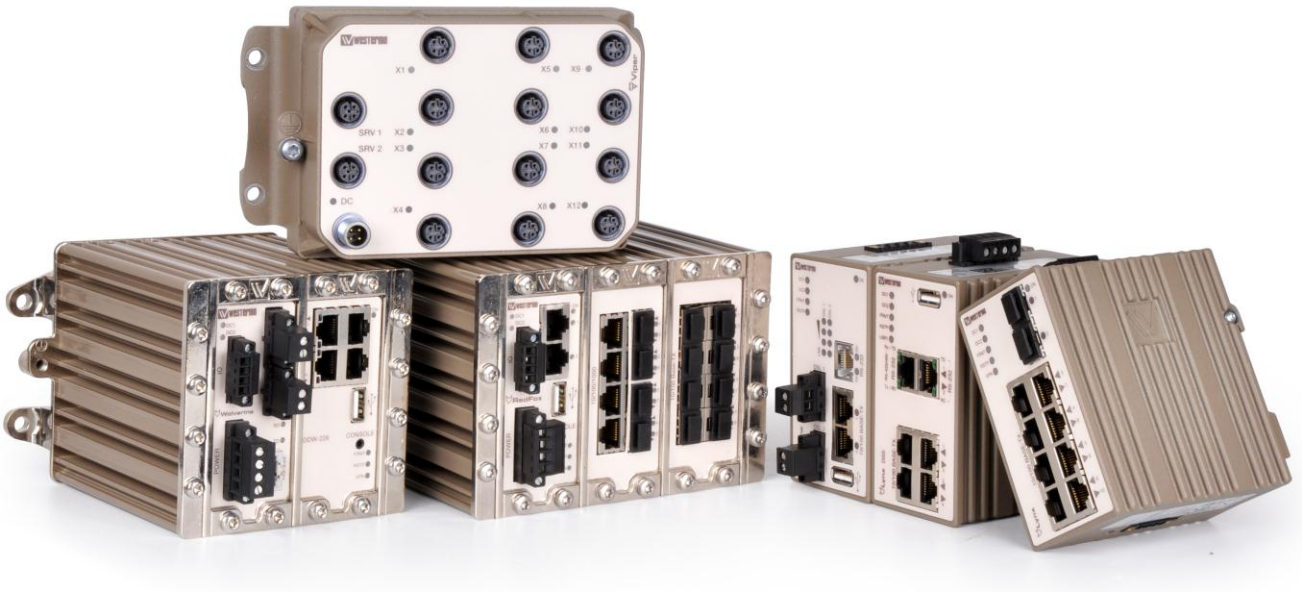
APPLICATION NOTE 002

Westermo WeOS Port Security

Configure IEEE 802.1X with RADIUS Authentication and MAC-Authentication using MAC-Filters



Disclaimer

Please Note! All settings not related to WeOS in this document are outside of the Westermo scope, third party software are not supported and might show settings that are not optimal for live networks.
Third party software configurations are only suggestions in order to get IEEE 802.1X up and running in a test environment.
In live applications security policies needs to be considered before IEEE 802.1X is deployed in the network.

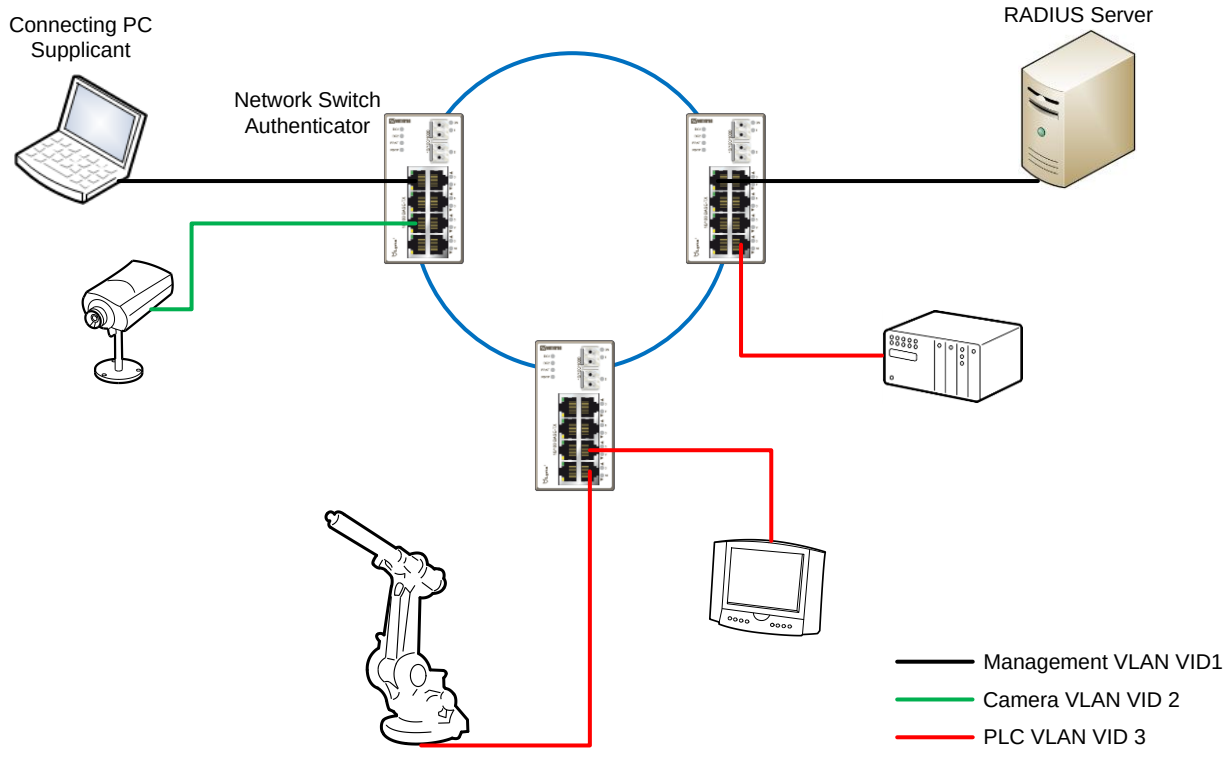
Application Note Network Layout

This Application Note shows how to configure port security on WeOS units using IEEE 802.1x and MAC-Authentication.

Background

IEEE 802.1X is a function that will demand a PC connecting to a switch to authenticate it self with a username and password. Otherwise the user will not be able to send packets into the network.
MAC-Authentication, filters packets based on source MAC-Addresses and only allowed addresses will be able to communicate on the network.
To deploy IEEE 802.1X in a network, a RADIUS authentication server is needed for the authentication of the client PC connecting to a WeOS switch.
IP-Cameras, PLCs etc usually do not support IEEE 802.1X, so in order to control ports where this type of equipment is to be connected MAC-Authentication can be used.
Combining IEEE 802.1X and MAC-Authentication with link alarms the network administrator will have a very good view of who connects to the network.
This Application Note will show how to protect the Management VLAN with IEEE 802.1X and the two VLANs for Cameras and PLCs with MAC-Authentication.

All configuration in this Application Note is done using WeOS version 4.15.0.
FreeRADIUS version 2.1.12 run on Ubuntu 14.04.
Supplicant configuration is done on Ubuntu 14.04 and MS Windows 7 Proffesional.



IEEE 802.1X

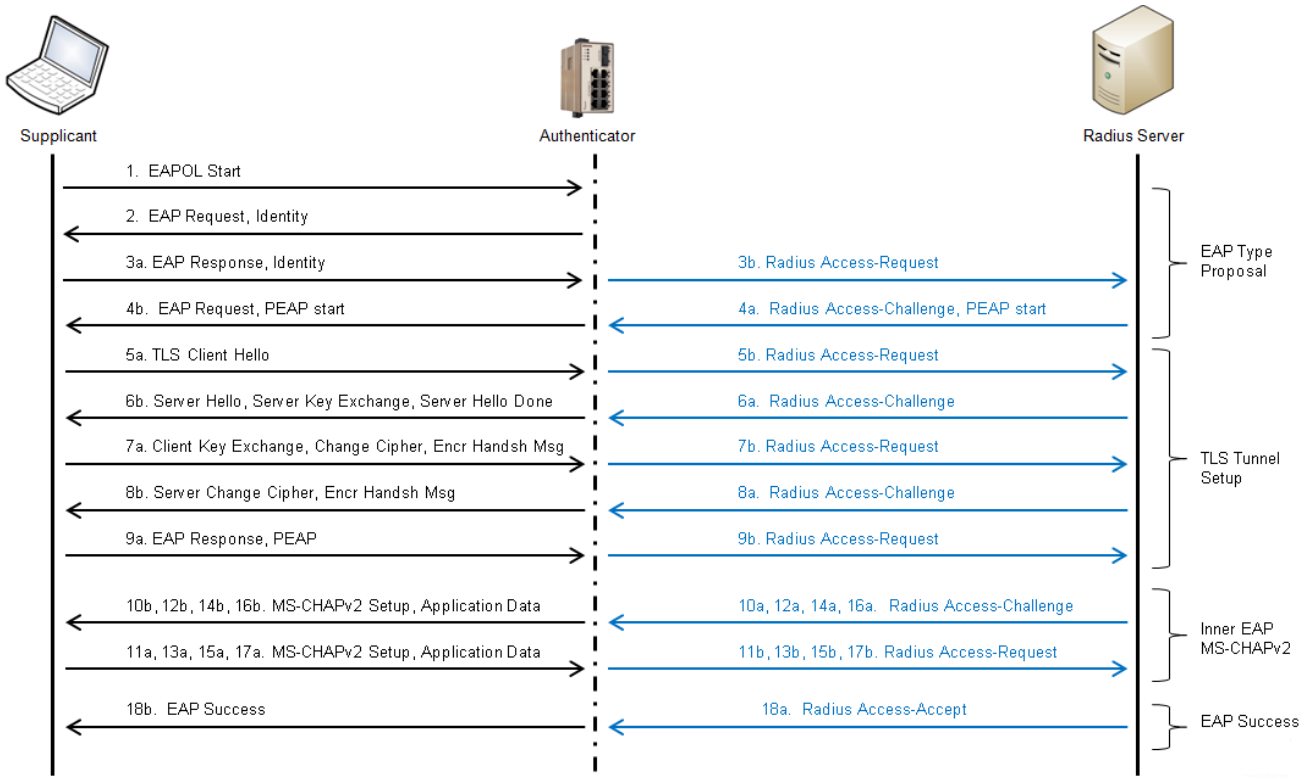
How the PC Authenticates

When using IEEE 802.1X the WeOS switch takes on the role of Authenticator which relays authentication requests from a client PC, the Supplicant, that wants to connect to the network.
The Supplicant uses the EAP (Extensible Authentication Protocol) protocol to authenticate itself in the network. The EAP packets are picked up by the Authenticator who relays the packets to the RADIUS Server using the RADIUS protocol.
For increased security the PEAP (Protected EAP) protocol is commonly used instead of plain EAP. PEAP sets up a TLS tunnel to encrypt the credentials exchange.
The configuration examples in this Application Note use PEAP for IEEE 802.1X authentication.

To setup IEEE 802.1X in the network three steps are needed:
1. Configuration of the RADIUS sever
2. Configuration of the Authenticator (WeOS Lynx-110-F2G)
3. Configuration of the Supplicant (Ubuntu 14.04 and Win7 PC)
All steps are covered in this document.

