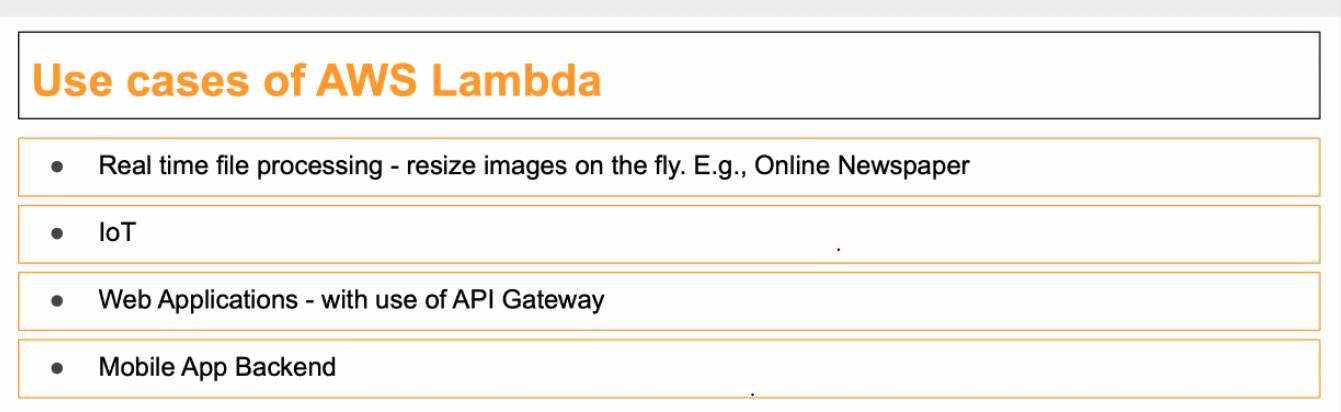
**############# ELASTIC BEANSTALK:PAAS…############**

