

Jenkins Assignment

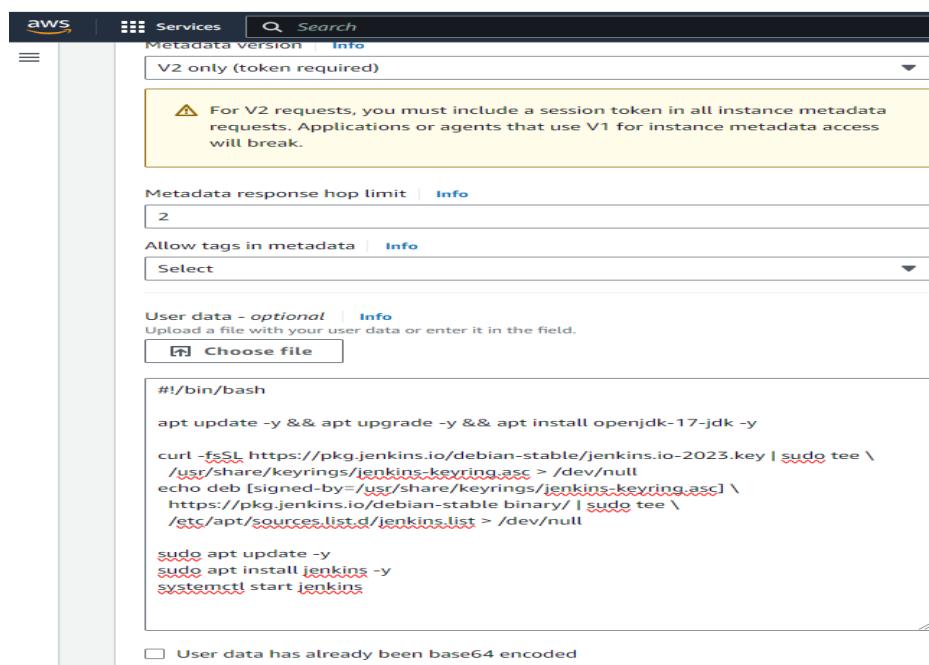
1) How to change default port of Jenkins (8080) to some other port.

Default port for Jenkins [8080]

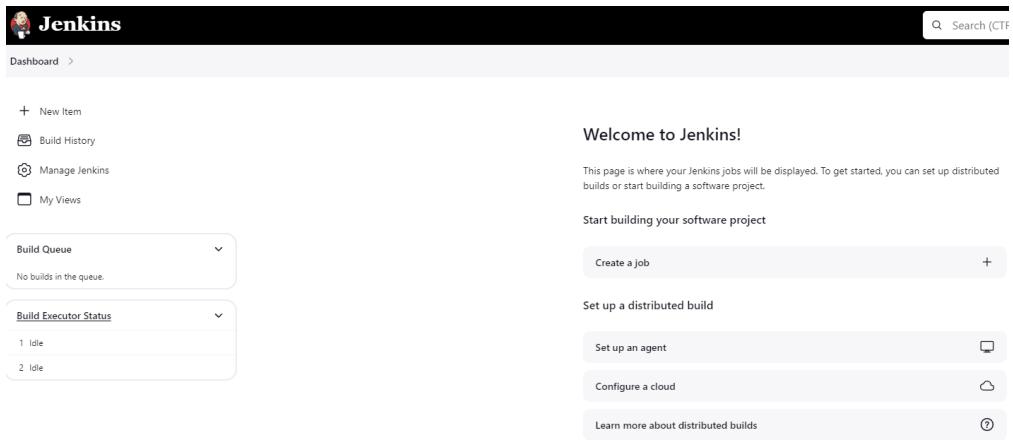
New Port for Jenkins [8081]

- To change the default port of Jenkins, we need to have a Jenkins Server EC2 instance.

Add the below commands in user data section of EC2 instance before launching it to install Jenkins on it



- Once the EC2 instance is up, Post deployment steps are performed for configuring Jenkins to setup the Jenkins server. Once Jenkins is configured, we see the below dashboard screen.



- Now we will go to the terminal and do the below changes:

vi /etc/default/jenkins (Edit the jenkins configuration file)

```
root@ip-172-31-94-44:~# vi /etc/default/jenkins
```

- Scroll down until you find the following lines:

```
# port for HTTP connector (default 8080; disable with -1)
HTTP_PORT=8080
```

```
# port for HTTP connector (default 8080; disable with -1)
HTTP_PORT=8080
```

- Edit the second line to include the port number you want to specify)

```
# port for HTTP connector (default 8080; disable with -1)
HTTP_PORT=8081
```

```
# port for HTTP connector (default 8080; disable with -1)
HTTP_PORT=8081
```

- Esc, then: wq! (Save and exit the file)
- Now we will edit the jenkins service file with new port number

vi usr/lib/systemd/system/jenkins.service (Open Jenkins Service)

```
root@ip-172-31-94-44:/#
root@ip-172-31-94-44:/# vi usr/lib/systemd/system/jenkins.service
```

- Scroll down until you find the below line:

Environment="JENKINS_PORT=8080"

```
# directive below.
Environment="JENKINS_PORT=8080"
```

- Change the port with the new port number

Environment="JENKINS_PORT=8081"

```
# directive below.
Environment="JENKINS_PORT=8081"
```

Esc, then: wq! (Save and exit the file)

- Reload the Daemon

systemctl daemon-reload

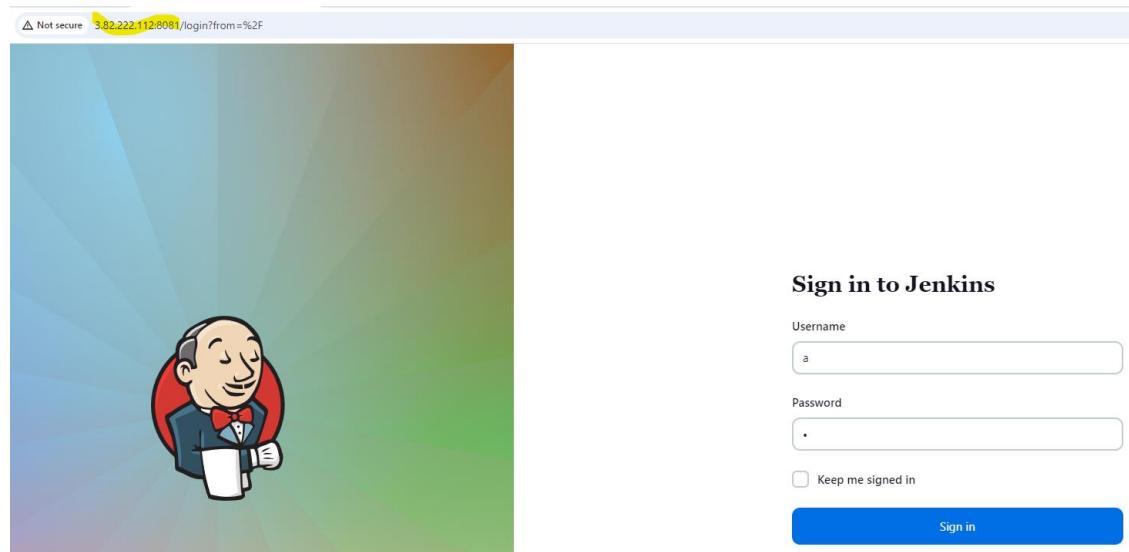
```
root@ip-172-31-94-44:/# systemctl daemon-reload
root@ip-172-31-94-44:/#
```

- Restart Jenkins Service

systemctl restart jenkins

```
root@ip-172-31-94-44:/# systemctl restart jenkins
root@ip-172-31-94-44:/#
```

- Verify on the Webpage with Public IP: New port Number



- Now to make sure all the functionalities of Jenkins are working properly.
Need make change to Jenkins URL to reflect the new port Number

Manage Jenkins > System > Jenkins Location > Jenkins URL

Before making the change.

The screenshot shows the Jenkins 'Manage Jenkins > System > Jenkins Location' interface. The 'Jenkins URL' field contains the value 'http://3.82.222.112:8080/'. The URL in the browser bar at the top is '3.82.222.112:8081/login?from=%2F'.

change from URL with old port Number to URL with new port number and hit save

Dashboard > Manage Jenkins > System >

Jenkins Location

Jenkins URL ?
http://3.82.222.112:8081/

System Admin e-mail address ?
address not configured yet <nobody>

Serve resource files from another location

Resource Root URL ?
Without a resource root URL, resources will be served from the Jenkins URL.

Global properties

Disable deferred wipeout on the master
 Disk Space Monitoring Threshold
 Environment variables
 Tool Locations

Metrics

Access keys ?

Save Apply

- Restart Jenkins Service

`systemctl restart jenkins`

```
root@ip-172-31-94-44:/# systemctl restart jenkins
root@ip-172-31-94-44:/#
```

- The port change is completed successfully.

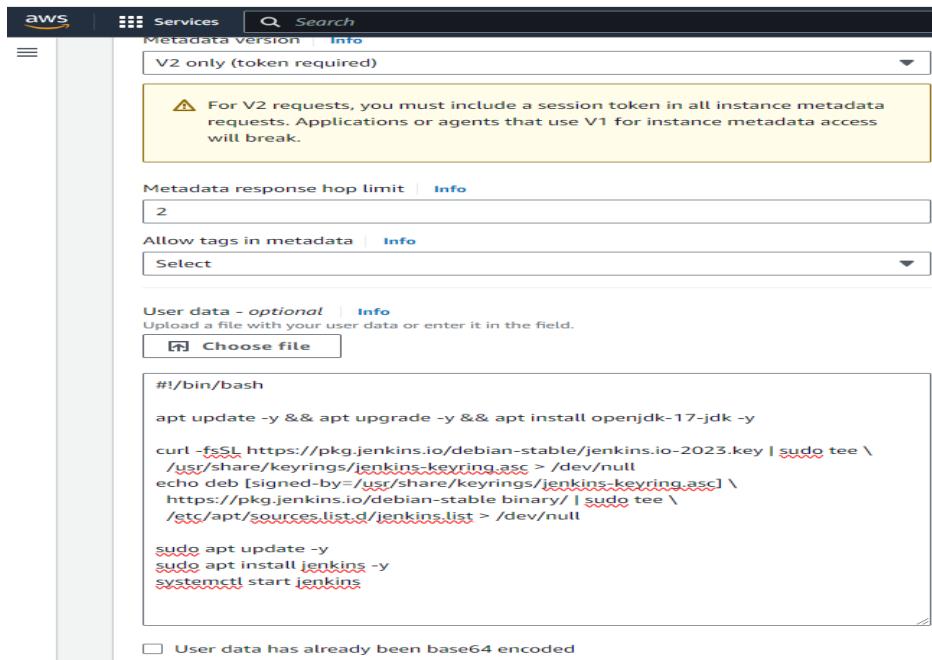
2) How to change Home Directory of Jenkins.

Default home directory path [/var/lib/jenkins]

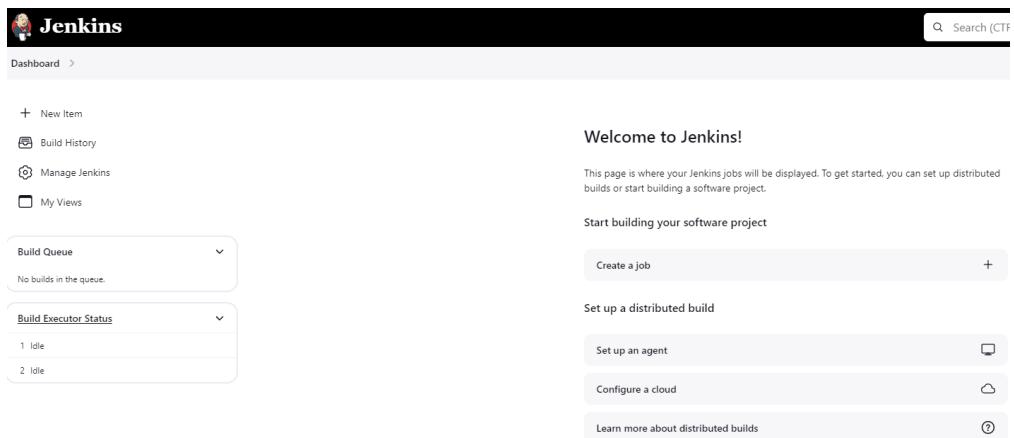
New Path we want to give is [/home/jenkins_home/jenkins]

- To change the default home directory of Jenkins, we need to have a Jenkins Server EC2 instance.

Add the below commands in user data section of EC2 instance before launching it to install Jenkins on it



- Once the EC2 instance is up, Post deployment steps are performed for configuring Jenkins to setup the Jenkins server. Once Jenkins is configured, we see the below dashboard screen.



- Now we will go to the terminal and do the below changes:

- Stop Jenkins Service

```
systemctl stop jenkins
```

```
root@ip-172-31-94-44:/# systemctl stop jenkins
root@ip-172-31-94-44:/#
```

- Create a new Jenkins Home directory using the mkdir command

```
mkdir /home/jenkins_home
```

```
root@ip-172-31-94-44:/# mkdir /home/jenkins_home
root@ip-172-31-94-44:/#
```

- Change permissions for the new home directory

```
chown jenkins:jenkins /home/jenkins_home
```

```
root@ip-172-31-94-44:/# chown jenkins:jenkins /home/jenkins_home
root@ip-172-31-94-44:/#
```

- Copy the contents from the old Jenkins Home directory to the new one

```
cp -prv /var/lib/jenkins /home/jenkins_home
```

```
root@ip-172-31-94-44:/# cp -prv /var/lib/jenkins /home/jenkins_home
'/var/lib/jenkins' -> '/home/jenkins_home/jenkins'
'/var/lib/jenkins/.java' -> '/home/jenkins_home/jenkins/.java'
'/var/lib/jenkins/.java/fonts' -> '/home/jenkins_home/jenkins/.java/fonts'
'/var/lib/jenkins/.java/fonts/17.0.12' -> '/home/jenkins_home/jenkins/.java/fonts/17.0.12'
'/var/lib/jenkins/.java/fonts/17.0.12/fcinfo-1-ip-172-31-94-44-Ubuntu-24.04-en-.properties' -> '/home/jenkins_home/jenkins/.java/fonts/17.0.12/fcinfo-1-ip-172-31-94-44-Ubuntu-24.04-en-.properties'
'/var/lib/jenkins/secret.key' -> '/home/jenkins_home/jenkins/secret.key'
'/var/lib/jenkins/secret.key.not-so-secret' -> '/home/jenkins_home/jenkins/secret.key.not-so-secret'
'/var/lib/jenkins/plugins' -> '/home/jenkins_home/jenkins/plugins'
'/var/lib/jenkins/plugins/ionicons-api.jpi' -> '/home/jenkins_home/jenkins/plugins/ionicons-api.jpi'
'/var/lib/jenkins/plugins/ionicons-api' -> '/home/jenkins_home/jenkins/plugins/ionicons-api'
'/var/lib/jenkins/plugins/ionicons-api/META-INF' -> '/home/jenkins_home/jenkins/plugins/ionicons-api/META-INF'
'/var/lib/jenkins/plugins/ionicons-api/META-INF/MANIFEST.MF' -> '/home/jenkins_home/jenkins/plugins/ionicons-api/META-INF/MANIFEST.MF'
'/var/lib/jenkins/plugins/ionicons-api/META-INF/maven' -> '/home/jenkins_home/jenkins/plugins/ionicons-api/META-INF/maven'
```

- Assign Jenkins as the user for the new home directory

```
usermod -d /home/jenkins_home jenkins
```

```
root@ip-172-31-94-44:/# usermod -d /home/jenkins_home jenkins
root@ip-172-31-94-44:/#
```

- Edit the Jenkins configuration file.

```
vi /etc/default/jenkins
```

```
[root@ip-172-31-94-44 ~]# vi /etc/default/jenkins
```

- Scroll down until you reach the JENKINS_HOME entry

Edit the line to include the path to the new home directory

In our Case

```
NAME=jenkins
```

```
# pulled in from  
NAME=jenkins
```

- Old home directory path

```
JENKINS_HOME=/var/lib/$NAME
```

```
# jenkins home location  
JENKINS_HOME=/var/lib/$NAME
```

- New home directory path

```
JENKINS_HOME=/home/jenkins_home/$NAME
```

```
# jenkins home location  
JENKINS_HOME=/home/jenkins_home/$NAME
```

Esc, then: wq! (Save and exit the file)

- Now we will edit the jenkins service file with new port number

```
vi usr/lib/systemd/system/jenkins.service
```

```
root@ip-172-31-94-44:/#
root@ip-172-31-94-44:/# vi /usr/lib/systemd/system/jenkins.service
```

- Scroll down until you reach the Environment and Working Directory

Edit the line to include the path to the new home directory

In our Case

- OLD Entry of Jenkins home path

Environment="JENKINS_HOME=/var/lib/jenkins"
WorkingDirectory=/var/lib/jenkins

```
# Directory where Jenkins stores its configuration and workspaces
Environment="JENKINS_HOME=/var/lib/jenkins"
WorkingDirectory=/var/lib/jenkins
```

- NEW Entry of Jenkins home path

Environment="JENKINS_HOME=/home/jenkins_home/jenkins"
WorkingDirectory=/home/jenkins_home/jenkins

Esc, then: wq! (Save and exit the file)

- Reload the Daemon

systemctl daemon-reload

```
root@ip-172-31-94-44:/# systemctl daemon-reload
root@ip-172-31-94-44:/#
```

- Rename the old Jenkins home directory

mv /var/lib/jenkins /var/lib/jenkins.old

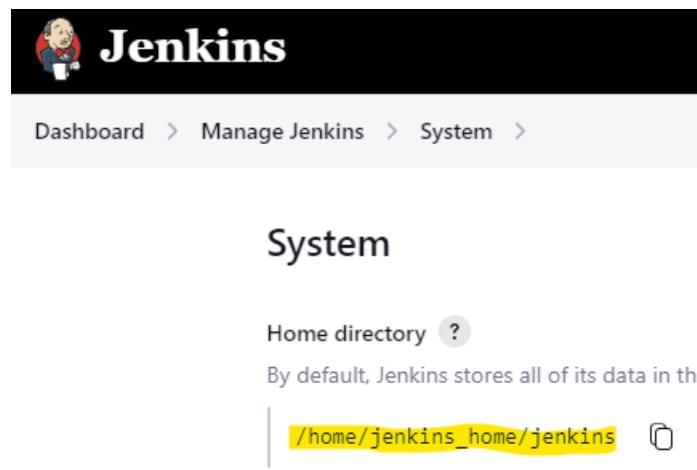
```
root@ip-172-31-94-44:/# mv /var/lib/jenkins /var/lib/jenkins.old
root@ip-172-31-94-44:/#
```

- Restart Jenkins Service

```
systemctl restart jenkins
```

```
root@ip-172-31-94-44:/# systemctl restart jenkins
root@ip-172-31-94-44:/#
```

- Verify the New Home directory under > Manage Jenkins > System > Home directory.



