Tienwoningenweg 46
7312 DN Apeldoorn
Tel: +31-6-22660412
Fax: +31-55-5431951

KvK Apeldoorn: 08073144
BTW NL100626956B02

sander@steffann.nl

steffann

MANRS Lab

Server Installation Guide

Version 2 – 30 March 2019

Table of Contents

Server requirements

The most important requirement of a lab server is the amount of memory. Virtualisation, used for emulating routers, servers and other devices, needs a lot of memory.

CPU speed is of secondary concern.

Example configuration

The server running gns3.steffann.nl consists of:

- 32 GB of RAM
- Dual Intel Xeon L5640 processors (6 cores each at 2.27GHz)
- 1TB of disk space

When running 10 Cisco-based MANRS labs the memory usage was around 10GB and the CPU load was between 15% and 50%, depending on the progress in the exercise. As can be expected the better MANRS is implemented the lower the CPU load. Disk usage was below 20 GB.

Because the MANRS lab is build on GNS3 the scalability of the lab is closely tied to the scalability of GNS3. GNS3 allows clustering servers, so the lab can be scaled by adding more servers when necessary.
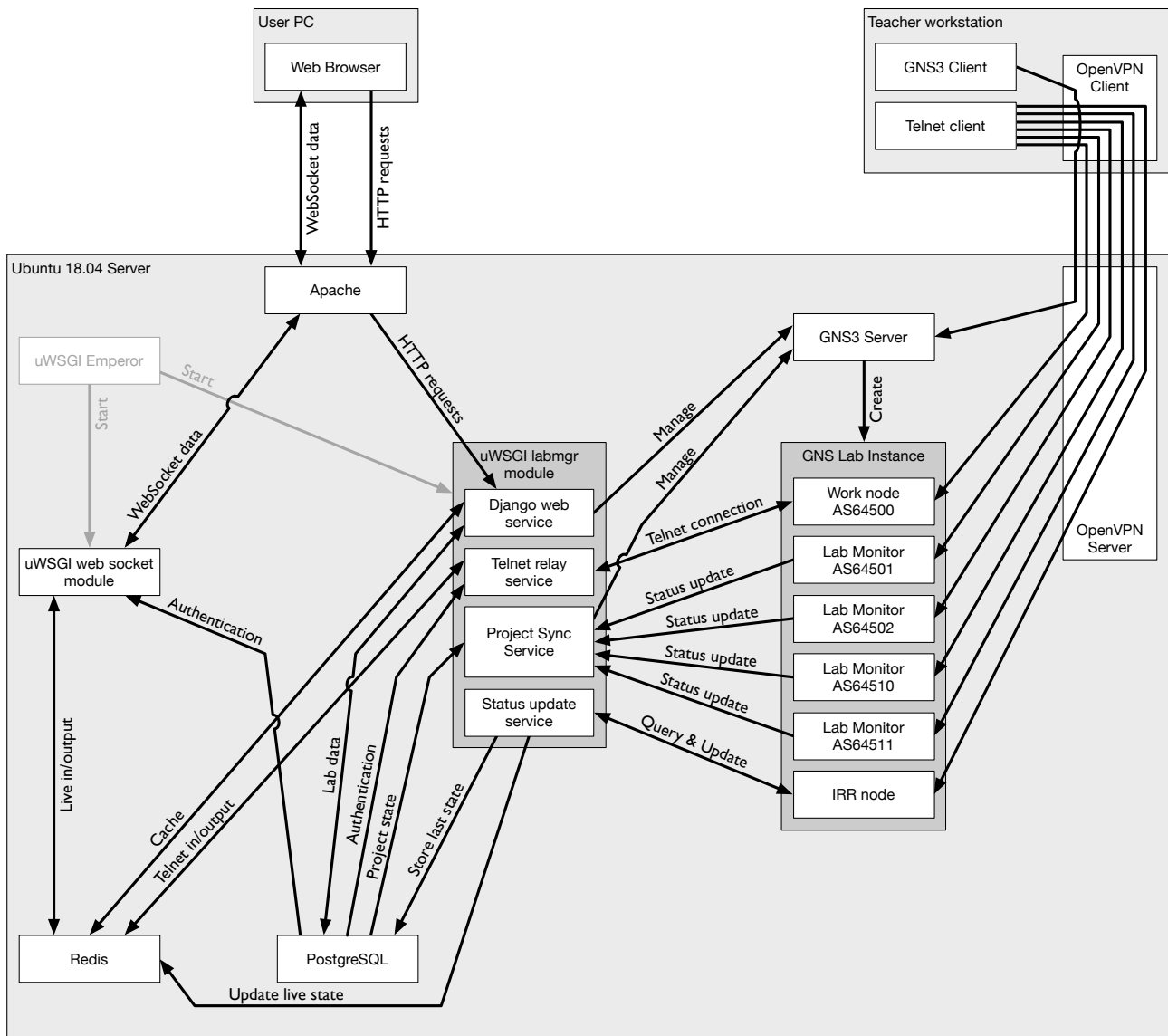
Operating system requirements