The following page shows what the IEEE 802.1X authentication process looks like.

Flowchart of the IEEE 802.1X PEAP authentication process



Wireshark log from the Supplicant

No.	Time	Source	Destination	Protocol	Length	Info
7	1.02443100	00:17:08:39:0e:81	Nearest	EAPOL	18	Start
8	1.03604800	00:07:7c:8b:20:20	00:17:08:39:0e:81	EAP	60	Request, Identity
9	1.03618800	00:17:08:39:0e:81	Nearest	EAP	28	Response, Identity
11	1.05056900	00:07:7c:8b:20:20	00:17:08:39:0e:81	EAP	60	Request, Protected EAP (EAP-PEAP)
12	1.05077500	00:17:08:39:0e:81	Nearest	TLSv1	237	Client Hello
13	1.08315200	00:07:7c:8b:20:20	00:17:08:39:0e:81	TLSv1	736	Server Hello, Certificate, Server Key Exchange, Server Hello Done
14	1.08710700	00:17:08:39:0e:81	Nearest	TLSv1	162	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
16	1.11620700	00:07:7c:8b:20:20	00:17:08:39:0e:81	TLSv1	83	Change Cipher Spec, Encrypted Handshake Message
17	1.11640200	00:17:08:39:0e:81	Nearest	EAP	24	Response, Protected EAP (EAP-PEAP)
18	1.12846100	00:07:7c:8b:20:20	00:17:08:39:0e:81	TLSv1	61	Application Data
19	1.12851700	00:17:08:39:0e:81	Nearest	TLSv1	98	Application Data, Application Data
20	1.13938600	00:07:7c:8b:20:20	00:17:08:39:0e:81	TLSv1	77	Application Data
21	1.14010100	00:17:08:39:0e:81	Nearest	TLSv1	162	Application Data, Application Data
22	1.15832100	00:07:7c:8b:20:20	00:17:08:39:0e:81	TLSv1	109	Application Data
23	1.15846200	00:17:08:39:0e:81	Nearest	TLSv1	98	Application Data, Application Data
24	1.16928300	00:07:7c:8b:20:20	00:17:08:39:0e:81	TLSv1	61	Application Data
25	1.16935700	00:17:08:39:0e:81	Nearest	TLSv1	98	Application Data, Application Data
26	1.18276700	00:07:7c:8b:20:20	00:17:08:39:0e:81	EAP	60	Success

```
⊞ Frame 7: 18 bytes on wire (144 bits), 18 bytes captured (144 bits) on interface 0
⊞ Ethernet II, Src: 00:17:08:39:0e:81 (00:17:08:39:0e:81), Dst: Nearest (01:80:c2:00:00:03)
⊞ 802.1x Authentication
```

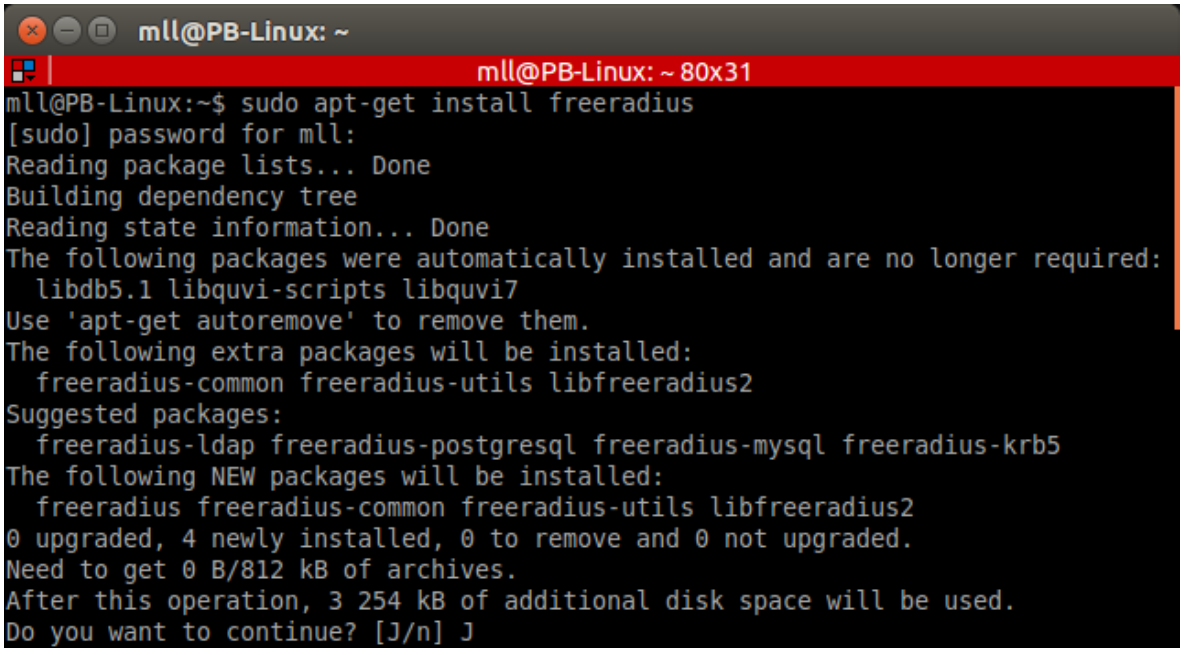
Wireshark log from the RADIUS Server

No.	Time	Source	Destination	Protocol	Length	Info
11	16.2992810	192.168.2.201	192.168.2.10	RADIUS	179	Access-Request(1) (id=56, l=137)
12	16.2998830	192.168.2.10	192.168.2.201	RADIUS	106	Access-Challenge(11) (id=56, l=64)
13	16.3107630	192.168.2.201	192.168.2.10	RADIUS	406	Access-Request(1) (id=57, l=364)
14	16.3317880	192.168.2.10	192.168.2.201	RADIUS	822	Access-Challenge(11) (id=57, l=780)
15	16.3463600	192.168.2.201	192.168.2.10	RADIUS	331	Access-Request(1) (id=58, l=289)
16	16.3531980	192.168.2.10	192.168.2.201	RADIUS	165	Access-Challenge(11) (id=58, l=123)
17	16.3636160	192.168.2.201	192.168.2.10	RADIUS	193	Access-Request(1) (id=59, l=151)
18	16.3640370	192.168.2.10	192.168.2.201	RADIUS	143	Access-Challenge(11) (id=59, l=101)
19	16.3748250	192.168.2.201	192.168.2.10	RADIUS	267	Access-Request(1) (id=60, l=225)
20	16.3754590	192.168.2.10	192.168.2.201	RADIUS	159	Access-Challenge(11) (id=60, l=117)
21	16.3873920	192.168.2.201	192.168.2.10	RADIUS	331	Access-Request(1) (id=61, l=289)
22	16.3883970	192.168.2.10	192.168.2.201	RADIUS	191	Access-Challenge(11) (id=61, l=149)
23	16.3987030	192.168.2.201	192.168.2.10	RADIUS	267	Access-Request(1) (id=62, l=225)
24	16.3992590	192.168.2.10	192.168.2.201	RADIUS	143	Access-Challenge(11) (id=62, l=101)
25	16.4101530	192.168.2.201	192.168.2.10	RADIUS	267	Access-Request(1) (id=63, l=225)
26	16.4107960	192.168.2.10	192.168.2.201	RADIUS	209	Access-Accept(2) (id=63, l=167)

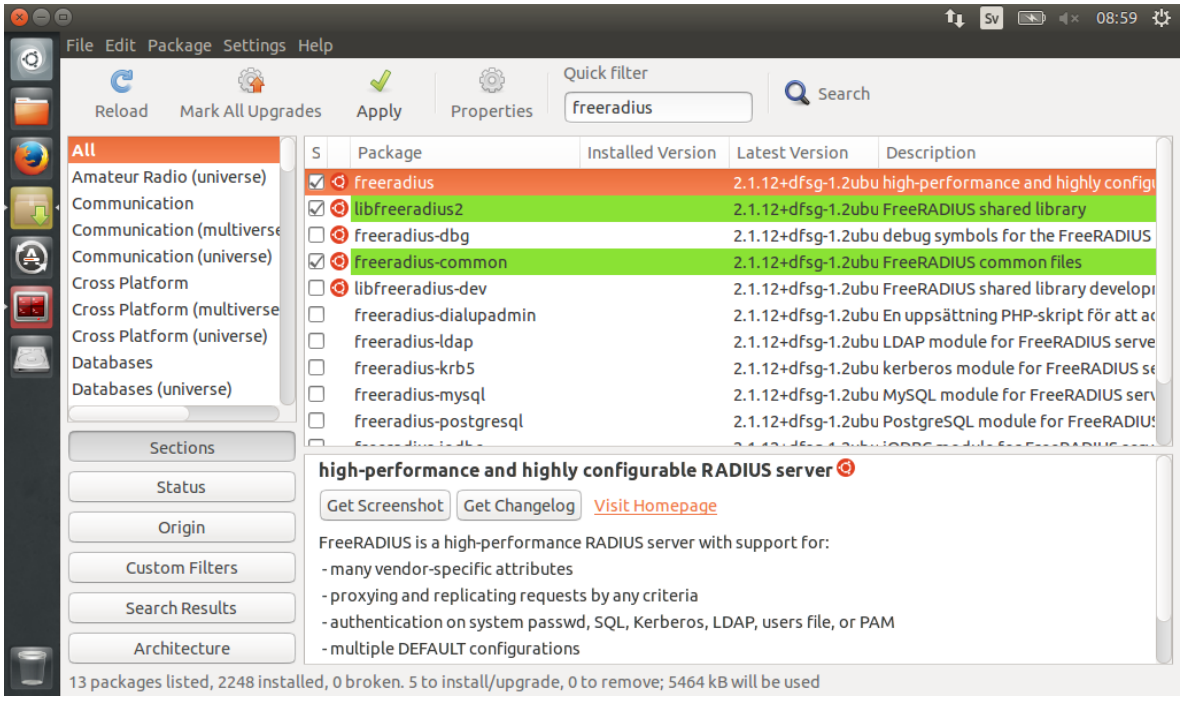
```
⊞ Frame 11: 179 bytes on wire (1432 bits), 179 bytes captured (1432 bits) on interface 0
⊞ Ethernet II, Src: 00:07:7c:8b:20:23 (00:07:7c:8b:20:23), Dst: 00:23:5a:8c:13:dc (00:23:5a:8c:13:dc)
⊞ Internet Protocol Version 4, Src: 192.168.2.201 (192.168.2.201), Dst: 192.168.2.10 (192.168.2.10)
⊞ User Datagram Protocol, Src Port: 33514 (33514), Dst Port: radius (1812)
⊞ Radius Protocol
```

1. Configure the FreeRADIUS Server on Ubuntu 14.04

Start by installing the FreeRADIUS server software using:
either the terminal with *sudo apt-get install freeradius*



