Jenkins Real_World_Project_5

Configuring Jenkins CICD pipeline to deploy to AWS EKS.

Git project link: https://github.com/srizvi0/Real_World_Jenkins_Proj_C.git

Steps are laid out below

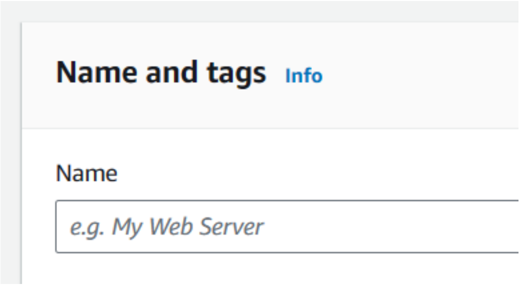
AMAZON EC2

Step 1) Sign in to AWS Management Console

Step 2) Navigate to EC2 Dashboard

Step 3) Launch Instance: click on launch instance button

Step 4) Enter name of EC2

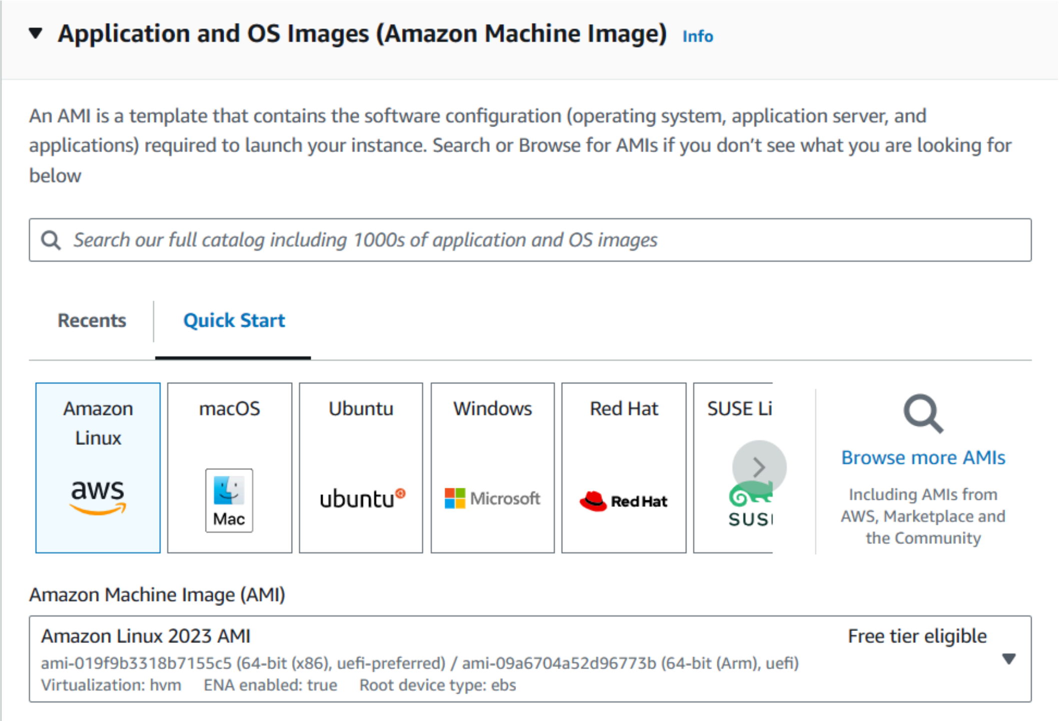


Name and tags Info

Name

e.g. My Web Server

Step 5) Choose application and OS image



▼ Application and OS Images (Amazon Machine Image) Info

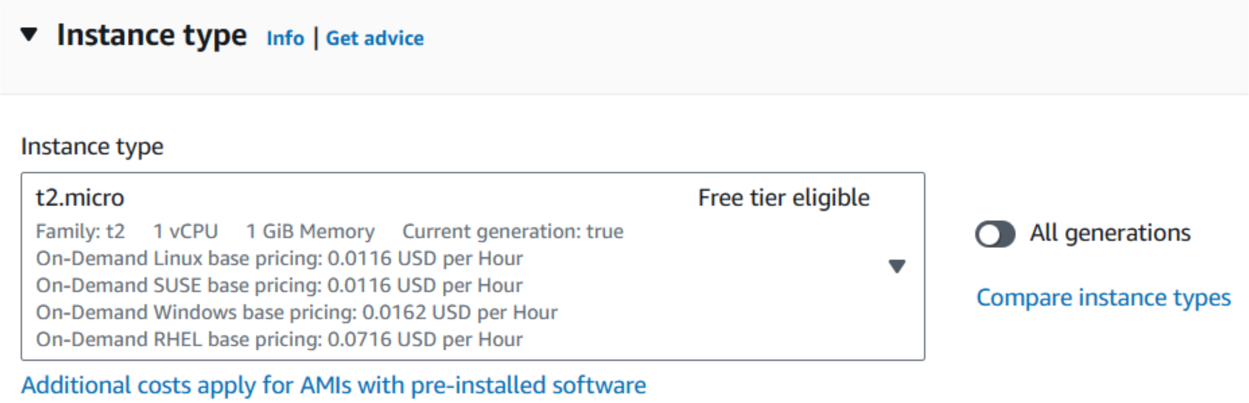
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

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

Amazon Machine Image (AMI)

Amazon Linux 2023 AMI — Free tier eligible
ami-019f9b3318b7155c5 (64-bit (x86), uefi-preferred) / ami-09a6704a52d96773b (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs

Step 6) Choose instance type

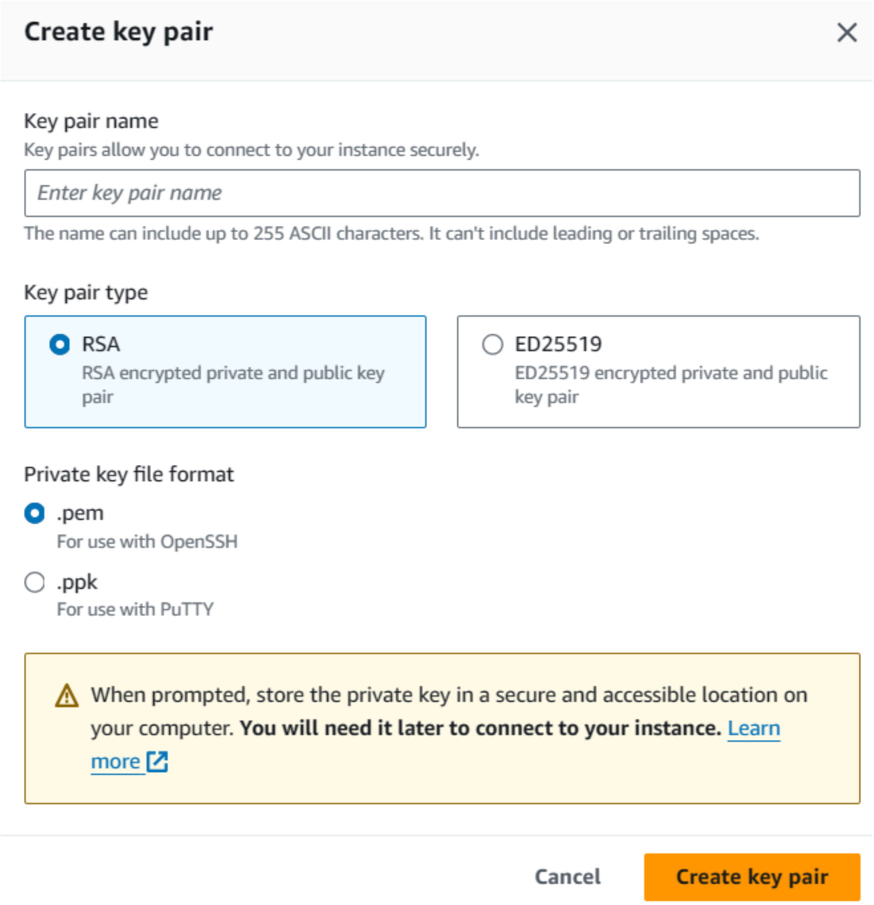


▼ Instance type Info | Get advice

Instance type

t2.micro — Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Linux base pricing: 0.0116 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour
On-Demand Windows base pricing: 0.0162 USD per Hour
On-Demand RHEL base pricing: 0.0716 USD per Hour

Step 7) Create new key-pair with following configurations



Create key pair

Key pair name

Key pairs allow you to connect to your instance securely.

Enter key pair name

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

● RSA
RSA encrypted private and public key pair

○ ED25519
ED25519 encrypted private and public key pair

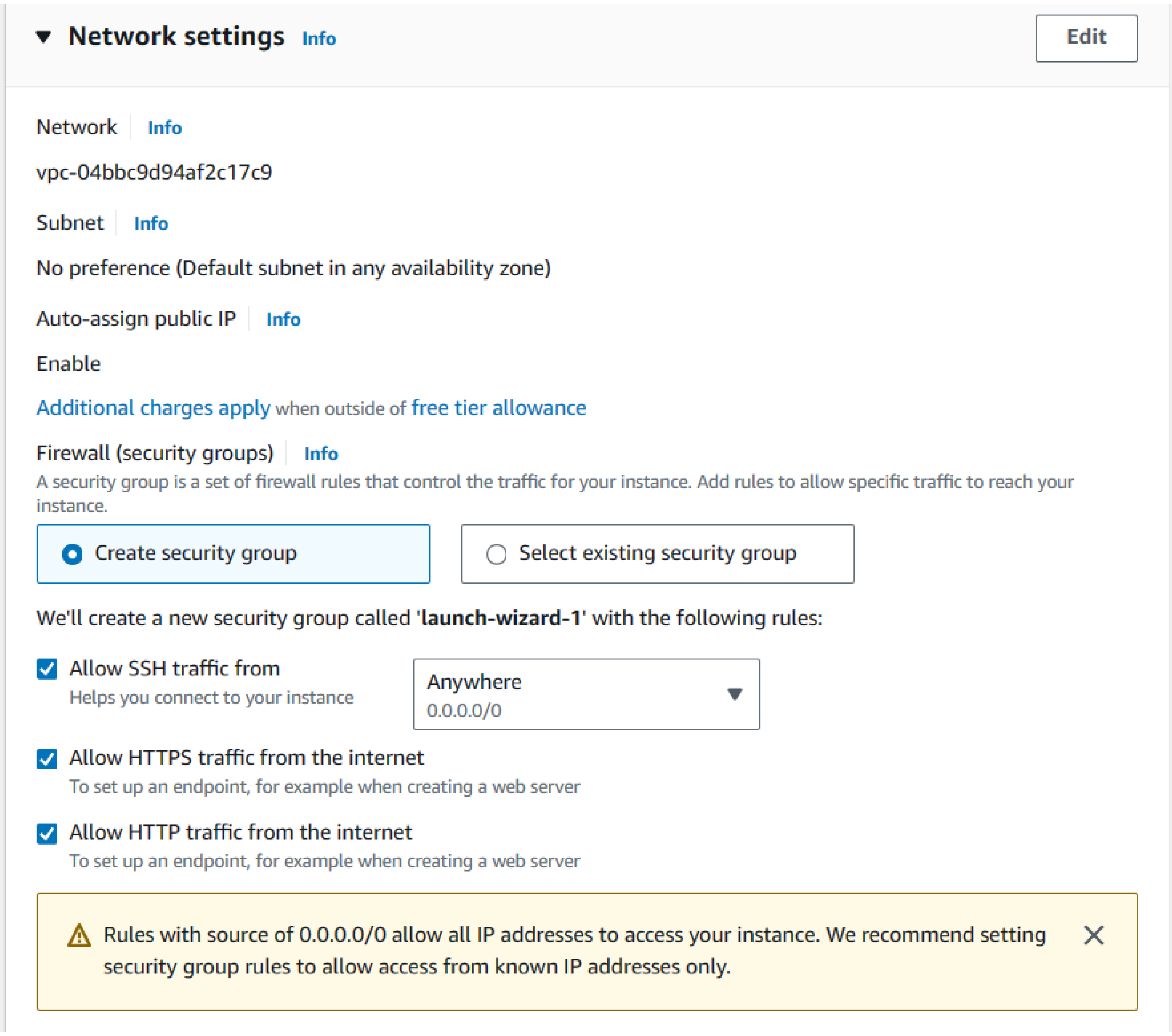
Private key file format

● .pem
For use with OpenSSH

○ .ppk
For use with PuTTY

⚠ When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** Learn more

Step 8) Create a new security group



▼ **Network settings** Info

Edit

Network | Info

vpc-04bbc9d94af2c17c9

Subnet | Info

No preference (Default subnet in any availability zone)

Auto-assign public IP | Info

Enable

Additional charges apply when outside of free tier allowance

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:

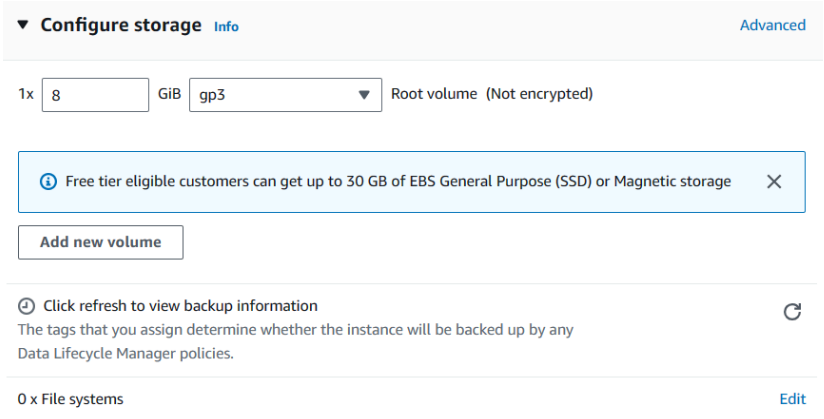
☑ Allow SSH traffic from
Helps you connect to your instance
Anywhere ▼
0.0.0.0/0

☑ Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server

☑ Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only. ✕

Step 9) Configure storage



▼ **Configure storage** Info Advanced

1x 8 GiB gp3 ▼ Root volume (Not encrypted)

ⓘ Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage ✕

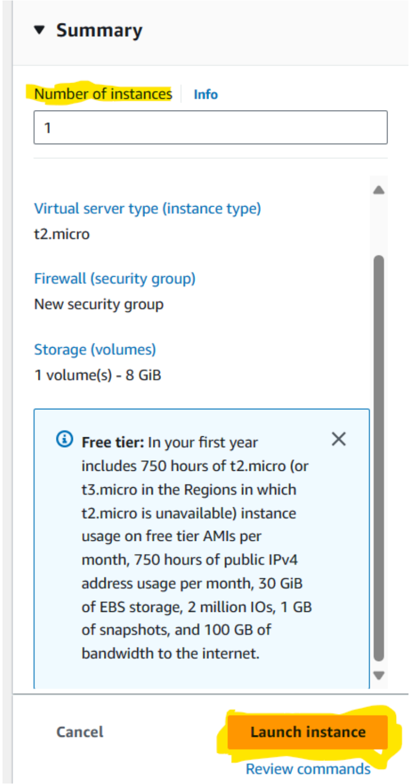
Add new volume

Click refresh to view backup information
The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.