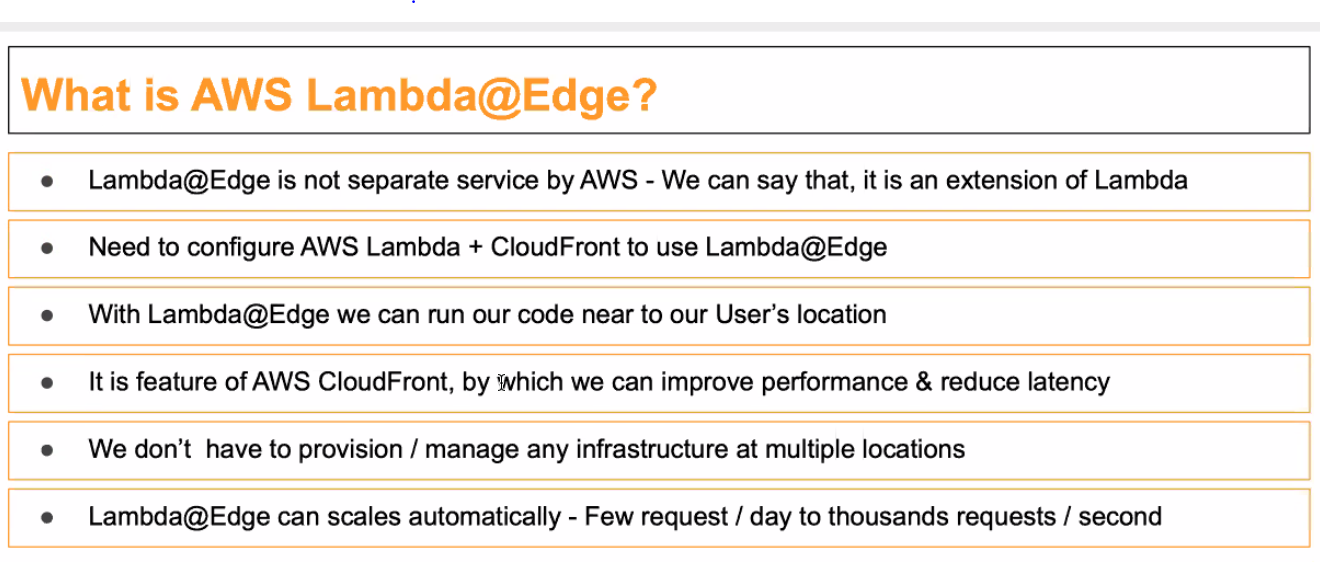


**IT IS COMPUTE SERVICE**

**############# LAMBDA###########**

