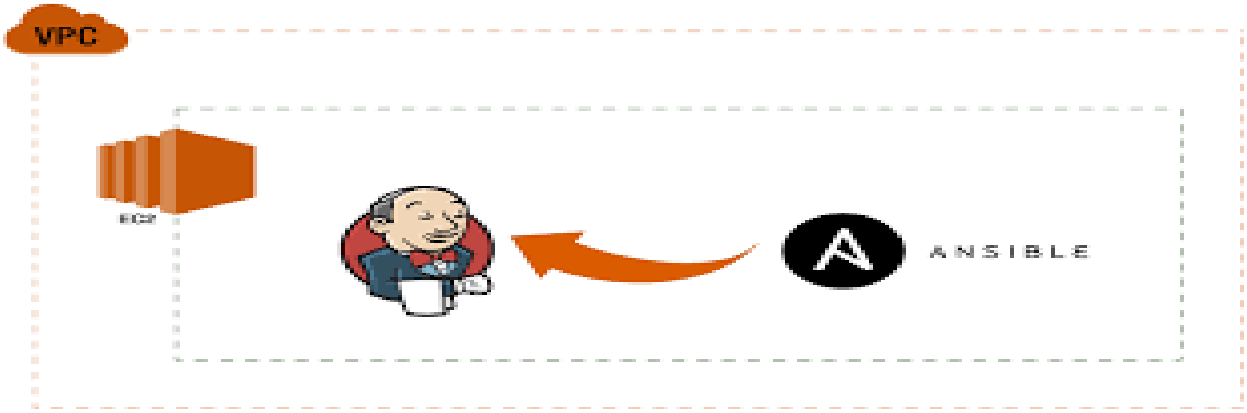
cprime

Automating Jenkins Installation and Configuration Using Ansible



NAME : T MANOJ

EMPID : LYAKE2KHS

Overview

This document provides a step-by-step guide to launching an instance, configuring it as an Ansible master, connecting slave nodes, and automating the installation and configuration of Jenkins. The setup also includes retrieving the initial Jenkins password, modifying the default port using variables, and using a Jinja2 template to update the Jenkins XML configuration file.

Prerequisites

- AWS or a cloud environment with instances ready to be configured.
- Ansible installed on the master node.
- Target nodes accessible via SSH with appropriate permissions.
- A basic understanding of Ansible playbooks.

Jenkins and Ansible: Overview and How They Work Together

1. What is Jenkins?

Jenkins is an open-source automation server used for **Continuous Integration (CI) and Continuous Deployment (CD)**. It automates tasks like building, testing, and deploying applications.

Key Features of Jenkins:

- Automates the software development process (build, test, deploy).
- Supports integration with various tools (Git, Docker, Kubernetes, AWS, etc.).
- Provides a web-based UI and REST API for automation.
- Uses **Jenkins Pipelines** (written in Groovy) to define workflows.
- Can be extended with plugins.

Jenkins Workflow:

1. **Developer commits code** → Pushes changes to GitHub/GitLab.
2. **Jenkins detects changes** → Triggers a build automatically.
3. **Jenkins builds the project** → Compiles and packages the code.
4. **Jenkins runs tests** → Ensures the code works as expected.
5. **Jenkins deploys the application** → Pushes the code to a server, Kubernetes, or cloud services.

Why Use Jenkins?

- Automates repetitive tasks.
- Ensures code quality through automated testing.
- Reduces human errors in deployments.
- Speeds up software releases.

2. What is Ansible?

Ansible is an open-source **IT automation tool** that helps manage and configure servers. It is agentless, meaning it doesn't require any software installation on the target servers.

Key Features of Ansible:

- **Declarative Language:** Uses YAML-based Playbooks.
- **Agentless Architecture:** Uses SSH (Linux) or WinRM (Windows) instead of installing agents.
- **Idempotent Execution:** Ensures the same task doesn't repeat unnecessarily.
- **Scalability:** Can manage thousands of servers.

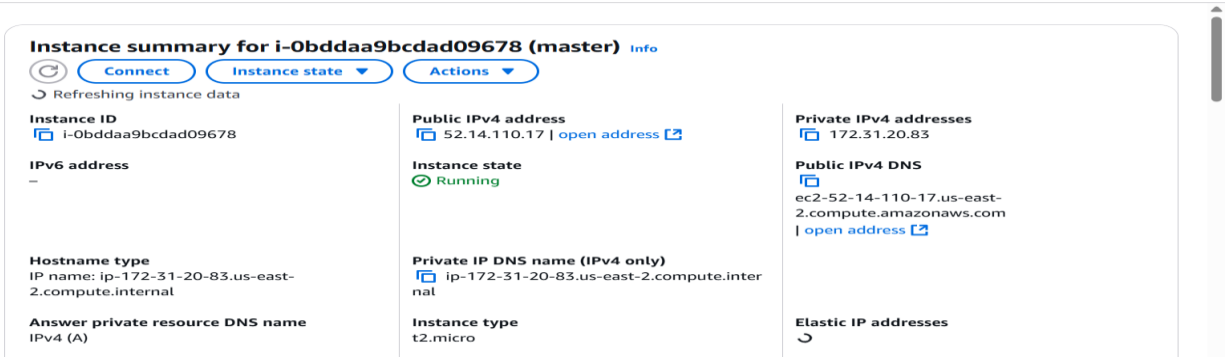
How Ansible Works:

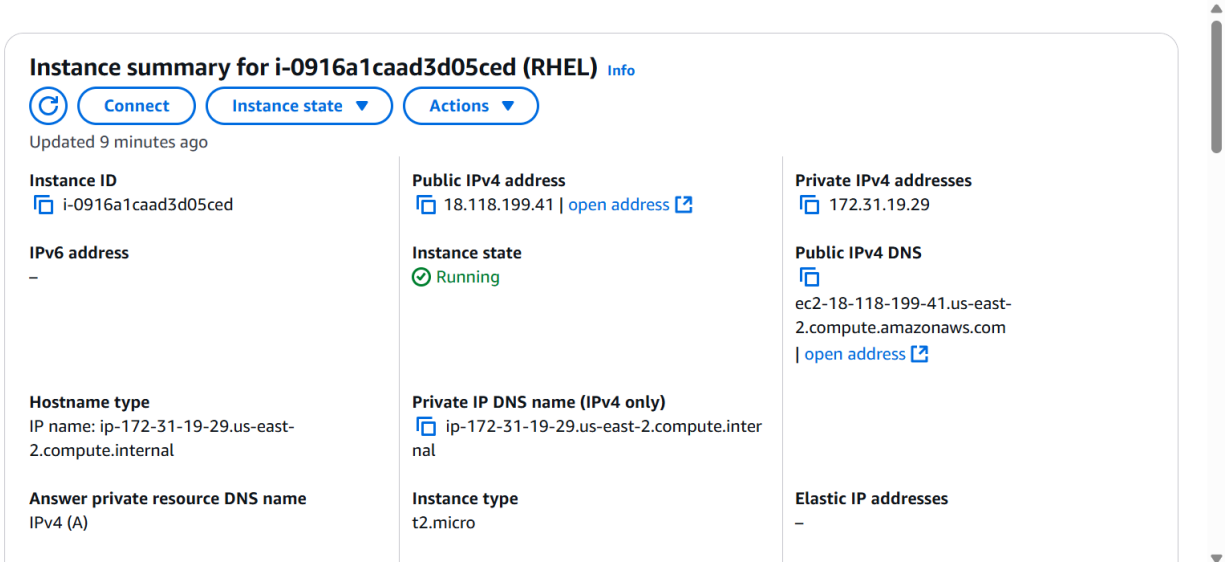
1. The **control node** (where Ansible is installed) runs commands.
2. It connects to **managed nodes** (servers) using SSH.
3. It executes tasks **defined in Playbooks** (YAML files).

Project Steps:

Step 1: Launching and Connecting Instances

1. Launch a master instance and multiple slave instances.
2. Configure SSH key-based authentication between the master and slaves.





3. Update the Ansible inventory file with the target hosts.
4. Ensure the Ansible master can communicate with the nodes using:

```
ansible all -m ping
```

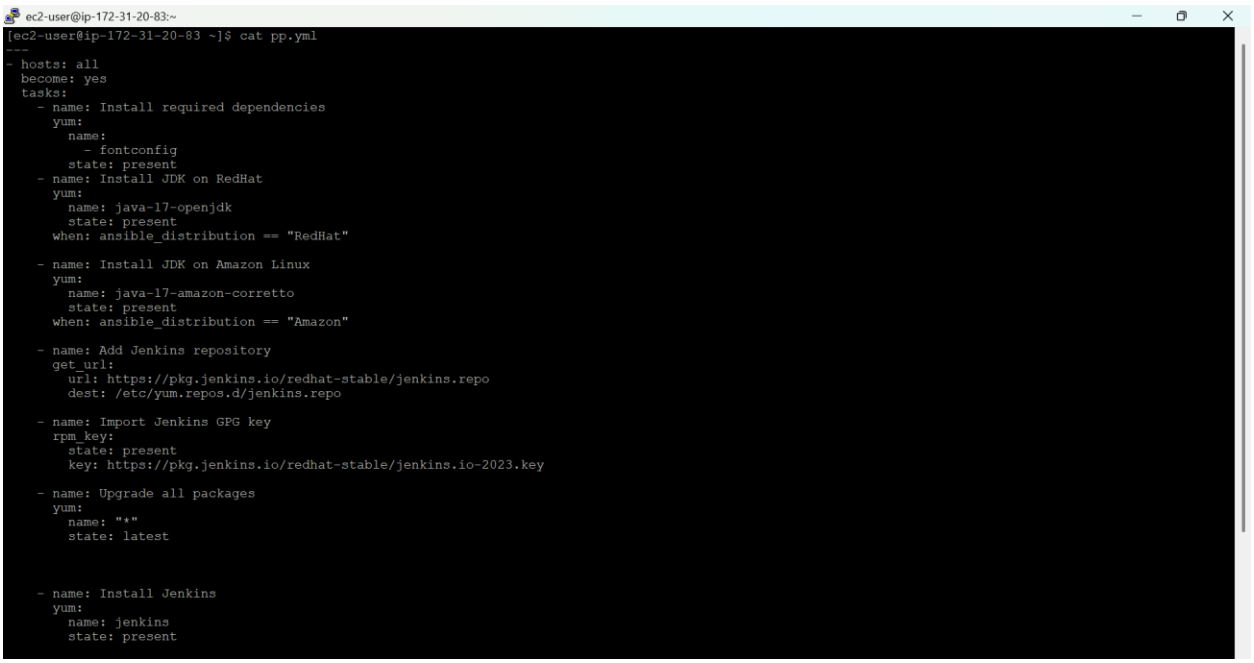
```
[ec2-user@ip-172-31-20-83 ~]$ ansible all -i slaves.txt -m ping
[WARNING]: log file at /var/log/ansible.log is not writeable and we cannot creat
e it, aborting

[WARNING]: Platform linux on host 172.31.21.100 is using the discovered Python
interpreter at /usr/bin/python3.9, but future installation of another Python
interpreter could change the meaning of that path. See
https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
172.31.21.100 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.9"
    },
    "changed": false,
    "ping": "pong"
}
172.31.27.235 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Step 2: Installing Jenkins with Ansible

The following Ansible playbook installs Jenkins and its dependencies on all target nodes.

