Install Guacamole with Docker

Learn to install Apache Guacamole with Docker using the JWetzell Docker image.

- Install Guacamole using Docker
- Setup OIDC for Guacamole

Install Guacamole using Docker

https://www.youtube.com/embed/DGw6P5Lkj-U

A really terrific RDP client that makes your remote machine desktops and environments available through your web browser. The docker image we'll be using was built off of the original Oznu image work, but has been updated to provide some of the more recent features.

JWetzell's image allows for the extended authentication extensions to be used, and this really adds a lot of great security to the offering. The docker-compose.yml is very straight-forward and clean.

Installation

What you'll need

- Docker-CE and Docker-Compose installed
- (optional) Portainer-CE
- (optional) NGinX Proxy Manager
- (optional) Authelia (for an additional security layer)
- About 10 minutes of your time.

Installing Docker-CE and Docker-Compose

You can easily install Docker-CE, Docker-Compose, Portainer-CE, and NGinX Proxy manager by using this quick install script I created and maintain on Github. Just use the command:

```
wget https://gitlab.com/bmcgonag/docker_installs/-/raw/main/install_docker_nproxyman.sh
```

To download the script to your desired host.

Change the permissions to make the script executable:

```
chmod +x ./install_docker_nproxyman.sh
```

and then run the script with the command:

```
./install_docker_nproxyman.sh
```

When run, the script will prompt you to select your host operating system, then will ask you which bits of software you want to install.

Simply enter 'y' for each thing you want to install.

At some point, you may be asked for your super user (sudo) password as well.

Allow the script to complete installation.

At this point, you might want to log out and back in, as this will allow you to use the `docker` and `docker-compose` commands without the need of sudo in front of them.

Installing Guacamole with Docker and Docker-Compose

Let's create a good base structure for keeping our docker organized, and easy to backup and restore if needed. We'll make a folder called "docker" that we'll keep everything in.

`mkdir docker`

Next, let's move into that folder, and continue working from there.

`cd docker`

Now, we'll make a new folder called "guacamole":

`mkdir guacamole`

Let's move into our new guacamole folder, and then we'll create a new file in that directory called "docker-compose.yml":

`cd guacamole`

`nano docker-compose.yml`

Inside that file we just opened, we want to copy and paste the code block below.

```
version: "3"
services:
  guacamole:
    image: jwetzell/guacamole
    container_name: guacamole
    volumes:
      - ./postgres:/config
    ports:
```

```
      - 8080:8080
volumes:
  postgres:
    driver: local
```

After pasting the code-block above into your nano text editor (use right-click >> paste, or CTRL + Shift +V to paste), you may want / need to adjust the port mapping to a different port. I find that port 8080 is a very common port, so I like to change it to something less common to avoid port conflicts with other containers I run.

To change a port mapping, change the port number on the left side of the colon only. Never change the right side, as the right side is what the container is expecting for the port to be. The left side, however, is the port for your host machine, and you can change that to any unused (open) port on your host.

In this case, I used port 8190 as my open port for guacamole.

so my port mapping section looks like:

```
    ports:
      - 8190:8080
```

Once you've made this change (if you want / need to), save the file with CTRL + O, then press Enter to confirm, and exit the nano text editor with CTRL + X.

We are ready to bring up our Guacamole Remote Access application.

In the terminal at a fresh prompt, simply enter the command:

`docker-compose up -d`

Allow the images to be pulled down, and the container to start. You should see a "done" message in the terminal when the process has completed.

You can use a tool like Portainer to easily check the logs. If you don't have portainer, not to worry. You can use the command

`docker-compose logs -f`

to view the logs of the container. We are simply checking for any errors in the logs that might indicate our install didn't work for some reason. Hopefully you'll find no errors, and you can now open a browser window and navigate to the IP address of your host on the port you set on the left side of the port mapping.

In my case it was `http://192.168.10.147:8190`

You should now be greeted with the guacamole login screen.

Login with the default credentials of:

- username: guacadmin
- password: guacadmin

Setting up a new Guacamole User and Deleting the Default User

In the Guacamole interface, you'll want to go to the upper right and click on the user avatar. This will present a drop-down menu of options. Select the 'Settings' option, and then you'll see a set of tabs with which you can setup the system.