0 x File systems Edit

Step 10) Launch instance

Once complete, specify number of instances and then launch the instance.



▼ Summary

Number of instances Info

1

Virtual server type (instance type)
t2.micro

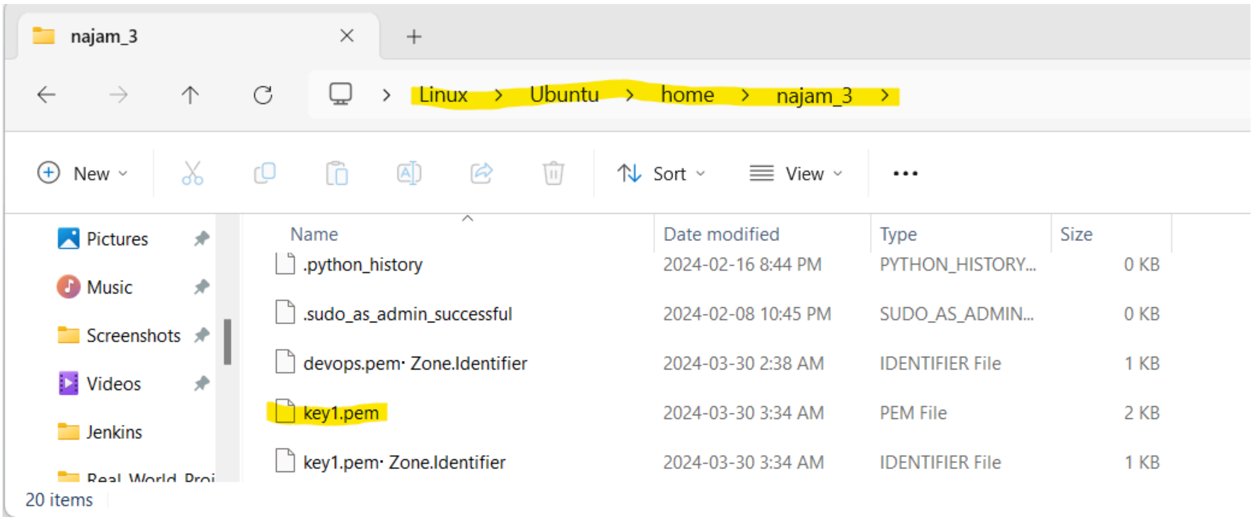
Firewall (security group)
New security group

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 t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

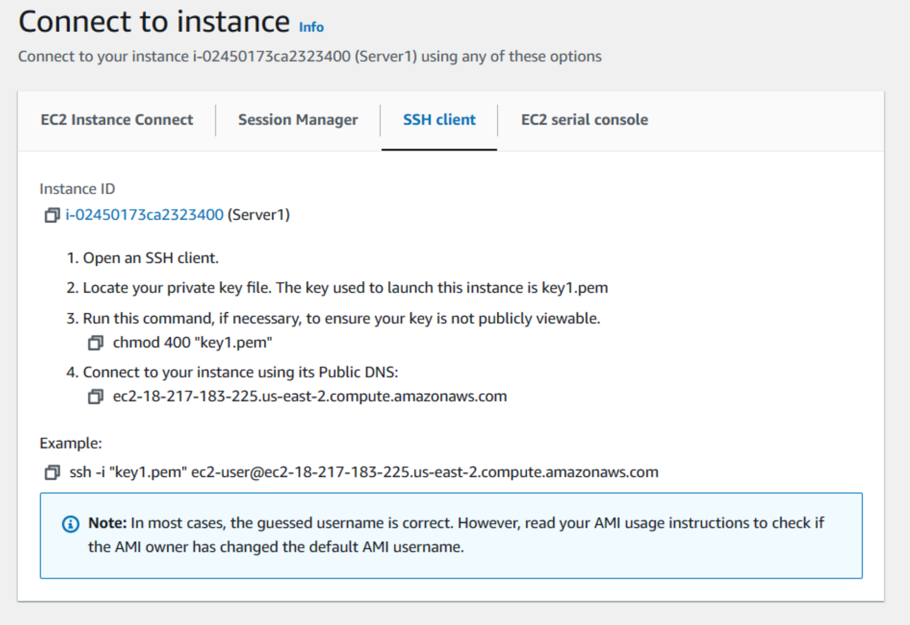
<u>Connecting to EC2 VM</u>

Step 1) Download “Key1.pem” file and move it to the linux->ubuntu server directory



Name	Date modified	Type	Size
.python_history	2024-02-16 8:44 PM	PYTHON_HISTORY...	0 KB
.sudo_as_admin_successful	2024-02-08 10:45 PM	SUDO_AS_ADMIN...	0 KB
devops.pem: Zone.Identifier	2024-03-30 2:38 AM	IDENTIFIER File	1 KB
key1.pem	2024-03-30 3:34 AM	PEM File	2 KB
key1.pem: Zone.Identifier	2024-03-30 3:34 AM	IDENTIFIER File	1 KB

Step 2) click on running instances and connect using the ssh client



Connect to instance Info

Connect to your instance i-02450173ca2323400 (Server1) using any of these options

Instance ID

i-02450173ca2323400 (Server1)

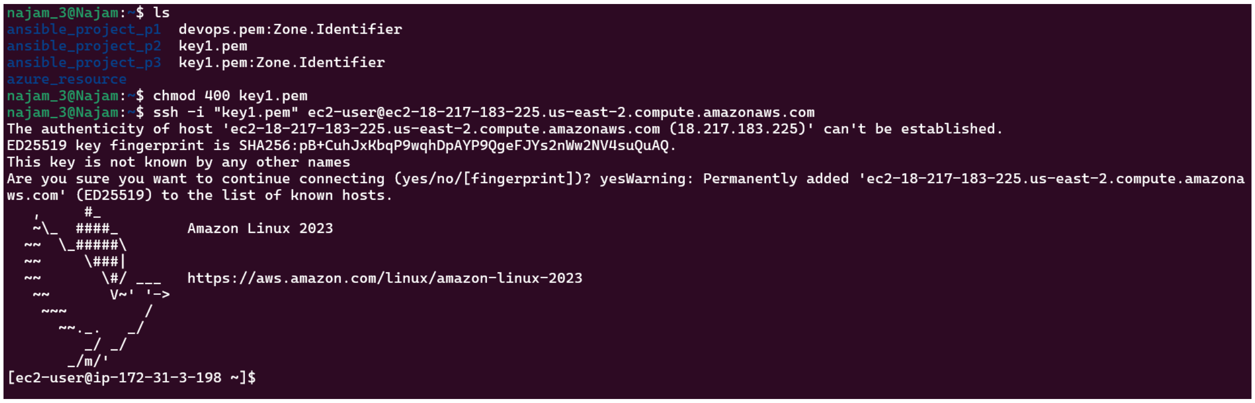
1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is key1.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   chmod 400 "key1.pem"
4. Connect to your instance using its Public DNS:
   ec2-18-217-183-225.us-east-2.compute.amazonaws.com

Example:

ssh -i "key1.pem" ec2-user@ec2-18-217-183-225.us-east-2.compute.amazonaws.com

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

Step 3) On ubuntu terminal run following commands



```
najam_3@Najam:~$ ls
ansible_project_p1  devops.pem:Zone.Identifier
ansible_project_p2  key1.pem
ansible_project_p3  key1.pem:Zone.Identifier
azure_resource
najam_3@Najam:~$ chmod 400 key1.pem
najam_3@Najam:~$ ssh -i "key1.pem" ec2-user@ec2-18-217-183-225.us-east-2.compute.amazonaws.com
The authenticity of host 'ec2-18-217-183-225.us-east-2.compute.amazonaws.com (18.217.183.225)' can't be established.
ED25519 key fingerprint is SHA256:pB+CuhJxKbqP9wqhDpAYP9QgeFJYs2nWw2NV4suQuAQ.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yesWarning: Permanently added 'ec2-18-217-183-225.us-east-2.compute.amazona
ws.com' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-3-198 ~]$
```

Install AWS CLI on EC2 server

Run following command to install/update aws cli on EC2 machine.

***By default any installation is placed in home/tmp**

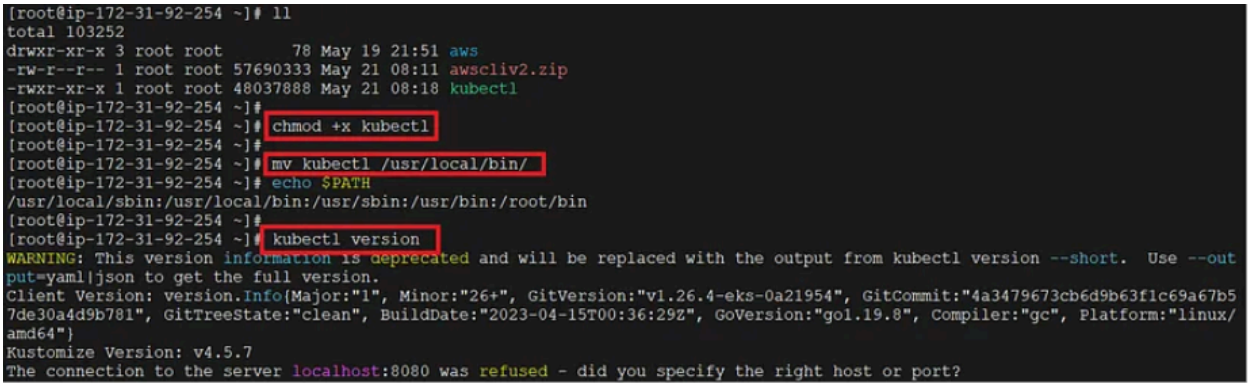
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install

Setting up Kubectl

Step 1) First download kubectl using command below

```
curl -O
https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.4/2023-05-11/bin/linux/amd64/kubectl
```

Step 2) Grant execution permission and move to kubectl to /usr/local/bin directory



```
[root@ip-172-31-92-254 ~]# ll
total 103252
drwxr-xr-x 3 root root       78 May 19 21:51 aws
-rw-r--r-- 1 root root 57690333 May 21 08:11 awscliv2.zip
-rwxr-xr-x 1 root root 48037888 May 21 08:18 kubectl
[root@ip-172-31-92-254 ~]#
[root@ip-172-31-92-254 ~]# chmod +x kubectl
[root@ip-172-31-92-254 ~]#
[root@ip-172-31-92-254 ~]# mv kubectl /usr/local/bin/
[root@ip-172-31-92-254 ~]# echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin
[root@ip-172-31-92-254 ~]#
[root@ip-172-31-92-254 ~]# kubectl version
WARNING: This version information is deprecated and will be replaced with the output from kubectl version --short.  Use --out
put=yaml|json to get the full version.
Client Version: version.Info{Major:"1", Minor:"26+", GitVersion:"v1.26.4-eks-0a21954", GitCommit:"4a3479673cb6d9b63f1c69a67b5
7de30a4d9b781", GitTreeState:"clean", BuildDate:"2023-04-15T00:36:29Z", GoVersion:"go1.19.8", Compiler:"gc", Platform:"linux/
amd64"}
Kustomize Version: v4.5.7
The connection to the server localhost:8080 was refused - did you specify the right host or port?
```

Setup eksctl

Step 1) Download latest release

```
curl --silent --location
"https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C /tmp
```

Step 2) Move extracted binary to /usr/local/bin

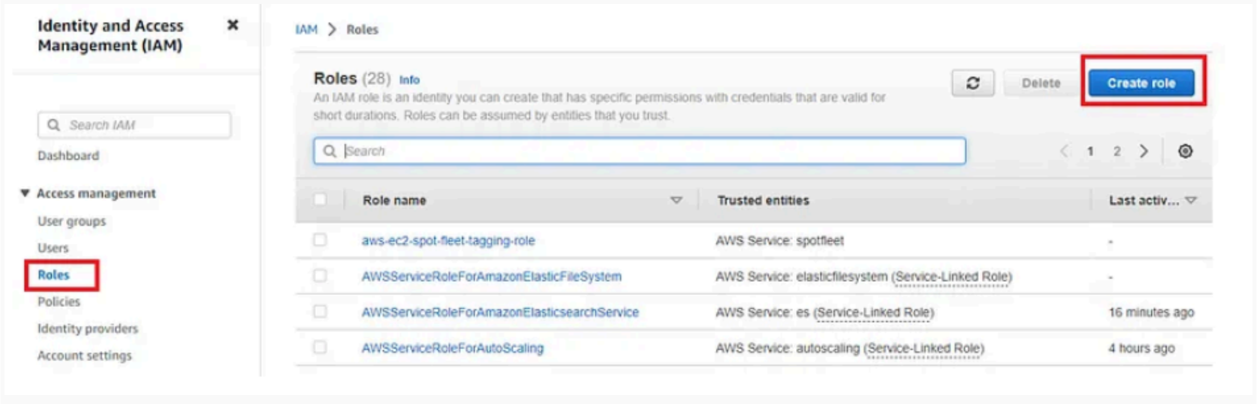
```
sudo mv /tmp/eksctl /usr/local/bin
```

