<u>Name :-anjali punsi</u>
<u>Class :- D20B</u>
<u>Roll no :- 57</u>
<u>Experiment No. 9</u>

Aim

To study and implement Security as a Service on AWS/Azure

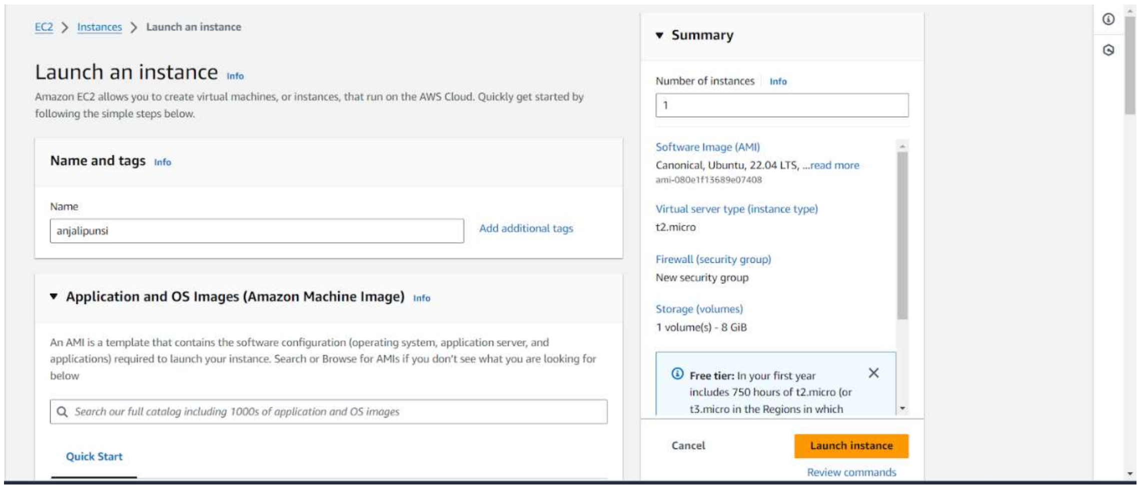
Theory

Security as a Service (SECaaS) is a cloud computing model that delivers security services over the internet. It allows organizations to access a wide range of security services without the need to maintain on-premises hardware or software. SECaaS providers typically offer services such as antivirus, intrusion detection, encryption, and identity and access management.

In the context of AWS and Azure, Security as a Service involves leveraging the security services provided by these cloud providers to secure your applications and data. AWS Offers services such as AWS Identity and Access Management (IAM), AWS Key Management Service (KMS), AWS Shield for DDoS protection, and AWS Inspector for security assessment. Azure provides services like Azure Active Directory for identity and access management, Azure Key Vault for key management, and Azure Security Center for threat detection and response.

By implementing Security as a Service on AWS/Azure, organizations can benefit from scalable, cost-effective security solutions that help protect their data and applications from cyber threats.

Steps and Output:-
Steps 1 :- Login to your AWS Management Console and Create a new AWS EC2 instance with Ubuntu OS.



EC2 > Instances > Launch an instance

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name: anjalipunsi

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

Quick Start

Summary

Number of instances Info: 1

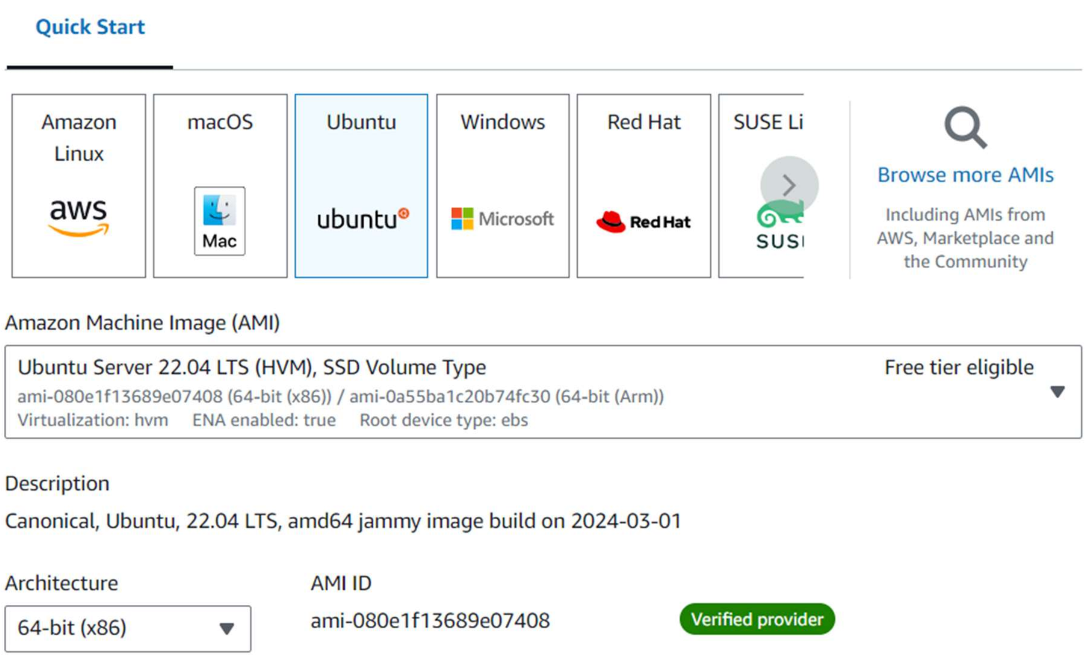
Software Image (AMI): Canonical, Ubuntu, 22.04 LTS, ...read more
ami-080e1f13689e07408

