How to Install Apache Guacamole via Docker on Ubuntu 22.04

Apache Guacamole is a free and open source remote desktop gateway that allows you to remotely connect to your computer/server using different protocols such as SSH, RDP and VNC.

Apache Guacamole is maintained by the Apache Software Foundation and is under the Apache 2.0 License.

Apache Guacamole is a remote desktop gateway without a client. You can access Apache Guacamole using just a web browser from anywhere at any time. You should use Apache Guacamole if you have multiple remote operating systems with different protocols, such as Windows with RDP and Linux systems with VNC and SSH.

In this tutorial, you will install Apache Guacamole – Remote Desktop/Server Gateway – via Docker on an Ubuntu 22.04 server. This includes installing and configuring Nginx as a reverse proxy for Apache Guacamole. Finally, you will have Apache Guacamole running as a Docker container and securing the installation via an SSL/TLS certificate on the Nginx reverse proxy.

Prerequisites

To get started with this guide, you must have the following requirements:

1. A Linux server running Ubuntu 22.04 Server.
2. Non-root users have sudo/root admin privileges.
3. A domain name points to the server's IP address.

Once the requirements are ready, you can now start installing Apache Guacamole.

Install Docker Engine and Docker Compose

In this tutorial, you will run and install Apache Guacamole as a container service through Docker and Docker Compose. This example uses a fresh Ubuntu 22.04 server, so includes Docker and Docker Compose installation.

To get started, run the apt command below to install the basic dependencies. Enter y when prompted and press ENTER to continue.

```
sudo apt install ca-certificates curl gnupg lsb-release
```

Output:

```
root@test-server:~#
root@test-server:~# sudo apt install ca-certificates curl gnupg lsb-release
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
lsb-release is already the newest version (11.1.0ubuntu4).
ca-certificates is already the newest version (20211016ubuntu0.22.04.1).
ca-certificates set to manually installed.
curl is already the newest version (7.81.0-1ubuntu1.6).
gnupg is already the newest version (2.2.27-3ubuntu2.1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@test-server:~#
```


Next, run the command below to add the GPG key and repository for Docker packages.

```
sudo mkdir -p /etc/apt/keyrings curl -fsSL https://download.docker.com/linux/ubu
echo  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker
```

Output:

```
root@test-server:~#
root@test-server:~# sudo mkdir -p /etc/apt/keyrings
root@test-server:~#
root@test-server:~# curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
root@test-server:~#
root@test-server:~# echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
root@test-server:~#
```


Then, update and refresh your Ubuntu package index via the apt command below.

```
sudo apt update
```

Output:

```
root@test-server:~#
root@test-server:~# sudo apt update
Hit:1 https://download.docker.com/linux/ubuntu jammy InRelease
Hit:2 https://mirrors.edge.kernel.org/ubuntu jammy InRelease
Hit:3 https://mirrors.edge.kernel.org/ubuntu jammy-updates InRelease
Hit:4 https://mirrors.edge.kernel.org/ubuntu jammy-backports InRelease
Hit:5 https://mirrors.edge.kernel.org/ubuntu jammy-security InRelease
Reading package lists... Done
```


With the Docker repository added, you can now install the Docker engine and Docker Compose plugin using the apt command below. When prompted, type y , then press ENTER to continue.

```
sudo apt install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

Output:

Service Docker will start and activate automatically. You can verify the Docker service through the following systemctl command utility.

```
sudo systemctl is-enabled docker sudo systemctl status docker
```

You will receive a result indicating that the Docker service has been enabled and will automatically run on startup. And the status of the Docker service is running.

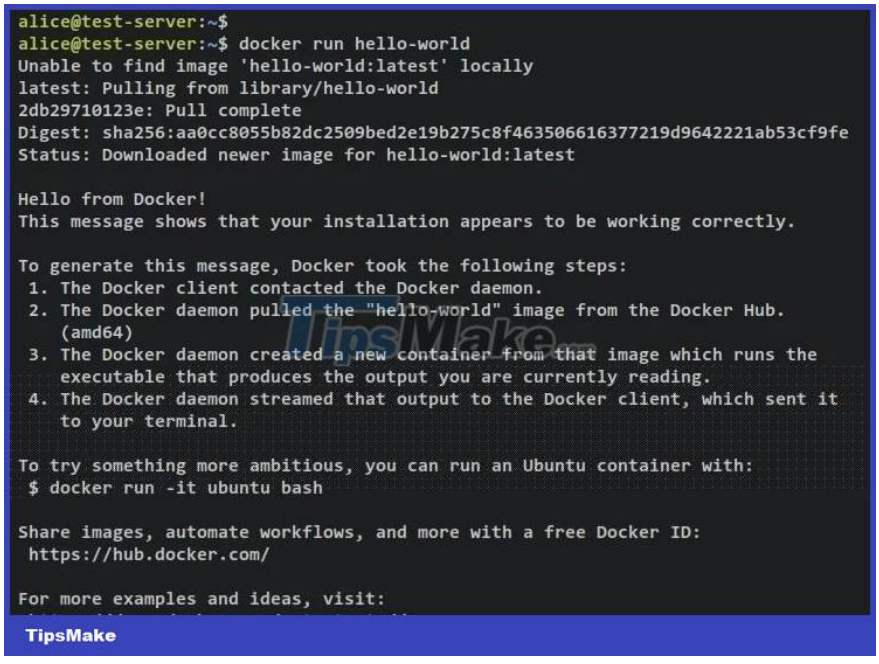
Finally, to allow non-root users to run Docker containers, you must add your user to the 'docker' group. Run the usermod command below to add your user to the 'docker' group. Also, remember to change the username to your user.