The system is tested to run on Ubuntu 18.10, but should be able to run on any modern Linux based system which has the correct uwsgi packages available.

The instructions below assume Ubuntu 18.10.

Lab server architecture

The following diagram shows all the components of the lab management software and their relations to each other:

- **GNS3** is the network emulation software.
- **Apache** is the front end webserver.
- **uWSGI** is the web server framework that manages the server code.
- **Django** is used as the web application framework.
- **PostgreSQL** is the database that stores all data.
- **Redis** is used as a cache and for live communication between the other components.
- **OpenVPN** (or equivalent) is used for secure access to the back end systems.

Server software requirements

Installing GNS3

The default GNS3 installation script unfortunately does not work because we need a non-LTS Ubuntu release for the required `uwsgi` packages and the installation only works on LTS releases. We have found that removing the check in the installation script works (for now). Download the installation script:

```
curl https://raw.githubusercontent.com/GNS3/gns3-server/master/scripts/
    remote-install.sh > gns3-remote-install.sh
```

Then remove the following lines from that script:

```
lsb_release -d | grep "LTS" > /dev/null
if [ $? != 0 ]
then
  echo "This script can only be run on a Linux Ubuntu LTS release"
  exit 1
fi
```

And then run the script:

```
sudo bash gns3-remote-install.sh --with-openvpn
```

We do not need the IOU extension or the 32-bit packages, so those options can be skipped. The OpenVPN option is recommended so that trainers can securely access the lab backend to develop new lab templates or to debug running labs.

Make sure the gns3 service is enabled and started:

```
sudo systemctl enable gns3.service
sudo systemctl start gns3.service
```

Installing Python 3

The lab management software is written in Python 3 and uses several libraries to implement common features and communicate with other components. Some of these libraries are built when installing them, so for the installation we need Python as well as the development packages needed for those libraries. Make sure the following packages are installed:

- build-essential
- python3
- python3-dev
- python3-pexpect
- python3-pip
- python3-ptyprocess
- python3-requests
- python3-urllib3
- python3-virtualenv
- python3-yaml

Other dependencies will be installed together with the corresponding components where necessary.