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which

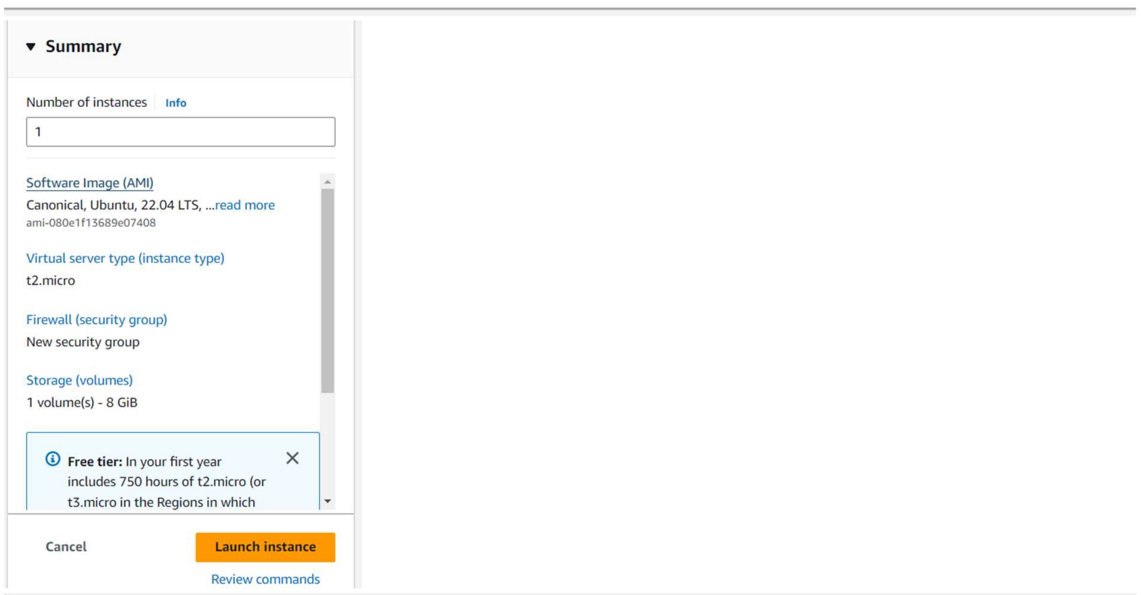


Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-080e1f13689e07408 (64-bit (x86)) / ami-0a55ba1c20b74fc30 (64-bit (Arm))
Virtualization: hvm   ENA enabled: true   Root device type: ebs

Description
Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2024-03-01

Architecture: 64-bit (x86)
AMI ID: ami-080e1f13689e07408
Verified provider



Number of instances Info
1

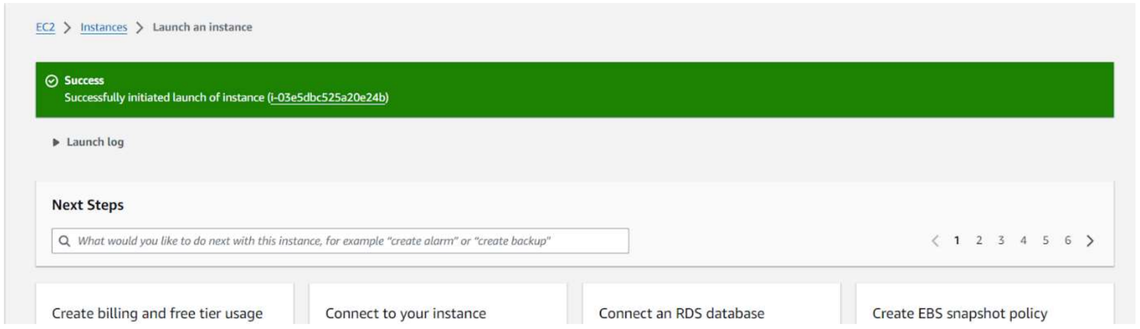
Software Image (AMI)
Canonical, Ubuntu, 22.04 LTS, ...read more
ami-080e1f13689e07408

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

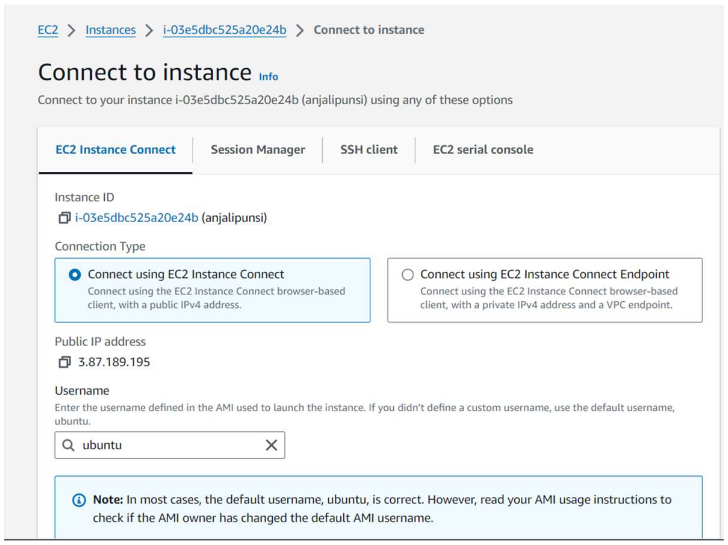
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which


EC2 > Instances > Launch an instance

Success
Successfully initiated launch of instance (i-03e5dbc525a20e24b)

Launch log
Next Steps
What would you like to do next with this instance, for example "create alarm" or "create backup"
1 2 3 4 5 6
Create billing and free tier usage
Connect to your instance
Connect an RDS database
Create EBS snapshot policy

Connect with the EC2 instance using the AWS built in console.