Step 3) Get version

```
eksctl version
```

IAM role

Step 1) Create IAM role, go to Access Management-> roles -> create role

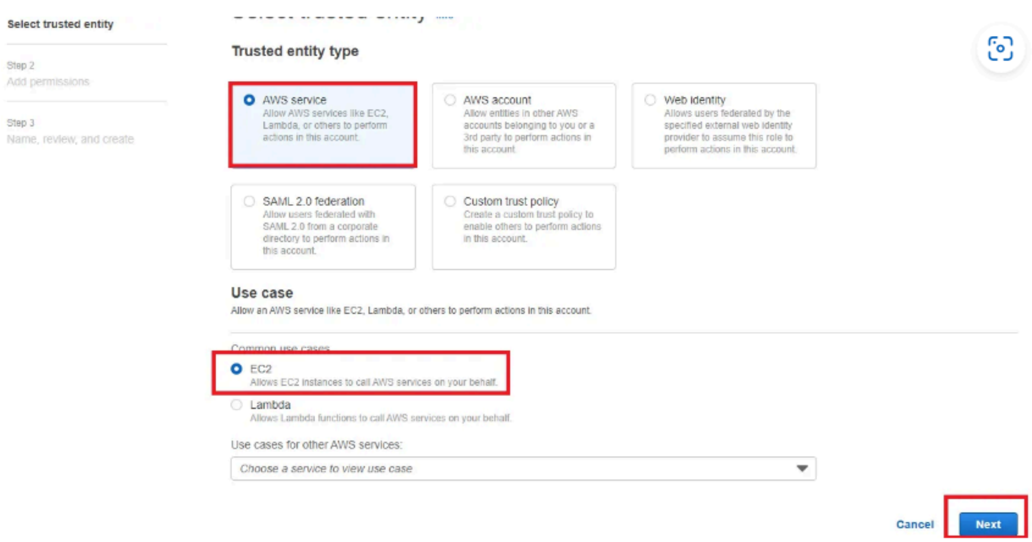


Roles (28) Info

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Role name	Trusted entities	Last activ...
aws-ec2-spot-fleet-tagging-role	AWS Service: spotfleet	-
AWSServiceRoleForAmazonElasticFileSystem	AWS Service: elasticfilesystem (Service-Linked Role)	-
AWSServiceRoleForAmazonElasticsearchService	AWS Service: es (Service-Linked Role)	16 minutes ago
AWSServiceRoleForAutoScaling	AWS Service: autoscaling (Service-Linked Role)	4 hours ago

Step 2) select the EC2 service and click next



Trusted entity type

● AWS service
Allow AWS services like EC2, Lambda, or others to perform actions in this account.

○ AWS account
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

○ Web identity
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.

○ SAML 2.0 federation
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.

○ Custom trust policy
Create a custom trust policy to enable others to perform actions in this account.

Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Common use cases

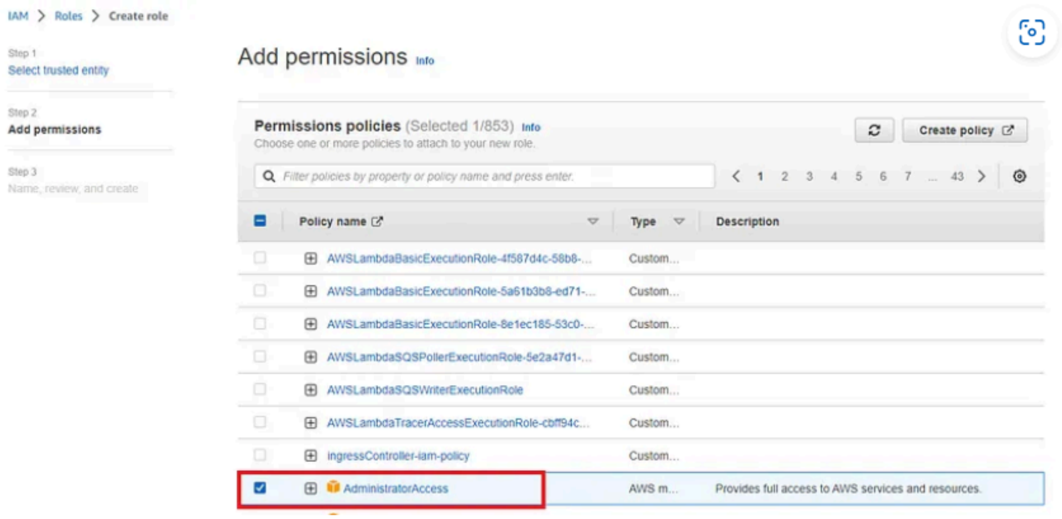
● EC2
Allows EC2 instances to call AWS services on your behalf.

○ Lambda
Allows Lambda functions to call AWS services on your behalf.

Use cases for other AWS services:

Choose a service to view use case

Step 3) Provide full administrative access to user



IAM > Roles > Create role

Step 1
Select trusted entity

Step 2
Add permissions

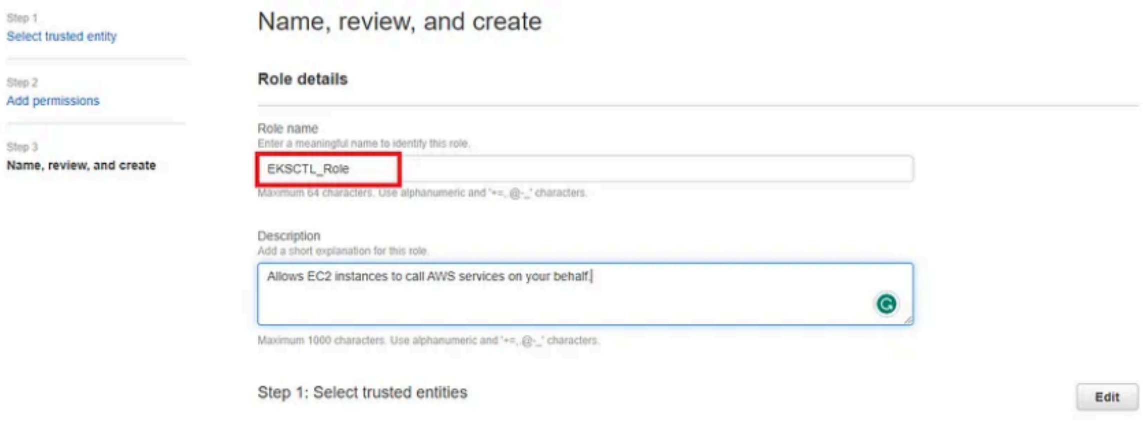
Step 3
Name, review, and create

Add permissions Info

Permissions policies (Selected 1/853) Info
Choose one or more policies to attach to your new role.

	Policy name	Type	Description
☐	AWSLambdaBasicExecutionRole-4f587d4c-58b8-...	Custom...	
☐	AWSLambdaBasicExecutionRole-5a61b3b8-ed71-...	Custom...	
☐	AWSLambdaBasicExecutionRole-8e1ec185-53c0-...	Custom...	
☐	AWSLambdaSQSPollerExecutionRole-5e2a47d1-...	Custom...	
☐	AWSLambdaSQSWriterExecutionRole	Custom...	
☐	AWSLambdaTracerAccessExecutionRole-cbff94c...	Custom...	
☐	ingressController-iam-policy	Custom...	
☑	AdministratorAccess	AWS m...	Provides full access to AWS services and resources.

Step 4) Enter name of role and create the role



Step 1
Select trusted entity

Step 2
Add permissions

Step 3
Name, review, and create

Name, review, and create

Role details

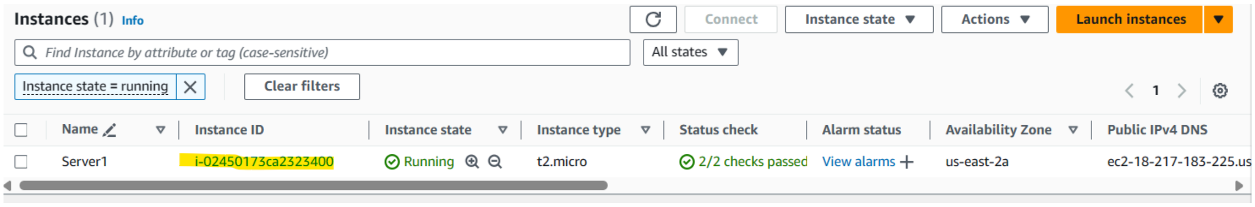
Role name
Enter a meaningful name to identify this role.
EKSCTL_Role
Maximum 64 characters. Use alphanumeric and '+=,.@-_' characters.

Description
Add a short explanation for this role.
Allows EC2 instances to call AWS services on your behalf.
Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.

Step 1: Select trusted entities

Step 5) Add role to EC2 instance

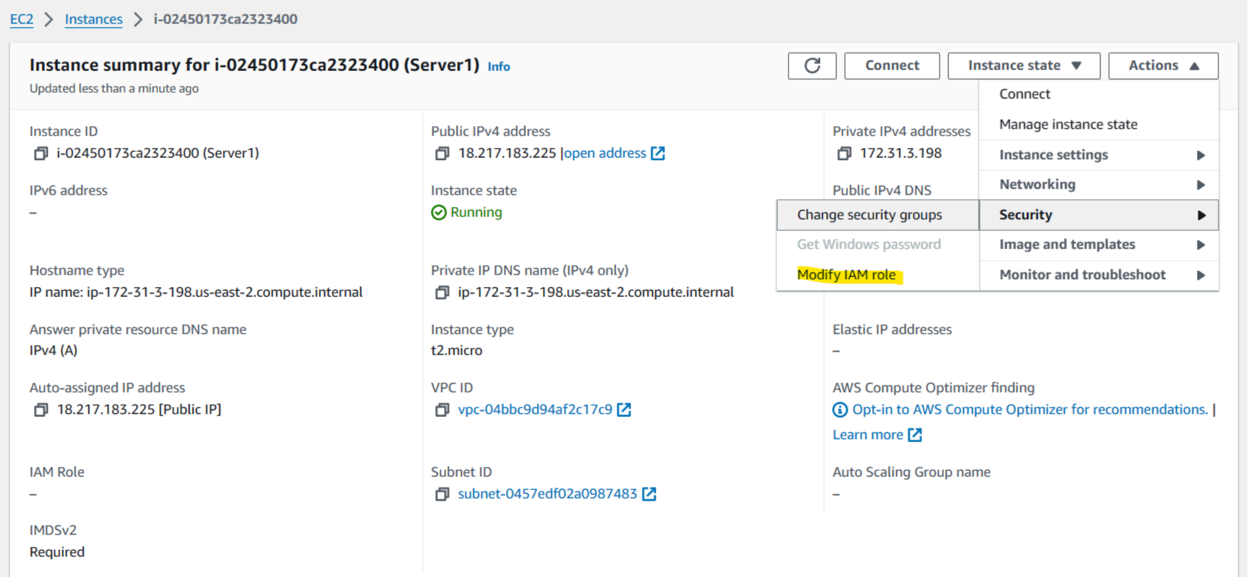
Click on instance ID



Instances (1) Info

☐	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
☐	Server1	i-02450173ca2323400	Running	t2.micro	2/2 checks passed	View alarms +	us-east-2a	ec2-18-217-183-225.us

Click on actions-> security-> modify IAM roles



Instance summary for i-02450173ca2323400 (Server1) Info
Updated less than a minute ago

Instance ID
i-02450173ca2323400 (Server1)

Public IPv4 address
18.217.183.225 |open address

Private IPv4 addresses
172.31.3.198

IPv6 address
–

Instance state
Running

Public IPv4 DNS

Hostname type
IP name: ip-172-31-3-198.us-east-2.compute.internal

Private IP DNS name (IPv4 only)
ip-172-31-3-198.us-east-2.compute.internal

Answer private resource DNS name
IPv4 (A)

Instance type
t2.micro

Elastic IP addresses
–

Auto-assigned IP address
18.217.183.225 [Public IP]

VPC ID
vpc-04bbc9d94af2c17c9

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

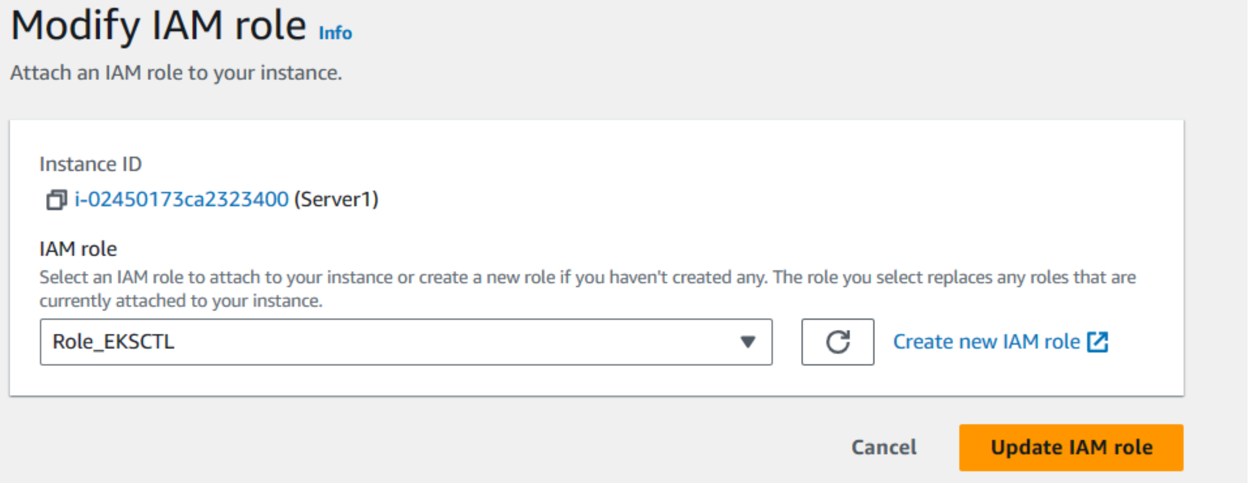
IAM Role
–

Subnet ID
subnet-0457edf02a0987483

Auto Scaling Group name
–

IMDSv2
Required

Select role and click on “Update IAM role”



Modify IAM role Info

Attach an IAM role to your instance.