Installing the uWSGI framework

The lab management software uses the uWSGI framework for scaling the web services up/down and handling all background processes, connections with the GNS3 server etc. Make sure the following packages are installed on the server:

- uwsgi-core

- uwsgi-emperor

- uwsgi-plugin-gevent-python3

- uwsgi-plugin-greenlet-python3

- uwsgi-plugin-python3

Make sure the uWSGI Emperor service is configured as follows in /etc/uwsgi-emperor/emperor.ini:

```
[uwsgi]
# try to autoload appropriate plugin if "unknown" option has been specified
autoload = true

# enable master process manager
master = true

# spawn 2 uWSGI emperor worker processes
workers = 2

# automatically kill workers on master's death
no-orphans = true

# place timestamps into log
log-date = true

# user identifier of uWSGI processes
uid = www-data

# group identifier of uWSGI processes
gid = www-data

# vassals directory
emperor = /etc/uwsgi-emperor/vassals

emperor-tyrant = true
cap = setgid,setuid
```

Make sure the uwsgi-emperor service is enabled and started:

```
sudo systemctl enable uwsgi-emperor.service
sudo systemctl start uwsgi-emperor.service
```

Installing the database

The lab management system uses a PostgreSQL database to store all the lab data. Make sure the following packages are installed:

- postgresql

- libpq5

- libpq-dev

Make sure the database service is enabled and started:

```
sudo systemctl enable postgresql.service
sudo systemctl start postgresql.service
```

After installing the database server create a user and empty database:

```
sudo createuser --pwprompt labmgr
sudo createdb --owner labmgr labmgr
```

Remember the password you configured for the `labmgr` user, you will need it later.

Installing Redis

The lab management system uses Redis for cacheing and as a communications channel between components. Make sure the following packages are installed:

- redis-server

- redis-tools

Make sure the redis service is enabled and started:

```
sudo systemctl enable redis-server.service
sudo systemctl start redis-server.service
```

Installing the lab management system

When the GNS3 server is installed and running the lab management system can be installed. Make sure the following packages are installed:

- git

- npm

Downloading the code

The system must run as a regular user without any special privileges. In this documentation I will assume the user is called `exampleuser` but any username is fine. The easiest way to install this is by cloning its git repository:

```
cd ~exampleuser
git clone https://github.com/MANRS-Lab/labmgr.git
```

This will create a labmgr subdirectory with all the required code in it.

Installing dependencies

There are a number of Python dependencies. These can be automatically installed with:

```
cd ~exampleuser/labmgr
pip3 install -r requirements.txt
```

This will install and build the necessary Python modules to run the lab management software.

Then run the following command to install the required node modules and to compile and optimise the Javascript code:

```
npm install
npm run build
```

That will create the files necessary to serve the dashboard to the students. The last step is to collect all the static files so they can be served efficiently:

```
./manage.py collectstatic --no-input
```

Creating configuration file

Now a configuration file needs to be created with the settings specific to this installation. The location of the file is the `~exampleuser/labmgr/labmgr` directory and the file name is `local_settings.py`. An example of the contents is given below:

```python
from django.utils.translation import gettext_lazy as _

SITE_HEADER = _('MANRS Lab Manager')
SITE_TITLE = _('Lab Manager')
INDEX_TITLE = _('Lab administration')

ADMINS = [('Kevin Chege', 'chege@isoc.org')]

DEFAULT_FROM_EMAIL = 'chege@isoc.org'
SERVER_EMAIL = DEFAULT_FROM_EMAIL

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'put a long string of random characters here'

# Allowed host names for cross-site protections
ALLOWED_HOSTS = ['lab.manrs.org']

# Database
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'HOST': 'localhost',
        'NAME': 'labmgr',
        'USER': 'labmgr',
        'PASSWORD': 'your password',
    }
}

# GNS3 settings
GNS3 = {
    'HOST': '172.16.253.1',
    'PORT': 3080,
}

# Debugging
DEBUG = False
```

Make sure you randomise the `SECRET_KEY`, that `ALLOWED_HOSTS` contains the hostname of the server and that the `GNS3 HOST` address matches the address in `/etc/gns3/gns3_server.conf`.

When things don't work you can turn on `DEBUG` for easier problem solving, but make sure it is not enabled in production!

Activating uWSGI modules

There are two uWSGI modules that need to run. The first module contains the web service and the background processes (called "mules" in uWSGI) that handle synchronising with GNS3 and receiving status updates from the lab nodes. The second module is optimised for handling WebSockets, which are used for communication between the user's browser and the rest of the system.

