# IT Technology

# Week 45 International Project

# Technical Documentation 4.0v

# 2017.11.11


Lillebaelt Academy of

Professional Higher Education

Author:

Milan Kristof Vince: mila1025@edu.eal.dk

November, 11, 2017

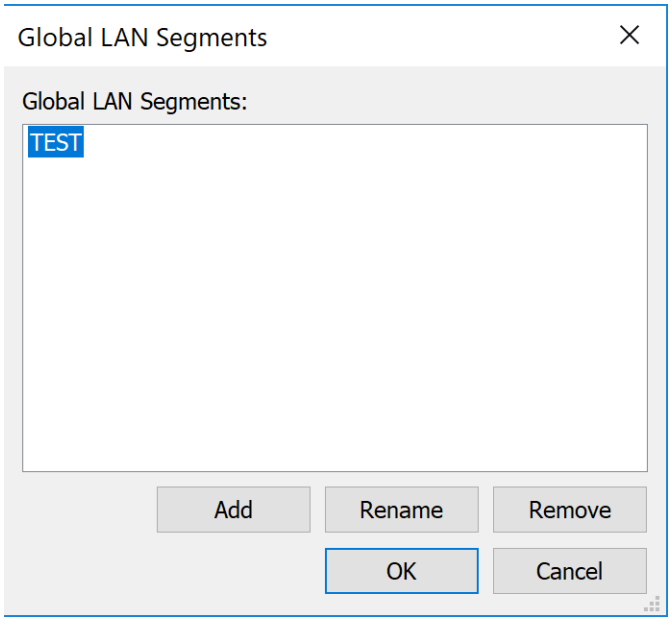
# Contents

## Introduction

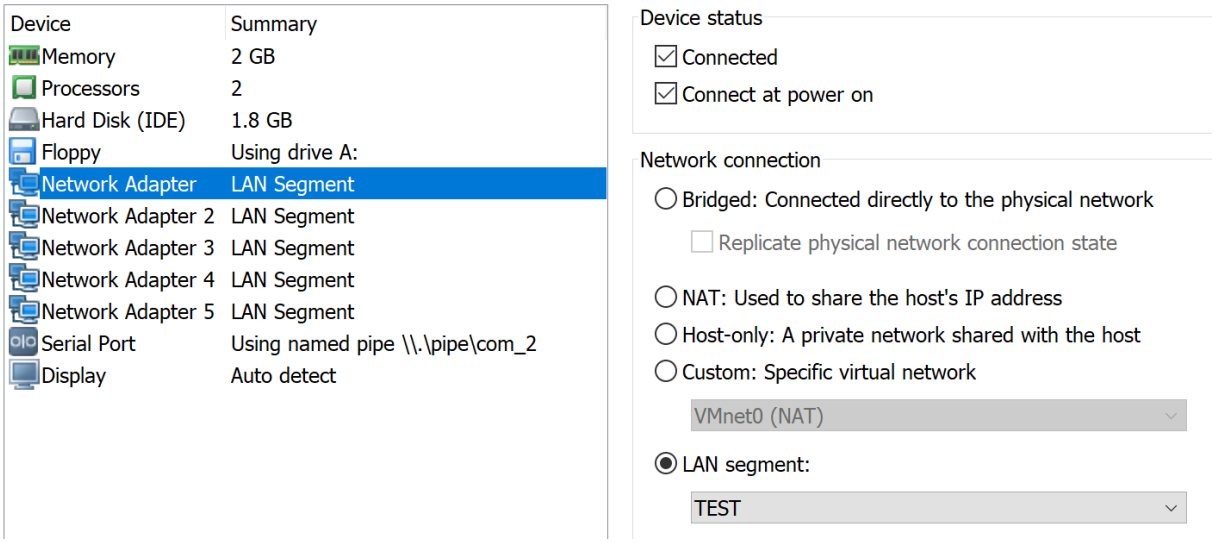
This document contains a guide about the technical instruction of the International Project at Week 45 at EAL in order to help all the students, who stucked somewhere throughout the project.