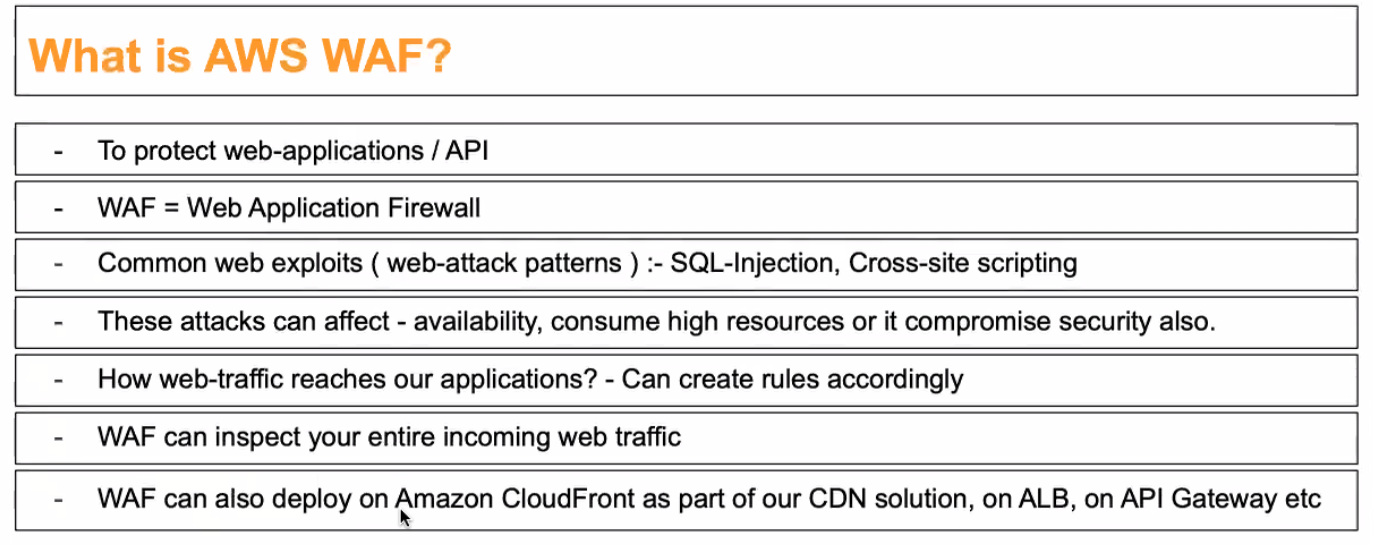




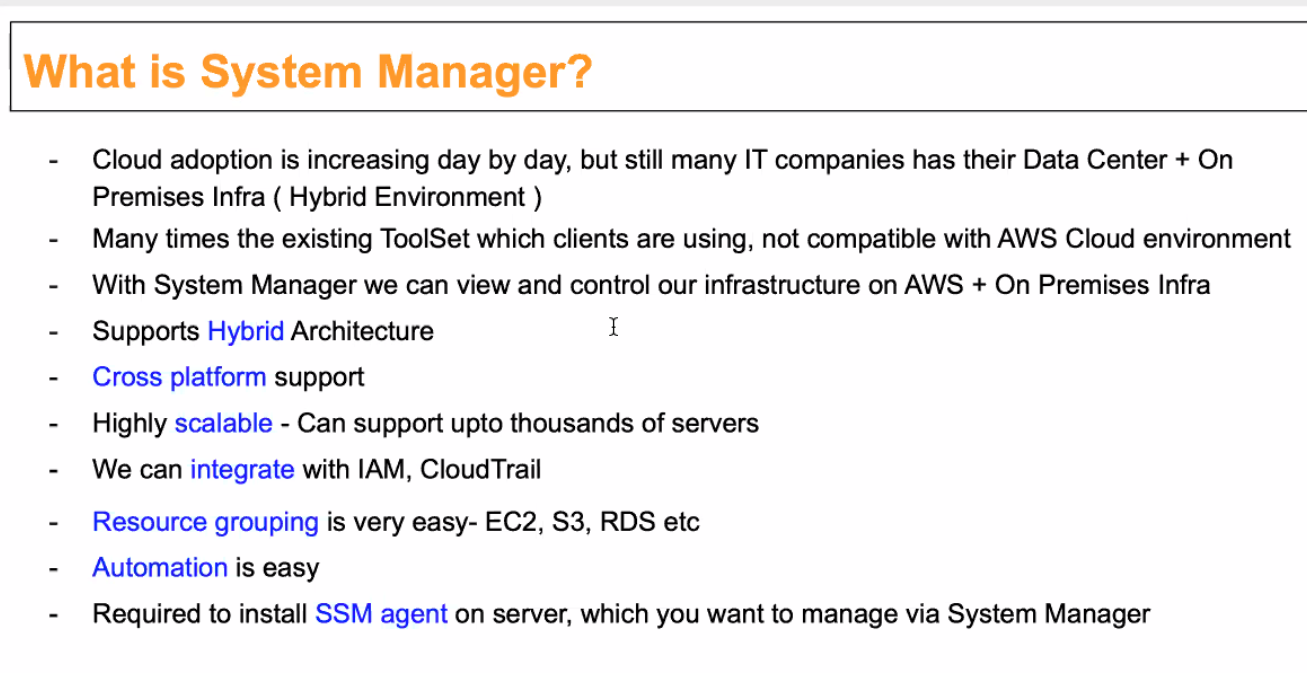


3 may 2020 Sunday session

