ICT3203 Lab Quiz CheatSheet

It’s not a holy bible but should be able to get the job done.

# VM Environment Setup

In case we are getting a fresh VM and needs to install the packages required to do the test. This will install Docker and nginx.

Ubuntu 18 <

*sudo apt update && sudo apt install ca-certificates curl gnupg lsb-release -y && echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null && sudo apt-get update && sudo apt-get install docker-ce docker-ce-cli containerd.io nginx -y y*

ubuntu 20 >

sudo apt-get update && sudo apt-get install apt-transport-https ca-certificates curl software-properties-common -y && curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add - && sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable" && sudo apt install docker-ce docker-ce-cli containerd.io nginx -y

After all installation:

*sudo systemctl start nginx && sudo systemctl enable nginx*

*sudo ufw enable && sudo ufw allow ‘nginx full’ && sudo ufw reload*

# Cheatsheet

## Docker

Commonly used Docker’s commands are boxed in red.

Table

Description automatically generated

## Nginx

Some useful commands to use during configuration

* To check for syntax error
  + sudo nginx -t
* To restart / reload the service **after amending** the conf files (must do for new configuration to take effect)
  + sudo systemctl restart / reload nginx
* Check if nginx is up and running
  + sudo systemctl status nginx

# Jenkins Configuration

What you need to configure Jenkins successfully:

1. Dockerfile
2. Jenkinsfile
3. Patient
4. And pray hard

## DockerFile

Run this to build a Jenkins Docker Image

*cd ~ && vim Dockerfile*

CP the following code into the Dockerfile and save. P.s. can use whatever text editors that you are comfortable with. VIM or NANO. Once saved, run the next command

FROM jenkins/jenkins:2.303.3-jdk11

USER root

RUN apt-get update && apt-get install -y lsb-release

RUN curl -fsSLo /usr/share/keyrings/docker-archive-keyring.asc \

https://download.docker.com/linux/debian/gpg

RUN echo "deb [arch=$(dpkg --print-architecture) \

signed-by=/usr/share/keyrings/docker-archive-keyring.asc] \

https://download.docker.com/linux/debian \

$(lsb\_release -cs) stable" > /etc/apt/sources.list.d/docker.list

RUN apt-get update && apt-get install -y docker-ce-cli

USER jenkins

RUN jenkins-plugin-cli --plugins "blueocean:1.25.1 docker-workflow:1.26"

*sudo docker build -t myjenkins-blueocean:1.1 .*

## Setting up Docker:dind (docker-inside-dociker)

docker network create Jenkins

sudo mkdir /var/Jenkins\_home

docker run --name jenkins-docker --rm --detach \

--privileged --network jenkins --network-alias docker \

--env DOCKER\_TLS\_CERTDIR=/certs \

--volume jenkins-docker-certs:/certs/client \

--volume jenkins-data:/var/jenkins\_home \

--publish 2376:2376 docker:dind --storage-driver overlay2

## Setting up Docker:sock (docker-outside-docker) (Recommended)

*sudo mkdir /var/Jenkins\_home*

Depending on the requirement for the test, change the IP addr and port number accordingly

<Port number of Host>:<Port number of containers>

*sudo docker run --name jenkins-blueocean --rm --detach --user root --volume /var/run/docker.sock:/var/run/docker.sock --volume /var/jenkins\_home:/var/jenkins\_home --publish 127.0.0.1:8080:8080 myjenkins-blueocean:1.1*

## Setting up Jenkins to Serve Over HTTPS with Nginx as Reverse Proxy

Create a .conf in /etc/nginx/sites-available and symlink over to /etc/nginx/sites-enabled. Include these the following the the .conf file for Jenkins to serve over HTTPS with Nginx as reverse proxy. Here, we are assuming that the labtest wants us to serve Jenkins over URl with port 8443 specified. **If serving over URL is not test, directly access Jenkins via 127.0.0.1:8080**

**Main things to change if this is tested (serving over URL):**

* Server IP\_addr:port fail\_timeout=0;
* Server\_name

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

## Accessing Jenkins for the First Time

To get the password to access Jenkins for the first time,

*sudo docker ps*

Get the containerID of the Jenkins

*sudo docker logs <containerID>*

Text

Description automatically generated

## Setting Jenkins pipeline project to listen for Github webhook

Here we will talk about how to set up pipeline project to listen for webhook. Laymen term would be for Jenkins to auto deploy when you pushed to github lah.

Dashboard>New Item>pipeline

Application

Description automatically generated with medium confidence

Remember to create your Personal Authentication Token from Github. Project URL format as such: https://<PAT>@github.com/path/to/git/project.git

Graphical user interface, application

Description automatically generated

A picture containing shape

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

## Jenkinsfile Structure

Here we talk about how to write a Jenkinsfile. If suddenly you don’t know how to write the Jenkinsfile and can’t find the syntax online, theres inbuilt syntax builder.

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

Incase you need to deploy Global variables, visit <https://JenkinsIP:port/configure> . Once configured, you can call the global var in your Jenkinsfile via ${env.GLOBALVARNAME}

Graphical user interface, text, application, chat or text message

Description automatically generated

# OWASP Dependency Checker

## Installation

Go to <http://jenkinsIP:port/pluginManager/available>

Under ‘Available’, find **OWASP Dependency-Check** and install it.

Once done, head to <http://jenkinsIP:port/configureTools> and name your dependency-check. This name is important as it will be used for your odcInstallation

Background pattern

Description automatically generated with low confidence

## Jenkinsfile configuration of Dependency Checker

A computer screen capture

Description automatically generated with medium confidence

# Automated Testing and Selenium Testing

## Automated Testing

Configure the Jenkinsfile as follows:

Text

Description automatically generated

## Selenium Testing

if deploy.sh is given, include these: main thing to take note is the containerport and volume (-v). The volume should targets your work files.

Text

Description automatically generated

Text

Description automatically generated

For Jenkinsfile, remember to give execute permission to deploy.sh and kill.sh

Text

Description automatically generated

# Jenkins Warning

Install Warning next generation, maven Integration, and Maven Invoker plugins

A picture containing background pattern

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Next, install maven inside

Two ways to install:

1. Under /configureTools/ ,GUI installation
   1. Graphical user interface, text, application, email

      Description automatically generated
2. Terminal
   1. *cd /var/Jenkins\_home*
   2. *sudo curl http://mirrors.estointernet.in/apache/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz --output apache-maven-3.6.3-bin.tar.gz && sudo tar -xvzf apache-maven-3.6.3-bin.tar.gz && cd apache-maven-3.6.3*
   3. *Graphical user interface, text, application, email

      Description automatically generated*

As for the Jenkinsfile, configure as such

Text

Description automatically generated

To add more plugin for warning-ng-plugin, amend this following line with reference to this link (<https://github.com/jenkinsci/warnings-ng-plugin/blob/master/SUPPORTED-FORMATS.md>)

<https://github.com/jenkinsci/warnings-ng-plugin/blob/master/doc/Documentation.md#configuration>

A computer screen capture

Description automatically generated with low confidence

# Sonarqube Configuration

Install sonarqube scanner

Graphical user interface, text, application

Description automatically generated

Install sonarqube image and docker run it:

*sudo docker pull sonarqube && sudo docker run -d --name sonarqube -e SONAR\_ES\_BOOTSTRAP\_CHECKS\_DISABLE=true -p 9000:9000 sonarqube:latest*

How to create project:

1. Graphical user interface, text, application

   Description automatically generated
2. Graphical user interface, application

   Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence