VPROFILE PROJECT SETUP

## Prerequisite

1. Oracle VM Virtualbox
2. Vagrant
3. Vagrant plugins
   1. vagrant plugin install vagrant-hostmanager
   2. vagrant plugin install vagrant-vbguest
4. Git bash or equivalent editor

# VM SETUP

1. Clone source code.
2. Cd into the repository.
3. Switch to the local-setup branch.
4. cd into vagrant/Manual\_provisioning.

**Bring up vm’s**

$ vagrant up

Can check all the VM’s hostname with $cat /etc/hosts

# PROVISIONING

### Services

1. Nginx:
2. Tomcat

Web Service

Application Server

1. RabbitMQ

Broker/Queuing Agent

1. Memcache

DB Caching

1. ElasticSearch

Indexing/Search service

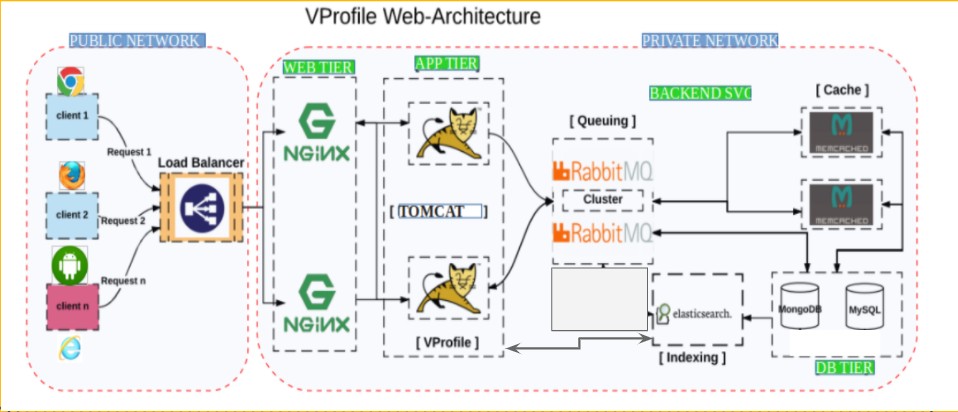
1. MySQL

SQL Database

Setup should be done in below mentioned order

1. MySQL (Database SVC)
2. Memcache (DB Caching SVC)
3. RabbitMQ (Broker/Queue SVC)
4. Tomcat (Application SVC)
5. Nginx (Web SVC)

Note- Good practice to switch to root user privileges with $sudo -i



## MYSQL Setup

### Login to the db vm

$ vagrant ssh db01

### Verify Hosts entry, if entries missing update the it with IP and hostnames

# cat /etc/hosts

### Update OS with latest patches

# yum update -y

* Good practice to patch bugs and vulnerabilities in OS before development

### Set Repository

# yum install epel-release -y

Set variable

* Temporary method: $DATABASE\_PASS=’admin123’ -> can use $echo $DATABASE\_PASS to call value
* Permanent method*:* edit profile configuration file with $vi /etc/profile and add the key-value pair and then read and execute file content with $source /etc/profile

### Install Maria DB Package

# yum install git mariadb-server -y

* MySQL package

### Starting & enabling mariadb-server

# systemctl start mariadb

# systemctl enable mariadb

### RUN mysql secure installation script.

# mysql\_secure\_installation

***NOTE****: Set db root password, I will be using* ***admin123 as password***

Set root password? [Y/n] Y New password:

Re-enter new password:

Password updated successfully! Reloading privilege tables..

... Success!

By default, a MariaDB installation has an anonymous user, allowing anyone to log into MariaDB without having to have a user account created for

them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a

production environment.

Remove anonymous users? [Y/n] Y

... Success!

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] n

... skipping.

By default, MariaDB comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed

before moving into a production environment.

Remove test database and access to it? [Y/n] Y

* Dropping test database...

... Success!

* Removing privileges on test database...

... Success!

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

Reload privilege tables now? [Y/n] Y

... Success!