## Create LAN Segment as a connection point

At the particular VM, click on Settings, then Network Adapter, choose LAN Segments and create a new one:



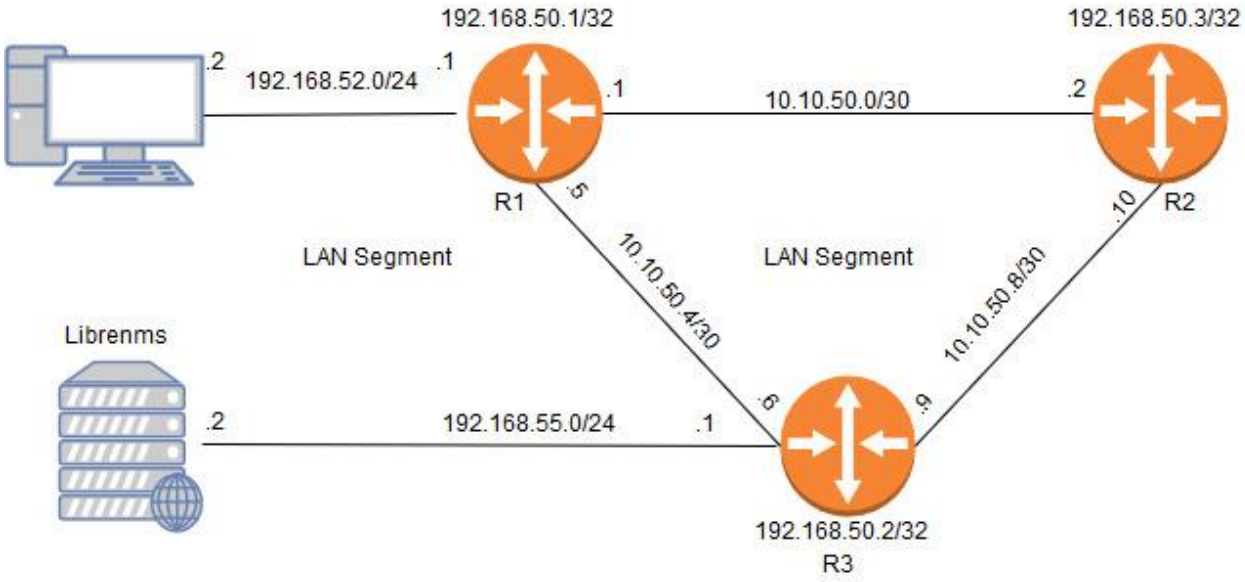

Then assign all the adapters to the chosen Segment:




Milan Kristof Vince:
mila1025@edu.eal.dk

## Network Topology

Network Diagram 1:




We are going to show the configuration of Network Diagram-1.

## Setup R1

First of all let's setup the physical interfaces, ge-0/0/0 and ge-0/0/1.

ge-0/0/0:

```
root# set interfaces ge-0/0/0 unit 0 family inet address 10.10.50.1/30
```

ge-0/0/1:

```
root# set interfaces ge-0/0/1 unit 0 family inet address 10.10.50.5/30
```

Loopback Address:

```
root# set interfaces lo0 unit 0 family inet address 192.168.50.1/32
```

Then commit changes:

```
root# commit
```

After commit it should look like:

```
root# show interfaces
ge-0/0/0 {
    unit 0 {
        family inet {
            address 10.10.50.1/30;
ge-0/0/1 {
    unit 0 {
        family inet {
            address 10.10.50.5/30;
lo0 {
    unit 0 {
        family inet {
            address 192.168.50.1/32;
```

OSPF protocol configuration on Router 1 :

```
set protocols ospf area 0.0.0.5 interface ge-0/0/0.0
set protocols ospf area 0.0.0.5 interface ge-0/0/1.0
set protocols ospf area 0.0.0.5 interface lo0
commit
```

After commit it should look like as:

```
root# show protocols
ospf {
    area 0.0.0.5 {
        interface ge-0/0/0.0;
        interface ge-0/0/1.0;
        interface lo0.0;
```

## Setup R2:

Let's setup the physical interfaces, ge-0/0/0 and ge-0/0/1.

ge-0/0/0:

```
root# set interfaces ge-0/0/0 unit 0 family inet address 10.10.50.2/30
```

ge-0/0/1:

```
root# set interfaces ge-0/0/1 unit 0 family inet address 10.10.50.10/30
```

Loopback Address:

```
root# set interfaces lo0 unit 0 family inet address 192.168.50.3/32
```

Then commit changes:

```
root# commit
```

After commit it should look like:

```
root# show interfaces
ge-0/0/0 {
    unit 0 {
        family inet {
            address 10.10.50.2/30;
ge-0/0/1 {
    unit 0 {
        family inet {
            address 10.10.50.10/30;
lo0 {
    unit 0 {
        family inet {
            address 192.168.50.3/32
```

