Network Applications

Reference information for my network related applications

- NGINX Proxy Manager
- Uptime Kuma
- GoAccess

NGINX Proxy Manager

What is Nginx Proxy Manager?

Nginx Proxy Manager is a Docker application that lets you quickly and easily expose your selfhosted services to the outside world. NPM includes Letsencrypt SSL certificate management, which permits you to obtain free SSL certificates for secure hosting of your sites.

Installation

NGINX Proxy Manager (NPM) is installed as a Docker container.

You must have Docker and Docker Compose installed to use NPM. I am currently using Docker CE (community edition).

You also have a choice of databases to use with NPM. The default database installed is SQLite. I chose to utilize MariaDB instead of the default as it is open-source and MySQL compatible but with a richer feature set and better performance than either MySQL or SQLite.

> Please note, that DB_MYSQL_* environment variables will take precedent over DB_SQLITE_* variables. So if you keep the MySQL variables, you will not be able to use SQLite. #

This installation guide is for NPM with MariaDB (MySQL).

Using MariaDB Database with NPM

If you opt for the MariaDB configuration you will have to provide the database server yourself. The current minimum supported version is:

- MariaDB v10.2.7+

It's easy to use another docker container for your database also and link it as part of the docker stack, so that's what the following examples are going to use.

Here is my `docker-compose.yml` using a MariaDB container. You can use it as example :

```
version: '3'
services:
  app:
    image: 'jc21/nginx-proxy-manager:latest'
    ports:
      - '80:80'
      - '81:81'
      - '443:443'
    environment:
      DB_MYSQL_HOST: "db"
      DB_MYSQL_PORT: 3306
      DB_MYSQL_USER: "your MySQL username"
      DB_MYSQL_PASSWORD: "your MySQL password"
      DB_MYSQL_NAME: "nginx"
      TZ: America/New_York
    volumes:
      - /localpathtoyourNPMdata:/data
      - /localpathtoyourNPMletsencryptcertificatedata:/etc/letsencrypt
  db:
    image: 'mariadb'
    environment:
      MYSQL_ROOT_PASSWORD: 'your MySQL root password'
      MYSQL_DATABASE: 'nginx'
      MYSQL_USER: 'your MySQL username'
      MYSQL_PASSWORD: 'your MySQL password'
      TZ: America/New_York
    volumes:
      - /localpathtoyourNPMdatabase:/var/lib/mysql
```

> Make sure you change DB_MYSQL_USER, DB_MYSQL_PASSWORD and MYSQL_ROOT_PASSWORD to whatever username and passwords you intend to use.

>

> Make sure you change the local path of your volumes to the path you intend to use to store NGINX Proxy Manager data, certificates and the database.