Instance ID
i-02450173ca2323400 (Server1)

IAM role
Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.

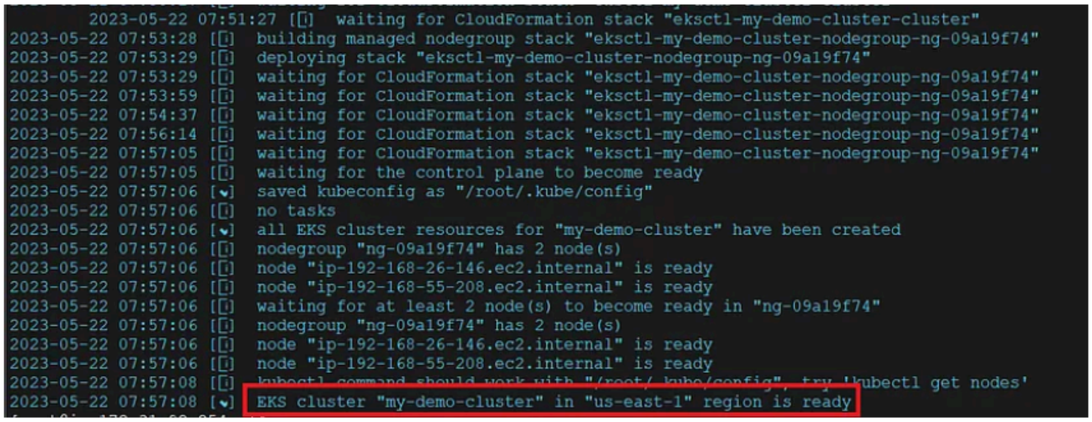
Role_EKSCTL ▼

Create cluster and nodes

Step 1) Create cluster using eksclt

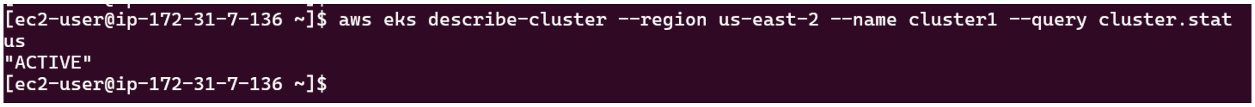
```
eksctl create cluster --name my-demo-cluster \
  --region us-east-1 \
--node-type t2.small \
```

*Note this will take around 10-20 mins



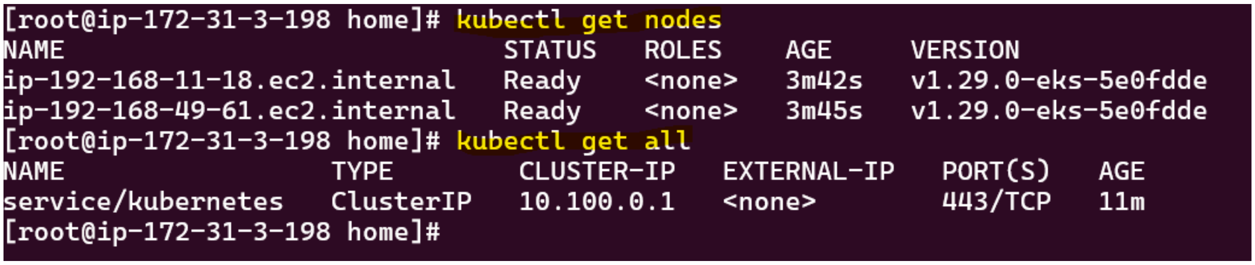
```
        2023-05-22 07:51:27 [ℹ]  waiting for CloudFormation stack "eksctl-my-demo-cluster-cluster"
2023-05-22 07:53:28 [ℹ]  building managed nodegroup stack "eksctl-my-demo-cluster-nodegroup-ng-09a19f74"
2023-05-22 07:53:29 [ℹ]  deploying stack "eksctl-my-demo-cluster-nodegroup-ng-09a19f74"
2023-05-22 07:53:29 [ℹ]  waiting for CloudFormation stack "eksctl-my-demo-cluster-nodegroup-ng-09a19f74"
2023-05-22 07:53:59 [ℹ]  waiting for CloudFormation stack "eksctl-my-demo-cluster-nodegroup-ng-09a19f74"
2023-05-22 07:54:37 [ℹ]  waiting for CloudFormation stack "eksctl-my-demo-cluster-nodegroup-ng-09a19f74"
2023-05-22 07:56:14 [ℹ]  waiting for CloudFormation stack "eksctl-my-demo-cluster-nodegroup-ng-09a19f74"
2023-05-22 07:57:05 [ℹ]  waiting for CloudFormation stack "eksctl-my-demo-cluster-nodegroup-ng-09a19f74"
2023-05-22 07:57:05 [ℹ]  waiting for the control plane to become ready
2023-05-22 07:57:06 [✔]  saved kubeconfig as "/root/.kube/config"
2023-05-22 07:57:06 [ℹ]  no tasks
2023-05-22 07:57:06 [✔]  all EKS cluster resources for "my-demo-cluster" have been created
2023-05-22 07:57:06 [ℹ]  nodegroup "ng-09a19f74" has 2 node(s)
2023-05-22 07:57:06 [ℹ]  node "ip-192-168-26-146.ec2.internal" is ready
2023-05-22 07:57:06 [ℹ]  node "ip-192-168-55-208.ec2.internal" is ready
2023-05-22 07:57:06 [ℹ]  waiting for at least 2 node(s) to become ready in "ng-09a19f74"
2023-05-22 07:57:06 [ℹ]  nodegroup "ng-09a19f74" has 2 node(s)
2023-05-22 07:57:06 [ℹ]  node "ip-192-168-26-146.ec2.internal" is ready
2023-05-22 07:57:06 [ℹ]  node "ip-192-168-55-208.ec2.internal" is ready
2023-05-22 07:57:08 [ℹ]  kubectl command should work with "/root/.kube/config", try 'kubectl get nodes'
2023-05-22 07:57:08 [✔]  EKS cluster "my-demo-cluster" in "us-east-1" region is ready
```

Run command to check status of cluster: aws eks describe-cluster --region us-east-2 --name cluster1 --query cluster.status



```
[ec2-user@ip-172-31-7-136 ~]$ aws eks describe-cluster --region us-east-2 --name cluster1 --query cluster.stat
us
"ACTIVE"
[ec2-user@ip-172-31-7-136 ~]$
```

Step 2) Verify using Kubectl commands



```
[root@ip-172-31-3-198 home]# kubectl get nodes
NAME                           STATUS   ROLES    AGE    VERSION
ip-192-168-11-18.ec2.internal  Ready    <none>   3m42s  v1.29.0-eks-5e0fdde
ip-192-168-49-61.ec2.internal  Ready    <none>   3m45s  v1.29.0-eks-5e0fdde
[root@ip-172-31-3-198 home]# kubectl get all
NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   11m
[root@ip-172-31-3-198 home]#
```

Setup Kubernetes Cluster on Amazon EKS | by Mudasir | Medium

Downloading and installing Jenkins

Step 1) Ensure software packages are up to date

sudo yum update

Step 2) Add jenkins repo using following command

sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo

Step 3) Import a key file from Jenkins-CI to enable installation from the package:

```
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
```

```
sudo yum upgrade
```

Step 4) Install Java

```
sudo dnf install java-17-amazon-corretto -y
```

Step 5) Install Jenkins

```
sudo yum install jenkins -y
```

Step 6) Enable the Jenkins service to start at boot:

```
sudo systemctl enable jenkins
```

Step 7) Start Jenkins as a service:

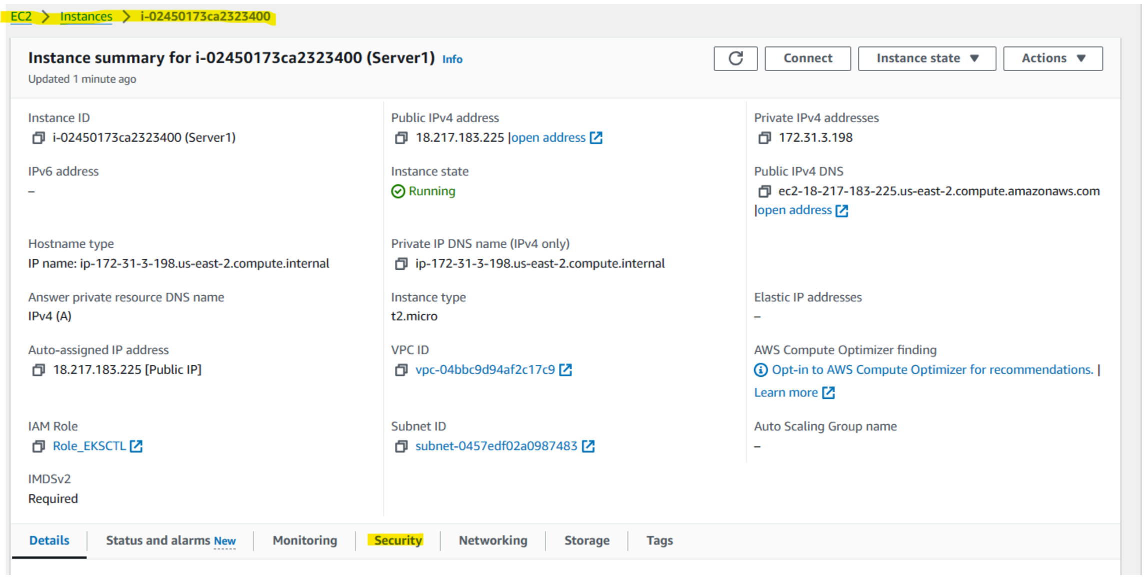
```
sudo systemctl start jenkins
```

Step 8) Check status of Jenkins

```
sudo systemctl status jenkins
```

Configuring Jenkins

Step 1) Go to AWS Management Console -> instance -> EC2 and click on security



Instance summary for i-02450173ca2323400 (Server1) Info

Updated 1 minute ago

Instance ID
i-02450173ca2323400 (Server1)

IPv6 address
–

Hostname type
IP name: ip-172-31-3-198.us-east-2.compute.internal

Answer private resource DNS name
IPv4 (A)

Auto-assigned IP address
18.217.183.225 [Public IP]

IAM Role
Role_EKSCTL

IMDSv2
Required

Public IPv4 address
18.217.183.225 |open address

Instance state
Running

Private IP DNS name (IPv4 only)
ip-172-31-3-198.us-east-2.compute.internal

Instance type
t2.micro

VPC ID
vpc-04bbc9d94af2c17c9

Subnet ID
subnet-0457edf02a0987483

Private IPv4 addresses
172.31.3.198

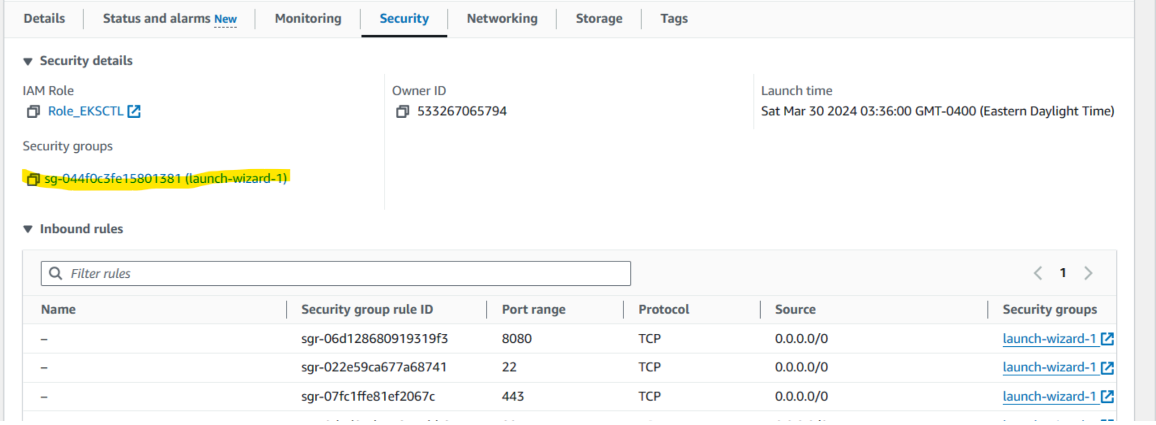
Public IPv4 DNS
ec2-18-217-183-225.us-east-2.compute.amazonaws.com
|open address