OSPF protocol configuration on Router 2:

```
set protocols ospf area 0.0.0.5 interface ge-0/0/0.0
set protocols ospf area 0.0.0.5 interface ge-0/0/1.0
set protocols ospf area 0.0.0.5 interface lo0
commit
```

After commit it should look like as:

```
root# show protocols
ospf {
    area 0.0.0.5 {
        interface ge-0/0/0.0;
        interface ge-0/0/1.0;
        interface lo0.0;
```

## Setup R3:

Let's setup the physical interfaces, ge-0/0/0 and ge-0/0/1.

ge-0/0/0:

```
root# set interfaces ge-0/0/0 unit 0 family inet address 10.10.50.6/30
```

ge-0/0/1:

```
root# set interfaces ge-0/0/1 unit 0 family inet address 10.10.50.9/30
```

Loopback Address:

```
root# set interfaces lo0 unit 0 family inet address 192.168.50.2/32
```

Then commit changes:

```
root# commit
```

After commit it should look like:

```
root# show interfaces
ge-0/0/0 {
    unit 0 {
        family inet {
            address 10.10.50.6/30;
ge-0/0/1 {
    unit 0 {
        family inet {
            address 10.10.50.9/30;
```

```
lo0 {
    unit 0 {
        family inet {
            address 192.168.50.2/32;
```

OSPF protocol configuration on Router 2:

```
set protocols ospf area 0.0.0.5 interface ge-0/0/0.0
set protocols ospf area 0.0.0.5 interface ge-0/0/1.0
set protocols ospf area 0.0.0.5 interface lo0
commit
```

After commit it should look like as:

```
root# show protocols
ospf {
    area 0.0.0.5 {
        interface ge-0/0/0.0;
        interface ge-0/0/1.0;
        interface lo0.0;
```

## Verifying an OSPF Configuration of Router 1

To check ospf connectivity of let's go from the CLI enter the *show ospf interface* command.

```
root> show ospf interface
Interface           State   Area            DR ID           BDR ID
Nbrs
ge-0/0/0.0          DR      0.0.0.5         192.168.50.1    192.168.2.1
1
ge-0/0/1.0          DR      0.0.0.5         192.168.50.1    192.168.50.2
1
lo0.0               DR      0.0.0.5         192.168.50.1    0.0.0.0
0
```

Under DR ID, the IP address of the OSPF network's designated router appears.

## Verifying an OSPF Neighbors of Router 1

From CLI enter the *show ospf neighbour* command.

```
root> show ospf neighbor

Address          Interface              State     ID               Pri
Dead
10.10.50.2       ge-0/0/0.0             Full      192.168.2.1      128
31
10.10.50.6       ge-0/0/1.0             Full      192.168.50.2     128
35
```

The output shows a list of the device's OSPF neighbors and their addresses, interfaces, states etc.

Each neighbour shows a state of *Full.* Because full OSPF connectivity is established over a series of packet exchanges between clients, the OSPF link might take several seconds to establish.

## Verifying the Number of OSPF Routes of Router 1

```
root> show ospf route

Topology default Route Table:


Prefix             Path  Route      NH       Metric NextHop       Nexthop
                   Type  Type       Type             Interface
Address/LSP
192.168.2.1        Intra Router     IP             1 ge-0/0/0.0
10.10.50.2
192.168.50.2       Intra Router     IP             1 ge-0/0/1.0
10.10.50.6
10.10.50.0/30      Intra Network    IP             1 ge-0/0/0.0
10.10.50.4/30      Intra Network    IP             1 ge-0/0/1.0
10.10.50.8/30      Intra Network    IP             2 ge-0/0/0.0
10.10.50.2


ge-0/0/1.0    10.10.50.6
192.168.50.1/32    Intra Network    IP             0 lo0.0
192.168.50.2/32    Intra Network    IP             1 ge-0/0/1.0
10.10.50.6
192.168.50.3/32    Intra Network    IP             1 ge-0/0/0.0
10.10.50.2
```

Routes are shown with a route type of Network, and loopback addresses are shown with a route type of Router.

