**Part 0: General System and QEMU setup**

SYSTEM REQUIREMENTS FOR CYBERLAB-

Gigabit network and server with Virtualization capabilities

1 simultaneous user: 16 GB Ram, 16 CPU Threads, 200GB Storage

For each additional simultaneous user: 12 GB Ram, 12 CPU Threads, 60-80GB Storage

Install QEMU and Libvirt to the system and ensure they are running (systemctl status libvirtd) and libvirtd-tcp.socket is running and enabled at boot. Install any QEMU modules and architectures required by the labs you will be running.

Make sure to install Apache Tomcat (Java application server) and MariaDB (MySQL DB Server) on your server and configure them as needed. MySQL only needs to be accessed by local components such as Apache Guacamole

**Part 1: Apache Guacamole Setup**

**Step A: Download**  
  
Go to <https://guacamole.apache.org/releases/> and select the latest version. This documentation assumes you are using Apache Guacamole 1.5.3 from July/August 2023

On your server create a temp directory in your home folder (or where ever you want)  
*mkdir guacamole-prep && cd guacamole-prep*

Download the Apache Guacamole SERVER source code from the version you selected. Right click and copy the link and run it in the terminal with wget like so  
*wget* [*https://apache.org/dyn/closer.lua/guacamole/1.5.3/source/guacamole-server-1.5.3.tar.gz?action=download*](https://apache.org/dyn/closer.lua/guacamole/1.5.3/source/guacamole-server-1.5.3.tar.gz?action=download)

NOTE For some reason my download name was messed up. You can fix it by making sure the file has the .gz extension on the end. I ran this command to fix it.  
  
*mv guacamole-server-1.5.3.tar.gz\?action\=download guacamole-server-1.5.3.tar.gz*  
  
Extract it with   
*tar -xzf guacamole-server-1.5.3.tar.gz*

Move into the guacamole server directory with (make sure the version is correct for your command)  
*cd guacamole-server-1.5.3*

**Step B: Compile**

DEBIAN/UBUNTU (May not be complete)

sudo apt install libavcodec-dev libavformat-dev libavutil-dev libswscale-dev freerdp2-dev libpango1.0-dev libssh2-1-dev libtelnet-dev libvncserver-dev libwebsockets-dev libpulse-dev libssl-dev libvorbis-dev libwebp-dev

RHEL/FEDORA (Tested on Fedora 38 Workstation):

sudo dnf install libpng-devel libjpeg-turbo-devel cairo-devel libuuid-devel freerdp-devel pango-devel libssh2-devel libtelnet-devel libvncserver-devel libvorbis-devel libwebsockets-devel libwebp-devel ffmpeg-devel

If you get an error such as “configure: error: "Cairo is required for drawing instructions"” then use dnf to search for the required package. It is often the name -devel such as cairo-devel.

Run this command to prepare the files and configure

*autoreconf -fi*

*CFLAGS=-Wno-error ./configure --with-init-dir=/etc/init.d*

Make sure ALL of the “Library status” options say yes except for wsock32. Make sure all protocol support is yes and make sure all services/tools are yes.

Next, run the make command. To speed it up put -jN where N is the number of CPU cores. For example, My laptop has 8 cores so I use -j8. Run make install to install it.

*make -j8*

*make install*

**Step C:** Apache Tomcat Installation for RHEL/Fedora

Download Tomcat9 and extract it.

*export VER="9.0.74"*

*wget* [*https://archive.apache.org/dist/tomcat/tomcat-9/v$*](https://archive.apache.org/dist/tomcat/tomcat-9/v$)*{VER}/bin/apache-tomcat-${VER}.tar.gz*

*tar xvf apache-tomcat-$VER.tar.gz*

Move the tomcat data that was extracted to the correct directory

*sudo mv apache-tomcat-${VER} /usr/libexec/tomcat9*

Setup Tomcat user and group

*sudo groupadd --system tomcat*

*sudo useradd -M -d /usr/libexec/tomcat9 -g tomcat tomcat*

Set permissions for the Tomcat directory

*sudo chown -R tomcat:tomcat /usr/libexec/tomcat9*

Fix SELinux for Tomcat

*sudo restorecon -rv /usr/libexec/tomcat9/*

Create the Tomcat systemd file. Copy the whole block from the sudo to the EOF at the end.