WAF AND SHEILD

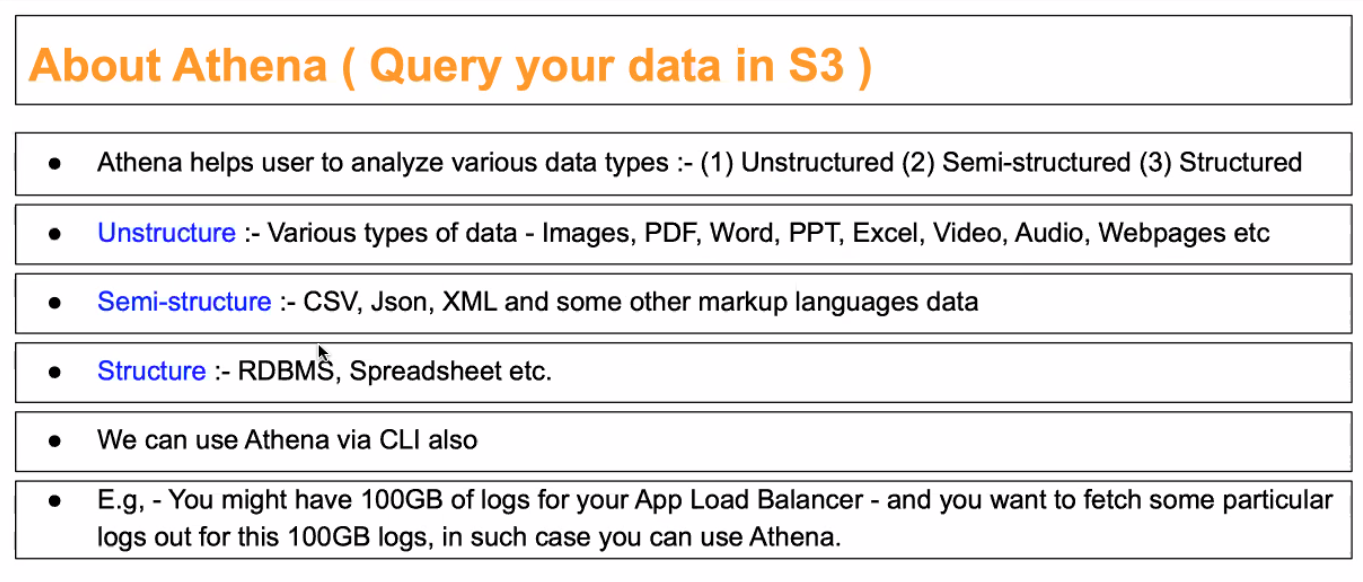
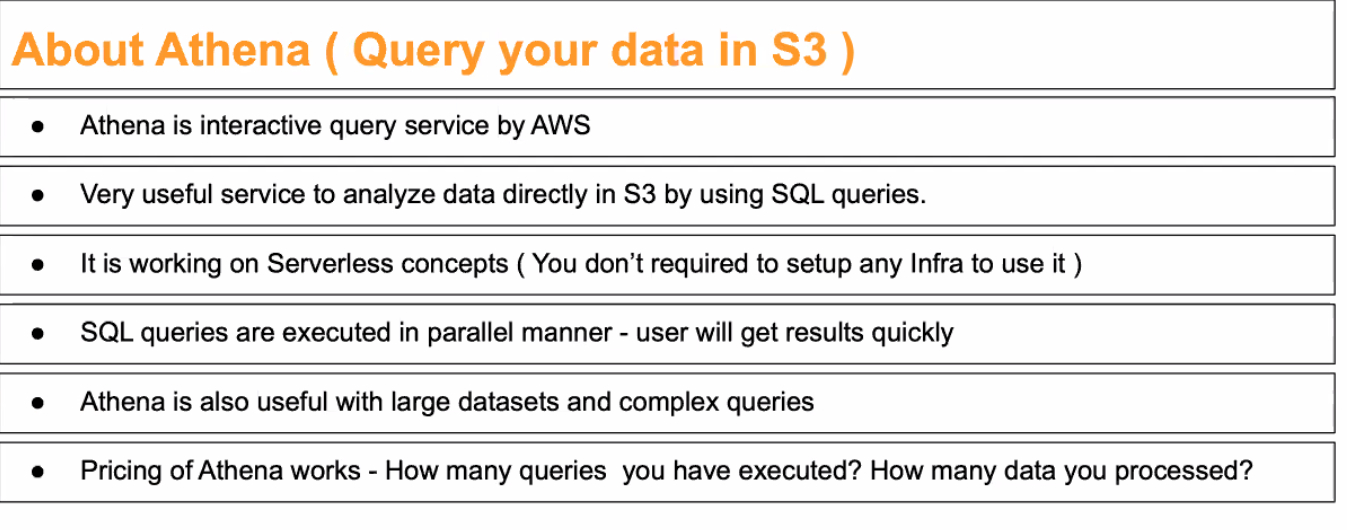