- The Home Directory path has been changed successfully.

3) How to run Remote Job in Jenkins.

We can trigger a job remotely through a Web URL in Jenkins also.

We will follow the below steps to achieve the same.

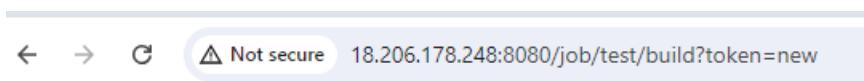
While configuring the job, we need to select Trigger builds remotely under Build Triggers, then we can give any authentication token eg new and Click Save.

The screenshot shows the Jenkins job configuration page for a job named 'test'. The left sidebar lists configuration sections: General, Source Code Management, Build Triggers (selected), Build Environment, Build Steps, and Post-build Actions. The 'Build Triggers' section contains a checked checkbox for 'Trigger builds remotely (e.g., from scripts)' and a sub-section for 'Authentication Token' with a 'new' button. Other trigger options like 'Build after other projects are built', 'Build periodically', 'GitHub hook trigger for GITScm polling', and 'Poll SCM' are available but unchecked. The 'Build Environment' section includes a checked checkbox for 'Delete workspace before build starts' with an 'Advanced' dropdown, and options for 'Use secret text(s) or file(s)', 'Add timestamps to the Console Output', and 'Inspect build log for published build scans'. At the bottom are 'Save' and 'Apply' buttons.

To trigger the build remotely we need use the below URL

JENKINS_URL/job/test/build?token=TOKEN_NAME or
/buildWithParameters?token=TOKEN_NAME

Below screenshot is in our case



We see the 2nd Build is also executed successfully.

The screenshot shows a Jenkins project named 'test'. The top navigation bar includes 'Dashboard > test >'. On the left, a sidebar lists project actions: Status (highlighted), Changes, Workspace, Build Now, Configure, Delete Project, and Rename. To the right, a green checkmark icon indicates the build status is 'test'. Below this is a 'Permalinks' section with a list of recent builds:

- Last build (#1), 10 min ago
- Last stable build (#1), 10 min ago
- Last successful build (#1), 10 min ago
- Last completed build (#1), 10 min ago

Below the permalinks is a 'Build History' section showing two builds: #2 (Aug 21, 2024, 12:05 PM) and #1 (Aug 21, 2024, 11:55 AM). At the bottom of the history are links for 'Atom feed for all' and 'Atom feed for failures'.

Alternate Way to run the job through remotely is by using a plugin

We need to install the plugin Build Authorization Token Root

The screenshot shows the Jenkins 'Manage Jenkins > Plugins' page. The left sidebar has sections for Updates, Available plugins (highlighted), Installed plugins, Advanced settings, and Download progress. The main area shows a search bar with 'build auth' and a list of available plugins:

Install	Name	Released
<input checked="" type="checkbox"/>	Build Authorization Token Root 1.0.0	2 yr 1 mo ago
<input type="checkbox"/>	Build Token Trigger 1.0.0	6 yr 4 mo ago

Plugins

Updates

Available plugins

Installed plugins

Advanced settings

Download progress

Ant		Success
Gradle		Success
Pipeline		Success
Github Branch Source		Success
Pipeline: GitHub Groovy Libraries		Success
Pipeline Graph View		Success
Git		Success
SSH Build Agents		Success
Matrix Authorization Strategy		Success
PAM Authentication		Success
LDAP		Success
Email Extension		Success
Mailer		Success
Dark Theme		Success
Loading plugin extensions		Success
Simple Theme		Success
Loading plugin extensions		Success
Role-based Authorization Strategy		Success
Loading plugin extensions		Success
JavaMail API		Success
SSH server		Success
Build Authorization Token Root		Success
Loading plugin extensions		Success

Now we will run the below URL to see if build runs.

<Jenkins_URL>buildByToken/build?job=<Job_Name>&token=<token Name>

18.206.178.248:8080/buildByToken/build?job=test&token=new

See the Build 3 is also completed successfully.

Dashboard > test >

Status
test

- Changes
- Workspace
- Build Now
- Configure
- Delete Project
- Rename

Build History trend ▾

Build	Date
#3	Aug 21, 2024, 12:14 PM
#2	Aug 21, 2024, 12:05 PM
#1	Aug 21, 2024, 11:55 AM

[Atom feed for all](#) [Atom feed for failures](#)

We can also run it on the Ubuntu terminal using the below command.

Just need to add “\” before the & and then run.

```
<Jenkins_URL>buildByToken/build?job=<Job_Name>\&token=<token Name>
```

```
root@ip-172-31-33-31:/var/lib/jenkins/workspace/test# curl http://18.206.178.248:8080/buildByToken/build?job=test\&token=new
root@ip-172-31-33-31:/var/lib/jenkins/workspace/test#
```

Build 4 is completed successfully

The screenshot shows the Jenkins interface for the 'test' project. At the top, there's a navigation bar with a Jenkins logo and the word 'Jenkins'. Below it, a breadcrumb navigation shows 'Dashboard > test >'. On the left, a sidebar lists project management options: Status (highlighted), Changes, Workspace, Build Now, Configure, Delete Project, and Rename. The main content area has a green checkmark icon next to the project name 'test'. Below it, a section titled 'Permalinks' lists four recent builds. To the right, a large box displays the 'Build History' for the 'test' project. It shows four builds: #4 (highlighted in yellow), #3, #2, and #1, each with a timestamp from Aug 21, 2024. At the bottom of the history box, there are links for 'Atom feed for all' and 'Atom feed for failures'.

Build	Date
#4	Aug 21, 2024, 12:20 PM
#3	Aug 21, 2024, 12:14 PM
#2	Aug 21, 2024, 12:05 PM
#1	Aug 21, 2024, 11:55 AM

4) How to create GitHub hook trigger for GITScm polling through Jenkins.

This is used when a job needs to be triggered when there is any change in the GitHub repo.

Steps to followed are below:

While configuring the job, Check the option of GitHub hook trigger for GITScm polling under Build Triggers and then click Save.

The screenshot shows the Jenkins 'Configure' screen for a job named 'test'. The left sidebar lists configuration sections: General, Source Code Management, Build Triggers (which is selected and highlighted in grey), Build Environment, Build Steps, and Post-build Actions. The 'Build Triggers' section contains several options: 'Trigger builds remotely (e.g., from scripts)', 'Build after other projects are built', 'Build periodically', 'GitHub hook trigger for GITScm polling' (which is checked and highlighted in yellow), and 'Poll SCM'. Below the triggers is the 'Build Environment' section, which includes options like 'Delete workspace before build starts' (checked), 'Advanced' (with a dropdown menu), 'Use secret text(s) or file(s)', 'Add timestamps to the Console Output', 'Inspect build log for published build scans', 'Terminate a build if it's stuck', and 'With Ant'. At the bottom right are 'Save' and 'Apply' buttons.

Then on the GitHub side, Go to Settings

sourabh913 / Train-Ticket-Reservation-System

Code Pull requests Actions Projects Wiki Security Insights Settings

Train-Ticket-Reservation-System Public
forked from shashirajraja/Train-Ticket-Reservation-System

master 1 Branch 0 Tags

Go to file Add file Code

Select Webhooks > Add webhook

sourabh913 / Train-Ticket-Reservation-System

Type to search

Code Pull requests Actions Projects Wiki Security Insights Settings

General Webhooks

Add webhook

Access Collaborators Moderation options

Code and automation Branches Tags Rules Actions

Webhooks

Need to put in password for your GitHub account.

Signed in as @sourabh913

Password

Forgot password?

Confirm

Tip: You are entering [sudo mode](#). After you've performed a sudo-protected action, you'll only be asked to re-authenticate again after a few hours of inactivity.

Then under the Payload URL, give the Jenkins URL /github webhook

And under Content type > Select application/json.

Click Add Webhook

The screenshot shows the GitHub settings interface for a repository. On the left, a sidebar lists various settings categories: General, Access, Collaborators, Moderation options, Code and automation (with Branches, Tags, Rules, Actions), Webhooks (which is selected and highlighted in blue), Environments, Codespaces, Pages, Security (with Code security and analysis, Deploy keys, and Secrets and variables), Integrations (with GitHub Apps and Email notifications). The main content area is titled "Webhooks / Add webhook". It contains instructions about sending POST requests to a URL with event details. The "Payload URL *" field is filled with "http://18.206.178.248:8080/github-webhook". The "Content type *" dropdown is set to "application/json". There is a "Secret" input field which is empty. Under "SSL verification", there is a note that SSL certificates are verified by default, and two radio buttons: "Enable SSL verification" (selected) and "Disable (not recommended)". Below this, it asks "Which events would you like to trigger this webhook?", with three options: "Just the push event." (selected), "Send me everything.", and "Let me select individual events.". A checked checkbox labeled "Active" indicates that event details will be delivered when triggered. At the bottom is a green "Add webhook" button.

The screenshot shows the GitHub settings interface for a repository. On the left, the sidebar includes General, Access, Collaborators, Moderation options, Code and automation (with Branches), and Webhooks. The main content area is titled "Webhooks" and contains a list of active webhooks. One webhook is listed with the URL "http://18.206.178.248:8080/github-... (all events)". To the right of the URL are "Edit" and "Delete" buttons. A note below the list states "Last delivery was successful."

For Example, in my case I made a change under my repo > screenshots > rename viewprofil.png to viewprofile.png

Before Change

The image consists of two screenshots from GitHub illustrating a file rename operation.

Screenshot 1: File List

This screenshot shows a list of files in the 'Screenshots' directory of the 'Train-Ticket-Reservation-System' repository. The file 'viewprofile.png' has been renamed to 'viewprofilा.png'. The commit message is 'Rename viewprofile.png to viewprofilा.png'. The file was committed 2 minutes ago by user 'sourabh913'.

Name	Last commit message	Last commit date
..		
Availability.png	Screenshots of WebPages for this project	5 years ago
Search.png	Screenshots of WebPages for this project	5 years ago
TicketBook.png	Screenshots of WebPages for this project	5 years ago
addtrains.png	Screenshots of WebPages for this project	5 years ago
booknow.png	Screenshots of WebPages for this project	5 years ago
fare result.png	Screenshots of WebPages for this project	5 years ago
fareenquiry.png	Screenshots of WebPages for this project	5 years ago
login.png	Screenshots of WebPages for this project	5 years ago
passwordchange.png	Screenshots of WebPages for this project	5 years ago
registeruser.png	Screenshots of WebPages for this project	5 years ago
usehome.png	Screenshots of WebPages for this project	5 years ago
viewprofile.png	Rename viewprofile.png to viewprofilा.png	2 minutes ago

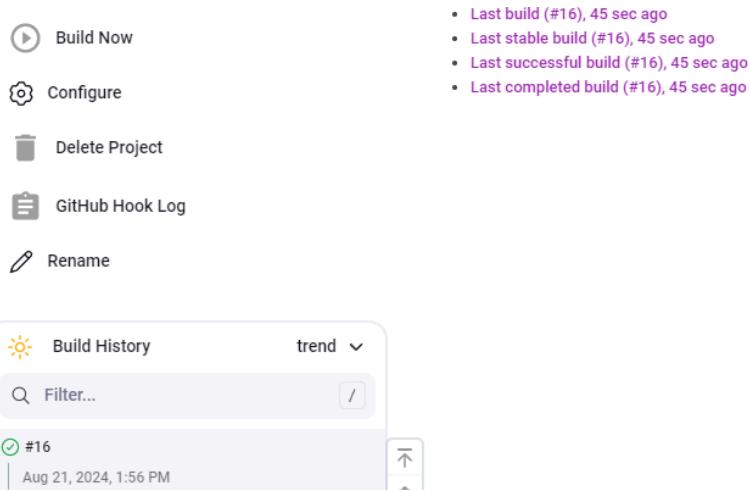
Screenshot 2: Commit Changes Dialog

This screenshot shows the 'Commit changes' dialog for the file 'viewprofile.png'. The commit message is 'Rename viewprofilा.png to viewprofile.png'. The extended description field contains the placeholder 'Add an optional extended description..'. The 'Commit directly to the master branch' radio button is selected. The 'Commit changes' button is highlighted in green.

After Change

Train-Ticket-Reservation-System / Screenshots /		
		Add file ...
sourabh913 Rename viewprofile.png to viewprofile.png		064c16f · now · History
This branch is 2 commits ahead of shashiraJraJa/Train-Ticket-Reservation-System:master .		I Contribute · Sync fork ·
Name	Last commit message	Last commit date
 Availability.png	Screenshots of WebPages for this project	5 years ago
 search.png	Screenshots of WebPages for this project	5 years ago
 TicketBook.png	Screenshots of WebPages for this project	5 years ago
 addtrains.png	Screenshots of WebPages for this project	5 years ago
 booknow.png	Screenshots of WebPages for this project	5 years ago
 fare result.png	Screenshots of WebPages for this project	5 years ago
 fareenquiry.png	Screenshots of WebPages for this project	5 years ago
 login.png	Screenshots of WebPages for this project	5 years ago
 passwordchange.png	Screenshots of WebPages for this project	5 years ago
 registeruser.png	Screenshots of WebPages for this project	5 years ago
 userhome.png	Screenshots of WebPages for this project	5 years ago
 viewprofile.png	Rename viewprofile.png to viewprofile.png	now

Immediately after the change, we see that build ran in Jenkins.



Build Now

- Last build (#16), 45 sec ago
- Last stable build (#16), 45 sec ago
- Last successful build (#16), 45 sec ago
- Last completed build (#16), 45 sec ago

Configure

Delete Project

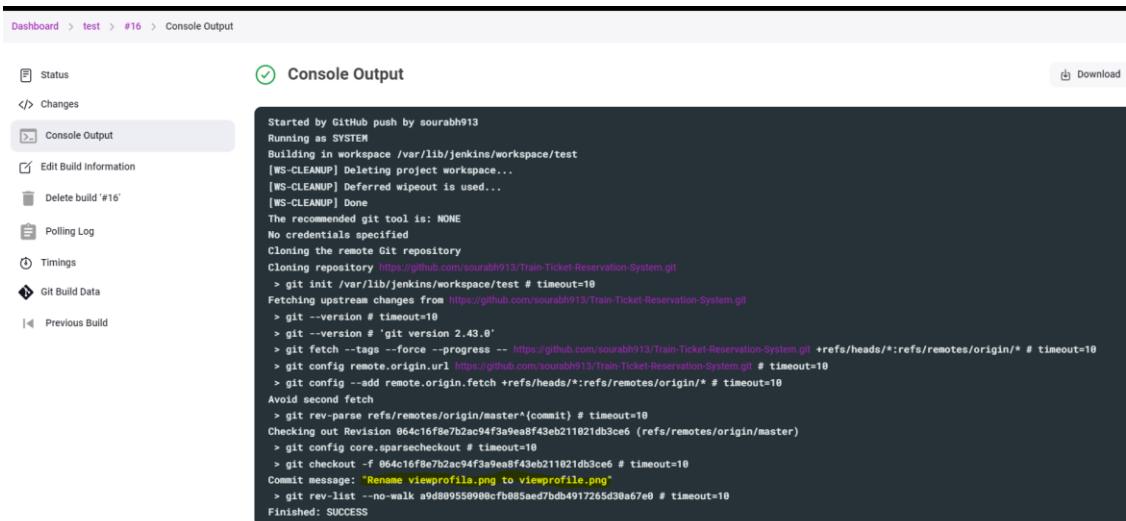
GitHub Hook Log

Rename

Build History

Filter... #16 Aug 21, 2024, 1:56 PM

We can see the changes in the console output reflected successfully



Status > test > #16 > Console Output

Console Output

```

Started by GitHub push by sourabh913
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/test
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] Done
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/sourabh913/Train-Ticket-Reservation-System.git
> git init /var/lib/jenkins/workspace/test # timeout=10
Fetching upstream changes from https://github.com/sourabh913/Train-Ticket-Reservation-System.git
> git --version # timeout=10
> git --version # git version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/sourabh913/Train-Ticket-Reservation-System.git +refs/heads/*:refs/remotes/origin/*
> git config remote.origin.url https://github.com/sourabh913/Train-Ticket-Reservation-System.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/*
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 064c16f8e7b2ac94f3a9ea8f43eb211021db3ce6 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 064c16f8e7b2ac94f3a9ea8f43eb211021db3ce6 # timeout=10
Commit message: "Rename viewprofile.png to viewprofile.png"
> git rev-list --no-walk a9d809558900cfb085aed7bdb4917265d30a67e0 # timeout=10
Finished: SUCCESS

```

5) How to install Jenkins in Tomcat.