The first module file is /etc/uwsgi-emperor/vassals/labmgr.ini and contains:

```
[uwsgi]
include = /home/exampleuser/labmgr/uwsgi-django.ini
chdir = /home/exampleuser/labmgr
```

The second module file is /etc/uwsgi-emperor/vassals/labmgr-ws.ini and contains:

```
[uwsgi]
include = /home/exampleuser/labmgr/uwsgi-websocket.ini
chdir = /home/exampleuser/labmgr
```

Note: both files must be owned by the user that owns the labmgr directory:

```
sudo chown exampleuser:exampleuser /etc/uwsgi-emperor/vassals/labmgr.ini
sudo chown exampleuser:exampleuser /etc/uwsgi-emperor/vassals/labmgr-ws.ini
```

This will activate the uWSGI services.

Create a superuser

To log in to the admin interface of the lab management system a superuser is required. To create one perform the following:

```
cd ~exampleuser/labmgr
./manage.py createsuperuser
```

The command will ask you for all the necessary details.

If there is an error while executing this command it might be that the uWSGI Emperor service hasn't started the Django service yet. The database tables are created during startup, and if this hasn't happened yet then the `createsuperuser` command will miss those tables. You can restart the uWSGI Emperor with:

```
sudo systemctl restart uwsgi-emperor.service
```

If the error persists please check your system logs.

Installing Apache

Apache is the web server that users will access. It will forward incoming requests to the corresponding back-end components. We will use a newer version of Apache than is currently provided by Ubuntu. This newer version provides easier LetsEncrypt integration and HTTP/2 support.

Activating extra Apache repository

To activate the repository with newer Apache packages use:

```
sudo add-apt-repository ppa:ondrej/apache2
sudo apt-get update
```

Installing packages

Make sure the following packages are installed:

- apache2

- apache2-utils

- libapache2-mod-proxy-uwsgi

TLS Configuration

For security and performance it is recommended to enable TLS as well as HTTP/2. Create the file `/etc/apache2/conf-available/tls.conf` with this content:

```
# Agree to the LetsEncrypt agreement for mod_md
MDCertificateAgreement https://letsencrypt.org/documents/LE-SA-v1.2-
   November-15-2017.pdf

# We staple, so make it mandatory
MDMustStaple on

# We require HTTPS
MDRequireHttps permanent

# Based on:
# https://mozilla.github.io/server-side-tls/ssl-config-generator/?
   server=apache-2.4.28&openssl=1.0.2g&hsts=yes&profile=modern
# modern configuration, tweak to your needs
SSLProtocol                        all -SSLv3 -TLSv1 -TLSv1.1
SSLCipherSuite                     ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-
   RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-
   POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-
   ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-
   SHA256:ECDHE-RSA-AES128-SHA256
SSLHonorCipherOrder                on
SSLCompression                     off
SSLSessionTickets                  off

# OCSP Stapling, only in httpd 2.3.3 and later
SSLUseStapling                     on
SSLStaplingResponderTimeout        5
SSLStaplingReturnResponderErrors   off
SSLStaplingCache                   shmcb:/var/run/ocsp(128000)
```

Enable configuration and modules

These commands will enable all required and recommended configurations and modules:

```
sudo a2enconf tls
sudo a2enmod mpm_event http2 md ssl proxy_uwsgi proxy_wstunnel
```

