University of Piraeus

Department of Digital Systems

Post graduate Program in Digital Systems Security

Master Thesis

Installing and Configuring Security Mechanisms
Freeradius-MySQL
Freeradius-LDAP
PAM/USB Modules
LinOTP

Niskopoulos Nikolaos

Student No: MTE 1058

Supervising Professor: Xenakis Christos

Piraeus

March 2012

This thesis is dedicated to my parents who supported all my choices and efforts

Acknowledgements

My sincere thanks to Assistant Professor Christos Xenakis and the Postdoctoral researcher Christoforos Ntantogian, my project supervisors, for their valuable input, guidance and support.

Thanks to all the lecturers and students at the Postgraduate Program for sharing their knowledge.

Special thanks to my family and friends for their patience, ongoing support and encouragement throughout the years.

Abstract

This master thesis is a compilation of instructions- “how to” guides in order to install and configure security mechanisms that are of crucial importance considering the numerous threats any system administrator has to confront during his daily obligations.

The first security mechanism that has been installed and configured was Freeradius combined with MySQL database in Ubuntu 11.10 operating system. The freeradius was installed to a local network and wan configured to authenticate users stored to the SQL database via PAP (Password Authentication Protocol) and EAP-TLS (Extended Authentication Protocol-Transport Layer Security) by using certificates created using the free OpenSSL tool.

The second security mechanism was again Freeradius only this time was supported by an LDAP database in Ubuntu 11.10 operating system. The Lightweight Directory Access Protocol (LDAP) is an open standard for accessing directory services, X.500. The protocol runs over transport layer (OSI) where in the internet case is TCP/IP. The directory service is a database that organizes records and improves the procedures of accessing and searching. In this thesis we configure the LDAP database via the web user interface **phpldapandmin** and **JXplorer** tool in order to create groups and users who authenticate to our network through freeradius.

Furthermore a PAM/USB module was installed in order to authenticate to the Operating System (which was once again Ubuntu 11.10) via USB flash drive without the need of a password. Two factor authentication was also implemented to the OS by using something we possess (USB drive) and something we know (Password).

Finally the LinOTP2 tool was installed and configured in order to authenticate to our operating system via One Time Password. The LinOTP server and an application to our mobile which generates the one time passwords are used to achieve the OTP authentication. Additionally we implement two-factor authentication by using a combined password of something that we know and something that we generate real time. The last configuration is a triple factor authentication using the previous two factors (password and phone OTP) in addition to the PAM/USB module that was installed in the third part of this Thesis.

Contents

Picture Table

Freeradius Using MySQL on Linux Ubuntu 11.04

Introduction

FreeRADIUS is a modular, high performance free RADIUS suite developed and distributed under the GNU General Public License, and is free for download and use. The FreeRADIUS Suite includes a RADIUS server, a BSD-licensed RADIUS client library, a PAM library, an Apache module, and numerous additional RADIUS related utilities and development libraries. In most cases, the word "FreeRADIUS" refers to the free open source RADIUS server from this suite.

FreeRADIUS is the most popular open source RADIUS server and the most widely deployed RADIUS server in the world. It supports all common authentication protocols, and the server comes with a PHP-based web user administration tool.

Installing FreeRadius Server

We start by installing Freeradius on Ubuntu and do some basic testings:

1) Update our apt-get tool

sudo apt-get update

2) Install the tasksel tool in order to install lamp-server later on

apt-get install tasksel

3) Install lamp-server

Installing MySQL and PHP are all we need but to make sure that there are no dependencies errors later, we will install the whole package (lamp-server stands for Linux-Apache-MySQL-PHP server).

sudo tasksel install lamp-server

(you will need to enter root password, which is “setupRADIUS” by default, to continue the installation)

4) Install phpmyadmin

We run the following command:

sudo apt-get install phpmyadmin

Once phpMyAdmin is installed point your browser to http://localhost/phpmyadmin to start using it. You should be able to login using any users you've setup in MySQL during install. If no users have been setup, use *admin* with no password to login.

Should you get a 404 "Not Found" error when you point your browser to the location of phpMyAdmin (such as: http://localhost/phpmyadmin) this is likely caused by not checking the 'Apache 2' selection during installation.

To redo the installation run the following:

sudo dpkg-reconfigure -plow phpmyadmin

Then select Apache 2 for the webserver you wish to configure. (**we select the apache 2 using space bar not enter!** a red dot will appear when apache is selected)

If this does not work, then you can do the following to include the phpMyadmin-shipped Apache configuration into Apache:

sudo ln -s /etc/phpmyadmin/apache.conf /etc/apache2/conf.d/phpmyadmin.conf

sudo /etc/init.d/apache2 reload

5) Install freeradius package

sudo apt-get install freeradius

(Enter password when asked. When you are asked to continue, type 'y', then press Enter)

6) Install freeradius with ldap authentication

sudo apt-get install freeradius-ldap

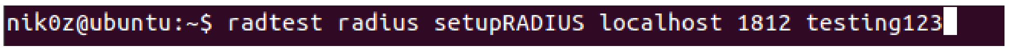
7) Install freeradius to run with mysql

sudo apt-get install freeradius-mysql

8) After finishing the above installations, restart the FreeRADIUS service

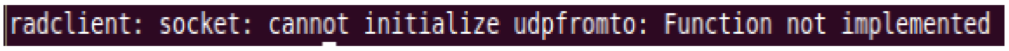
sudo /etc/init.d/freeradius restart

9) Now, you can test the Radius Server using radtest, the command will be as below:



```
nik0z@ubuntu:~$ radtest radius setupRADIUS localhost 1812 testing123
```

At this stage, you might get an error as follow:

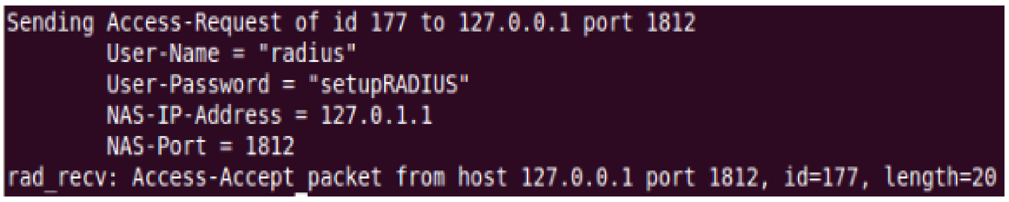


```
radclient: socket: cannot initialize udpfromto: Function not implemented
```

If that's the case, in order to fix the error, you open /etc/hosts as root, then comment out the line below:

::1 localhost ip6-localhost ip6-loopback

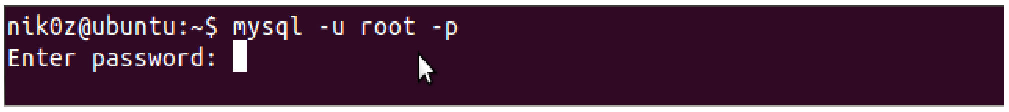
The correct result should be as follow:



```
Sending Access-Request of id 177 to 127.0.0.1 port 1812
        User-Name = "radius"
        User-Password = "setupRADIUS"
        NAS-IP-Address = 127.0.1.1
        NAS-Port = 1812
rad_recv: Access-Accept packet from host 127.0.0.1 port 1812, id=177, length=20
```

10) Now, we create the database for FreeRADIUS and a user account which will be used by FreeRADIUS to access into database. We run the following command:

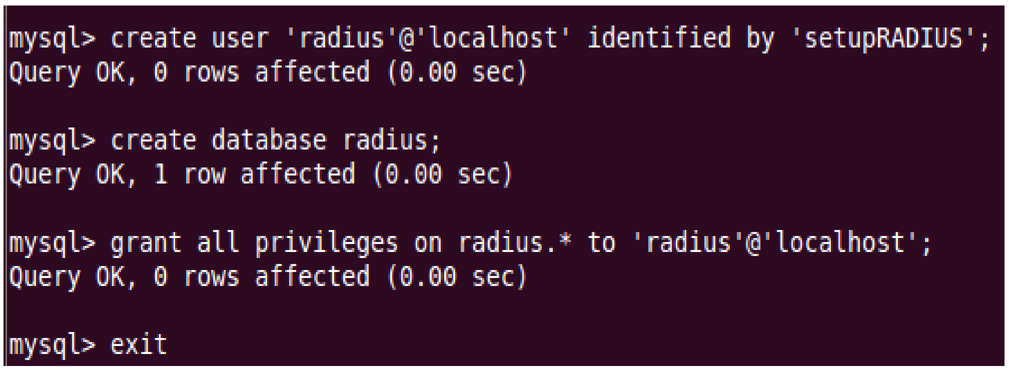
- Login MySQL Database



```
nik0z@ubuntu:~$ mysql -u root -p
Enter password:
```

(Enter the password for root user)

- Then you can create user, database, and set permissions as follow:



```
mysql> create user 'radius'@'localhost' identified by 'setupRADIUS';
Query OK, 0 rows affected (0.00 sec)

mysql> create database radius;
Query OK, 1 row affected (0.00 sec)

mysql> grant all privileges on radius.* to 'radius'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> exit
```

11) Then, we are going to create our database using phpmyadmin

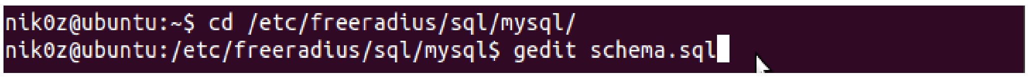
We open a terminal and run the following commands:

cd /etc/freeradius/sql/mysql/

in order to go to the proper directory

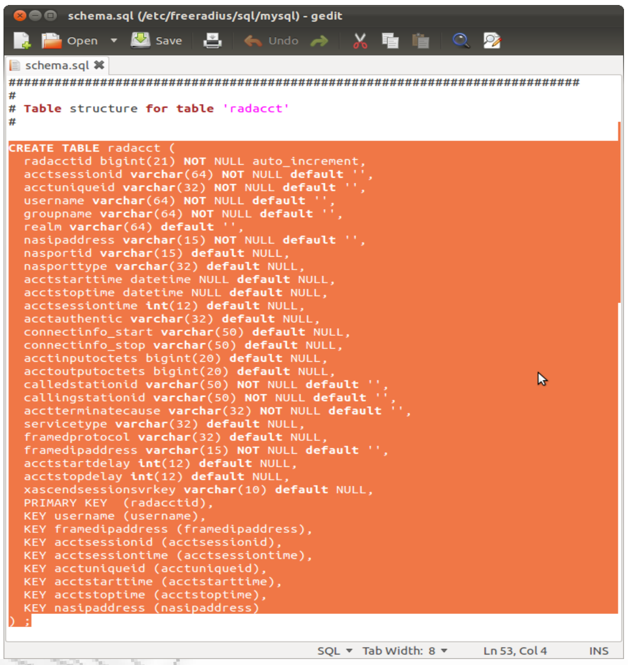
gedit schema.sql

with this command the schema.sql file will open with the gedit text editor



```
nik0z@ubuntu:~$ cd /etc/freeradius/sql/mysql/
nik0z@ubuntu:/etc/freeradius/sql/mysql$ gedit schema.sql
```

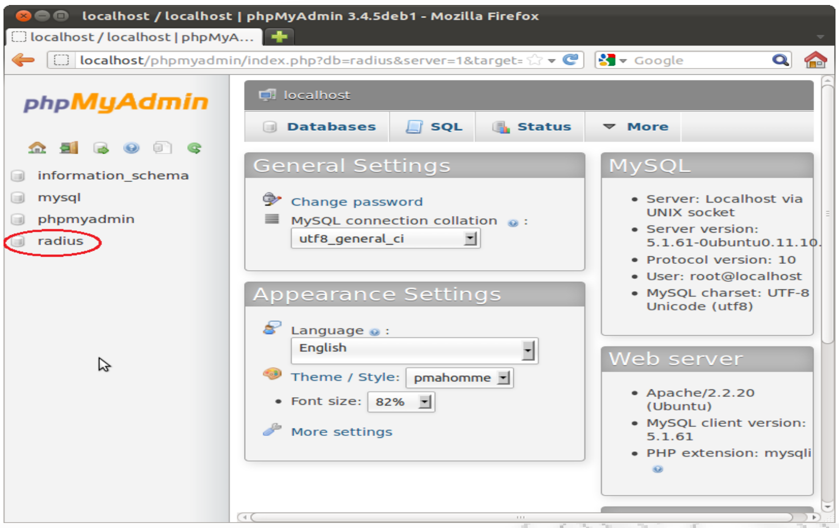
we select the radacct table like in the screenshot that follows and then right click copy'.



```
###########################################################################
#
# Table structure for table 'radacct'
#

CREATE TABLE radacct (
  radacctid bigint(21) NOT NULL auto_increment,
  acctsessionid varchar(64) NOT NULL default '',
  acctuniqueid varchar(32) NOT NULL default '',
  username varchar(64) NOT NULL default '',
  groupname varchar(64) NOT NULL default '',
  realm varchar(64) default '',
  nasipaddress varchar(15) NOT NULL default '',
  nasportid varchar(15) default NULL,
  nasporttype varchar(32) default NULL,
  acctstarttime datetime NULL default NULL,
  acctstoptime datetime NULL default NULL,
  acctsessiontime int(12) default NULL,
  acctauthentic varchar(32) default NULL,
  connectinfo_start varchar(50) default NULL,
  connectinfo_stop varchar(50) default NULL,
  acctinputoctets bigint(20) default NULL,
  acctoutputoctets bigint(20) default NULL,
  calledstationid varchar(50) NOT NULL default '',
  callingstationid varchar(50) NOT NULL default '',
  acctterminatecause varchar(32) NOT NULL default '',
  servicetype varchar(32) default NULL,
  framedprotocol varchar(32) default NULL,
  framedipaddress varchar(15) NOT NULL default '',
  acctstartdelay int(12) default NULL,
  acctstopdelay int(12) default NULL,
  xascendsessionsvrkey varchar(10) default NULL,
  PRIMARY KEY  (radacctid),
  KEY username (username),
  KEY framedipaddress (framedipaddress),
  KEY acctsessionid (acctsessionid),
  KEY acctsessiontime (acctsessiontime),
  KEY acctuniqueid (acctuniqueid),
  KEY acctstarttime (acctstarttime),
  KEY acctstoptime (acctstoptime),
  KEY nasipaddress (nasipaddress)
) ;
```

Picture 1:radacct table

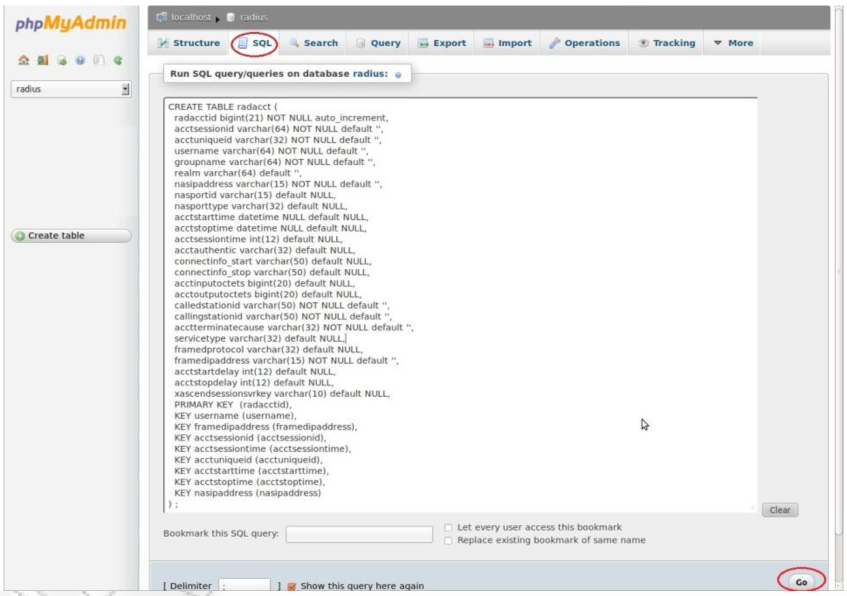
We open a browser and we point it to http://localhost/phpmyadmin we use the username and password we chose during the installation. On the left column we should see the radius database we created on step 10 and we click on that.



```
localhost / localhost | phpMyAdmin 3.4.5deb1 - Mozilla Firefox
localhost/phpmyadmin/index.php?db=radius&server=1&target=

General Settings
  Change password
  MySQL connection collation: utf8_general_ci

Appearance Settings
  Language: English
  Theme / Style: pmahomme
  Font size: 82%
  More settings

MySQL
  • Server: Localhost via UNIX socket
  • Server version: 5.1.61-0ubuntu0.11.10
  • Protocol version: 10
  • User: root@localhost
  • MySQL charset: UTF-8 Unicode (utf8)

Web server
  • Apache/2.2.20 (Ubuntu)
  • MySQL client version: 5.1.61
  • PHP extension: mysqli
```

Picture 1: http://localhost/phpmyadmin

We select SQL like in the pic that follows, we paste the table from schema.sql file and hit go. We repeat this procedure for all the following tables in the schema.sql file (radcheck, radgroupcheck, radgroupreply, radpostauth, radreply, radusergroup).



Run SQL query/queries on database radius:

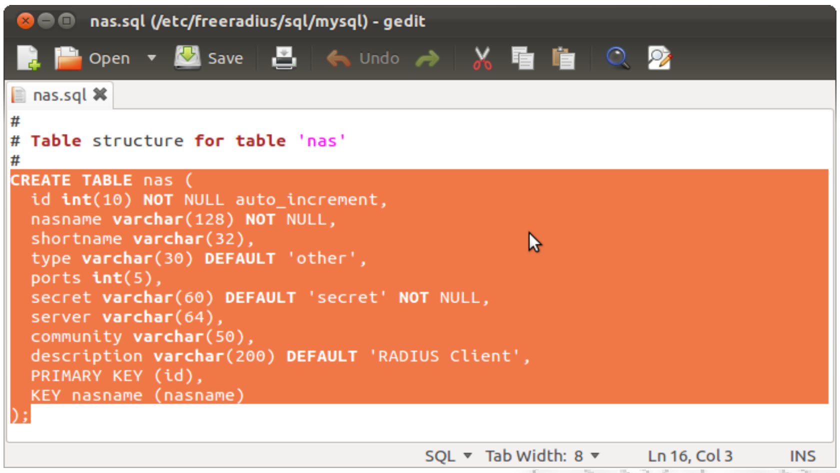
```
CREATE TABLE radacct (
  radacctid bigint(21) NOT NULL auto_increment,
  acctsessionid varchar(64) NOT NULL default '',
  acctuniqueid varchar(32) NOT NULL default '',
  username varchar(64) NOT NULL default '',
  groupname varchar(64) NOT NULL default '',
  realm varchar(64) default '',
  nasipaddress varchar(15) NOT NULL default '',
  nasportid varchar(15) default NULL,
  nasporttype varchar(32) default NULL,
  acctstarttime datetime NULL default NULL,
  acctstoptime datetime NULL default NULL,
  acctsessiontime int(12) default NULL,
  acctauthentic varchar(32) default NULL,
  connectinfo_start varchar(50) default NULL,
  connectinfo_stop varchar(50) default NULL,
  acctinputoctets bigint(20) default NULL,
  acctoutputoctets bigint(20) default NULL,
  calledstationid varchar(50) NOT NULL default '',
  callingstationid varchar(50) NOT NULL default '',
  acctterminatecause varchar(32) NOT NULL default '',
  servicetype varchar(32) default NULL,
  framedprotocol varchar(32) default NULL,
  framedipaddress varchar(15) NOT NULL default '',
  acctstartdelay int(12) default NULL,
  acctstopdelay int(12) default NULL,
  xascendsessionsvrkey varchar(10) default NULL,
  PRIMARY KEY  (radacctid),
  KEY username (username),
  KEY framedipaddress (framedipaddress),
  KEY acctsessionid (acctsessionid),
  KEY acctsessiontime (acctsessiontime),
  KEY acctuniqueid (acctuniqueid),
  KEY acctstarttime (acctstarttime),
  KEY acctstoptime (acctstoptime),
  KEY nasipaddress (nasipaddress)
) ;
```

Picture 2: Creating the tables in SQL tab of phpMyAdmin

Then we return to our terminal and run:

gedit nas.sql

in order to copy the nas table to our radius database as we did with the previous tables.



```
#
# Table structure for table 'nas'
#
CREATE TABLE nas (
  id int(10) NOT NULL auto_increment,
  nasname varchar(128) NOT NULL,
  shortname varchar(32),
  type varchar(30) DEFAULT 'other',
  ports int(5),
  secret varchar(60) DEFAULT 'secret' NOT NULL,
  server varchar(64),
  community varchar(50),
  description varchar(200) DEFAULT 'RADIUS Client',
  PRIMARY KEY (id),
  KEY nasname (nasname)
);
```

Picture 3: NAS table

Finally our radius database should look like the picture that follows

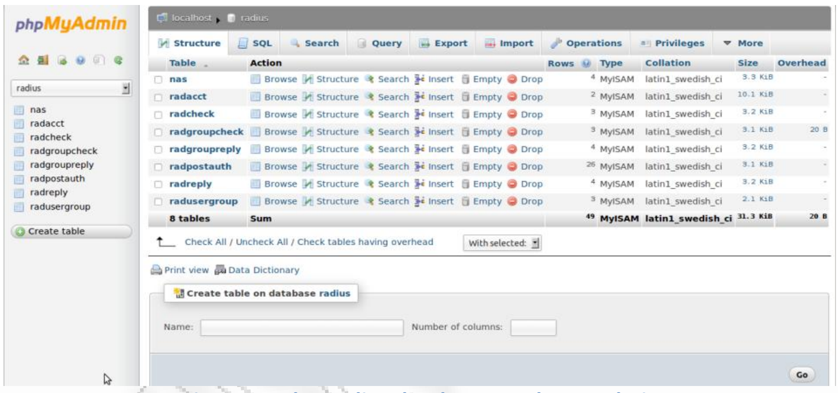


Table	Action	Rows	Type	Collation	Size	Overhead
nas	Browse Structure Search Insert Empty Drop	4	MyISAM	latin1_swedish_ci	3.3 KiB	-
radacct	Browse Structure Search Insert Empty Drop	2	MyISAM	latin1_swedish_ci	10.1 KiB	-
radcheck	Browse Structure Search Insert Empty Drop	3	MyISAM	latin1_swedish_ci	3.2 KiB	-
radgroupcheck	Browse Structure Search Insert Empty Drop	3	MyISAM	latin1_swedish_ci	3.1 KiB	20 B
radgroupreply	Browse Structure Search Insert Empty Drop	4	MyISAM	latin1_swedish_ci	3.2 KiB	-
radpostauth	Browse Structure Search Insert Empty Drop	26	MyISAM	latin1_swedish_ci	3.1 KiB	-
radreply	Browse Structure Search Insert Empty Drop	4	MyISAM	latin1_swedish_ci	3.2 KiB	-
radusergroup	Browse Structure Search Insert Empty Drop	3	MyISAM	latin1_swedish_ci	2.1 KiB	-
8 tables	Sum	49	MyISAM	latin1_swedish_ci	31.3 KiB	20 B

phpMyAdmin
localhost radius
Structure SQL Search Query Export Import Operations Privileges More
radius
nas
radacct
radcheck
radgroupcheck
radgroupreply
radpostauth
radreply
radusergroup
Create table
Check All / Uncheck All / Check tables having overhead
With selected:
Print view Data Dictionary
Create table on database radius
Name:
Number of columns:
Go

Picture 4: The Radius database on phpMyAdmin

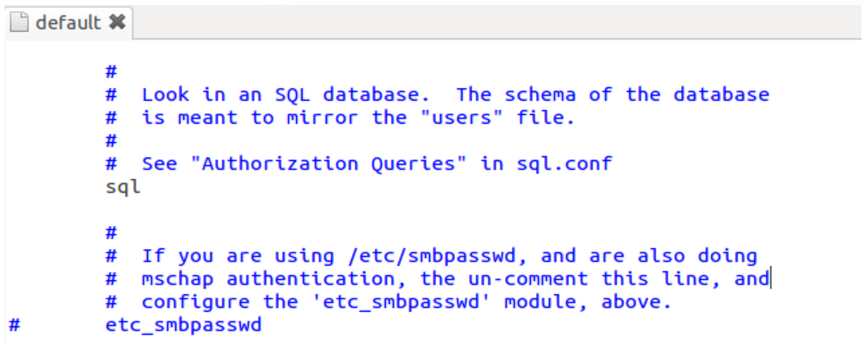
12) Configure the file /etc/freeradius/sites-available/default so that the FreeRADIUS can startup, creates a default virtual host, and connects to MySQL Database. We follow the below steps:

Run the command:

sudo gedit /etc/freeradius/sites-enabled/default

In the file “default”, you check if the following settings are correct or not (the settings in the belowpictures are correct settings):

In the authorize{} module

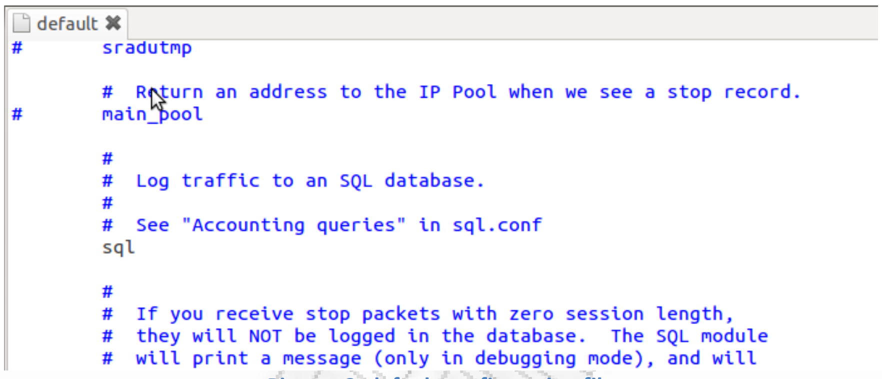


```
default ✖
        #
        #  Look in an SQL database.  The schema of the database
        #  is meant to mirror the "users" file.
        #
        #  See "Authorization Queries" in sql.conf
        sql

        #
        #  If you are using /etc/smbpasswd, and are also doing
        #  mschap authentication, the un-comment this line, and
        #  configure the 'etc_smbpasswd' module, above.
#       etc_smbpasswd
```

Picture 5: default configuration file

(uncomment the sql setting)

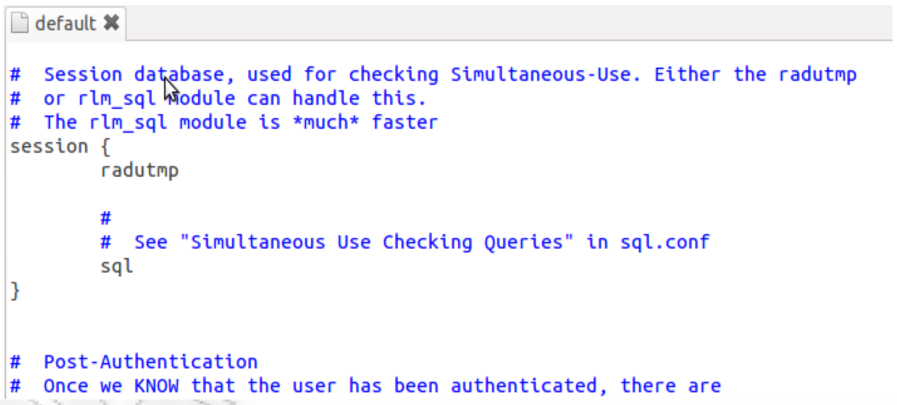
In the accounting{} module



```
default ✖
#       sradutmp

        #  Return an address to the IP Pool when we see a stop record.
#       main_pool

        #
        #  Log traffic to an SQL database.
        #
        #  See "Accounting queries" in sql.conf
        sql

        #
        #  If you receive stop packets with zero session length,
        #  they will NOT be logged in the database.  The SQL module
        #  will print a message (only in debugging mode), and will
```

Picture 6: default configuration file

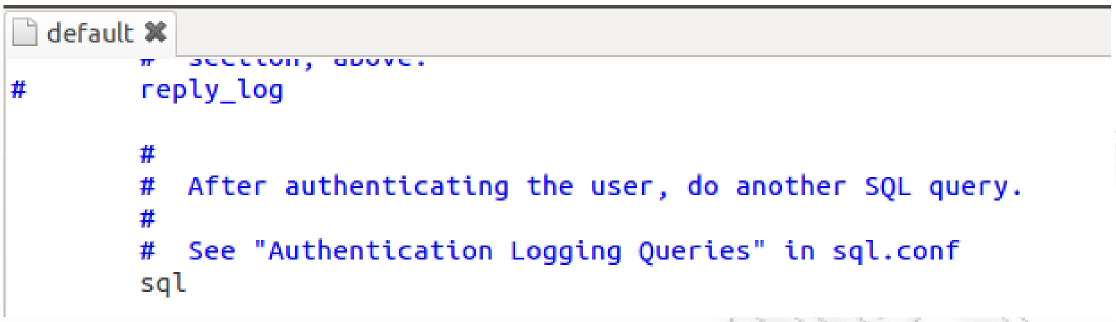
In the session{} module



```
default ✖

#  Session database, used for checking Simultaneous-Use. Either the radutmp
#  or rlm_sql module can handle this.
#  The rlm_sql module is *much* faster
session {
        radutmp

        #
        #  See "Simultaneous Use Checking Queries" in sql.conf
        sql
}


#  Post-Authentication
#  Once we KNOW that the user has been authenticated, there are
```

Picture 7: default configuration file

In the post-auth{} module


```
#       reply_log

        #
        #  After authenticating the user, do another SQL query.
        #
        #  See "Authentication Logging Queries" in sql.conf
        sql
```

Picture 8: default configuration file

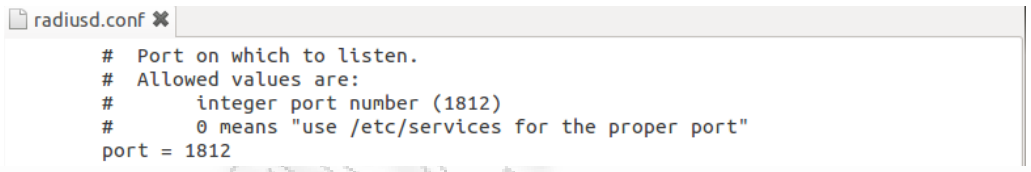
We always save each file before we exit.

13) **Now, we configure the "radiusd.conf". This file contains general settings for the Radius Server. We follow the below steps:**

sudo gedit /etc/freeradius/radiusd.conf

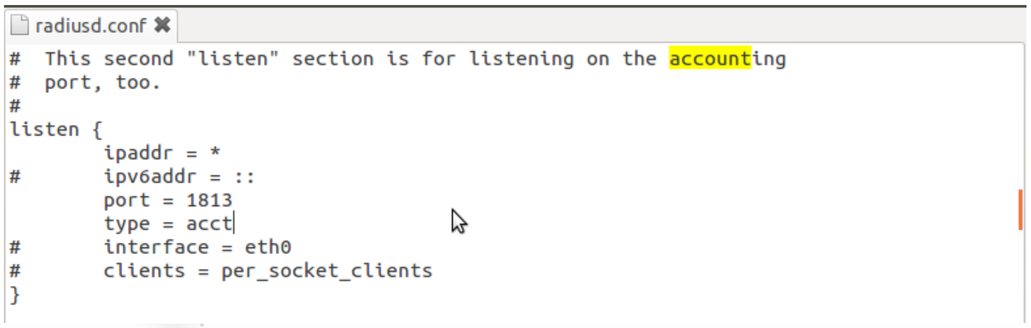
Then, we check if the settings in the file radiusd.conf are correct or not.

The port for **radius server** to listen for authentication request is 1812


```
        #  Port on which to listen.
        #  Allowed values are:
        #       integer port number (1812)
        #       0 means "use /etc/services for the proper port"
        port = 1812
```

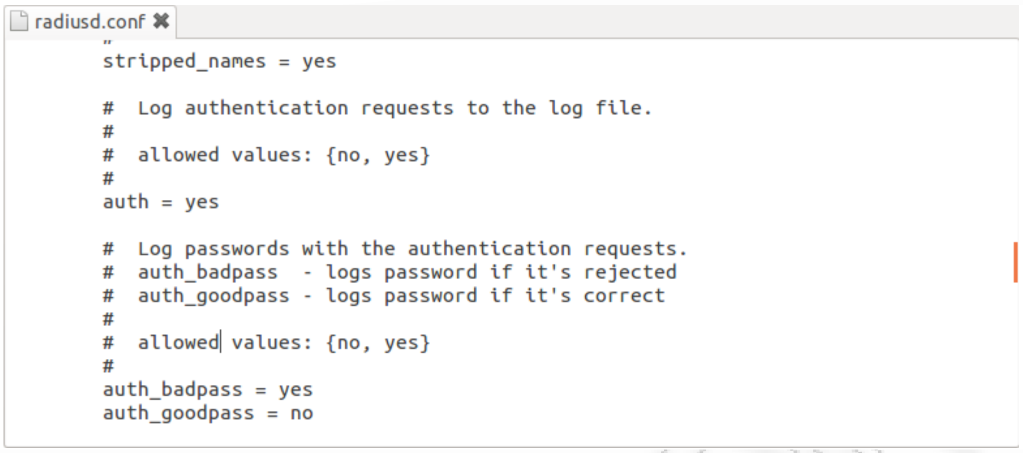
Picture 9: radius.conf configuration file

Port for accounting is 1813


```
#  This second "listen" section is for listening on the accounting
#  port, too.
#
listen {
        ipaddr = *
#       ipv6addr = ::
        port = 1813
        type = acct
#       interface = eth0
#       clients = per_socket_clients
}
```

Picture 10: radius.conf configuration file

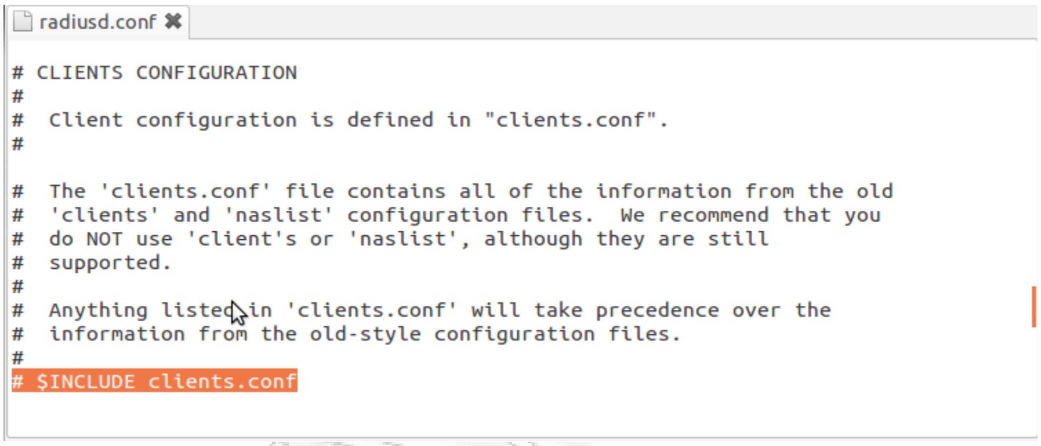
Some settings for logging username, password, etc



```
radiusd.conf
	stripped_names = yes

	#  Log authentication requests to the log file.
	#
	#  allowed values: {no, yes}
	#
	auth = yes

	#  Log passwords with the authentication requests.
	#  auth_badpass  - logs password if it's rejected
	#  auth_goodpass - logs password if it's correct
	#
	#  allowed values: {no, yes}
	#
	auth_badpass = yes
	auth_goodpass = no
```

Picture 11: radius.conf configuration file

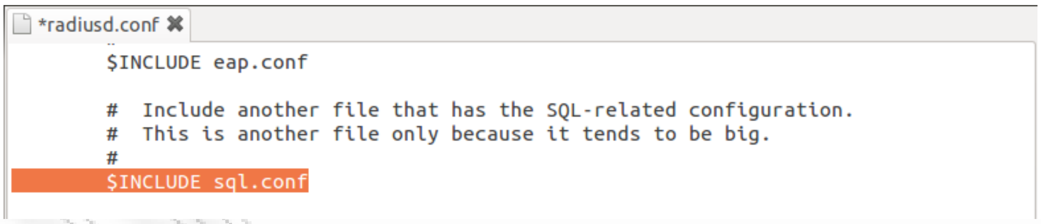
Then we make sure that freeradius will not read clients from the clients.conf file



```
radiusd.conf
# CLIENTS CONFIGURATION
#
#  Client configuration is defined in "clients.conf".
#

#  The 'clients.conf' file contains all of the information from the old
#  'clients' and 'naslist' configuration files.  We recommend that you
#  do NOT use 'client's or 'naslist', although they are still
#  supported.
#
#  Anything listed in 'clients.conf' will take precedence over the
#  information from the old-style configuration files.
#
# $INCLUDE clients.conf
```

Picture 12: radius.conf configuration file

And also that the sql.conf file will be included



```
*radiusd.conf
	$INCLUDE eap.conf

	#  Include another file that has the SQL-related configuration.
	#  This is another file only because it tends to be big.
	#
	$INCLUDE sql.conf
```

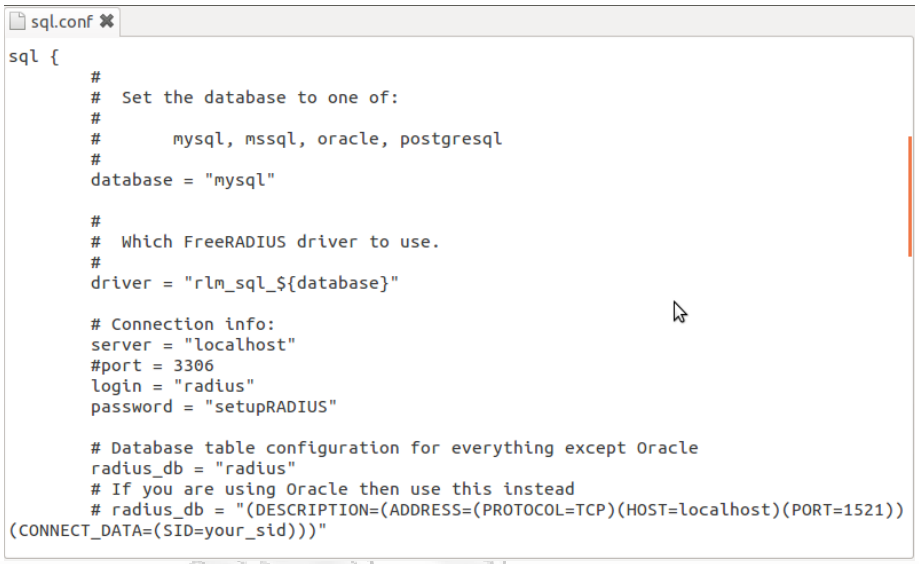
Picture 13: radius.conf configuration file

14) Now, we configure the file "sql.conf". This file contains information about how to connect to database, database tables that are used by the FreeRADIUS server, etc

Run the command:

sudo gedit /etc/freeradius/sql.conf

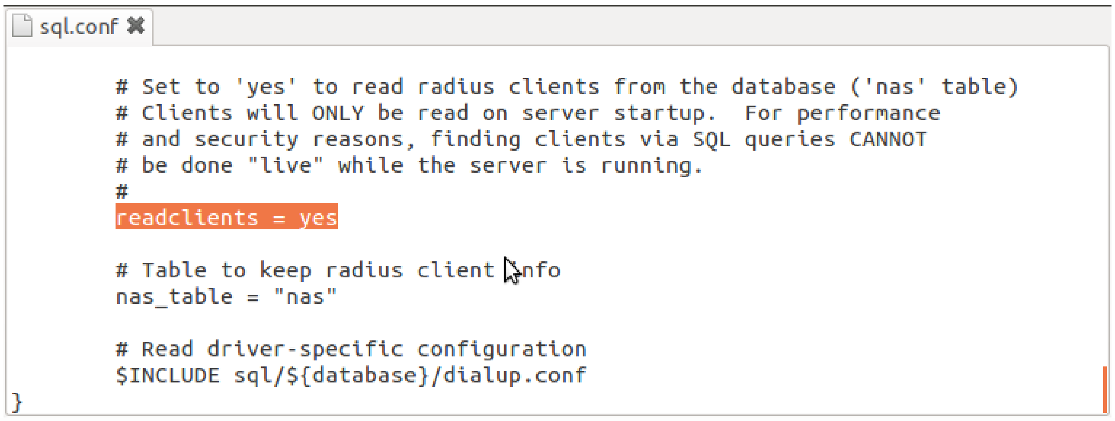
Then, you should check the following settings:



```
sql.conf ✖
sql {
        #
        #  Set the database to one of:
        #
        #       mysql, mssql, oracle, postgresql
        #
        database = "mysql"

        #
        #  Which FreeRADIUS driver to use.
        #
        driver = "rlm_sql_${database}"

        # Connection info:
        server = "localhost"
        #port = 3306
        login = "radius"
        password = "setupRADIUS"

        # Database table configuration for everything except Oracle
        radius_db = "radius"
        # If you are using Oracle then use this instead
        # radius_db = "(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=localhost)(PORT=1521))
(CONNECT_DATA=(SID=your_sid)))"
```

Picture 14: sql.conf configuration file

Check the setting that allow FreeRADIUS to read Radius Clients from database



```
sql.conf ✖

        # Set to 'yes' to read radius clients from the database ('nas' table)
        # Clients will ONLY be read on server startup.  For performance
        # and security reasons, finding clients via SQL queries CANNOT
        # be done "live" while the server is running.
        #
        readclients = yes

        # Table to keep radius client info
        nas_table = "nas"

        # Read driver-specific configuration
        $INCLUDE sql/${database}/dialup.conf
}
```

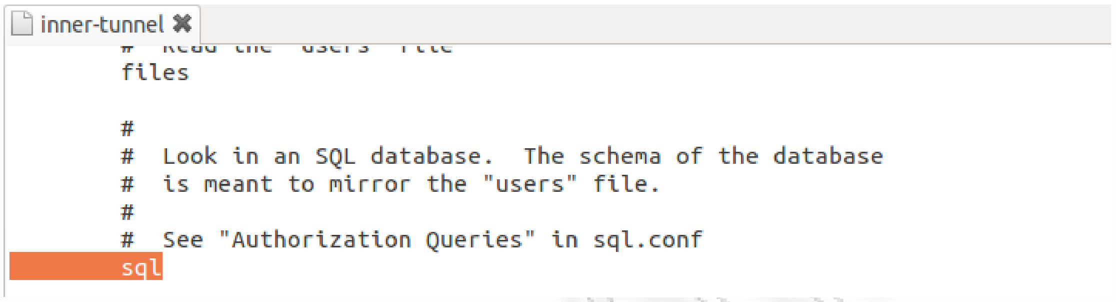
Picture 15: sql.conf configuration file

15) We configure inner tunnel

Run the command

sudo gedit

we remove '#' before sql in the authorize{} module



```
inner-tunnel
        #  Read the users file
        files

        #
        #  Look in an SQL database.  The schema of the database
        #  is meant to mirror the "users" file.
        #
        #  See "Authorization Queries" in sql.conf
        sql
```

Picture 16: inner-tunnel configuration file

16) Test if the Freeradius Server is working properly

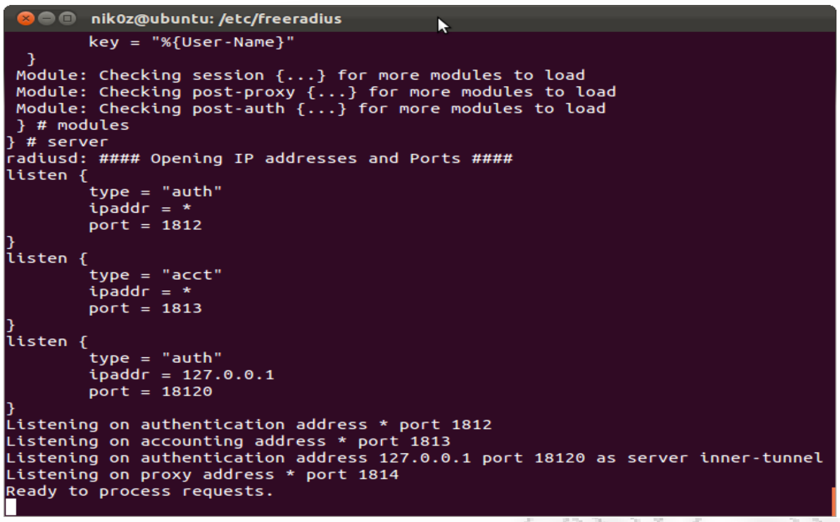
Run the commands:

/etc/init.d/freeradius stop

```
 * Stopping FreeRADIUS daemon freeradius
 * /var/run/freeradius/freeradius.pid not found...                    [ OK ]
```

freeradius -X This command runs freeradius in debug mode and it is very helpful if we want to find out what went wrong.

If everything went well so far, the following picture is the expected response.



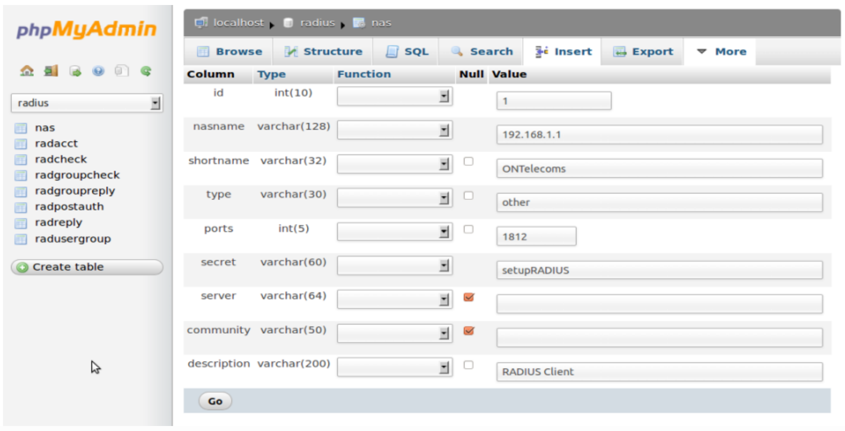
```
nik0z@ubuntu: /etc/freeradius
        key = "%{User-Name}"
 }
 Module: Checking session {...} for more modules to load
 Module: Checking post-proxy {...} for more modules to load
 Module: Checking post-auth {...} for more modules to load
 } # modules
} # server
radiusd: #### Opening IP addresses and Ports ####
listen {
        type = "auth"
        ipaddr = *
        port = 1812
}
listen {
        type = "acct"
        ipaddr = *
        port = 1813
}
listen {
        type = "auth"
        ipaddr = 127.0.0.1
        port = 18120
}
Listening on authentication address * port 1812
Listening on accounting address * port 1813
Listening on authentication address 127.0.0.1 port 18120 as server inner-tunnel
Listening on proxy address * port 1814
Ready to process requests.
```

Picture 17: The radius server is up and running waiting for requests

Configuring the Freeradius Server

We begin by creating our clients (our Access Points)
We open our browser and point it to http://localhost/phpmyadmin we select the radius database we have already created and we click on the 'nas' table we select 'Insert' and we fill the blanks as in the following picture.

id	: it will be filled automatically
nasname	: the ip-address of our access point
shortname	: the ssid of our access point
type	: other
ports	: 1812
secret	: anything you want (we will use it when we will configure our access point)
description	: RADIUS Client



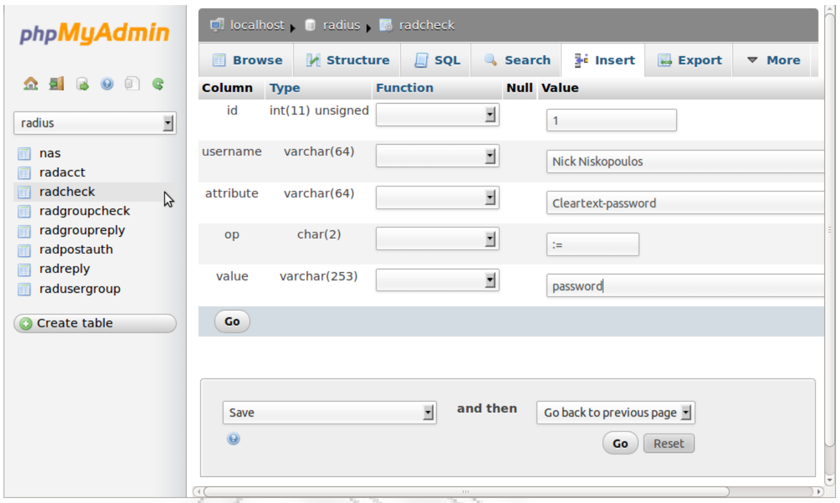
Column	Type	Function	Null	Value
id	int(10)			1
nasname	varchar(128)			192.168.1.1
shortname	varchar(32)		☐	ONTelecoms
type	varchar(30)		☐	other
ports	int(5)		☐	1812
secret	varchar(60)			setupRADIUS
server	varchar(64)		✓	
community	varchar(50)		✓	
description	varchar(200)		☐	RADIUS Client

Picture 18: Creating the radius clinets on NAS

Then we will have to create our users. We select the radcheck table and we fill the blanks as in the following picture.

username	: the username of our user
attribute	: Cleartext-password
op	: : = (this symbol means that any value that has already potentially been installed as Cleartext-password for our user will be overwrite by the new one. The simple equation (=) symbol wouldn't overwrite any previous value.)
value	: our password

and click go to insert the new user.