- To install jenkins in Tomcat, we need to have a EC2 instance with open-jdk installed on it.

Name	Instance ID	Instance state	Instance type
Tomcat	i-077770ae682cae1e1	Running	t2.micro

Add the below commands in user data section of EC2 instance before launching it to install open-jdk on it.

```
#!/bin/bash  
  
apt update -y && apt upgrade -y && apt install openjdk-17-jdk -y
```

- Once EC2 is up, we will verify that OpenJDK is installed properly by checking its version.

java --version (Verify open jdk is installed)

```
root@ip-172-31-80-208:/home/ubuntu# java --version  
openjdk 17.0.12 2024-07-16  
OpenJDK Runtime Environment (build 17.0.12+7-Ubuntu-1ubuntu224.04)  
OpenJDK 64-Bit Server VM (build 17.0.12+7-Ubuntu-1ubuntu224.04, mixed mode, sharing)  
root@ip-172-31-80-208:/home/ubuntu#
```

- Now we install Apache tomcat version 9 so that we can install jenkins on it.
- Since Tomcat version 9 is not available as default package in ubuntu so we have to install Tomcat 9 manually. Below are the steps to download and install tomcat 9.

```
wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.93/bin/apache-tomcat-9.0.93.tar.gz
```

ls -ltr to check if the file is downloaded

```
root@ip-172-31-80-208:/home/ubuntu# wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.93/bin/apache-tomcat-9.0.93.tar.gz
--2024-09-02 10:26:31-- https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.93/bin/apache-tomcat-9.0.93.tar.gz
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42:644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 12122732 (12M) [application/x-gzip]
Saving to: 'apache-tomcat-9.0.93.tar.gz'

100%[=====] 11.56M --.-KB/s in 0.1s

2024-09-02 10:26:31 (111 MB/s) - 'apache-tomcat-9.0.93.tar.gz' saved [12122732/12122732]

root@ip-172-31-80-208:/home/ubuntu# ls -ltr
total 11840
-rw-r--r-- 1 root root 12122732 Aug 2 22:59 apache-tomcat-9.0.93.tar.gz
root@ip-172-31-80-208:/home/ubuntu#
```

- Untar the tar file using below command and check tomcat folder is created or not.

tar xzvf apache-tomcat-9.0.93.tar.gz

```
root@ip-172-31-80-208:/home/ubuntu#
root@ip-172-31-80-208:/home/ubuntu# tar xzvf apache-tomcat-9.0.93.tar.gz
apache-tomcat-9.0.93/conf/
apache-tomcat-9.0.93/conf/catalina.policy
apache-tomcat-9.0.93/conf/catalina.properties
apache-tomcat-9.0.93/conf/context.xml
apache-tomcat-9.0.93/conf/jaspic-providers.xml
apache-tomcat-9.0.93/conf/jaspic-providers.xsd
apache-tomcat-9.0.93/conf/logging.properties
apache-tomcat-9.0.93/conf/server.xml
apache-tomcat-9.0.93/conf/tomcat-users.xml
apache-tomcat-9.0.93/conf/tomcat-users.xsd
apache-tomcat-9.0.93/conf/web.xml
apache-tomcat-9.0.93/bin/
apache-tomcat-9.0.93/lib/
apache-tomcat-9.0.93/logs/
apache-tomcat-9.0.93/temp/
apache-tomcat-9.0.93/webapps/
apache-tomcat-9.0.93/webapps/ROOT/
apache-tomcat-9.0.93/webapps/ROOT/WEB-INF/
apache-tomcat-9.0.93/webapps/docs/
```

ls -ltr

```
root@ip-172-31-80-208:/home/ubuntu# ls -ltr
total 11844
-rw-r--r-- 1 root root 12122732 Aug 2 22:59 apache-tomcat-9.0.93.tar.gz
drwxr-xr-x 9 root root 4096 Sep 2 10:27 apache-tomcat-9.0.93
drwxr-xr-x 2 root root 4096 Sep 2 10:27 apache-tomcat-9.0.93/
```

- Now rename the extracted folder tomcat so that it's easy to remember and check.

```
mv apache-tomcat-9.0.93 tomcat
```

```
ls -ltr
```

```
root@ip-172-31-80-208:/home/ubuntu# mv apache-tomcat-9.0.93 tomcat
root@ip-172-31-80-208:/home/ubuntu#
root@ip-172-31-80-208:/home/ubuntu# ls -ltr
total 11844
-rw-r--r-- 1 root root 12122732 Aug  2 22:59 apache-tomcat-9.0.93.tar.gz
drwxr-xr-x 9 root root     4096 Sep  2 10:27 tomcat
root@ip-172-31-80-208:/home/ubuntu#
```

- Now we have to create the tomcat service manually in the system location mentioned below and add the contents in the service file.

```
vi /etc/systemd/system/tomcat.service
```

```
root@ip-172-31-80-208:/home/ubuntu# vi /etc/systemd/system/tomcat.service
```

Contents of tomcat.service file

[Unit]

Description=Tomcat 9 servlet container

After=network.target

[Service]

Type=forking

Environment="JAVA_HOME=/usr/lib/jvm/java-17-openjdk-amd64"

Environment="JAVA_OPTS=-Djava.security.egd=file:///dev/urandom -Djava.awt.headless=true"

Environment="CATALINA_BASE=/home/ubuntu/tomcat/"

Environment="CATALINA_HOME=/home/ubuntu/tomcat/"

Environment="CATALINA_PID=/home/ubuntu/tomcat/temp/tomcat.pid"

Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"

ExecStart=/home/ubuntu/tomcat/bin/startup.sh

ExecStop=/home/ubuntu/tomcat/bin/shutdown.sh

[Install]

WantedBy=multi-user.target

```
root@ip-172-31-80-208: /home/ubuntu
[Unit]
Description=Tomcat 9 servlet container
After=network.target

[Service]
Type=forking

Environment="JAVA_HOME=/usr/lib/jvm/java-17-openjdk-amd64"
Environment="JAVA_OPTS=-Djava.security.egd=file:///dev/urandom -Djava.awt.headless=true"

Environment="CATALINA_BASE=/home/ubuntu/tomcat/"
Environment="CATALINA_HOME=/home/ubuntu/tomcat/"
Environment="CATALINA_PID=/home/ubuntu/tomcat/temp/tomcat.pid"
Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"

ExecStart=/home/ubuntu/tomcat/bin/startup.sh
ExecStop=/home/ubuntu/tomcat/bin/shutdown.sh

[Install]
WantedBy=multi-user.target
```

- Now we will reload the daemon and start tomcat service and also verify that the service is running.

systemctl daemon-reload

systemctl start tomcat

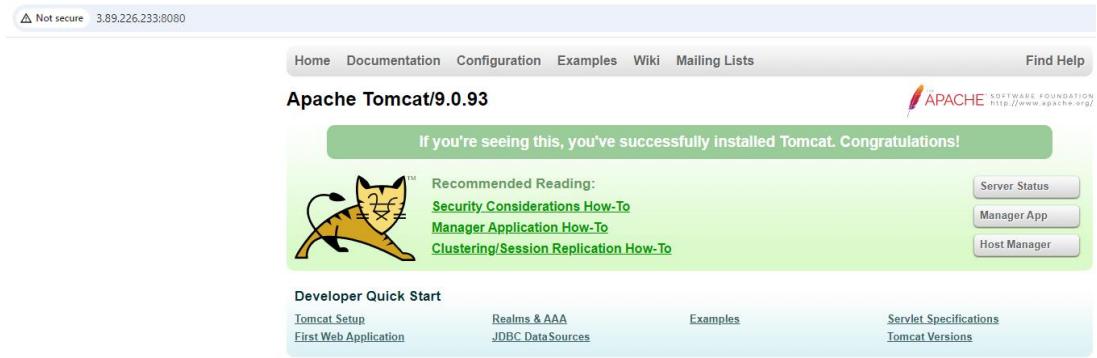
systemctl status tomcat

```
root@ip-172-31-80-208: /home/ubuntu# systemctl daemon-reload
root@ip-172-31-80-208: /home/ubuntu#
root@ip-172-31-80-208: /home/ubuntu#
root@ip-172-31-80-208: /home/ubuntu#
root@ip-172-31-80-208: /home/ubuntu# systemctl start tomcat
root@ip-172-31-80-208: /home/ubuntu# systemctl status tomcat
● tomcat.service - Tomcat 9 servlet container
   Loaded: loaded (/etc/systemd/system/tomcat.service; disabled; preset: enabled)
   Active: active (running) since Mon 2024-09-02 10:37:15 UTC; 8s ago
     Process: ExecStart=/home/ubuntu/tomcat/bin/startup.sh (code-exited, status=0/SUCCESS)
    Main PID: 15789 (java)
       Tasks: 31 (limit: 1130)
      Memory: 155.9M (peak: 158.7M)
        CPU: 2.585s
       CGroup: /system.slice/tomcat.service
               └─15796 /usr/lib/jvm/java-17-openjdk-amd64/bin/java -Djava.util.logging.config.file=/home/ubuntu/tomcat/conf/logging.properties -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djava.security.egd=file:///dev/urandom

Sep 02 10:37:15 ip-172-31-80-208 systemd[1]: Starting tomcat.service - Tomcat 9 servlet container...
Sep 02 10:37:15 ip-172-31-80-208 startup.sh[15789]: Tomcat started.
Sep 02 10:37:15 ip-172-31-80-208 systemd[1]: Started tomcat.service - Tomcat 9 servlet container.
lines 1-14/14 (END)
```

- We will verify that tomcat is running or not on the WEG UI by accessing tomcat URL – PublicIP of EC2 instance:8080 (Tomcat works on port 8080)

<http://3.89.226.233:8080>



- Once we have apache tomcat 9 up and running, we will download jenkins.war file from the official webpage from the terminal by running the below command.

wget <http://mirrors.jenkins.io/war-stable/latest/jenkins.war>

```
root@ip-172-31-80-208:/home/ubuntu# wget http://mirrors.jenkins.io/war-stable/latest/jenkins.war
--2024-09-02 10:38:57-- https://mirrors.jenkins.io/war-stable/latest/jenkins.war
Resolving mirrors.jenkins.io (mirrors.jenkins.io)... 20.7.178.24, 2603:1690:4005:15::15a
Connecting to mirrors.jenkins.io (mirrors.jenkins.io)|20.7.178.24|:80... connected.
HTTP request sent, awaiting response... 308 Permanent Redirect
Location: https://mirrors.jenkins.io/war-stable/latest/jenkins.war [following]
--2024-09-02 10:38:57-- https://mirrors.jenkins.io/war-stable/latest/jenkins.war
Connecting to mirrors.jenkins.io (mirrors.jenkins.io)|20.7.178.24|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://archives.jenkins.io/war-stable/latest/jenkins.war [following]
--2024-09-02 10:38:57-- https://archives.jenkins.io/war-stable/latest/jenkins.war
Resolving archives.jenkins.io (archives.jenkins.io)... 46.101.121.132, 2a03:b0c0:0:3:d0::9bc:0001
Connecting to archives.jenkins.io (archives.jenkins.io)|46.101.121.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 93287150 (89M)
Saving to: 'jenkins.war'

jenkins.war                                              [  0%] 88.96M 7.20MB/s    in 15s
2024-09-02 10:39:13 (5.81 MB/s) - 'jenkins.war' saved [93287150/93287150]
root@ip-172-31-80-208:/home/ubuntu#
```

ls -ltr

```
root@ip-172-31-80-208:/home/ubuntu# ls -ltr
total 102952
-rw-r--r-- 1 root root 12122732 Aug  2 22:59 apache-tomcat-9.0.93.tar.gz
-rw-r--r-- 1 root root 93287150 Aug  7 10:12 jenkins.war
drwxr-xr-x 9 root root     4096 Sep  2 10:27 tomcat
root@ip-172-31-80-208:/home/ubuntu#
```

- Now we need to copy this file to webapps directory located in Tomcat home. So, if you have tomcat installed in home directory, then run the following command.

cp -rv jenkins.war tomcat/webapps

```
root@ip-172-31-80-208:/home/ubuntu# cp -rv jenkins.war tomcat/webapps  
'jenkins.war' -> 'tomcat/webapps/jenkins.war'  
root@ip-172-31-80-208:/home/ubuntu#
```

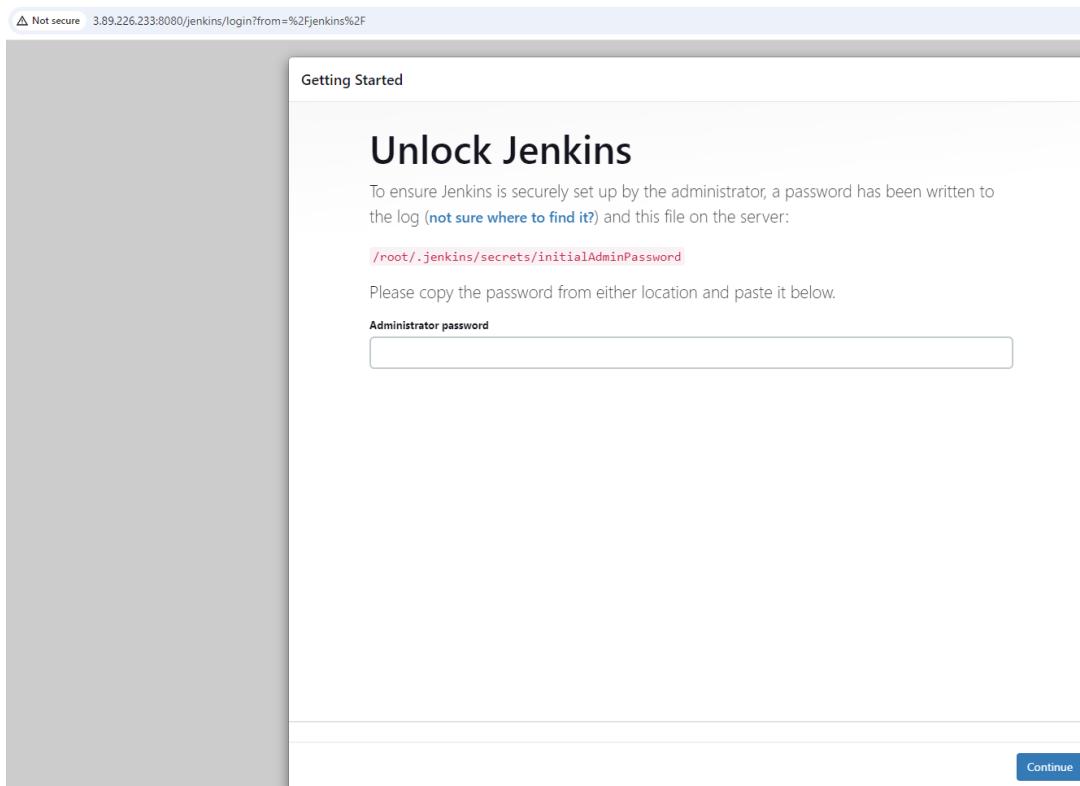
- We will also verify that jenkins.war has been copied successfully and also jenkins folder is also created.

ls -ltr tomcat/webapps

```
root@ip-172-31-80-208:/home/ubuntu# ls -ltr tomcat/webapps  
total 91128  
drwxr-x--- 3 root root 4096 Sep 2 10:27 ROOT  
drwxr-x--- 16 root root 4096 Sep 2 10:27 docs  
drwxr-x--- 7 root root 4096 Sep 2 10:27 examples  
drwxr-x--- 6 root root 4096 Sep 2 10:27 host-manager  
drwxr-x--- 6 root root 4096 Sep 2 10:27 manager  
-rw-r--r-- 1 root root 93287150 Sep 2 10:40 jenkins.war  
drwxr-x--- 10 root root 4096 Sep 2 10:40 jenkins  
root@ip-172-31-80-208:/home/ubuntu#
```

- Since our tomcat server is up. To start using the jenkins, we will visit the following URL.

