How To Run Graylog Server in Docker Containers

All applications generate information when running, this information is stored as logs. As a system administrator, you need to monitor these logs to ensure the proper functioning of the system and therefore prevent risks and errors. These logs are normally scattered over servers and management becomes harder as the data volume increases.

Graylog is a free and open-source log management tool that can be used to capture, centralize and view real-time logs from several devices across a network. It can be used to analyze both structured and unstructured logs. The Graylog setup consists of MongoDB, Elasticsearch, and the Graylog server. The server receives data from the clients installed on several servers and displays it on the web interface.

1. Install Docker and Docker-Compose on Linux

Of course, you need the docker engine to run the docker containers. To install the docker engine, use the dedicated guide below:

sudo apt install docker.io

```
sudo usermod -aG docker $USER
```

```
newgrp docker
```

```
docker version
```

With docker installed, proceed and install docker-compose using the guide below:

You need curl and wget installed on your system for this operation. And definitely, access to the Terminal as a user with sudo privileges.

```
sudo apt install -y curl wget
```

Once curl has been installed, download the latest Compose on your Linux machine.

```
curl -s https://api.github.com/repos/docker/compose/releases/latest | grep
browser_download_url  | grep docker-compose-linux-x86_64 | cut -d '"' -f 4 |
wget -qi -
```

Make the binary file executable.

```
chmod +x docker-compose-linux-x86_64
```

Move the file to your PATH.

```
sudo mv docker-compose-linux-x86_64 /usr/local/bin/docker-compose
```

Confirm version.

```
docker-compose version
```

```
sudo systemctl start docker && sudo systemctl enable docker
```

2. Provision the Graylog Container

The Graylog container will consist of the Graylog server, Elasticsearch, and MongoDB. To be able to achieve this, we will capture the information and settings in a ***YAML*** file.

Create the YAML file as below:

```
nano docker-compose.yml

In the file, add the below lines:

version: '2'
services:
  # MongoDB: https://hub.docker.com/_/mongo/
  mongodb:
    image: mongo:4.2
    networks:
      - graylog
  #DB in share for persistence
    volumes:
      - /mongo_data:/data/db
   # Elasticsearch:
https://www.elastic.co/guide/en/elasticsearch/reference/7.10/docker.html
  elasticsearch:
    image: docker.elastic.co/elasticsearch/elasticsearch-oss:7.10.2
    #data folder in share for persistence
    volumes:
      - /es_data:/usr/share/elasticsearch/data
    environment:
      - http.host=0.0.0.0
      - transport.host=localhost
      - network.host=0.0.0.0
      - "ES_JAVA_OPTS=-Xms512m -Xmx512m"
```

```
    ulimits:
      memlock:
        soft: -1
        hard: -1
    mem_limit: 1g
    networks:
      - graylog
  # Graylog: https://hub.docker.com/r/graylog/graylog/
  graylog:
    image: graylog/graylog:4.2
    #journal and config directories in local NFS share for persistence
    volumes:
      - /graylog_journal:/usr/share/graylog/data/journal
    environment:
      # CHANGE ME (must be at least 16 characters)!
      - GRAYLOG_PASSWORD_SECRET=password@1234567890
      # Password: admin
      -
GRAYLOG_ROOT_PASSWORD_SHA2=e1b24204830484d635d744e849441b793a6f7e1032ea1eef4074
7d95d30da592
      - GRAYLOG_HTTP_EXTERNAL_URI=http://192.168.1.2:9000/
    entrypoint: /usr/bin/tini -- wait-for-it elasticsearch:9200 --
/docker-entrypoint.sh
    networks:
      - graylog
    links:
      - mongodb:mongo
      - elasticsearch
    restart: always
    depends_on:
      - mongodb
      - elasticsearch
    ports:
      # Graylog web interface and REST API
      - 9000:9000
      # Syslog TCP
      - 1514:1514
      # Syslog UDP
      - 1514:1514/udp
      # GELF TCP
      - 12201:12201
      # GELF UDP
      - 12201:12201/udp
# Volumes for persisting data, see
https://docs.docker.com/engine/admin/volumes/volumes/
volumes:
  mongo_data:
    driver: local
  es_data:
    driver: local
  graylog_journal:
    driver: local
networks:
    graylog:
      driver: bridge
```

In the file, replace:

- **GRAYLOG_PASSWORD_SECRET** with your own password which must be at least 16 characters
- **GRAYLOG_ROOT_PASSWORD_SHA2** with a SHA2 password obtained using the command:

```
echo -n "Enter Password: " && head -1 </dev/stdin | tr -d '\n' | sha256sum | cut -d" " -f1
```

GRAYLOG_HTTP_EXTERNAL_URI with the IP address of your server.

3. Create Persistent volumes

In order to persist the data, you will use external volumes to store the data. In this guide, we have already mapped the volumes in the YAML file. Create the 3 volumes for MongoDB, Elasticsearch, and Graylog as below:

```
sudo mkdir /mongo_data
sudo mkdir /es_data
sudo mkdir /graylog_journal
```

Set the right permissions:

```
sudo chmod 777 -R /mongo_data
sudo chmod 777 -R /es_data
sudo chmod 777 -R /graylog_journal
```

4. Run the Graylog Server in Docker Containers

With the container provisioned, we can now spin it easily using the command:

```
docker-compose up -d

Once all the images have been pulled and containers started, check the status as below:

docker ps

If you have a firewall enabled, allow the Graylog service port through it.

sudo ufw allow 9000/tcp
```

5. Access the Graylog Web UI

Now open the Graylog web interface using the URL http://IP_address:9000.

Log in using the username **admin** and SHA2 password(StrongPassw0rd) set in the YAML.

On the dashboard, let's create the first **input** to get logs by navigating to the systems tab and selecting **input.**

Now search for ***Raw/Plaintext TCP*** and click **launch new input**

Once launched, a pop-up window will appear as below. You only need to change the name for the input, port(1514), and select the node, or "Global" for the location for the input. Leave the other details as they are.

Save the file and try sending a plain text message to the Graylog Raw/Plaintext TCP input on port **1514.**

```
echo 'First log message' | nc localhost 1514

##OR from another server##

echo 'First log message' | nc 192.168.205.4 1514
```

On the running Raw/Plaintext Input, **show received messages**

The received message should be displayed

Conclusion

That is it!

We have triumphantly walked through how to run the Graylog Server in Docker Containers. Now you can monitor and access logs on several servers with ease. I hope this was significant to you.