The first thing I like to do is to create a new user (under the users tab), and make that user an Admin of the system by checking all of the check boxes for user permissions. I give the user a strong password, and save.

Next, I log out of the default "guacadmin" user, and login as my newly created user. Once logged in as the new user, I go back into "Settings >> Users" and I delete the "guacadmin" default user by clicking on the username for "guacadmin", scrolling to the bottom of the edit screen, and selecting the "Delete" button, then confirming.

You can now add more users if you have others who you'd like to have access to your Guacamole instance. I suggest, however, that you first add all of the Remote Connections you'll need, then add the users, as part of the User add process is selecting which machines / groups of machines each user should be able to access.

Setup Connections and Connection Groups

Think about the machines you'll be adding. If it's only for home use, and only 1 or 2 machines, you may not need to create connection Groups. If, however, you'll be adding multiple machines for home, work, clients, end-useer support, etc, then you may want to consider the best groupings of those machines, and first create some Connection Groups. This will allow you to organize your connectins in a way that makes them easier to locate later.

You might create Connection Groups like:

- Home / Personal
- Client Desktops
- Web Servers
- Document Storage Servers
- Media Servers
- End User Consoles

Really, you can group machines in so many ways, i couldn't possibly provide a decent list here. You can always add connections to a group later as well.

Adding a new Connection is as easy as selecting the Connections tab in the Settings area, and clicking 'New Connection'.

Now just select how you want to connect, and provide the necessary connection details (IP or Hostname, Port, User Credentials, etc), and then save. One really great feature is the "Clone" capability. If you're setting up a lot of machines that are essentially the same settings and just different IPs and User Credentials, it will be a real time saver to make your first Connection, save it, then open it to Edit mode, and scroll to the bottom to click the Clone button. As soon as you click clone, you are in the Cloned connection, and ready to edit it.

Support my Channel

Support my Channel and ongoing efforts through Patreon:

https://www.patreon.com/bePatron?u=234177

Setup OIDC for Guacamole

https://www.youtube.com/embed/WIyAGuMbjmY

Setting up a Domain Name and Reverse Proxy

I use NGinX Proxy Manager as my reverse proxy of choice. Feel free to use any reverse proxy you like, but my expectation is that you'll know how to configure it to match my settings as needed.

Open NGinX Proxy Manager, click on the Proxy Hosts option, then select ‘Add New Host’ from the top.

In the pop-up modal, enter the domain or sub-domain you want to use for your guacamole install.

> NOTE: You need to have ports 80 and 443 forwarded through your firewall to the host running NGinX Proxy Manger.

After typing out your domain name, press Enter or Tab to make sure it turns into a “chip” entry. Next move to the IP Address field, and enter the LAN IP address of your server running Guacamole. If you are running Guacamole on the same server as NGinX Proxy Manager, you can enter “localhost” in this field.

If you are running both Guacamole and NPM in the same docker network (which cannot be the default docker network) then you can also enter the Guacamole container name. You can find this by typing:

```
docker ps
```

in the terminal, and looking for the name entry.

I run Guacamole and NPM on different physical servers, so I'll be using the IP address of my Guacamole host server in the IP field.

Next, enter the port you used in the port mapping of your docker-compose.yml file on the left side. If you didn't change it, this port number will be 8080.

Now enable the options for ‘Block Common Exploits’ and ‘Websockets Support’.

Move to the ‘SSL’ tab, and select “Request a new certificate” from the drop-down menu. Next, enable the options for ‘Force SSL’ and ‘HTTPS/2 Support’. Make sure your email is filled in, and

enable the option to accept the LetsEncrypt Terms of Service.

Finally, click ‘Save’ and wait patiently. If all is setup properly, you'll be issued a LetsEncrypt certificate for your domain, and you'll be able to now access your Guacamole instance using the domain name you just setup. You should also be accessing it over SSL encrypted HTTPS. Awesome!

If you don't intend to use an OpenID Connect server for authentication, then you are set, and ready to start creating connections to any machines you want to access remotely.

Setup OIDC (OpenID Connect) for Your Guacamole Install

If you happen to run, or are thinking of starting to run your own authentication system, then being able to login with SSO becomes a huge time-saver, and blissful gift to your mind that's overburdened with tens or hundreds of passwords.

Today, I'll show you how to add OIDC to the Guacamole install we've just done. None-the-less, you should still go through the steps above first, and make a new administrative user account with the Guacamole login that is not the default set of credentials, and remove the default credentials as a solid measure of security.

When you've done that, and you are ready to move forward with OIDC, then read on.

Adding OIDC to the Guacamole Docker Container

First, we need to adjust our Guacamole container so it will include the necessary extension for OIDC to work for us.

After that we'll add some configuration values that are also necessary for Guacamole to recognize the OIDC Authentication server, and for that server to know where to send us to after a successful login.

First, we'll adjust our docker-compose.yml file. Make sure you are in the directory on the server where you put the Guacamole docker-compose.yml file, and open the file in a text editor. I use nano because it comes with most distros built in, but feel free to use VI, VIM, Emacs, or any other text editor you prefer.

```
nano docker-compose.yml
```

Now, move down below the ‘ports’ entries, and create a new section at the same level called ‘environment:’, then below that indent two spaces, and add the following:

```
- EXTENSIONS=auth-sso-openid
```

Once complete the full docker-compose.yml should look like the following:

```
version: "3"
services:
  guacamole:
    image: jwetzell/guacamole
    container_name: guacamole
    volumes:
      - ./postgres:/config
    ports:
      - 8080:8080
    environment:
      - EXTENSIONS=auth-sso-openid
    restart: unless-stopped
volumes:
  postgres:
    driver: local
```

Next, we need to adjust our Guacamole properties file. To do this, we'll edit the file located in the path ./postgres/guacamole/guacamole.propertiees.

If you change the volume mapping from what i have above, you'll need to find this file wherever you mapped the volume on your system.

`nano postgres/guacamole/guacamole.properties`

> NOTE: You may need to use sudo nano postgres/guacamole/guacamole.properties if you have not set the folder permissions for this folder to be owned by your user.

Inside this file, you'll likely see some settings already configured. Do not change these settings. All we need to do is add a few settings below them. The settings we want to add below them are shown below.

Feel free to copy and paste, but keep in mind you'll need to replace my place-holder values with the real values from your authentication system. I am using Authentik currently, and these values are made very easy to locate in the Provider I created for Guacamole.

An Aside for those using Authentik

If you are using Authentik, and aren't sure how to setup an OIDC provider, fear not, it's pretty straight forward.

Navigate to your Administrator settings area.

1. On the left, expand the Applications section, and choose the Providers option.
2. Create a new Provider.
3. Select the OpenID Connect (OIDC) Provider type, and click ‘Next’.
4. Give the Provider a name that makes sense for this application (e.g. Guacamole-OIDC).
5. Choose the default Authentication flow (unless you've setup a custom one, then feel free to use that).
6. Choose the default Authorization flow (explicit).
7. Leave it as ‘Confidential’.
8. In the Redirect URIs/Origin field, enter the URL of your Guacamole server (e.g. https://guac.mygreatdomain.org)
9. Click ‘Finish’.

1. Now click on the Application option on the left navigation menu.
2. Click ‘Create’.
3. Enter a Name for the application (e.g. Guacamole). Note the slug will auto populate based on what you enter in the Name field.
4. If you use permission groups for your applications, add the groups that can access the Guacamole server.
5. In the Provider drop-down select your freshly created Guacamole-OIDC provider.
6. Click Create.

Now you'll have all the values you need in order to get this running.

Get the Values

Note that for the `openid-group-claims-type` value, I entered ‘admins’. This is the only group I have in Guacamole. For your server you may want to set a lower level group, but this works for me as the only user on Guacamole.

```
openid-authorization-endpoint: https://auth.mygreatdomain.org/application/o/authorize/
openid-jwks-endpoint: https://auth.mygreatdomain.org/application/o/guacamole/jwks/
openid-issuer: https://auth.mygreatdomain.org/application/o/guacamole/
openid-client-id: some-super-Long-5trinG-0f-Ch4rac7ers
openid-redirect-uri: https://guac.mygreatdomain.org
openid-groups-claim-type: admins
extension-priority: openid
```