Column	Type	Function	Null	Value
id	int(11) unsigned			1
username	varchar(64)			Nick Niskopoulos
attribute	varchar(64)			Cleartext-password
op	char(2)			:=
value	varchar(253)			password

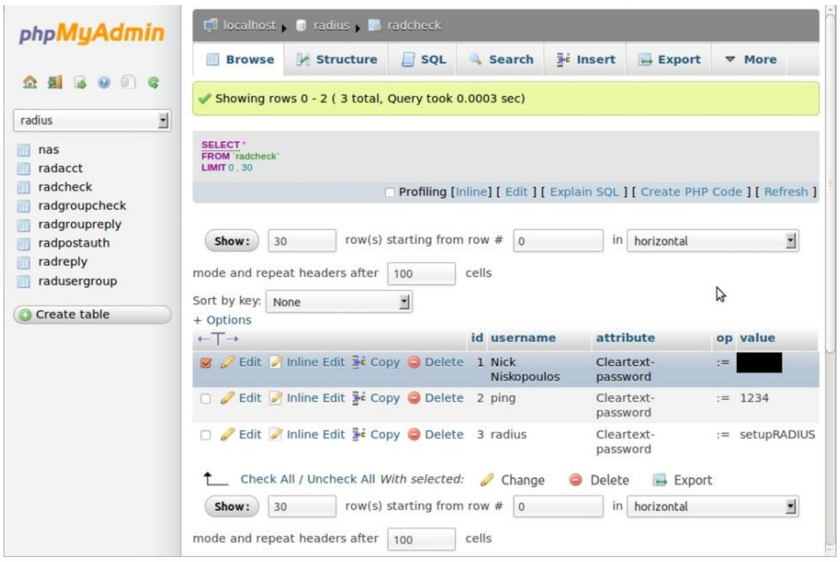
Go

Save and then Go back to previous page

Go Reset

Picture 19: creating the users of our radius database

As we can see here we have installed three users to our freeradius-mysql server.



✓ Showing rows 0 - 2 (3 total, Query took 0.0003 sec)

```
SELECT *
FROM `radcheck`
LIMIT 0 , 30
```

Profiling [Inline] [Edit] [Explain SQL] [Create PHP Code] [Refresh]

Show: 30 row(s) starting from row # 0 in horizontal mode and repeat headers after 100 cells

Sort by key: None

+ Options

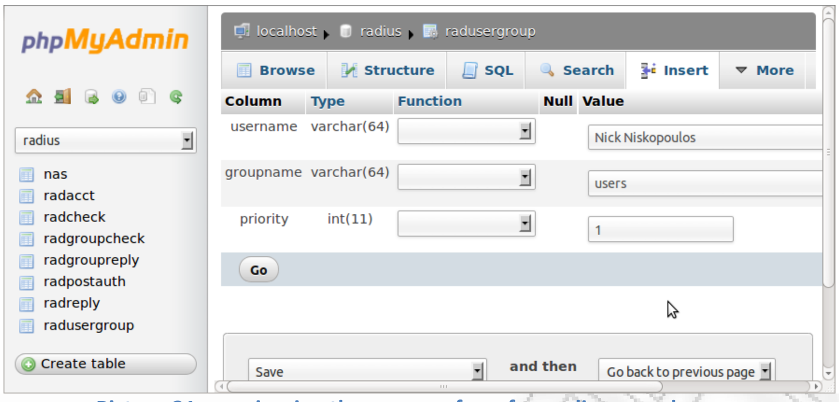
	id	username	attribute	op	value
Edit Inline Edit Copy Delete	1	Nick Niskopoulos	Cleartext-password	:=	[redacted]
Edit Inline Edit Copy Delete	2	ping	Cleartext-password	:=	1234
Edit Inline Edit Copy Delete	3	radius	Cleartext-password	:=	setupRADIUS

Check All / Uncheck All *With selected:* Change Delete Export

Show: 30 row(s) starting from row # 0 in horizontal mode and repeat headers after 100 cells

Picture 20: The users in our freeradius-mysql server

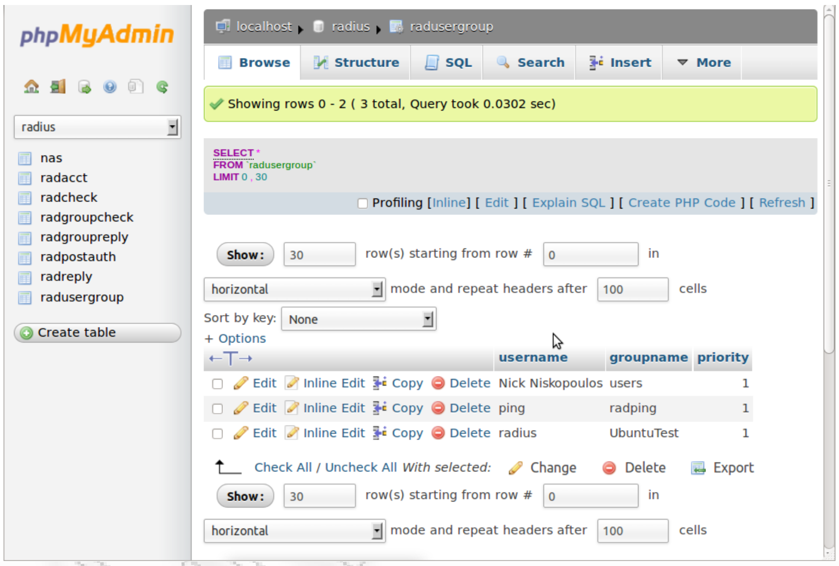
Then we configure our groups of users. We select the radusergroup and by using the insert option we simply define wich user belongs to what group.



Column	Type	Function	Null	Value
username	varchar(64)			Nick Niskopoulos
groupname	varchar(64)			users
priority	int(11)			1

Picture 21: congiguring the groups of our freeradius-mysql server

As we can see we can create unlimited combinations of users and groups very easily.



Showing rows 0 - 2 (3 total, Query took 0.0302 sec)

```
SELECT *
FROM `radusergroup`
LIMIT 0 , 30
```

Show: 30 row(s) starting from row # 0 in horizontal mode and repeat headers after 100 cells

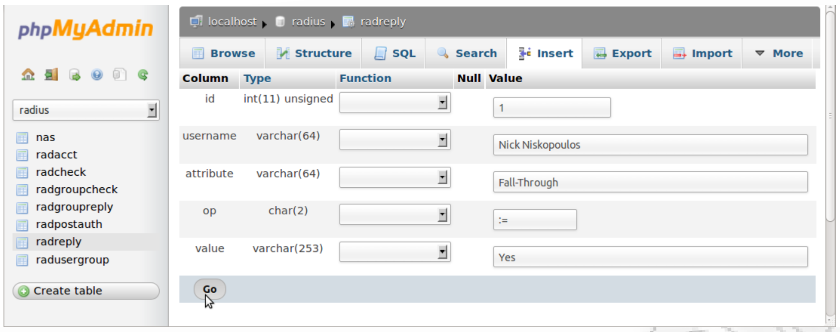
Sort by key: None

					username	groupname	priority
☐	Edit	Inline Edit	Copy	Delete	Nick Niskopoulos	users	1
☐	Edit	Inline Edit	Copy	Delete	ping	radping	1
☐	Edit	Inline Edit	Copy	Delete	radius	UbuntuTest	1

Picture 22: the existing groups of the freeradius server

We select the radreply table and then insert in order to configure our users.

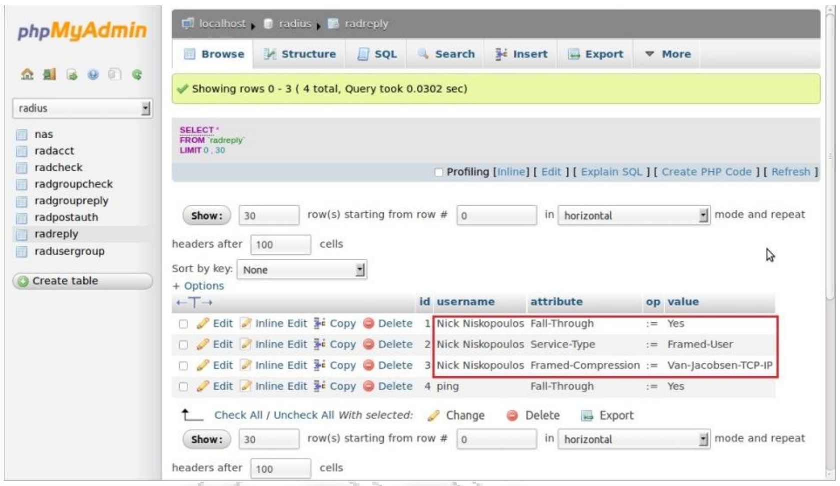
username	: we select the user we want
attribute	: we select what attribute we want to change
op	: we select our operation
value	: we give a value to our attribute



Column	Type	Function	Null	Value
id	int(11) unsigned			1
username	varchar(64)			Nick Niskopoulos
attribute	varchar(64)			Fall-Through
op	char(2)			:=
value	varchar(253)			Yes

Picture 23: configuring the user attributes

In the following picture we can see what adjustments we made for the user Nick Niskopoulos. This is the default configuration for each user.



Showing rows 0 - 3 (4 total, Query took 0.0302 sec)

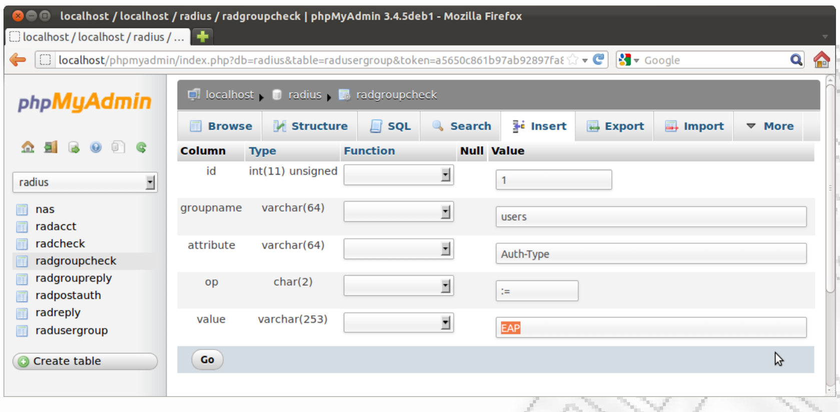
```
SELECT *
FROM `radreply`
LIMIT 0 , 30
```

id	username	attribute	op	value
1	Nick Niskopoulos	Fall-Through	:=	Yes
2	Nick Niskopoulos	Service-Type	:=	Framed-User
3	Nick Niskopoulos	Framed-Compression	:=	Van-Jacobsen-TCP-IP
4	ping	Fall-Through	:=	Yes

Picture 24: The configuration for user Nisk Niskopoulos

- All entries are processed in the order as they appear in the users file or in the radcheck table since we use MySQL database instead of the users file. If an entry matches the user name, radiusd will stop scanning the users file unless the attribute "Fall-Through = Yes" is set.
- Service-Type = Framed-User (with this rule only framed-user service types are allowed –no telnet, rlogin, tcp-clear-)
- Framed-Compression : = Van-jacobsen-TCP-IP . This framed compression attribute indicates a compression protocol to be used for the link. It may be used in Access-Accept packets. It may be used in an Access-Request packet as a hint to the server that the NAS would prefer to use that compression, but the server is not required to honor the hint. More than one compression protocol attribute may be sent. It is the responsibility of the NAS to apply the proper compression protocol to appropriate link traffic.

Following we select the radgroupcheck table and the 'Insert' tab and we set the authentication method for all users of the group 'users' to be EAP.

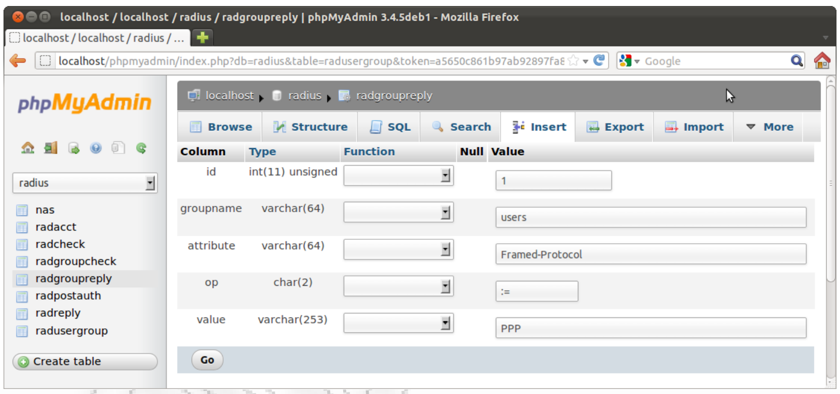

localhost / localhost / radius / radgroupcheck | phpMyAdmin 3.4.5deb1 - Mozilla Firefox

localhost/phpmyadmin/index.php?db=radius&table=radusergroup&token=a5650c861b97ab92897fa£

Column	Type	Function	Null	Value
id	int(11) unsigned			1
groupname	varchar(64)			users
attribute	varchar(64)			Auth-Type
op	char(2)			:=
value	varchar(253)			EAP

Picture 25: onfiguring the authentication method

Finaly we select the radgroupreply table to configure the Framed-Protocol for all users of the group 'users' to be PPP. This rule allows PPP sessions (no SLIP, CSLIP, etc.).


localhost / localhost / radius / radgroupreply | phpMyAdmin 3.4.5deb1 - Mozilla Firefox

localhost/phpmyadmin/index.php?db=radius&table=radusergroup&token=a5650c861b97ab92897fa£

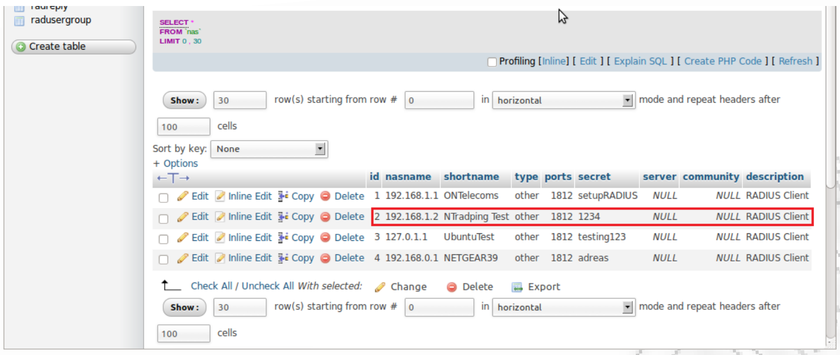
Column	Type	Function	Null	Value
id	int(11) unsigned			1
groupname	varchar(64)			users
attribute	varchar(64)			Framed-Protocol
op	char(2)			:=
value	varchar(253)			PPP

Picture 26: Configuring the Framed-Protocol the group users

Testing our configuration

This is the basic configuration for the Freeradius Server that works with EAP authentication methods. In order to test our installation we can download the NTRadPing 1.5 RADIUS Test Utility from here: http://www.novell.com/coolsolutions/tools/14377.html and make the following adjustments.

We create a new client 'NTradping Test' in the NAS table of our database. (We set as nasname the ip of the operating system that the test utility will run)

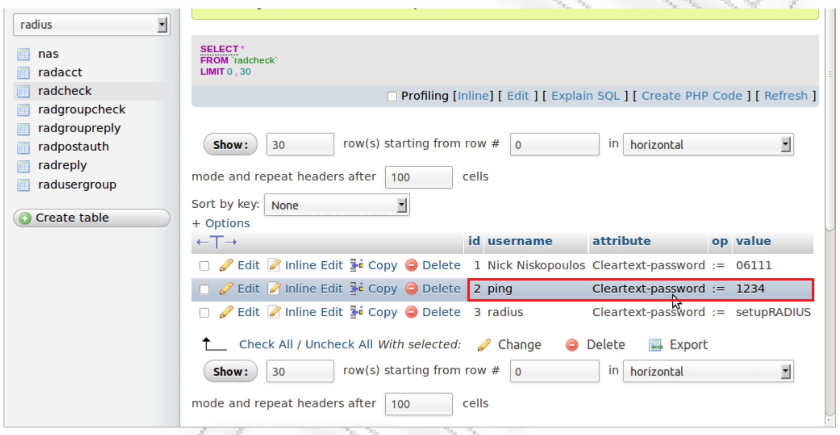


```
SELECT *
FROM `nas`
LIMIT 0 , 30
```

id	nasname	shortname	type	ports	secret	server	community	description
1	192.168.1.1	ONTelecoms	other	1812	setupRADIUS	NULL	NULL	RADIUS Client
2	192.168.1.2	NTradping Test	other	1812	1234	NULL	NULL	RADIUS Client
3	127.0.1.1	UbuntuTest	other	1812	testing123	NULL	NULL	RADIUS Client
4	192.168.0.1	NETGEAR39	other	1812	adreas	NULL	NULL	RADIUS Client

Picture 27: Creating Client NTRadping Test

We create a new user ‘ping’ with password 1234

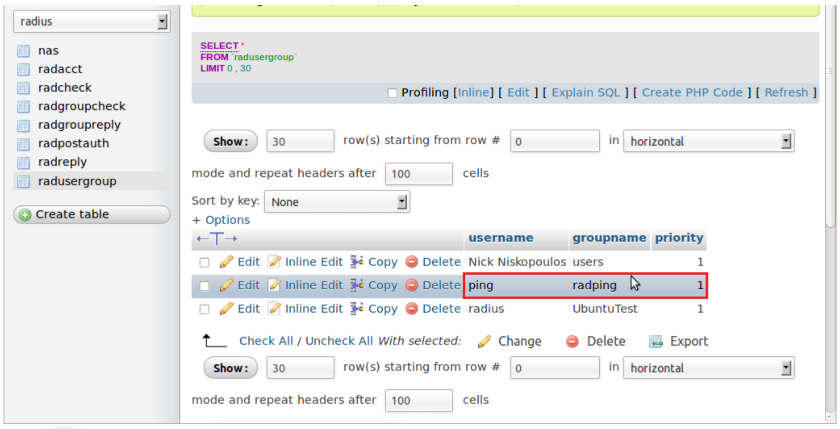


```
SELECT *
FROM `radcheck`
LIMIT 0 , 30
```

id	username	attribute	op	value
1	Nick Niskopoulos	Cleartext-password	:=	06111
2	ping	Cleartext-password	:=	1234
3	radius	Cleartext-password	:=	setupRADIUS

Picture 28: Creating user ping

We create a group ‘radping’ for our user ‘ping’.

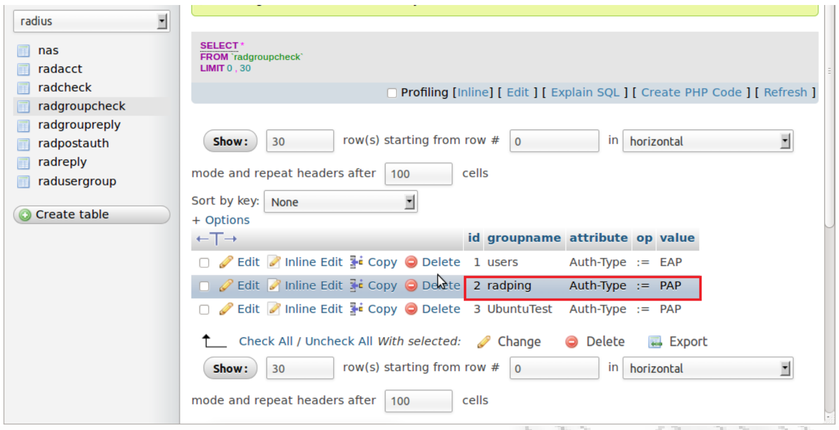


```
SELECT *
FROM `radusergroup`
LIMIT 0 , 30
```

username	groupname	priority
Nick Niskopoulos	users	1
ping	radping	1
radius	UbuntuTest	1

Picture 29: Creating group radping

We set the Authantication Type for our group ‘radping’ to PAP. This means that in order to authenticate the users, only username and password is required.

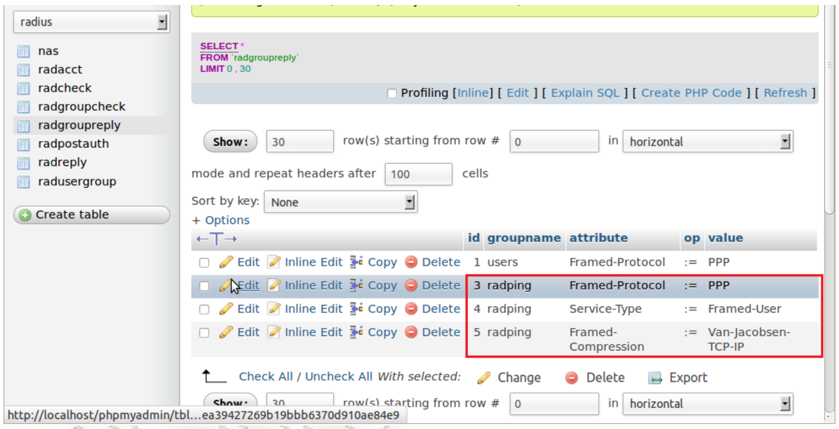

radius
nas
radacct
radcheck
radgroupcheck
radgroupreply
radpostauth
radreply
radusergroup
Create table
Profiling [Inline] [Edit] [Explain SQL] [Create PHP Code] [Refresh]
Show: 30 row(s) starting from row # 0 in horizontal
mode and repeat headers after 100 cells
Sort by key: None
+ Options
Check All / Uncheck All With selected: Change Delete Export

```
SELECT *
FROM `radgroupcheck`
LIMIT 0 , 30
```

id	groupname	attribute	op	value
1	users	Auth-Type	:=	EAP
2	radping	Auth-Type	:=	PAP
3	UbuntuTest	Auth-Type	:=	PAP

Picture 30: Setting the authentication type for group radping

We make the same configuration that we did before only this time our settings are valid for all users of radping group

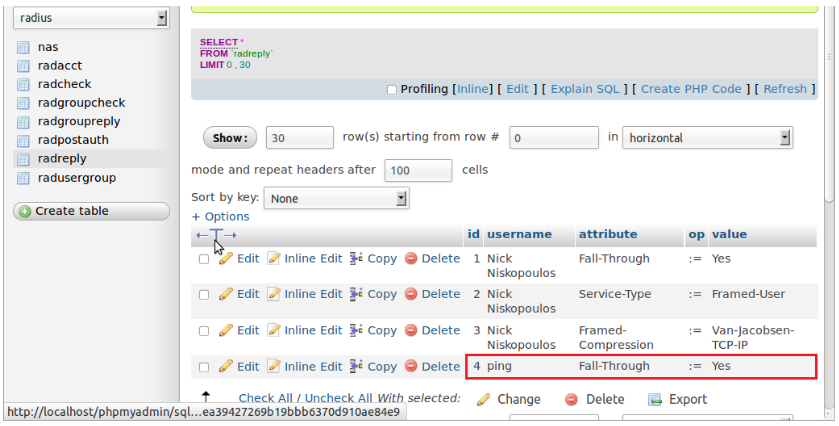

radius
nas
radacct
radcheck
radgroupcheck
radgroupreply
radpostauth
radreply
radusergroup
Create table
Profiling [Inline] [Edit] [Explain SQL] [Create PHP Code] [Refresh]
Show: 30 row(s) starting from row # 0 in horizontal
mode and repeat headers after 100 cells
Sort by key: None
+ Options
Check All / Uncheck All With selected: Change Delete Export
http://localhost/phpmyadmin/tbl...ea39427269b19bbb6370d910ae84e9

```
SELECT *
FROM `radgroupreply`
LIMIT 0 , 30
```

id	groupname	attribute	op	value
1	users	Framed-Protocol	:=	PPP
3	radping	Framed-Protocol	:=	PPP
4	radping	Service-Type	:=	Framed-User
5	radping	Framed-Compression	:=	Van-Jacobsen-TCP-IP

Picture 31: Configuring the radping group

Finaly in radreply table we set 'ping' user to Fall-Through : = Yes


radius
nas
radacct
radcheck
radgroupcheck
radgroupreply
radpostauth
radreply
radusergroup
Create table
Profiling [Inline] [Edit] [Explain SQL] [Create PHP Code] [Refresh]
Show: 30 row(s) starting from row # 0 in horizontal
mode and repeat headers after 100 cells
Sort by key: None
+ Options
Check All / Uncheck All With selected: Change Delete Export
http://localhost/phpmyadmin/sql...ea39427269b19bbb6370d910ae84e9

```
SELECT *
FROM `radreply`
LIMIT 0 , 30
```

id	username	attribute	op	value
1	Nick Niskopoulos	Fall-Through	:=	Yes
2	Nick Niskopoulos	Service-Type	:=	Framed-User
3	Nick Niskopoulos	Framed-Compression	:=	Van-Jacobsen-TCP-IP
4	ping	Fall-Through	:=	Yes

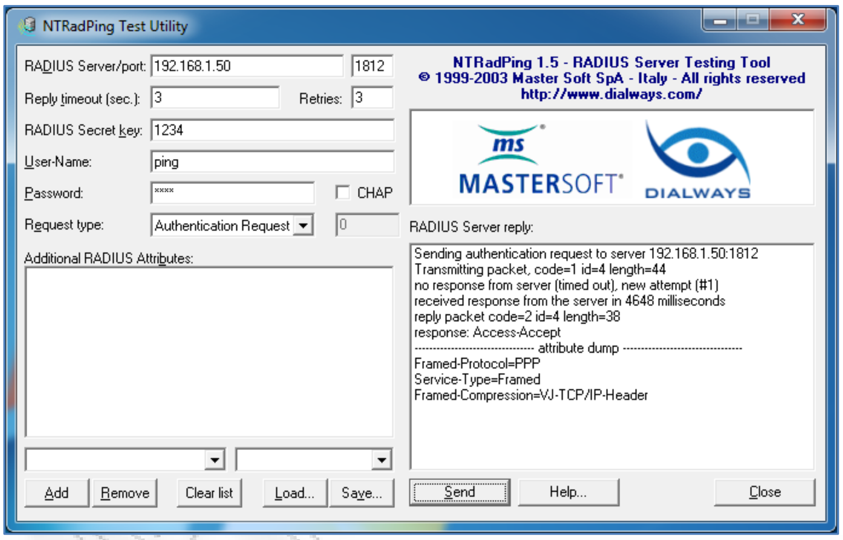
Picture 32: Configuring user ping

Every time we make any change to the freeradius configuration we run:
/etc/init.d/freeradius stop
and
freeradius -X

We run the NTRadPing Test Utility we have downloaded and make the following configuration

RADIUS Server/port	: we set our radius server IP and the authentication port
RADIUS Secret Key	: The Key of NTradping Test client we created before
User-Name	: The new user we created
Password	: The users Password

And click Send. If everything went well we should see response: Access-Accept as in the following picture.