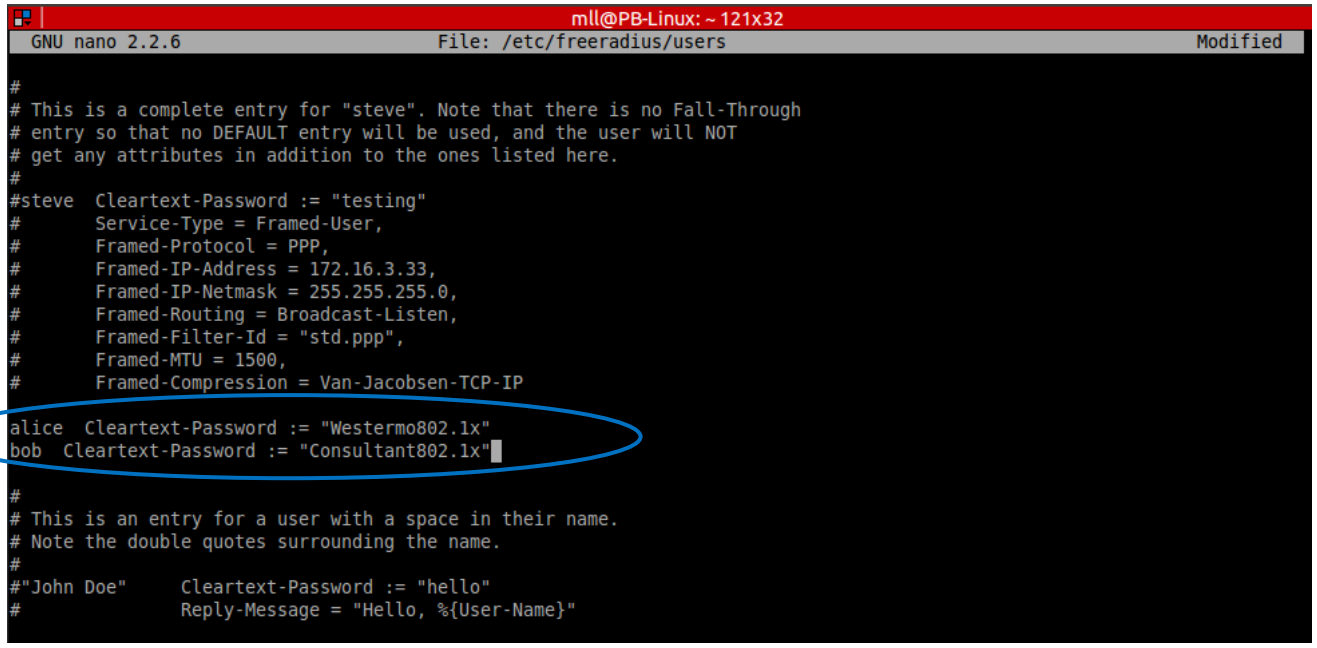
the Synaptic Packet Manager or Ubuntu Software Centre.



Configure the FreeRADIUS server by editing the *users*, *clients.conf* and if needed the *eap.conf* files in /etc/freeradius/

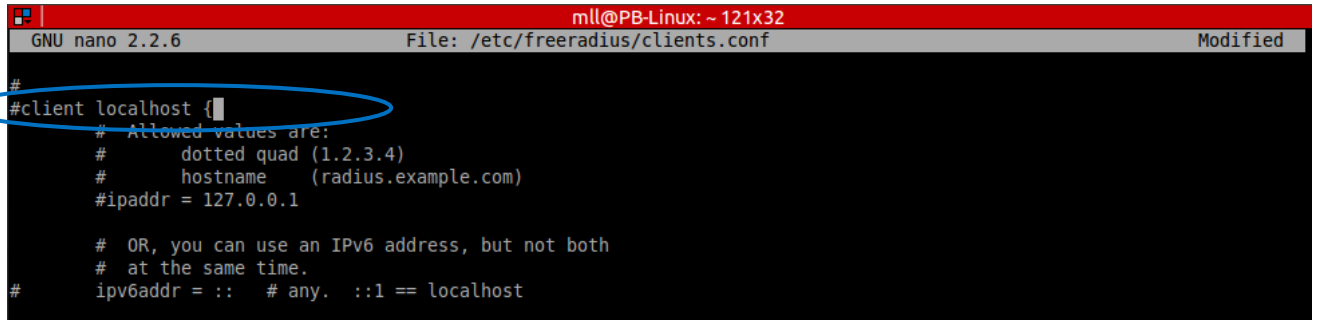
Example: sudo nano /etc/freeradius/users

In the *users* file add the Supplicants that are allowed to access the network.

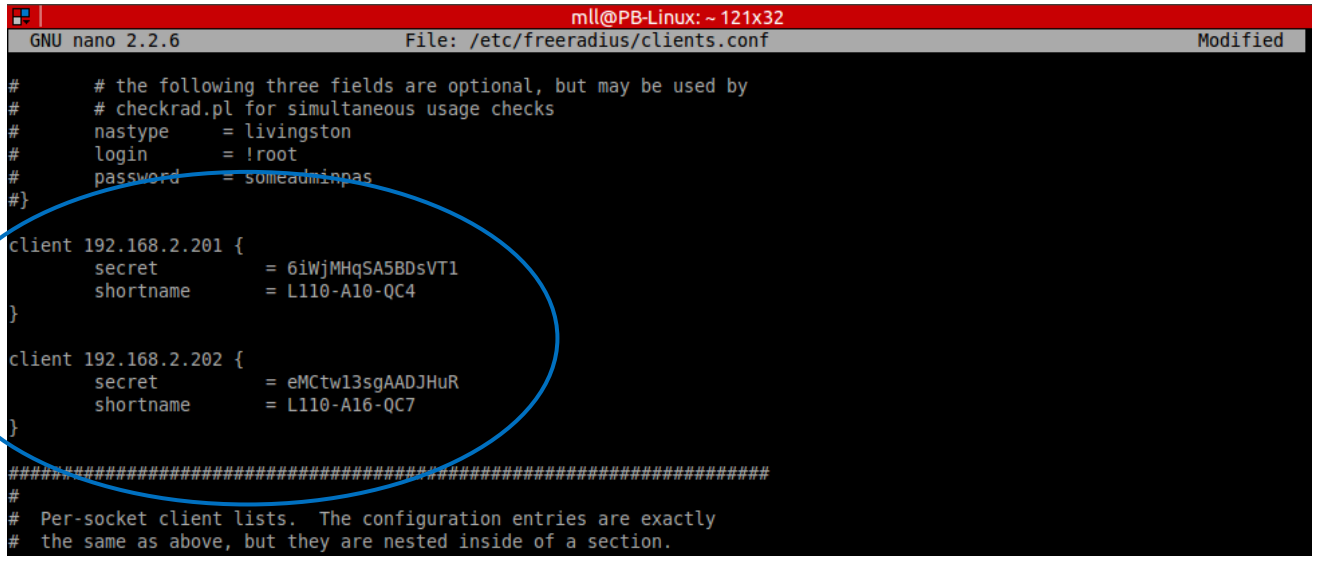


Clients.conf contains all Authenticators in the network.

Start by commenting out all lines associated with the client localhost if localhost access is not desired. Use the # sign to comment out a line in the RADIUS server config files.



Then add the Authenticators (switches).
client <IP-address of the switch>
Secret this is to most important component as the RADIUS protocol are completely dependent of the security of this password.
Therefore use complex passwords that are:
- Not words
- Composed of numbers, upper and lower case letters
- At least 10 characters long, preferably 16 characters long.
shortname is not necessary but can be added for ease of documentation



Edit *eap.conf* if an eap type needs to be disabled.

MD5 is the default protocol, to change this edit default_eap_type to the desired protocol, in this case PEAP. Finish by commenting out the MD5 part.

```
mll@PB-Linux: ~ 121x29
 GNU nano 2.2.6                File: /etc/freeradius/eap.conf

#  EAP types NOT listed here may be supported via the "eap2" module.
#  See experimental.conf for documentation.
#
        eap {
                #  Invoke the default supported EAP type when
                #  EAP-Identity response is received.
                #
                #  The incoming EAP messages DO NOT specify which EAP
                #  type they will be using, so it MUST be set here.
                #
                #  For now, only one default EAP type may be used at a time.
                #
                #  If the EAP-Type attribute is set by another module,
                #  then that EAP type takes precedence over the
                #  default type configured here.
                #
                default_eap_type = peap
```

```
                #md5 {
                #}
```

If the FreeRADIUS Server is run in debug mode it is easier to see if Supplicants are beeing authenticated correctly. FreeRADIUS is run as a service by default so it needs to be stopped before it can be started in debug mode, to do this use these commands: *sudo service freeradius stop* and then *sudo freeradius -X* for debug mode.

```
mll@PB-Linux: ~ 80x24
mll@PB-Linux:~$ sudo service freeradius stop
[sudo] password for mll:
freeradius stop/waiting
mll@PB-Linux:~$ sudo freeradius -X
FreeRADIUS Version 2.1.12, for host i686-pc-linux-gnu, built on Feb 24 2014 at 1
5:00:10
Copyright (C) 1999-2009 The FreeRADIUS server project and contributors.
There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A
PARTICULAR PURPOSE.
You may redistribute copies of FreeRADIUS under the terms of the
GNU General Public License v2.
Starting - reading configuration files ...
including configuration file /etc/freeradius/radiusd.conf
including configuration file /etc/freeradius/proxy.conf
including configuration file /etc/freeradius/clients.conf
including files in directory /etc/freeradius/modules/
including configuration file /etc/freeradius/modules/chap
including configuration file /etc/freeradius/modules/preprocess
including configuration file /etc/freeradius/modules/ldap
including configuration file /etc/freeradius/modules/detail.log
including configuration file /etc/freeradius/modules/mschap
including configuration file /etc/freeradius/modules/otp
including configuration file /etc/freeradius/modules/acct_unique
including configuration file /etc/freeradius/modules/wimax
```