Create web site configuration

The web site configuration looks as follows. Please adjust the server name, admin contact details and directory names as appropriate. In this example we are putting this in `/etc/apache2/sites-available/labmgr.conf`:

```
MDomain lab.manrs.org

<VirtualHost *:80>
    ServerName lab.manrs.org
    ServerAdmin admin@manrs.org
</VirtualHost>

<VirtualHost *:443>
    ServerName lab.manrs.org
    ServerAdmin admin@manrs.org

    Protocols h2 http:/1.1
    SSLEngine on

    ProxyPass /static/ !
    ProxyPass /ws/ ws://localhost:8000/ws/
    ProxyPass / uwsgi://localhost:8001/
    ProxyPassReverse / uwsgi://localhost:8001/

    Alias /static/ /home/exampleuser/labmgr/static/
</VirtualHost>

<Directory /home/exampleuser/labmgr/static/>
    Require all granted
    AllowOverride None
</Directory>
```

Activate web site configuration

The default installation may have activated some default web site settings. To disable these and to enable the lab manager use the following commands:

```
sudo a2dissite 000-default default-ssl
sudo a2ensite labmgr
```

Restart the web server

After updating the configuration the web server must be restarted. Use the following command:

```
sudo systemctl restart apache2.service
```

The web server will restart and will start to get a LetsEncrypt certificate. Until this is completed the website will not be accessible. This can take up to a few minutes. Then restart the web server again with the command given above, and the web server will start up with TLS enabled and the web site accessible.

Client software requirements

Browser

The lab is accessed through a modern web browser. Current versions of Safari, Chrome and Firefox have been tested.

Installing GNS3 client software

These instructions are only relevant for server operators, teachers developing new lab templates or when debugging running labs. Usage of existing lab templates does not require anything beyond a web browser.

To access the server securely an OpenVPN client is used. Instructions on how to use the VPN can be found on https://docs.gns3.com/1c2Iyiczy6efnv-TS_4Hc7p11gn03-ytz9ukgwFfckDk/index.html. Popular OpenVPN client applications are Viscosity and Tunnelblick.

Teachers can then use the GNS3 client on their PC to get direct access to the lab configurations over the VPN. The client software can be downloaded from https://github.com/GNS3/gns3-gui/releases and instructions on how to configure it are at https://docs.gns3.com/1K_OVfincey0cUw6CP4dWVgs_pBXMdIJ6gdFGjNy8EZQ/index.html. Make sure to select the "Run everything on a remote server" option.

When downloading the GNS3 client software make sure you download the exact same version as is used on the server. If the versions do not match the client will refuse to connect to the server.

Connecting the GNS3 Client

After creating a VPN connection to access the GNS3 server back-end, start the GNS3 client application. When starting it for the first time it will ask you which server to use:
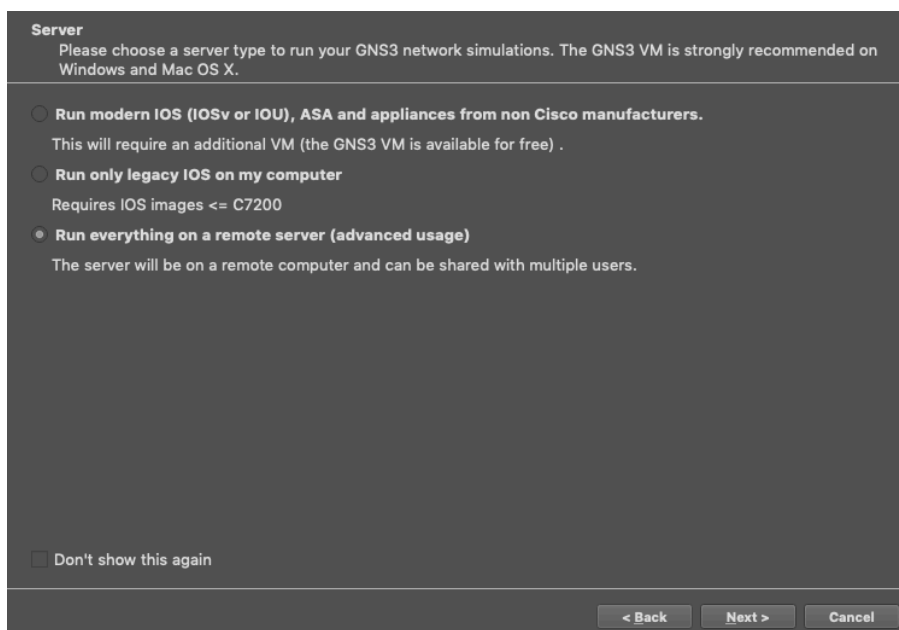

Choose "Run everything on a remote server" here. The next screen will ask you for the host and port of the server. Enter the hostname or IP address of the GNS3 server. The port number is usually `3080`.