EC2 > Instances > i-03e5dbc525a20e24b > Connect to instance

Connect to instance Info

Connect to your instance i-03e5dbc525a20e24b (anjalipunsi) using any of these options

Instance ID

i-03e5dbc525a20e24b (anjalipunsi)

Connection Type

(●) Connect using EC2 Instance Connect
Connect using the EC2 Instance Connect browser-based client, with a public IPv4 address.

( ) Connect using EC2 Instance Connect Endpoint
Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

Public IP address

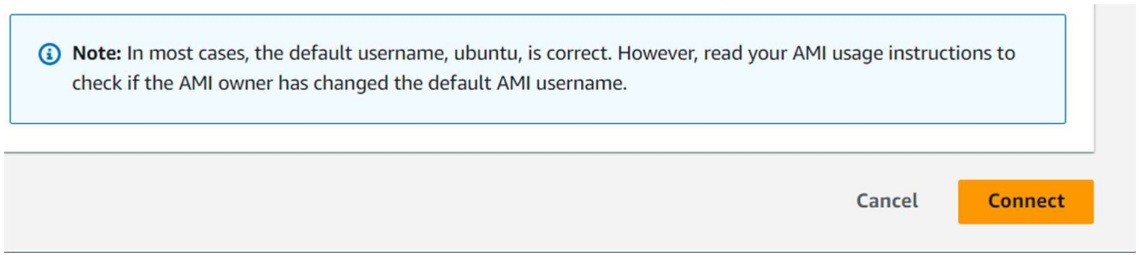
3.87.189.195

Username

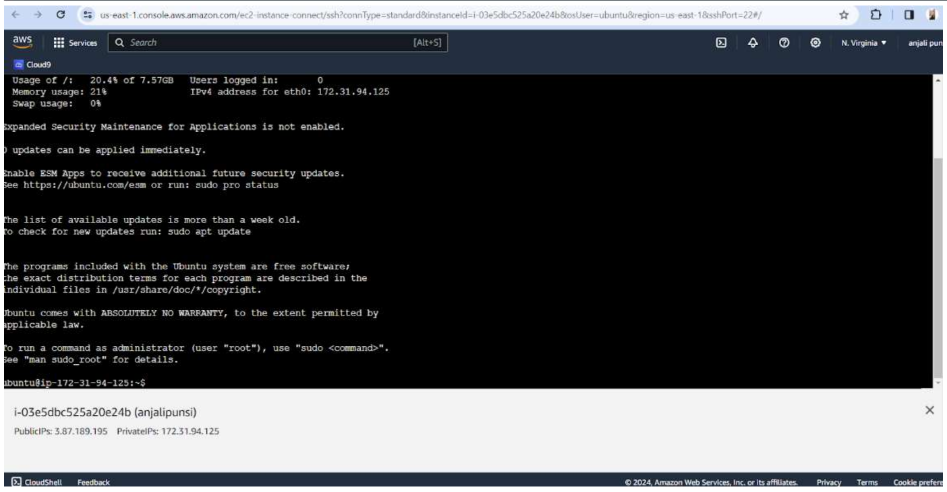
Enter the username defined in the AMI used to launch the instance. If you didn't define a custom username, use the default username, ubuntu.

ubuntu

Note: In most cases, the default username, ubuntu, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.



Note: In most cases, the default username, ubuntu, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.



us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId=i-03e5dbc525a20e24b&osUser=ubuntu&region=us-east-1&sshPort=22#/

```
 Usage of /:   20.4% of 7.57GB   Users logged in:       0
 Memory usage: 21%               IPv4 address for eth0: 172.31.94.125
 Swap usage:   0%

xpanded Security Maintenance for Applications is not enabled.

 updates can be applied immediately.

nable ESM Apps to receive additional future security updates.
ee https://ubuntu.com/esm or run: sudo pro status


he list of available updates is more than a week old.
o check for new updates run: sudo apt update


he programs included with the Ubuntu system are free software;
he exact distribution terms for each program are described in the
ndividual files in /usr/share/doc/*/copyright.

buntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
pplicable law.

o run a command as administrator (user "root"), use "sudo <command>".
ee "man sudo_root" for details.

buntu@ip-172-31-94-125:~$
```

i-03e5dbc525a20e24b (anjalipunsi)

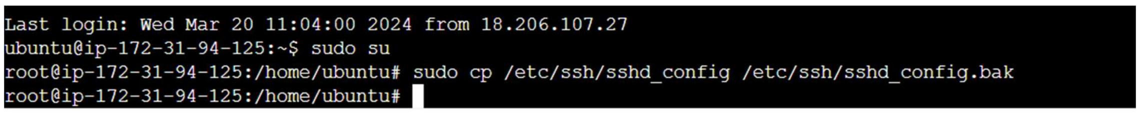
PublicIPs: 3.87.189.195 PrivateIPs: 172.31.94.125

<u>Step 2 :-</u>

Create a backup of the file using the following cp command:

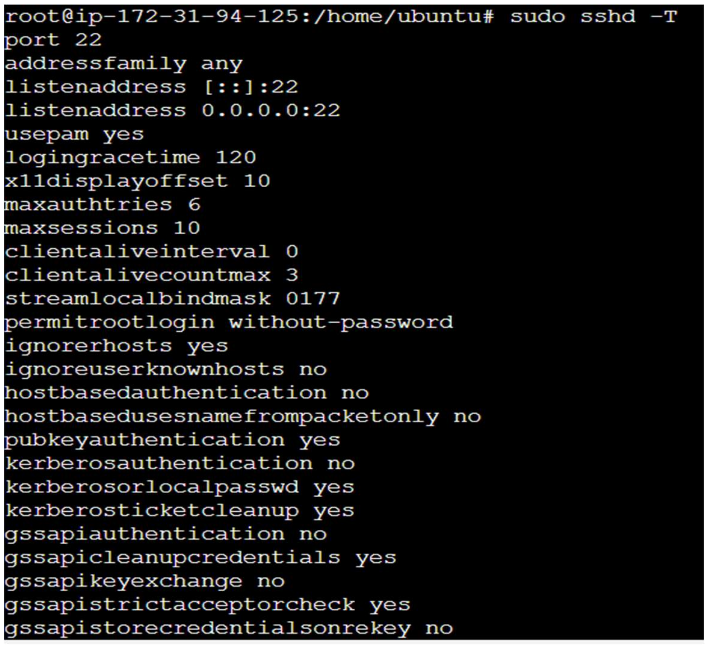
`sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config.bak`

This will save a backup copy of the file to /etc/ssh/sshd_config.bak.



```
Last login: Wed Mar 20 11:04:00 2024 from 18.206.107.27
ubuntu@ip-172-31-94-125:~$ sudo su
root@ip-172-31-94-125:/home/ubuntu# sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config.bak
root@ip-172-31-94-125:/home/ubuntu#
```

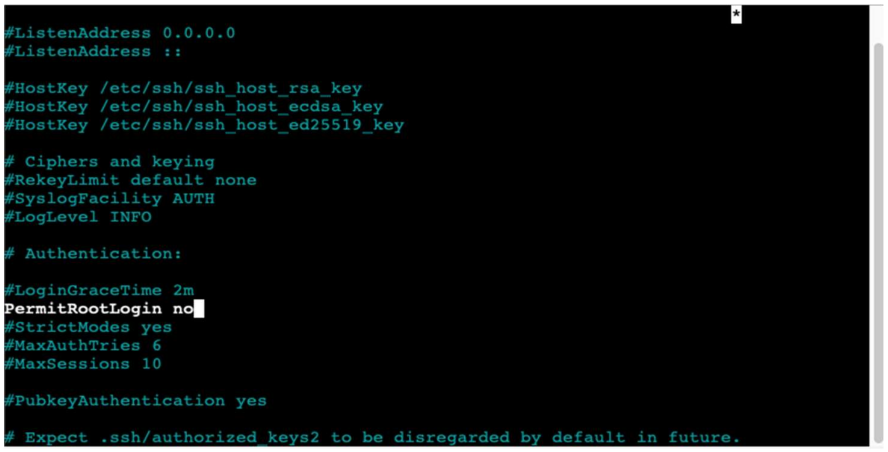
Step 3 :- run the following command: `sudo sshd -T`



```
root@ip-172-31-94-125:/home/ubuntu# sudo sshd -T
port 22
addressfamily any
listenaddress [::]:22
listenaddress 0.0.0.0:22
usepam yes
logingracetime 120
x11displayoffset 10
maxauthtries 6
maxsessions 10
clientaliveinterval 0
clientalivecountmax 3
streamlocalbindmask 0177
permitrootlogin without-password
ignorerhosts yes
ignoreuserknownhosts no
hostbasedauthentication no
hostbasedusesnamefrompacketonly no
pubkeyauthentication yes
kerberosauthentication no
kerberosorlocalpasswd yes
kerberosticketcleanup yes
gssapiauthentication no
gssapicleanupcredentials yes
gssapikeyexchange no
gssapistrictacceptorcheck yes
gssapistorecredentialsonrekey no
```

Step 5:- `sudo nano /etc/ssh/sshd_config :` the PermitRootLogin option to no by uncommenting or editing the line in sshd_config:

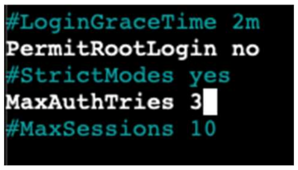
`PermitRootLogin no`



```
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
```

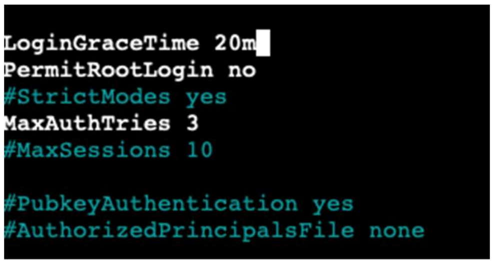
Step 6 :- Next, you can limit the maximum number of authentication attempts for a particular login session by configuring the MaxAuthTries option in sshd_config