When you've added all of your values, you should have a file that looks like this:

```
postgresql-hostname: localhost
postgresql-port: 5432
```

```
postgresql-database: guacamole_db
postgresql-username: somegreatuserintheether
postgresql-password: an-awesome-strong-long-complex-crazy-but-memorable password

# ldap-hostname: ldap.example.net
# ldap-port: 389
# ldap-encryption-method: none
# ldap-max-search-results: 1000
# ldap-search-bind-dn:
# ldap-search-bind-password:
# ldap-user-base-dn: ou=people,dc=example,dc=net
# ldap-username-attribute: uid
# ldap-user-search-filter: (objectClass=*)

openid-authorization-endpoint: https://auth.mygreatdomain.org/application/o/authorize/
openid-jwks-endpoint: https://auth.mygreatdomain.org/application/o/guacamole/jwks/
openid-issuer: https://auth.mygreatdomain.org/application/o/guacamole/
openid-client-id: some-super-Long-5trinG-0f-Ch4rac7ers
openid-redirect-uri: https://guac.mygreatdomain.org
openid-groups-claim-type: admins
extension-priority: openid

enable-clipboard-integration: true
```

> NOTE: The LDAP information was already there in my file, so I left it. It was also already commented out, so there is no harm in leaving it.

Once you've added all of the values from your OIDC provider, save your changes with CTRL + O, then press Enter to confirm. Next, press CTRL + X to exit the nano editor.

Now, you need to go back to your docker/guacamole folder, and run the command:

`docker compose down`

After it comes down completely, bring it back up, and watch the log output as it starts.

`docker compose up -d && docker compose logs -f`

> NOTE: You can probably just do docker compose restart instead of the down and back up again, I just do it this wayto make sure it completely restarts.

This time, make sure you aren't seeing any errors in the log as it scrolls. If you have any values set that the properties file and extension can't read, you'll likely get errors and possibly a message about an exit code.

If you see this, stop the log output with CTRL + C, then bring down the container with:

```
docker compose down
```

Next, check all of your entries in the properties file again. Make sure you didn't add any extra spaces, quotation marks, or other characters, and ensure all the values are correct.

Also check the docker-compose and make sure you've got it all aligned properly.

Then bring your container back up with

```
docker compose up -d && docker compose logs -f
```

When successful, you should see no errors or exit code messages (possibly exit code 0), but not on this container as far as I recall.

Once you feel it's up and running properly, navigate to your FQDN (https://guac.mygreatdomain.org) and you should be presented with your Authentication system login prompt.

> NOTE: On Firefox, because of the caching it does, I initially got into a login loop, but logging into a private window or separate browser will usually get you in with no issues. After closing fFrefox completely, and then logging in, the login loop no longer occurred.

Creating a Guacamole Admin User for OIDC

If you've already brought up OIDC and tried to login, you may notice that you have a new user created vs. logging in as an admin user. This is because Guacamole creates a new user if it can't find a user to match on. Let's set up an admin user to match.

First to

```
docker compose down
```

Then go into the guacamole.properties file, and change the extension-priority value to be like this:

```
extension-priority: *, openid
```

Now bring Gucamole back up with

```
docker compose up -d
```

Log into Guacamole with the original admin user you created.

It's important to realize that Guacamole will use the username field for your OIDC user matching. You need to use the email address coming from your OIDC provider for Guacamole to match on.

If you don't create an admin user for your OIDC to match to, you'll have a hard time with Guacamole, so let's create one.

1. Go into Settings in Guacamole
2. Click on the 'Users' tab.
3. Create New User
4. In the username filed, enter the email address of the admin user that will be coming from your OIDC provider.
5. IMPORTANT! Do not enter anything in the password fields. In fact, do not tab or click into the password fields. If you enter this field, Guacamole will see it as "dirtied" and will not let you save the user with a blank password.
6. Go through the rest of the fields, and assign your admin user all permissions, and access to all Connections already created.
7. You can additionally add them to any admin groups you may have already created.
8. Click 'Save'.

One last time, you'll do

```
docker compose down
```

Wait for Guacamole to stop.

Change the Login to be "openid" only in the configuration file, then restart Guacamole with

```
docker compose up -d && docker compose logs -f
```