## Set MPLS Mode packet based

You must enable MPLS mode packet base under security, otherwise the SRX will work as a firewall and OSPF won't work. They are designed as pure firewall while the branch office devices are designed for the multipurpose of routing switching and security. Enabling MPLS packet base will change the device for 'Packet-based' mode. SRX works in 'flow based' by default.

Change the mode to packet-based using following command:

```
[edit]
root# set security forwarding-options family mpls mode packet-based
```

Then commit

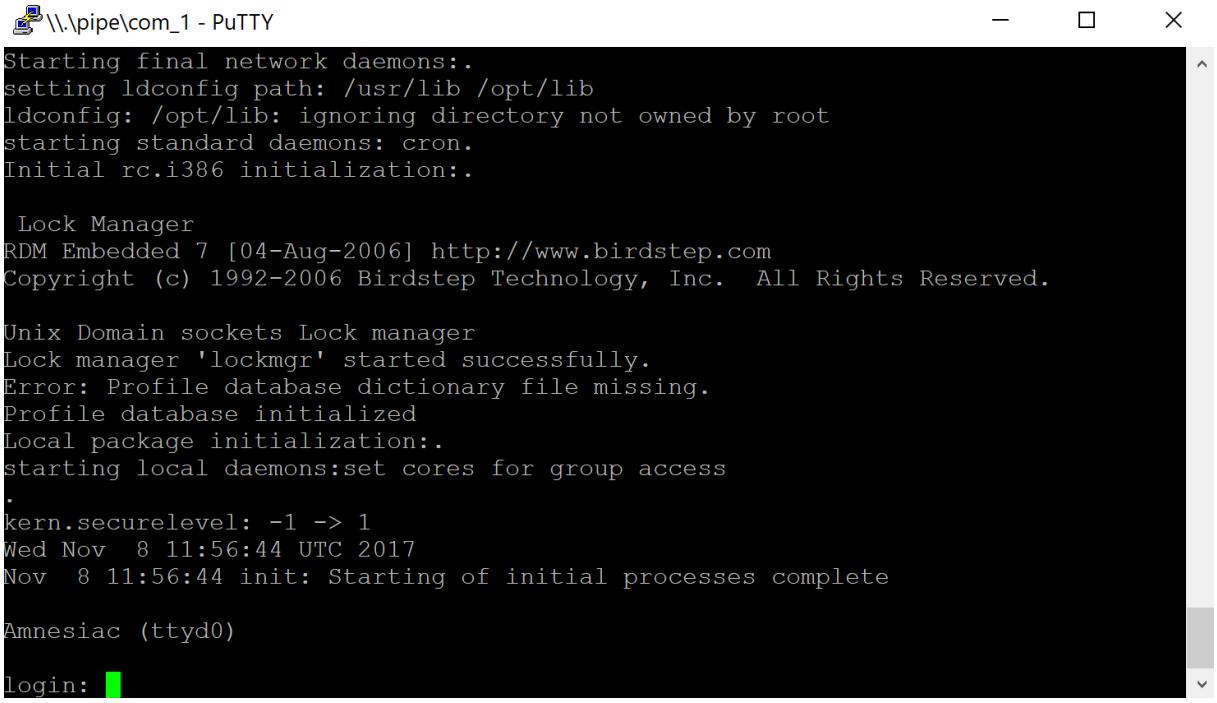
```
[edit]
root# commit
```

## Using Serial Port on the Host Computer with Putty

We will set up a Virtual Serial port in the virtual machine settings to be able to connect by Putty to the SRX-s.

To install virtual serial ports on the host computer by using the named pipe, take the following steps:

1. Open settings at the particular VM
2. Click to Serial Port
3. Choose "Use named pipe:"
4. Under "named pipe:" type: \\.\pipe\com_1 (use com_2 for the next device)
5. Set the first option to "The end is the server."
6. Set the second option to "The other end is an application."
7. Click OK to save a settings.
8. Open Putty
9. Go to "Serial Connection type:"
10. Paste \\.\pipe\com_1 to the text box.
11. Click open, then you should be able to see:




## Adding a Webserver to the topology

Reading the requirements, one of them to setup a web server, (optional). This short section will show how to make it work.

### Install Apache2

Apache is open source web-server software that powers much of the web today. It is maintained by apache-http-project.