```
sudo usermod -aG docker alice
```

Now, you can log in as your user and run the Docker container via the command below.

```
su - alice docker run hello-world
```

On success, you will receive a hello-world notification from the Docker container as shown in the following screenshot.



After installing Docker and Docker Compose, you will next start creating a project directory to deploy Apache Guacamole.

Set up project directory

First, make sure you are logged in as a non-root user by running the following command.

```
su - alice
```

Create a new project directory '~/guacamole-server' and move your working directory into it.

```
mkdir -p guacamole-server; cd guacamole-server/
```

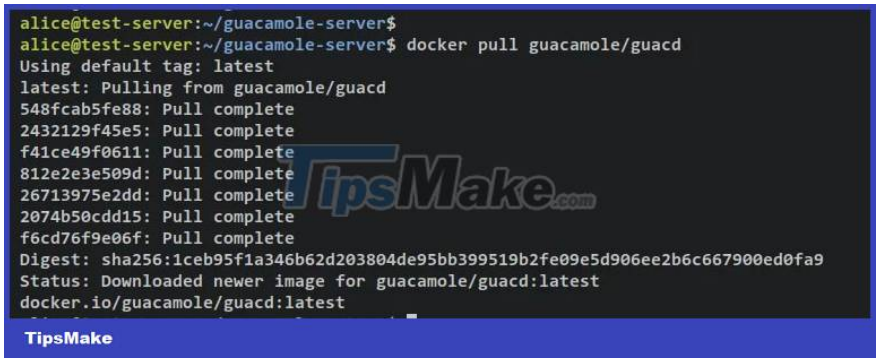
Then, in the '~/guacamole-server' directory , create a new directory 'init' and a 'docker-compose.yml' file.

```
mkdir -p init touch docker-compose.yml
```

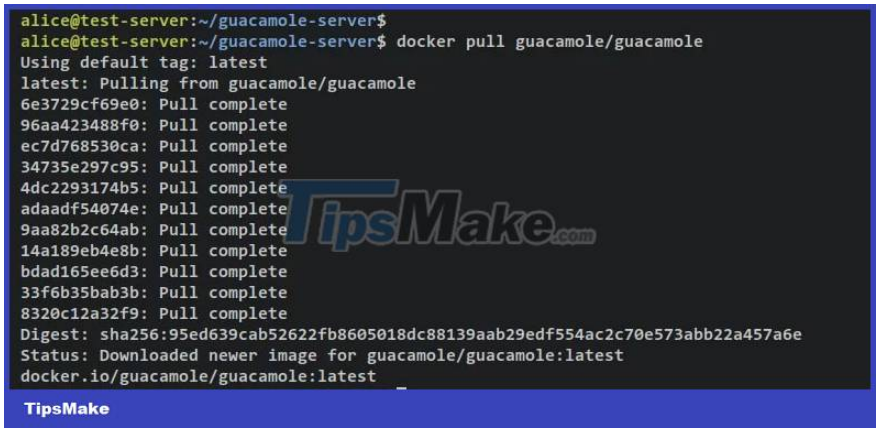
Next, run the following 'docker pull' command to download the Docker images needed for the Apache Guacamole installation. You will download 3 different images, guacd as the proxy manager, guacamole as the frontend of Apache Guacamole and postgres:13 will be used as the database backend for the Apache Guacamole container.