Elastic IP addresses
–

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. |
Learn more

Auto Scaling Group name
–

Step 2) Click on Security Groups



▼ Security details

IAM Role
Role_EKSCTL

Owner ID
533267065794

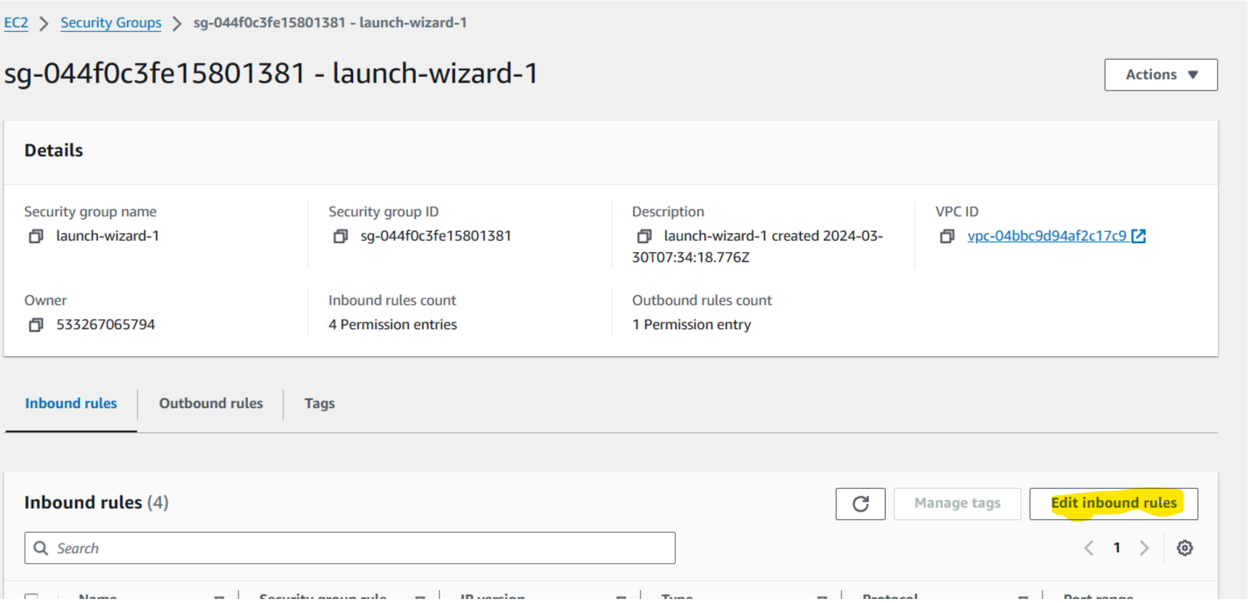
Launch time
Sat Mar 30 2024 03:36:00 GMT-0400 (Eastern Daylight Time)

Security groups
sg-044f0c3fe15801381 (launch-wizard-1)

▼ Inbound rules

Name	Security group rule ID	Port range	Protocol	Source	Security groups
–	sgr-06d128680919319f3	8080	TCP	0.0.0.0/0	launch-wizard-1
–	sgr-022e59ca677a68741	22	TCP	0.0.0.0/0	launch-wizard-1
–	sgr-07fc1ffe81ef2067c	443	TCP	0.0.0.0/0	launch-wizard-1

Step 3) Click on Edit inbound rules


EC2 > Security Groups > sg-044f0c3fe15801381 - launch-wizard-1

sg-044f0c3fe15801381 - launch-wizard-1

Details

Security group name
launch-wizard-1

Security group ID
sg-044f0c3fe15801381

Description
launch-wizard-1 created 2024-03-30T07:34:18.776Z

VPC ID
vpc-04bbc9d94af2c17c9

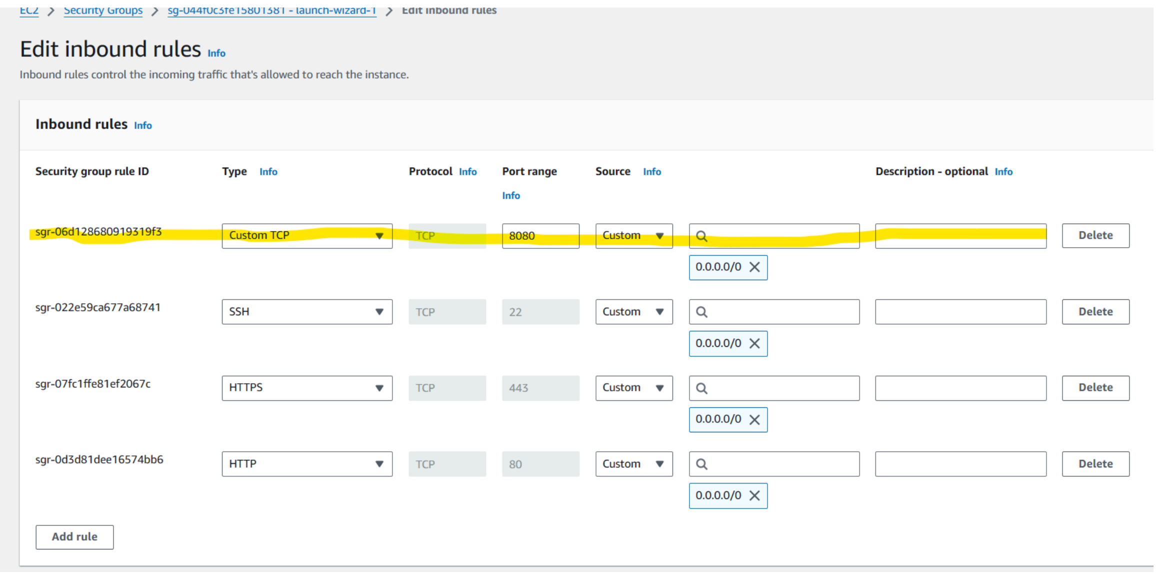
Owner
533267065794

Inbound rules count
4 Permission entries

Outbound rules count
1 Permission entry

Inbound rules (4)

Step 4) Add rule 8080 for jenkins and then click save



EC2 > Security Groups > sg-044f0c3fe15801381 - launch-wizard-1 > Edit inbound rules

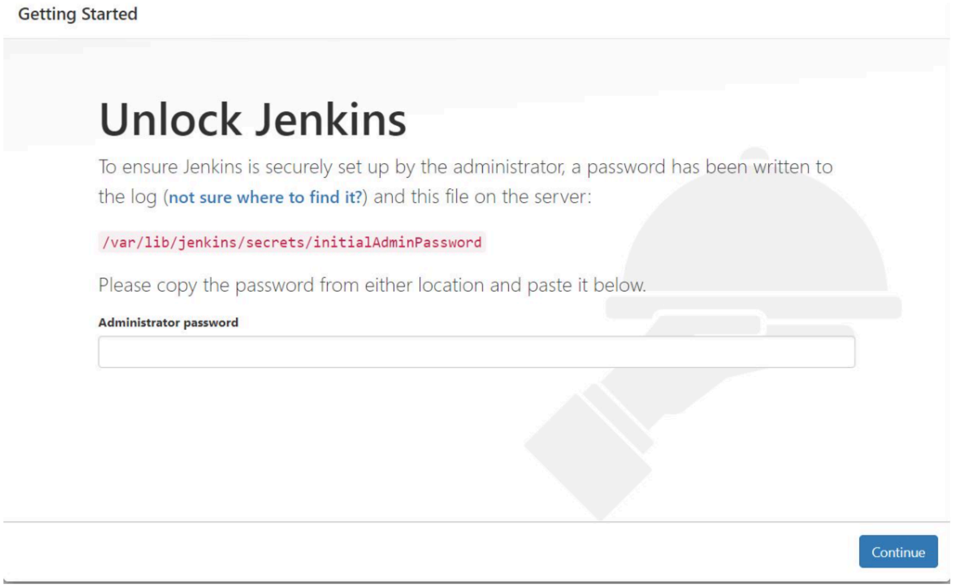
Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-06d128680919319f3	Custom TCP ▼	TCP	8080	Custom ▼	🔍 0.0.0.0/0 ✕		Delete
sgr-022e59ca677a68741	SSH ▼	TCP	22	Custom ▼	🔍 0.0.0.0/0 ✕		Delete
sgr-07fc1ffe81ef2067c	HTTPS ▼	TCP	443	Custom ▼	🔍 0.0.0.0/0 ✕		Delete
sgr-0d3d81dee16574bb6	HTTP ▼	TCP	80	Custom ▼	🔍 0.0.0.0/0 ✕		Delete

Connect to http://<your_server_public_DNS>:8080 from your browser. You will be able to access Jenkins through its management interface:



Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

`/var/lib/jenkins/secrets/initialAdminPassword`

Please copy the password from either location and paste it below.

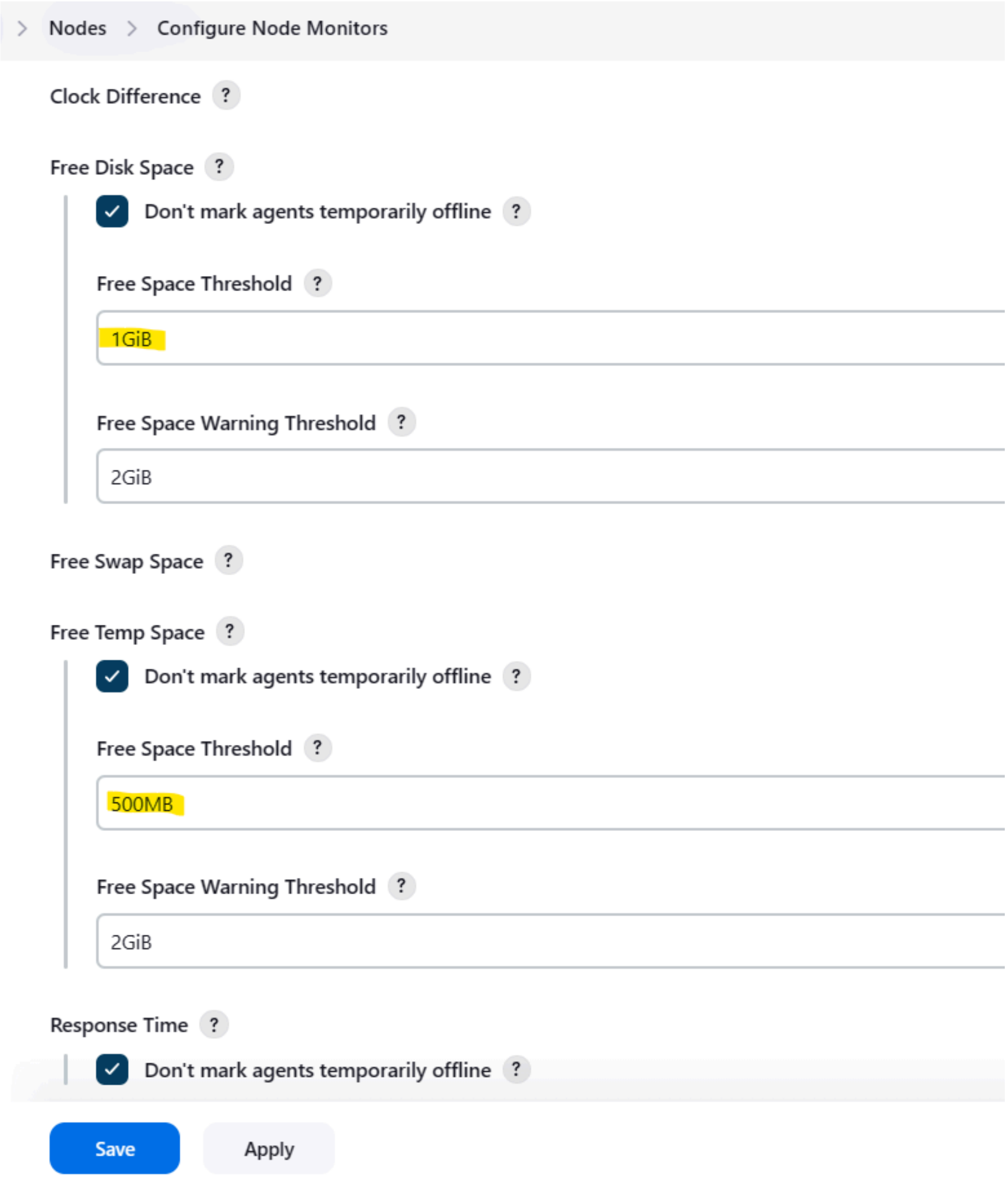
Administrator password

Step 5) The administrative password can be found in following

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

Step 6) Fix disk space too low issue

Go to Dashboard-> node -> configure monitor



Clock Difference ?

Free Disk Space ?

☑ Don't mark agents temporarily offline ?

Free Space Threshold ?

1GiB

Free Space Warning Threshold ?

2GiB

Free Swap Space ?

Free Temp Space ?

☑ Don't mark agents temporarily offline ?

Free Space Threshold ?

500MB

Free Space Warning Threshold ?

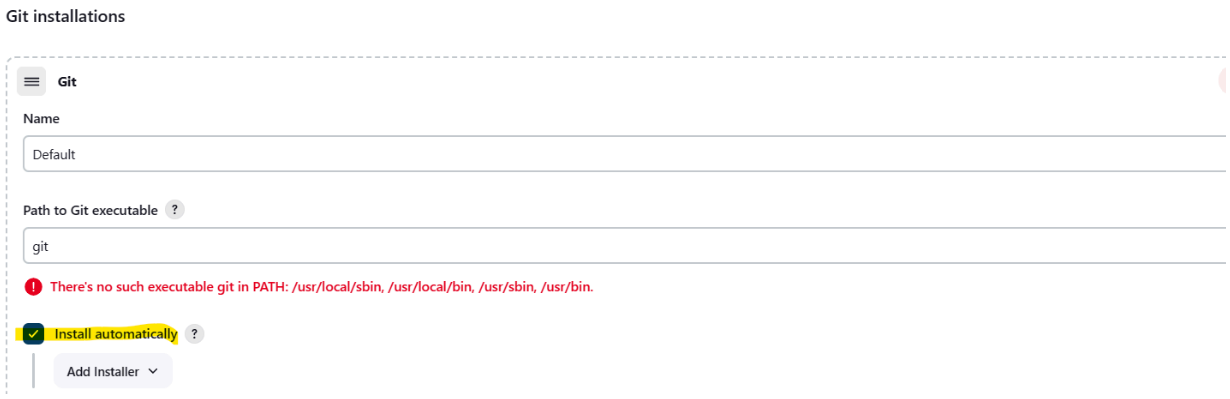
2GiB

Response Time ?