NTRadPing Test Utility

NTRadPing 1.5 - RADIUS Server Testing Tool
© 1999-2003 Master Soft SpA - Italy - All rights reserved
http://www.dialways.com/

RADIUS Server/port: 192.168.1.50 1812
Reply timeout (sec.): 3 Retries: 3
RADIUS Secret key: 1234
User-Name: ping
Password: ****
Request type: Authentication Request 0

RADIUS Server reply:

```
Sending authentication request to server 192.168.1.50:1812
Transmitting packet, code=1 id=4 length=44
no response from server (timed out), new attempt (#1)
received response from the server in 4648 milliseconds
reply packet code=2 id=4 length=38
response: Access-Accept
------------------------------ attribute dump ------------------------------
Framed-Protocol=PPP
Service-Type=Framed
Framed-Compression=VJ-TCP/IP-Header
```

Picture 33: The NTRadtest Utility

802.1X with EAP-TLS Authentication

How to create the Certificates

On our Ubuntu system we open a new terminal and run the following commands

sudo su to run as administrator

cd /etc/ssl to change directory and go to ssl folder

mkdir newcerts private users crl to create these new folders in the ssl directory

touch serial to create a file named serial

gedit serial to open the serial file with the gedit text editor (write 01 save and close the file)

touch ./index.txt to create an empty txt file nemed index

gedit openssl.cnf to openssl.cnf file. Change

```
[ CA_default ]
dir             = ./demoCA                # Where everything is kept
```
to
```
dir             = ./                      # Where everything is kept
```
save and close the file.

touch xpextensions to create a file named xpextensions
gedit xpextensions open xpextensions and write

```
[xpclient_ext]
extendedKeyUsage=1.3.6.1.5.5.7.3.2

[xpserver_ext]
extendedKeyUsage=1.3.6.1.5.5.7.3.1
```

save and close the file. This is a workaround for a bug in Windows XP SP2 which could have negative impact on the TLS process if a third-party radius server is used like Freeradius.

Now we will use openssl to create our certificates

Generating the ROOT-CA-Certificate

openssl req -new -x509 -keyout private/cakey.pem -out cacert.pem -days 1095 -config openssl.cnf

```
rt.pem -days 1095 -config openssl.cnf
Generating a 1024 bit RSA private key
........++++++
.................................................++++++
writing new private key to 'private/cakey.pem'
Enter PEM pass phrase: the root ca password
```

```
Verifying - Enter PEM pass phrase: the root ca password
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:GR
State or Province Name (full name) [Some-State]:Athens
Locality Name (eg, city) []:Piraeus
Organization Name (eg, company) [Internet Widgits Pty Ltd]:PA.PEI
Organizational Unit Name (eg, section) []:Digital Systems
Common Name (eg, YOUR name) []:NISKOPOULOS
Email Address []:radius@gmail.com
```

openssl pkcs12 -export -in cacert.pem -inkey private\cakey.pem -out caroot.p12 -cacerts -descert

```
Enter pass phrase for private/cakey.pem: the root ca password
Enter Export Password: caroot_p12_password
Verifying - Enter Export Password: caroot_p12_password
```
openssl pkcs12 -in caroot.p12 -out caroot.pem
```
Enter Import Password: caroot_p12_password
MAC verified OK
Enter PEM pass phrase: caroot_pem_password
Verifying - Enter PEM pass phrase: caroot_pem_password
```

openssl x509 -in cacert.pem -inform PEM -out cacert.der -outform DER

<There is no output for this command>

Generating the radius server request certificate

openssl req -nodes -new -x509 -keyout radius-req.pem -out radius-req.pem -days 730 -config openssl.cnf
```
Generating a 1024 bit RSA private key
......................................++++++
...++++++
writing new private key to 'radius-req.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
```

```
Country Name (2 letter code) [AU]:GR
State or Province Name (full name) [Some-State]:Athens
Locality Name (eg, city) []:Piraeus
Organization Name (eg, company) [Internet Widgits Pty Ltd]:PA.PEI.
Organizational Unit Name (eg, section) []:Digital Systems
Common Name (eg, YOUR name) []:Radius Server
Email Address []:niskopoulos@gmail.com
```

openssl x509 -x509toreq -in radius-req.pem -signkey radius-req.pem -out radius-tmp.pem

```
Getting request Private Key
Generating certificate request
```

Now we certify the request certificate

openssl ca -config openssl.cnf -policy policy_anything -out radiuscert.pem -extensions xpserver_ext -extfile xpextensions -infiles radius-tmp.pem

```
Using configuration from openssl.cnf
Enter pass phrase for .//private/cakey.pem: the root ca password
Check that the request matches the signature
Signature ok
Certificate Details:

    Serial Number: 1 (0x1)
        Validity
                Not Before: May 23 23:27:50 2012 GMT
                Not After : May 23 23:27:50 2013 GMT
        Subject:
                countryName                = GR
                stateOrProvinceName        = Athens
                localityName               = Piraeus
                organizationName           = PA.PEI.
                organizationalUnitName     = Digital Systems
                commonName                 = Radius Server
                emailAddress               = niskopoulos@gmail.com
        X509v3 extensions:
                X509v3 Extended Key Usage:
                TLS Web Server Authentication
Certificate is to be certified until May 23 23:27:50 2013 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Creating client certificates (We have to repeat this step for every user)

```
openssl req -nodes -new -x509 -keyout nick.nisko-req.pem -out nick.nisko-req.pem -days 730 -config openssl.cnf
Generating a 1024 bit RSA private key
......................................++++++
.........++++++
writing new private key to 'nick.nisko-req.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:GR
State or Province Name (full name) [Some-State]:Athens
Locality Name (eg, city) []:Piraeus
Organization Name (eg, company) [Internet Widgits Pty Ltd]:PA.PEI
Organizational Unit Name (eg, section) []:Digital Systems
Common Name (eg, YOUR name) []:NISKOPOULOS NIKOS
Email Address []:niskopoulos@gmail.com
```

```
openssl x509 -x509toreq -in nick.nisko-req.pem -signkey nick.nisko-req.pem -out nick.nisko-tmp.pem
Getting request Private Key
Generating certificate request
```

```
openssl ca -config openssl.cnf  -policy policy_anything -out nick.nisko-cert.pem -extensions xpserver_ext  -extfile xpextensions  -infiles nick.nisko-tmp.pem

Using configuration from openssl.cnf
Enter pass phrase for .//private/cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 2 (0x2)
        Validity
            Not Before      : May 23 23:50:21 2012 GMT
            Not After       : May 23 23:50:21 2013 GMT
        Subject:
            countryName                     = GR
            stateOrProvinceName             = Athens
            localityName                    = Piraeus
            organizationName                = PA.PEI
            organizationalUnitName          = Digital Systems
            commonName                      = NISKOPOULOS NIKOS
            emailAddress                    = niskopoulos@gmail.com
        X509v3 extensions:
            X509v3 Extended Key Usage:
                TLS Web Server Authentication
Certificate is to be certified until May 23 23:50:21 2013 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

openssl pkcs12 -export -in nick.nisko-cert.pem -out nick.nisko-cert.p12 -inkey nick.nisko-req.pem -descert

```
Enter Export Password: nick.nisko_password
Verifying - Enter Export Password: nick.nisko_password
```

openssl pkcs12 -nodes -in nick.nisko-cert.p12 -out nick.nisko-key.pem

```
Enter Import Password: nick.nisko_password
MAC verified OK
```

Now we create a Diffi-Hellmann-Parameter (DH)

openssl dhparam -out dh1024.pem 1024

```
Generating DH parameters, 1024 bit long safe prime, generator 2
This is going to take a long time
................+........................   …..    ...++*++*++*.
```

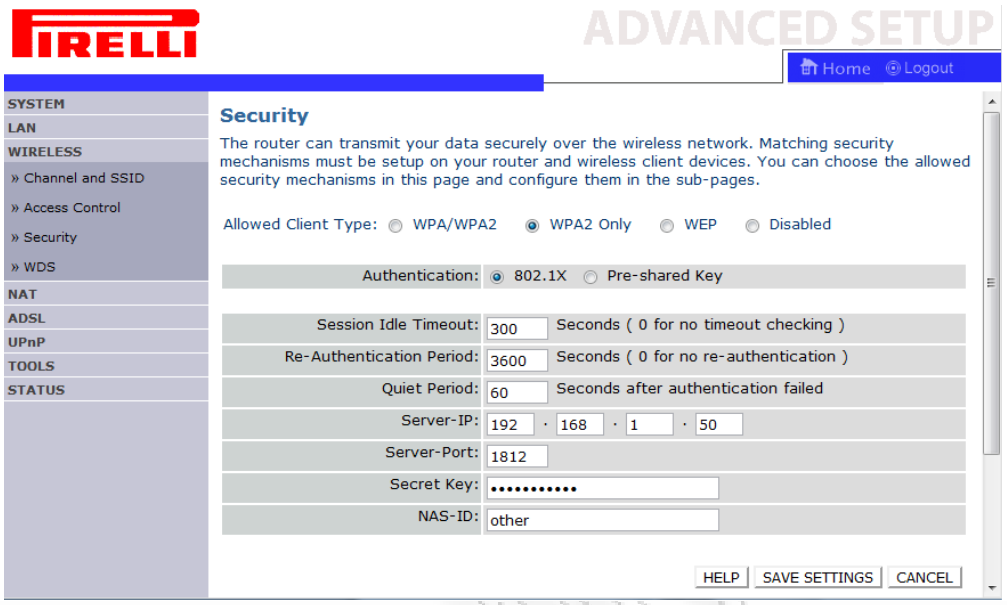
Freeradius configuration for EAP-TLS

Configuring the wireless router (Client)

We are going to use a Pirelli wireless router but the basic configuration is approximately the same for all wireless access points.

- We open a browser and we login to our access-point. We select Wireless>Security and we set authentication type to 802.1X.
- We set the IP of our Freeradius Server, in our case 192.168.1.50 (we suggest that we set a static IP to the radius server)
- We set the Server-Port to 1812
- We use the Secret Key that we used when we configured the NAS table on our database, in our case setupRADIUS.

- We set the NAS-ID to other
- Finally we save our settings



Security

The router can transmit your data securely over the wireless network. Matching security mechanisms must be setup on your router and wireless client devices. You can choose the allowed security mechanisms in this page and configure them in the sub-pages.

Allowed Client Type: WPA/WPA2 [✓ WPA2 Only] WEP Disabled

Authentication:	[✓ 802.1X] Pre-shared Key
Session Idle Timeout:	300 Seconds (0 for no timeout checking)
Re-Authentication Period:	3600 Seconds (0 for no re-authentication)
Quiet Period:	60 Seconds after authentication failed
Server-IP:	192 . 168 . 1 . 50
Server-Port:	1812
Secret Key:	•••••••••••
NAS-ID:	other

Picture 34: The Pirelli router setup web page

Configuring the Radius Server

We will have to copy **radius-req.pem**, **radiuscert.pem**, **cacert.pem**, **dh1024.pem** that we have created in the /etc/ssl directory into /etc/freeradius/certs directory. When we do that, we open a terminal:

cd /etc/freeradius/ to change directory

gedit eap.conf to open eap.conf configuration file.
Make the following changes:

```
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 = tls
.
.
.
.
cisco_accounting_username_bug = yes
.
.
.
tls {
                #
                #  These is used to simplify later configurations.
                #
                certdir = ${confdir}/certs
                cadir = ${confdir}/certs

                #private_key_password = 1234
                private_key_file = ${certdir}/radius-req.pem

                #  If Private key & Certificate are located in
                #  the same file, then private_key_file &
                #  certificate_file must contain the same file
                #  name.
                #
                #  If CA_file (below) is not used, then the
                #  certificate_file below MUST include not
                #  only the server certificate, but ALSO all
                #  of the CA certificates used to sign the
                #  server certificate.
                certificate_file = ${certdir}/radius-cert.pem


                #  Trusted Root CA list
                #
                #  ALL of the CA's in this list will be trusted
                #  to issue client certificates for authentication.
                #
                #  In general, you should use self-signed
                #  certificates for 802.1x (EAP) authentication.
                #  In that case, this CA file should contain
                #  *one* CA certificate.
                #
                #  This parameter is used only for EAP-TLS,
                #  when you issue client certificates.  If you do
                #  not use client certificates, and you do not want
                #  to permit EAP-TLS authentication, then delete
                #  this configuration item.
                CA_file = ${cadir}/cacert.pem

                #
                #  For DH cipher suites to work, you have to
                #  run OpenSSL to create the DH file first:
                #
                #        openssl dhparam -out certs/dh 1024
                #
                #dh_file = ${certdir}/dh
                dh_file = ${certdir}/dh1024.pem
                random_file = /dev/urandom
```

```
.
.
.
 fragment_size = 1024
.
.
 include_length = yes                    remove #
.
.
 check_cert_cn = %{User-Name}
```

save and close the window.

gedit radiusd.conf to open radiusd file.

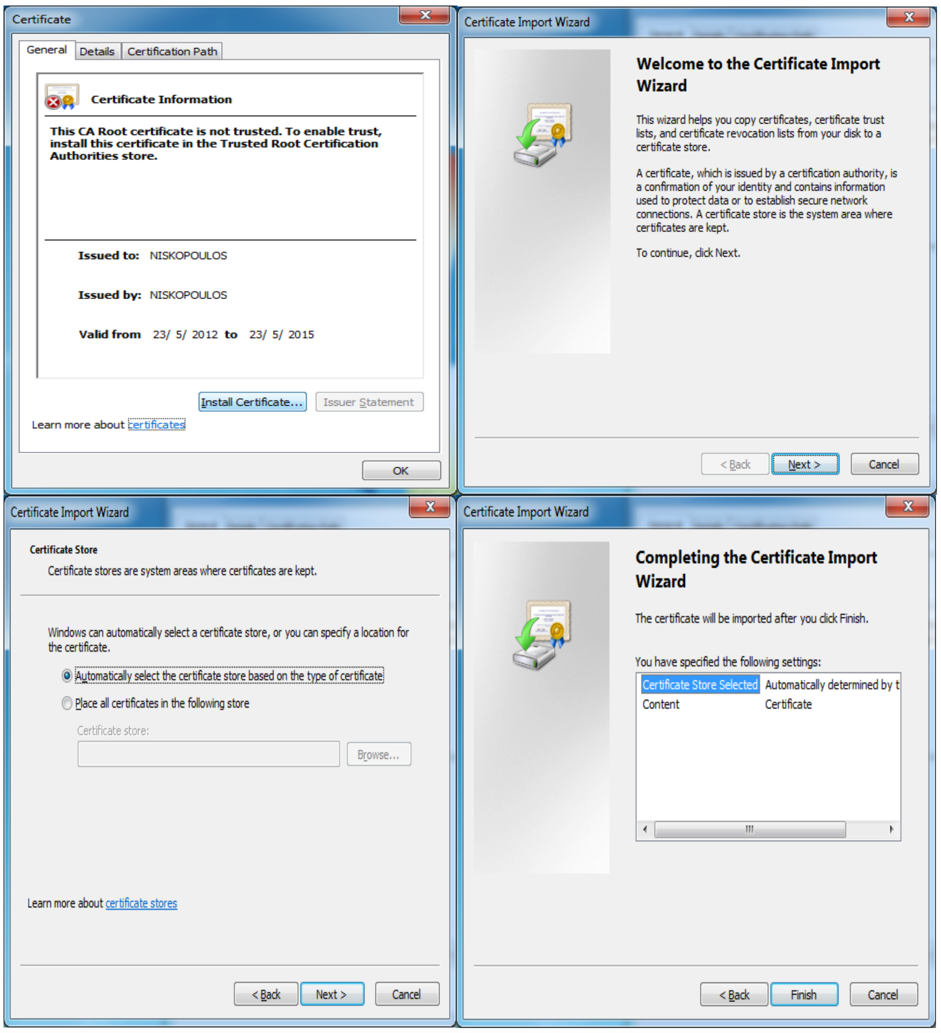
```
prefix = /usr
exec_prefix = /usr
sysconfdir = /etc
localstatedir = /var
sbindir = ${exec_prefix}/sbin
logdir = /var/log/freeradius
raddbdir = /etc/freeradius
radacctdir = ${logdir}/radacct
certdir = ${sysconfdir}/freeradius/certs        add this line
```

What a client needs to authenticate with EAP-TLS method

Linux:
nick.nisko-cert.pem
nick.nisko-req.pem
cacert.pem

Windows:
nick.nisko-cert.p12
cacert.der

We are going to test to authenticate a user from win7 operating system.
We copy these two files to our system nick.nisko-cert.p12, cacert.der and first we double click on the cacert file and click install certificate>Next>leave the default "Automaticaly select the certificate store based on the type of certificate" and Next>Finish.



Certificate

General | Details | Certification Path

Certificate Information

This CA Root certificate is not trusted. To enable trust, install this certificate in the Trusted Root Certification Authorities store.

Issued to: NISKOPOULOS

Issued by: NISKOPOULOS

Valid from 23/ 5/ 2012 to 23/ 5/ 2015

Install Certificate... | Issuer Statement

Learn more about certificates

OK

Certificate Import Wizard

Welcome to the Certificate Import Wizard

This wizard helps you copy certificates, certificate trust lists, and certificate revocation lists from your disk to a certificate store.

A certificate, which is issued by a certification authority, is a confirmation of your identity and contains information used to protect data or to establish secure network connections. A certificate store is the system area where certificates are kept.

To continue, click Next.

< Back | Next > | Cancel

Certificate Import Wizard

Certificate Store

Certificate stores are system areas where certificates are kept.

Windows can automatically select a certificate store, or you can specify a location for the certificate.

[✓ Automatically select the certificate store based on the type of certificate]

Place all certificates in the following store

Certificate store:

Browse...

Learn more about certificate stores

< Back | Next > | Cancel

Certificate Import Wizard

Completing the Certificate Import Wizard

The certificate will be imported after you click Finish.

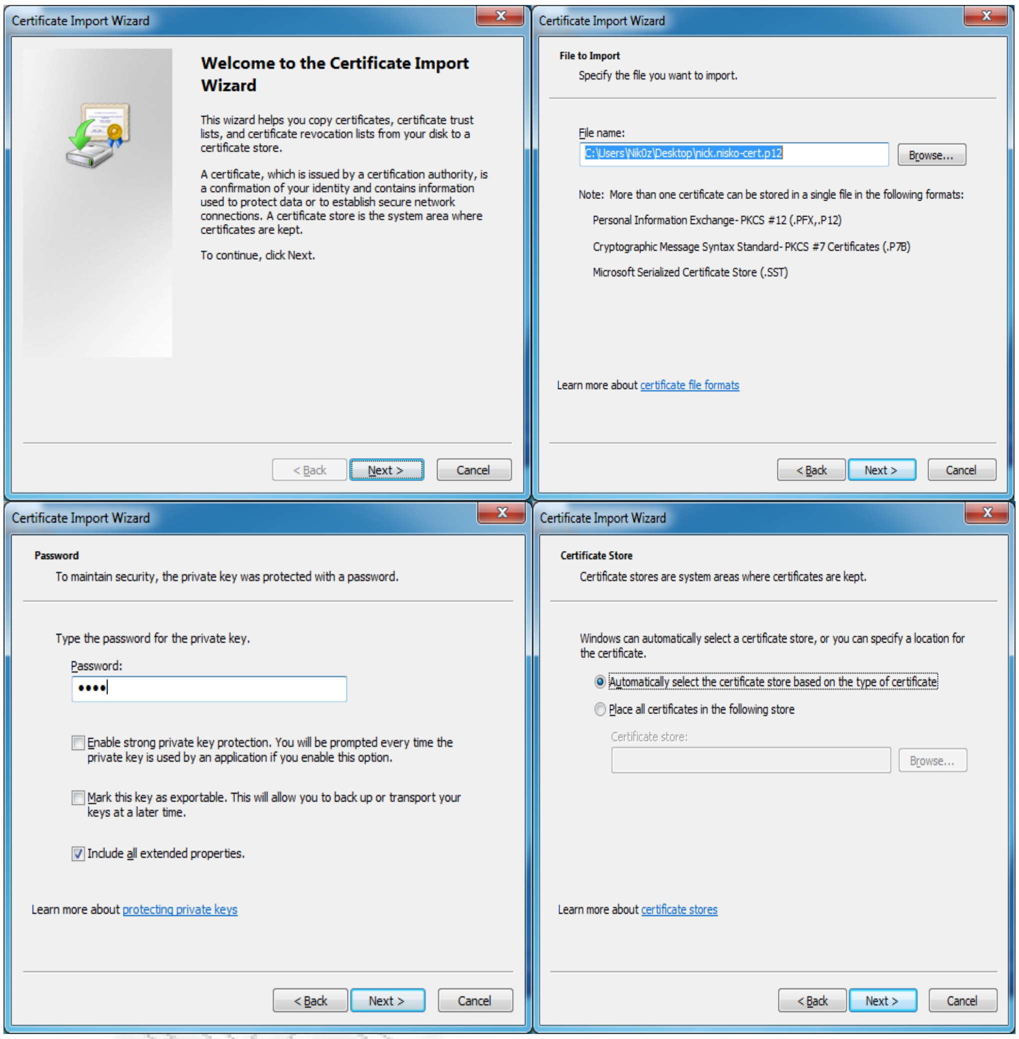
You have specified the following settings:

Certificate Store Selected	Automatically determined by t
Content	Certificate

< Back | Finish | Cancel

Picture 35:How to install certification authority to windows client

Then we do the same procedure for the users' certificate. Double click on nick.nisko-cert.p12>Next>Next>insert the nick.nisko_password and make sure the "Enable strong private key protection" is NOT checked>and Finish.


Certificate Import Wizard

Welcome to the Certificate Import Wizard

This wizard helps you copy certificates, certificate trust lists, and certificate revocation lists from your disk to a certificate store.

A certificate, which is issued by a certification authority, is a confirmation of your identity and contains information used to protect data or to establish secure network connections. A certificate store is the system area where certificates are kept.

To continue, click Next.

< Back | Next > | Cancel

Certificate Import Wizard

File to Import
Specify the file you want to import.

File name:
C:\Users\Nik0z\Desktop\nick.nisko-cert.p12 | Browse...

Note: More than one certificate can be stored in a single file in the following formats:

Personal Information Exchange- PKCS #12 (.PFX,.P12)

Cryptographic Message Syntax Standard- PKCS #7 Certificates (.P7B)

Microsoft Serialized Certificate Store (.SST)

Learn more about certificate file formats

< Back | Next > | Cancel

Certificate Import Wizard

Password
To maintain security, the private key was protected with a password.

Type the password for the private key.

Password:
••••

[ ] Enable strong private key protection. You will be prompted every time the private key is used by an application if you enable this option.

[ ] Mark this key as exportable. This will allow you to back up or transport your keys at a later time.

[✓] Include all extended properties.

Learn more about protecting private keys

< Back | Next > | Cancel

Certificate Import Wizard

Certificate Store
Certificate stores are system areas where certificates are kept.

Windows can automatically select a certificate store, or you can specify a location for the certificate.

● Automatically select the certificate store based on the type of certificate

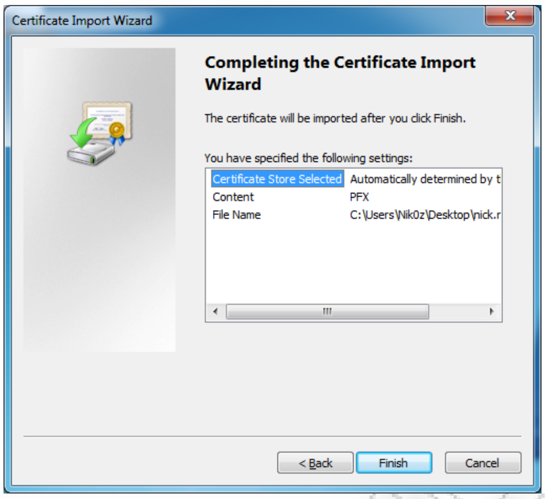
○ Place all certificates in the following store

Certificate store:
Browse...

Learn more about certificate stores

< Back | Next > | Cancel

Picture 36: How to install user certificate to windows 7


Completing the Certificate Import Wizard

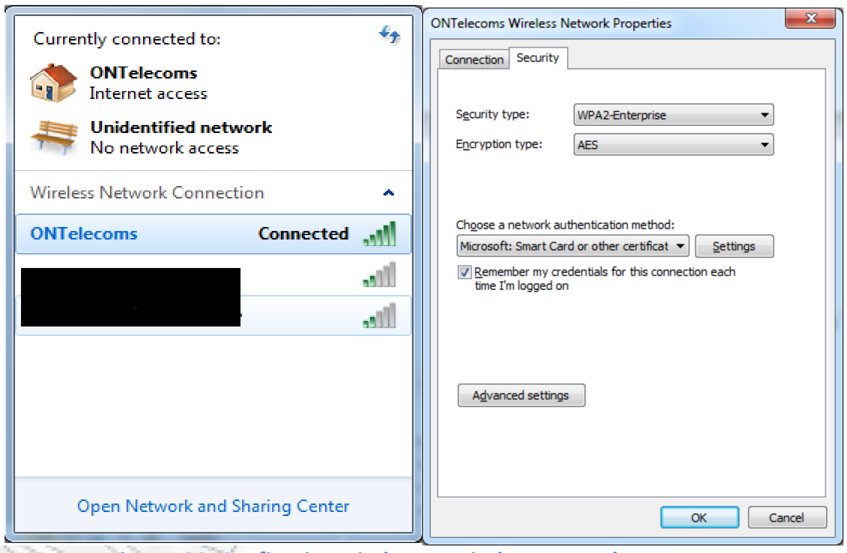
The certificate will be imported after you click Finish.

You have specified the following settings:

Certificate Store Selected	Automatically determined by t
Content	PFX
File Name	C:\Users\Nik0z\Desktop\nick.r

Picture 37: How to install user certificate to windows 7

Now we are going to configure the wireless setting in windows in order to authenticate the user by using the certificates. We click on our wireless connections> right click on our access point>properties> we change the
Security type to WPA2-Enterprise
Encryption Type to AES
Network authentication method to Microsoft: smartcard or other certificate

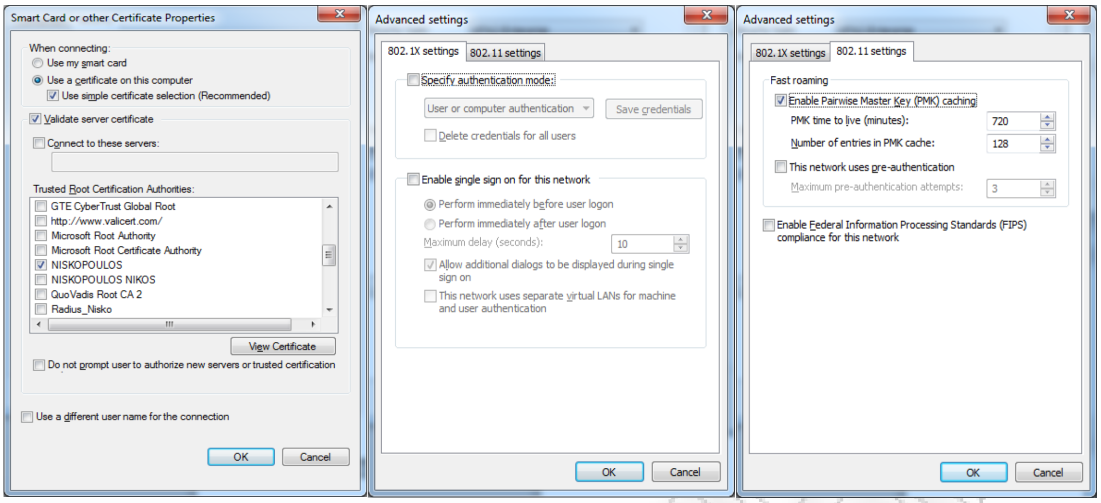

Currently connected to:
ONTelecoms
Internet access
Unidentified network
No network access
Wireless Network Connection
ONTelecoms Connected
[redacted]

ONTelecoms Wireless Network Properties

Security type: WPA2-Enterprise
Encryption type: AES
Choose a network authentication method: Microsoft: Smart Card or other certificat
☑ Remember my credentials for this connection each time I'm logged on

Picture 38: Configuring windows 7 wireless network center

Click on settings and check the Trusted Root Certification Authority that you have already inserted, click OK.
Then click on Advanced settings and check the following configuration for 802.1X and 802.11 settings, then click OK.


Smart Card or other Certificate Properties

When connecting:
- Use my smart card
- [✓ Use a certificate on this computer]
  - [✓] Use simple certificate selection (Recommended)

[✓] Validate server certificate

[ ] Connect to these servers:

Trusted Root Certification Authorities:
- [ ] GTE CyberTrust Global Root
- [ ] http://www.valicert.com/
- [ ] Microsoft Root Authority
- [ ] Microsoft Root Certificate Authority
- [✓] NISKOPOULOS
- [ ] NISKOPOULOS NIKOS
- [ ] QuoVadis Root CA 2
- [ ] Radius_Nisko

[ ] Do not prompt user to authorize new servers or trusted certification

[ ] Use a different user name for the connection

Advanced settings — 802.1X settings

[ ] Specify authentication mode:
User or computer authentication
[ ] Delete credentials for all users

[ ] Enable single sign on for this network
- [✓ Perform immediately before user logon]
- Perform immediately after user logon

Maximum delay (seconds): 10

[✓] Allow additional dialogs to be displayed during single sign on

[ ] This network uses separate virtual LANs for machine and user authentication

Advanced settings — 802.11 settings

Fast roaming

[✓] Enable Pairwise Master Key (PMK) caching

PMK time to live (minutes): 720

Number of entries in PMK cache: 128

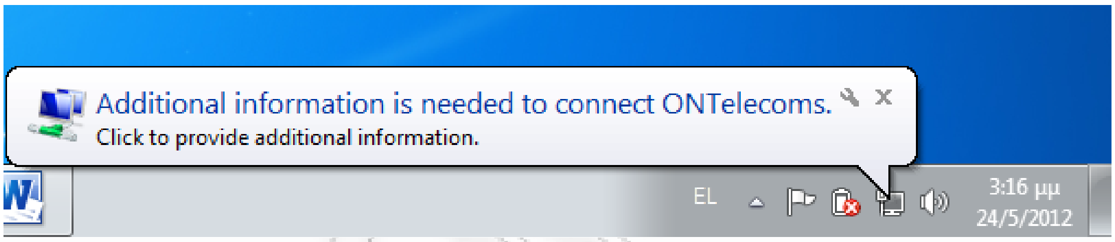
[ ] This network uses pre-authentication

Maximum pre-authentication attempts: 3

[ ] Enable Federal Information Processing Standards (FIPS) compliance for this network

Picture 39: Congiguring wireless 802.1x settings

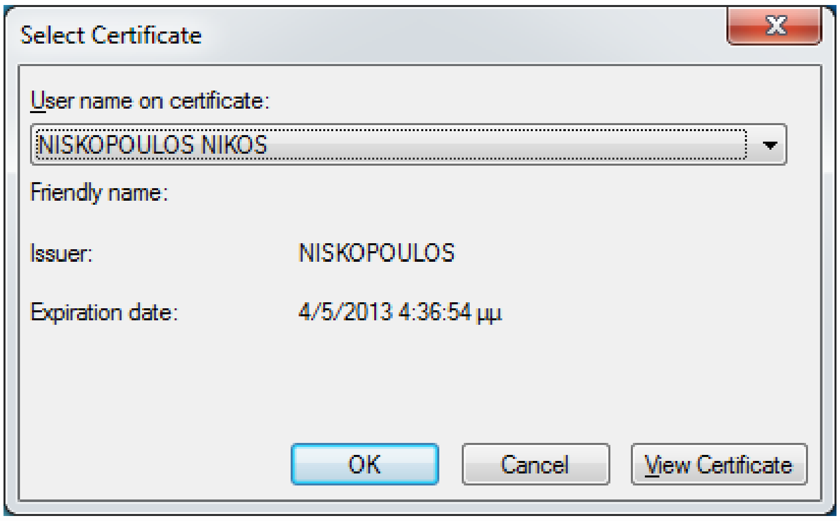
Go to your wireless access point and click connect. After a few seconds a message like the one above will appear click on it and the Select Certificate Window will appear.


Additional information is needed to connect ONTelecoms.

Click to provide additional information.

Picture 40: Windows network configuration screenshot

Select the certificate you have created and click OK.


Select Certificate

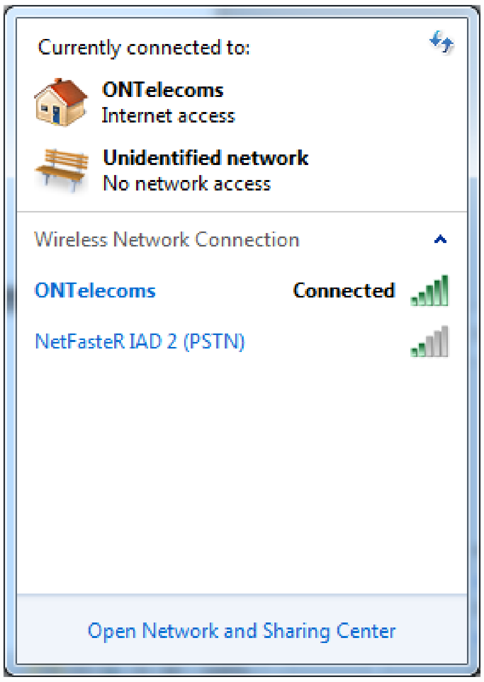
User name on certificate: NISKOPOULOS NIKOS

Friendly name:

Issuer: NISKOPOULOS

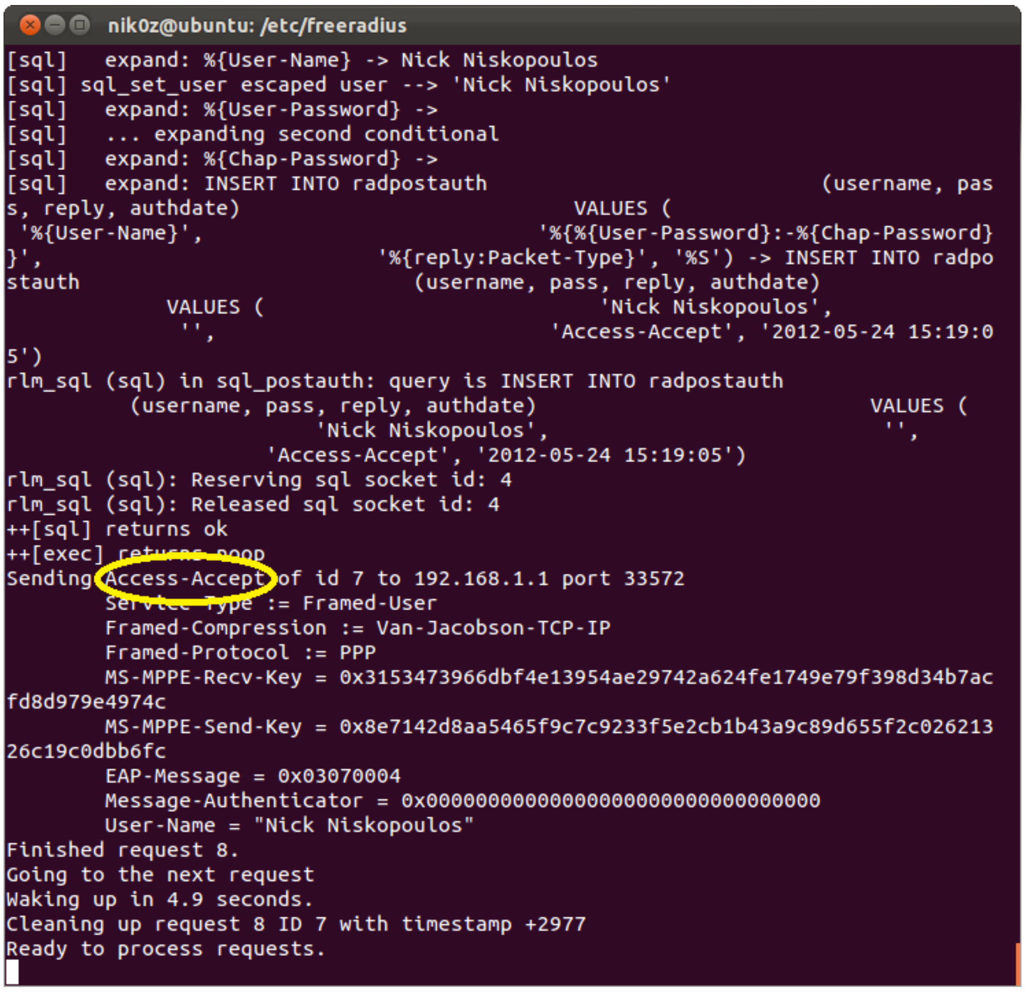
Expiration date: 4/5/2013 4:36:54 μμ

Picture 41: Selecting the preferable certificate



Picture 42: Connection to wireless client ONTelecoms

We finally connect to our wireless network. The radius server will automatically authenticate you and the response 'Access-Accept' of the radius server in debug mode will be like in the following picture.



```
[sql]   expand: %{User-Name} -> Nick Niskopoulos
[sql] sql_set_user escaped user --> 'Nick Niskopoulos'
[sql]   expand: %{User-Password} ->
[sql]   ... expanding second conditional
[sql]   expand: %{Chap-Password} ->
[sql]   expand: INSERT INTO radpostauth                           (username, pas
s, reply, authdate)                                    VALUES (
 '%{User-Name}',                                    '%{%{User-Password}:-%{Chap-Password}
}',                               '%{reply:Packet-Type}', '%S') -> INSERT INTO radpo
stauth                               (username, pass, reply, authdate)
             VALUES (                        'Nick Niskopoulos',
              '',                                   'Access-Accept', '2012-05-24 15:19:0
5')
rlm_sql (sql) in sql_postauth: query is INSERT INTO radpostauth
          (username, pass, reply, authdate)                          VALUES (
                         'Nick Niskopoulos',                          '',
                     'Access-Accept', '2012-05-24 15:19:05')
rlm_sql (sql): Reserving sql socket id: 4
rlm_sql (sql): Released sql socket id: 4
++[sql] returns ok
++[exec] returns noop
Sending Access-Accept of id 7 to 192.168.1.1 port 33572
        Service-Type := Framed-User
        Framed-Compression := Van-Jacobson-TCP-IP
        Framed-Protocol := PPP
        MS-MPPE-Recv-Key = 0x3153473966dbf4e13954ae29742a624fe1749e79f398d34b7ac
fd8d979e4974c
        MS-MPPE-Send-Key = 0x8e7142d8aa5465f9c7c9233f5e2cb1b43a9c89d655f2c026213
26c19c0dbb6fc
        EAP-Message = 0x03070004
        Message-Authenticator = 0x00000000000000000000000000000000
        User-Name = "Nick Niskopoulos"
Finished request 8.
Going to the next request
Waking up in 4.9 seconds.
Cleaning up request 8 ID 7 with timestamp +2977
Ready to process requests.
```

Picture 43: Freeradius server screenshot

Radius Server Response

The complete response is approximately 15 pages here are the most interesting parts:

```
rad_recv: Access-Request packet from host 192.168.1.1 port 33568, id=3, length=165
        User-Name = "Nick Niskopoulos"
        NAS-IP-Address = 0.0.0.0
        Called-Station-Id = "00-1C-A2-B5-86-A1:ONTelecoms"
        Calling-Station-Id = "00-21-5D-4D-A3-50"
        NAS-Identifier = "other"
        NAS-Port = 29
        Service-Type = Framed-User
        Framed-MTU = 1400
        NAS-Port-Type = Wireless-802.11
        EAP-Message = 0x02030015014e69636b204e69736b6f706f756c6f73
        Message-Authenticator = 0xcdf863571f57a51d7525748111efeb8d
.
.
.
[eap] EAP packet type response id 3 length 21
.
.
.
[sql]   expand: SELECT id, username, attribute, value, op          FROM radcheck
        WHERE username = '%{SQL-User-Name}'        ORDER BY id -> SELECT
        id, username, attribute, value, op          FROM radcheck          WHERE
        username = 'Nick Niskopoulos'       ORDER BY id
[sql] User found in radcheck table
.
.
.
[sql] sql_set_user escaped user --> 'Nick Niskopoulos'
rlm_sql (sql): Reserving sql socket id: 4
.
.
Found Auth-Type = EAP
[eap] processing type tls
[eap] EAP Identity
[tls] Requiring client certificate
[tls] Initiate
[tls] Start returned 1
++[eap] returns handled
.
.
.
Sending Access-Challenge of id 3 to 192.168.1.1 port 33568
```

```
        Service-Type := Framed-User
        Framed-Compression := Van-Jacobson-TCP-IP
        Framed-Protocol := PPP
        EAP-Message = 0x010400060d20
        Message-Authenticator = 0x00000000000000000000000000000000
        State = 0x248c22ae24882fb57eb42a13aa8c525b
Finished request 4.
Going to the next request
Waking up in 4.9 seconds.
.
.
.
rad_recv: Access-Request packet from host 192.168.1.1 port 33569, id=4, length=267
        User-Name = "Nick Niskopoulos"
        NAS-IP-Address = 0.0.0.0
        Called-Station-Id = "00-1C-A2-B5-86-A1:ONTelecoms"
        Calling-Station-Id = "00-21-5D-4D-A3-50"
        NAS-Identifier = "other"
        NAS-Port = 29
        Service-Type = Framed-User
        Framed-MTU = 1400
        NAS-Port-Type = Wireless-802.11
        State = 0x248c22ae24882fb57eb42a13aa8c525b
        EAP-Message                                                         =
        0x020400690d800000005f160301005a0100005603014fbe272d545bf7b1d187
        2459b09e357bba0eab025405d36258ef36c733ed0161000018002f0035000500
        0ac013c014c009c00a003200380013000401000015ff01000100000a00060004
        00170018000b00020100
Sending Access-Challenge of id 4 to 192.168.1.1 port 33569
        Service-Type := Framed-User
        Framed-Compression := Van-Jacobson-TCP-IP
        Framed-Protocol := PPP
        EAP-Message                                                         =
        0x010504000dc0000006d7160301003102000002d03014fbe272df8c32cd499cb
        d294cc859df30e3c5bf406ddb9dbc1645582c95adacd00002f000005ff0100010
        016030105e70b0005e30005e00002cc308202c830820231a003020102020101 3
        00d06092a864886f70d0101050500308 19d310b3009060355040613024752 31
        0f300d06035504080c06417468656e73310f300d06035504070c06417468656e
        7331133011060355040a0c0a556e2e506972616575733118301606035504 0b0
        c0f4469676974616c5f53797374656d7331153013060355040 30c0c526164697
        5735f4e69736b6f3126302406092a864886f70d01090116176e69736b6f70
        EAP-Message                                                         =
        0x6f756c6f7340686f746d61696c2e636f6d301e170d31323035313031353035 3
        0315a170d3133303531303135303530315a30819c310b30090603550406130 2
        4752310f300d06035504080c06417468656e73310f300d06035504070c064174
        68656e7331133011060355040a0c0a556e2e50697261657573311830160603 5
        5040b0c0f4469676974616c5f53797374656d73311630140603550403 0c0d526
        1646975735f53657276657231243022060 92a864886f70d01090116156e69736
        b6f706f756c6f7340676d61696c2e636f6d30819f300d06092a864886f70d0101
        01050003818d0030818902818100b848ec932ce3658f5dfae67d36d3b4c7
        EAP-Message                                                      =
        0x4753d280be029c084cac0c0c235ea56f91938f7b69f58999798fe72caf45bf15
        c6c946f69e5dbf8f1f6893e77115e35cb95bbeb04adeea836b0553099839c415a7
        a002a51abc4c00e12c33bb2c4674b7092437a5c49a13e4c3a473b2238e78c5753
        b8e5ff6562a9eeed9b2f009a5a79d0203010001a317301530130603551d25040c
        300a06082b0601050507030130 0d06092a864886f70d0101050500038181 00b
        1898737bf6af45c1acfae838d929c36af6c83e2c7b5572fd425a195adef2385cf0a
        1c0a22ce4728afca782091198ef4d062123dc204700c4a3cabcd44036d7ca30ef2
        95d1360679626911964230ecf363ddcf9df6527cc4df9c7706
        EAP-Message                                                      =
        0x6a5c37a7618a4a760bcd9d149a078e4b27bf9b05a5edb0195d6cba6e28c1e62
        f753dd49900030e3082030a30820273a003020102020900f2988bbd5e4d585f3
        00d06092a864886f70d0101050500308 19d310b3009060355040613024752 31
        0f300d06035504080c06417468656e73310f300d06035504070c06417468656e
        7331133011060355040a0c0a556e2e5069726165757331183016060355040b0
        c0f4469676974616c5f53797374656d73311530130603550403 0c0c526164697
        5735f4e69736b6f3126302406092a864886f70d01090116176e69736b6f706f75
        6c6f7340686f746d61696c2e636f6d301e170d3132303531303134333432
        EAP-Message = 0x325a170d3135303531303134
        Message-Authenticator = 0x00000000000000000000000000000000
        State = 0x248c22ae25892fb57eb42a13aa8c525b
Finished request 5.
Going to the next request
Waking up in 4.9 seconds.

rad_recv: Access-Request packet from host 192.168.1.1 port 33570, id=5, length=168
        User-Name = "Nick Niskopoulos"
        NAS-IP-Address = 0.0.0.0
        Called-Station-Id = "00-1C-A2-B5-86-A1:ONTelecoms"
        Calling-Station-Id = "00-21-5D-4D-A3-50"
        NAS-Identifier = "other"
        NAS-Port = 29
        Service-Type = Framed-User
        Framed-MTU = 1400
        NAS-Port-Type = Wireless-802.11
        State = 0x248c22ae25892fb57eb42a13aa8c525b
        EAP-Message = 0x020500060d00
        Message-Authenticator = 0x689e2c354fbf85ec13783da74882a50d
# Executing section authorize from file /etc/freeradius/sites-enabled/default
+- entering group authorize {...}
++[preprocess] returns ok
.
```

```
.
.
Sending Access-Accept of id 7 to 192.168.1.1 port 33572
            Service-Type := Framed-User
            Framed-Compression := Van-Jacobson-TCP-IP
            Framed-Protocol := PPP
            MS-MPPE-Recv-Key                                                    =
            0x3153473966dbf4e13954ae29742a624fe1749e79f398d34b7acfd8d979e4974
            c
            MS-MPPE-Send-Key                                                    =
            0x8e7142d8aa5465f9c7c9233f5e2cb1b43a9c89d655f2c02621326c19c0dbb6fc
            EAP-Message = 0x03070004
            Message-Authenticator = 0x00000000000000000000000000000000
            User-Name = "Nick Niskopoulos"
Finished request 8.
Going to the next request
Waking up in 4.9 seconds.
Cleaning up request 8 ID 7 with timestamp +2977
Ready to process requests.
```

As we can see everything works properly, the sql database found both the client and the user and the tls authentication method was completed successfully.