```
docker pull guacamole/guacd docker pull guacamole/guacamole docker pull postgres
```

Download the guacd image.



Download image guacamole.



Download the PostgreSQL 13 image.

```
alice@test-server:~/guacamole-server$
alice@test-server:~/guacamole-server$ docker pull postgres:13
13: Pulling from library/postgres
8740c948ffd4: Already exists
c8dbd2beab50: Already exists
05d9dc9d0fbd: Already exists
ddd89d5ec714: Already exists
f98bb9f03867: Already exists
0554611e703f: Already exists
64e0a8694477: Already exists
8b868a753f47: Already exists
29acea527529: Pull complete
04331b81a289: Pull complete
faa16d756995: Pull complete
8992a861a703: Pull complete
3a0e6483774c: Pull complete
Digest: sha256:1c36dd0c70477d7f7e472df3cddcdcc1c95526e74ca02b75e4a2c2c4ebd2f1db
Status: Downloaded newer image for postgres:13
docker.io/library/postgres:13
```


Once the necessary Docker images are downloaded, execute the following command to run the new guacamole container and run the 'initdb.sh' script to create the database schema for your deployment. You will create the guacamole database schema as 'init/initdb.sql'.

```
docker run --rm guacamole/guacamole /opt/guacamole/bin/initdb.sh --postgres > ini
```

Verify the contents of the guacamole database schema via the cat command below.

```
cat init/initdb.sql
```

Output:

```
alice@test-server:~/guacamole-server$
alice@test-server:~/guacamole-server$ mkdir -p init
alice@test-server:~/guacamole-server$
alice@test-server:~/guacamole-server$ docker run --rm guacamole/guacamole /opt/guacamole/bin/initdb.sh --postgres > init/initdb.sql
alice@test-server:~/guacamole-server$
alice@test-server:~/guacamole-server$ cat init/initdb.sql
--
-- Licensed to the Apache Software Foundation (ASF) under one
-- or more contributor license agreements.  See the NOTICE file
-- distributed with this work for additional information
-- regarding copyright ownership.  The ASF licenses this file
-- to you under the Apache License, Version 2.0 (the
-- "License"); you may not use this file except in compliance
-- with the License.  You may obtain a copy of the License at
--
--   http://www.apache.org/licenses/LICENSE-2.0
--
-- Unless required by applicable law or agreed to in writing,
-- software distributed under the License is distributed on an
-- "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
-- KIND, either express or implied.  See the License for the
-- specific language governing permissions and limitations
-- under the License.
--

--
-- Connection group types
--

CREATE TYPE guacamole_connection_group_type AS ENUM(
    'ORGANIZATIONAL',
    'BALANCING'
);

--
-- Entity types
--

CREATE TYPE guacamole_entity_type AS ENUM(
    'USER',
    'USER_GROUP'
);

--
-- Object permission types
--

CREATE TYPE guacamole_object_permission_type AS ENUM(
    'READ',
    'UPDATE',
```

Set up docker-compose.yml

Now that the necessary Docker images have been downloaded, you can start configuring the 'docker-compose.yml' script and setting up the Apache Guacamole installation.

Start by opening the 'docker-compose.yml' configuration file with the following nano editor command.

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

Add the following lines to the file.

```
version: '3.7' # networks networks: guacnet: driver: bridge # services services:
```

Save and close the 'docker-compose.yml' file when finished.

With this 'docker-compose.yml' script , you will create 3 containers/services as follows:

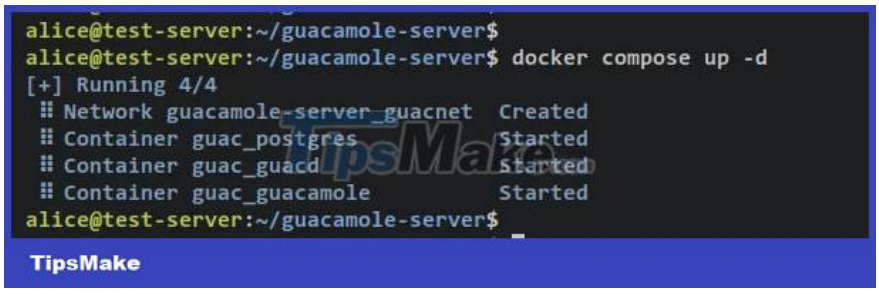
1. guacd – main component of Apache Guacamole will be used to proxy many protocols such as SSH, RDP, VNC, etc.
2. postgres – database backend for your Apache Guacamole installation. Your data will be stored in this container.
3. guacamole – Apache Guacamole web application connected to PostgreSQL and guacd services. This container will expose port 8080 on your server.

Start Apache Guacamole

Before starting, make sure you are in the 'guacamole-server' project directory. Then, run the following 'docker compose' command to create and start deploying Apache Guacamole.