☑ Don't mark agents temporarily offline ?

Step 7) Install Git on jenkins

Go to Dashboard-> ManageJenkins -> Tools and click on install automatically for Git



Git installations

Git

Name: Default

Path to Git executable: git

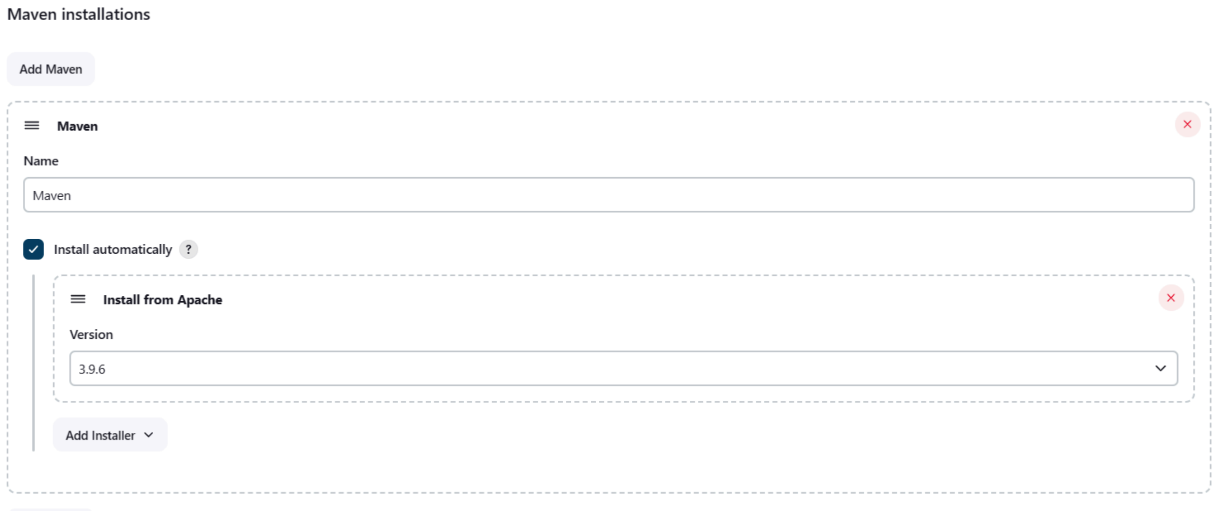
There's no such executable git in PATH: /usr/local/sbin, /usr/local/bin, /usr/sbin, /usr/bin.

☑ Install automatically

Step 8) Install Git on local ubuntu

```
sudo yum install git
```

Step 9) Install Maven on Jenkins



Maven installations

Maven

Name: Maven

☑ Install automatically

Install from Apache

Version: 3.9.6

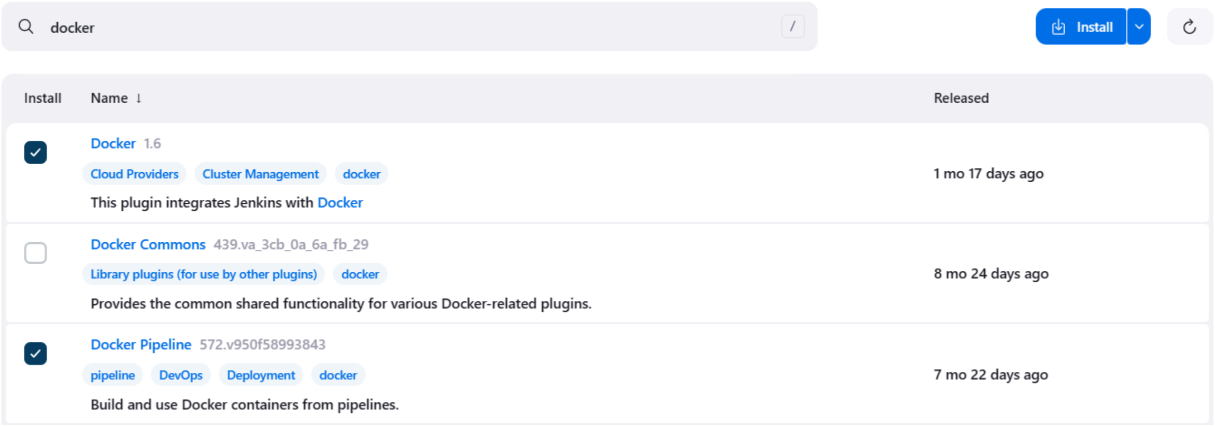
Step 10) Go to pipeline and add following script

```
 tools{
       maven 'Maven'
   }
```

Step 11) Locate where pom.xml file is and then in pipeline run following command

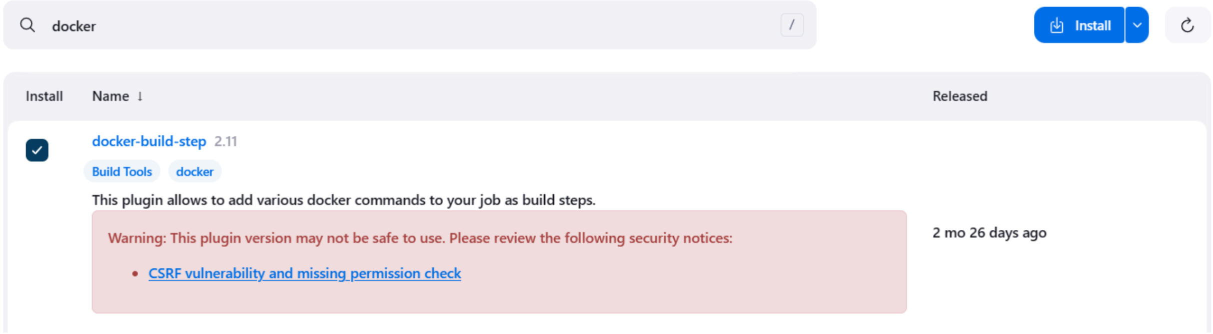
sh ‘mvn clean install’

Step 12) Go to manage Jenkins and install docker plugins



docker

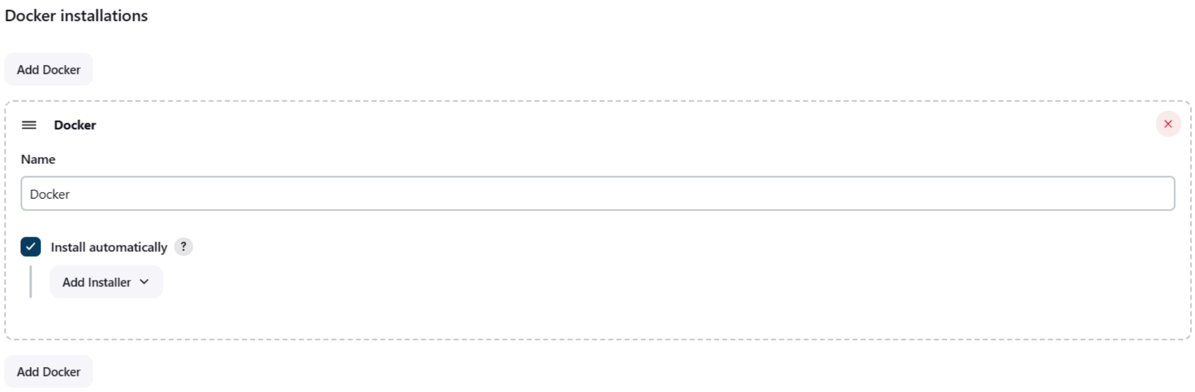
Install	Name ↓	Released
☑	Docker 1.6 Cloud Providers · Cluster Management · docker This plugin integrates Jenkins with Docker	1 mo 17 days ago
□	Docker Commons 439.va_3cb_0a_6a_fb_29 Library plugins (for use by other plugins) · docker Provides the common shared functionality for various Docker-related plugins.	8 mo 24 days ago
☑	Docker Pipeline 572.v950f58993843 pipeline · DevOps · Deployment · docker Build and use Docker containers from pipelines.	7 mo 22 days ago



docker

Install	Name ↓	Released
☑	docker-build-step 2.11 Build Tools · docker This plugin allows to add various docker commands to your job as build steps. Warning: This plugin version may not be safe to use. Please review the following security notices: • CSRF vulnerability and missing permission check	2 mo 26 days ago

Step 13) Go to manage jenkins tools and install docker



Docker installations

Add Docker

Docker

Name

Docker

☑ Install automatically ?

Add Installer

Add Docker

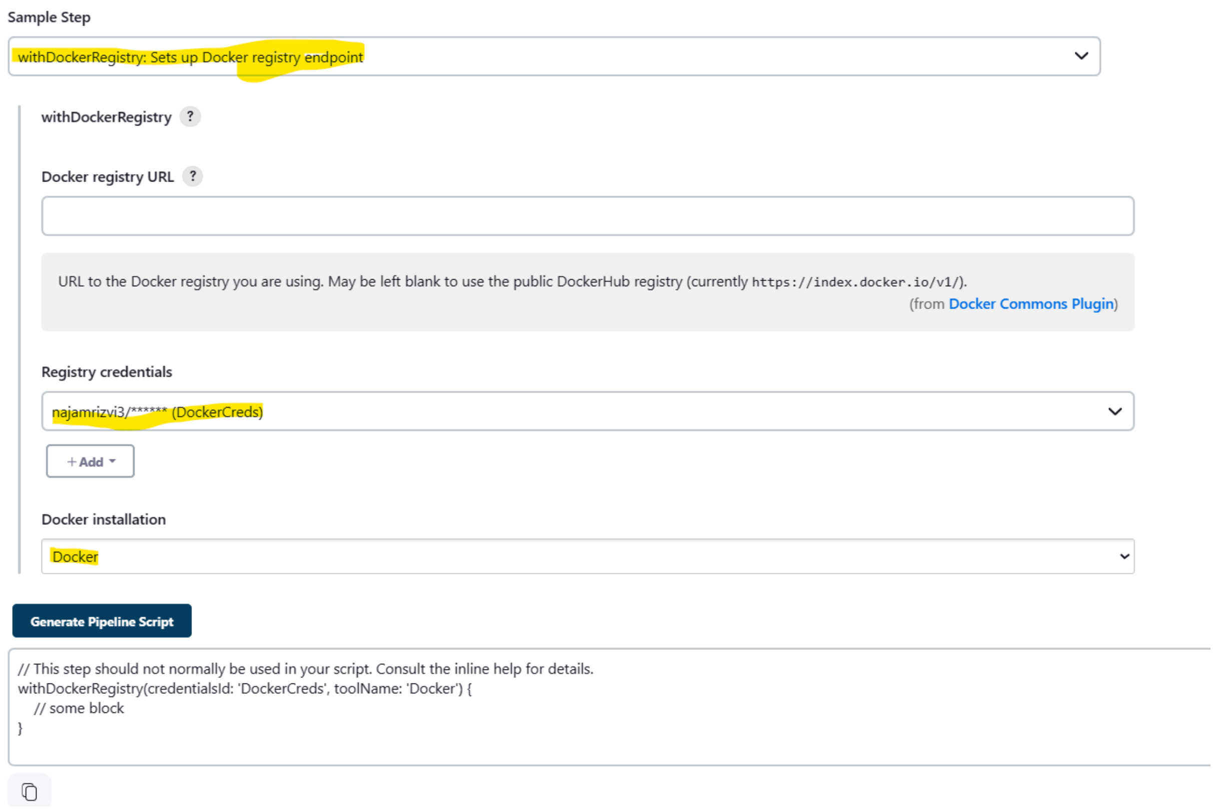
Step 14) Install docker on Ubuntu

```
sudo yum update

sudo yum install -y yum-utils device-mapper-persistent-data lvm2

sudo yum-config-manager --add-repo
https://download.docker.com/linux/centos/docker-ce.repo

sudo yum install docker

sudo systemctl start docker
sudo systemctl enable docker

usermod -a -G docker jenkins *add jenkins user to docker

sudo chmod 666 /var/run/docker.sock

docker login
```

username: najamrizvi3
Password: ChallowE123456

***Note if your jenkins freezes stop your ec2 vm and start and connect again**

Copy Generated pipeline script from below



Sample Step

withDockerRegistry: Sets up Docker registry endpoint

withDockerRegistry ?

Docker registry URL ?