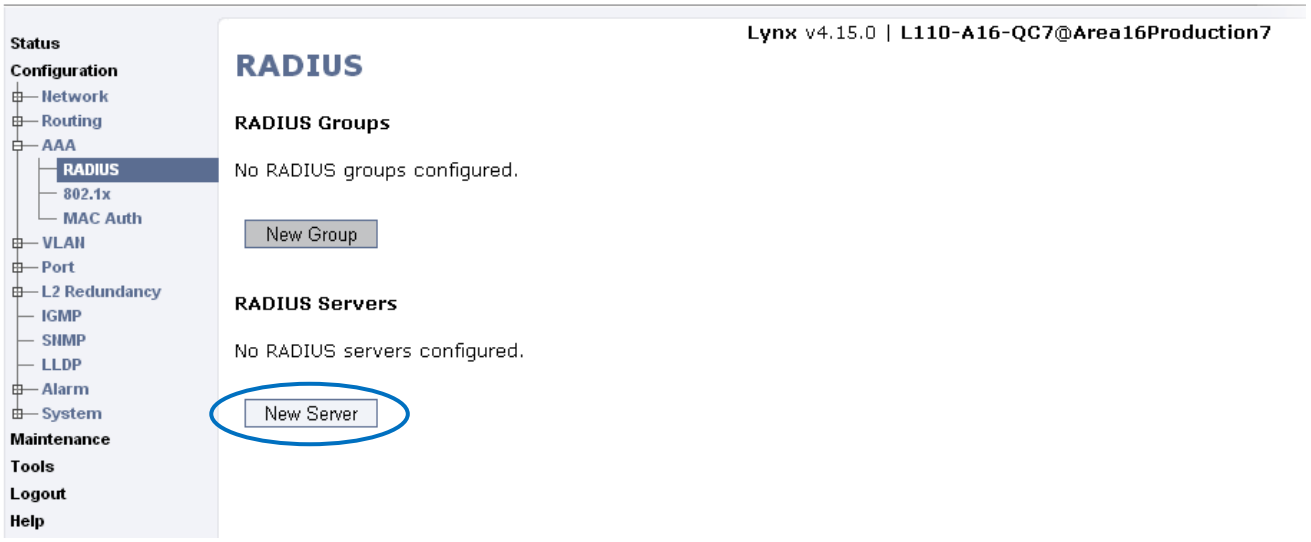
2. Configure IEEE 802.1X on WeOS Units

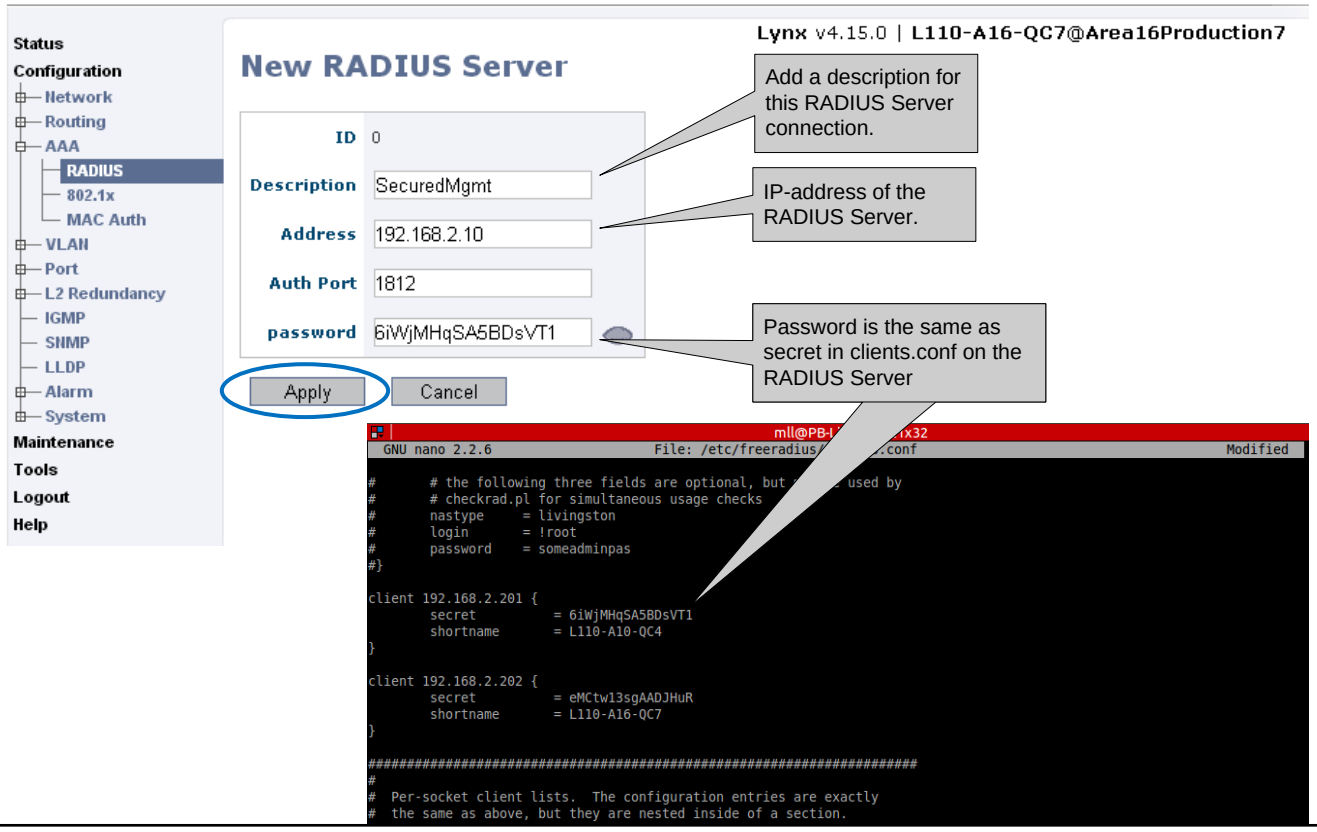
There are three steps to complete in order to configure IEEE 802.1X on a WeOS unit.

i. Configure the connection to the RADIUS Server.
ii. Setup which RADIUS Server to be used with IEEE 802.1x.
iii. Finally activate IEEE 802.1X per VLAN and assign ports to be authenticated.

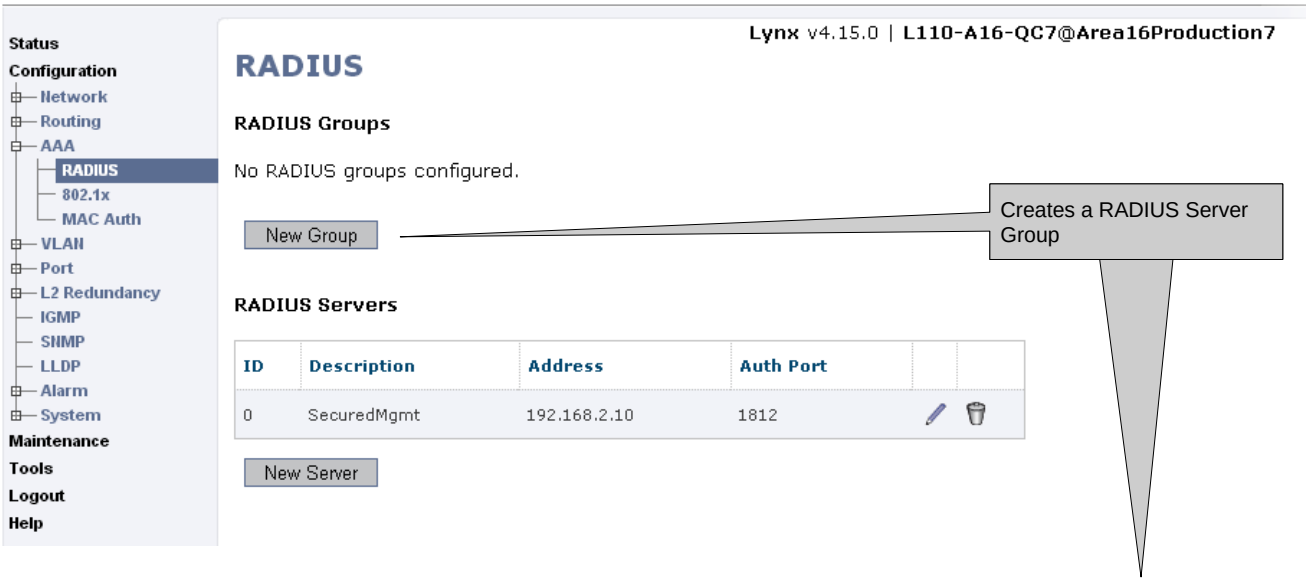
i. Configure the connection to the RADIUS Server.
Up to six RADIUS Server connections can be added to the list.

Go to **Configuration** -> **AAA** -> **RADIUS** to configure a connection to a RADIUS Server.

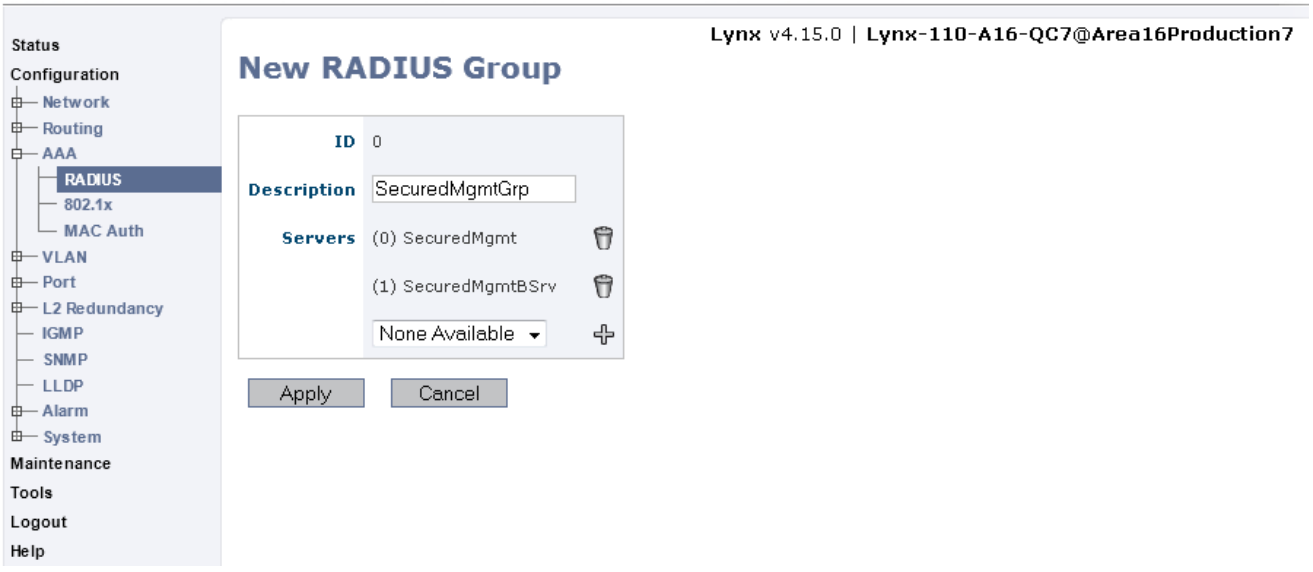




The RADIUS Server connection is now added to the list of usable servers.