Basic GNS3 appliances

A few GNS3 appliance types have been developed to work together with the lab management software. It is strongly recommended to install these appliances first because they will be used in (almost) every exercise.

First go to https://github.com/MANRS-Lab/GNS3-Appliances and download all `.gns3a` files. Then use the GNS3 client to install these appliances. From the `File` menu choose `Import Appliance` and open the `.gns3a` file you just downloaded. Each appliance will use one or more files that you need to download:
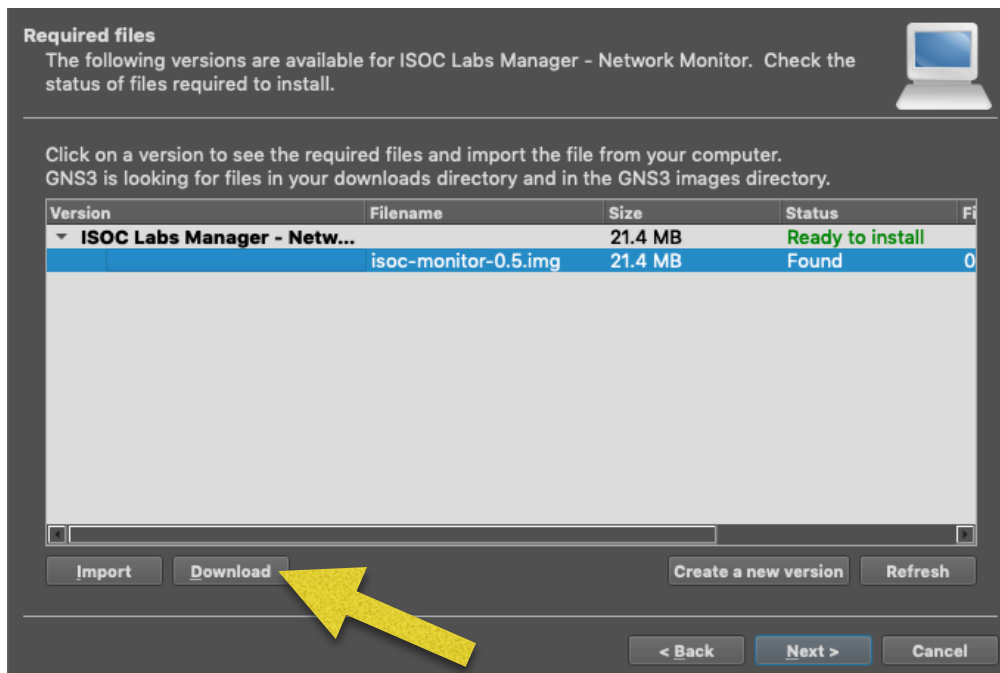
Click the download button to use a web browser to download the required files. Once all files have been downloaded you can proceed. Repeat this process for each `.gns3a` file.

Verification

If all the steps above have been performed the web service should be available now. Type the name of the server in your browser's address bar (in this example gns3.steffann.nl) and you should see a welcome screen.

Click the "Log in" link in the top right of the screen and log in with the super user you created. You should now be at the welcome screen again with an "Admin interface" link available in the top right corner. Click that link. You should be presented with the admin home screen with links to edit Groups, Users, Exercise templates, Exercises, IRR templates and Monitor templates.

Continue with the **Exercise Creation Guide** for further information.