```
---
- hosts: all
  become: yes
  tasks:
    - name: Install required dependencies
      yum:
        name:
          - fontconfig
        state: present

    - name: Install JDK on RedHat
      yum:
        name: java-17-openjdk
        state: present
      when: ansible_distribution == "RedHat"

    - name: Install JDK on Amazon Linux
```

```
    yum:
      name: java-17-amazon-corretto
      state: present
  when: ansible_distribution == "Amazon"

  - name: Add Jenkins repository
    get_url:
      url: https://pkg.jenkins.io/redhat-stable/jenkins.repo
      dest: /etc/yum.repos.d/jenkins.repo

  - name: Import Jenkins GPG key
    rpm_key:
      state: present
      key: https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key

  - name: Upgrade all packages
    yum:
      name: "*"
      state: latest

  - name: Install Jenkins
    yum:
      name: jenkins
      state: present

  - name: Reload systemd daemon
    systemd:
      daemon_reload: yes

  - name: Start and enable Jenkins service
    systemd:
      name: jenkins
      state: started
      enabled: yes
```


ec2-user@ip-172-31-20-83:~
[ec2-user@ip-172-31-20-83 ~]$ cat pp.yml

- hosts: all
  become: yes
  tasks:
    - name: Install required dependencies
      yum:
        name:
          - fontconfig
        state: present
    - name: Install JDK on RedHat
      yum:
        name: java-17-openjdk
        state: present
      when: ansible_distribution == "RedHat"

    - name: Install JDK on Amazon Linux
      yum:
        name: java-17-amazon-corretto
        state: present
      when: ansible_distribution == "Amazon"

    - name: Add Jenkins repository
      get_url:
        url: https://pkg.jenkins.io/redhat-stable/jenkins.repo
        dest: /etc/yum.repos.d/jenkins.repo

    - name: Import Jenkins GPG key
      rpm_key:
        state: present
        key: https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key

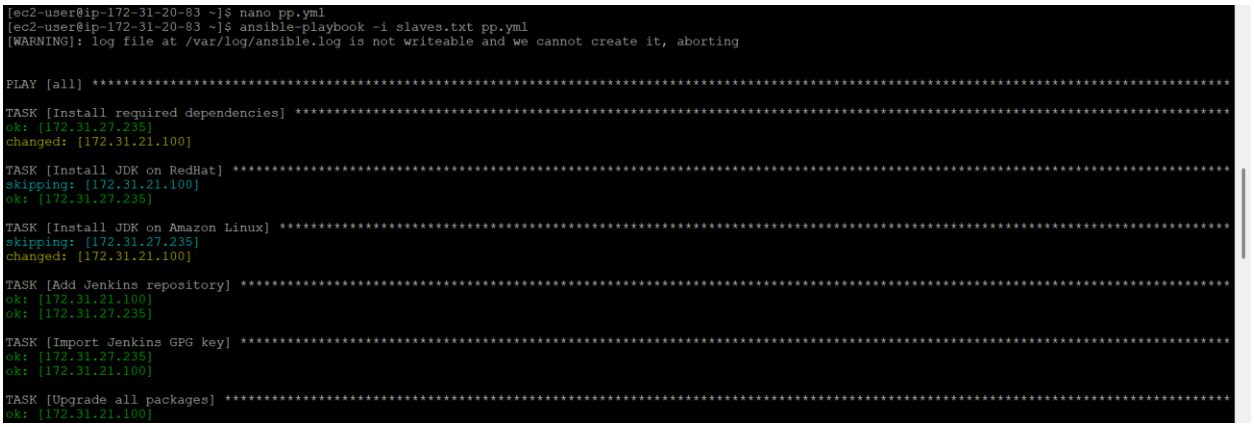
    - name: Upgrade all packages
      yum:
        name: "*"
        state: latest

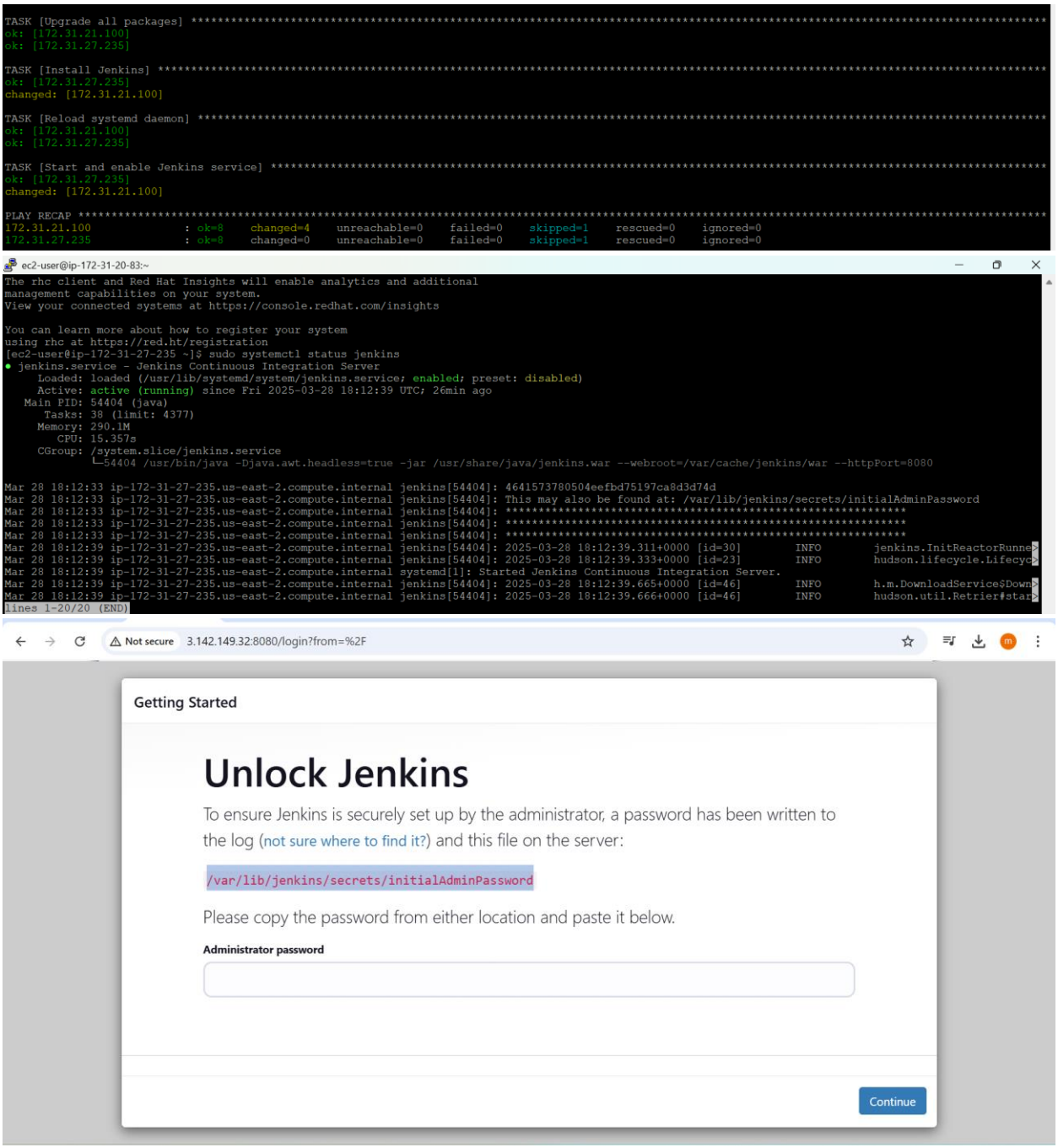
    - name: Install Jenkins
      yum:
        name: jenkins
        state: present


- name: Install Jenkins
  yum:
    name: jenkins
    state: present

- name: Reload systemd daemon
  systemd:
    daemon_reload: yes

- name: Start and enable Jenkins service
  systemd:
    name: jenkins
    state: started
    enabled: yes
[ec2-user@ip-172-31-20-83 ~]$

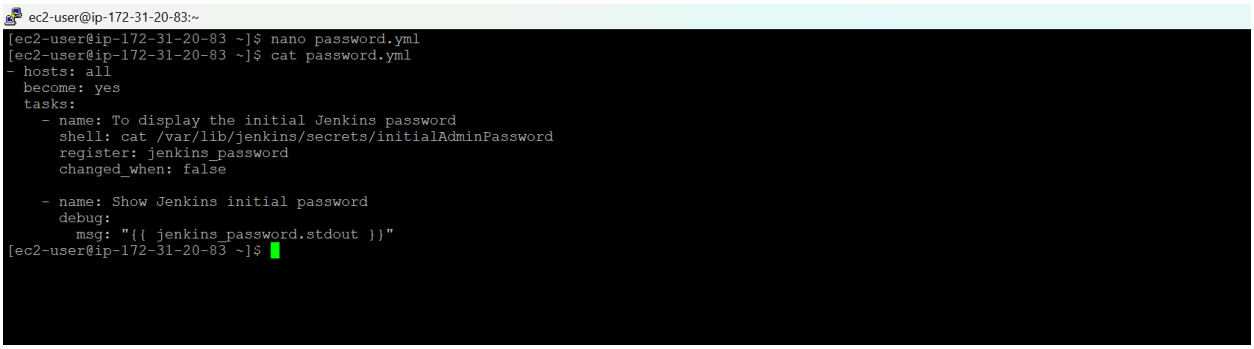

[ec2-user@ip-172-31-20-83 ~]$ nano pp.yml
[ec2-user@ip-172-31-20-83 ~]$ ansible-playbook -i slaves.txt pp.yml
[WARNING]: log file at /var/log/ansible.log is not writeable and we cannot create it, aborting
PLAY [all] ****
TASK [Install required dependencies] ****