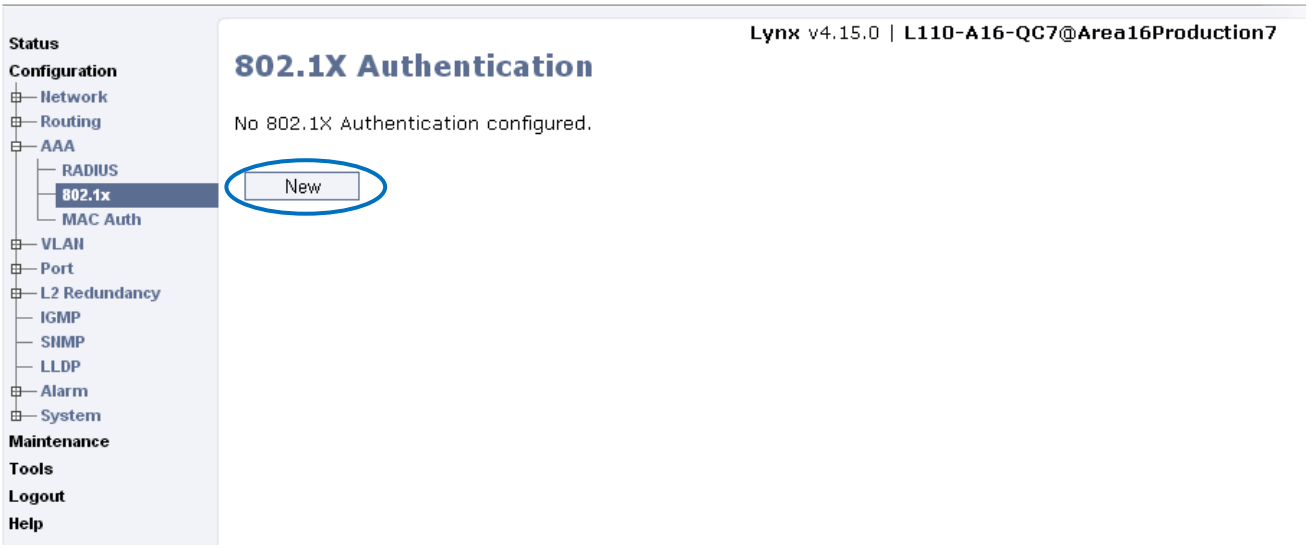


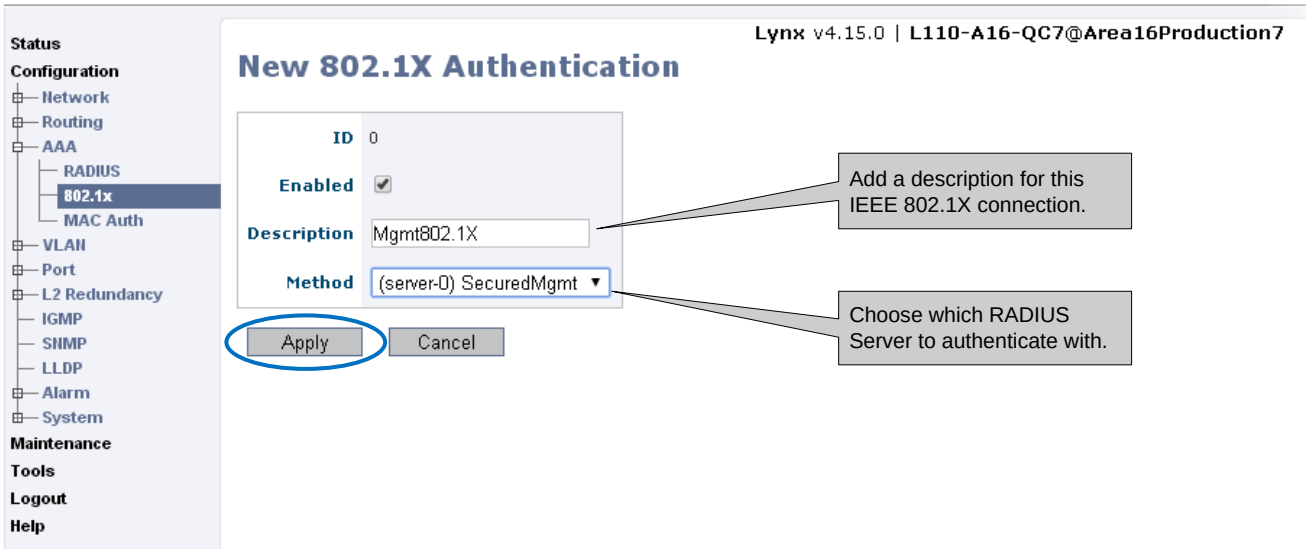
If needed a group of RADIUS Servers can be configured in order to create server redundancy.
Two groups with up to three RADIUS Servers each can be set up.
The servers are tried in the order they are added to the list so it is important that the primary server is added first.



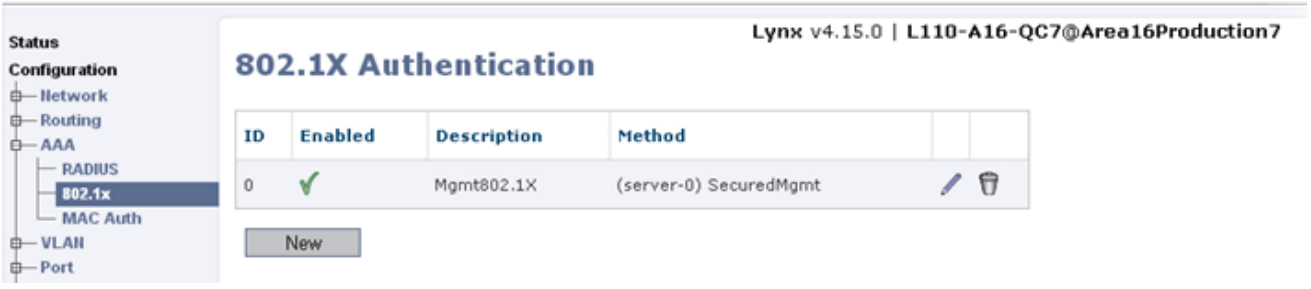
ii. Setup which RADIUS Server to be used with IEEE 802.1x.

Go to **Configuration** -> **AAA** -> **802.1x** to associate a RADIUS Server connection with IEEE 802.1X.



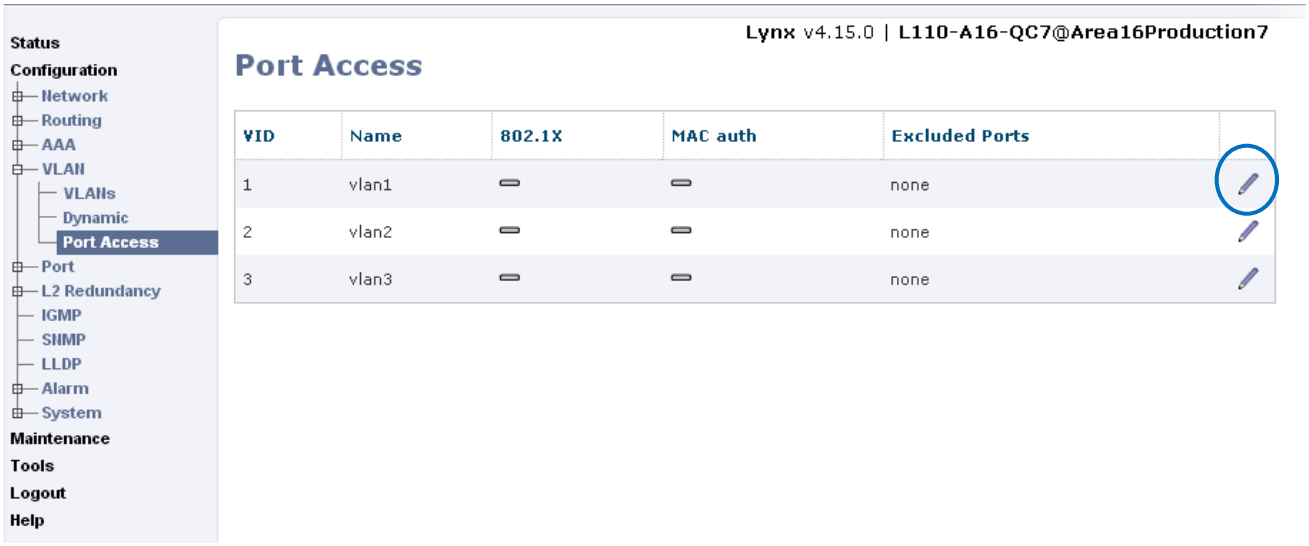


Now IEEE 802.1X has a RADIUS Server to connect to.

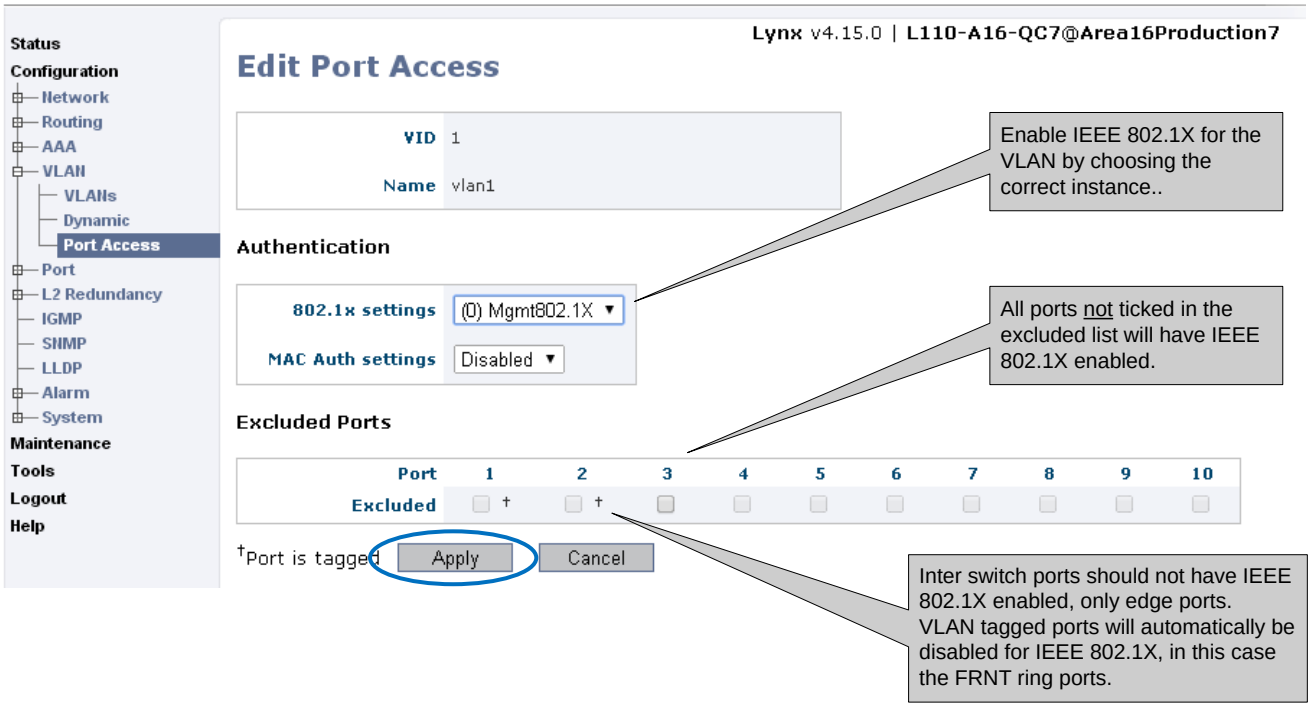


iii. Finally activate IEEE 802.1X per VLAN and assign ports to be authenticated.