Freeradius with LDAP on Linux Ubuntu 11.10

Freeradius with LDAP

A Radius Server, is a daemon for unix operating systems which allows one to set up a radius protocol server, which is usually used for authentication and accounting of dial-up users. To use server, you also need a correctly setup client which will talk to it, usually a terminal server or a PC with appropriate which emulates it (PortSlave, radiusclient etc).

Radius has its own database of users since this information is already contained in LDAP it is far more convenient to use its advantages.There are several freeware Radius servers, the one that has good support for LDAP and will be used in this implementation is the FreeRadius server.

LDAP stands for *Lightweight Directory Access Protocol*. LDAP is a *protocol* which provides access to a compliant *directory* via TCP/IP. The strengths of LDAP-compliant directories include speed, simplicity, and the ability to be replicated and distributed across several servers. A LDAP directory can be used to store a great deal of information: from user login credentials to company telephone directories.

LDAP was created as a less complicated implementation of the *Directory Access Protocol* (DAP), and is based on the OSI X.500 standard. These standards establish directories as being hierarchical---representing the structure of an organization. There are many directories that support the LDAP protocol, each with their own benefits and drawbacks. Examples include openLDAP, the open-source implementation that ships with SLES and eDirectory, Novell's flagship identity management product.

Why LDAP instead of MySQL? The largest general difference between directories and databases is complexity. Databases are capable of storing almost any arbitrary set of information and can be greatly customized for a specific purpose. They also provide a complex query interface, allowing for flexible searches returning customized results. Directories, on the other hand, tend to have very specific implementations that follow a strict pattern or schema. This allows them to be extremely fast, and allows for easy organization and comprehension of the data they store.

Installing Freeradius with LDAP

1. We start by running the commands as admin

```
sudo su
```

2. We install freeradius

```
apt-get install freeradius
```

3. And install freeradius-ldap

```
apt-get install freeradius-ldap
```

4. We install the openldap server and the necessary utilities

```
apt-get install slapd ldap-utils
```

5. We reconfigure the ldap server to make some changes

```
dpkg-reconfigure slapd
```

[

Omit OpenLDAP…> NO

DNS domain name..> example.com

Organization name..> Nick Nisko (anything you want)

Password..> 1234

Confirm..>1234

Database backend…> BDB

NO

YES

NO

]

Now we start the freeradius configuration

1) We congigure the clinet file of freeradius as below

gedit /etc/freeradius/clients.conf

```
[

//at the end of the file we add the clients that we want, these clients //that I have added are referred to my access point and the NTRadping //client that I use from my windows system to test the radius server

        Clinet 192.168.88.1 {

                Secret          = 1234

                Shortname       = radping

        }

        Clinet 192.168.1.1 {

                Secret          = setupRADIUS

                Shortname       = ONTelecoms

        }

]
```

2) We configure the ldap file

gedit /etc/freeradius/modules/ ldap

```
 [

p

]
```

3) Then we configure the default file

gedit /etc/freeradius/sites-available/default

```
[

authorize {
```

```
#  Read the 'users' file
#  files
.
.
  ldap
.
.
}
authenticate {
.
  Auth-Type LDAP {
                ldap
        }
.
}
]
```

4) And also in the sites-available directory we configure the inner-tunnel file

    gedit inner-tunnel

```
{
authorize
        # files
        .
        .
          ldap
        .
        .
        Auth-Type LDAP{
                Ldap

        }

}
```

5) Final at the freeradius configuration we make the following changes at the radius.conf file

   gedit /etc/freeradius/radiusd.conf

```
{

$Include clients

}
```

Installing phpladapadmin

1. Now we should install phpldapadmin in order to easily access and configure our ldap database.

   apt-get install phpldapadmin

//This is a web based tool which helps us to manage our LDAP database. For the //former approach, just edit /etc/phpldapadmin/config.php. You'll probably need to //change the following lines (the most important is to add your own ip address of the //ldap server instead of the highlighted one):

```
$ldapservers->SetValue($i,'server','name','My LDAP Server');  // The name to display

$ldapservers->SetValue($i,'server','host','192.168.1.50');    // Address of the LDAP server

$ldapservers->SetValue($i,'server','port','389');   // Port number

$ldapservers->SetValue($i,'server','base',array('dc=example,dc=com'));  // Base dn

$ldapservers->SetValue($i,'login','string','uid=<username>,ou=People,dc=example,dc=com');
```

2. We make the following changes in the ldap server’s ldap.cong file

   gedit /etc/ldap/ldap.conf

```
{
BASE  dc=example, dc= com
URI     ldap://192.168.1.50
}
```

(

Go and check on a browser 192.168.1.50/phpldapadmin

If error (Your PHP memory limit is low-currently 16M):

gedit /etc/php5/apache2/php.ini

```
{
memory_limit = 32M
}
```

Invoke-rc.d apache2 restart

)

3. We want to restart our radius server so we open a new terminal and run the commands

/etc/init.d/freeradius stop

freeradius -X

(Should you have any problem with permissions "Opening file : ...... Permission denied" try

sudo /etc/init.d/freeradius stop

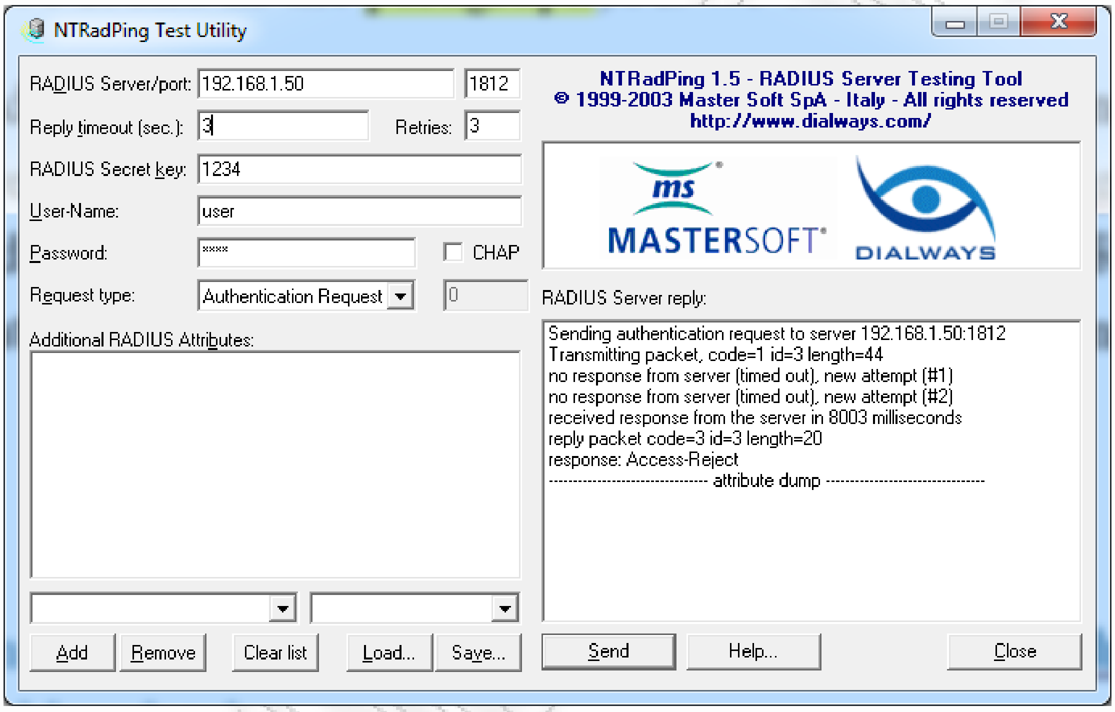
sudo freeradius -X   )

NTRadping Test Utility

We will use NTRadping to test if all the configuration is correct

We use NTRadping tool on our windows OS and send an authentication request to test if our radius server works (instructions regarding the NTRadPing tool and its installation can be found in the paragraph).As we can see our server works properly.

We use the ip address of our radius server and the authentication port 1812, we use the Radius Secret key that we have entered for our windows (radping) client and for User-Name user and password 1234 we receive a response: Access Reject since we haven't yet created such a user in our LDAP database, as we can see in the following picture.



NTRadPing Test Utility
NTRadPing 1.5 - RADIUS Server Testing Tool
© 1999-2003 Master Soft SpA - Italy - All rights reserved
http://www.dialways.com/
Additional RADIUS Attributes:
Add
Remove
Clear list
Load...
Save...
Send
Help...
Close

RADIUS Server/port: 192.168.1.50 1812
Reply timeout (sec.): 3 Retries: 3
RADIUS Secret key: 1234
User-Name: user
Password: ****
CHAP: ☐
Request type: Authentication Request 0

RADIUS Server reply:

```
Sending authentication request to server 192.168.1.50:1812
Transmitting packet, code=1 id=3 length=44
no response from server (timed out), new attempt (#1)
no response from server (timed out), new attempt (#2)
received response from the server in 8003 milliseconds
reply packet code=3 id=3 length=20
response: Access-Reject
------------------------------ attribute dump ------------------------------
```

Picture 44:Testing the curent configuration with NTRadPing Test utility

Configuring the LDAP Directory

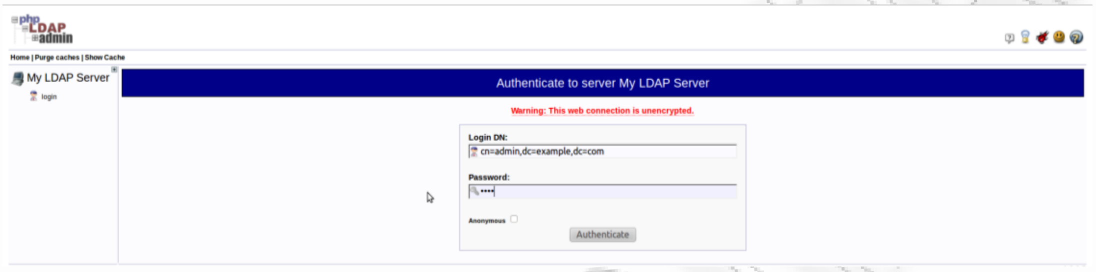
Now we can open a browser and go to **localhost/phpldapadmin/** in order to manage our ldap database

How to create user user1

We will create user1 and we will configure our LDAP database in order to make a successful authentication.

Open a browser and locate it to localhost/phpldapadmin

Click on login and use the password you used during the ldap server installation


Home | Purge caches | Show Cache

My LDAP Server

login

Authenticate to server My LDAP Server

Warning: This web connection is unencrypted.

Login DN: cn=admin,dc=example,dc=com

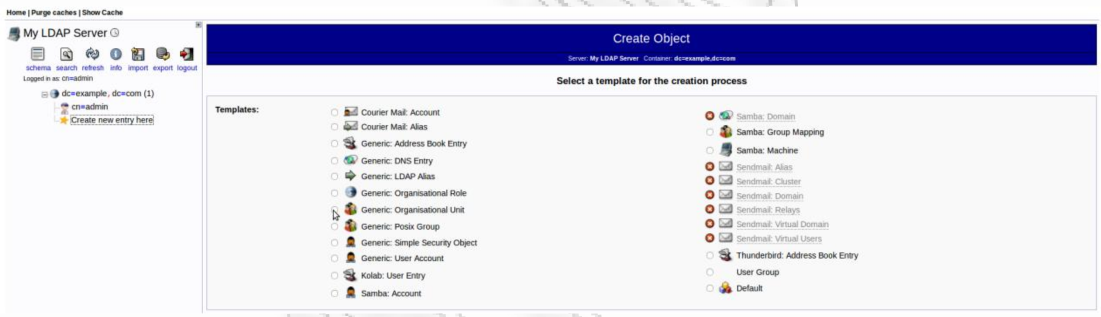
Password: ••••

Anonymous

Authenticate

Picture 45: Login to LDAP Directory

Click on 'Create new entry' and then select Generic: Organizational Unit


Home | Purge caches | Show Cache

My LDAP Server

schema search refresh info import export logout

Logged in as: cn=admin

dc=example, dc=com (1)

cn=admin

Create new entry here

Create Object

Server: My LDAP Server Container: dc=example,dc=com

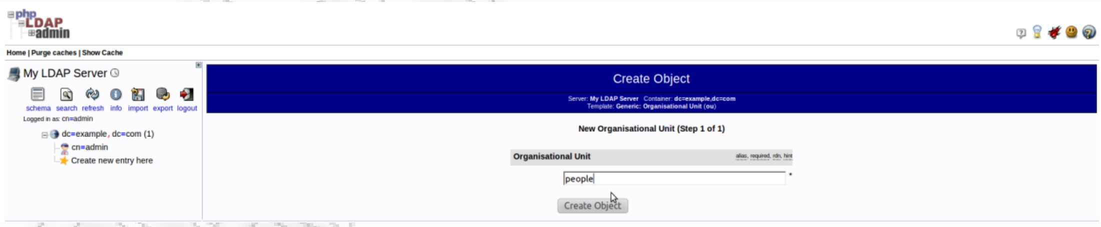
Select a template for the creation process

Templates:

- Courier Mail: Account
- Courier Mail: Alias
- Generic: Address Book Entry
- Generic: DNS Entry
- Generic: LDAP Alias
- Generic: Organisational Role
- Generic: Organisational Unit
- Generic: Posix Group
- Generic: Simple Security Object
- Generic: User Account
- Kolab: User Entry
- Samba: Account
- Samba: Domain
- Samba: Group Mapping
- Samba: Machine
- Sendmail: Alias
- Sendmail: Cluster
- Sendmail: Domain
- Sendmail: Relays
- Sendmail: Virtual Domain
- Sendmail: Virtual Users
- Thunderbird: Address Book Entry
- User Group
- Default

Picture 46: Creating new entry

Name the new organizational unit people and click 'Create Object'


Home | Purge caches | Show Cache

My LDAP Server

schema search refresh info import export logout

Logged in as: cn=admin

dc=example, dc=com (1)

cn=admin

Create new entry here

Create Object

Server: My LDAP Server Container: dc=example,dc=com

Template: Generic: Organisational Unit (ou)

New Organisational Unit (Step 1 of 1)

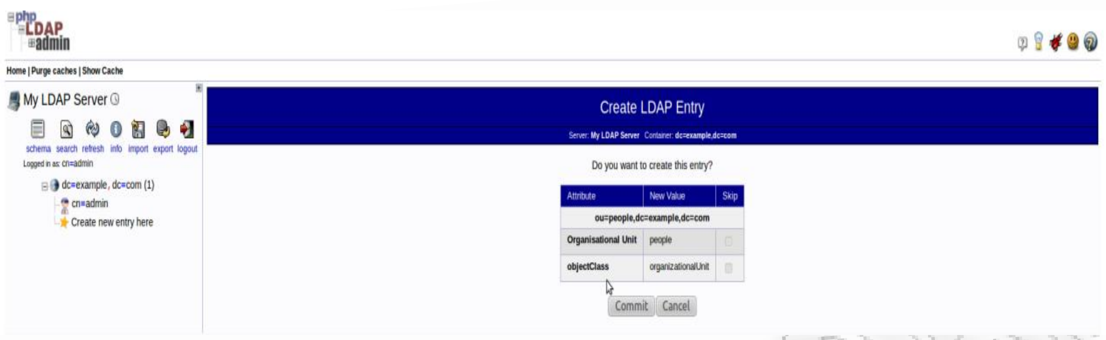
Organisational Unit: people *

alias, required, rdn, hint

Create Object

Picture 47: Creating a new object

Click on 'Commit'



Create LDAP Entry

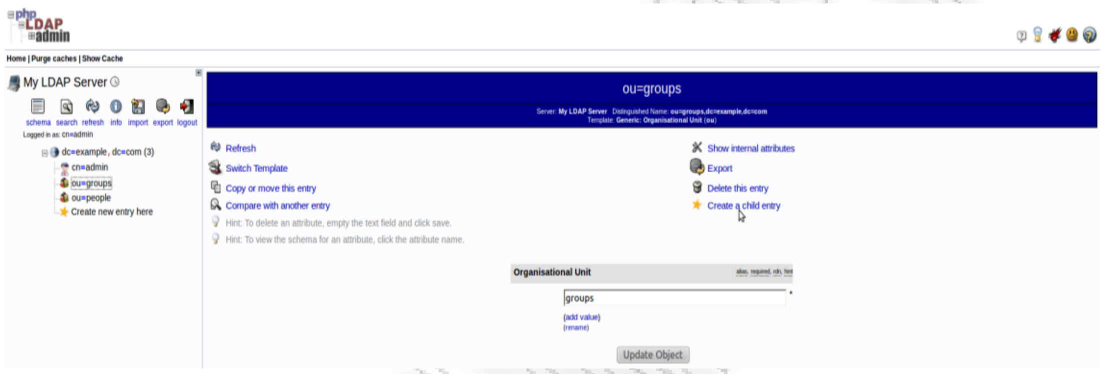
Server: My LDAP Server Container: dc=example,dc=com

Do you want to create this entry?

Attribute	New Value	Skip
ou=people,dc=example,dc=com		
Organisational Unit	people	
objectClass	organizationalUnit	

Picture 48: Creating LDAP entry

Then we create by following the same procedure another organizational unit called 'groups'.



ou=groups

Server: My LDAP Server Distinguished Name: ou=groups,dc=example,dc=com
Template: Generic: Organisational Unit (ou)

Hint: To delete an attribute, empty the text field and click save.

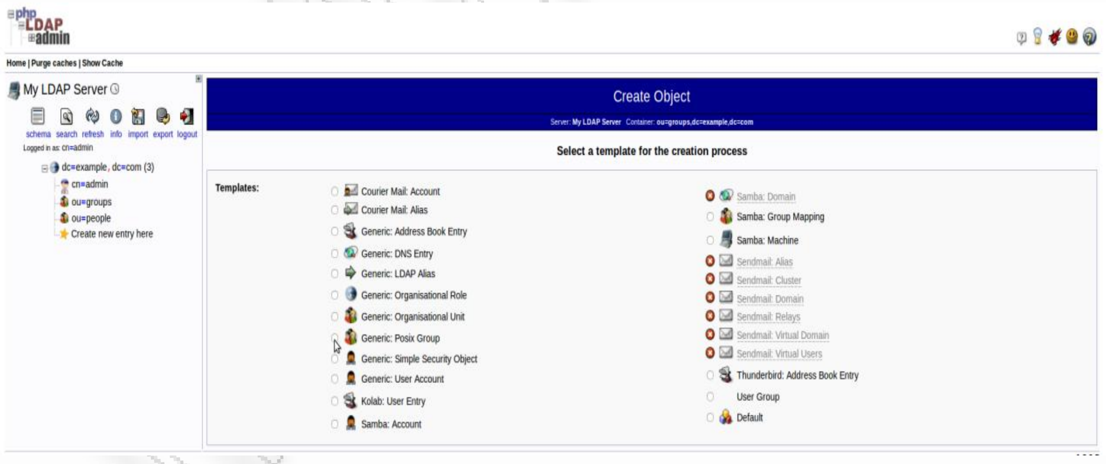
Hint: To view the schema for an attribute, click the attribute name.

Organisational Unit: groups

Picture 49: Creating an organizational unit

We click on ou=groups and select 'create a child entry'

Then we select 'generic : Posix Group'



Create Object

Server: My LDAP Server Container: ou=groups,dc=example,dc=com

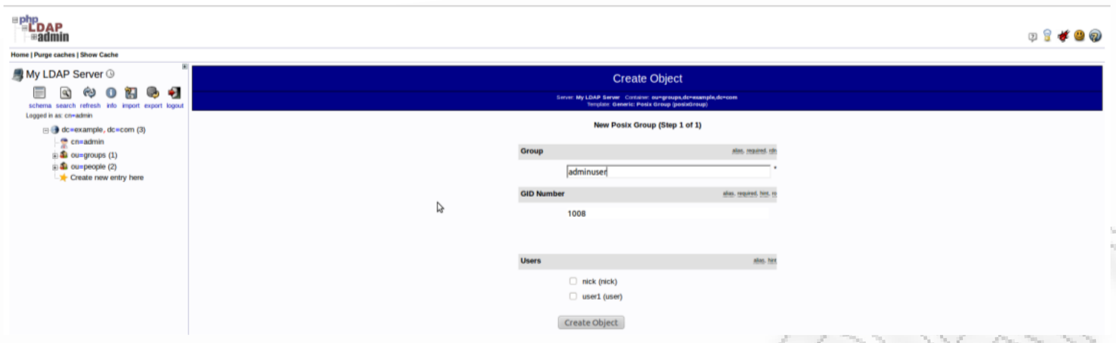
Select a template for the creation process

Templates:

- Courier Mail: Account
- Courier Mail: Alias
- Generic: Address Book Entry
- Generic: DNS Entry
- Generic: LDAP Alias
- Generic: Organisational Role
- Generic: Organisational Unit
- Generic: Posix Group
- Generic: Simple Security Object
- Generic: User Account
- Kolab: User Entry
- Samba: Account
- Samba: Domain
- Samba: Group Mapping
- Samba: Machine
- Sendmail: Alias
- Sendmail: Cluster
- Sendmail: Domain
- Sendmail: Relays
- Sendmail: Virtual Domain
- Sendmail: Virtual Users
- Thunderbird: Address Book Entry
- User Group
- Default

Picture 50: Generic Posix Group

We name our Posix Group adminuser


Home | Purge caches | Show Cache

My LDAP Server

schema search refresh info import export logout

Logged in as: cn=admin

dc=example, dc=com (3)
cn=admin
ou=groups (1)
ou=people (2)
Create new entry here

Create Object

Server: My LDAP Server Container: ou=groups,dc=example,dc=com
Template: Generic: Posix Group (posixGroup)

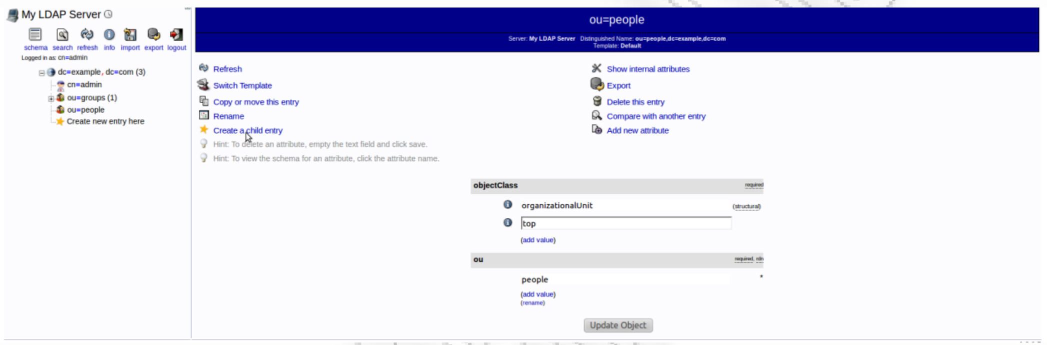
New Posix Group (Step 1 of 1)

Group	adminuser
GID Number	1008
Users	nick (nick) user1 (user)

Create Object

Picture 51: Posix Group adminuser

Then we will create a user under the people organizational unit. Click on people and then click on ‘Create a new child entry’


My LDAP Server

schema search refresh info import export logout

Logged in as: cn=admin

dc=example, dc=com (3)
cn=admin
ou=groups (1)
ou=people
Create new entry here

ou=people

Server: My LDAP Server Distinguished Name: ou=people,dc=example,dc=com
Template: Default

Refresh
Switch Template
Copy or move this entry
Rename
Create a child entry
Hint: To delete an attribute, empty the text field and click save.
Hint: To view the schema for an attribute, click the attribute name.