AWS SYSTEM MANAGER

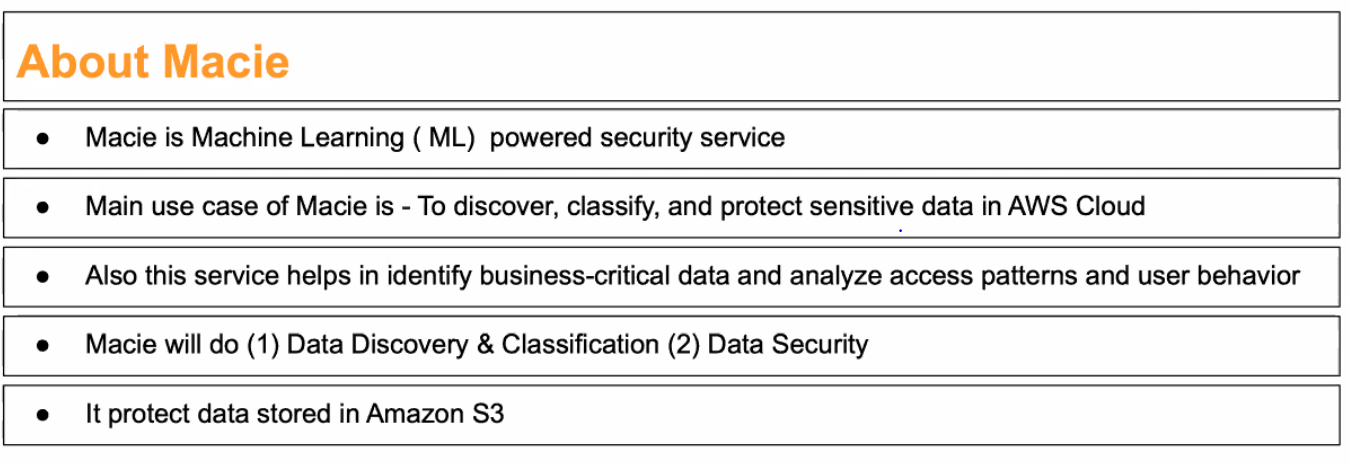


HSCL =🡺 HARDWARE SOFTWARE

ATHENA AWS



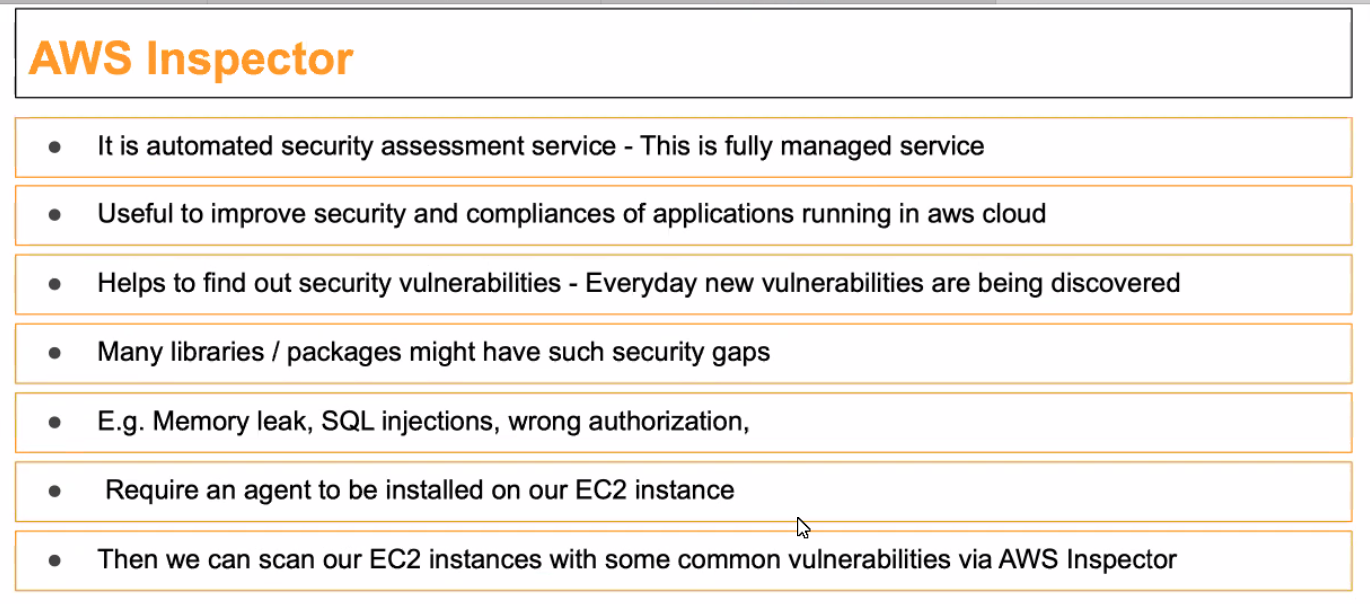
MACIE AWS

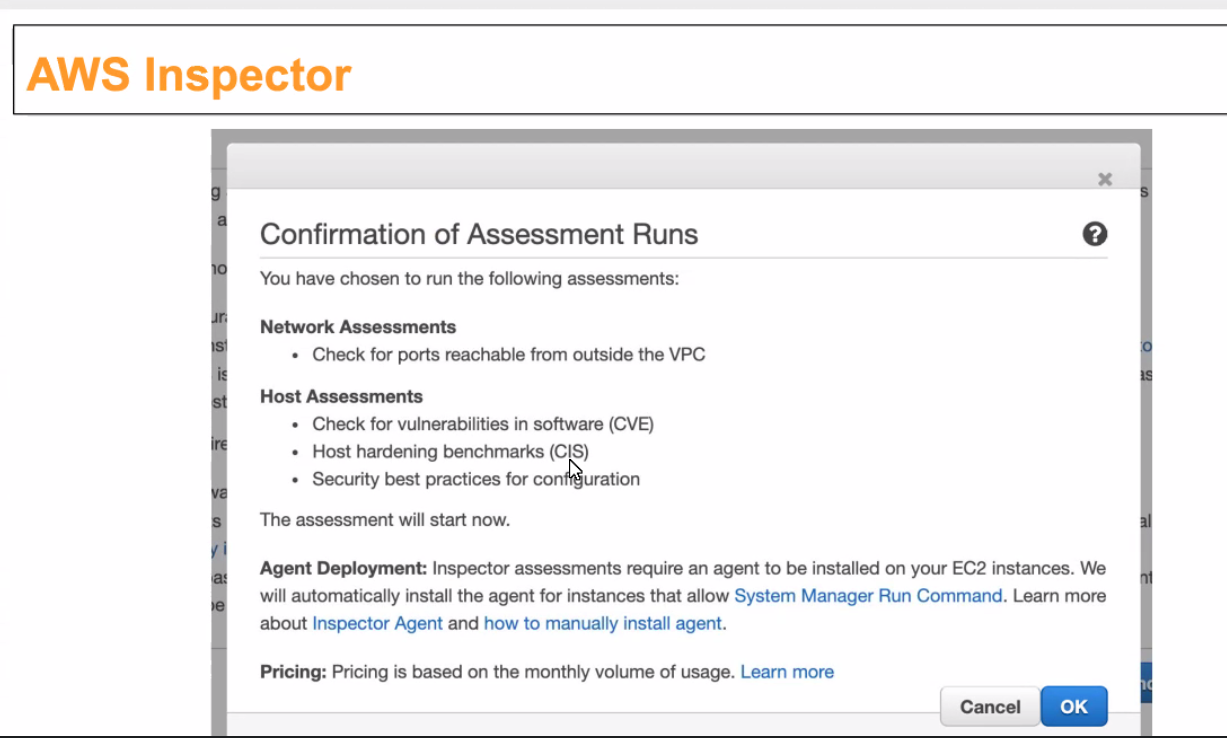


**AWS CLOUDTRAIL : AS A DEVOPS ENGINEER, YOU WILL FIND , HOW AND WHEN AND WHERE IT HAPPENED. HOW EC2 DELTETED OR ANY OTHER SERVICE. SO CT KEEPS A RECORD OF YOUR ACTIVITIES.**



AWS INSPECTOR:





AWS TRUSTED ADVISOR