`MaxAuthTries 3`



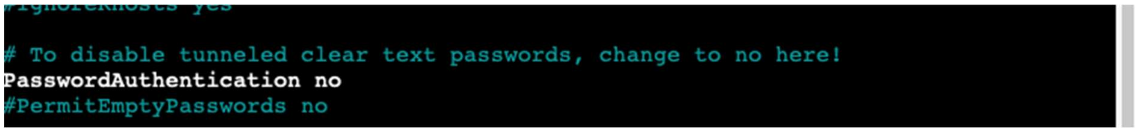
```
#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
MaxAuthTries 3
#MaxSessions 10
```

Step 7 :- `LoginGraceTime 20`



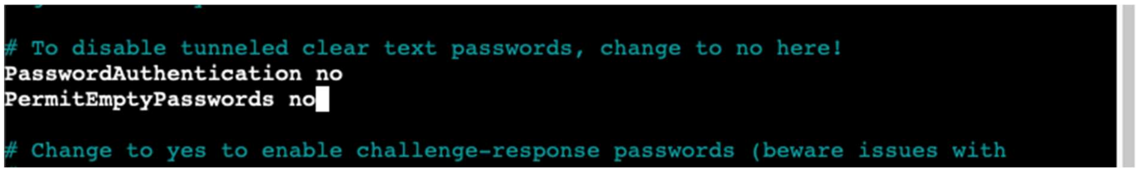
```
LoginGraceTime 20m
PermitRootLogin no
#StrictModes yes
MaxAuthTries 3
#MaxSessions 10

#PubkeyAuthentication yes
#AuthorizedPrincipalsFile none
```

Step 8 :- `PasswordAuthentication no`



```
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
#PermitEmptyPasswords no
```

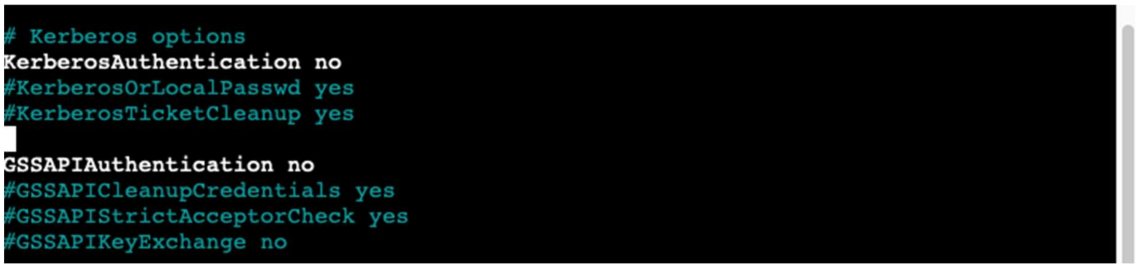
`Step 9 :- PermitEmptyPasswords no`



```
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with
```

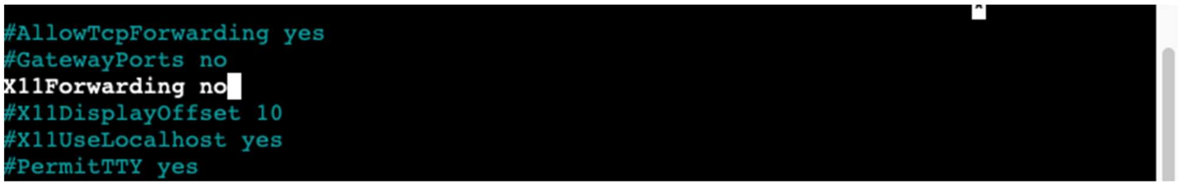
`Step 10 :-` If these are not required, you can disable them to further reduce the attack surface of your SSH server in sshd_config

```
ChallengeResponseAuthentication no
KerberosAuthentication no
GSSAPIAuthentication no
```



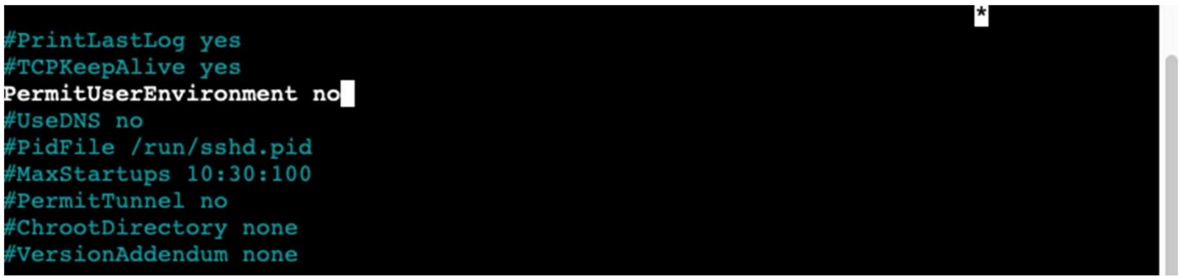
```
# Kerberos options
KerberosAuthentication no
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes

GSSAPIAuthentication no
#GSSAPICleanupCredentials yes
#GSSAPIStrictAcceptorCheck yes
#GSSAPIKeyExchange no
```

`Step 11:- X11Forwarding no`



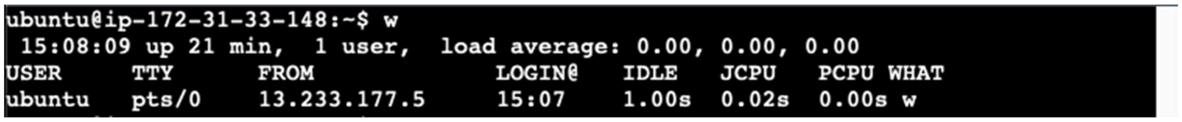
```
#AllowTcpForwarding yes
#GatewayPorts no
X11Forwarding no
#X11DisplayOffset 10
#X11UseLocalhost yes
#PermitTTY yes
```