Open your terminal and type in commands:

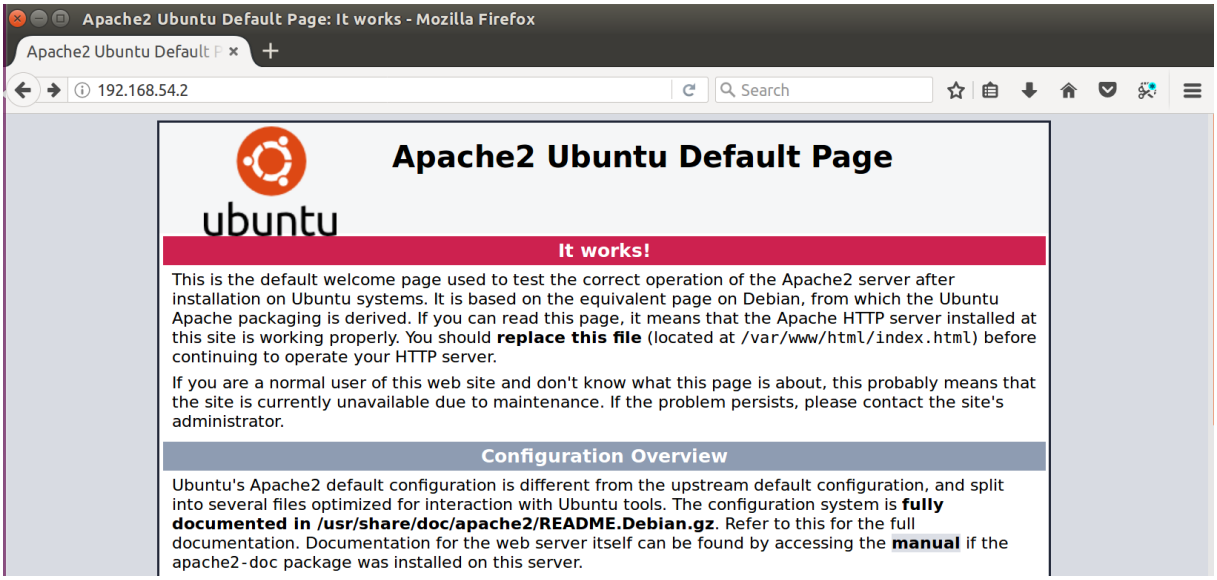
```
sudo apt-get update

sudo apt-get install apache2
```

Check if apache2 is installed properly:

```
sudo service apache2 restart
```

Open your web-browser while typing in localhost 127.0.0.1 or the ip-address of your server (192.168.54.2):

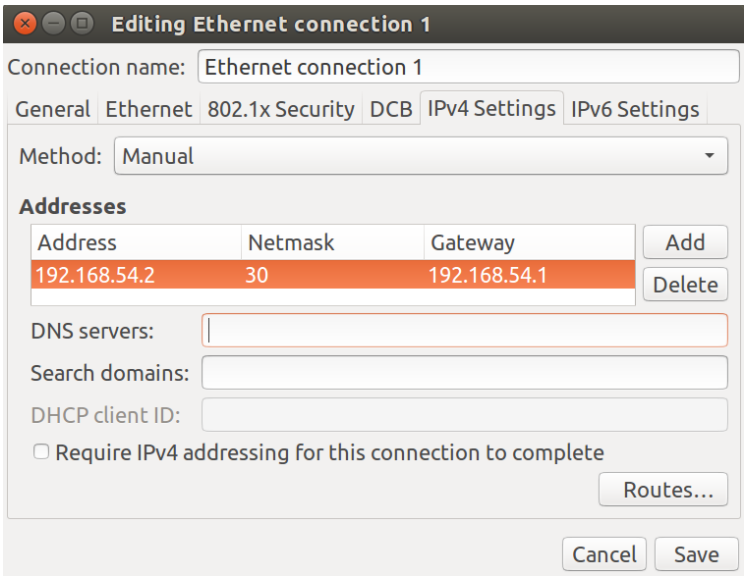



You can even install the GUI, because the server installation media doesn't install a GUI by, default but it's just one package installation away:

```
sudo apt-get update

sudo apt-get install ubuntu-desktop
```

To add an IPv4 address to the server click at networking and edit connections. At edit, just choose IPv4 and we can add our server manually:




The gateway address has to be the same as the interface address of that particular vSRX, that we planned to connect with the webserver.

## Install LibreNMS Monitoring Tool

LibreNMS is a open-source monitoring tool, great thing to monitor our devices on our network.

