Apache Guacamole Setup

Pre-setup

Install Maven

sudo wget <http://mirror.olnevhost.net/pub/apache/maven/maven-3/3.9.9/binaries/apache-maven-3.9.9-bin.tar.gz>

sudo tar xzvf apache-maven-3.9.9-bin.tar.gz

Git

For transferring files from GH to EC2 instance

sudo dnf install git -y

Python

To run scripts

sudo dnf install python -y

Maven

sudo dnf install maven -y

Git clone

sudo git clone https://github.com/nickadeamus/scripts.git (installs dependencies)

python scripts/dependencies\_guac.py

Install Tomcat (webserver)

cd to /opt

sudo wget <https://dlcdn.apache.org/tomcat/tomcat-11/v11.0.1/bin/apache-tomcat-11.0.1.tar.gz> (periodically update)

sudo tar xzvf apache-tomcat-11.0.1.tar.gz

sudo mv apache-tomcat-11.0.1 tomcat (should rename thel untar-ed file to “tomcat”)

sudo rm -R apache-tomcat-11.0.1.tar.gz

Step 1: Create a Tomcat User

If you haven't already created a user for Tomcat, you can do so:

Create tomcat user + group

sudo useradd tomcat

Step 2: Edit the systemd Service File

Create Tomcat service file:

sudo vi /etc/systemd/system/tomcat.service

[Unit]

Description=Tomcat servlet container

After=network.target

[Service]

Type=forking

User=tomcat

Group=tomcat

Environment=JAVA\_HOME=/usr/lib/jvm/jre-22

Environment=CATALINA\_PID=/opt/tomcat/temp/tomcat.pid

Environment=CATALINA\_HOME=/opt/tomcat

Environment=CATALINA\_BASE=/opt/tomcat

ExecStart=/opt/tomcat/bin/startup.sh

ExecStop=/opt/tomcat/bin/shutdown.sh

Restart=on-failure

[Install]

WantedBy=multi-user.target

[Install]

WantedBy=multi-user.target

Make .sh files executable

sudo chmod +x /opt/tomcat/bin/startup.sh /opt/tomcat/bin/shutdown.sh

Edit the .bash\_profile file for the tomcat user:

sudo vi /home/tomcat/.bash\_profile

Add:

export CATALINA\_HOME=/opt/tomcat

Edit server.xml to allow port 80 connections

sudo nano /opt/tomcat/conf/server.xml

~~restart tomcat~~

~~sudo systemctl restart tomcat~~

**~~setenv.sh~~**

~~sudo touch /bin/setenv.sh~~

~~must be in var/lib/tomcat/bin~~

~~Install Java JDk~~

~~sudo wget~~ [~~https://download.oracle.com/java/23/latest/jdk-23\_linux-x64\_bin.rpm~~](https://download.oracle.com/java/23/latest/jdk-23_linux-x64_bin.rpm)

~~sudo rpm -ivh jdk-23\_linux-x64\_bin.rpm~~

Step 3: Change ownership of the Tomcat directory to the tomcat user:

sudo chown -R tomcat:tomcat tomcat

Change mod for .service file

sudo chmod u=rwx,g=rw,o=r /etc/systemd/system/tomcat.service

~~chown -R tomcat: /usr/local/tomcat~~

After making changes to the service file, reload systemd:

sudo systemctl daemon-reload

---------------------------------------------------------------------------------------------------------------------------------------

To run the Apache Tomcat service as a non-root user (e.g., tomcat), follow these steps to ensure the service is instantiated by the tomcat user:

Step 4: Start and Enable the Tomcat Service

Now you can start the Tomcat service, and it will run under the tomcat user:

sudo systemctl start tomcat

sudo systemctl enable tomcat

Step 5: Check the Status

To verify that Tomcat is running under the correct user:

sudo systemctl status tomcat

You should see that the Main PID is associated with the tomcat user.

