DevOps simple projects

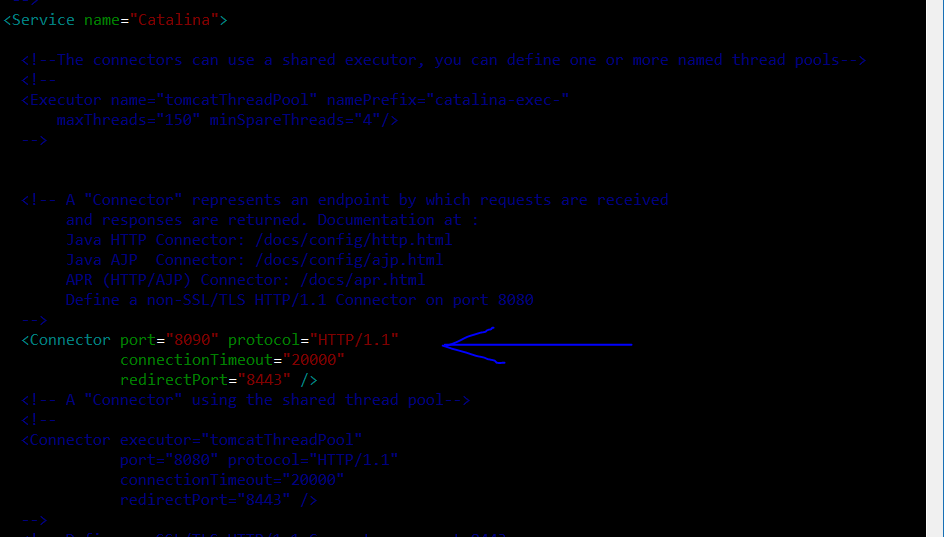
Project 1: CI/CD using Jenkins on apache-tomcat server

1. Upload code (webapp) to git.
   1. Write a simple java code for a webpage.
   2. Name the project webapp.
   3. Include the pom.xml file
2. Install Jenkins server on ubuntu (port 8080)
   1. Install Java (jdk &jre)
      1. sudo su
      2. apt-get update -y
      3. apt-cache search jdk
      4. apt-get install jdk-
      5. java -version
      6. set path for java
         1. export JAVA\_HOME=/usr/lib/jvm/java-8-openjdk-amd64
         2. export PATH=$PATH:$JAVA\_HOME/bin
   2. Install Jenkins
      1. wget -q -O - <https://pkg.jenkins.io/debian-stable/jenkins.io.key> | sudo apt-key add –
      2. vi /etc/apt/sources.list

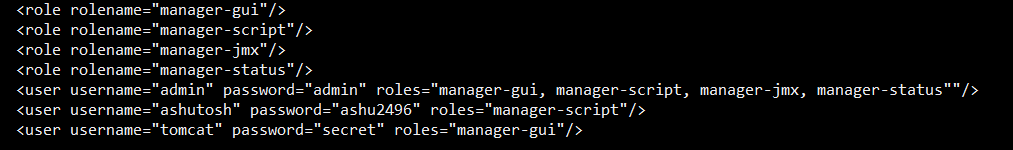
add following line: deb [https://pkg.jenkins.io/debian-stable binary/](https://pkg.jenkins.io/debian-stable%20binary/)

* + 1. apt-get update
    2. apt-get install jenkins
    3. systemctl status jenkins
    4. cat /var/lib/jenkins/secrets/initialAdminPassword #copy password
    5. open <public-ip>:8080 and paste password
  1. Install maven
     1. apt-get install maven
     2. systemctl status maven

1. Install apache-tomcat on another machine (port 8090)
   1. Install Java (jdk &jre)
      1. apt-get update -y
      2. apt-cache search java
      3. apt-get install jdk-
      4. java -version
      5. set path for java
         1. export JAVA\_HOME=/usr/lib/jvm/java-8-openjdk-amd64
         2. export PATH=$PATH:$JAVA\_HOME/bin
   2. install apache-tomcat package
      1. download tomcat package
         1. wget https://mirrors.estointernet.in/apache/tomcat/tomcat-9/v9.0.45/bin/apache-tomcat-9.0.45.tar.gz
         2. tar -xvzf apache-tomcat-9.0.45.tar.gz
      2. copy this file to /opt/tomcat directory
         1. mkdir -p /opt/tomcat
         2. cp -rf apache-tomcat-9.0.45/\* /opt/tomcat/
      3. change port no. conf file
         1. cd /opt/tomcat/
         2. vi conf/server.xml