<http://3.89.226.233:8080/jenkins>



- We will have to configure the jenkins server now. As mentioned on the webpage, we can find the admin password from location '/root/.jenkins/secrets/initialAdminPassword'. Get the password with the command,

cat /root/.jenkins/secrets/initialAdminPassword

```
root@ip-172-31-80-208:/home/ubuntu# cat /root/.jenkins/secrets/initialAdminPassword
5d866465f9bb44ac9bb7ad7e9b738451
root@ip-172-31-80-208:/home/ubuntu#
```

- Now complete the setup by filling out the required details. After the setup is complete, we can access jenkins & can also create new jobs on the jenkins server.

6) Install old version of Jenkins and then upgrade it to a latest version

- To install old version of jenkins, we need to have a EC2 instance with open-jdk installed on it.
- Add the below commands in user data section of EC2 instance before launching it to install open-jdk on it.

```
#!/bin/bash

apt update -y && apt upgrade -y && apt install openjdk-17-jdk -y
```

- Once EC2 is up, we will verify that OpenJDK is installed properly by checking its version.

java --version (Verify open jdk is installed)

```
root@ip-172-31-81-148:/home/ubuntu# java --version
openjdk 17.0.12 2024-07-16
OpenJDK Runtime Environment (build 17.0.12+7-Ubuntu-1ubuntu224.04)
OpenJDK 64-Bit Server VM (build 17.0.12+7-Ubuntu-1ubuntu224.04, mixed mode, sharing)
root@ip-172-31-81-148:/home/ubuntu#
```

- Now we will prepare the system for jenkins in ubuntu by executing the following command.
- Below is the Debian package repository of Jenkins to automate installation and upgrade. To use this repository, first add the key to your system

```
wget -O /usr/share/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
```

```
root@ip-172-31-81-13:/home/ubuntu# sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
> https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
--2024-09-02 13:04:39- https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.38.133, 2a04:4e42:79::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|146.75.38.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3175 (3.1K) [application/pgp-keys]
Saving to: '/usr/share/keyrings/jenkins-keyring.asc'

/usr/share/keyrings/jenkins-k 100%[=====] 3.10K --.-KB/s in 0s
2024-09-02 13:04:39 (45.8 MB/s) - '/usr/share/keyrings/jenkins-keyring.asc' saved [3175/3175]

root@ip-172-31-81-13:/home/ubuntu#
```

- Add a Jenkins apt repository entry:

```
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \
https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
```

```
root@ip-172-31-81-13:/home/ubuntu# echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \
> https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
> /etc/apt/sources.list.d/jenkins.list > /dev/null
root@ip-172-31-81-13:/home/ubuntu#
```

apt-get update

```
root@ip-172-31-81-13:/home/ubuntu# apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Ign:4 https://pkg.jenkins.io/debian-stable binary/ InRelease
Get:5 https://pkg.jenkins.io/debian-stable binary/ Release [2044 B]
Get:6 https://pkg.jenkins.io/debian-stable binary/ Release.gpg [833 B]
Hit:7 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:8 https://pkg.jenkins.io/debian-stable binary/ Packages [27.6 kB]
Fetched 157 kB in 1s (205 kB/s)
Reading package lists... Done
root@ip-172-31-81-13:/home/ubuntu#
```

- Now we will check which all versions of Jenkins are available, so that we can install an old version and then perform an upgrade to the latest version. Below is the command for the same. (It will give all the versions starting from the oldest to the latest.

apt-cache madison jenkins

- Here we have randomly selected a version to install i.e. 2.387.3. To install the specific jenkins version, we need to run the below command.

```
apt-get install jenkins=2.387.3 -y
```

```

root@ip-172-31-81-13:/home/ubuntu# apt-get install jenkins=2.387.3 -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  net-tools
The following NEW packages will be installed:
  jenkins net-tools
0 upgraded, 2 newly installed, 0 to remove and 2 not upgraded.
Need to get 98.1 MB of archives.
After this operation, 99.2 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble/main amd64 net-tools amd64 2.10-0.1ubuntu4 [204 kB]
Get:2 https://pkg.jenkins.io/debian-stable binary/ jenkins 2.387.3 [97.9 MB]
Fetched 98.1 MB in 18s (5348 kB/s)
Selecting previously unselected package net-tools.
(Reading database ... 113852 files and directories currently installed.)
Preparing to unpack .../net-tools_2.10-0.1ubuntu4_amd64.deb ...
Unpacking net-tools (2.10-0.1ubuntu4) ...

```

- We can see in the below screenshot that Jenkins 2.387.3 version is installed.

```

Selecting previously unselected package jenkins.
Preparing to unpack .../jenkins_2.387.3_all.deb ...
Unpacking jenkins (2.387.3) ...
Setting up net-tools (2.10-0.1ubuntu4) ...
Setting up jenkins (2.387.3) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jenkins.service.
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
  6.8.0-1012-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.8.0-1014-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...
  systemctl restart acpid.service chrony.service cron.service multipathd.service polkit.service udisks2.service

Service restarts being deferred:
  systemctl restart ModemManager.service
  /etc/needrestart/restart.d/dbus.service
  systemctl restart getty@tty1.service
  systemctl restart networkd-dispatcher.service

```

- Check status of Jenkins service.

systemctl status jenkins

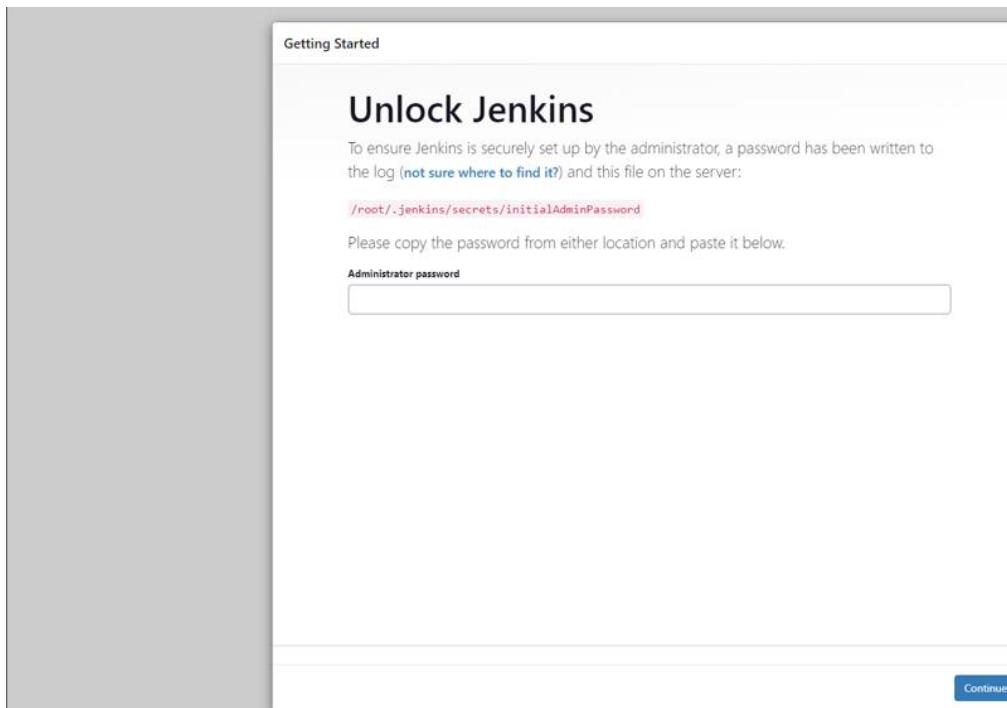
```

root@ip-172-31-25-17:/home/ubuntu# systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
  Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
  Active: active (running) since Mon 2024-09-02 14:41:12 UTC; 32s ago
    Main PID: 16218 (java)
       Tasks: 42 (limit: 1130)
      Memory: 265.3M (peak: 352.8M)
        CPU: 47.809s
       CGroup: /system.slice/jenkins.service
               └─16218 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jen

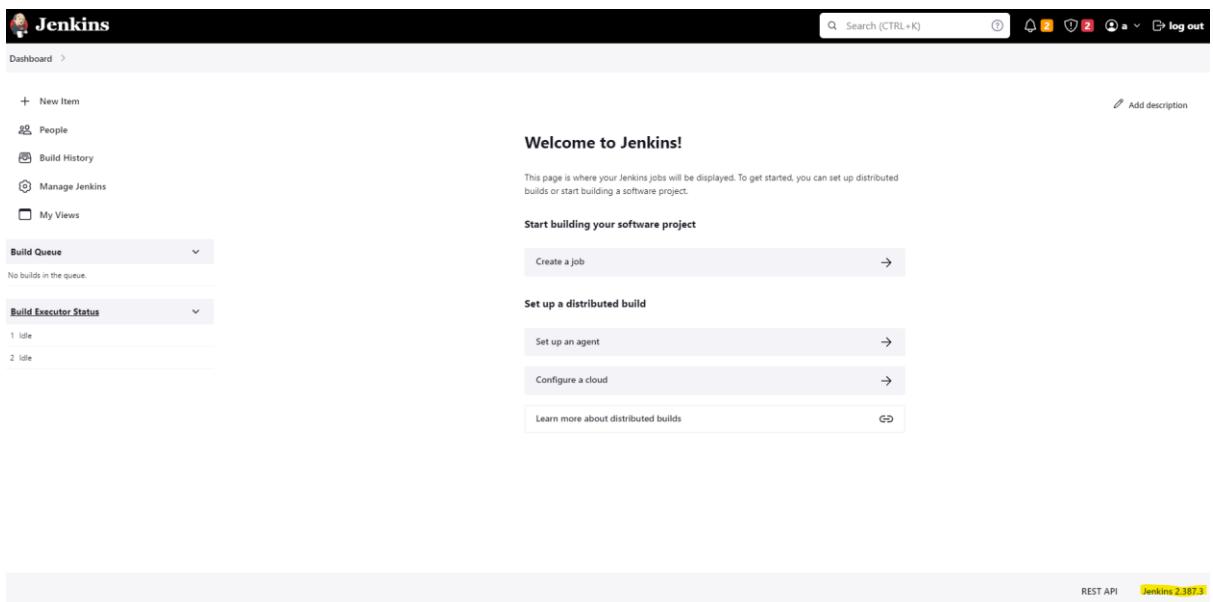
Sep 02 14:40:35 ip-172-31-25-17 jenkins[16218]: ee4fdf3fd91744b1a562091c3172130c
Sep 02 14:40:35 ip-172-31-25-17 jenkins[16218]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Sep 02 14:40:35 ip-172-31-25-17 jenkins[16218]: ****
Sep 02 14:40:35 ip-172-31-25-17 jenkins[16218]: ****
Sep 02 14:40:35 ip-172-31-25-17 jenkins[16218]: ****
Sep 02 14:41:12 ip-172-31-25-17 jenkins[16218]: 2024-09-02 14:41:12.809+0000 [id=31]      INFO      jenkins.InitR
Sep 02 14:41:12 ip-172-31-25-17 jenkins[16218]: 2024-09-02 14:41:12.870+0000 [id=24]      INFO      hudson.lifecycle
Sep 02 14:41:12 ip-172-31-25-17 systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Sep 02 14:41:13 ip-172-31-25-17 jenkins[16218]: 2024-09-02 14:41:13.010+0000 [id=46]      INFO      h.m.DownloadS
Sep 02 14:41:13 ip-172-31-25-17 jenkins[16218]: 2024-09-02 14:41:13.012+0000 [id=46]      INFO      hudson.util.R
lines 1-20/20 (END)

```

- We will verify that jenkins server is up and running and perform post deployment steps



- Once the EC2 instance is up, Post deployment steps are performed for configuring Jenkins to setup the Jenkins server. Once Jenkins is configured, we see the below dashboard screen and below screenshot will give you the confirmation of version 2.387.3 jenkins setup.



- Before upgrading to the latest version, we need to stop jenkins service by using the below command.

systemctl stop jenkins

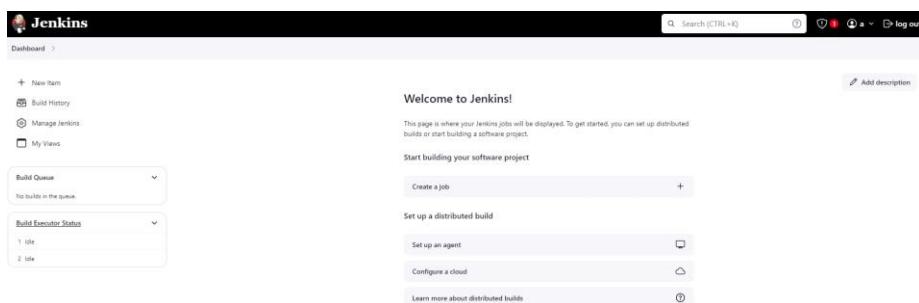
```
root@ip-172-31-25-17:/home/ubuntu# systemctl stop jenkins
root@ip-172-31-25-17:/home/ubuntu#
```

- Once the jenkins service is stopped, we will upgrade this version 2.387.3 to the latest version i.e. 2.462.1, by running the below command on the terminal. Below screenshot shows latest version is setting up.

apt-get upgrade -y

```
root@ip-172-31-25-17:/home/ubuntu# apt-get upgrade -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following upgrades have been deferred due to phasing:
  ubuntu-pro-client ubuntu-pro-client-l10n
The following packages will be upgraded:
  jenkins
1 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
Need to get 91.2 MB of archives.
After this operation, 5068 KB disk space will be freed.
Get:1 https://pkg.jenkins.io/debian-stable binary/ jenkins 2.462.1 [91.2 MB]
Fetched 91.2 MB in 2s (38.4 MB/s)
(Reading database ... 113914 files and directories currently installed.)
Preparing to unpack .../jenkins_2.462.1_all.deb ...
Unpacking jenkins (2.462.1) over (2.387.3) ...
Setting up jenkins (2.462.1) ...
Installing new version of config file /etc/init.d/jenkins ...
Scanning processes...
```

- Now we will verify the version is updated successfully by logging into the jenkins console again. Below screenshot confirms the same.



7) Jenkins Installation and Configuration Script

- Jenkins installation and configuration script is broken down into three components.
 - Jenkins Installation
 - Create Admin User
 - Installation of recommended plugins
 - Confirming Jenkins URL
- Jenkins installation script is as below:

```
#!/bin/bash
```

```
#JENKINS INSTALLATION
```

```
# Update and Upgrade Packages of Ubuntu OS  
sudo apt update -y && sudo apt upgrade -y
```

```
#Install Java SDK 17  
sudo apt install openjdk-17-jdk -y
```

```
# Check Java version  
java -version
```

```
# Add Jenkins repository key  
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc  
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
```

```
# Add Jenkins repository to sources list  
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]  
https://pkg.jenkins.io/debian-stable binary/" | sudo tee  
/etc/apt/sources.list.d/jenkins.list > /dev/null
```

```
# Update package lists  
sudo apt-get update
```

```
# Install Jenkins  
sudo apt-get install jenkins -y  
  
#Start Jenkins  
sudo systemctl start jenkins  
  
# Enable Jenkins to run on Boot  
sudo systemctl enable jenkins
```

- Jenkins create admin user script is as below: **Highlighted** items in below script will be changed according to your requirements

```
#!/bin/bash
```

#JENKINS ADMIN USER CREATION

```
# Install python3-pip (Prerequisite)  
sudo apt install python3-pip -y  
  
# Install Dev Packages of Python (Prerequisite)  
sudo apt install -y build-essential libssl-dev libffi-dev python3-dev  
  
url=<Jenkins URL>  
password=$(sudo cat /var/lib/jenkins/secrets/initialAdminPassword)
```

```
# NEW ADMIN CREDENTIALS URL ENCODED USING PYTHON  
  
username=$(python3 -c "import urllib.parse;print  
(urllib.parse.quote(input(), safe=''))" <<< "user")  
  
new_password=$(python3 -c "import urllib.parse;print  
(urllib.parse.quote(input(), safe=''))" <<< "password")
```

```
fullname=$(python3 -c "import urllib.parse;print(urllib.parse.quote(input(), safe=''))" <<< "full name")
email=$(python3 -c "import urllib.parse;print (urllib.parse.quote(input(), safe=''))" <<< "hello@world.com")

# GET THE CRUMB AND COOKIE
cookie_jar="$(mktemp)"
full_crumb=$(curl -u "admin:$password" --cookie-jar "$cookie_jar" $url/crumbIssuer/api/xml?xpath=concat\("//crumbRequestField,%22:%22,//crumb\)")
arr_crumb=(${full_crumb//:/ })
only_crumb=$(echo ${arr_crumb[1]})

# MAKE THE REQUEST TO CREATE AN ADMIN USER
curl -X POST -u "admin:$password" $url/setupWizard/createAdminUser \
-H "Connection: keep-alive" \
-H "Accept: application/json, text/javascript" \
-H "X-Requested-With: XMLHttpRequest" \
-H "$full_crumb" \
-H "Content-Type: application/x-www-form-urlencoded" \
--cookie $cookie_jar \
--data-raw
"username=$username&password1=$new_password&password2=$new_password&fullname=$fullname&email=$email&Jenkins-Crumb=$only_crumb&json=%7B%22username%22%3A%20%22$username%22%2C%20%22password1%22%3A%20%22$new_password%22%2C%20%22%24redact%22%3A%20%5B%22password1%22%2C%20%22password2%22%5D%2C%20%22password2%22%3A%20%22$new_password%22%2C%20%22fullname%22%3A%20%22$fullname%22%2C%20%22email%22%3A%20%22$email%22%2C%20%22Jenkins-
```