URL to the Docker registry you are using. May be left blank to use the public DockerHub registry (currently `https://index.docker.io/v1/`).
(from Docker Commons Plugin)

Registry credentials

najamrizvi3/****** (DockerCreds)

Docker installation

Docker

```
// This step should not normally be used in your script. Consult the inline help for details.
withDockerRegistry(credentialsId: 'DockerCreds', toolName: 'Docker') {
    // some block
}
```

Step 15) Docker image build

```
withDockerRegistry(credentialsId: 'DockerCreds2', url: "") {
        sh 'docker build -t image03312 .'
}
```

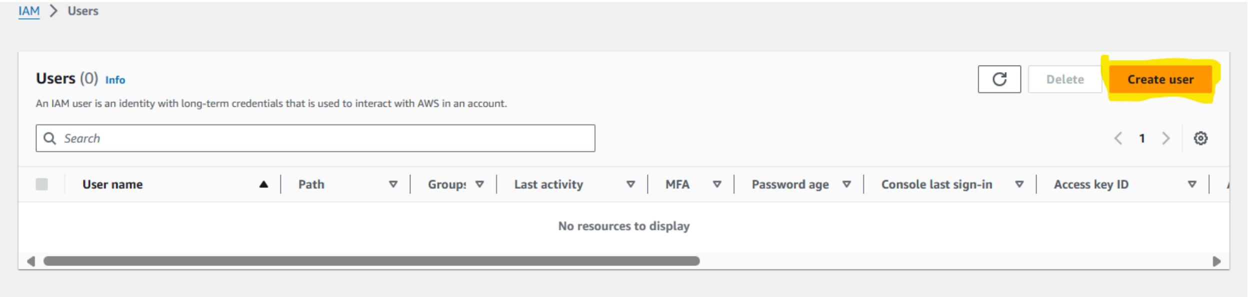
***note image03312 is image name**

Step 16) Docker tag and push

```
withDockerRegistry(credentialsId: 'DockerCreds2', url: "") {
        sh 'docker tag image03312 najamrizvi3/projc:latest'
        sh 'docker push najamrizvi3/projc:latest'
}
```

Step 17) Create user and Assign Jason policy to it

Go to IAM -> users -> createuser



IAM > Users

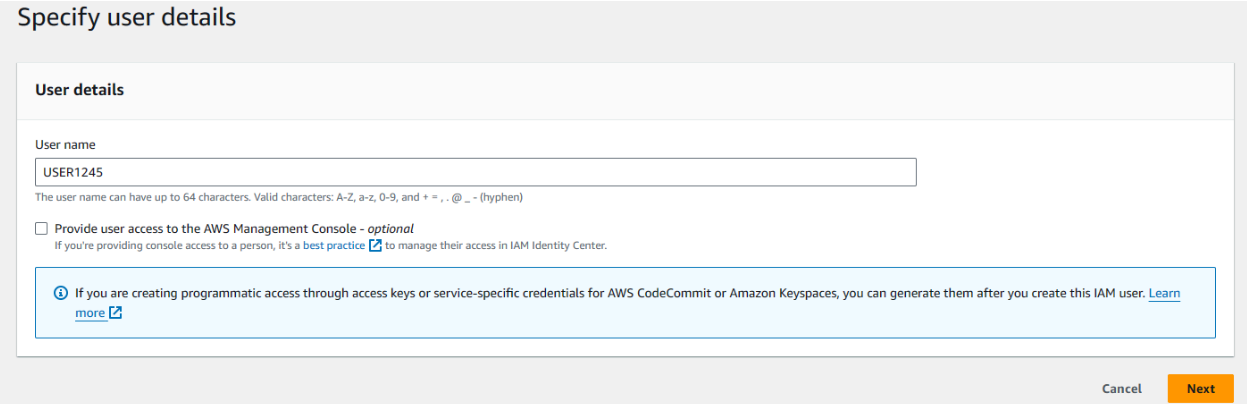
Users (0) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

	User name	Path	Group:	Last activity	MFA	Password age	Console last sign-in	Access key ID

No resources to display

Enter username and click next



Specify user details

User details

User name

USER1245

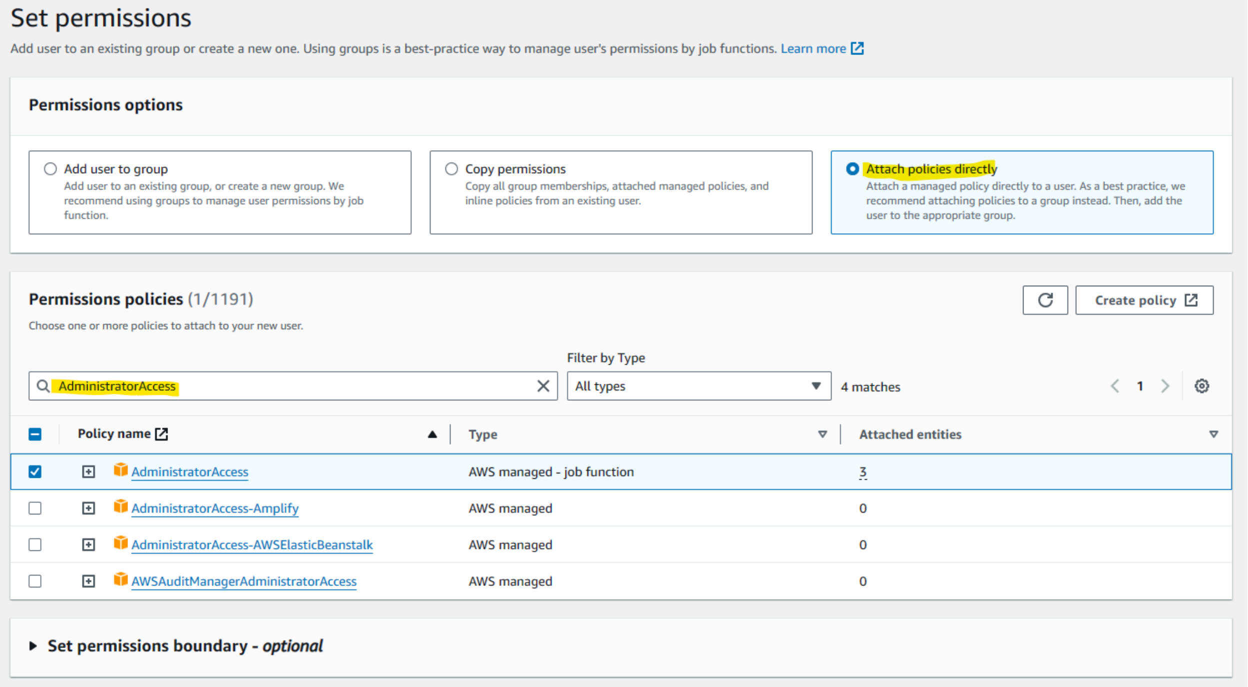
The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

□ Provide user access to the AWS Management Console - *optional*

If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.

If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. Learn more

Go to Attach policies directly-> AdministratorAccess



Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. Learn more

Permissions options

○ Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

○ Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.

● Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Permissions policies (1/1191)

Choose one or more policies to attach to your new user.

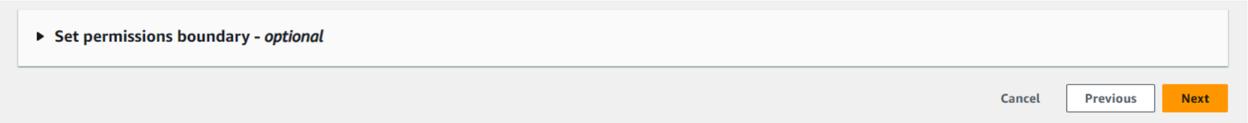
Search: AdministratorAccess

Filter by Type: All types

4 matches

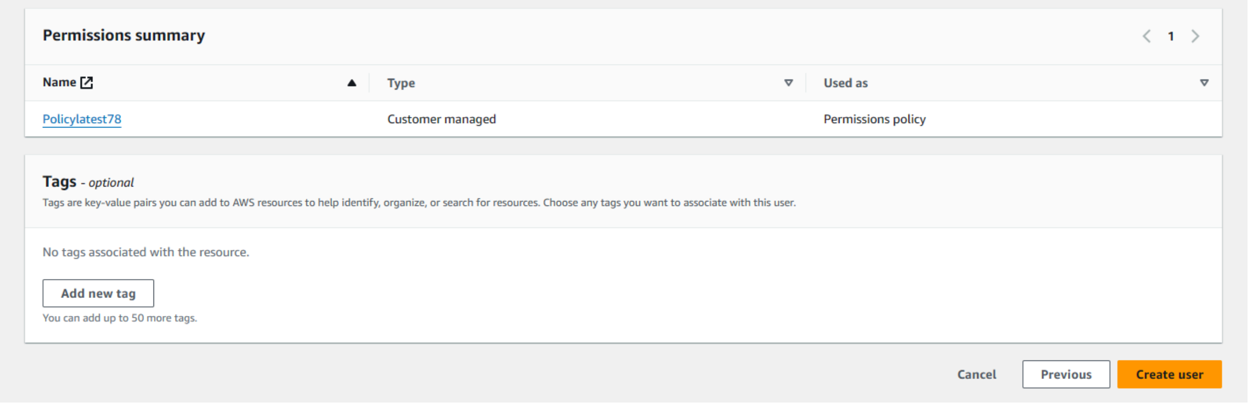
	Policy name	Type	Attached entities
☑	AdministratorAccess	AWS managed - job function	3
□	AdministratorAccess-Amplify	AWS managed	0
□	AdministratorAccess-AWSElasticBeanstalk	AWS managed	0
□	AWSAuditManagerAdministratorAccess	AWS managed	0

▸ **Set permissions boundary -** ***optional***



▸ **Set permissions boundary -** ***optional***

Click on create user



Permissions summary

Name	Type	Used as
Policylatest78	Customer managed	Permissions policy

Tags - *optional*

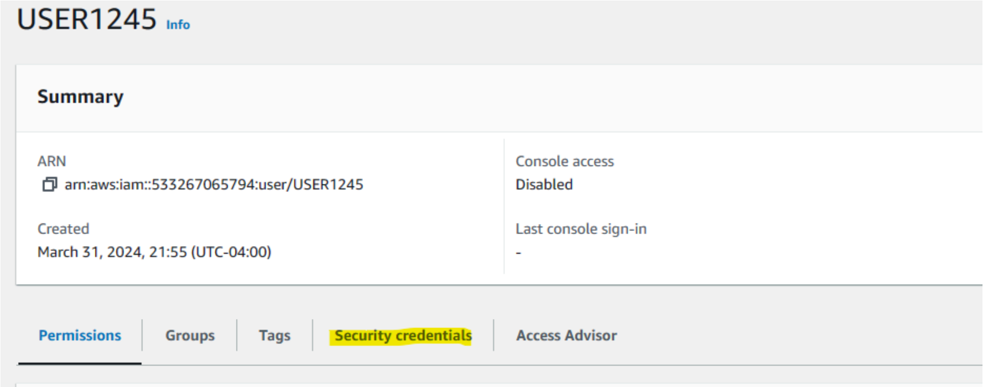
Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

You can add up to 50 more tags.

Step 18) Configuring AWS

Go to IAM -> click on user -> Security credentials



Summary

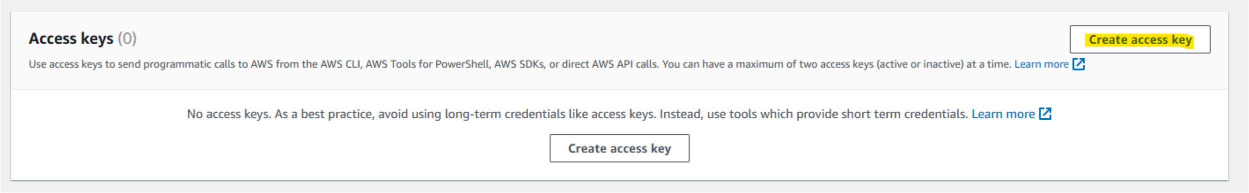
ARN
arn:aws:iam::533267065794:user/USER1245

Created
March 31, 2024, 21:55 (UTC-04:00)

Console access
Disabled

Last console sign-in
-

Create new access key

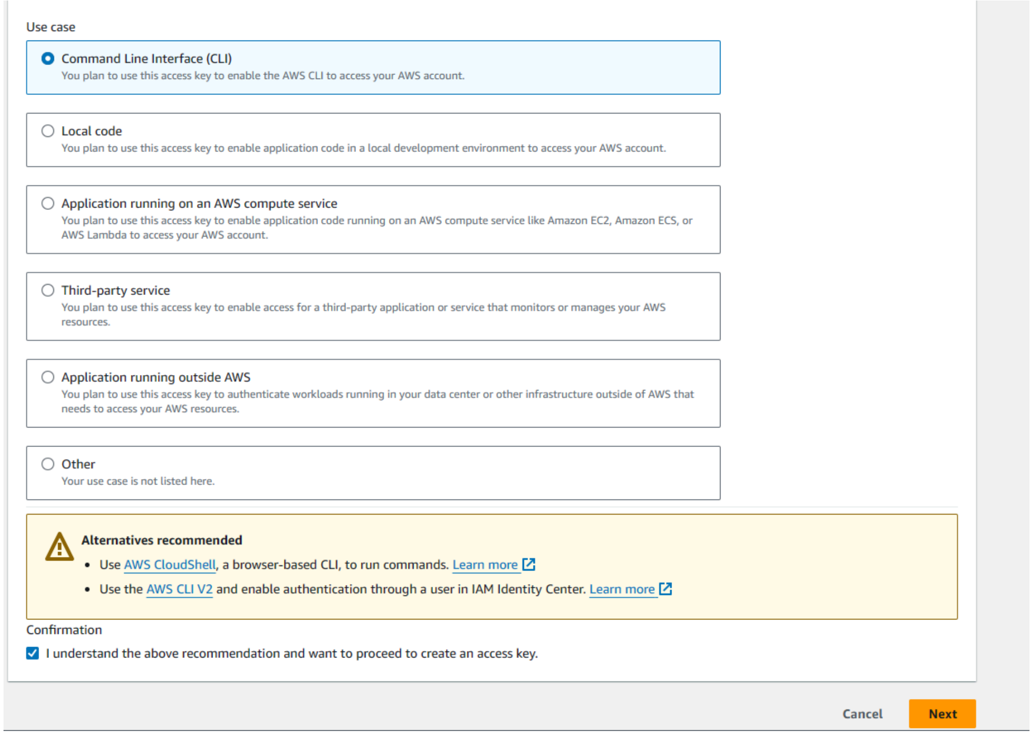


Access keys (0)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more

No access keys. As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials. Learn more

Select CLI and click next



Use case

● Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

○ Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.

○ Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

○ Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

○ Application running outside AWS
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

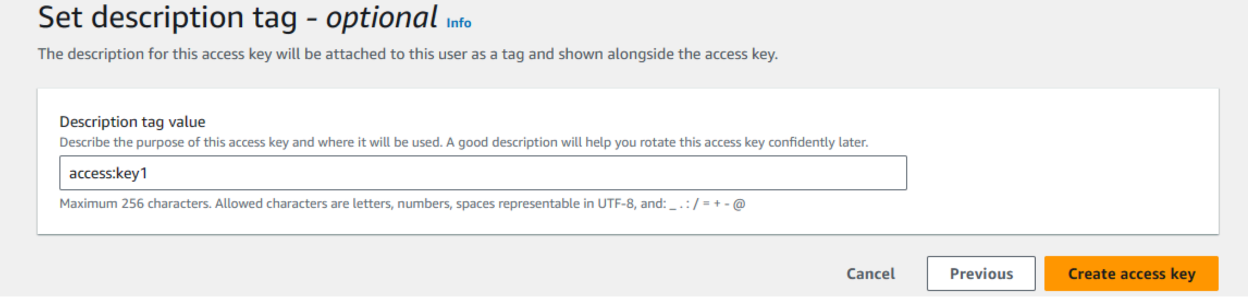
○ Other
Your use case is not listed here.

Alternatives recommended
- Use AWS CloudShell, a browser-based CLI, to run commands. Learn more
- Use the AWS CLI V2 and enable authentication through a user in IAM Identity Center. Learn more

Confirmation

☑ I understand the above recommendation and want to proceed to create an access key.

Create tags



Set description tag - *optional* Info

The description for this access key will be attached to this user as a tag and shown alongside the access key.

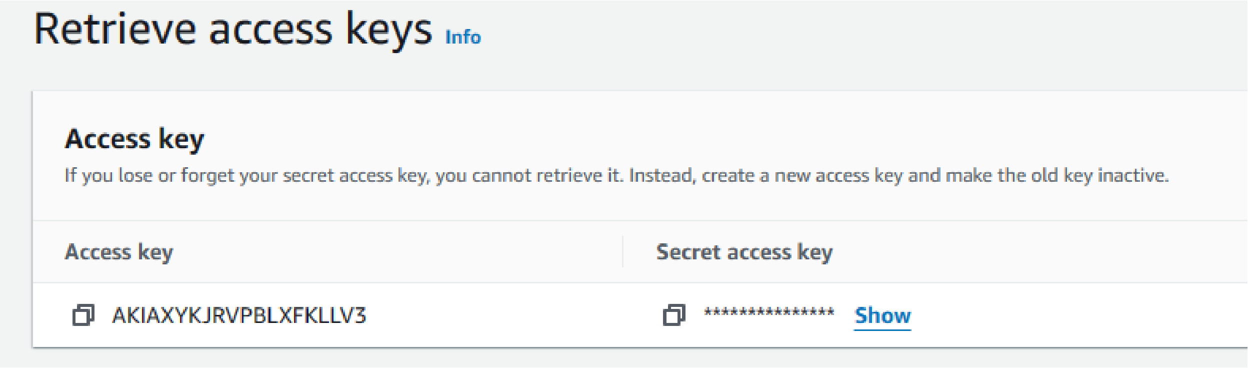
Description tag value

Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently later.

access:key1

Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: _ . : / = + - @

Access Key ID and Secret Access Key can be found below



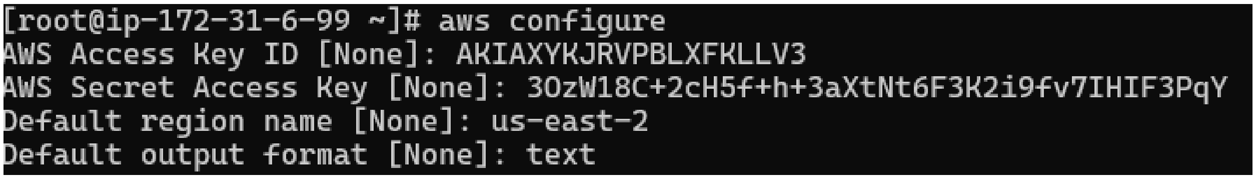
Retrieve access keys Info

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIAXYKJRVPBLXFKLLV3	*************** Show

Go to terminal type “aws configure”

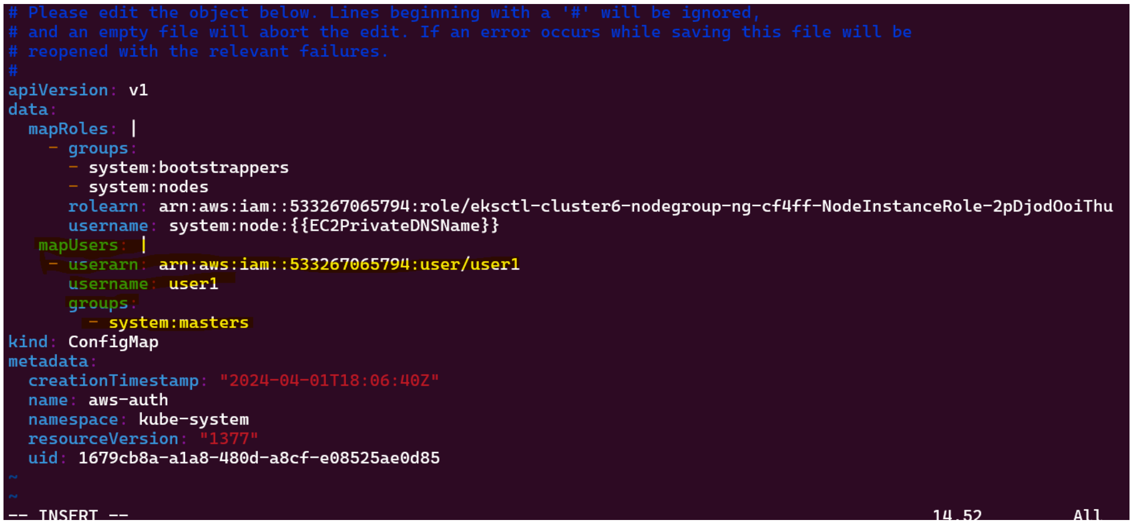


```
[root@ip-172-31-6-99 ~]# aws configure
AWS Access Key ID [None]: AKIAXYKJRVPBLXFKLLV3
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-2
Default output format [None]: text
```

Step 19) Run following command in terminal

kubectl edit configmap aws-auth -n kube-system

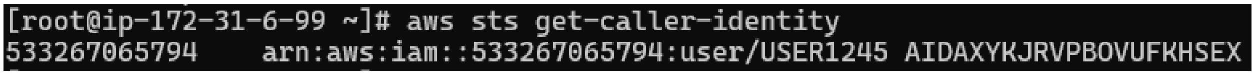
Step 20) Make the following change



```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  mapRoles: |
    - groups:
      - system:bootstrappers
      - system:nodes
      rolearn: arn:aws:iam::533267065794:role/eksctl-cluster6-nodegroup-ng-cf4ff-NodeInstanceRole-2pDjodOoiThu
      username: system:node:{{EC2PrivateDNSName}}
   mapUsers: |
    - userarn: arn:aws:iam::533267065794:user/user1
      username: user1
      groups:
        - system:masters
kind: ConfigMap
metadata:
  creationTimestamp: "2024-04-01T18:06:40Z"
  name: aws-auth
  namespace: kube-system
  resourceVersion: "1377"
  uid: 1679cb8a-a1a8-480d-a8cf-e08525ae0d85
~
~
-- INSERT --                                                                              14,52          All
```

Step 21) Check in terminal if aws access is configured using following command

aws sts get-caller-identity



```
[root@ip-172-31-6-99 ~]# aws sts get-caller-identity
533267065794    arn:aws:iam::533267065794:user/USER1245 AIDAXYKJRVPBOVUFKHSEX
```

Step 22) Update kubeconfig file

aws eks update-kubeconfig --name clusterlatest --region us-east-2

***Make sure to enter correct name of cluster and its region**

Step 23) Get the kube-config file

cat .kube/config

*Copy it and save it in text file

Step 24) Get nodes and service

kubectl get svc
kubectl get nodes

Step 24) Go to Jenkins in terminal by entering following command:

```
sudo -su jenkins
```

Step 25) Run following command

```
aws configure
```

Step 26) Now run this command

```
aws eks update-kubeconfig --name cluster6 --region us-east-2
```

*Check name of cluster and region

Step 27) Run following command now

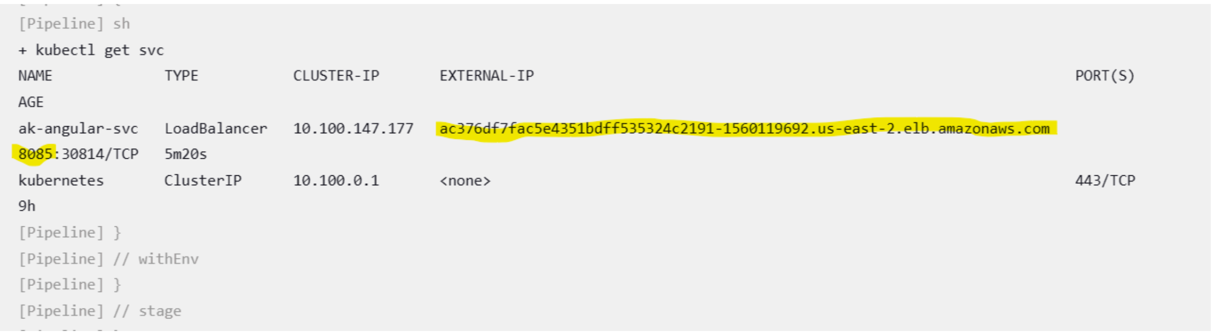
```
sh 'kubectl get nodes'
sh 'kubectl apply -f Deployment.yml'
```

Step 28) Run following command to get service

```
sh 'kubectl get svc'
```

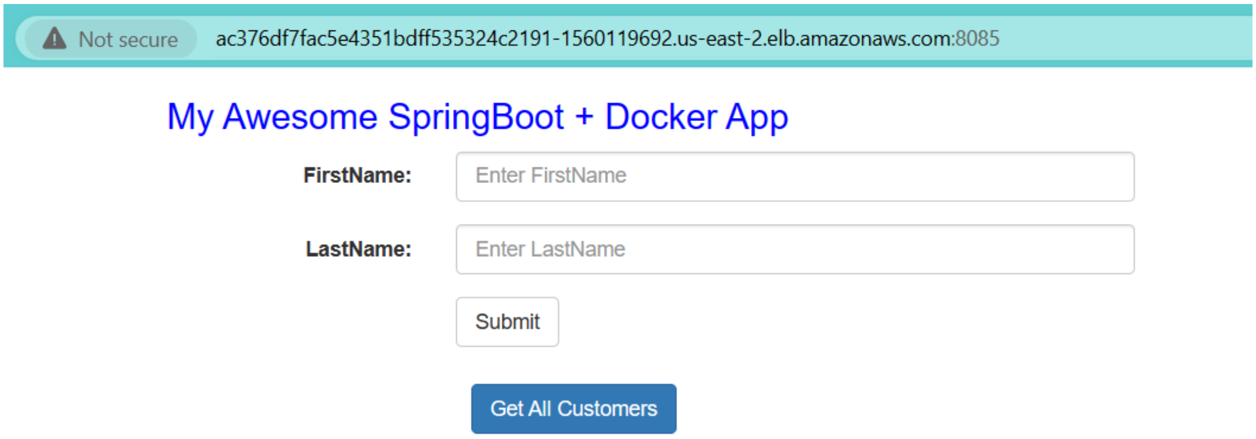
Step 29) You can access the application on following in web-browser

My Awesome Spring Boot Example (ac376df7fac5e4351bdff535324c2191-1560119692.us-east-2.elb.amazonaws.com)



```
[Pipeline] sh
+ kubectl get svc
NAME             TYPE           CLUSTER-IP       EXTERNAL-IP                                                                 PORT(S)
AGE
ak-angular-svc   LoadBalancer   10.100.147.177   ac376df7fac5e4351bdff535324c2191-1560119692.us-east-2.elb.amazonaws.com
8085:30814/TCP   5m20s
kubernetes       ClusterIP      10.100.0.1       <none>                                                                      443/TCP
9h
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
```

Step 30) Output


Not secure ac376df7fac5e4351bdff535324c2191-1560119692.us-east-2.elb.amazonaws.com:8085

My Awesome SpringBoot + Docker App

FirstName: Enter FirstName

LastName: Enter LastName

```
pipeline {
    agent any

    tools{
        maven 'Maven'
    }


    stages {
        stage('Git Checkout') {
            steps {
                git branch: 'main', url: 'https://github.com/srizvi0/Real_World_Jenkins_Proj_C.git'
            }
        }

        stage('Maven Build'){
            steps{
                sh 'mvn clean install'
            }
        }

        stage('Docker build'){
            steps{
                withDockerRegistry(credentialsId: 'DockerCreds2', url: "") {
                    sh 'docker build -t image03312 .'
                }
            }
        }

        stage ('Docker Tag & Push'){
            steps{
                withDockerRegistry(credentialsId: 'DockerCreds2', url: "") {
                    sh 'docker tag image03312 najamrizvi3/projc:latest'
                    sh 'docker push najamrizvi3/projc:latest'
                }
            }
        }
        stage ('Install Kubectl'){
            steps{
               sh 'curl -O
https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.4/2023-05-11/bin/linux/amd64/kubectl'
            }
        }
        stage ('Deploy'){
            steps{
                sh 'aws eks update-kubeconfig --name cluster6 --region us-east-2'
                sh 'kubectl get nodes'
                sh 'kubectl apply -f Deployment.yml'
            }
        }
        stage ('Get svc'){
            steps{
                sh 'kubectl get svc'
            }
        }
    }
}
```