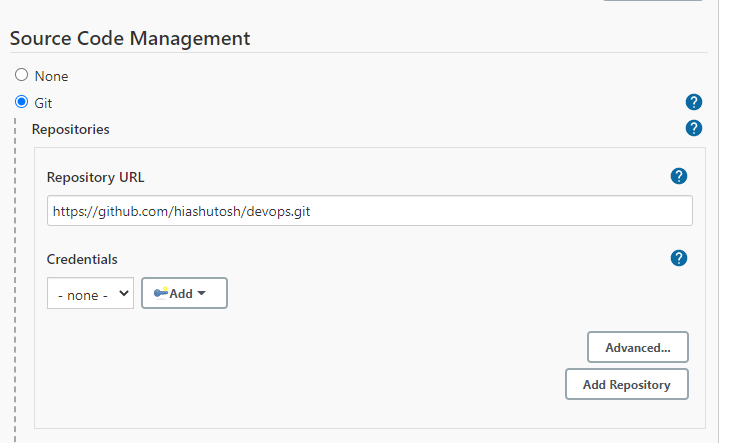


* + 1. add users to tomcat-user.xml file under conf dir
       1. vi conf/tomcat-user.xml



* + 1. change ownership of startup.sh and shutdown.sh
       1. cd /opt/tomcat/bin
       2. chmod +x startup.sh
       3. chmod +x shutdown.sh
    2. create a softlink for starting tomcat
       1. ln -s /opt/tomcat/startup.sh /usr/local/bin/tomcatup
       2. ln -s /opt/tomcat/shutdown.sh /usr/local/bin/tomcatdown
    3. OR create a romcat.service file for system
       1. sudo vi /etc/systemd/tomcat.service
       2. paste the following:

1. [Unit]
2. Description=Tomcat
3. After=syslog.target network.target
4. [Service]
5. Type=forking
6. User=tomcat
7. Group=tomcat
8. ExecStart=/opt/tomcat/bin/catalina.sh start
9. ExecStop=/opt/tomcat/bin/catalina.sh stop
10. [Install]
11. WantedBy=multi-user.target
    * + 1. Reload the systemd : systemctl daemon-reload
      1. 
      2. start tomcat server
         1. tomcatup
         2. OR sytemctl start tomcat
12. Create a job in jenkins to build jar/war file from git and deploy to tomcat
    1. Install plugin: Deploy to container.
    2. Create a new job and add source as git
    3. In post build action select with <public-ip-tomcat>:8090





Project 2: CI/CD using Jenkins and deploying using ansible on apache-tomcat server

Integration with previous project

1. Configure ansible node
   1. vi /etc/ssh/sshd\_config

apply following changes

uncomment PermitRootLogin yes

uncomment PasswrdAuthentication yes

comment PasswordAuthentication no

* 1. adduser <username>
  2. passwd <username>
  3. usermod -aG root <usermod>
  4. visudo : add line
     1. <username> ALL:(ALL) NOPASSWD:ALL
  5. vi /etc/ssh/sshd\_config
     1. uncomment PasswrdAuthentication yes
     2. comment PasswordAuthentication no
  6. service sshd restart

1. Configure ansible host
   1. apt-get install ansible -y
   2. vi /etc/ssh/sshd\_config
      1. uncomment PasswrdAuthentication yes
      2. comment PasswordAuthentication no
   3. vi /etc/ansible/ansible.cfg

Uncomment following

# inventory = /etc/ansible/hosts

#sudo-user = root

* 1. Add tomcat server details to /etc/ansible/hosts
     1. echo “<server-ip>” >> /etc/ansible/hosts
  2. adduser <username>
  3. passwd <username>
  4. usermod -aG root <usermod>
  5. visudo : add line
     1. <username> ALL:(ALL) NOPASSWD:ALL

<username> should have sudo access then only it can perform action on node

* 1. su <username>
  2. ssh-keygen
  3. ssh-copy-id <username>@<node-ip>

1. Add plugin: Publish over SSH
2. Enable connection b/w ansible and jenkins
   1. Manage jenkins> configure system > publish over SSH > SSH servers
   2. Add server
      1. Hostname: <ansible-server-ip>
      2. Username: <username>
      3. Password: <pass>
   3. Test connection
3. Create a playbook on path: /opt/playbooks

---

hosts: all

become: true

vars:

- ansible\_sudo\_pass: “<user-pass>”

# sudo pass should be provided generally for debian/ubuntu family

tasks:

- name: copy jar/war into tomcat server

copy:

src: /opt/playbooks/webapp/target/webapp.war

dest: /opt/apache-tomcat-9.0.45/webapps

1. create jenkins job
   1. repository: <git-url-code>
   2. build: invoke top-level maven
      1. root POM: pom.xml
      2. Goals and options: clean install package
      3. Post build: send files or execute over SSH
         1. SSH server: ansible-server
         2. Sources files: /target/\*.war
         3. Remote Directory: //opt//playbooks//webapp

Remote dir should be owned by same user by which ssh connection is established.

* + 1. Post build: send files or execute over SSH
       1. SSH server: ansible-server
       2. Exec command:
          1. ansible-playbook /opt/playbooks/<playbook>.yml

1. execute job and open <tomcat-ip>:8090/webapp

Project 3: CI/CD using Jenkins on docker container: tomcat

1. Install docker on host
   1. apt-get install docker.io
   2. service docker start
2. Manage docker host
   1. useradd <username>
   2. passwd <username>
   3. usermod -aG docker <username>
3. Write Dockerfile
   1. mkdir /opt/docker
   2. vi Dockerfile

# base image

From tomcat:8-jre

COPY ./webapp.war /usr/local/tomcat/webapps

Directory in which Dockerfile is created ( or where we need to PUT webapp.war file should be owned by same user by which ssh connection is established.

1. Add docker server in jenkins
   1. Manage jenkins > configure system > Publish over SSH > add Docker server and credentials
2. Create Jenkins job
   1. Repo: <git-url-code>
   2. Build
      1. Root: pom.xml
      2. Goals and options: clean install package
   3. Send files or execute command over SSH
      1. Name: docker-host
      2. Source file: target/\*war
      3. Remove prefix: target
      4. Remote directory: //opt/docker
      5. Execute commands:
         1. cd /opt/docker
         2. docker build -t <image-name> .

“.” Should be provided if Dockerfile is same directory in where we run build command. We can also provide whole path while building image

* 1. Send files or execute command over SSH
     1. Name: docker-host
     2. Exec command:
        1. docker run -d --name <cont-name> -p 8090:8080 <image-name>

1. execute jenkins job
2. access web app <docker-host-ip>:8090/webapp

Project 4: CI/CD using Jenkins and deployed using ansible on dockerhub and pull it into container: tomcat and including versioning in dockerhub.

Prerequisites:

Jenkins

Ansible: host and node

Dockerhub account

1. Create a Dockerfile into github

# base image

From tomcat:8-jre

COPY ./webapp.war /usr/local/tomcat/webapps

1. Create a Jenkins job
   1. Repo: git
   2. Build
      1. Root POM: pom.xml
      2. Goals: clean install package
   3. Post step: send file or execute over SSH
      1. Name: ansible\_server
      2. Souce file: target/\*.war
      3. Remove prefix: target
      4. Remote directory: //opt//docker
   4. Post step: send file or execute over SSH
      1. Name: ansible\_server
      2. Souce file: Dockerfile
      3. Remote Directory: //opt//docker
      4. Exec command:
         1. cd /opt/docker
         2. docker build -t <img\_name> .
         3. docker tag <img\_name> <dockerhub>/<img\_name>
         4. docker push <dockerhub>/<img\_name>
         5. docker rmi <img\_name> <dockerhub>/<img\_name>
2. Create a playbook: dock\_cont.yml

---

- hosts: <host>

user:

become: true

vars:

- ansible\_sudo\_pass: “<node-pass>”

tasks:

- name: Stop previous docker version

docker\_container:

name: <cont\_name>

path: docker

state: stopped

- name: remove stopped container

docker\_container:

name: <cont\_name>

path: docker

state: absent

- name: remove docker image

docker\_image:

name: <img\_name>

path: docker

state: absent

- name: Build Docker image from Dockerfile

docker\_image:

name: <img\_name>

path: docker

state: build

- name: Tag and push to dockerhub

docker\_image:

name: < img\_name>

repository: <dockerhub>/<img\_name>

- name: run container from image

shell: docker run -d --name <cont\_name> -p 8090:8080 <dockerhub>/<img\_name>

Dockerfile and .war file should be transferred to node

Login into docker hub in node for pushing images

Project 4: CI/CD using Jenkins and deployed using ansible on dockerhub and pull it into container: tomcat and including versioning in dockerhub.

Prerequisites:

Jenkins

Ansible: host and node

Dockerhub account

1. Create a Dockerfile into github

# base image

From tomcat:8-jre

COPY ./webapp.war /usr/local/tomcat/webapps

1. Create a Jenkins job
   1. Repo: git
   2. Build
      1. Root POM: pom.xml
      2. Goals: clean install package
   3. Post step: send file or execute over SSH
      1. Name: ansible\_host
      2. Souce file: target/\*.war
      3. Remove prefix: target
      4. Remote directory: //opt//docker
   4. Post step: send file or execute over SSH
      1. Name: ansible\_host
      2. Souce file: Dockerfile
      3. Remote directory: //opt//docker
   5. Post step: send file or execute over SSH
      1. Name: ansible\_server
      2. Souce file: cont.yml
      3. Remote Directory: //opt//docker
      4. Exec command:
         1. cd /opt/docker
         2. ansible-playbook cont.yml
2. create a playbook cont.yml in git

|  |
| --- |
| --- |
|  | - hosts: all |
|  | vars: |
|  | - ansible\_sudo\_pass: "1234" |
|  | user: ansible |
|  | become: true |
|  | become\_method: sudo |
|  | connection: ssh |
|  | tasks: |
|  | - name: change build docker |
|  | shell: cd /home/ansible/playbook |
|  | - name: build docker |
|  | shell: docker build -t demo . |
|  |  |
|  | - name: tag container latest |
|  | shell: docker tag demo yadavashu/demo:latest |
|  |  |
|  | - name: push prev container |
|  | shell: docker push yadavashu/demo:latest |
|  |  |
|  | - name: create docker image |
|  | shell: docker run -d --name cont\_44 -p 8090:8080 yadavashu/demo:latest |