ok: [172.31.27.235]
changed: [172.31.21.100]
TASK [Install JDK on RedHat] ****
skipping: [172.31.21.100]
ok: [172.31.27.235]
TASK [Install JDK on Amazon Linux] ****
skipping: [172.31.27.235]
changed: [172.31.21.100]
TASK [Add Jenkins repository] ****
ok: [172.31.21.100]
ok: [172.31.27.235]
TASK [Import Jenkins GPG key] ****
ok: [172.31.27.235]
ok: [172.31.21.100]
TASK [Upgrade all packages] ****
ok: [172.31.21.100]

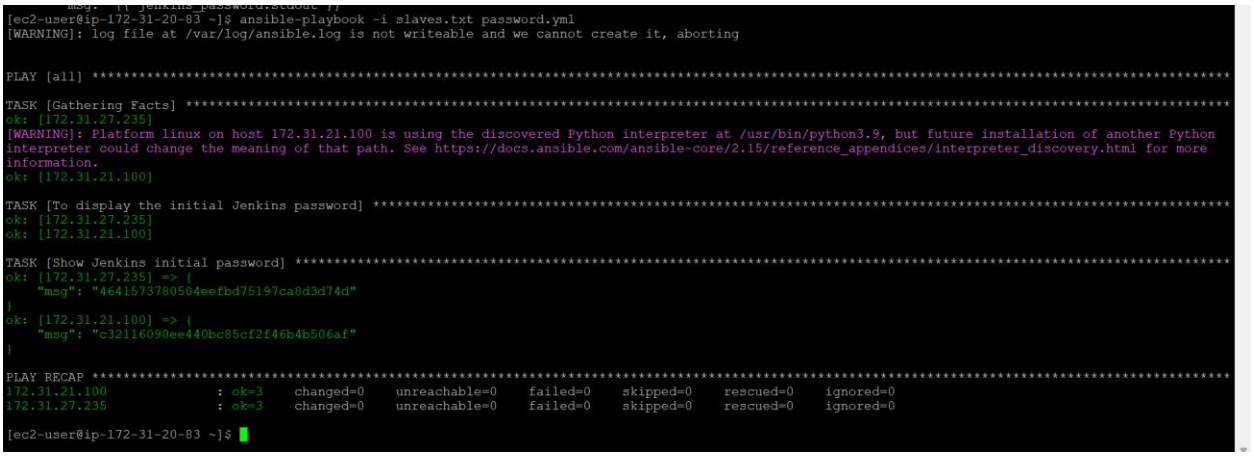


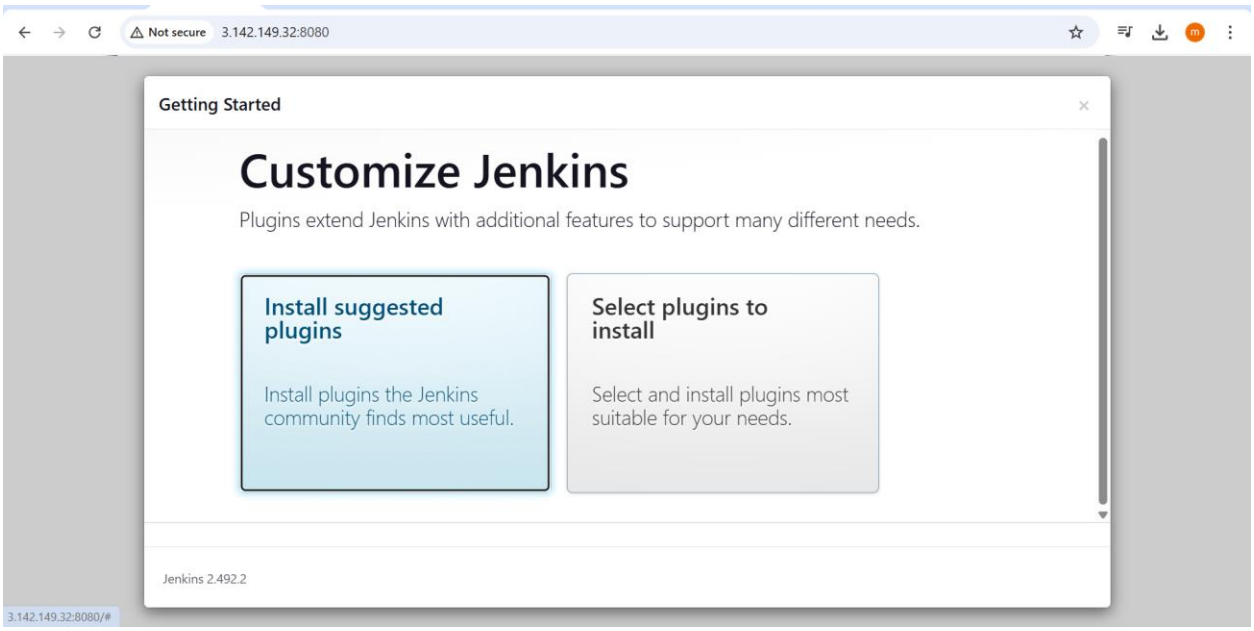
Step 3: Displaying the Initial Jenkins Password

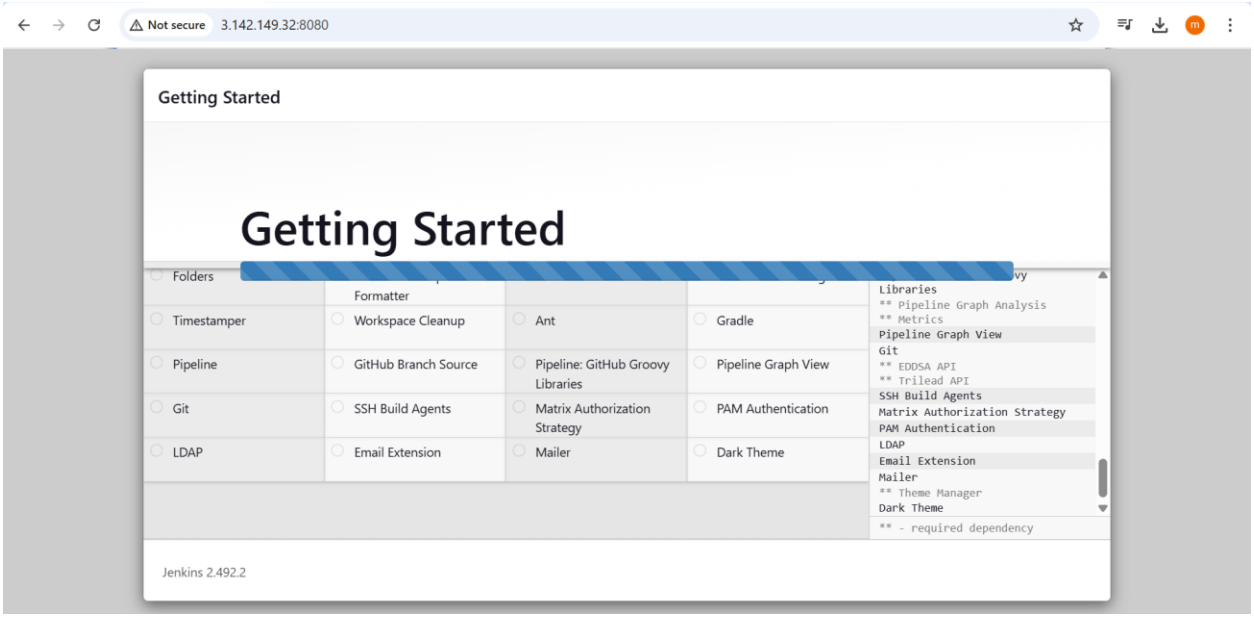
After Jenkins is installed, the initial admin password must be retrieved for first-time access.

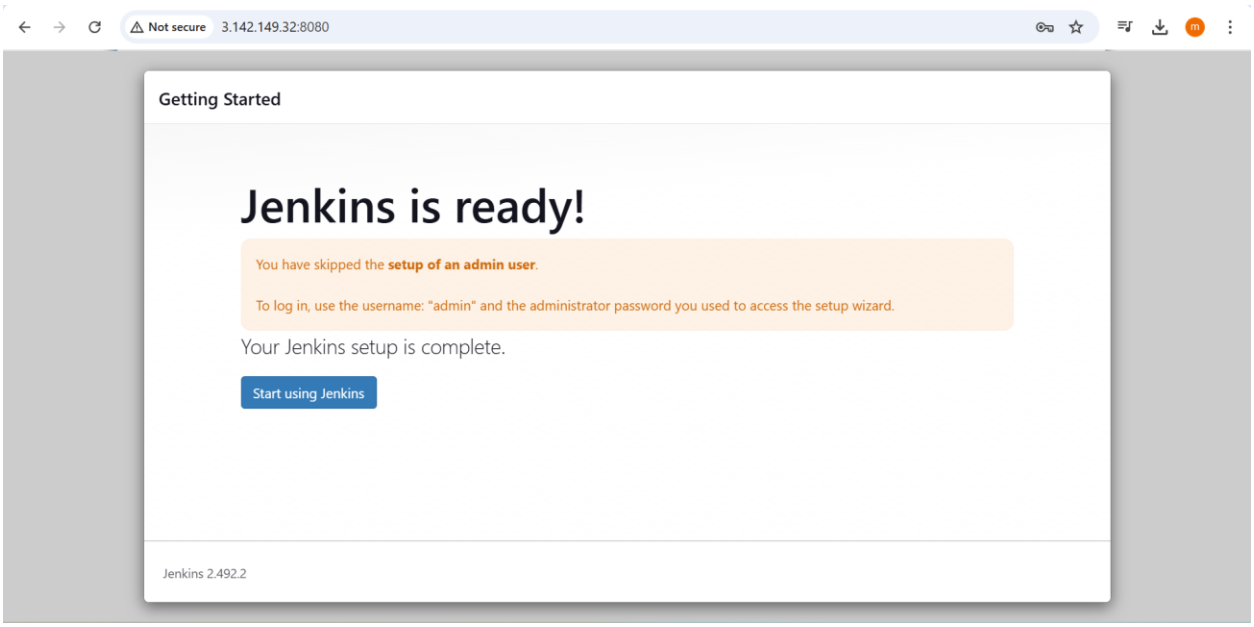
```
- hosts: all
  become: yes
  tasks:
    - name: Retrieve the initial Jenkins password
      shell: cat /var/lib/jenkins/secrets/initialAdminPassword
      register: password