Step 12 :- `PermitUserEnvironment no`



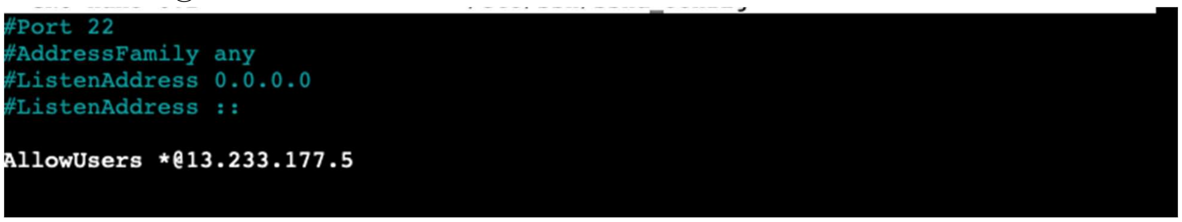
```
#PrintLastLog yes
#TCPKeepAlive yes
PermitUserEnvironment no
#UseDNS no
#PidFile /run/sshd.pid
#MaxStartups 10:30:100
#PermitTunnel no
#ChrootDirectory none
#VersionAddendum none
```

Step 13 :- Implementing an IP Address Allowlist :- You can identify the IP address that you're currently connecting to your server with by using the w command:



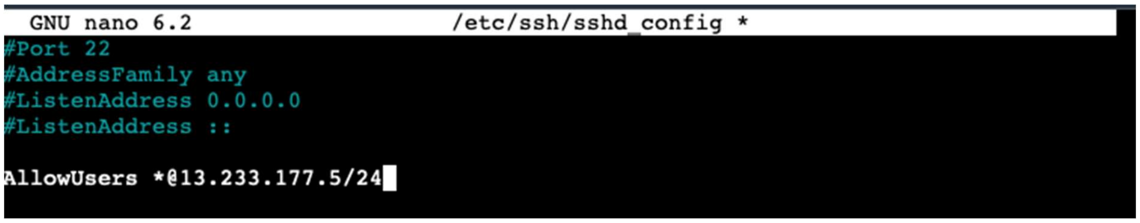
```
ubuntu@ip-172-31-33-148:~$ w
 15:08:09 up 21 min,  1 user,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
ubuntu   pts/0    13.233.177.5     15:07    1.00s  0.02s  0.00s w
```

Step 14 :- Locate your user account in the list and take a note of the connecting IP address.
Here we use the example IP of 13.233.177.5
Restrict all users to a specific IP address:
AllowUsers *@203.0.113.1



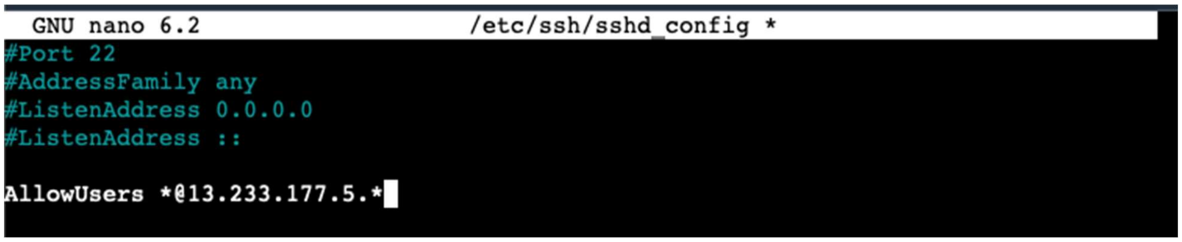
```
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

AllowUsers *@13.233.177.5
```

Step 15 :- AllowUsers *@203.0.113.0/24



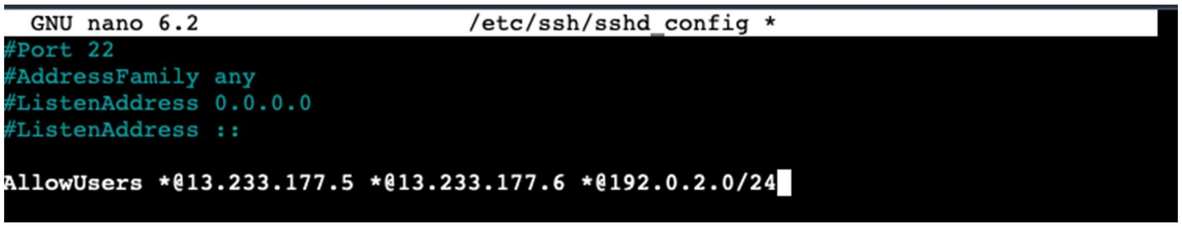
```
  GNU nano 6.2                     /etc/ssh/sshd_config *
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

AllowUsers *@13.233.177.5/24
```

