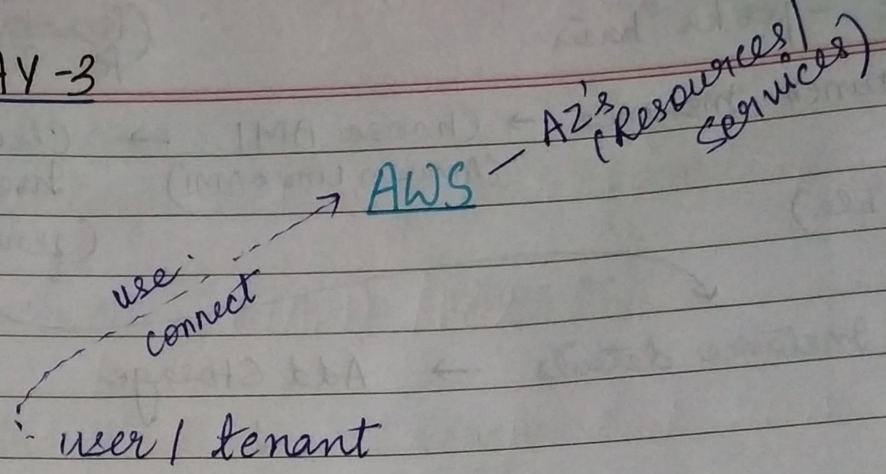
DAY-3



▷ 3 ways to connect to AWS:

① Graphical way :- Web UI (manual thing) (Web Uses Interface)

Industry work ② Command line Interface (CLI) - Automation

③ Programming language ← Python. Automation
(SDK)
↳ code — Terraform.

▷ Cloud Engineers have black screen OS, so only option available to connect to AWS is CLI.

Startup / Team
A ✓ — all power → User ← Root Account / user / owner)
↳ email
B ✓ one role
C ✓

→ create a new account (user) in root account with limited priviledges.

→ limited power (ROLE) / access.
Policy

Identity

Account Id - IAM user will be the part of same root account Id.

▷ AWS IAM

Identity and Access Management

Openstack
↓
Keystone

→ We can create new users with limited access in your root account.
(RULE)

AWS
WebApp.
Program/OS/Website
↓
If we want to login
↓
We need username / password. (Manual)
↓
Slow.

(Automation) Some code that logins / some command. is used for login.
↓
fast

→ we never use password here
→ we use key.

(Program will login faster in place of human with use of Key/Token)

(key) access → user
(key) secret →

IAM User Creation steps: → AWS Mgmt Console → IAM → Users → Access type
- o Program
- o AWS Mgmt Console

↓
Permission
(Power / Permission)
↓
Tags
↓ (Review)
Create → download the Credentials (as shown once)

Power Users Access [Permission]
→ do any thing
→ no visibility to account info/billing
→ no access to IAM.

(Externally) Name ↙ recognize any image through ID (AMI) (Internally)

▷ GUI :-

Login and select region (ap-south-1)

↓

Service (IAM) / EC2

↓

Intension: launch and run instance.

↓

Choose AMI (Which OS we want to launch)

Amazon linux 2 AMI, SSD Volume

↓

Instance type: t2.micro.

↓

no of Intance count → 1

(network) VPC - default vpc-260c114e

(Subnet) ~~subnet - aa57e5d1~~.

subnet - 1a Subnet 8848...

↓

Add Storage (EBS Creates Automatically)

↓

Add Tags. Key name Value firstos

↓

Security: firewall: launch-wizard-1 (security group name)

↓

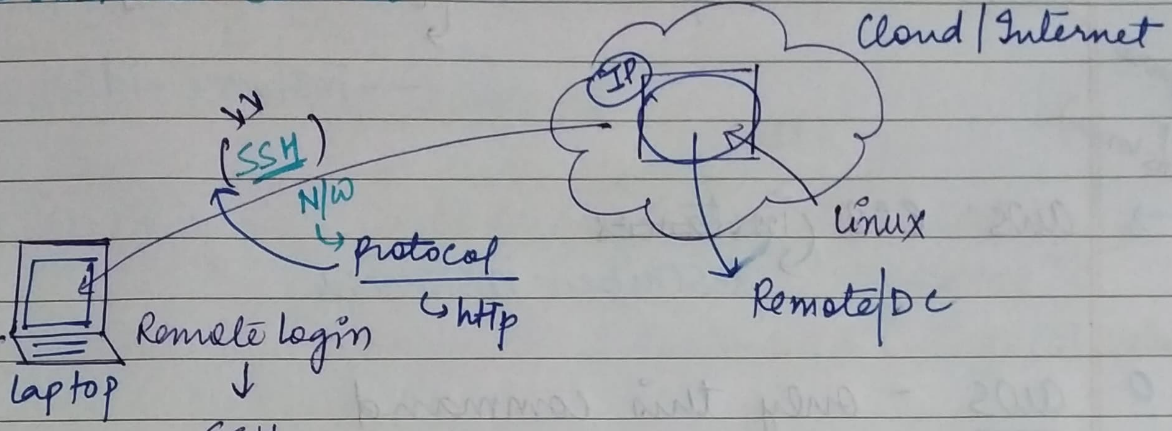
Review / Launch.

↓

Choose existing keypair / Create new key and download. Key Pair.

(Create Key for OS)

↪ Browser based login only for Amazon Linux.
EC2 Instance Connect



Remote login
↓
SSH.

To connect to any other OS → Program/Command.(cmd)
↳ SSH ↳ putty

Require for SSH → IPV4 public
Username → ec2-user
password → OS password.(Key)

↓ login

To connect to redhat OS ↗ Command Prompt (CLI)/Command (program) → aws.

command → lscpu (redhat cmd)

~~SSH -l ec2~~

↓
aws cli (install for windows)
↓
Install the CLI version2
↓
Cmd (CLI) aws --version (check)
↓
aws configure
↳ aws access Key → —
↳ aws secret Key → —
↓
region → ↓ [ap-south-1a]
↓
defuault output format -x. (none)

enable SSH feature in windows

optional feature
↓
manage optional feature
↓
add a feature
↓
OpenSSH Client
↓
cmd (command prompt)
↓
ssh

→ aws ec2 start-instances
stop-instances [instance id]
--instance-ids

(require
stop Instance)

→ aws ec2 ~~instances~~
describe-instances

- aws - only this command
  - we can create our own command.

→ aws ec2 help.

→ aws ec2 describe-Key-pairs.

→ create Key using CLI

→ aws ec2 describe-Key-pairs --query KeyPairs

- (give value of this variable)

```
[
  {
  } ] value 1
  {
  } ] value 2
]
```

- aws ec2 describe-Key-pairs --query KeyPairs[0]["KeyN

▷ retrieve ip of first OS :- (Public IP Address)   Automatic Retrieve of IP.

↓

Variable - PublicIPAddress

--query }

← for all.

Reservations[0]. Instances [$\overset{*}{0}$]. ["PublicIPAddress", "KeyName"]

Reservations [0]. Instances [*]. {["InstanceId", Tags[0].values]

* ssh -i mykeypem -l ec2-user publicIPAddress.

→ aws ec2 terminate-instances InstanceID.

launch Instance from Command line

```
# aws ec2 help.
  aws ec2 run-instances help.
```

↓

```
aws ec2 run-instances --image-id ami-id
--instance-type t2.micro --count 1 --subnet-id
subnetid --security-group-ids launch-wizard -1
--key-name myKey1111
```