Crumb%22%3A%20%22\$only_crumb%22%7D&core%3Aapply=&Submit
=Save&json=%7B%22username%22%3A%20%22\$username%22%2C%20
%22password1%22%3A%20%22\$new_password%22%2C%20%22%24re
dact%22%3A%20%5B%22password1%22%2C%20%22password2%22%5
D%2C%20%22password2%22%3A%20%22\$new_password%22%2C%20
%22fullname%22%3A%20%22\$fullname%22%2C%20%22email%22%3A
%20%22\$email%22%2C%20%22Jenkins-
Crumb%22%3A%20%22\$only_crumb%22%7D"

- Jenkins install recommended plugin script is as below:

```
#!/bin/bash
```

```
#INSTALLATION OF RECOMMENDED PLUGINS IN JENKINS
```

```
url=<jenkins URL>
```

```
#ENTER THE CREDENTIALS HIGHLIGHTED IN THE ABOVE CODE
```

```
user=user
```

```
password=password
```

```
cookie_jar="$(mktemp)"  
full_crumb=$(curl -u "$user:$password" --cookie-jar "$cookie_jar"  
$url/crumbIssuer/api/xml?xpath=concat\("//crumbRequestField,%22%22  
,//crumb\())  
arr_crumb=(${full_crumb//:/ })  
only_crumb=$(echo ${arr_crumb[1]})
```

```
# MAKE THE REQUEST TO DOWNLOAD AND INSTALL REQUIRED  
MODULES
```

```
curl -X POST -u "$user:$password" $url/pluginManager/installPlugins \  
-H 'Connection: keep-alive' \  
-H 'Accept: application/json, text/javascript, */*; q=0.01' \  
-H 'X-Requested-With: XMLHttpRequest' \  
-H "$full_crumb" \  
-H 'Content-Type: application/json' \  
-H 'Accept-Language: en,en-US;q=0.9,it;q=0.8' \  
-H 'User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4369.90 Safari/537.36' \  
-H 'Accept-Encoding: gzip, deflate' \  
-H 'DNT: 1' \  
-H 'Referer: https://jenkins:8080/pluginManager/installPlugins' \  
-H 'Cookie: crumb=$only_crumb'
```

```
--cookie $cookie_jar \
--data-raw "{'dynamicLoad':true,'plugins':[['cloudbuilder','folder','antisamy-markup-formatter','build-timeout','credentials-binding','timestamper','ws-cleanup','ant','gradle','workflow-aggregator','github-branch-source','pipeline-github-lib','pipeline-stage-view','git','ssh-slaves','matrix-auth','pam-auth','ldap','email-ext','mailer'],'Jenkins-Crumb':'$only_crumb'}"
```

- Confirming Jenkins URL script is as below:

```
#!/bin/bash

#CONFIRMING JENKINS URL

url=<your_jenkins_url>

user=user
password=password

url_urlEncoded=$(python3 -c "import urllib.parse;print(urllib.parse.quote(input(), safe=''))" <<< "$url")

cookie_jar="$(mktemp)"
full_crumb=$(curl -u "$user:$password" --cookie-jar "$cookie_jar" "$url/crumbIssuer/api/xml?xpath=concat\\(//crumbRequestField,%22%22,//crumb\\))"
arr_crumb=(${full_crumb//:/ })
only_crumb=$(echo ${arr_crumb[1]})

curl -X POST -u "$user:$password" $url/setupWizard/configureInstance \
-H 'Connection: keep-alive' \
-H 'Accept: application/json, text/javascript, */*; q=0.01' \
-H 'X-Requested-With: XMLHttpRequest' \
-H "$full_crumb" \
-H 'Content-Type: application/x-www-form-urlencoded' \
-H 'Accept-Language: en,en-US;q=0.9,it;q=0.8' \
--cookie $cookie_jar \
```

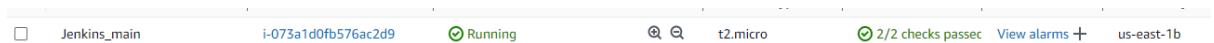
```
--data-raw "rootUrl=$url_urlEncoded%2F&Jenkins-Crumb=$only_crumb&json=%7B%22rootUrl%22%3A%20%22$url_encoded%2F%22%2C%20%22Jenkins-Crumb%22%3A%20%22$only_crumb%22%7D&core%3Aapply=&Submit=Save&json=%7B%22rootUrl%22%3A%20%22$urlEncoded%2F%22%2C%20%22Jenkins-Crumb%22%3A%20%22$only_crumb%22%7D"
```

NOTE: <https://kevin-denotariis.medium.com/download-install-and-setup-jenkins-completely-from-bash-unlock-create-admin-user-and-more-debd3320414a> (Script Reference, You can refer the above link for detail explanation)

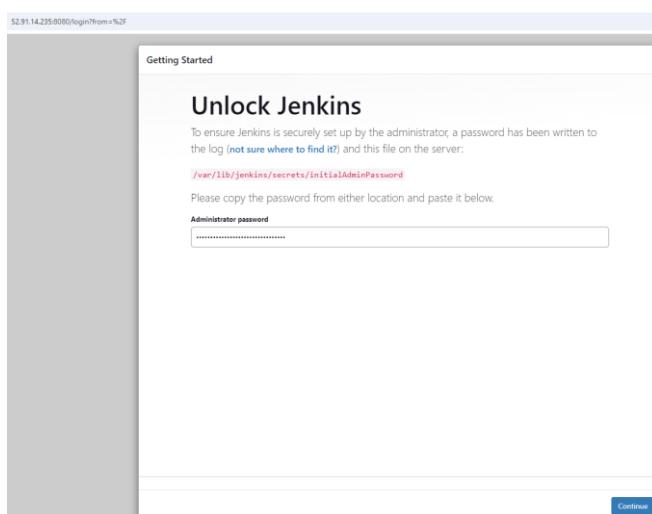
8) How to take backup in Jenkins and Restore.

- To perform the backup of Jenkins and Restore the backup on other machine, we need to install/perform following prerequisites:
 - ❖ Need to have two AWS EC2 instance of Jenkins.
 - First Jenkins will be the primary one where we will create jobs and there will build history and all and we will take the backup of this server which will include jobs, configs and all.
 - Second Jenkins will be the backup one, which will be simple jenkins installed instance with default settings and all and this instance will be used to restore the backup of the first Jenkins server.
 - ❖ Need to have S3 bucket created which will be used to store the jenkins backup and also will be used for restore.
 - ❖ IAM role for S3 full access, this role is required so that we can upload the backup on S3 bucket from first Jenkins instance and also download backup to restore on second Jenkins instance.

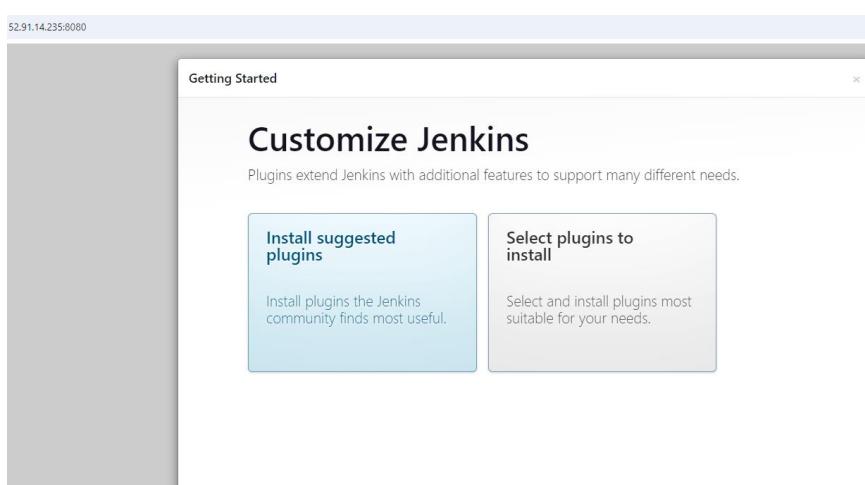
- ❖ We need have aws cli installed using python on both the jenkins instances so that we can run the commands to upload to S3 bucket and download from the S3 bucket.
- Now will setup the first jenkins instance. In our case below is the first server.



- ✓ We will get the default admin password and put on the below screen.
NOTE: Make a note of default password



- ✓ We will install suggested plugins



- ✓ We will skip the next screen and continue as admin

Not secure 52.91.14.235:8080

Getting Started

Create First Admin User

Username

Password

Confirm password

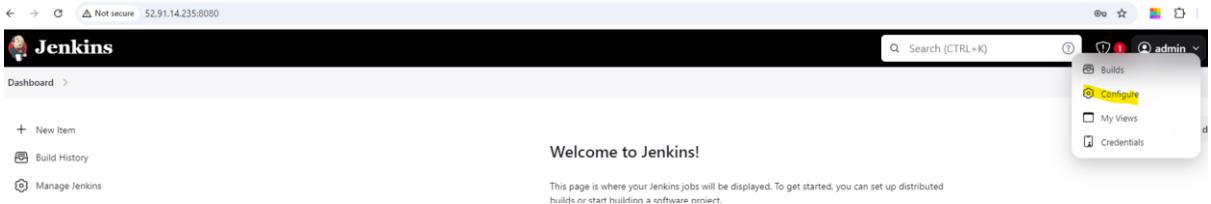
Full name

E-mail address

Jenkins 2.462.1

[Skip and continue as admin](#) [Save and Continue](#)

- ✓ Verify Jenkins URL and start using Jenkins.
- ✓ Now we will change the admin password as per our convenience by dropping down on the admin > Configure



- ✓ Change the password at the location mentioned in the below screenshot and click Apply and Save

Not secure 52.91.14.235:8080/user/admin/configure

Dashboard > admin > Configure

NOTIFICATION URL

Notification URL ?

Password

Password:

Confirm Password:

- ✓ Below Screen will appear



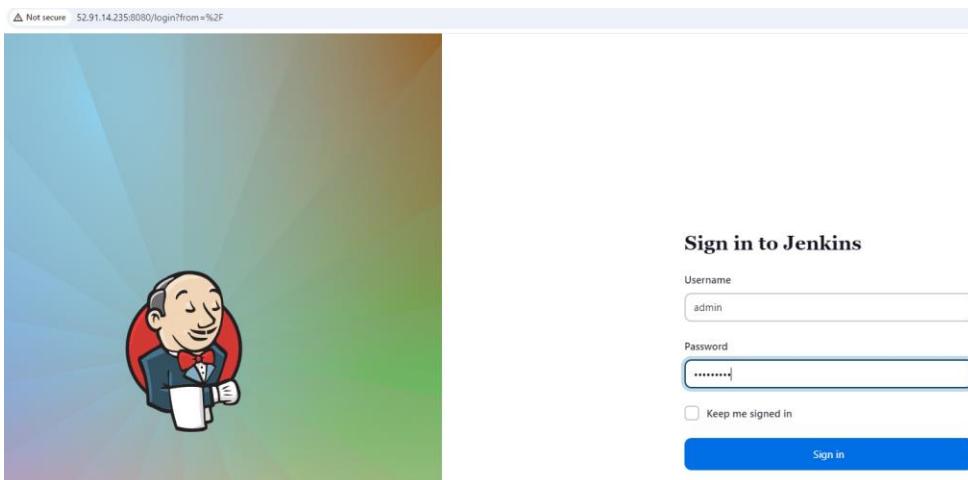
Not secure 52.91.14.235:8080/user/admin/configSubmit

HTTP ERROR 403 No valid crumb was included in the request

URI: /user/admin/configSubmit
STATUS: 403
MESSAGE: No valid crumb was included in the request
SERVLET: Stapler

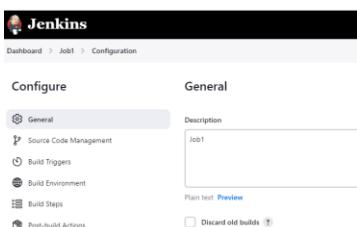
Powered by Jetty:// 10.0.20

- ✓ Now login to admin user with the new credentials and verify the password change



- ✓ You should be able to get in with new credentials.
- ✓ Now we will create two jobs so that we have some data is there which we can backup.
- ✓ Job 1 is freestyle project and below are the configurations.

General > Description > Job1



The image shows the Jenkins configuration screen for a job named "Job1". The "General" tab is selected in the sidebar. The "Description" field contains "Job1". There is also a "Plain text Preview" link and a "Discard old builds" checkbox.

✓ Build Steps > Execute Shell >Command

```
echo "Jenkins-Primary"
```

```
echo "Hello All"
```

```
more /etc/os-release
```

```
pwd
```

```
date
```

The screenshot shows the Jenkins configuration interface for a job named 'Job1'. The left sidebar has 'Build Steps' selected. Under 'Build Steps', there is a section titled 'Execute shell' containing the following command:

```
echo "Jenkins-Primary"  
echo "Hello All"  
more /etc/os-release  
pwd  
date
```

At the bottom right of the configuration page are 'Save' and 'Apply' buttons.

✓ Build Now and build is successful.

The screenshot shows the Jenkins status page for 'Job1'. The top navigation bar shows 'Status' is green. The main area displays the following information:

- Job1**
- Permalinks**:
 - Last build (#1), 1 min 15 sec ago
 - Last stable build (#1), 1 min 15 sec ago
 - Last successful build (#1), 1 min 15 sec ago
 - Last completed build (#1), 1 min 15 sec ago
- Build History**: A table showing one entry: 'Jan 3, 2014, 8:31 PM'.
- Actions**: Buttons for 'Atom feed for all' and 'Atom feed for failure'.

- ✓ console output shows the below output.

The screenshot shows the Jenkins interface for a job named 'Job1' with build '#1'. The 'Console Output' tab is selected. The output window displays the following log:

```

Started by user admin
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Job1
[Job1] $ /bin/sh -xe /tmp/jenkins279651957544497642.sh
+ echo Jenkins-Primary
Jenkins-Primary
+ echo Hello All
Hello All
+ more /etc/os-release
::::::::::::::::::
/etc/os-release
::::::::::::::::::
PRETTY_NAME="Ubuntu 24.04.1 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04.1 LTS (Noble Numbat)"
VERSION_CODENAME=noble
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=noble
LOGO=ubuntu-logo
+ pwd
/var/lib/jenkins/workspace/Job1
+ date
Tue Sep 3 20:31:03 UTC 2024
Finished: SUCCESS

```

- ✓ Job 2 is pipeline project and below are the configurations.

General > Description > Job2

The screenshot shows the Jenkins interface for a pipeline project named 'Job2'. The 'General' tab is selected. The 'Description' field contains the text 'Job2'.

Configure	General
General	Description Job2
Advanced Project Options	
Pipeline	

Under Pipeline > Pipeline script > Script

```

pipeline {
    agent any
    stages {
        stage ("cleaningup workspace") {
            steps {

```

```

        cleanWs()
    }
}

stage ("checkout scm code") {
    steps {
        checkout scmGit(branches: [[name: '/main']], extensions: [],
userRemoteConfigs: [[url:
'https://github.com/sourabh913/Devops_Notes_Assignments.git']] )
    }
}
}
}

```

Pipeline

Definition

Pipeline script

Script ?

```

1 pipeline {
2     agent any
3     stages {
4         stage ("cleaningup workspace") {
5             steps {
6                 cleanWs()
7             }
8         }
9         stage ("checkout scm code") {
10            steps {
11                checkout scmGit(branches: [[name: '/main']], extensions: [], userRemoteConfigs: [[url: 'https://github.com/sourabh913/Devops_Notes_Assignments.git']] )
12            }
13        }
14    }
15 }
16

```

Use Groovy Sandbox ?

[Pipeline Syntax](#)

- ✓ Build Now and build is successful.