Install Guac

<https://guacamole.apache.org/doc/gug/installing-guacamole.html>

Required dependencies

In order to build guacamole-server, you will need Cairo, libjpeg (or libjpeg-turbo), libpng, and libuuid (or the OSSP UUID library). These libraries are strictly required in all cases - Guacamole cannot be built without them.

Cd /

sudo git clone <https://github.com/apache/guacamole-server.git>

cd guacamole-server/

sudo autoreconf -fi

sudo ./configure --with-init-dir=/etc/init.d

sudo make

sudo make install

cd ..

sudo git clone <https://github.com/apache/guacamole-client.git>

cd /guacamole-client

sudo mvn package

sudo cp guacamole/target/guacamole-1.6.0.war /opt/tomcat/webapps/guacamole.war

sudo cp guacamole-1.5.5.war /opt/tomcat/webapps/guacamole.war

/etc/init.d/guacd start

sudo systemctl daemon-reload

sudo systemctl start tomcat

sudo systemctl enable tomcat

Creating a directory named .guacamole, within the home directory of the user running the servlet container. This directory will automatically be used for GUACAMOLE\_HOME if it exists. For us that is tomcat

**/etc/guacamole/guacamole.properties**

**/etc/guacamole/user-mapping.xml**

Using Guacamole with Dynamic Provisioning (Automating Connections)

A. Dynamic User and Connection Management:

Rather than manually editing the user-mapping.xml file each time you spin up a new EC2 instance, you could automate the creation and management of user connections using a database-based backend or another dynamic configuration method.

Install PostgreSQL server

sudo dnf install postgresql15.x86\_64 postgresql15-server -y

initialize

sudo postgresql-setup –initdb

start and check

sudo systemctl start postgresql  
sudo systemctl enable postgresql  
sudo systemctl status postgresql

(LAST) change postgres password

sudo passwd postgres

iamcomplete1234

Extension (download as postgres user)

postgresql/guacamole-auth-jdbc-postgresql-1.5.5.jar

sudo wget <https://downloads.apache.org/guacamole/1.5.5/binary/guacamole-auth-jdbc-1.5.5.tar.gz>

tar -xzvf guacamole-auth-jdbc-1.5.5.tar.gz

placed in: GUACAMOLE\_HOME/extensions

create .guacamole dir in /home/tomcat

sudo mkdir /home/tomcat/.guacamole

create dirs. in home/tomcat/.guacamole/lib and /home/tomcat/.guacamole/extensions

sudo mkdir /home/tomcat/.guacamole/lib

sudo mkdir /home/tomcat/.guacamole/extensions

sudo cp -r /home/ec2-user/guacamole-auth-jdbc-1.5.5/ /home/tomcat/.guacamole/extensions

SQL schema scripts

/home/tomcat/.guacamole/extensions/guacamole-auth-jdbc-1.5.5/postgresql/schema/

JDBC driver:

wget <https://jdbc.postgresql.org/download/postgresql-42.7.3.jar>

sudo cp -r /home/ec2-user/postgresql-42.7.3.jar /home/tomcat/.guacamole/lib

placed in: GUACAMOLE\_HOME/lib.

change to postgres user, create user and DB

su – postgres (TYPE IN MANUALLY)

psql

add postgres as sudoer

MUST BE DONE AS postgres USER

Also chars are odd here

Create Guac DB

createdb guacamole\_db

**$** ls /home/tomcat/.guacamole/extensions/guacamole-auth-jdbc-1.5.5/postgresql/schema/

001-create-schema.sql 002-create-admin-user.sql

**$** cat /home/tomcat/.guacamole/extensions/guacamole-auth-jdbc-1.5.5/postgresql/schema/001-create-schema.sql /home/tomcat/.guacamole/extensions/guacamole-auth-jdbc-1.5.5/postgresql/schema/002-create-admin-user.sql | psql -d guacamole\_db -f –

sudo chmod 666 /home/tomcat/.guacamole/extensions/guacamole-auth-jdbc-1.5.5/postgresql/schema/\*.sql

CREATE TYPE

CREATE TYPE

CREATE TYPE

CREATE TABLE

CREATE INDEX

...