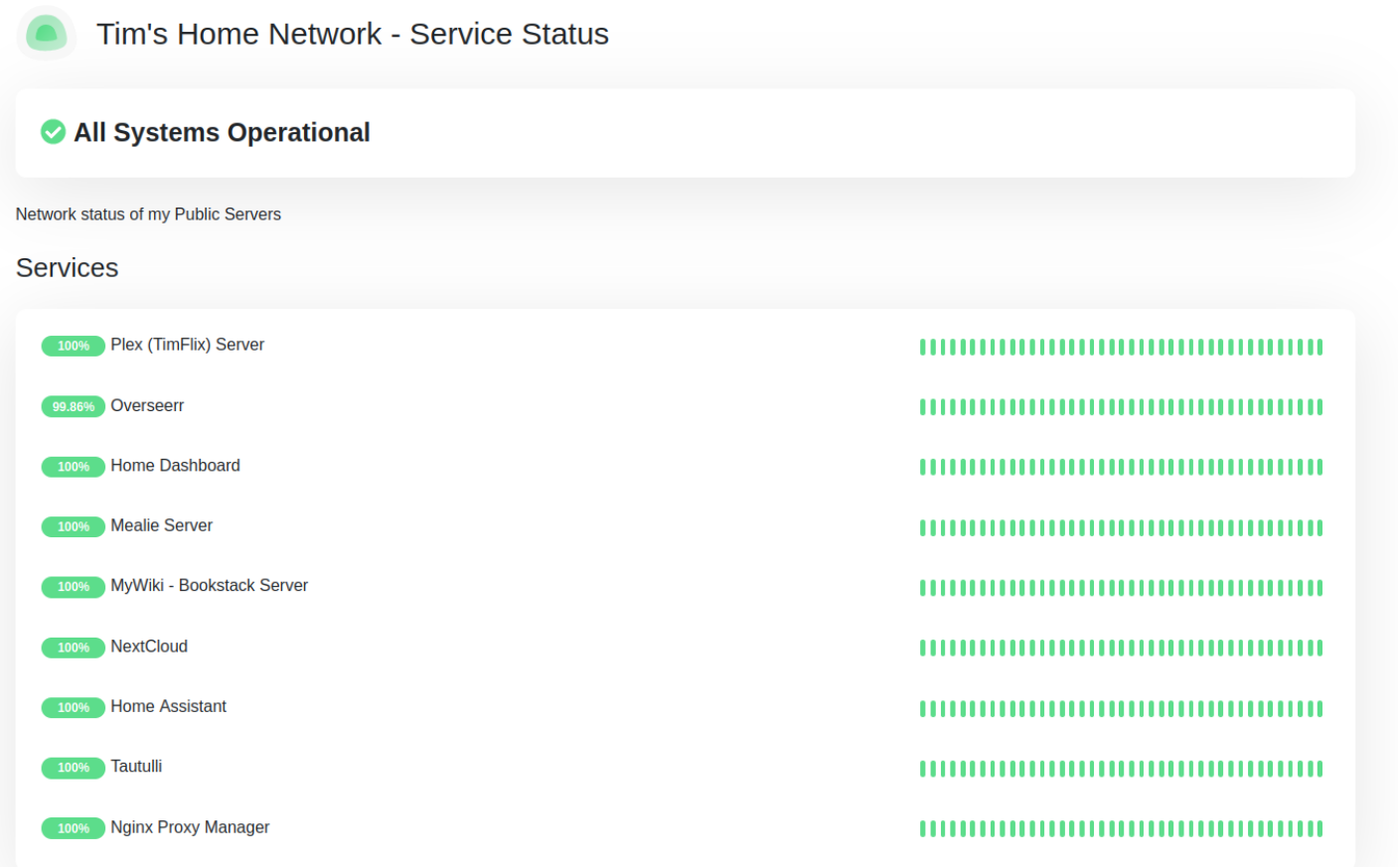
> Also, make sure you change the timezone (TZ) parameter to reflect your timezone as it affects the certificate timestamps you get from Let's Encrypt. You can find your timezone from here: Wikipedia TZ Database

Uptime Kuma

[Uptime Kuma](#) is a self-hosted, open source, fancy uptime monitoring and alerting system. It can monitor HTTP, HTTP with keyword, TCP, Ping, and DNS systems.

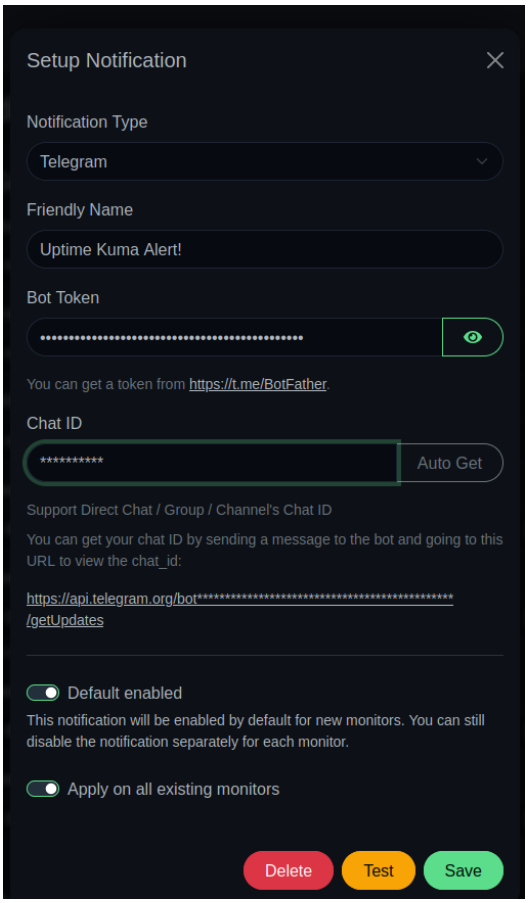
Uptime Kuma is an easy way to know if your systems are up and running. You can even add your favorite Internet sites (i.e., Facebook, Amazon, Twitter, etc.) if you want to be sure they are working. I use Uptime Kuma primarily for my home LAN components and any websites/applications I host from home.

Uptime Kuma even permits me to provide status pages for my users so they can know at a glance if any of my services are down or under maintenance.



In addition, you can setup any number of ways to be notified if any monitored service becomes unavailable. Uptime Kuma supports notifications via email, SMS and more. I currently use Telegram for notifications from Kuma to my phone.

> As Kuma also supports **Apprise** (which supports 50+ notification methods by itself), I will likely move to that notification platform in the future.



Installation

There are both stand-alone and container installation methods for Uptime Kuma. I chose the container method as I am heavily invested in Docker and Docker-Compose on my systems.

To install via Docker, use the following code for the default values.

```
docker run -d --restart=always -p 3001:3001 -v uptime-kuma:/app/data --name uptime-kuma louislam/uptime-kuma:1
```

Otherwise, you can specify your port and data storage (volume) using this docker template.

```
docker run -d --restart=always -p <YOUR_PORT>:3001 -v <YOUR_DIR OR VOLUME>:/app/data --name uptime-kuma louislam/uptime-kuma:1
```

If you use Docker Compose, use the following instructions:

> Shell instructions.

```
mkdir uptime-kuma
cd uptime-kuma
```

```
touch docker-compose.yml
nano docker-compose.yml # copy the contents from the docker-compose.yml example below
mkdir data
ls
docker-compose up -d --force-recreate
```

> Docker compose file contents. This goes in the docker-compose.yml file you created above.

```
---
version: "3.1"

services:
  uptime-kuma:
    image: louislam/uptime-kuma:1
    container_name: uptime-kuma
    volumes:
      - <Uptime Kuma data volume>:/app/data
    ports:
      - <Uptime Kuma port>:3001
    restart: unless-stopped
    security_opt:
      - no-new-privileges:true
```

> Make sure you replace <Uptime Kuma data volume> with the path on your local machine that you want to save your Kuma configuration and data files. For example, /var/lib/docker/volumes/uptime-kuma

> Also make sure you change <Uptime Kuma port> to whatever port you want to use. The default port is 3001. I use the default port because it doesn't conflict with any other software on the system where I have Kuma installed. Your system may be different.

So, as an example, if you wanted to use the path and port as listed above, your docker-compose.yml would look like this:

```
---
version: "3.1"

services:
  uptime-kuma:
```

```
image: louislam/uptime-kuma:1
container_name: uptime-kuma
volumes:
  - /var/lib/docker/volumes/uptime-kuma:/app/data
ports:
  - 3001:3001
restart: unless-stopped
security_opt:
  - no-new-privileges:true
```

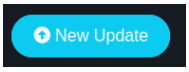
Accessing Uptime Kuma

Once you have started your Uptime Kuma container, you should now be able to access from a browser by going to

http://<your server IP>:<your port>

where **<your server ip>** is the IP address of the server where you installed Uptime Kuma and **<your port>** is the port you chose to use in the docker-compose.yml file. So if your server's IP address is 192.168.1.10 and you chose port 3001, the URL for your browser should look like: http://192.168.1.10:3001

Updating Uptime Kuma

When a new version of Uptime Kuma becomes available, you will see 

appear at the top right of your Uptime Kuma dashboard. Clicking this will take you to the Uptime Kuma Github page where you can see what changes have been made to the new version.

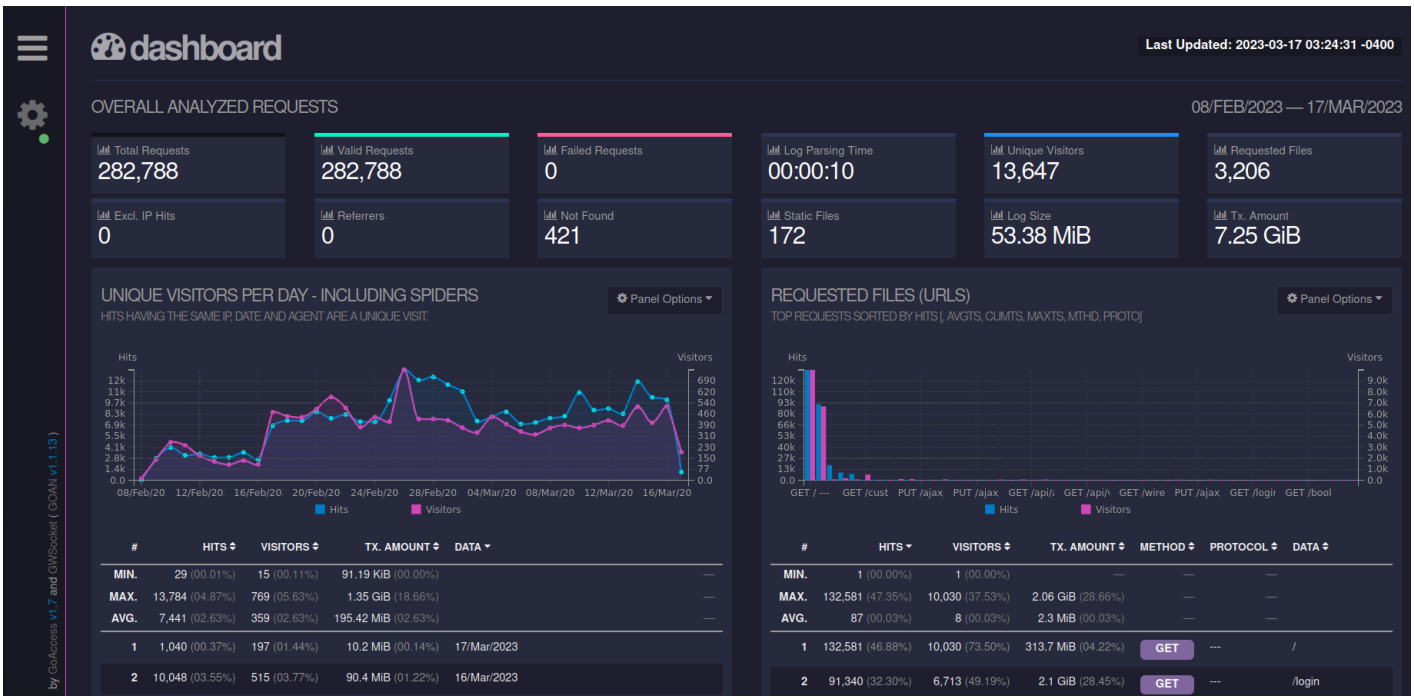
If you installed Uptime Kuma using Docker or Docker-Compose (not stand-alone) you may not yet be able to update your instance until the image is built. This is explicitly stated on the Kuma Update page...

> For every new release, it takes some time to build the docker image, please be patient if it is not available yet

I currently use **Watchtower** to monitor and automatically update all my Docker images. If you do not have some method installed to automate updating your Docker images you may use the excellent instructions provided by the Uptime Kuma author louislam to manually update your Uptime Kuma container.

GoAccess

GoAccess Web Server Statistics



GoAccess is an open source **real-time web log analyzer** and interactive viewer that runs in a **terminal** in *nix systems or through your **browser**.

It provides **fast** and valuable HTTP statistics for system administrators that require a visual server report on the fly.

I use GoAccess to monitor my self-hosted websites traversing my reverse proxy and Cloudflare.

Installation

GoAccess installation methods can be found by going to the **official GoAccess website**

I use a Docker image specifically designed for GoAccess to pull logs from an instance of **Nginx Proxy Manager**. If you have that setup, then this is my recommended Docker Compose file setup.

```
version: '3.3'
services:
    goaccess:
        image: 'xavierh/goaccess-for-nginxproxymanager:latest'
```

```
    container_name: goaccess
    restart: always
    ports:
        - '7880:7880'
    environment:
        - TZ=America/New_York
        - SKIP_ARCHIVED_LOGS=False #optional
        - DEBUG=False #optional
        - BASIC_AUTH=False #optional
        - BASIC_AUTH_USERNAME=user #optional
        - BASIC_AUTH_PASSWORD=pass #optional
        - EXCLUDE_IPS=127.0.0.1 #optional - comma delimited
        - LOG_TYPE=NPM #optional - more information below
    volumes:
    - /path-to-your-nginxproxymanager/logs:/opt/log
    - /path/to/host/custom:/opt/custom #optional, required if using log_type = CUSTOM
```

> Be sure to replace "/path-to-your-nginxproxymanager/logs" with your actual path to your Nginx Proxy Manager logs. This docker-compose.yml also assumes port 7880 is available on your server. If not, change to an available port of your choosing.

Explanations of the "optional" portions of the docker compose file can be found by going to **the Docker image author's Github repository**.