

* Tools

Oracle VirtualBox

Vagrant (automation)

Git Bash (version control, for CMD)

IDE (sublime, Notepad ++)

* Objective – VM automation locally how to setup real world project setup locally
* Architect – Ngnix, tomcat, RabbitMQ, MemcacheD, MySQL
* Design for the project –

1. User can access services with browser
2. User will be redirect to load balancer which is ngnix for this project then forward request to application server
3. Apache tomcat will be running our java application will be running we can also use NFS( shared storage) application server then forward the request to rabbitMQ
4. RabbitMq is our message broker it will forward request to memcacheD
5. MemcacheD Is going to SQL queries my mysql

* Automation design for vagrant-

Vagrant will be used for setting up our VM with Oracle Virtual box

Then will be using bash commands for setup Vm for each service we will require for this project

For this we need to write a Vagrant file with all the setup information

* Flow of execution

1. Setup all the tools
2. Clone Git repository
3. Cd into vagrant directory
4. And bring up the VM
5. Validate
6. Setup all services
7. Mysql
8. memcacheD
9. rabbitMQ
10. apche tomcat
11. nginix
12. build and deploy
13. verify from browser

* Let’s get started

First we need to copy the Repo from github to our local machine

For this we need to git clone run command and URL for the Repo.

For this project we are working on the branch called **local-setup**.

In this branch there will be a vagrant file( which will launch all the VM)

1. launch git bash cd into the directory where you want to store the Repo
2. run command **git clone** [**https://github.com/devopshydclub/vprofile-project**](https://github.com/devopshydclub/vprofile-project)
3. this will clone the entire repo in the local machine
4. Cd into the project repo
5. You can move to local-setup branch by using command **git checkout local-setup**
6. Vagrant file is store in vprofile-project\vagrant\Manual\_provisioning you can use tree command to check out the directory structure Tree command in git bash with **tree.com //a**
7. cd into folder by using **cd vagrant\Manual\_provisioning** & **ls** to seethe files
8. Vagrant file will be used to create 5 VM for 5 Different services ie. Nginx, Tomcat, RabbitMQ, MemcacheD, DB
9. Vagrant file is added with name and os with private IP you can change the IP address also if you want
10. Nginx- web01- ubuntu/xenial64- 192.168.33.11 -1GB ram
11. Tomcat - app01 - centos7- 192.168.33.12 – 2GB ram
12. RabbitMQ- rmq01- centos7- 192.168.33.16 – 512 MB ram
13. MemcacheD- mc01- centos7- 192.168.33.14 – 512 MB ram
14. DataBase - db01- centos7- 192.168.33.15 – 512 MB ram

We need at least 2.5 GB of ram

1. Installing vagrant host manager plug-in will help to update hostname in each machine in etc host fileit will be mapped
2. Installation Vagrant – **Vagrant init** to initialize the vagrant
3. To run the vagrant file you need to be in the same directory as the vagrant file before running this file use this command check and validate the vagrant file **vagrant validate** after that to run the setup you need to use **Vagrant up** command.

{Issue:- Unknown configuration section 'hostmanager'

Cause:- I have defined the hostmanager in my Vagrantfile but it’s an additional plug-in which you need to install first before you can make the use of it.

Resolution:- Run the following command to resolve this issue - **vagrant plugin install vagrant-hostmanager**

Issue - next InitializeSecurityContext failed: Unknown error (0x80092012) - The revocation function was unable to check revocation for the certificate.

The problem was my antivirus **(Kaspersky)** that was preventing me from adding the box. I deactivated the anti virus and the command worked:}

**vagrant plugin list –** to see list of all the plug-in installed

**( vagrant plugin install hostmanager –** to install hostnamanger **vagrant plugin uninstall hostmanager –** to uninstall host manager )

1. You can see that the Oracle VirtualBox will be having 5 different VM with the respective name and specification as per vagrant file
2. You can ssh with the command **vagrant ssh <hostname>**
3. Ping the server from the VM and test if all are good
4. Now its time to provision the services in each VM we will setup in following order
5. MySQL ( DB SVC)
6. MemcacheD (DB caching SVC)
7. RabbitMQ( Message broker/queue SVC)
8. Tomcat ( Application SVC)
9. Nginx (Web SVC)
10. Lets strat with MySQL
11. Log in to DB01 with **vagrant ssh db01**
12. Login as **sudo –i**
13. Update the package by **yum update –y**
14. Now we will setup a variable as we will use this as a db password so you can change as per your need **DATABASE\_PASS-‘admin123’**
15. **echo $DATABASE\_PASS** this will make a variable on temporary basis
16. we can use this as a permanent by adding it into the /etc/profile fileby **vi /etc/profile** add it and save it **source /etc/profile**
17. now we can setup a additional repository of redhat by using **yum install epel-release -y**
18. we need to install git and mariadb  **yum install git mariadb-server**
19. we can start mariadb service by **systemctl start mariadb** and **systemctl enable mariadb** and check its status by **systemctl status mariadb**
20. now we will run **mysql\_secure\_installation** for installing the mysql
21. it will ask for the current password we have not set any password so hit enter
22. the it will ask for setting up root password use the same password as we have saved in the database pass variable
23. say yes to remove anonymous user
24. say no to disallow root login remotely
25. say yes to remove test db
26. say yes to reload privilege table now
27. we can test the password with **mysql –u root –p** enter password if everything is ok the exit
28. now we need to clone source code in the db server use the git clone command **git clone** [**https://github.com/devopshydclub/vprofile-project**](https://github.com/devopshydclub/vprofile-project)
29. cd into directory as this will have a mysql db file which will initilse the db automatically with the table attribute and data **cd vprofile-project\src\main\resources**
30. we need to create the DB and also create a admin user

**mysql -u root –p”$DATABASE\_PASS”**

mysql> **create database accounts**

mysql> **grant all privileges on accounts.\* TO 'admin'@% identified by 'admin123' ;**

mysql> **FLUSH PRIVILEGES;**

mysql> **exit**

1. now run the db backup file to create the tables and add the data into the table

**mysql -u root -padmin123 accounts < src/main/resources/db\_backup.sql**

1. login into database and verify **mysql -u root -padmin123 accounts**

mysql> **show databases;**

mysql> **use accounts;**

mysql> **show tables;**

1. **logout** from db server and log In to memcacheD server using **vagrant ssh mc01**
2. **sudo –i**
3. **yum update –y**
4. **yum install epel-release –y**
5. **yum install memcached –y**
6. we can start memcached service by **systemctl start memcached** and **systemctl enable memcached** and check its status by **systemctl status memcached**
7. we need to run the command to enable memcached to listen to tcp port 11211 and udp port 11111 ( # vi /etc/services service-name port/protocol [# comment] ) **memcached -p 11211 -U 11111 -u memcached –d**
8. we can validate the ports by using **ss –tunlp |grep 11211**

**ss –tunlp |grep 11111**

1. **logout**  from db server and log In to RabbitMQ server using **vagrant ssh rmq01**
2. **sudo –i**
3. **yum update –y** ( necessary to run this on rabbit mq)
4. **yum install epel-release –y**
5. **yum install socat –y** (dependency is a command line based utility that establishes two bidirectional byte streams and transfers data between them )
6. **yum install erlang –y** ( dependency Erlang is a programming language using which rabbit mq is developed )
7. **yum install wget –y** ( to download rabbitmq file)
8. **wget** [**http://packages.erlang-solutions.com/erlang-solutions-2.0-1.noarch.rpm**](http://packages.erlang-solutions.com/erlang-solutions-2.0-1.noarch.rpm)
9. **sudo rpm -Uvh erlang-solutions-2.0-1.noarch.rpm**
10. **curl -s https://packagecloud.io/install/repositories/rabbitmq/rabbitmq-server/script.rpm.sh | sudo bash** (download pkg curl is command line tool that developers use to transfer data to and from a server )
11. **yum install rabbitmq-server –y**
12. we can start rabbitmq service by **systemctl start rabbitmq-server** and **systemctl enable rabbitmq-server** and check its status by **systemctl status rabbitmq-server**
13. **logout**  from RabbitMQ and log In to tomcat server using **vagrant ssh app01**
14. **sudo –i**
15. **yum update –y**
16. **yum install epel-release –y**
17. **yum install java-1.8.0-openjdk –y** (JDK dependency)
18. **yum install git maven wget –y** (git, maven,wget depensency)
19. **wget** [**https://archive.apache.org/dist/tomcat/tomcat-8/v8.5.37/bin/apache-tomcat-8.5.37.tar.gz**](https://archive.apache.org/dist/tomcat/tomcat-8/v8.5.37/bin/apache-tomcat-8.5.37.tar.gz)(download pkg tomcat 8.5)
20. **tar xzvf apache-tomcat-8.5.37.tar.gz** (unzip file)
21. Now we have to create a user and make a dir as home directory we are going to run tomcat with systmctl command and start and stop service with it. For that so we have to setup user directory ownerships and systemctl file Add tomcat user **useradd --home-dir /usr/local/tomcat8 --shell /sbin/nologin** tomcat Copy data to tomcat homedir **cp -r /tmp/apache-tomcat-8.5.37/\* /usr/local/tomcat8/** Make tomcat user owner of tomcat home dir **chown -R tomcat.tomcat /usr/local/tomcat8**
22. Setup system ctl file **vi /etc/systemd/system/tomcat.service** & Update file with following content.
    1. [Unit]
    2. Description=Tomcat
    3. After=network.target
    4. [Service]
    5. User=tomcat
    6. WorkingDirectory=/usr/local/tomcat8
    7. Environment=JRE\_HOME=/usr/lib/jvm/jre
    8. Environment=JAVA\_HOME=/usr/lib/jvm/jre
    9. Environment=CATALINA\_HOME=/usr/local/tomcat8
    10. Environment=CATALINE\_BASE=/usr/local/tomcat8
    11. #for starting the service with systemctl start command it will use below script
    12. ExecStart=/usr/local/tomcat8/bin/catalina.sh run
    13. #for starting the service with systemctl start command it will use below script
    14. ExecStop=/usr/local/tomcat8/bin/shutdown.sh
    15. SyslogIdentifier=tomcat-%i
    16. [Install]
    17. WantedBy=multi-user.target
23. **# systemctl daemon-reload**
24. we can start tomcat service by **systemctl start tomcat** and **systemctl enable tomcat** and check its status by **systemctl status tomcat**
25. now we can clone the source code from gitrun command **git clone** [**https://github.com/devopshydclub/vprofile-project**](https://github.com/devopshydclub/vprofile-project)
26. We need to cd into vprofile-project directory and we need to be in local-setup branch we can check that with **git status** command or we can checkout to local-setup by using **git checkout local-setup** command
27. We need to connect the backend services for this we need to update the configuration file which located in **vi src/main/resources/application.properties**
28. Mvn install ( this will create our artefact veprofile-v2.war it stored in **cd target directory** )
29. Now we will remove the default application in rm –rf /usr/local/tomcat8/webapps/ROOT
30. Now copy the war file to the same directory with ROOT.war name **cp vprofile-v2.war /usr/local/tomcat8/webapps/ROOT.war**
31. After that we can start the tomcat server with **systemctl start tomcat**
32. **logout** from app server and log In to nginx server using **vagrant ssh web01**
33. as this is ubuntu machine **sudo apt update && sudo apt upgrade –y** for updating the latest version
34. **apt install nginx –y** ( install the nginx service)
35. we will create a conf file which will route the traffic to tomcat server at port 8080 and nginx will act as a load balancer
36. **vim /etc/nginx/sites-available/vproapp** and add the following configuration into the file

upstream vproapp {server app01:8080;

#this will add the backend service at this}server {

# it will listen the traffic at port 80 and add send it to upstream vproapps which are mention in the function listen 80;location / {proxy\_pass http://vproapp;}}

1. Remove default nginx conf  **rm -rf /etc/nginx/sites-enabled/default**
2. Create link to activate website **ln -s /etc/nginx/sites-available/vproapp /etc/nginx/sites-enabled/vproapp**
3. Restart Nginx **systemctl restart nginx**

* Validate Flow of execution

1. Setup all the tools
2. Clone Git repository
3. Cd into vagrant directory and bring up the VM
4. Validate
5. Setup all services
6. Mysql
7. memcacheD
8. rabbitMQ
9. apche tomcat
10. nginix
11. build and deploy
12. verify from browser

now get the web01 ip address with **ifconfig** and copy the ip address paste in the browser and use username admin\_vp and password admin\_vp if the login is successful then all the db server is connected are working fine you can use buttons in the web site **rabbitMq** for checking that service and **all user** for memcache

1. **vagrant destroy** for deleting the VM

* **some important vagrant commands**

**vagrant init** - This will create a file called Vagrantfile in your directory and it contains all the configuration settings of your VM.

**vagrant up** - to initialize the VM

**vagrant ssh** - This command will let you login into the VM with the default credentials via secure shell (ssh)

**Vagrant share** : Now that you have a web server running you can share the url of your app with your peers or clients for review. So you need a public ip for that and vagrant can generate a temporary public url by using ngrok internally. So download ngrok, unzip the archive to a folder and add that folder as a system environment variable. In my case the folder is located at C:/ngrok/ Now run the command vagrant share , you will be presented with a temporary url as shown in the image

**vagrant halt-** To stop the VM, use vagrant halt and to start again use vagrant up.

**vagrant suspend** -- suspends a virtual machine

**vagrant resume** -- resume a suspended machine

**vagrant snapshot -** What if you want to do some experiment on your VM and you want to take a backup

**vagrant status** - outputs status of the vagrant machine