Dashboard > Job 2 >

Status
Job 2

</> Changes
Job2

▷ Build Now
Permalinks

⚙️ Configure

🗑 Delete Pipeline

⠇ Stages

✍ Rename

ⓘ Pipeline Syntax

💡 Build History trend ▾

Filter... /

��色圆点 #1 | Sep 3, 2024, 8:39 PM

[Atom feed for all](#) [Atom feed for failures](#)

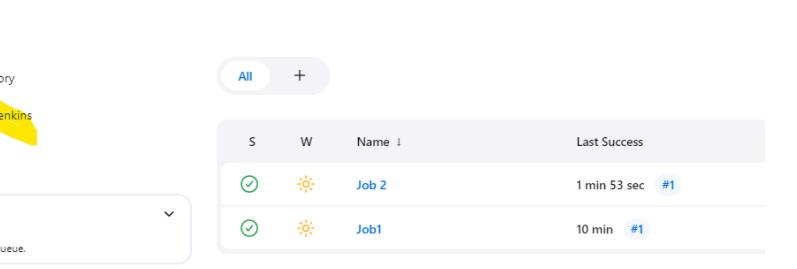
- ✓ console output shows the below output.

```
Dashboard > Job 2 > #1

Console Output
Started by user admin
[Pipeline] Start of Pipeline
[Pipeline] code
Running on Jenkins in /var/lib/jenkins/workspace/Job 2
[Pipeline] {
[Pipeline] stage
[Pipeline] cleanser
[Pipeline] [
[Pipeline] [WS-CLEANUP] Deleting project workspace...
[Pipeline] [WS-CLEANUP] Deferred wipeout is used...
[Pipeline] [WS-CLEANUP] done
[Pipeline] ]
[Pipeline] // stage
[Pipeline] stage
[Pipeline] [
[Pipeline] [WS-CLEANUP] Cleaning workspace: $WORKSPACE ($WORKSPACE)
[Pipeline] checkout
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/sourabh013/Devops_Notes_Assignments.git
git init /var/lib/jenkins/workspace/Job 2 # timeout=10
Fetching upstream changes from https://github.com/sourabh013/Devops_Notes_Assignments.git
git remote add origin https://github.com/sourabh013/Devops_Notes_Assignments.git
git fetch --tags --force --progress -- https://github.com/sourabh013/Devops_Notes_Assignments.git +refs/heads/*:refs/remotes/origin/* # timeout=10
git config remote.origin.url https://github.com/sourabh013/Devops_Notes_Assignments.git # timeout=10
git config --add remote.origin.fetch refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
git rebase refs/remotes/origin/main '{commit}' # timeout=10
Checking out Revision 23352240d42324edc2fcbe87e18303d4ade8 (refs/remotes/origin/main)
git config core.sparsecheckout true
git config core.sparseFile .git/info/sparse-checkout # timeout=10
git -C /var/lib/jenkins/workspace/Job 2 checkout e7e18303d4ade8dc # timeout=10
Commit message: "Update Jenkins_Installation_Configuration_script.sh"
First time build. Skipping changes.
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] end of Pipeline
Finishes: SUCCESS
```

- ✓ For Backup perspective, we will install any available plugin which is not installed by default with recommended plugins. In our case we will install [SonarQube Scanner](#) plugin.

We will go to Manage Jenkins > Plugins

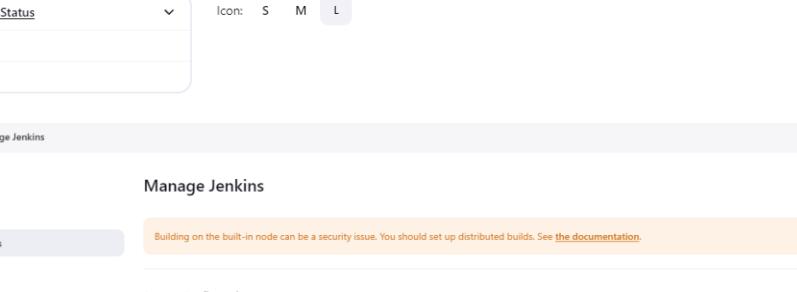


The screenshot shows the Jenkins dashboard. On the left sidebar, there are several links: '+ New Item', 'Build History', 'Manage Jenkins' (which has a yellow arrow pointing to it), and 'My Views'. Below these are two expandable sections: 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing '1 Idle' and '2 Idle'). The main content area is titled 'All' and displays a table of jobs:

S	W	Name	Last Success
✓	☀	Job 2	1 min 53 sec #1
✓	☀	Job1	10 min #1

Below the table, there's a 'Build Executor Status' section with icons for S, M, and L.

At the bottom of the page, the breadcrumb navigation shows 'Dashboard > Manage Jenkins'. The 'Manage Jenkins' link is also highlighted with a yellow arrow in this section.



The 'Manage Jenkins' page has a header 'Manage Jenkins' and a warning message: 'Building on the built-in node can be a security issue. You should set up distributed builds. See [the documentation](#).'. The page is divided into several sections:

- System Configuration**: Includes 'Configure global settings and paths.' and a gear icon.
- Tools**: Includes 'Configure tools, their locations and automatic installers.' and a wrench icon.
- Plugins**: Includes 'Add, remove, disable or enable plugins that can extend the functionality of Jenkins.' and a plug icon.

On the left, there are sections for 'Build Queue' (empty) and 'Build Executor Status' (1 Idle, 2 Idle).

Under Available plugins > Select SonarQubeScanner > Install

The screenshot shows the Jenkins 'Plugins' page. On the left, there's a sidebar with links for 'Updates', 'Available plugins' (which is highlighted in yellow), 'Installed plugins', 'Advanced settings', and 'Download progress'. The main area has a search bar at the top with 'sonar' typed in. Below it, there's a table with two rows. The first row is for 'SonarQube Scanner 2.17.2', which is checked for installation. It includes a 'Released' timestamp of '6 mo 17 days ago'. The second row is for 'Sonar Quality Gates 315.v1f12b_e81a_3a_4', which is not checked. It includes a timestamp of '7 days 22 hr ago'. At the bottom of the table, there are tabs for 'Library plugins (for use by other plugins)', 'analysis', and 'Other Post-Build Actions'.

- ✓ Plugin installed successfully.

The screenshot shows the Jenkins 'Plugins' page. The sidebar on the left is identical to the previous one. The main area lists numerous Jenkins plugins, each with a status indicator (green checkmark) and the word 'Success'. The listed plugins include Gson API, Git client, Git, GitHub, GitHub Branch Source, Pipeline: GitHub Groovy Libraries, Pipeline Graph Analysis, Metrics, Pipeline Graph View, Git, EDDSA API, Trilead API, SSH Build Agents, Matrix Authorization Strategy, PAM Authentication, LDAP, Email Extension, Mailer, Theme Manager, Dark Theme, Loading plugin extensions, SonarQube Scanner, and Loading plugin extensions.

- Now we will create S3 Bucket where we need to upload our backup
- ✓ On the AWS Console, go to console and search S3 and click on it.

The screenshot shows the AWS EC2 console. The top navigation bar has a back arrow, forward arrow, and a search icon. The URL is 'us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:v=3;\$case=tags:tru'. The sidebar on the left includes links for 'EC2 Dashboard', 'EC2 Global View', 'Events', 'Console-to-Code Preview', and a preview link. The main area has a search bar with 'S3' typed in. To the right, there's a 'Search results for 'S3'' section. This section shows a list of services: 'Services (8)', 'Features (39)', 'Resources New', and 'Documentation (26,966)'. On the far right, there's a large card for 'S3' with the subtext 'Scalable Storage in the Cloud'.

✓ Now Click Create bucket



- ✓ Now under general configuration, keep everything as default and just give a unique name to S3 bucket and click create bucket. In our case, bucket name is **jenkinss3backup**

Create bucket [Info](#)

Buckets are containers for data stored in S3.

General configuration

AWS Region: US East (N. Virginia) us-east-1

Bucket type: [Info](#)

General purpose
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

Directory
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name: [Info](#) Jenkins3backup

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.

[Choose bucket](#)

Format: s3://bucket/prefix

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its objects. If you have other buckets and want to turn on an all-public access setting for them without changing any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within it, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Block all public access
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

Block public access to buckets and objects granted through new access control lists (ACLs)
S3 will ignore all ACLs that grant public access to buckets and objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

Block public access to buckets and objects granted through any access control lists (ACLs)
S3 will ignore all ACLs that grant public access to buckets and objects.

Block public access to buckets and objects granted through new public bucket or access point policies
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

Block public and cross-account access to buckets and objects through any public bucket or access point policies
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

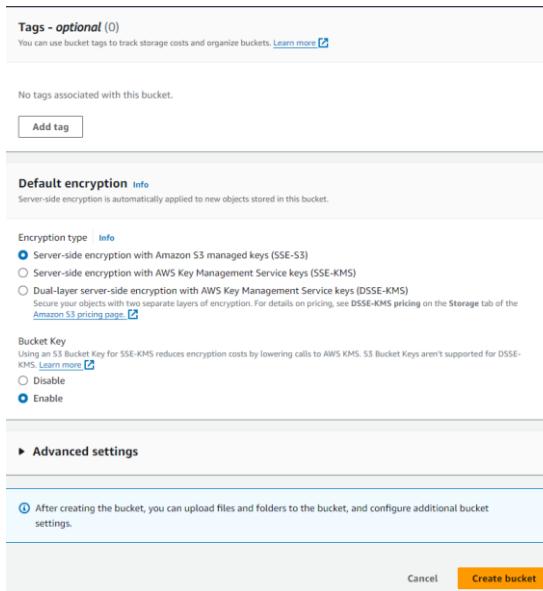
Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

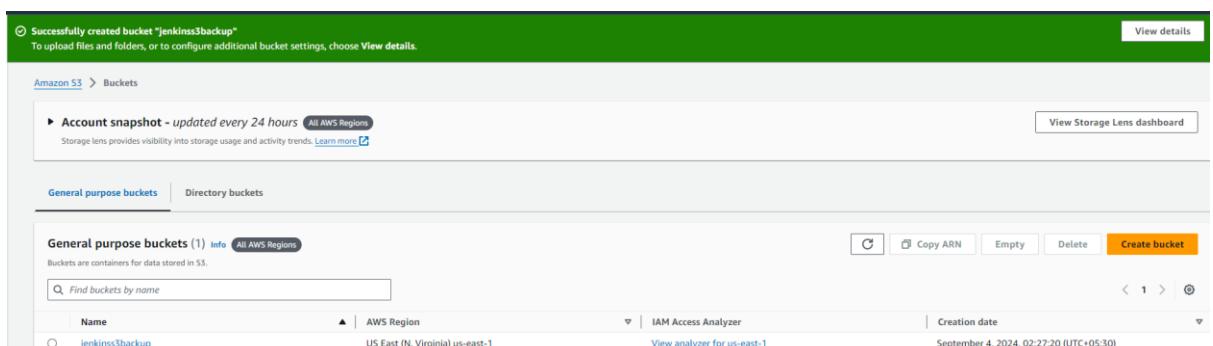
Bucket Versioning:

Disable

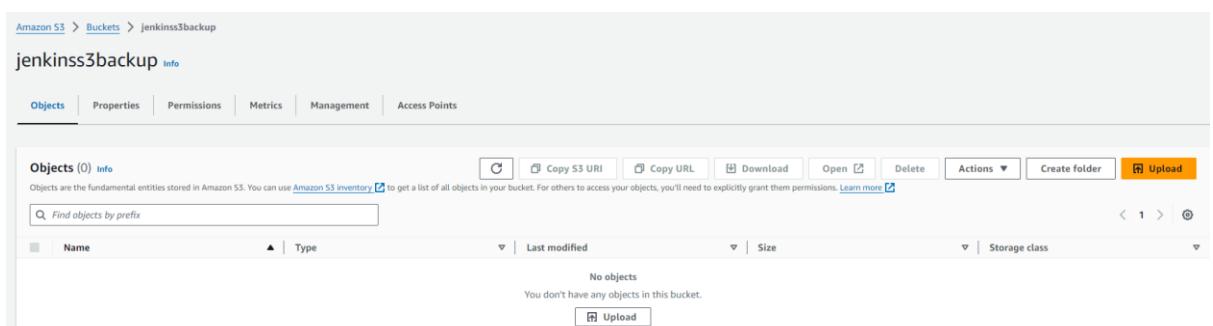
Enable



- ✓ Bucket is created successfully.

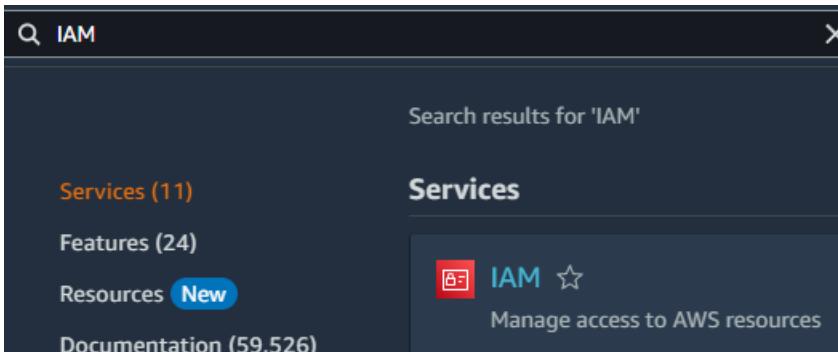


- ✓ Bucket is blank right now.

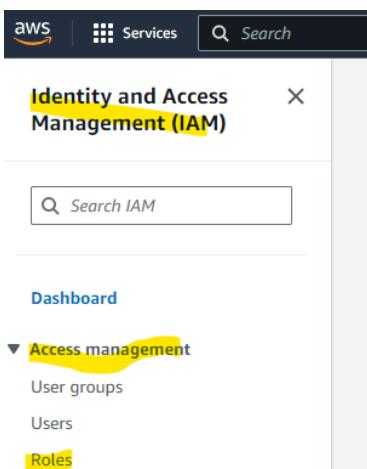


- Now we will create the IAM Role for S3 full access

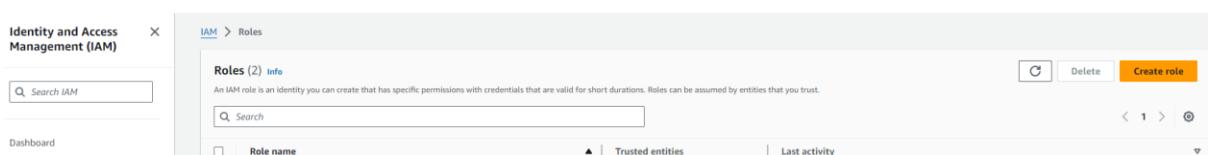
- ✓ On the AWS Console, go to console and search IAM and click on it.



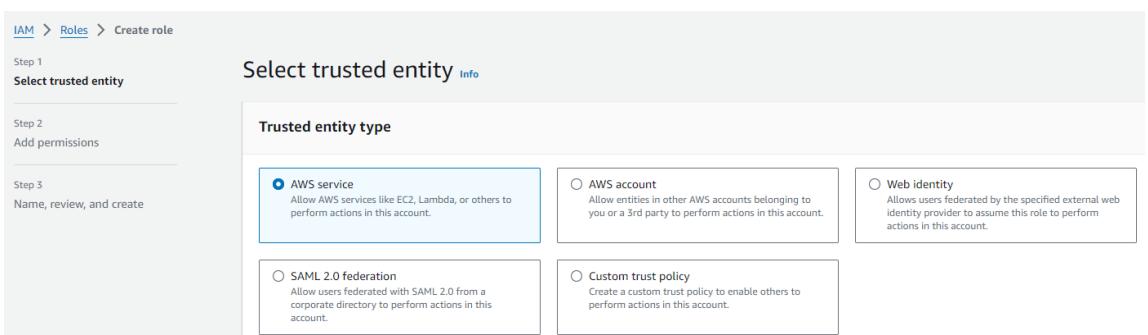
- ✓ Click on Roles under Access Management.



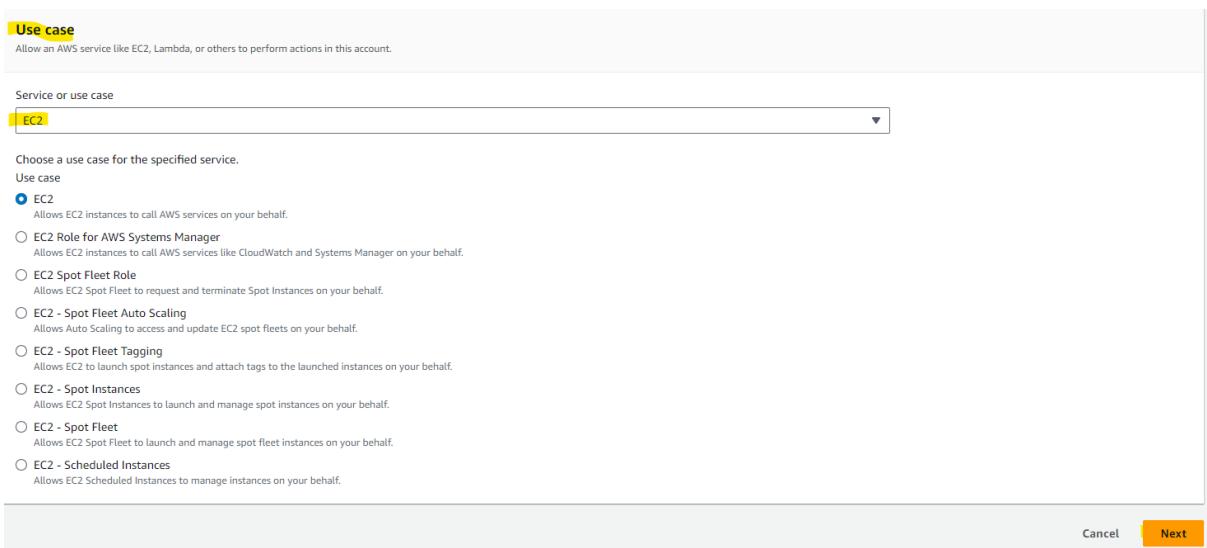
- ✓ Now Click Create Role.