*sudo tee /etc/systemd/system/tomcat9.service<<EOF*

*[Unit]*

*Description=Apache Tomcat 9*

*Documentation=http://tomcat.apache.org/tomcat-9.0-doc/*

*After=network.target syslog.target*

*[Service]*

*User=tomcat*

*Group=tomcat*

*Type=oneshot*

*ExecStart=/usr/libexec/tomcat9/bin/startup.sh*

*ExecStop=/usr/libexec/tomcat9/bin/shutdown.sh*

*RemainAfterExit=yes*

*[Install]*

*WantedBy=multi-user.target*

*EOF*

Reload the Daemon and Start tomcat and check it

*sudo systemctl daemon-reload*

*sudo systemctl enable tomcat9.service*

*sudo systemctl start tomcat9.service*

*sudo systemctl status tomcat9.service*

Fix Firewall if needed

*sudo firewall-cmd --add-port=8080/tcp*

*sudo firewall-cmd --reload*

**Step D:** Install the Apache Guacamole WAR

Download the WAR and rename it

*wget https://apache.org/dyn/closer.lua/guacamole/1.5.3/binary/guacamole-1.5.3.war?action=download*

*mv guacamole-1.5.3.war?action=download guacamole.war*

Move the Guacamole WAR to the correct directory and set the permission.

*sudo cp guacamole.war /usr/libexec/tomcat9/webapps*

*sudo chown -R tomcat:tomcat /usr/libexec/tomcat9/webapps/guacamole.war*

Prepare the guacamole directory (if it does not exist)

*sudo mkdir /etc/guacamole*

*sudo touch /etc/guacamole/guacamole.properties*

*sudo chown -R tomcat:tomcat /etc/guacamole*

Restart tomcat9 and configure MySQL authentication

*sudo systemctl restart tomcat9*

*sudo service start guacd*

**Step E:** MySQL Authentication Database

Run *mkdir /etc/guacamole/extensions/*

Run *mkdir /etc/guacamole/lib*

Follow the instructions here:

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

**NOTE:** Use the MySQL Driver if the MariaDB one is not working. Be sure to download the Java (platform independent) driver.

Go to <http://localhost:8080/guacamole> and login with username and password *guacadmin.* Create a new user called *cyberlab* and give it full admin permissions. Set a strong password and keep it handy. Log out and back into the *cyberlab* user and delete the guacadmin user.   
  
**NOTE:** For security, you can set the username to whatever you want just be sure to change it in the Cyberlab daemon configuration.  
  
**Complete guacamole.properites file in /etc/guacamole:**  
  
*# Hostname and port of guacamole proxy*

*guacd-hostname: ::1*

*guacd-port: 4822*

*# Location to read extra .jar's from*

*lib-directory: /etc/guacamole/lib*

*# Authentication provider class*

*auth-provider: net.sourceforge.guacamole.net.auth.mysql.MySQLAuthenticationProvider*

*# MySQL properties*

*mysql-hostname: localhost*

*mysql-database: guacamole\_db*

*mysql-username: guacamole\_user*

*mysql-password: PASSWORD*

*mysql-server-timezone: America/New\_York*

*mysql-ssl-mode: disabled*

**Part 2: Apache Webserver Setup**

Test

**Part 3: Python Configuration and Setup**

**Daemon:**

The daemon is the flask app that acts as a control platform for the front end. The front end sends HTTP JSON API requests to the Daemon running on one of one to many physical nodes. For a single server setup The Frontend and a Daemon can be installed and configured on a single machine.

Dependency:

Ubuntu: sudo apt install xvfb  
RHEL: sudo dnf install xorg-x11-server-Xvfb  
  
DAEMON PARAMETERS:

FrontEnd Key: Random security key to securely link the daemon and front end.

Security: Either None or SSL. Best practice is to use SSL in production

APIDomain: The (sub)domain that you have set for this daemon application. Used for SSL

GuacDomain: The (sub)domain that you are using for Apache Guacamole. Used for SSL

Storage: The Location of lab assets (usually disk images) used to build a lab.

CPUCores: The available CPU cores on the server

RAMGB: The amount of RAM available in gigabytes

Priority: The priority of this daemon when allocating resources to users.

**NOTE:** The Daemon manages MySQL, Guacamole, and Libvirt locally and therefore does not require SSL for MySQL and Libvirt. It is highly recommended that applications accessed remotely such as Guacamole and the API (Sub)Domains are secured with SSL.

INSTALL:

Step 1: Ensure Python 3.6+ is installed with pip3.

Step 2: Run the install script to create the daemon user and group and setup the flask app in an venv as well as install the systemd profile. **THIS NEEDS TO BE RUN AS ROOT**

Step 3: Configure the DAEMON PARAMETERS in the main program file. They are at the top. You may need to run your text editor with sudo to edit the file.

Step 4: Enable the systemd service and use systemd to Start the daemon.

Step 5: Check the systemd service to ensure that it has started correctly

Step 6: Configure the front-end Cyberlab server with the necessary parameters.