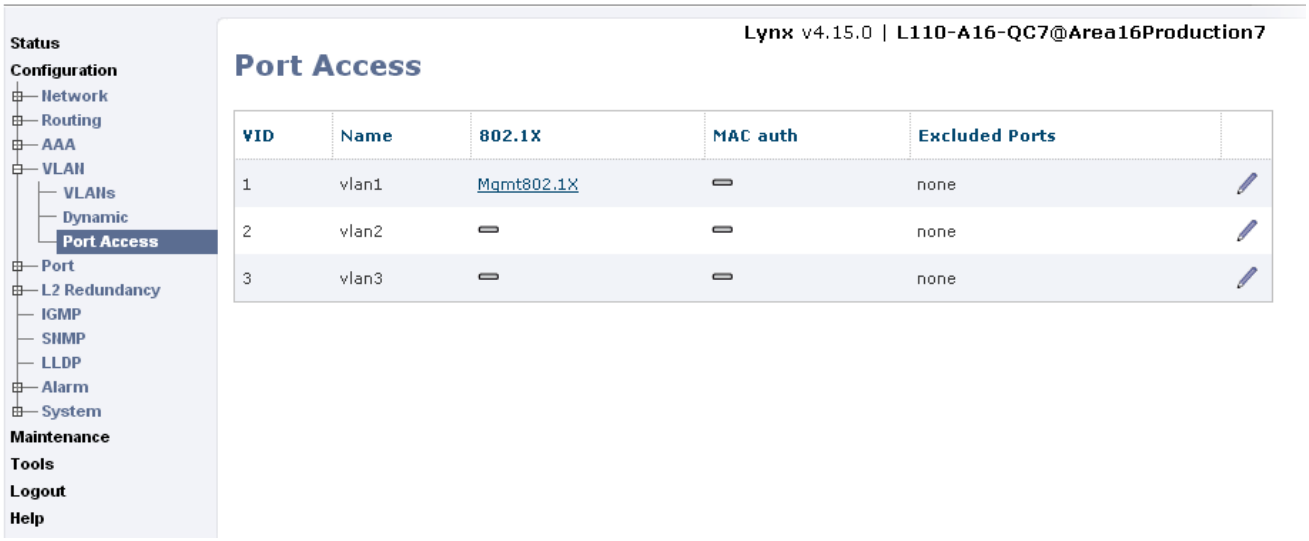
Go to **Configuration** -> **VLAN** -> **Port Access** to activate IEEE 802.1X on the Management VLAN.



In this case it is the Management VLAN 1 that should be protected using IEEE 802.1X.

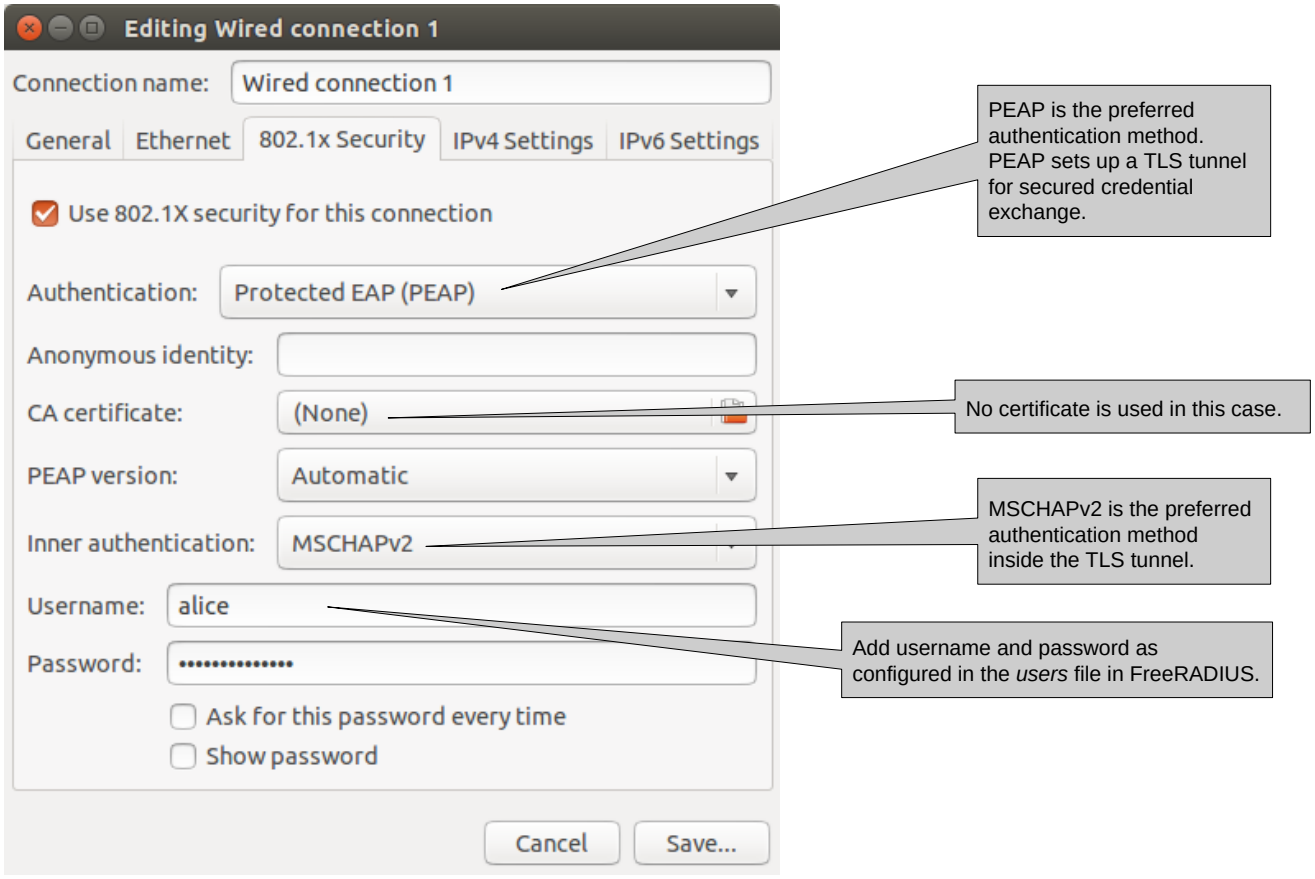


IEEE 802.1X is now enabled for the Management VLAN, all users connecting to port 3 of the switch must authenticate itself with the correct username and password inorder to connect to the network.