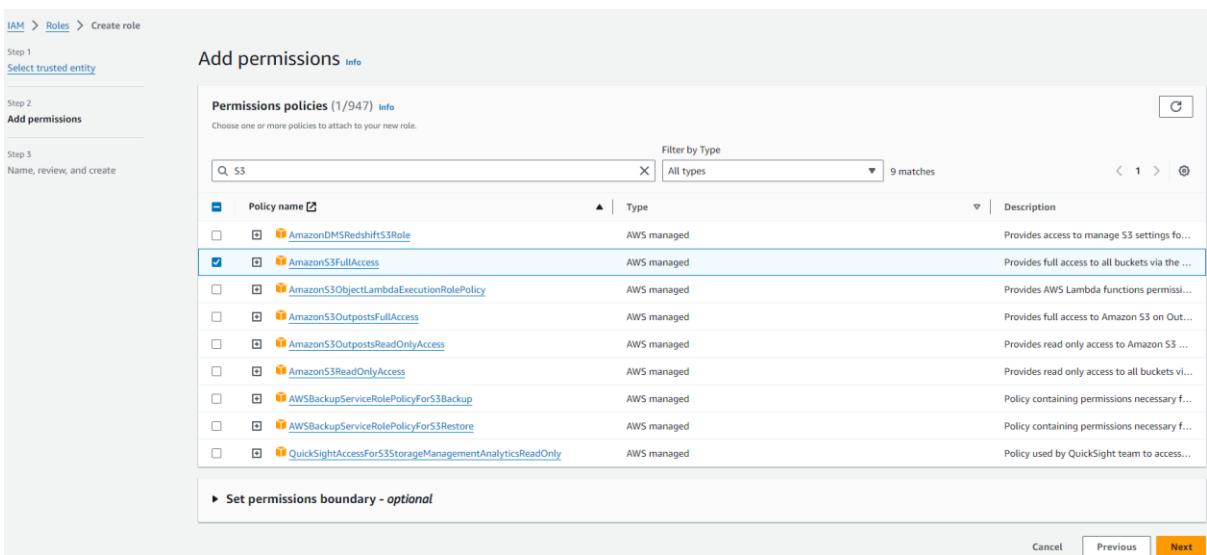
- ✓ Under Select Trusted Entity > Aws Service



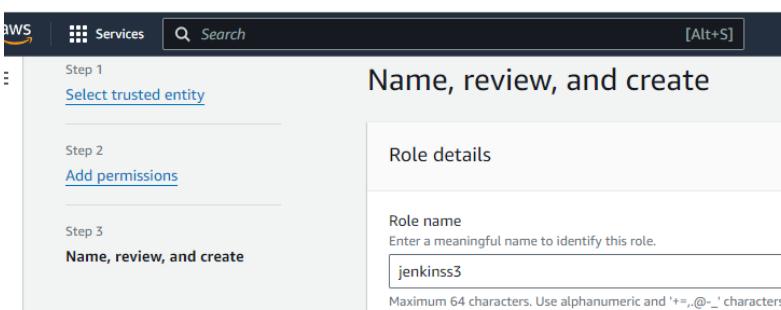
- ✓ Under Use Case > Drop down to EC2 and click Next



- ✓ Add Permissions > Search for S3 > AmazonS3FullAccess and click Next



- ✓ Give the Role Name. In our case, It will be **jenkinss3** and click create role.



```

1- {
2-     "Version": "2012-10-17",
3-     "Statement": [
4-         {
5-             "Effect": "Allow",
6-             "Action": [
7-                 "sts:AssumeRole"
8-             ],
9-             "Principal": [
10-                 "Service": [
11-                     "ec2.amazonaws.com"
12-                 ]
13-             ]
14-         }
15-     ]
16- }

```

Step 2: Add permissions

Permissions policy summary

Policy name	Type	Attached as
AmazonS3FullAccess	AWS managed	Permissions policy

Step 3: Add tags

Add tags - optional Info
Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

Add new tag
You can add up to 50 more tags.

Cancel Previous Create role

- ✓ Role is created.

Role jenkinss3 created.

IAM > Roles

Roles (3) 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 activity
AWSRoleForSupport	AWS Service: support (Service-Linked)	-
AWSRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service-Linked)	-
jenkinss3	AWS Service: ec2	-

- Now we will take the backup of first Jenkins instance.
- ✓ First of all, we need to stop the Jenkins service by using the below command

systemctl stop jenkins

```
root@ip-172-31-88-11:/home/ubuntu# systemctl stop jenkins
root@ip-172-31-88-11:/home/ubuntu#
```

- ✓ Now, the most important folder which has all the configs, secrets, jobs and builds is the home path of Jenkins /var/lib/jenkins

cd /var/lib/jenkins

ls -ltr

```
root@ip-172-31-88-11:/home/ubuntu# cd /var/lib/jenkins/
root@ip-172-31-88-11:/var/lib/jenkins#
root@ip-172-31-88-11:/var/lib/jenkins#
root@ip-172-31-88-11:/var/lib/jenkins#
root@ip-172-31-88-11:/var/lib/jenkins#
root@ip-172-31-88-11:/var/lib/jenkins#
root@ip-172-31-88-11:/var/lib/jenkins# ls -ltr
total 96
-rw-r--r-- 1 jenkins jenkins 64 Sep 3 20:08 secret.key
-rw-r--r-- 1 jenkins jenkins 0 Sep 3 20:08 secret.key.not-so-secret
-rw-r--r-- 1 jenkins jenkins 156 Sep 3 20:08 hudson.model.UpdateCenter.xml
-rw-r--r-- 1 jenkins jenkins 171 Sep 3 20:08 jenkins.telemetry.Correlator.xml
drwxr-xr-x 2 jenkins jenkins 4096 Sep 3 20:08 userContent
-rw-r--r-- 1 jenkins jenkins 1037 Sep 3 20:08 nodeMonitors.xml
drwxr-xr-x 3 jenkins jenkins 4096 Sep 3 20:08 users
-rw-r--r-- 1 jenkins jenkins 1660 Sep 3 20:08 config.xml
-rw-r--r-- 1 jenkins jenkins 1680 Sep 3 20:12 identity.key.enc
-rw-r--r-- 1 jenkins jenkins 370 Sep 3 20:12 hudson.plugins.git.GitTool.xml
-rw-r--r-- 1 jenkins jenkins 182 Sep 3 20:13 jenkins.model.JenkinsLocationConfiguration.xml
-rw-r--r-- 1 jenkins jenkins 7 Sep 3 20:13 jenkins.install.UpgradeWizard.state
-rw-r--r-- 1 jenkins jenkins 7 Sep 3 20:13 jenkins.install.InstallUtil.lastExecVersion
drwxr-xr-x 4 jenkins jenkins 4096 Sep 3 20:34 jobs
-rw-r--r-- 1 jenkins jenkins 504 Sep 3 20:39 org.jenkinsci.plugins.resourcedisposer.AsyncResourceDisposer.xml
drwxr-xr-x 4 jenkins jenkins 4096 Sep 3 20:39 workspace
-rw-r--r-- 1 jenkins jenkins 46 Sep 3 20:39 org.jenkinsci.plugins.workflow.flow.FlowExecutionList.xml
drwx----- 2 jenkins jenkins 4096 Sep 3 20:40 secrets
drwxr-xr-x 3 jenkins jenkins 4096 Sep 3 20:42 logs
drwxr-xr-x 91 jenkins jenkins 12288 Sep 3 20:46 plugins
drwxr-xr-x 2 jenkins jenkins 4096 Sep 3 20:46 updates
-rw-r--r-- 1 jenkins jenkins 313 Sep 3 20:46 hudson.plugins.sonar.SonarGlobalConfiguration.xml
-rw-r--r-- 1 jenkins jenkins 258 Sep 3 21:13 queue.xml
root@ip-172-31-88-11:/var/lib/jenkins#
```

- ✓ Now we compress all the files & directories under /var/lib/jenkins by creating a tar file.

tar -zcvf <file-name> /var/lib/jenkins/

In our case, it's as below

tar -zcvf jenkins-backup.tar.gz /var/lib/jenkins/

```
root@ip-172-31-88-11:/home/ubuntu# tar -zcvf jenkins-backup.tar.gz /var/lib/jenkins/
tar: Removing leading `/' from member names
/var/lib/jenkins/
/var/lib/jenkins/org.jenkinsci.plugins.resourcedisposer.AsyncResourceDisposer.xml
/var/lib/jenkins/secrets/
/var/lib/jenkins/secrets/hudson.model.Job.serverCookie
/var/lib/jenkins/secrets/jenkins.model.Jenkins.crumbsSalt
/var/lib/jenkins/secrets/org.jenkinsci.plugins.workflow.log.ConsoleAnnotators.consoleAnnotator
/var/lib/jenkins/secrets/initialAdminPassword
/var/lib/jenkins/secrets/master.key
/var/lib/jenkins/secrets/org.jenkinsci.main.modules.instance_identity.InstanceIdentity.KEY
/var/lib/jenkins/secrets/hudson.console.AnnotatedLargeText.consoleAnnotator
/var/lib/jenkins/secrets/hudson.util.Secret
/var/lib/jenkins/secrets/hudson.console.ConsoleNote.MAC
/var/lib/jenkins/hudson.plugins.sonar.SonarGlobalConfiguration.xml
/var/lib/jenkins/plugins/
/var/lib/jenkins/plugins/workflow-support/
/var/lib/jenkins/plugins/workflow-support/WEB-INF/
/var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/
/var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/jboss-marshalling-river-2.1.3.Final.jar
/var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/jboss-marshalling-2.1.3.Final.jar
/var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/workflow-support.jar
/var/lib/jenkins/plugins/workflow-support/WEB-INF/licenses.xml
/var/lib/jenkins/plugins/workflow-support/timestamp2
/var/lib/jenkins/plugins/workflow-support/META-INF/
/var/lib/jenkins/plugins/workflow-support/META-INF/MANIFEST.MF
/var/lib/jenkins/plugins/workflow-support/META-INF/maven/
/var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/
/var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/workflow-support/
/var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/workflow-support/pom.properties
/var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/workflow-support/pom.xml
/var/lib/jenkins/plugins/workflow-multibranch.jpi
/var/lib/jenkins/plugins/caffeine-api.jpi
/var/lib/jenkins/plugins/cloudbees-folder.ini
```

- ✓ All the files & directories under /var/lib/jenkins are compressed and tar file is created **jenkins-backup.tar.gz**

```
root@ip-172-31-88-11:/home/ubuntu# ls -ltr
total 301648
drwxr-xr-x 4 root root      4096 Sep  3 19:10 Devops_Notes_Assignments
-rw-r--r-- 1 root root      767 Sep  3 19:19 jenkins.sh
-rw-r--r-- 1 root root 308874325 Sep  3 21:21 jenkins-backup.tar.gz
root@ip-172-31-88-11:/home/ubuntu#
```

- Now we will install AWS CLI

- ✓ Get the zip file of aws cli using the below command.

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
"awscliv2.zip"
```

```
root@ip-172-31-88-11:/home/ubuntu# curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
% Total    % Received % Xferd  Average Speed   Time     Time     Current
          Dload  Upload Total   Spent   Left  Speed
100 58.0M  100 58.0M    0     0  107M    0 --:--:-- --:--:-- 107M
```

- ✓ Now we will unzip the file, but before that we need to install **unzip** using the below command.

apt install unzip -y

```
root@ip-172-31-88-11:/home/ubuntu# apt install unzip -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  zip
The following NEW packages will be installed:
  unzip
0 upgraded, 1 newly installed, 0 to remove and 2 not upgraded.
Need to get 175 kB of additional disk space will be used.
After this operation, 384 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 unzip amd64 6.0-28ubuntu4 [175 kB]
Fetched 175 kB in 0s (11.1 MB/s)
Selecting previously unselected package unzip.
(Reading database ... 121174 files and directories currently installed.)
Preparing to unpack .../unzip_6.0-28ubuntu4_amd64.deb ...
Unpacking unzip (6.0-28ubuntu4) ...
Setting up unzip (6.0-28ubuntu4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
  6.8.0-1012-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.8.0-1014-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...
```

- ✓ Now we will unzip the file **awscliv2.zip** using the below command

unzip awscliv2.zip

```
root@ip-172-31-88-11:/home/ubuntu# unzip awscliv2.zip
Archive: awscliv2.zip
  creating: aws/
  creating: aws/dist/
  inflating: aws/README.md
  inflating: aws/THIRD_PARTY_LICENSES
  inflating: aws/install
  creating: aws/dist/awscli/
  creating: aws/dist/cryptography/
  creating: aws/dist/docutils/
  creating: aws/dist/lib-dynload/
  inflating: aws/dist/aws
  inflating: aws/dist/aws_completer
  inflating: aws/dist/libpython3.11.so.1.0
  inflating: aws/dist/_awscrt.abi3.so
  inflating: aws/dist/_cffi_backend.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/_ruamel_yaml.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/libz.so.1
  inflating: aws/dist/liblzma.so.0
  inflating: aws/dist/libbz2.so.1
  inflating: aws/dist/libffi.so.5
  inflating: aws/dist/libsqlite3.so.0
  inflating: aws/dist/base_library.zip
  inflating: aws/dist/lib-dynload/_pickle.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_hashlib.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_sha3.cpython-311-x86_64-linux-gnu.so
```

- ✓ Once the unzip of the file is done, we need to run the below command to install AWS CLI

./aws/install

```
root@ip-172-31-88-11:/home/ubuntu#
root@ip-172-31-88-11:/home/ubuntu# ./aws/install
You can now run: /usr/local/bin/aws --version
root@ip-172-31-88-11:/home/ubuntu#
```

- ✓ We can verify that AWS CLI installed by typing below command.

aws

```
root@ip-172-31-88-11:/home/ubuntu# aws
usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

  aws help
  aws <command> help
  aws <command> <subcommand> help

aws: error: the following arguments are required: command
```

- Now we will upload the jenkins-backup.tar.gz using the below command via AWS CLI.

```
aws s3 cp <file-name> s3://<bucket-name>/<file-name>
```

In our case

```
aws s3 cp jenkins-backup.tar.gz s3://jenkinss3backup/jenkins-
backup.tar.gz
```

```
root@ip-172-31-88-11:/home/ubuntu# aws s3 cp jenkins-backup.tar.gz s3://jenkinss3backup/jenkins-backup.tar.gz
upload failed: ./jenkins-backup.tar.gz to s3://jenkinss3backup/jenkins-backup.tar.gz Unable to locate credentials
```

- Upload failed due to credentials were not located.
- Now we will attach the IAM Role to Jenkins Primary EC2 Instance
- We will go to the below path and modify IAM Role

The screenshot shows the AWS EC2 Instances page. It lists two instances: 'Jenkins_main' (Instance ID: i-073a1d0fb576ac2d9) and 'Jenkins_backup' (Instance ID: i-0a6522b7e01196ba0). Both instances are running, t2.micro type, and have 2/2 checks passed. The 'Actions' menu for the Jenkins_main instance is open, showing options like Connect, View details, Manage instance state, Instance settings, Networking, Security (selected), Image and templates, and Monitor and troubleshoot.

- We will choose the Role from dropdown and select jenkinss3 and click update IAM role

The screenshot shows the 'Modify IAM role' dialog for the Jenkins_main instance. It asks to attach an IAM role to the instance. A dropdown menu for 'Choose IAM role' is open, showing 'jenkinss3' as the selected option. A warning message in a yellow box says: 'The instance will be removed. Are you sure?'. At the bottom are 'Cancel' and 'Update IAM role' buttons.

- Now we will again upload the jenkins-backup.tar.gz using the below command via AWS CLI.

```
aws s3 cp jenkins-backup.tar.gz s3://jenkinss3backup/jenkins-backup.tar.gz
```

Now file got uploaded successfully.