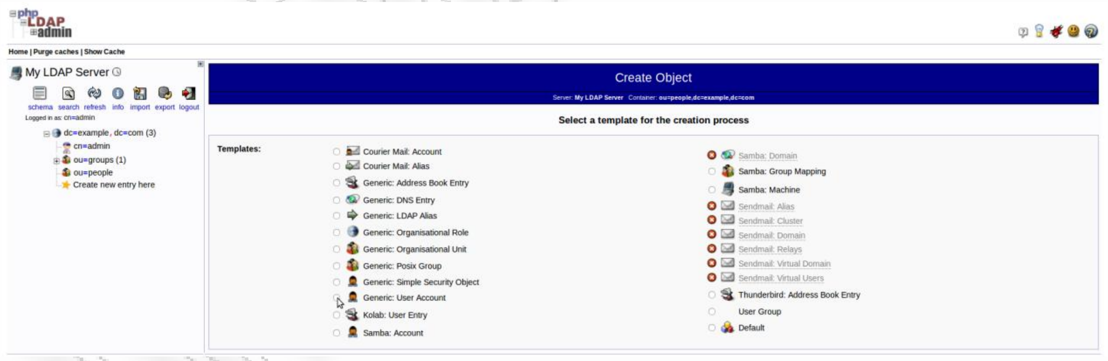
Show internal attributes
Export
Delete this entry
Compare with another entry
Add new attribute

objectClass	organizationalUnit (structural) top (add value)
ou	people (add value) (rename)

Update Object

Picture 52: Creating new child entry

We select to create a new user account


Home | Purge caches | Show Cache

My LDAP Server

schema search refresh info import export logout

Logged in as: cn=admin

dc=example, dc=com (3)
cn=admin
ou=groups (1)
ou=people
Create new entry here

Create Object

Server: My LDAP Server Container: ou=people,dc=example,dc=com

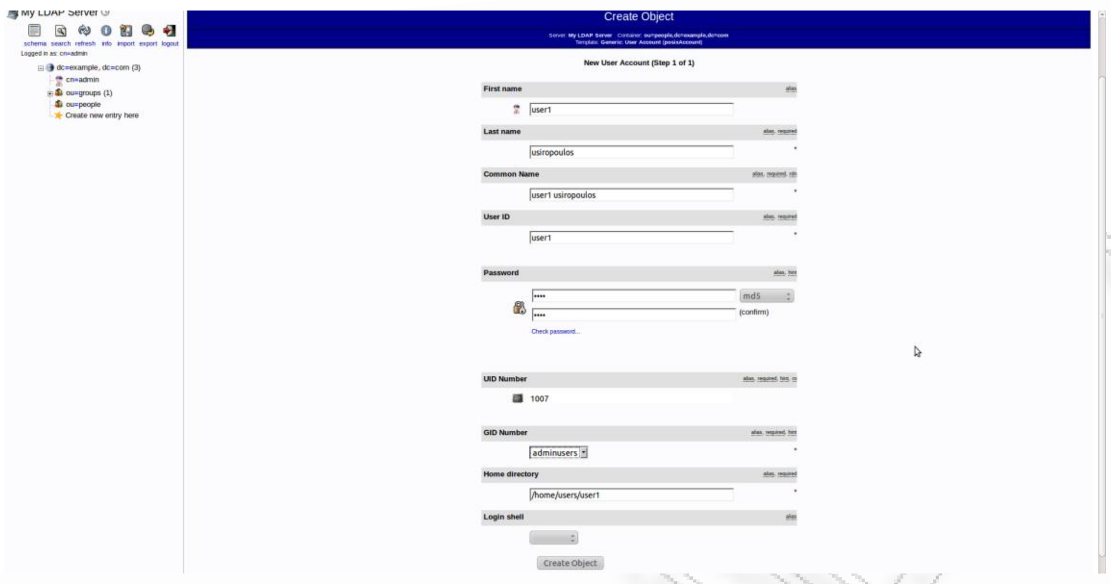
Select a template for the creation process

Templates:

Courier Mail: Account	Samba: Domain
Courier Mail: Alias	Samba: Group Mapping
Generic: Address Book Entry	Samba: Machine
Generic: DNS Entry	Sendmail: Alias
Generic: LDAP Alias	Sendmail: Cluster
Generic: Organisational Role	Sendmail: Domain
Generic: Organisational Unit	Sendmail: Relays
Generic: Posix Group	Sendmail: Virtual Domain
Generic: Simple Security Object	Sendmail: Virtual Users
Generic: User Account	Thunderbird: Address Book Entry
Kolab: User Entry	User Group
Samba: Account	Default

Picture 53: Create new user account

We create our user by filling the above form as we can see here and click ‘Create Object’


Create Object

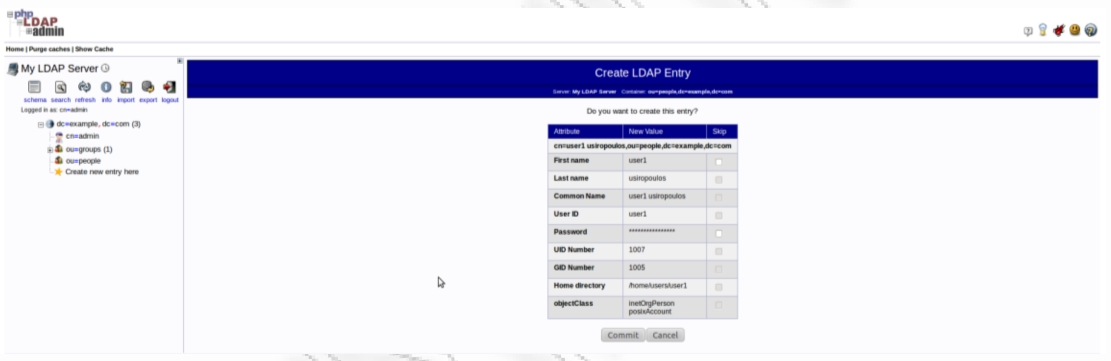
Server: My LDAP Server Container: ou=people,dc=example,dc=com
Template: Generic: User Account (posixAccount)

New User Account (Step 1 of 1)

First name: user1
Last name: usiropoulos
Common Name: user1 usiropoulos
User ID: user1
Password: **** md5
**** (confirm)
Check password...
UID Number: 1007
GID Number: adminusers
Home directory: /home/users/user1
Login shell:

Picture 54: Creating user1

Simply if we agree with our user that we just created we press ‘Commit’.


Create LDAP Entry

Server: My LDAP Server Container: ou=people,dc=example,dc=com

Do you want to create this entry?

Attribute	New Value	Skip
cn=user1 usiropoulos,ou=people,dc=example,dc=com		
First name	user1	☐
Last name	usiropoulos	☐
Common Name	user1 usiropoulos	☐
User ID	user1	☐
Password	****************	☐
UID Number	1007	☐
GID Number	1005	☐
Home directory	/home/users/user1	☐
objectClass	inetOrgPerson posixAccount	☐

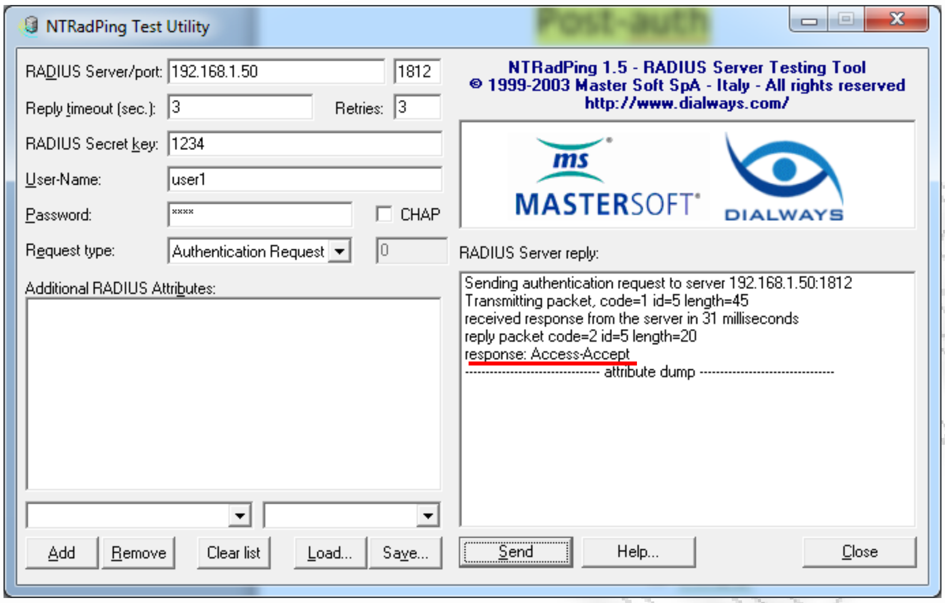
Picture 55: Confirm user1

Using NTRadping to test if all the configuration is correct

We use NTRadping tool on our windows OS and send an authentication request to test once more if our radius server works, only this time we expect to be successfully authenticated.

We use the ip address of our radius server and the authentication port 1812, we use the Radius Secret key we have entered for out windows (radping) client and for User-Name user1 and password 1234.

The proof of concept, as we can see in the screenshot above, is the response: Access Accept.



RADIUS Server/port: 192.168.1.50 1812
Reply timeout (sec.): 3 Retries: 3
RADIUS Secret key: 1234
User-Name: user1
Password: **** CHAP
Request type: Authentication Request 0
Additional RADIUS Attributes:

NTRadPing 1.5 - RADIUS Server Testing Tool
© 1999-2003 Master Soft SpA - Italy - All rights reserved
http://www.dialways.com/

RADIUS Server reply:

```
Sending authentication request to server 192.168.1.50:1812
Transmitting packet, code=1 id=5 length=45
received response from the server in 31 milliseconds
reply packet code=2 id=5 length=20
response: Access-Accept
---------------------- attribute dump ----------------------
```

Picture 56:Access Accept with NTRadtest Utility

And on our terminal on our Ubuntu OS we will see the above response of our freeradius-ldap server (running in debug mode).

rad_recv: Access-Request packet from host 192.168.1.2 port 63051, id=5, length=45

User-Name = "user1"

User-Password = "1234"

# Executing section authorize from file /etc/freeradius/sites-enabled/default

+- entering group authorize {...}

++[preprocess] returns ok

++[chap] returns noop

++[mschap] returns noop

++[digest] returns noop

[suffix] No '@' in User-Name = "user1", looking up realm NULL

[suffix] No such realm "NULL"

++[suffix] returns noop

[eap] No EAP-Message, not doing EAP

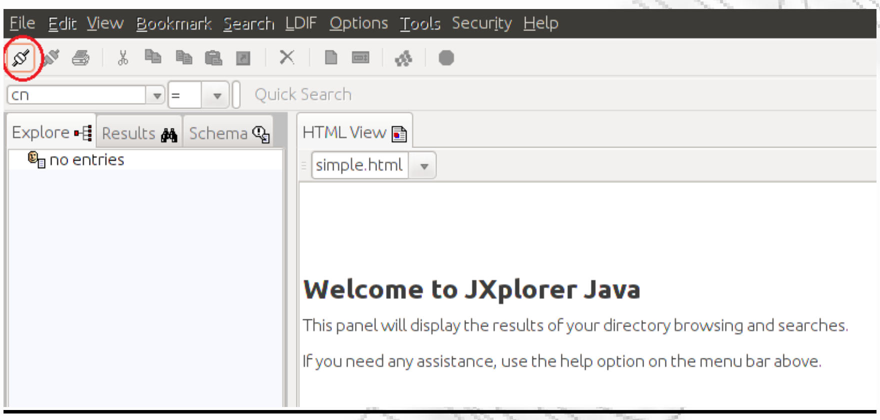
```
++[eap] returns noop
[ldap] performing user authorization for user1
[ldap]  expand: %{Stripped-User-Name} ->
[ldap]  ... expanding second conditional
[ldap]  expand: %{User-Name} -> user1
[ldap]  expand: (uid=%{%{Stripped-User-Name}:-%{User-Name}}) -> (uid=user1)
[ldap]  expand: dc=example, dc=com -> dc=example, dc=com
  [ldap] ldap_get_conn: Checking Id: 0
  [ldap] ldap_get_conn: Got Id: 0
  [ldap] attempting LDAP reconnection
  [ldap] (re)connect to 192.168.1.50:389, authentication 0
  [ldap] bind as cn=admin, dc=example, dc=com/1234 to 192.168.1.50:389
  [ldap] waiting for bind result ...
  [ldap] Bind was successful
  [ldap] performing search in dc=example, dc=com, with filter (uid=user1)
[ldap] No default NMAS login sequence
[ldap] looking for check items in directory...
  [ldap]          userPassword          ->          Password-With-Header          ==
"{MD5}gdyb21LQTcIANtvYMT7QVQ=="
[ldap] looking for reply items in directory...
[ldap] Setting Auth-Type = LDAP
[ldap] user user1 authorized to use remote access
  [ldap] ldap_release_conn: Release Id: 0
++[ldap] returns ok
++[expiration] returns noop
++[logintime] returns noop
[pap] Normalizing MD5-Password from base64 encoding
```

```
[pap] WARNING: Auth-Type already set.  Not setting to PAP
++[pap] returns noop
Found Auth-Type = LDAP
# Executing group from file /etc/freeradius/sites-enabled/default
+- entering group LDAP {...}
[ldap] login attempt by "user1" with password "1234"
[ldap] user DN: cn=user1,ou=nikos,dc=example,dc=com
  [ldap] (re)connect to 192.168.1.50:389, authentication 1
  [ldap] bind as cn=user1,ou=nikos,dc=example,dc=com/1234 to 192.168.1.50:389
  [ldap] waiting for bind result ...
  [ldap] Bind was successful
[ldap] user user1 authenticated succesfully
++[ldap] returns ok
Login OK: [user1] (from client radping port 0)
# Executing section post-auth from file /etc/freeradius/sites-enabled/default
+- entering group post-auth {...}
++[exec] returns noop
Sending Access-Accept of id 5 to 192.168.1.2 port 63051
Finished request 0.
Going to the next request
Waking up in 4.9 seconds.
Cleaning up request 0 ID 5 with timestamp +11
Ready to process requests.
```

Creating Users by using JXplorer tool.

If we don’t want to use phpldapadmin to configure our ldap database there is another tool called JXplorer which can be easily downloaded and installed by using a package manager or the Dash Home tool on our Ubuntu 11.10.

Once you download this application and load it you should click on the upper left symbol of bond as you can see above.



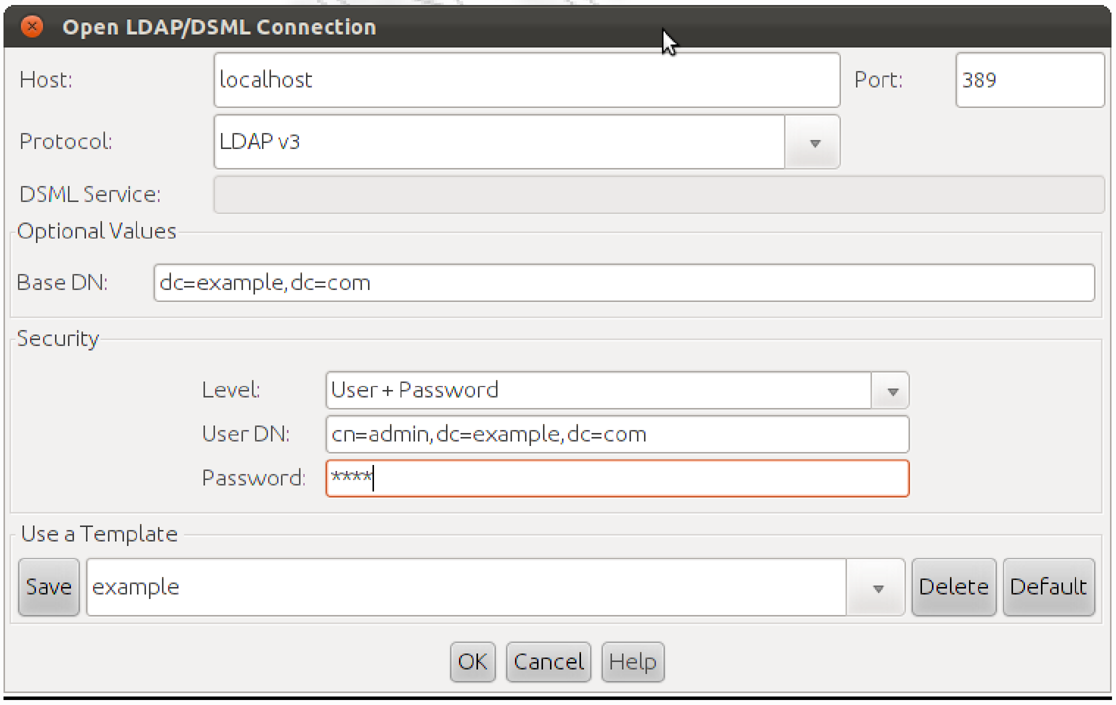
Welcome to JXplorer Java

This panel will display the results of your directory browsing and searches.

If you need any assistance, use the help option on the menu bar above.

Picture 57: The JXplorer Tool

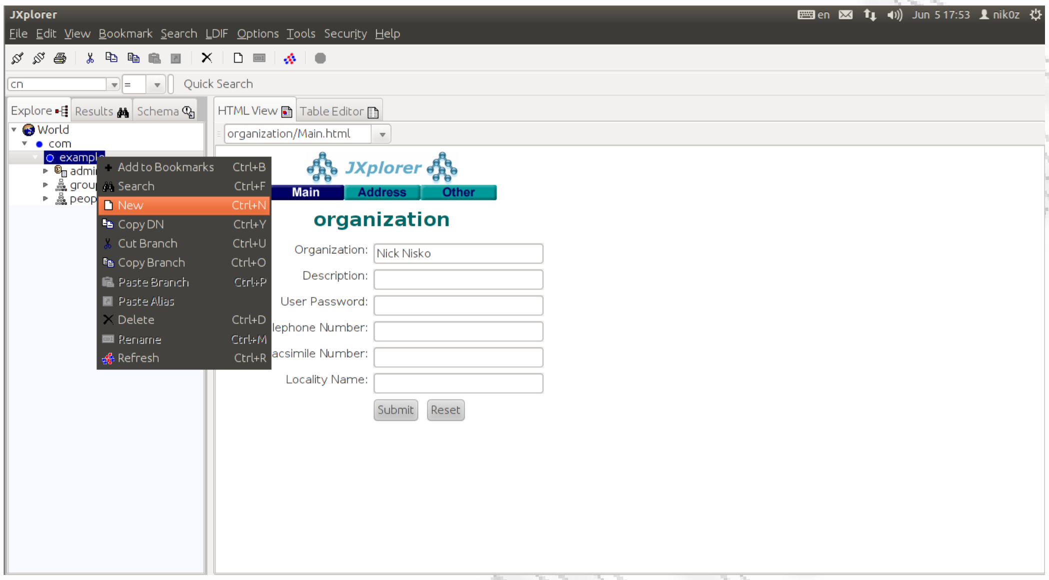
Then we should fill which ldap directory we want to connect to and we will select the database we created during the installation and configured before with phpldapadmin. We insert the following info to the new window that will come up as in the screenshot that follows. The password is the same with the one used before (1234).



Host: localhost
Port: 389
Protocol: LDAP v3
DSML Service:
Base DN: dc=example,dc=com
Level: User + Password
User DN: cn=admin,dc=example,dc=com
Password: ****
Template: example

Picture 58: Selecting LDAP directory

When our database is loaded we can see on the list the organizational users we created with phpldapadmin (this is a proof that the database functions properly). Then we right click on example and select New.



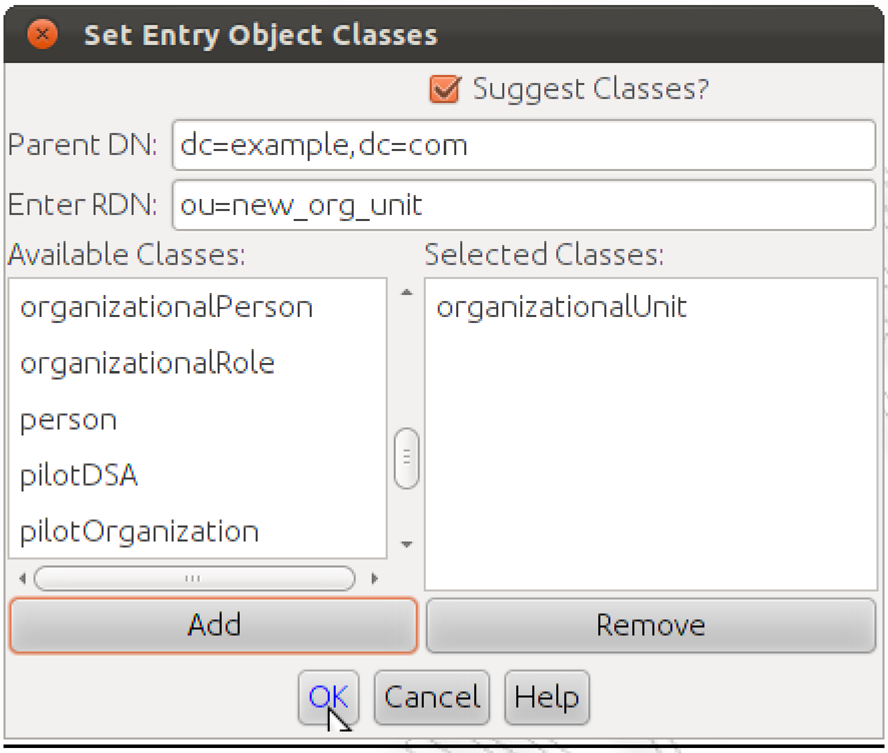
organization/Main.html

organization

Organization:	Nick Nisko
Description:	
User Password:	
lephone Number:	
acsimile Number:	
Locality Name:	

Picture 59: Creating new oragnizational unit

Now we can select what object we want to insert. We will create an organizational unit and name it new_org_unit (ou=new_org_unit). The parent is already set to be the ‘example.com’. From the available classes we will select ‘organizationalUnit’ and we will ‘Add’ it to our ‘Seleced Classes’ (if any unwanted Classes are preselected we can easily select and remove them) as we can see in the screenshot.


Set Entry Object Classes

Suggest Classes? [✓]

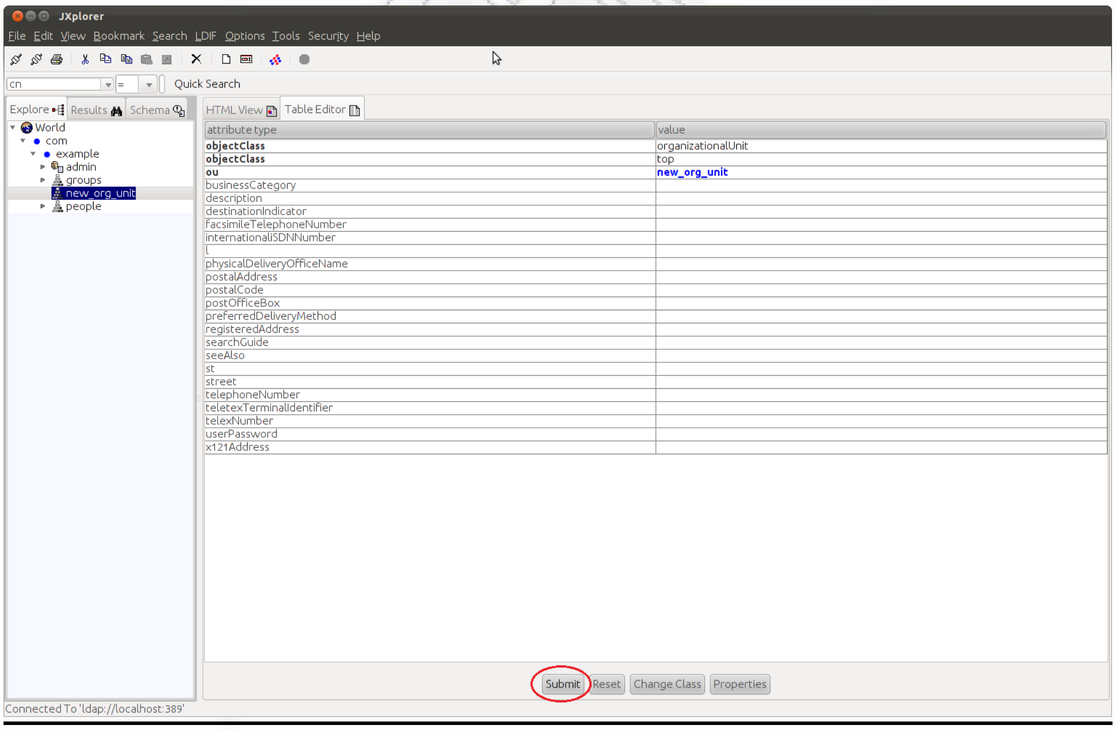
Parent DN: dc=example,dc=com

Enter RDN: ou=new_org_unit

Available Classes:	Selected Classes:
organizationalPerson	organizationalUnit
organizationalRole	
person	
pilotDSA	
pilotOrganization	

Picture 60: Configuring new oragnizational unit

If we agree with the organizational unit that we just created, we simply press ‘Submit’.