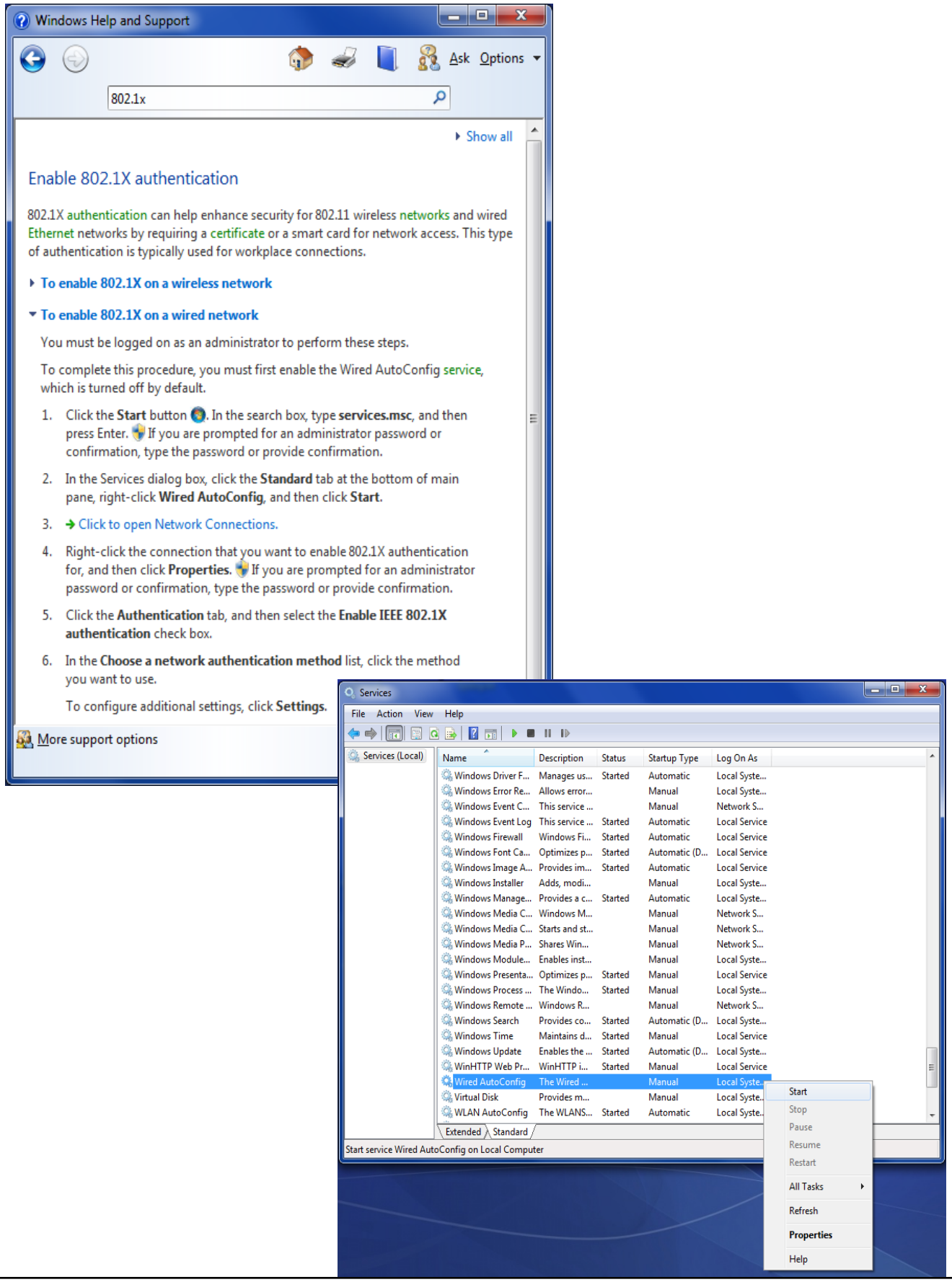


3. Activate IEEE 802.1X on Ubuntu 14.04 and MS Windows 7 Professional

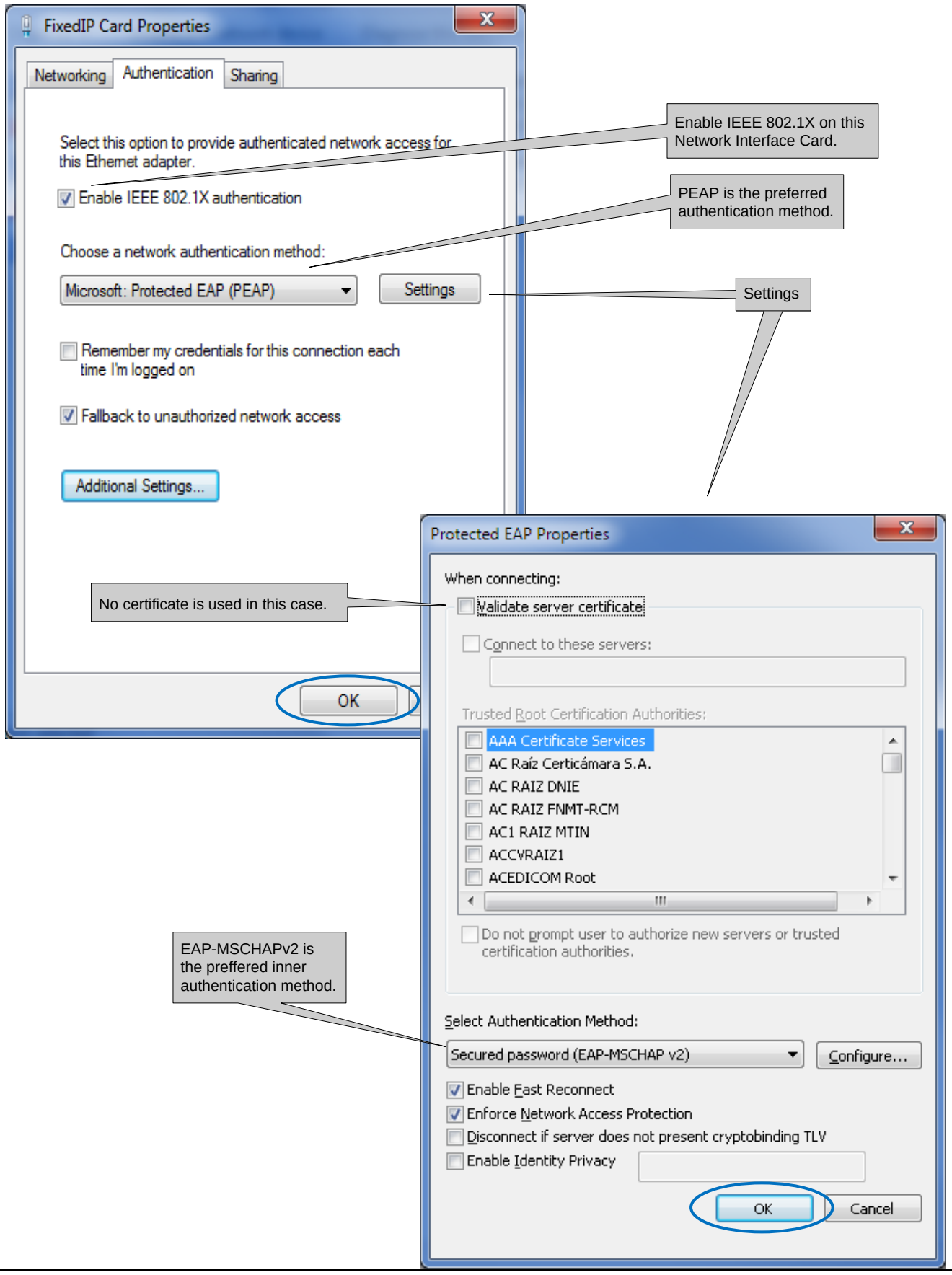
In Ubuntu 14.04 IEEE 802.1X is available by default in the NIC settings.



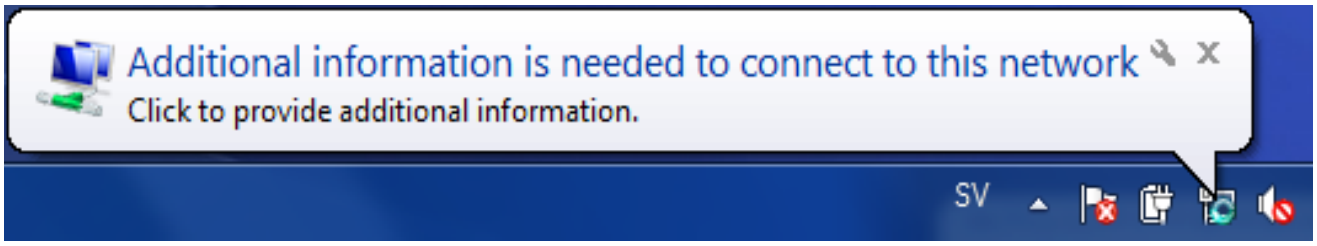
In MS Windows Professional the wired IEEE 802.1X service must first be enabled.

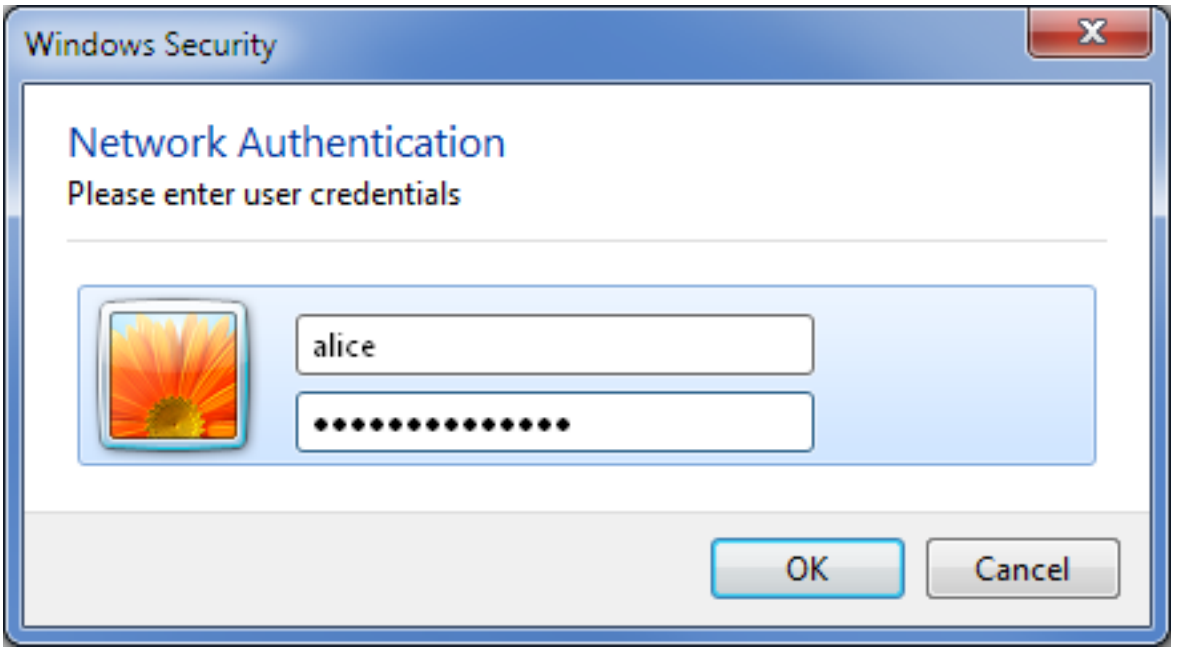


Then IEEE 802.1X can be activated in the NIC settings.

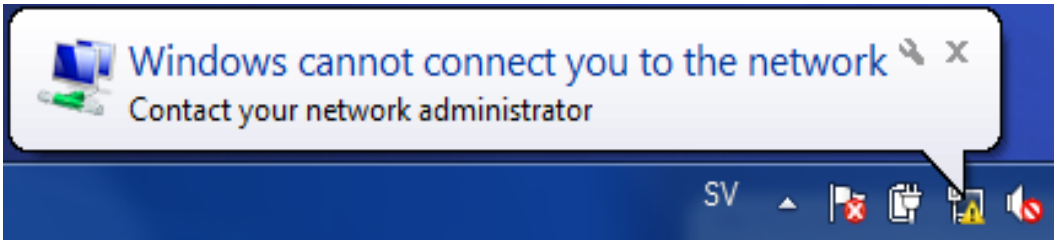


In MS Windows 7 the user will be notified that user credentials are needed to connect to this network.





If the wrong credentials are used the user will not be able to connect to the nework.



MAC-Authentication

For units that do not support IEEE 802.1X MAC-Authentication can be used.
MAC-Authentication is a list of allowed MAC-addresses that can connect to the network.
As of WeOS 4.15.0 a local database is used for filtering the MAC-addresses.

The MAC-address filters can use wildcards to authenticate any equipement from a specific vendor.

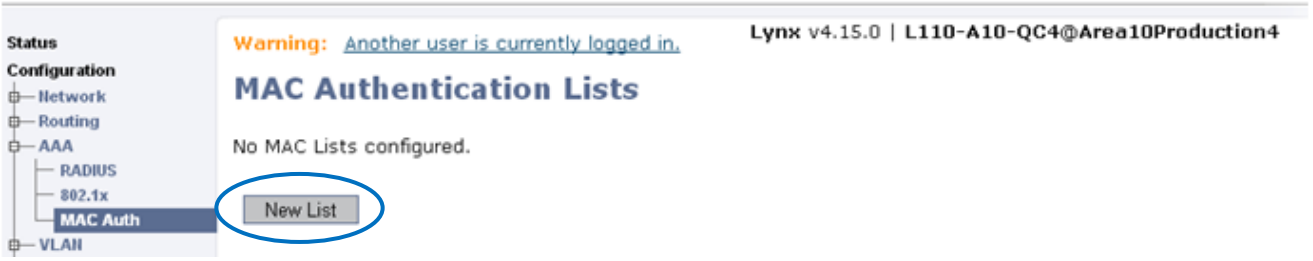
Example

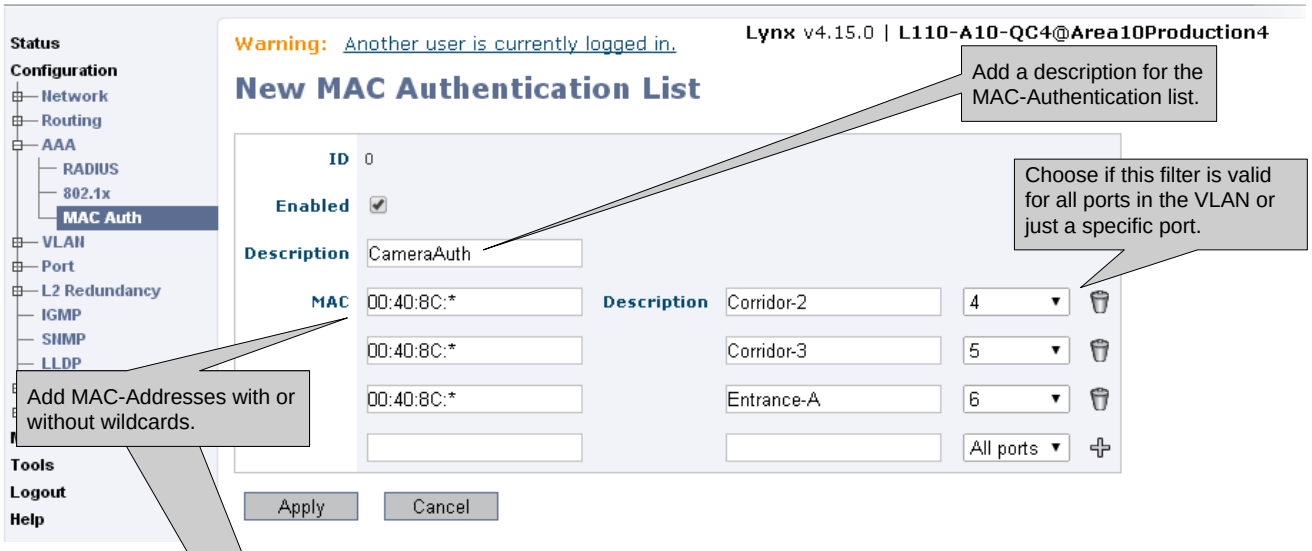
```
mac-auth-0/#>  mac match 00:D8:AA:2C:85:01
 or with wildcard...
mac-auth-0/#>  mac match 00:80:C8:*
 or with wildcard, limit filter, and description ...
mac-auth-0/#>  mac match 00:D8:BB:C5:* limit 1/2 description ''Laser printers on 1/2''
```