```
docker compose up -d
```

You will get output like this – There are 3 different containers guac_postgres, guac_guacd and guac_guacamole created and started.



Verify the list of services/containers running on your Apache Guacamole project via the following command.

```
docker compose ps
```

If you see 'STATUS' as 'Up' then the container/service is running. On the 'PORTS' section , you will see the ports displayed by container for the host.

Container 'guac_guacamole' exposes TCP port 8080 on both the container and the Docker host. With this, you can access your Apache Guacamole installation.

Open a web browser and go to the server IP address, then port 8080 (ie: http://192.168.5.100:8080/). You will see the Apache Guacamole login page.

Log in via default user 'guacadmin' and password 'guacadmin'. Then click Login to confirm.

On success, you will get the Apache Guacamole user dashboard.

With that, confirm that the installation of Apache Guacamole via Docker and Docker Compose is complete and successful. However, for this tutorial, there are still some actions that need to be taken to secure your Apache Guacamole deployment.

Additionally, when troubleshooting errors when deploying Apache Guacamole, you can check the logs for each container via the 'docker compose' command below.

Basic usage of 'docker compose' to check logs.

```
docker compose logs docker compose logs SERVICE
```

Check logs for specific containers/services via 'docker compose' command .

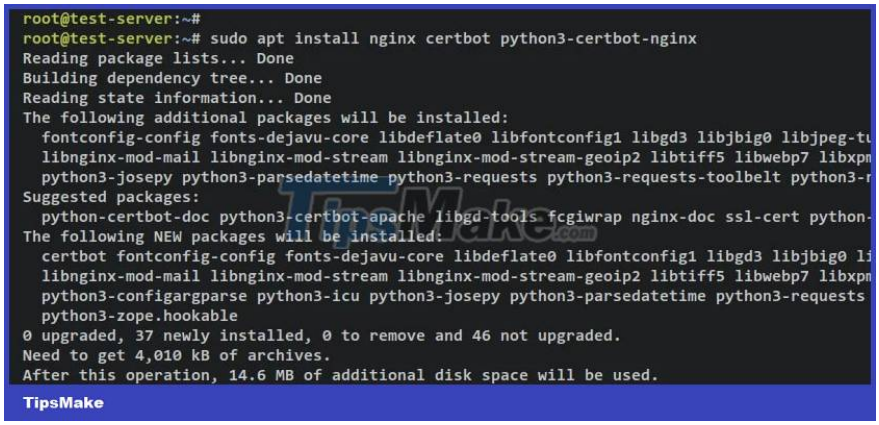
```
docker compose logs guacamole docker compose logs guacd docker compose logs postg
```

Install Nginx web server

For this tutorial, you will run Apache Guacamole with Nginx reverse proxy. In this section, you will install the Nginx web server and the Certbot tool to create SSL/TLS certificates. You will then verify the Nginx service to ensure that it is enabled and running.

Run the following apt command to install the Nginx, Certbot, and Certbot Nginx plugins. Enter y when prompted for confirmation and press ENTER to continue.

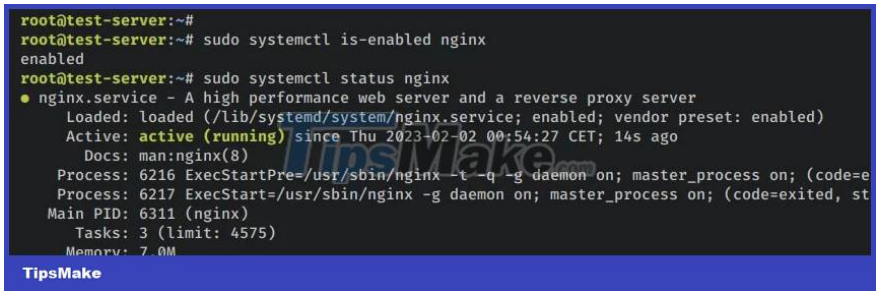
```
sudo apt install nginx certbot python3-certbot-nginx
```



After Nginx and Certbot are installed, run the following command to verify Nginx service status. This will ensure that the Nginx service is enabled and running on your system.