    - name: Display Jenkins initial password
      debug:
        msg: "{{ password.stdout }}"
```

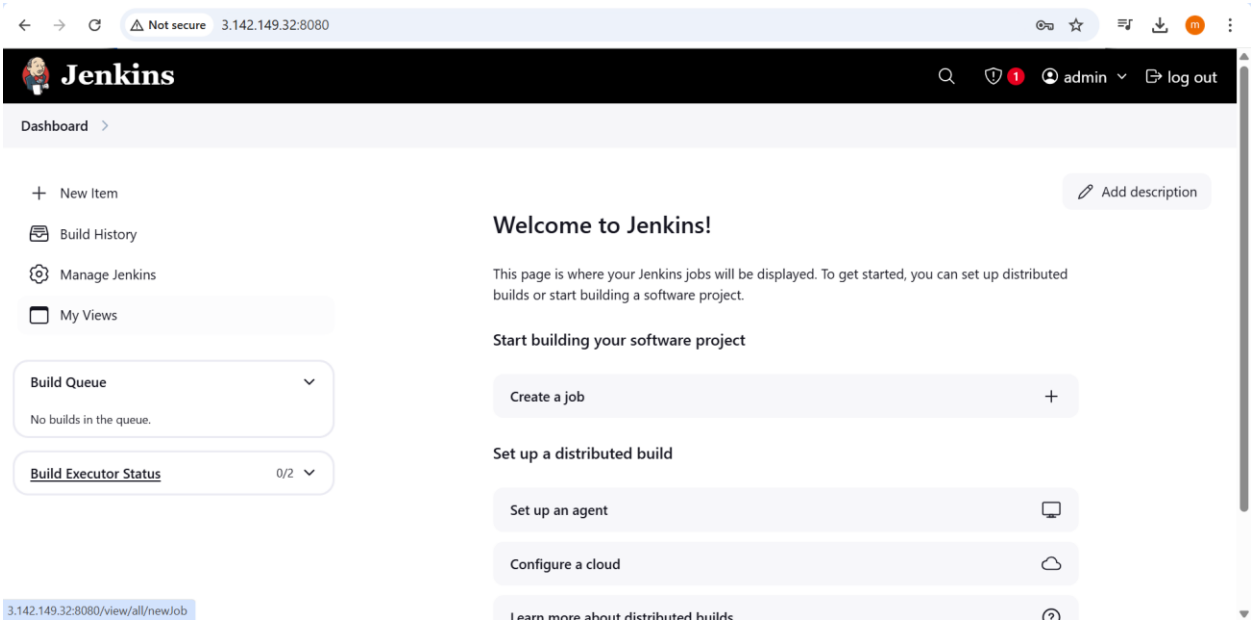





Not secure 3.142.149.32:8080
Getting Started
Customize Jenkins
Plugins extend Jenkins with additional features to support many different needs.
Install suggested plugins
Install plugins the Jenkins community finds most useful.
Select plugins to install
Select and install plugins most suitable for your needs.
Jenkins 2.492.2
3.142.149.32:8080/#


Not secure 3.142.149.32:8080
Getting Started
Getting Started
Folders
Formatter
Timestamper
Workspace Cleanup
Ant
Gradle
Pipeline
GitHub Branch Source
Pipeline: GitHub Groovy Libraries
Pipeline Graph View
Git
SSH Build Agents
Matrix Authorization Strategy
PAM Authentication
LDAP
Email Extension
Mailer
Dark Theme
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
** - required dependency
Jenkins 2.492.2





Step 4: Changing the Jenkins Port and Updating the Configuration

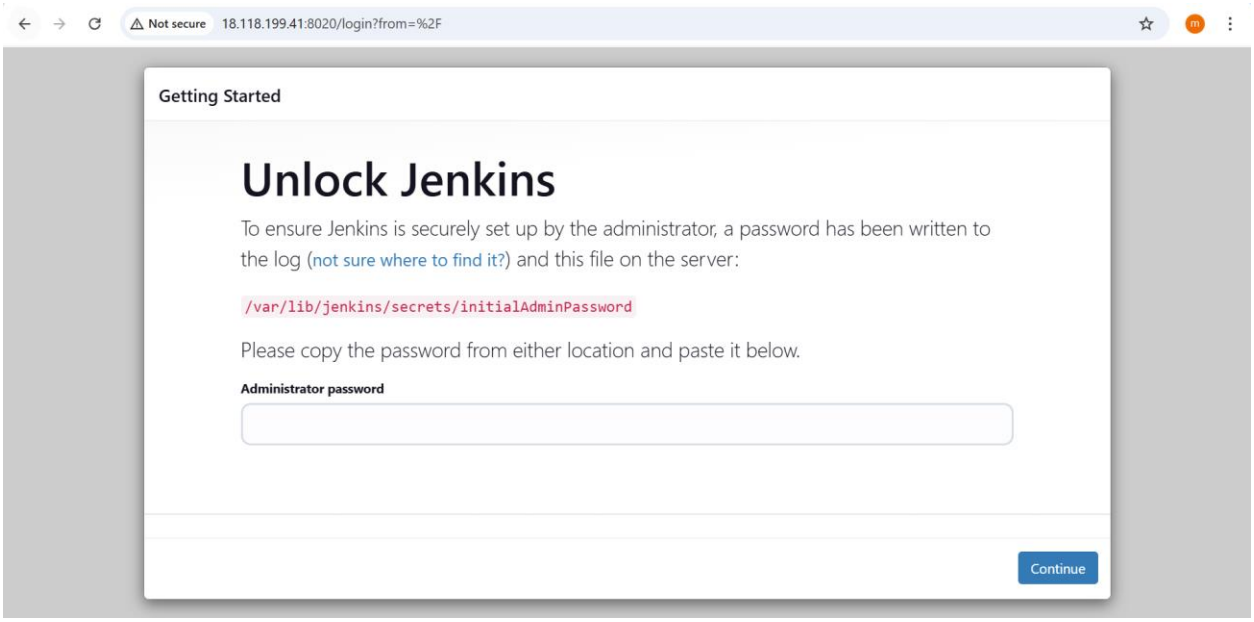
This section updates the default Jenkins port using an Ansible variable

```yaml
---
- name: Playbook to Install Jenkins and Amazon Corretto JDK
  hosts: all
  become: yes
  vars:
    portnum: 8040
  tasks:
    - name: Install required dependencies
      yum:
        name:
          - fontconfig
          - java-17-amazon-corretto
        state: present
    - name: Add Jenkins repository
      get_url:
        url: https://pkg.jenkins.io/redhat-stable/jenkins.repo
        dest: /etc/yum.repos.d/jenkins.repo
    - name: Import Jenkins repository key
      rpm_key:
        state: present
        key: https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
    - name: Add Amazon Corretto repository
      get_url:
        url: https://yum.corretto.aws/corretto.repo
        dest: /etc/yum.repos.d/corretto.repo
    - name: Upgrade all packages
      yum:
        name: "*"
        state: latest
    - name: Install Jenkins
      yum:
        name: jenkins
        state: present
    - name: Reload systemd daemon
      command: systemctl daemon-reload
    - name: Update Jenkins environment port in systemd service file
      lineinfile:
        path: /usr/lib/systemd/system/jenkins.service
        regexp: '^Environment="JENKINS_PORT='
        line: 'Environment="JENKINS_PORT={{ portnum }}"'
        state: present
    - name: Reload systemd daemon after modifying service file
      command: systemctl daemon-reload
```

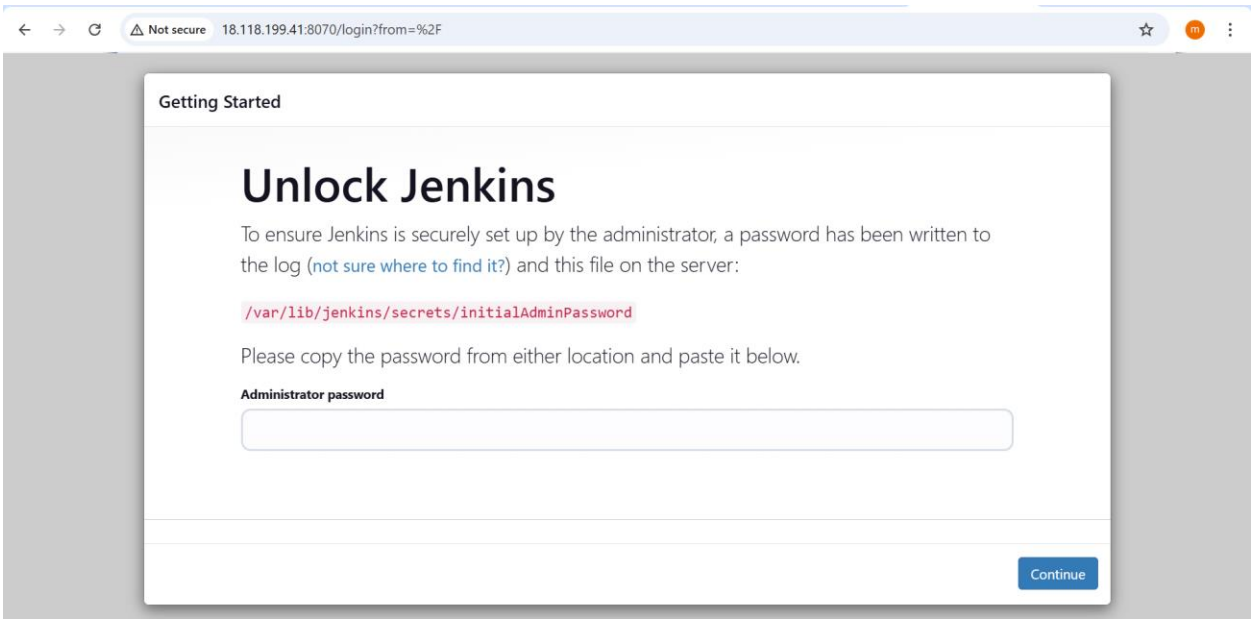
```yaml
    - name: Enable and restart Jenkins service
      systemd:
        name: jenkins
        enabled: yes
        state: restarted
```

```
    state: present
  - name: Reload systemd daemon
    command: systemctl daemon-reload
  - name: Update Jenkins environment port in systemd service file
    lineinfile:
      path: /usr/lib/systemd/system/jenkins.service
      regexp: '^Environment="JENKINS_PORT='
      line: 'Environment="JENKINS_PORT={{ portnum }}"'
      state: present
  - name: Reload systemd daemon after modifying service file
    command: systemctl daemon-reload
  - name: Enable and restart Jenkins service
    systemd:
      name: jenkins
      enabled: yes
      state: restarted
[ec2-user@ip-172-31-16-249 ~]$
```

Before changing the Port :



After Changing Port :



Conclusion

This playbook ensures that:

- Jenkins and its dependencies are installed on multiple nodes.
- The initial admin password is retrieved and displayed.
- The default Jenkins port is updated dynamically using an Ansible variable.

By using Ansible automation, Jenkins installation and configuration become efficient, scalable, and repeatable across multiple servers.