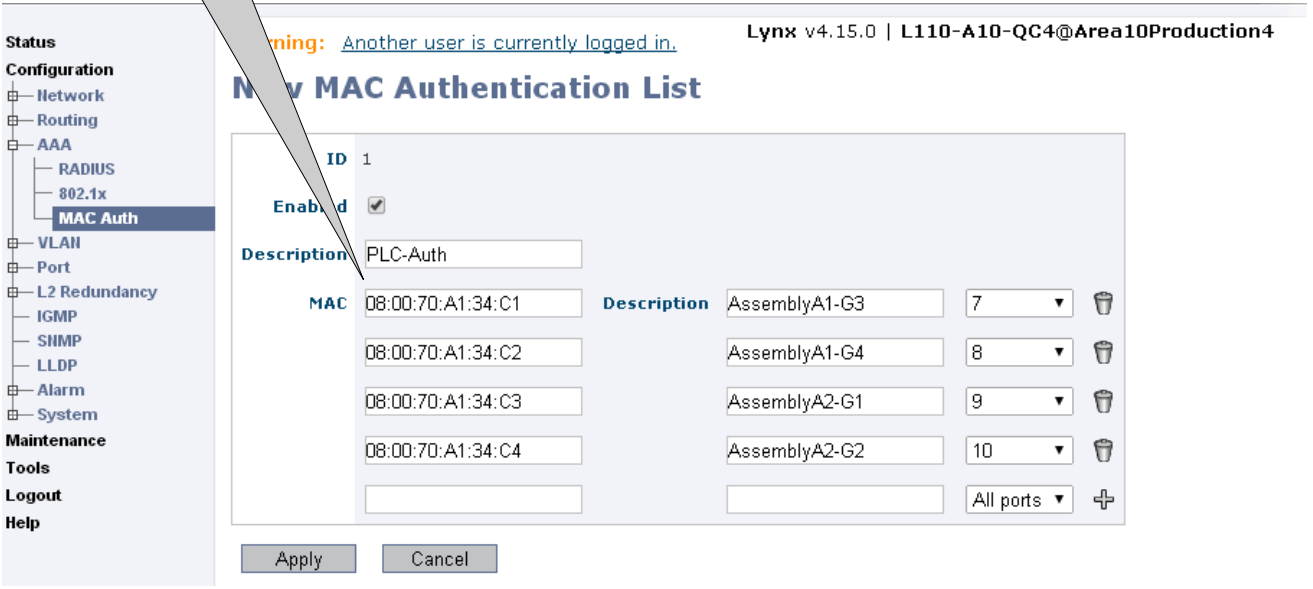
There are two steps needed for setting up MAC-Authentication on WeOS units.
1. Create local MAC-Authentication database lists.
2. Associate the local databases with the appropriate VLANs.

1. Create local MAC-Authentication database lists

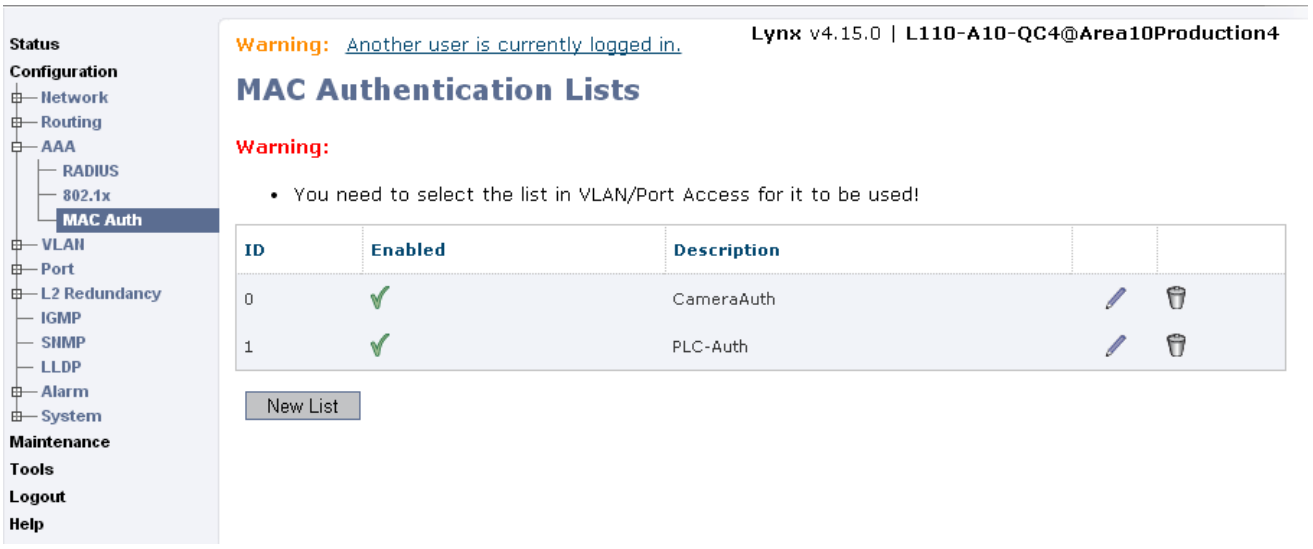
Go to **Configuration** -> **AAA** -> **MAC Auth** to configure MAC-Authentication lists.





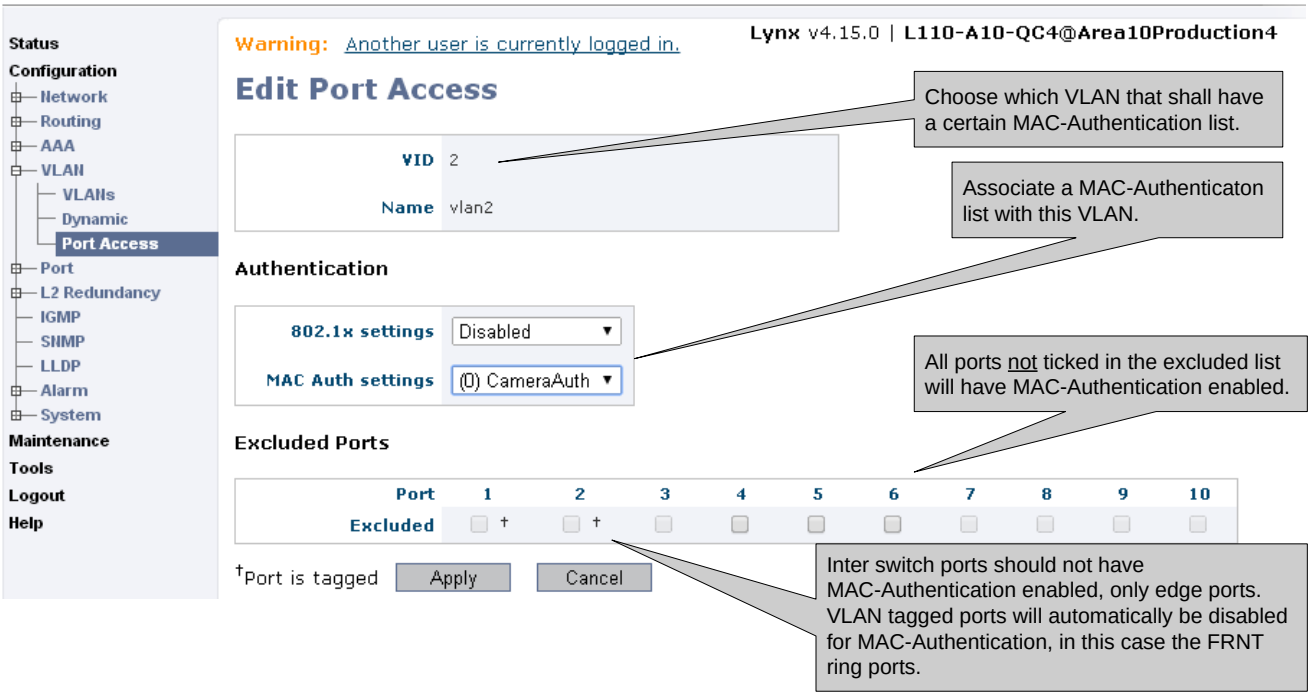


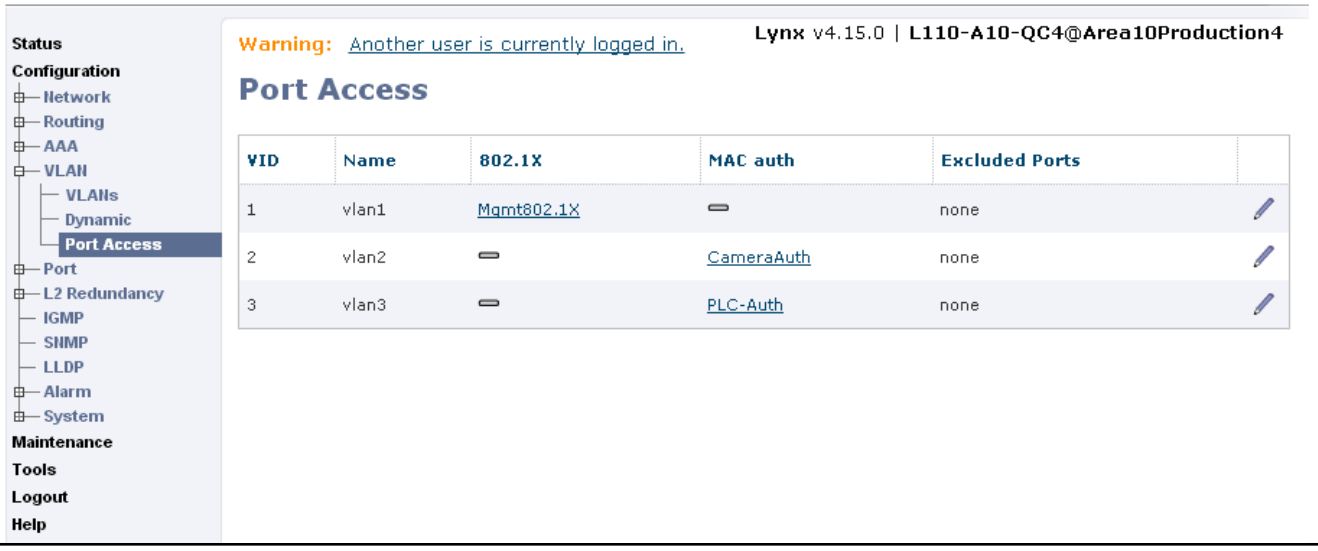
When all lists needed are created they must be associated with a VLAN.



2. Associate the local databases with the appropriate VLANs

Go to **Configuration** -> **VLAN** -> **Port Access** to associate MAC-Authentication lists with their appropriate VLANs.





Revision history for version 1.0

Revision	Rev by	Revision note	Date
00	ML	First version	140828
01			
02			
03			
04			
05			
06			
07			

HEAD OFFICE

Sweden
Westermo
SE-640 40 Stora Sundby
Tel: +46 (0)16 42 80 00
Fax: +46 (0)16 42 80 01
info@westermo.se
www.westermo.com

Sales Units

Westermo Data Communications

China
sales.cn@westermo.com
www.cn.westermo.com

France
infos@westermo.fr
www.westermo.fr

Germany
info@westermo.de
www.westermo.de

North America
info@westermo.com
www.westermo.com

Singapore
sales@westermo.com.sg
www.westermo.com

Sweden
info.sverige@westermo.se
www.westermo.se

United Kingdom
sales@westermo.co.uk
www.westermo.co.uk

Other Offices

For complete contact information, please visit our website at www.westermo.com/contact or scan the QR code with your mobile phone.