```
sudo systemctl is-enabled nginx sudo systemctl status nginx
```

The 'enabled' output confirms that the Nginx service is enabled and will run automatically on system boot. The 'active (running)' output confirms that the Nginx service is running.



Set up a UFW firewall

After installing Nginx, you will next set up the UFW firewall that is installed by default on your Ubuntu system. In this section, you will add the OpenSSH service to open port 22 and add the 'Nginx Full' service to open both

HTTP and HTTPS ports on ufw. Then you will start and enable ufw. Finally, you will verify the status of the ufw firewall.

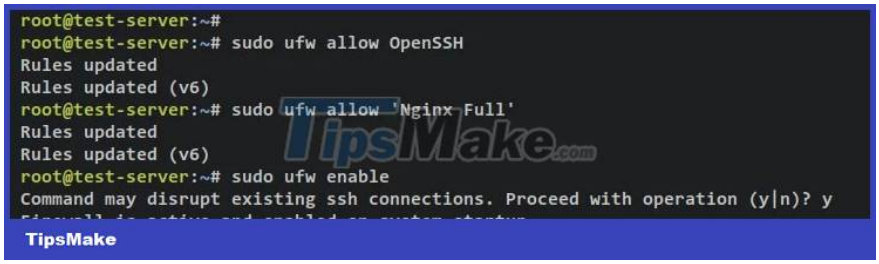
Enter the following command to add OpenSSH and 'Nginx Full' services to the ufw firewall. 'Rules updated' output confirms that new rules have been added to ufw.

```
sudo ufw allow OpenSSH sudo ufw allow 'Nginx Full'
```

Next, enter the following command to start and enable the ufw firewall. Enter y when prompted and press ENTER to continue.

```
sudo ufw enable
```

Now, you will get output like ' Firewall is active and enabled on system startup ', which means ufw firewall is running and enabled and will start automatically on system startup.



Verify the status of the ufw firewall by entering the following command.

```
sudo ufw status
```

You will get the status of ufw firewall as ' active ' and enabled services 'OpenSSH' will open SSH port 22 and service 'Nginx Full' will open both HTTP and HTTPS ports.

Set up Nginx as a reverse proxy

To secure your Apache Guacamole deployment, you will use Nginx as a reverse proxy and enable HTTPS secure connections on it.

In this section, you will create a new Nginx server block configuration that will be used as a reverse proxy for Apache Guacamole, then generate an SSL/TLS certificate through Certbot and Letsencrypt to secure the Apache Guacamole deployment.

Create a new Nginx server block configuration '/etc/nginx/sites-available/guacamole.conf' using the following nano editor command.

```
sudo nano /etc/nginx/sites-available/guacamole.conf
```

Add the following lines to the file and make sure to change the domain name in the configuration below. With this, you will set up Nginx as a reverse proxy for the Apache Guacamole container that has exposed port 8080 on the Docker host.

```
server { listen 80; server_name guacamole.hwdomain.io; root /var/www/html; index
```

Save and close the file when finished.

Next, run the command below to enable the Nginx server block file '/etc/nginx/sites-available/guacamole.conf'. Then, verify the Nginx configuration to ensure that you have the right settings.

```
sudo ln -s /etc/nginx/sites-available/guacamole.conf /etc/nginx/sites-enabled/ su
```

If successful, you will get a result like ' test successful – syntax ok '.

Now, run the following systemctl command utility to restart the Nginx service and apply the changes.

```
sudo systemctl restart nginx
```

Output:

At this point, Apache Guacamole is running with Nginx as a reverse proxy with your domain name – this example uses the domain name 'guacamole.hwdomain.io'. Now to secure your Apache Guacamole deployment, you will need to generate an SSL/TLS certificate through Certbot and Letsencrypt.

Enter the following certbot command to generate a new SSL certificate for your Nginx virtual server. Make sure to change the domain and email address details in the following command.

```
sudo certbot --nginx --agree-tos --redirect --hsts --staple-ocsp --email alice@hw
```

Once generated, your SSL certificate will be available in the '/etc/letsencrypt/live/guacamole.hwdomain.io/' directory. Additionally, your Nginx server block will be automatically changed when SSL is enabled and automatically redirected from HTTP to HTTPS.

Visit Apache Guacamole

Open your web browser and visit the domain of the Apache Guacamole installation (for example, https://guacamole.hwdomain.io/).

Log in with the default user and password 'guacadmin'.

On success, you will get the Apache Guacamole user dashboard.

At this point, you are done installing Apache Guacamole through Docker and Docker Compose. Additionally, you configured Nginx as a reverse proxy for Apache Guacamole and secured it via an SSL/TLS certificate from Letsencrypt.

You finished reading the article "**How to Install Apache Guacamole via Docker on Ubuntu 22.04**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.