With great graphs, a truck-load of notification options, including PushBullet, it's everything you want from a network tool.

**The following document contains the instructions, how to install Librenms on Ubuntu 16.04 Desktop.**

First of all check the inet address of the local machine:

```
ifconfig
```

Then let's install MySQL. (I have done in root.)

```
apt-get install mariadb-server mariadb-client
```

```
systemctl restart mysql
```

```
mysql -uroot -p
```

Then create a database:

```
CREATE DATABASE librenms CHARACTER SET utf8 COLLATE utf8_unicode_ci;
```

(Hit Enter)

```
CREATE USER 'librenms'@'localhost' IDENTIFIED BY 'YOURPASSWORD';
```

(Hit Enter)

```
GRANT ALL PRIVILEGES ON librenms.* TO 'librenms'@'localhost';
```

(Hit Enter)

```
FLUSH PRIVILEGES;
```

(Hit Enter)

```
exit
```

Open with one text editor /etc/mysql/mariadb.conf.d/50-server.cnf file:

```
nano /etc/mysql/mariadb.conf.d/50-server.cnf
```

Within the [mysqld] section add:

```
innodb_file_per_table=1
```

```
sql-mode=""
```

We are going to configure the web server. (Apache2)

Install and Configure Apache:

```
apt-get install libapache2-mod-php7.0 php7.0-cli php7.0-mysql php7.0-gd
php7.0-snmp php-pear php7.0-curl snmp graphviz php7.0-mcrypt php7.0-json
apache2 fping imagemagick whois mtr-tiny nmap python-mysqldb snmpd php-net-
ipv4 php-net-ipv6 rrdtool git
```

Note that we installed php.mod 7.0 because it is one Ubuntu 16.04, The 14.05 can handle php5-version.

Open /etc/php/7.0/apache2/php.ini and search for 'date.timezone' (If you opened with nano can search with "Ctrl + W")

```
nano /etc/php/7.0/apache2/php.ini
```

`;date.timezone` (delete ; and set your time zone. Valid example is: Europe/Copenhagen)

```
date.timezone = Europe/Copenhagen
```

Do the same in /etc/php/7.0/cli/php.ini:

```
nano /etc/php/7.0/cli/php.ini
```

```
date.timezone = Europe/Copenhagen
```

(Important to insert the timezone what your client is configured for.)

Execute the following commands: (Enables php7.0 module, Disable mpm_event, Considering conflict mpm_event and mpm preford)

```
a2enmod php7.0
```

```
a2dismod mpm_event
```

```
a2enmod mpm_prefork
```

```
phpenmod mcrypt
```

Add librenms user:

```
useradd librenms -d /opt/librenms -M -r
```

```
usermod -a -G librenms www-data
```

Go back to /opt and clone the repository:

```
cd /opt
```

```
git clone https://github.com/librenms/librenms.git librenms
```

Configure Web interface:

```
cd /opt/librenms
```

```
mkdir rrd logs
```

```
chmod 775 rrd
```

Open /etc/apache2/sites-available/librenms.conf:

```
nano /etc/apache2/sites-available/librenms.conf
```

And add the following lines:

```
<VirtualHost *:80>
  DocumentRoot /opt/librenms/html/
  ServerName  YOURSERVERNAME
  CustomLog /opt/librenms/logs/access_log combined
  ErrorLog /opt/librenms/logs/error_log
  AllowEncodedSlashes NoDecode
  <Directory "/opt/librenms/html/">
    Require all granted
    AllowOverride All
    Options FollowSymLinks MultiViews
  </Directory>
</VirtualHost>
```

Execute the following commands: (Enabling site librenms, Enabling module rewrite, Restarting apache2)

```
a2ensite librenms.conf
a2enmod rewrite
systemctl restart apache2
```

Disable the default site if this is the only site, what you are planning to host:

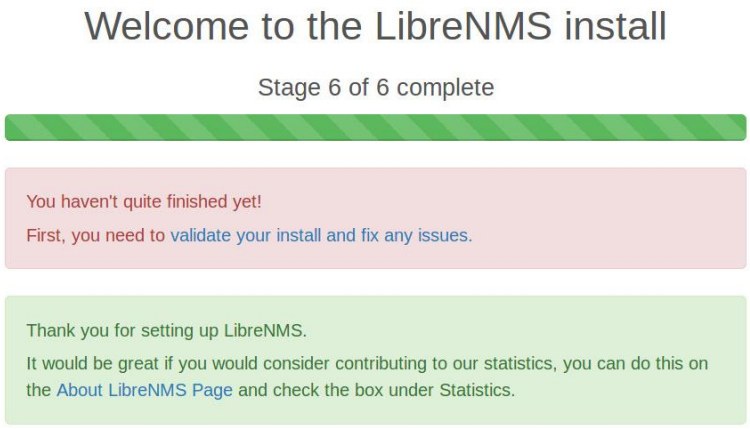
```
a2dissite 000-default
```

Web Installer

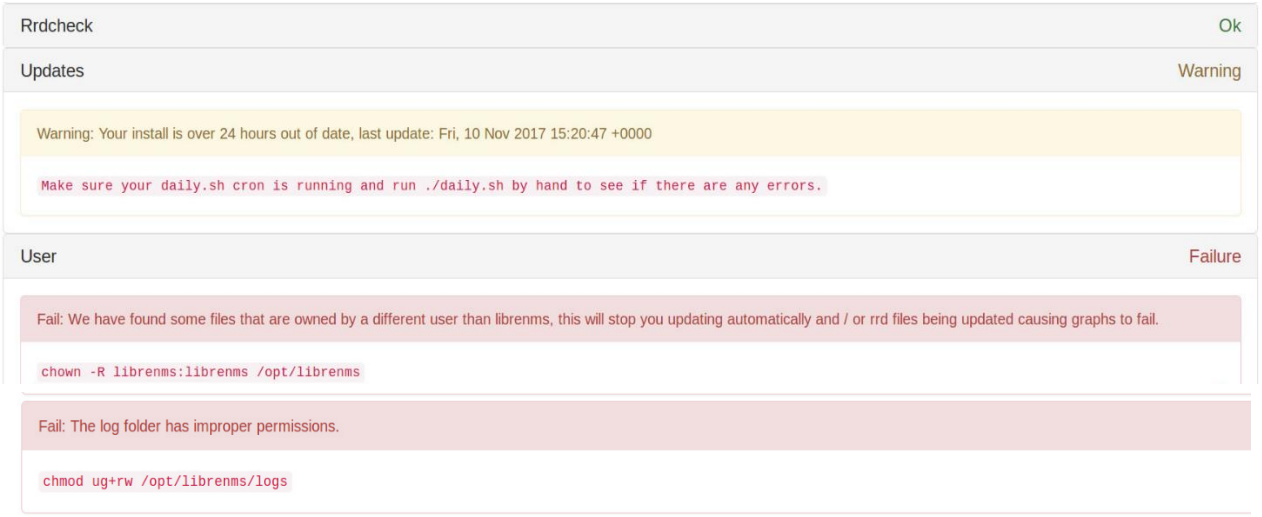
Open: http://localhost/install.php

And follow the instructions.

At Stage 5 it's possible that LibreNMS won't have access to create a config.php file, therefore we need to do it manually.




After Stage 6 it is possible that there can be some more steps, that Libre require for us, but it's a smart program and know what it needs. Follow the instructions.




*Stage 6*

## Configure snmpd

Execute the following two commands:

```
php addhost.php localhost public v2c
```

```
php discovery.php -h all
```

By default, the LibreNMS cronjob runs poller-wrapper.py with 16 threads. The current LibreNMS recommendation is to use 4 threads per core. The default if no thread count is 16 threads.
If the thread count needs to be changed, edit the cron file. Just add a number after poller-wrapper.py:

```
/opt/librenms/poller-wrapper.py 12 >> /dev/null 2>&1
```

```
cp librenms.nonroot.cron /etc/cron.d/librenms
```

```
cp /opt/librenms/snmpd.conf.example /etc/snmp/snmpd.conf
```

Open the configuration file:

```
nano /etc/snmp/snmpd.conf
```

And delete RANDOMSTRINGGOESHERE from the second line.

Then execute the following commands: (If curl isn't installed yet, do it: `apt-get install curl`)

```
curl -o /usr/bin/distro
https://raw.githubusercontent.com/librenms/librenms-
agent/master/snmp/distro
```

```
chmod +x /usr/bin/distro
```

```
systemctl restart snmpd
```

Make the Cron job:

```
cp librenms.nonroot.cron /etc/cron.d/librenms
```

Copy logrotate config.