Login to Database:

* mysql -u root -p
* mysql -u root -p"$DATABASE\_PASS" – sets username and password

### Download Source code & Initialize Database.

# git clone -b local-setup https://github.com/devopshydclub/vprofile-project.git

# cd *vprofile-project*

# cd src/main/resources -> there is an SQL file with queries in db\_backup.sql

Setup database

* + $mysql -u root -p"$DATABASE\_PASS" -e "create database accounts" - **creates a database named accounts**
  + $mysql -u root -p"$DATABASE\_PASS" -e "grant all privileges on accounts.\* TO 'admin'@'app01' identified by 'admin123' " -> **grants all privileges to admin user in app vm with password admin123**
  + $mysql -u root -p"$DATABASE\_PASS" accounts < src/main/resources/db\_backup.sql -> **executes sql file with given file path (depending on location)**
  + $mysql -u root -p"$DATABASE\_PASS" -e "FLUSH PRIVILEGES"-> **clears internal caches used my MariaDB**

Login to database

* + show databases;
  + use accounts;
  + show tables; - sql query has initialised database with tables shown

### Starting the firewall and allowing the mariadb to access from port no. 3306

# systemctl start firewalld

# systemctl enable firewalld

# firewall-cmd --get-active-zones

# firewall-cmd --zone=public --add-port=3306/tcp --permanent # firewall-cmd --reload

# systemctl restart mariadb

# MEMCACHE SETUP

Install, start & enable memcache on port 11211

#yum install epel-release -y

#yum install memcached -y #systemctl start memcached #systemctl enable memcached #systemctl status memcached

#memcached -p 11211 -U 11111 -u memcached -d **-> listen on tcp port and udp port respectively**

Validate port with ss -tunlp | grep 11211

### Starting the firewall and allowing the port 11211 to access memcached

# systemctl enable firewalld

# systemctl start firewalld

# systemctl status firewalld

# firewall-cmd --add-port=11211/tcp --permanent # firewall-cmd --reload

# memcached -p 11211 -U 11111 -u memcache -d

# RABBITMQ SETUP

### Login to the RabbitMQ vm

$ vagrant ssh rmq01

### Verify Hosts entry, if entries missing update the it with IP and hostnames

# cat /etc/hosts

### Update OS with latest patches

# yum update -y

### Set EPEL Repository

# yum install epel-release -y **-> extra packages for enterprise linux**

Install Dependencies *#sudo yum install wget -y #cd /tmp/*

#wget <http://packages.erlang-solutions.com/erlang-solutions-2.0-1.noarch.rpm> **-> download rpm**

#sudo rpm -Uvh erlang-solutions-2.0-1.noarch.rpm **-> install rpm**

#sudo yum -y install erlang socat **-> install erlang and socat**

Install Rabbitmq Server

*#curl -s https://packagecloud.io/install/repositories/rabbitmq/rabbitmq-server/script.rpm.sh | sudo bash* ***-> download shell script and execute simultaneously***

#sudo yum install rabbitmq-server -y **-> install rabbitmq server package**

### Start & Enable RabbitMQ

#sudo systemctl start rabbitmq-server #sudo systemctl enable rabbitmq-server #sudo systemctl status rabbitmq-server

Config Change

#sudo sh -c 'echo "[{rabbit, [{loopback\_users, []}]}]." > /etc/rabbitmq/rabbitmq.config' **-> echos output into specified file**

#sudo rabbitmqctl add\_user test test **-> creates user in rabbitmq with username and password**

#sudo rabbitmqctl set\_user\_tags test administrator **-> gives user admin privileges**

### Restart RabbitMQ service

# systemctl restart rabbitmq-server

### Enabling the firewall and allowing port 25672 to access the rabbitmq permanently

# systemctl start firewalld

# systemctl enable firewalld

# firewall-cmd --get-active-zones

# firewall-cmd --zone=public --add-port=25672/tcp --permanent # firewall-cmd --reload

# TOMCAT SETUP

### Login to the tomcat vm

$ vagrant ssh app01

### Verify Hosts entry, if entries missing update the it with IP and hostnames

# cat /etc/hosts

### Update OS with latest patches

# yum update -y

### Set Repository

# yum install epel-release -y

### Install Dependencies

# yum install java-1.8.0-openjdk -y **-> dependency for tomcat**

# yum install git maven wget -y **-> Maven is a build automation tool for Java projects and wget is a computer program that retrieves content from web servers**