INSERT 0 1

INSERT 0 4

INSERT 0 3

GRANT RIGHTS

psql -d guacamole\_db

psql (9.3.6)

Type "help" for help.

guacamole=# CREATE USER guac\_admin WITH PASSWORD 'aliens1234';

CREATE ROLE

guacamole=# GRANT SELECT,INSERT,UPDATE,DELETE ON ALL TABLES IN SCHEMA public TO guac\_admin;

GRANT

guacamole=# GRANT SELECT,USAGE ON ALL SEQUENCES IN SCHEMA public TO guac\_admin;

GRANT

guacamole=# \q

\q

change IP connections

cd

#listen\_addresses = ‘\*’ (specify IPs, or ‘localhost’)

change hba\_conf auth method

sudo vi /var/lib/pgsql/data/pg\_hba.conf

change local to md5

restart

sudo systemctl restart postgresql

try login

psql -U guac\_admin -d guacamole\_db

Installing database authentication

Guacamole extensions are self-contained .jar files which are located within the GUACAMOLE\_HOME/extensions directory. To install the database authentication extension, you must:

Create the GUACAMOLE\_HOME/extensions and GUACAMOLE\_HOME/lib directories, if they do not already exist.

Copy guacamole-auth-jdbc-mysql-1.5.5.jar or guacamole-auth-jdbc-postgresql-1.5.5.jar or guacamole-auth-jdbc-sqlserver-1.5.5.jar within GUACAMOLE\_HOME/extensions, depending on whether you are using MySQL/MariaDB, PostgreSQL, or SQL Server.

Copy the JDBC driver for your database to GUACAMOLE\_HOME/lib. Without a JDBC driver for your database, Guacamole will not be able to connect and authenticate users.

Configure Guacamole to use database authentication, as described below.

You will need to restart Guacamole by restarting your servlet container in order to complete the installation. Doing this will disconnect all active users, so be sure that it is safe to do so prior to attempting installation. If you do not configure the database authentication properly, Guacamole will not start up again until the configuration is fixed.

Additional properties must be added to **guacamole.properties** for Guacamole to properly connect to your database. These properties are specific to the database being used, and must be set correctly for authentication to work.

The properties absolutely required by the database authentication extension are relatively few and self-explanatory, describing only how the connection to the database is to be established, and how Guacamole will authenticate when querying the database:

Example:

# PostgreSQL properties

postgresql-hostname: localhost

postgresql-**database**: guacamole\_db

postgresql-username: guac\_admin

postgresql-password: aliens1234

password enforcement

postgresql-user-password-min-length: 8

postgresql-user-password-require-multiple-case: true

postgresql-user-password-require-symbol: true

postgresql-user-password-require-digit: true

postgresql-user-password-prohibit-username: true

postgresql-user-password-min-age: 7

postgresql-user-password-max-age: 90

postgresql-user-password-history-size: 6

postgresql-default-max-connections: 1

postgresql-default-max-group-connections: 1

postgresql-auto-create-accounts: true

Guacamole will only reread **guacamole.properties** and load newly-installed extensions during startup, so your servlet container will need to be restarted before the database authentication will take effect. Restart your servlet container and give the new authentication a try.

~~sudo yum install postgresql-jdbc~~

**~~Option 3: Using maven (if you're working with Java projects)~~**

~~If you're using Maven in your project and prefer to add the PostgreSQL JDBC driver as a dependency, you can include the following dependency in your pom.xml file:~~

~~xml~~

~~Copy code~~

~~<dependency>~~

~~<groupId>org.postgresql</groupId>~~

~~<artifactId>postgresql</artifactId>~~

~~<version>42.6.0</version> <!-- use the latest version -->~~

~~</dependency>~~