LibreNMS keeps logs in /opt/librenms/logs. To rotate out the old logs with using the provided logrotate config file:

```
cp misc/librenms.logrotate /etc/logrotate.d/librenms
```

Change the ownership on this directory:

```
chown -R librenms:librenms /opt/librenms
```

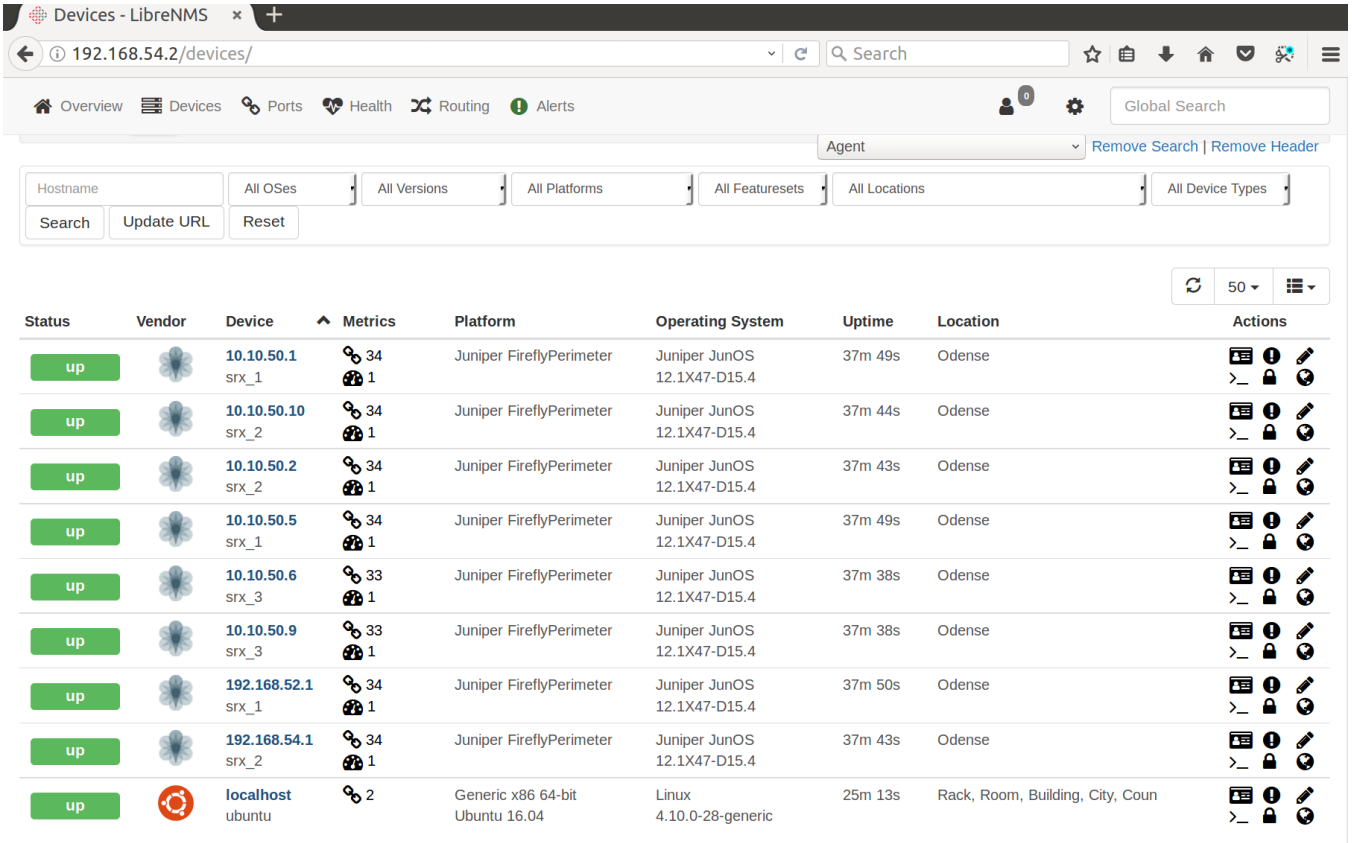
And for last Run validate.php as in the librenms directory:

```
cd /opt/librenms
./validate.php
```

To make sure it works, and saved all changes let's restart some of the programs:

```
systemctl restart snmpd
systemctl restart apache2
systemctl restart mysql
```

Final result should be:




*It's beautiful!*