### Change dir to /tmp

# cd /tmp/

### Download & Tomcat Package

# wget <https://archive.apache.org/dist/tomcat/tomcat-8/v8.5.37/bin/apache-tomcat-8.5.37.tar.gz>

# tar xzvf apache-tomcat-8.5.37.tar.gz **-> extracts the Tomcat software from this tar file**

Apache-tomcat-8.5.37 is the tomcat directory

### Add tomcat user

# useradd --home-dir /usr/local/tomcat8 --shell /sbin/nologin tomcat **-> adds tomcat home directory and username of tomcat**

# id tomcat **-> to switch to tomcat user**

### Copy data to tomcat home dir

# cp -r /tmp/apache-tomcat-8.5.37/\* /usr/local/tomcat8/ **-> copy all data to tomcat home directory**

### Make tomcat user owner of tomcat home dir

# chown -R tomcat.tomcat /usr/local/tomcat8 -R **-> changes privileges given to root user to tomcat user**

### Setup system for tomcat

Update file with following content.

vi /etc/systemd/system/tomcat.service **-> make this file to use systemctl to start/stop tomcat service**

[Unit] Description=Tomcat After=network.target

[Service] User=tomcat

WorkingDirectory=/usr/local/tomcat8 Environment=JRE\_HOME=/usr/lib/jvm/jre Environment=JAVA\_HOME=/usr/lib/jvm/jre Environment=CATALINA\_HOME=/usr/local/tomcat8 Environment=CATALINE\_BASE=/usr/local/tomcat8 ExecStart=/usr/local/tomcat8/bin/catalina.sh run ExecStop=/usr/local/tomcat8/bin/shutdown.sh SyslogIdentifier=tomcat-%i

[Install]

WantedBy=multi-user.target

# systemctl daemon-reload

# systemctl start tomcat

# systemctl enable tomcat

### Enabling the firewall and allowing port 8080 to access the tomcat

# systemctl start firewalld

# systemctl enable firewalld

# firewall-cmd --get-active-zones

# firewall-cmd --zone=public --add-port=8080/tcp --permanent

# firewall-cmd --reload

## CODE BUILD & DEPLOY (app01)

### Download Source code

# git clone -b local-setup https://github.com/devopshydclub/vprofile-project.git

### Update configuration

# cd vprofile-project

# vim src/main/resources/application.properties **-> important file for application**

# Update file with backend server details

### Build code

Run below command inside the repository (vprofile-project)

# mvn install **-> builds artifact**

Artifact can be found in target directory in vprofile-project **-> vprofile-v2.war**

### Deploy artifact

# systemctl stop tomcat

# rm -rf /usr/local/tomcat8/webapps/ROOT\* **-> replace default application in tomcat server with artifact**

# cp target/vprofile-v2.war /usr/local/tomcat8/webapps/ROOT.war

# systemctl start tomcat **-> extracts ROOT.war artifact into ROOT directory**

# chown tomcat.tomcat usr/local/tomcat8/webapps -R # systemctl restart tomcat

# NGINX SETUP

### Login to the Nginx vm

$ vagrant ssh web01

### Verify Hosts entry, if entries missing update the it with IP and hostnames

# cat /etc/hosts

### Update OS with latest patches

# apt update

# apt upgrade

### Install nginx

# apt install nginx -y

### Create Nginx conf file with below content

# vi /etc/nginx/sites-available/vproapp **-> custom nginx reverse proxy file**

*upstream vproapp { server app01:8080;*

*}*

*server { listen 80; location / {*

*proxy\_pass http://vproapp;*

*}*

*}*

* frontend listens on port 80
* route request to app01 server on port 8080- the default port that tomcat runs on

### Remove default nginx conf

# rm -rf /etc/nginx/sites-enabled/default

### Create link to activate website

# ln -s /etc/nginx/sites-available/vproapp /etc/nginx/sites-enabled/vproapp

### Restart Nginx

# systemctl restart nginx

To find the IP address use # ifconfig

**TROUBLESHOOTING VAGRANT UP:**

In your Vagrantfile:

config.vm.provider "virtualbox" do |vb|

vb.gui = true

end

Use the GUI to see the issue during vagrant up

https://stackoverflow.com/questions/23690124/vagrant-up-timeout