Step 16 :- AllowUsers *@203.0.113.*



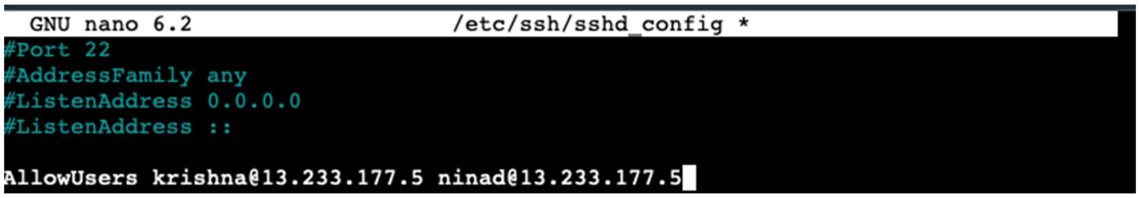
```
  GNU nano 6.2                     /etc/ssh/sshd_config *
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

AllowUsers *@13.233.177.5.*
```

Step 17 :- AllowUsers *@203.0.113.1 *@203.0.113.2 *@192.0.2.0/24 *@172.16.*.1



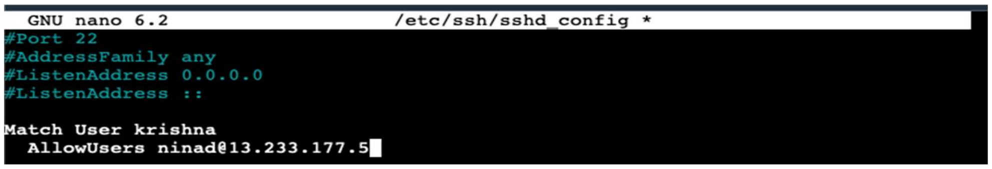
```
  GNU nano 6.2                     /etc/ssh/sshd_config *
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

AllowUsers *@13.233.177.5 *@13.233.177.6 *@192.0.2.0/24
```

Step 18 :- AllowUsers sammy@203.0.113.1 alex@203.0.113.2



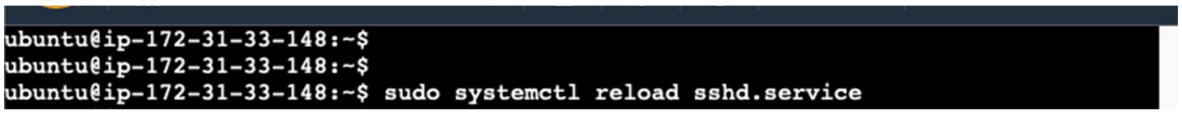
```
  GNU nano 6.2                     /etc/ssh/sshd_config *
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

AllowUsers krishna@13.233.177.5 ninad@13.233.177.5
```

Step 19 :- Match User ashley   AllowUsers ashley@203.0.113.1



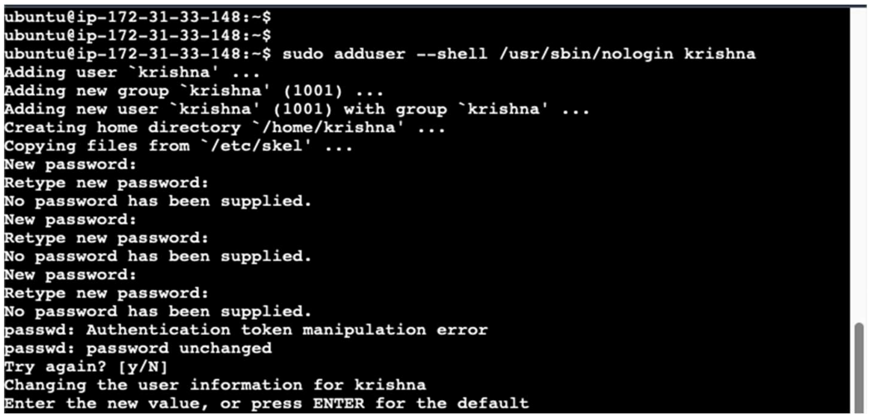
```
  GNU nano 6.2                     /etc/ssh/sshd_config *
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

Match User krishna
  AllowUsers ninad@13.233.177.5
```

Step 20 :- Save and close the file, and then proceed to test your syntax: sudo sshd -t
Step 21 :- `sudo systemctl reload sshd.service`



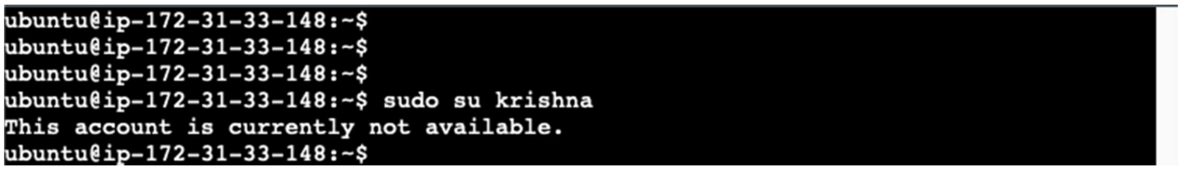
```
ubuntu@ip-172-31-33-148:~$
ubuntu@ip-172-31-33-148:~$
ubuntu@ip-172-31-33-148:~$ sudo systemctl reload sshd.service
```

Step 22 :- To create a new user with the nologin shell, use the following command:
sudo adduser --shell /usr/sbin/nologin krishna



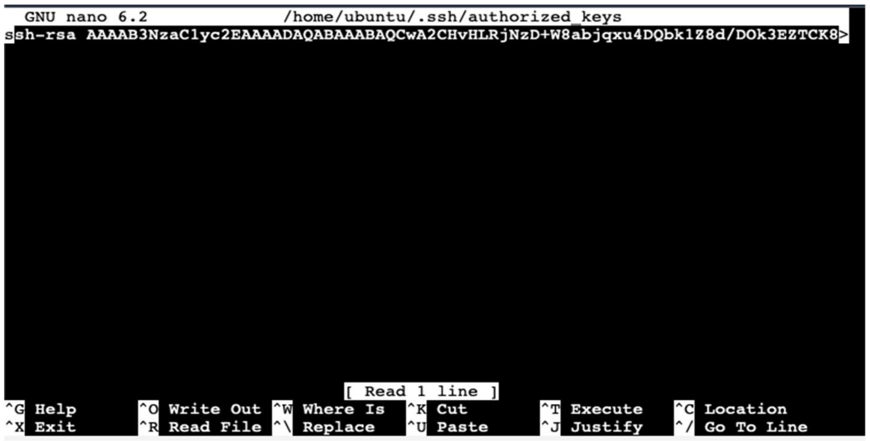
```
ubuntu@ip-172-31-33-148:~$
ubuntu@ip-172-31-33-148:~$
ubuntu@ip-172-31-33-148:~$ sudo adduser --shell /usr/sbin/nologin krishna
Adding user `krishna' ...
Adding new group `krishna' (1001) ...
Adding new user `krishna' (1001) with group `krishna' ...
Creating home directory `/home/krishna' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
No password has been supplied.
New password:
Retype new password:
No password has been supplied.
New password:
Retype new password:
No password has been supplied.
passwd: Authentication token manipulation error
passwd: password unchanged
Try again? [y/N]
Changing the user information for krishna
Enter the new value, or press ENTER for the default
```

Step 23 :- `sudo su alex`



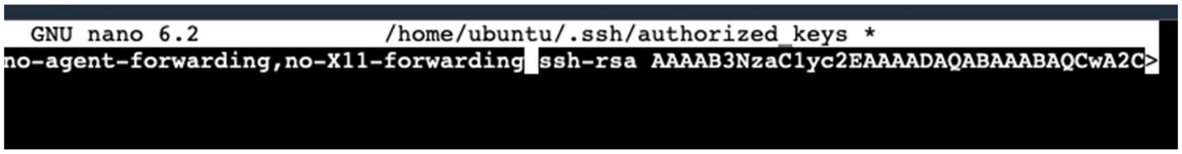
```
ubuntu@ip-172-31-33-148:~$
ubuntu@ip-172-31-33-148:~$
ubuntu@ip-172-31-33-148:~$
ubuntu@ip-172-31-33-148:~$ sudo su krishna
This account is currently not available.
ubuntu@ip-172-31-33-148:~$
```

Despite the rejection message on interactive logins, other actions such as file transfers will still be allowed.
Step 24 : Begin by opening your .ssh/authorized_keys file in nano or your preferred editor:
`nano ~/.ssh/authorized_keys`



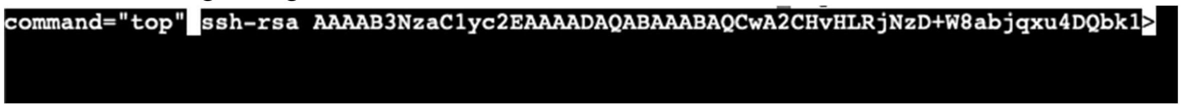
```
  GNU nano 6.2            /home/ubuntu/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCwA2CHvHLRjNzD+W8abjqxu4DQbk1Z8d/DOk3EZTCK8>




                              [ Read 1 line ]
^G Help       ^O Write Out ^W Where Is  ^K Cut        ^T Execute    ^C Location
^X Exit       ^R Read File ^\ Replace   ^U Paste      ^J Justify    ^/ Go To Line
```

Step 25 :- You can apply these to disable specific SSH features for specific keys. For example, to disable agent forwarding and X11 forwarding for a key, you would use the following configuration:



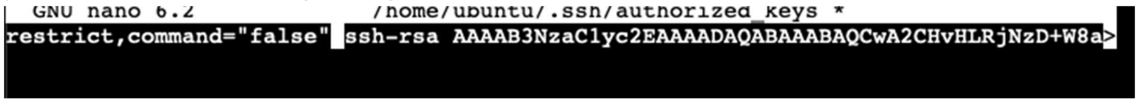
```
  GNU nano 6.2             /home/ubuntu/.ssh/authorized_keys *
no-agent-forwarding,no-X11-forwarding ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCwA2C>
```

Step 26 :- server configuration file is overwritten, edited, and so on. For example, to force users authenticating using a specific key to execute a specific command upon login, you can add the following configuration:



```
command="top" ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCwA2CHvHLRjNzD+W8abjqxu4DQbk1>
```

Step 27 :- Finally, to best use the per-key restrictions for the SFTP-only user that you created, you can use the following configuration:



```
  GNU nano 6.2             /home/ubuntu/.ssh/authorized_keys *
restrict,command="false" ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCwA2CHvHLRjNzD+W8a>
```

The restrict option will disable all interactive access, and the command="false" option acts as a second line of defense in the event that the ForceCommand option or nologin shell were to fail.Save and close the file to apply the configuration. This will take effect immediately for all new logins, so you don’t need to reload OpenSSH manually.

Conclusion

In conclusion, you reviewed your OpenSSH server configuration and implemented various hardening measures to help secure your server. The options that you configured have reduced the overall attack surface of your server by disabling unused features and locking down the access of specific users.