```
root@ip-172-31-88-11:/home/ubuntu# aws s3 cp jenkins-backup.tar.gz s3://jenkinss3backup/jenkins-backup.tar.gz
upload: ./jenkins-backup.tar.gz to s3://jenkinss3backup/jenkins-backup.tar.gz
root@ip-172-31-88-11:/home/ubuntu#
```

- Verified on S3 bucket also that upload was successful.

Name	Type	Last modified	Size	Storage class
jenkins-backup.tar.gz	gz	September 4, 2024, 03:35:34 (UTC+05:30)	294.6 MB	Standard

- Now will terminate the first Jenkins instance.

Name	Instance ID	Instance state	Instance type	Status check	Action
Jenkins_main	i-073a1d0fb576ac2d9	Running	t2.micro	2/2 checks passed	Terminate (delete) instance

- Now we will go on the Second Jenkins instance for restoring backup of first Jenkins instance.
- We will have install AWS CLI again by following the same method.

Get the zip file of aws cli using the below command.

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
```

```
root@ip-172-31-80-117:/home/ubuntu# curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
% Total    % Received % Xferd  Average Speed   Time   Time     Current
          Dload  Upload Total Spent   Left Speed
100 58.0M  100 58.0M    0      0  106M      0 --:--:-- --:--:-- 106M
root@ip-172-31-80-117:/home/ubuntu#
```

- ✓ Now we will unzip the file, but before that we need to install **unzip** using the below command.

apt install unzip -y

```
root@ip-172-31-88-11:/home/ubuntu# apt install unzip -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  zip
The following NEW packages will be installed:
  unzip
0 upgraded, 1 newly installed, 0 to remove and 2 not upgraded.
Need to get 175 kB of archives.
After this operation, 384 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble amd64 unzip amd64 6.0-28ubuntu4 [175 kB]
Fetched 175 kB in 0s (11.1 MB/s)
Selecting previously unselected package unzip.
(Reading database ... 121174 files and directories currently installed.)
Preparing to unpack .../unzip_6.0-28ubuntu4_amd64.deb ...
Unpacking unzip (6.0-28ubuntu4) ...
Setting up unzip (6.0-28ubuntu4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
  6.8.0-1012-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.8.0-1014-aws.
Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.
Restarting services...
```

- ✓ Now we will unzip the file **awscliv2.zip** using the below command
unzip awscliv2.zip

```
root@ip-172-31-88-11:/home/ubuntu# unzip awscliv2.zip
Archive: awscliv2.zip
  creating: aws/
  creating: aws/dist/
  inflating: aws/README.md
  inflating: aws/THIRD_PARTY_LICENSES
  inflating: aws/install
  creating: aws/dist/awscli/
  creating: aws/dist/cryptography/
  creating: aws/dist/docutils/
  creating: aws/dist/lib-dynload/
  inflating: aws/dist/aws
  inflating: aws/dist/aws_completer
  inflating: aws/dist/libpython3.11.so.1.0
  inflating: aws/dist/_awscrt.abi3.so
  inflating: aws/dist/_cffi_backend.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/_ruamel_yaml.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/libbz.so.1
  inflating: aws/dist/liblzma.so.0
  inflating: aws/dist/libbz2.so.1
  inflating: aws/dist/libffi.so.5
  inflating: aws/dist/libsqlite3.so.0
  inflating: aws/dist/base_library.zip
  inflating: aws/dist/lib-dynload/_pickle.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_hashlib.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_sha3.cpython-311-x86_64-linux-gnu.so
```

- ✓ Once the unzip of the file is done, we need to run the below command to install AWS CLI

`./aws/install`

```
root@ip-172-31-88-11:/home/ubuntu# ./aws/install
You can now run: /usr/local/bin/aws --version
  OR
  source /usr/local/bin/aws-completer.sh
```

- ✓ We can verify that AWS CLI installed by typing below command.

`aws`

```
root@ip-172-31-88-11:/home/ubuntu# aws
usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:
    aws help
    aws <command> help
    aws <command> <subcommand> help
aws: error: the following arguments are required: command
```

- ✓ we need to stop the jenkins service by using the below command

`systemctl stop jenkins`

```
root@ip-172-31-88-11:/home/ubuntu# systemctl stop jenkins
root@ip-172-31-88-11:/home/ubuntu#
```

- ✓ We need to remove the home path /var/lib/jenkins of current jenkins setup

`rm -rf /var/lib/jenkins`

```
root@ip-172-31-80-117:/home/ubuntu# rm -rf /var/lib/jenkins/
root@ip-172-31-80-117:/home/ubuntu#
```

- Now we will download the **jenkins-backup.tar.gz** using the below command via AWS CLI.

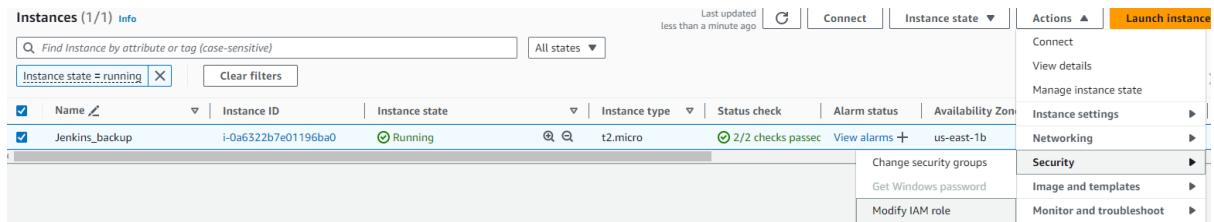
```
aws s3 cp s3://bucketname/filename filename
```

In our case

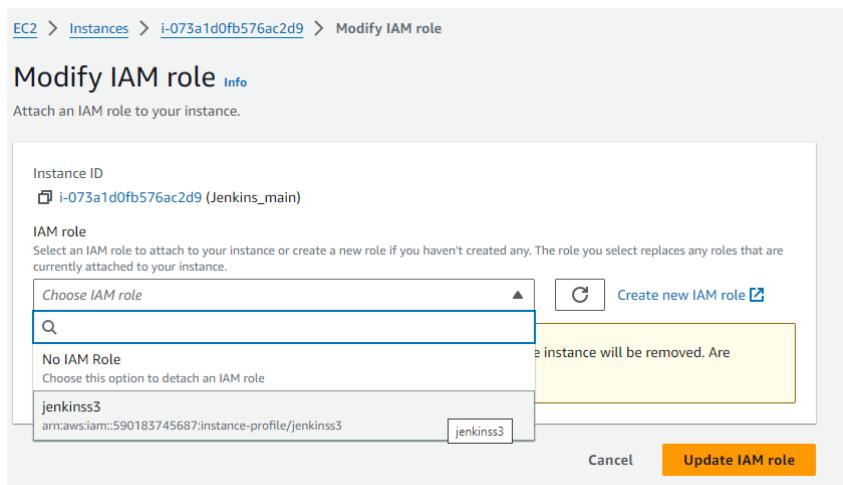
```
aws s3 cp s3://jenkinss3backup/jenkins-backup.tar.gz jenkins-backup.tar.gz
```

```
root@ip-172-31-80-117:/home/ubuntu# aws s3 cp s3://jenkinss3backup/jenkins-backup.tar.gz jenkins-backup.tar.gz
fatal error: Unable to locate credentials
```

- Unable to locate credentials.
- Now we will attach the IAM Role to Jenkins Secondary EC2 Instance
- ✓ We will go to the below path and modify IAM Role



- ✓ We will choose the Role from dropdown and select **jenkinss3** and click update IAM role



- Now we will again download the jenkins-backup.tar.gz using the below command via AWS CLI.

```
aws s3 cp s3://jenkinss3backup/jenkins-backup.tar.gz jenkins-backup.tar.gz
```

File downloaded successfully on the Instance

```
root@ip-172-31-80-117:/home/ubuntu# aws s3 cp s3://jenkinss3backup/jenkins-backup.tar.gz jenkins-backup.tar.gz
download: s3://jenkinss3backup/jenkins-backup.tar.gz to ./jenkins-backup.tar.gz
root@ip-172-31-80-117:/home/ubuntu#
```

- ✓ File is present on the location.

```
root@ip-172-31-80-117:/home/ubuntu# ls -ltr
total 361056
drwxr-xr-x 3 root root      4096 Aug 30 18:21 aws
drwxr-xr-x 4 root root      4096 Sep  3 19:10 Devops_Notes_Assignments
-rw-r--r-- 1 root root 308874325 Sep  3 22:05 jenkins-backup.tar.gz
-rw-r--r-- 1 root root  60830141 Sep  3 22:14 awscliv2.zip
root@ip-172-31-80-117:/home/ubuntu#
```

- ✓ Now will untar the file, so that it creates /etc/var/jenkins directory and copies all the backup of the primary instance.

```
tar -zxvf <file-name> -C /
```

In our case

```
tar -zxvf jenkins-backup.tar.gz -C /
```

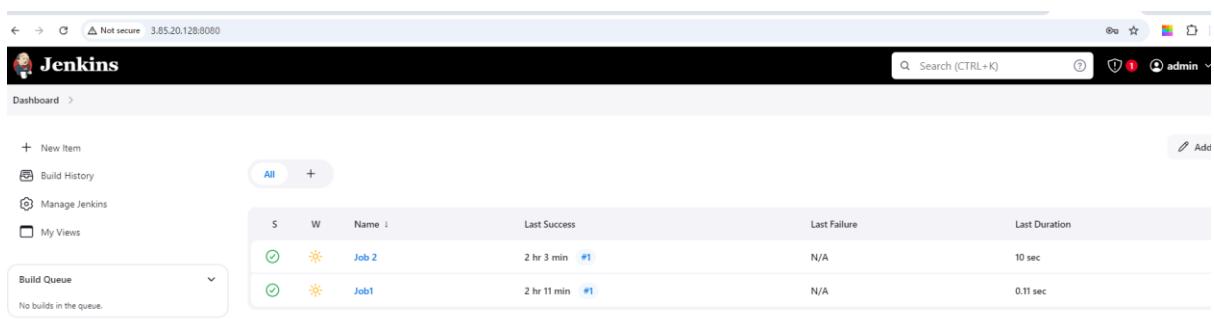
```
root@ip-172-31-80-117:/home/ubuntu# tar -zxvf jenkins-backup.tar.gz -C /
var/lib/jenkins/
var/lib/jenkins/org.jenkinsci.plugins.resourcedisposer.AsyncResourceDisposer.xml
var/lib/jenkins/secrets/
var/lib/jenkins/secrets/hudson.model.Job.serverCookie
var/lib/jenkins/secrets/jenkins.model.Jenkins.crumbsSalt
var/lib/jenkins/secrets/org.jenkinsci.plugins.workflow.log.ConsoleAnnotators.consoleAnnotator
var/lib/jenkins/secrets/initialAdminPassword
var/lib/jenkins/secrets/master.key
var/lib/jenkins/secrets/org.jenkinsci.main.modules.instance_identity.InstanceIdentity.KEY
var/lib/jenkins/secrets/hudson.console.AnnotatedLargeText.consoleAnnotator
var/lib/jenkins/secrets/hudson.util.Secret
var/lib/jenkins/secrets/hudson.console.ConsoleNote.MAC
var/lib/jenkins/hudson.plugins.sonar.SonarGlobalConfiguration.xml
var/lib/jenkins/plugins/
var/lib/jenkins/plugins/workflow-support/
var/lib/jenkins/plugins/workflow-support/WEB-INF/
var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/
var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/jboss-marshalling-river-2.1.3.Final.jar
var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/jboss-marshalling-2.1.3.Final.jar
var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/workflow-support.jar
var/lib/jenkins/plugins/workflow-support/WEB-INF/licenses.xml
var/lib/jenkins/plugins/workflow-support/.timestamp2
var/lib/jenkins/plugins/workflow-support/META-INF/
var/lib/jenkins/plugins/workflow-support/META-INF/MANIFEST.MF
var/lib/jenkins/plugins/workflow-support/META-INF/maven/
var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/
var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/workflow-support/pom.properties
var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/workflow-support/pom.xml
var/lib/jenkins/plugins/workflow-multibranch.jpi
var/lib/jenkins/plugins/caffeine-api.jpi
var/lib/jenkins/plugins/cloudbees-folder.jpi
var/lib/jenkins/plugins/junit.jpi
var/lib/jenkins/plugins/junit5.jpi
```

- ✓ Now we will start jenkins using the below command.

systemctl start jenkins

```
root@ip-172-31-80-117:/home/ubuntu# systemctl start jenkins
root@ip-172-31-80-117:/home/ubuntu#
```

- ✓ Now we will verify, if we are able to login to the Jenkins using the credentials which we set for Jenkins Primary Instance.



- ✓ We are able to login and we also see the jobs created in Jenkins Primary Instance.
- So, Jenkins Backup and Restore have been completed successfully.

9) How to Secure Your Jenkins Server with HTTPS.

- Before starting, we need to have the following prerequisites
 - A Jenkins Server Installed on EC2 Instance
 - Open SSL Installed on the same EC2 Instance
- A EC2 Instance is ready with Jenkins Server installed on it using the below shell script



- Open SSL is installed using the below command.

```
apt install openssl -y
```

```
root@ip-172-31-44-33:/home/ubuntu# apt install openssl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
openssl is already the newest version (3.0.13-0ubuntu3.4).
openssl set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
```

- Generate SSL Certificate

➤ The first step is to generate a self-signed SSL certificate using OpenSSL. Execute the following command: and follow the steps in below screenshot

```
openssl req -newkey rsa:2048 -nodes -keyout key.pem -x509 -days 365 -out certificate.pem
```

```
root@ip-172-31-44-33:/home/ubuntu# openssl req -newkey rsa:2048 -nodes -keyout key.pem -x509 -days 365 -out certificate.pem
-----+
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:IN
State or Province Name (full name) [Some-State]:Maharashtra
Locality Name (eg, city) []:Pune
Organization Name (eg, company) [Internet Widgits Pty Ltd]:TU Ltd
Organizational Unit Name (eg, section) []:Devops
Common Name (e.g. server FQDN or YOUR name) []:ec2-18-232-138-56.compute-1.amazonaws.com
Email Address []:ajda.com
root@ip-172-31-44-33:/home/ubuntu#
```

- Create PKCS12 File

➤ Now we will create a PKCS12 file from the generated key and certificate: Set an export password when prompted, verify it.

```
openssl pkcs12 -inkey key.pem -in certificate.pem -export -out certificate.p12
```

```
root@ip-172-31-44-33:/home/ubuntu# openssl pkcs12 -inkey key.pem -in certificate.pem -export -out certificate.p12
Enter Export Password:
Verifying - Enter Export Password:
```

- **Import Certificate to Keystore**

- Now Import the PKCS12 file into a Java keystore (JKS) using the keytool command:

You'll need to enter the destination keystore password and the source keystore password.

```
keytool -importkeystore -srckeystore ./certificate.p12 -srcstoretype pkcs12 -destkeystore jenkinsbuilddevops.jks -deststoretype JKS
```

```
root@ip-172-31-16-10:/home/ubuntu# keytool -importkeystore -srckeystore ./certificate.p12 -srcstoretype pkcs12 -destkeystore jenkinsbuilddevops.jks -deststoretype JKS
Importing Keystore ./certificate.p12 to jenkinsbuilddevops.jks...
Enter destination keystore password:
Re-enter new password:
Enter source keystore password:
Entry for alias 1 successfully imported.
Import command completed: 1 entries successfully imported, 0 entries failed or cancelled
Warning:
The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard format using "keytool -importkeystore -srckeystore jenkinsbuilddevops.jks -destkeystore jenkinsbuilddevops.jks -deststoretype pkcs12".
root@ip-172-31-16-10:/home/ubuntu#
```

- **Move Keystore to Jenkins Directory**

- Now Move keystore to jenkins directory. Copy the generated keystore file to the Jenkins directory:

```
cp -pav jenkinsbuilddevops.jks /var/lib/jenkins/
```

```
root@ip-172-31-44-33:/home/ubuntu# cp -pav jenkinsbuilddevops.jks /var/lib/jenkins/
'jenkinsbuilddevops.jks' -> '/var/lib/jenkins/jenkinsbuilddevops.jks'
root@ip-172-31-44-33:/home/ubuntu#
```

- Set Jenkins HTTPS Configuration: Modify the Jenkins service configuration to enable HTTPS. Edit the Jenkins service file:

```
vi /usr/lib/systemd/system/jenkins.service
```

```
root@ip-172-31-44-33:/home/ubuntu# vi /usr/lib/systemd/system/jenkins.service
root@ip-172-31-44-33:/home/ubuntu#
```

Change the below line

```
Environment="JENKINS_PORT=8080"
```

```
# add the CAP_NET_BIND_SERVICE capability
# directive below.
Environment="JENKINS_PORT=8080"
```

```
Environment="JENKINS_HTTPS_PORT=8090"
Environment="JENKINS_HTTPS_KEYSTORE=/var/lib/jenkins/jenkinsbuilddevops.jks"
Environment="JENKINS_HTTPS_KEYSTORE_PASSWORD=abc123"
```

```
Environment="JENKINS_HTTPS_PORT=8090"
Environment="JENKINS_HTTPS_KEYSTORE=/var/lib/jenkins/jenkinsbuilddevops.jks"
Environment="JENKINS_HTTPS_KEYSTORE_PASSWORD=abc123"
```

- Reload the systemd daemon and restart the Jenkins service:

```
systemctl daemon-reload
```

```
systemctl restart jenkins
```

```
root@ip-172-31-44-33:/home/ubuntu# systemctl daemon-reload
root@ip-172-31-44-33:/home/ubuntu#
root@ip-172-31-44-33:/home/ubuntu#
root@ip-172-31-44-33:/home/ubuntu#
root@ip-172-31-44-33:/home/ubuntu# systemctl restart jenkins
root@ip-172-31-44-33:/home/ubuntu#
```

- Now we will verify the https connection on the new port for Jenkins i.e. port 8090