JXplorer

cn = Quick Search

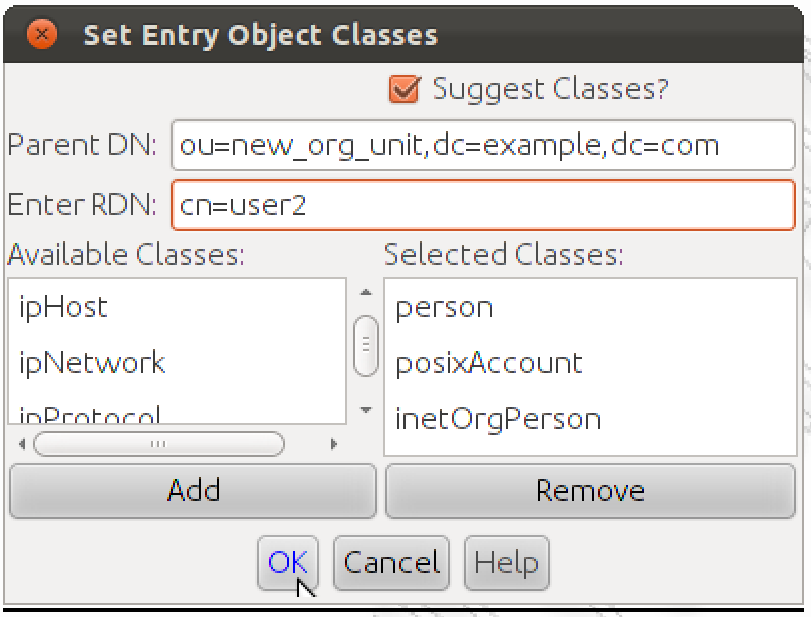
World > com > example > admin, groups, new_org_unit, people

attribute type	value
objectClass	organizationalUnit
objectClass	top
ou	**new_org_unit**
businessCategory	
description	
destinationIndicator	
facsimileTelephoneNumber	
internationaliSDNNumber	
l	
physicalDeliveryOfficeName	
postalAddress	
postalCode	
postOfficeBox	
preferredDeliveryMethod	
registeredAddress	
searchGuide	
seeAlso	
st	
street	
telephoneNumber	
teletexTerminalIdentifier	
telexNumber	
userPassword	
x121Address	

Connected To 'ldap://localhost:389'

Picture 61: Submit

Then as we did before in order to create the organizational unit, we right click on ‘new_org_unit’ and select ‘New’. We fill the ‘Enter RDN’ with name of the user we want to create ‘cn=user2’ and we select the classes inetOrgPerson, Person, posixAccount and click OK.



Set Entry Object Classes

[✓] Suggest Classes?

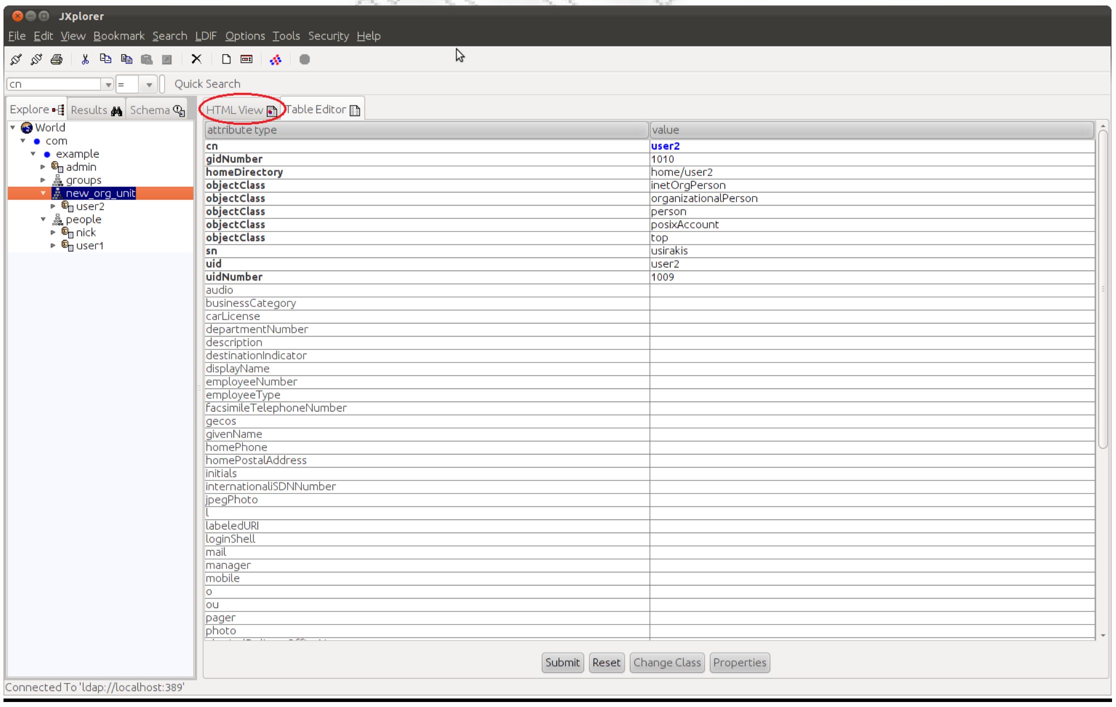
Parent DN: ou=new_org_unit,dc=example,dc=com

Enter RDN: cn=user2

Available Classes:	Selected Classes:
ipHost	person
ipNetwork	posixAccount
ipProtocol	inetOrgPerson

Picture 62: Creating the new user

We fill the gidNumber, homeDirectory, sn, uid and uidNumber attributes as in the following picture and then Submit and change to the ‘HTML View’ tab.

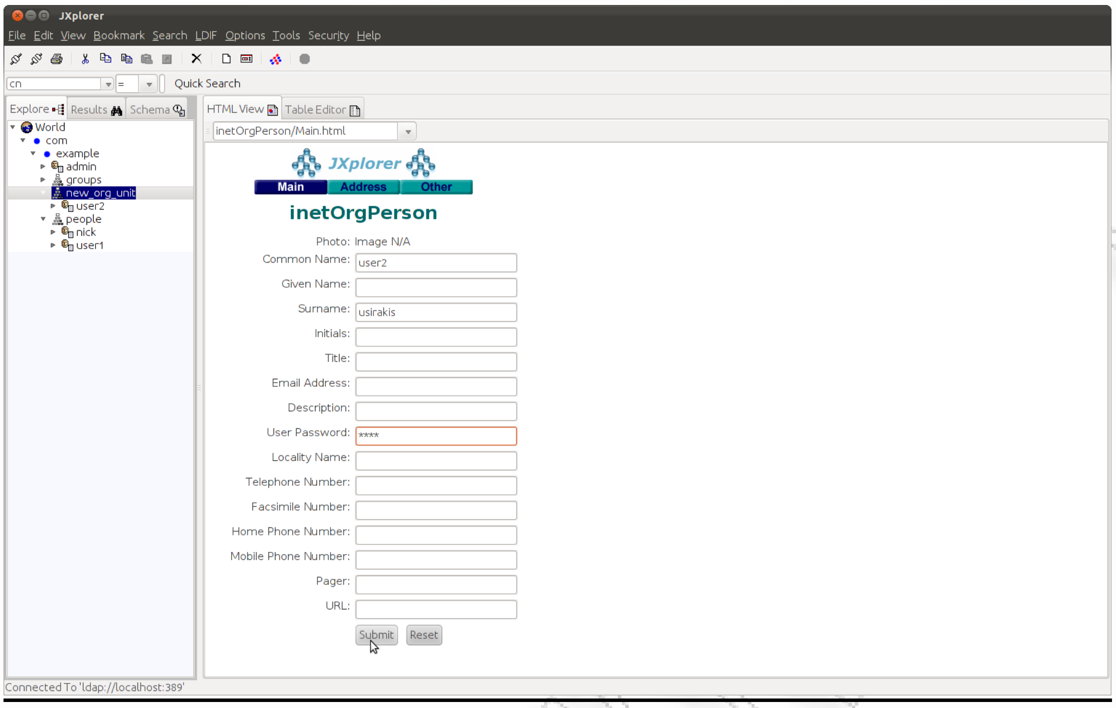


attribute type	value
cn	user2
gidNumber	1010
homeDirectory	home/user2
objectClass	inetOrgPerson
objectClass	organizationalPerson
objectClass	person
objectClass	posixAccount
objectClass	top
sn	usirakis
uid	user2
uidNumber	1009
audio	
businessCategory	
carLicense	
departmentNumber	
description	
destinationIndicator	
displayName	
employeeNumber	
employeeType	
facsimileTelephoneNumber	
gecos	
givenName	
homePhone	
homePostalAddress	
initials	
internationaliSDNNumber	
jpegPhoto	
l	
labeledURI	
loginShell	
mail	
manager	
mobile	
o	
ou	
pager	
photo	

Connected To 'ldap://localhost:389'

Picture 63: Creating the new user

We fill the user password section with the desired Password (in this case 4321) click ‘Submit’ and Return to the ‘Table Editor’ tab.



inetOrgPerson/Main.html

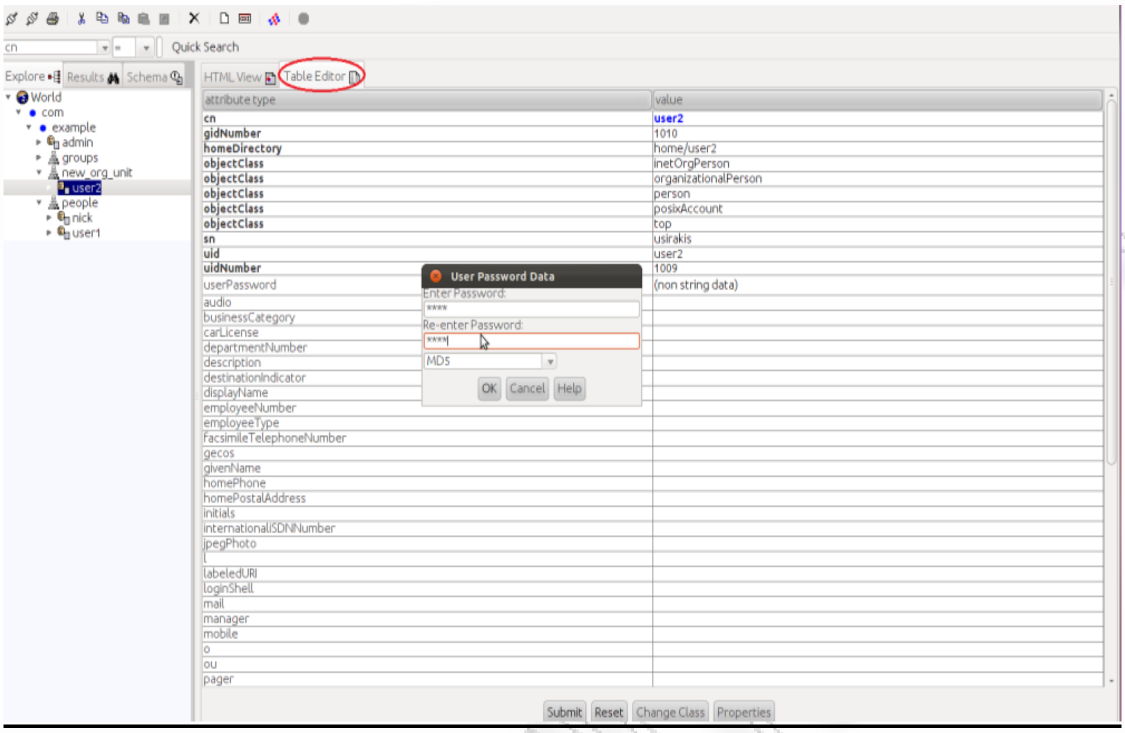
inetOrgPerson

Photo:	Image N/A
Common Name:	user2
Given Name:	
Surname:	usirakis
Initials:	
Title:	
Email Address:	
Description:	
User Password:	****
Locality Name:	
Telephone Number:	
Facsimile Number:	
Home Phone Number:	
Mobile Phone Number:	
Pager:	
URL:	

Connected To 'ldap://localhost:389'

Picture 64: Entrering Password to user

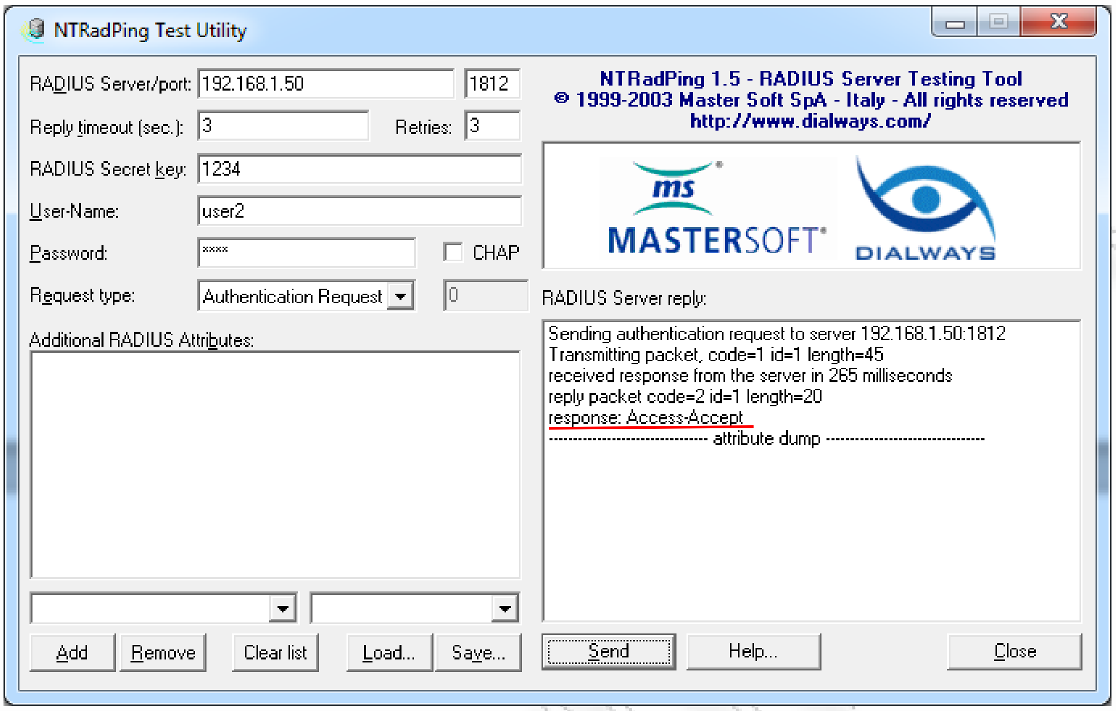
Since we are back to Table Editor tab click on (no string data) in the userPassword Atribute. Fill in the password and the encryption algorithm we desire and then 'OK'.



attribute type	value
cn	user2
gidNumber	1010
homeDirectory	home/user2
objectClass	inetOrgPerson
objectClass	organizationalPerson
objectClass	person
objectClass	posixAccount
objectClass	top
sn	usirakis
uid	user2
uidNumber	1009
userPassword	(non string data)
audio	
businessCategory	
carLicense	
departmentNumber	
description	
destinationIndicator	
displayName	
employeeNumber	
employeeType	
facsimileTelephoneNumber	
gecos	
givenName	
homePhone	
homePostalAddress	
initials	
internationaliSDNNumber	
jpegPhoto	
l	
labeledURI	
loginShell	
mail	
manager	
mobile	
o	
ou	
pager	

Picture 65: Confirm Password

Now that we have created another user, we should check if we can be authenticated to the freeradius-ldap server with these new credentials. We open the NTRadping Test Utility on our Windows OS and as we did before we set the ip of the Radius server and the port 1812 (which is the default port for authentication), the radius server key that we have used during the installation for our windows client, the username of the user that we have just created and his password (4321) click next and the proof of concept reveals, 'response: Access Accept'. We can see the response in the following screenshot.


NTRadPing Test Utility
NTRadPing 1.5 - RADIUS Server Testing Tool
© 1999-2003 Master Soft SpA - Italy - All rights reserved
http://www.dialways.com/
Add
Remove
Clear list
Load...
Save...
Send
Help...
Close

RADIUS Server/port: 192.168.1.50 1812
Reply timeout (sec.): 3 Retries: 3
RADIUS Secret key: 1234
User-Name: user2
Password: **** CHAP
Request type: Authentication Request 0
Additional RADIUS Attributes:

RADIUS Server reply:

```
Sending authentication request to server 192.168.1.50:1812
Transmitting packet, code=1 id=1 length=45
received response from the server in 265 milliseconds
reply packet code=2 id=1 length=20
response: Access-Accept
------------------------------ attribute dump ------------------------------
```

Picture 66: NTRadPing Test Utility

Our freeradius-ldap server running in debug mode has the following output:

Ready to process requests.

rad_recv: Access-Request packet from host 192.168.1.2 port 63121, id=1, length=45

User-Name = "user2"

User-Password = "4321"

# Executing section authorize from file /etc/freeradius/sites-enabled/default

+- entering group authorize {...}

++[preprocess] returns ok

++[chap] returns noop

++[mschap] returns noop

++[digest] returns noop

[suffix] No '@' in User-Name = "user2", looking up realm NULL

[suffix] No such realm "NULL"

++[suffix] returns noop

```
[eap] No EAP-Message, not doing EAP
++[eap] returns noop
[ldap] performing user authorization for user2
[ldap]  expand: %{Stripped-User-Name} ->
[ldap]  ... expanding second conditional
[ldap]  expand: %{User-Name} -> user2
[ldap]  expand: (uid=%{%{Stripped-User-Name}:-%{User-Name}}) -> (uid=user2)
[ldap]  expand: dc=example, dc=com -> dc=example, dc=com
  [ldap] ldap_get_conn: Checking Id: 0
  [ldap] ldap_get_conn: Got Id: 0
  [ldap] attempting LDAP reconnection
  [ldap] (re)connect to 192.168.1.50:389, authentication 0
  [ldap] bind as cn=admin, dc=example, dc=com/1234 to 192.168.1.50:389
  [ldap] waiting for bind result ...
  [ldap] Bind was successful
  [ldap] performing search in dc=example, dc=com, with filter (uid=user2)
[ldap] No default NMAS login sequence
[ldap] looking for check items in directory...
  [ldap] userPassword -> Password-With-Header == "{MD5}2TWRvfeGDh5O4vynmZESFQ=="
[ldap] looking for reply items in directory...
[ldap] Setting Auth-Type = LDAP
[ldap] user user2 authorized to use remote access
  [ldap] ldap_release_conn: Release Id: 0
++[ldap] returns ok
++[expiration] returns noop
++[logintime] returns noop
```

```
[pap] Normalizing MD5-Password from base64 encoding
[pap] WARNING: Auth-Type already set.  Not setting to PAP
++[pap] returns noop
Found Auth-Type = LDAP
# Executing group from file /etc/freeradius/sites-enabled/default
+- entering group LDAP {...}
[ldap] login attempt by "user2" with password "4321"
[ldap] user DN: cn=user2,ou=new_org_unit,dc=example,dc=com
  [ldap] (re)connect to 192.168.1.50:389, authentication 1
  [ldap] bind as cn=user2,ou=new_org_unit,dc=example,dc=com/4321 to
192.168.1.50:389
  [ldap] waiting for bind result ...
  [ldap] Bind was successful
[ldap] user user2 authenticated succesfully
++[ldap] returns ok
Login OK: [user2] (from client radping port 0)
# Executing section post-auth from file /etc/freeradius/sites-enabled/default
+- entering group post-auth {...}
++[exec] returns noop
Sending Access-Accept of id 1 to 192.168.1.2 port 63121
Finished request 0.
Going to the next request
Waking up in 4.9 seconds.
Cleaning up request 0 ID 1 with timestamp +9
Ready to process requests.
```

PAM/USB Module

Introduction

PAM-USB module is an authentication method which uses an external USB flash drive as a token to authenticate a user to a service or to the operating system that he uses. Here, an operating system authentication will be implemented were the OS will be Ubuntu 11.10 and the **pamusb** tool is going to be used.

Installing PAM/USB Module

1) Install the pam usb

We download the zip file (aluzzardi-pam_usb-0.5.0-23-g8d630b9.zip) to our desktop from http://pamusb.org/ and we extract the file into our Desktop (or anywhere we like since we remember were we saved it)

cd /home/youruser/Desktop

unzip aluzzardi-pam_usb-0.5.0-23-g8d630b9.zip

cd aluzzardi-pam_usb-8d630b9

Before we install the source code we have to install all of the above in order to avoid some problems they came up during the installation

sudo apt-get install udisks

sudo apt-get install pmount

sudo apt-get install libxml2

sudo apt-get install libxml2-dev

sudo apt-get install libdbus-1-dev

sudo apt-get install libpam0g-dev

2) We run

cd /home/youruser/Desktop/ aluzzardi-pam_usb-8d630b9 //or wherever you installed //the -aluzzardi-pam_usb- //file you downloaded on //step 1

make

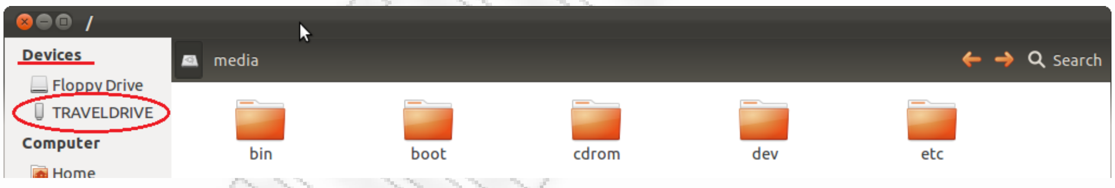
make install // if everything went well the installation should finish without any problems

3) We insert our usb flash drive into a usb port of our computer

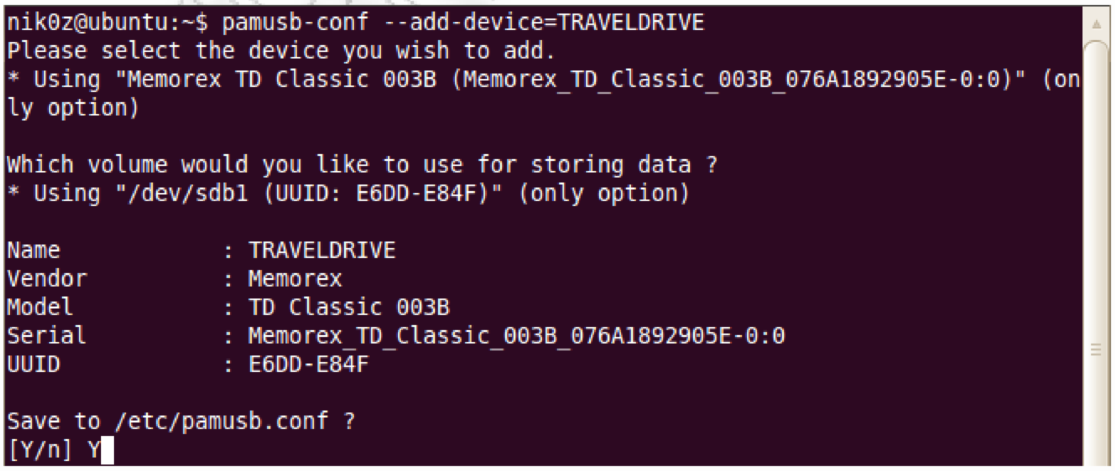
Important! If we use a virtual machine we have to make sure that our device is installed to the guest operating system and not to host. It might need to restart yout OS if the usb device does not appear to your guest OS.

4) We select the usb flash drive that we will use

pamusb-conf --add-device=TRAVELDRIVE //use the exact name of the device //we will use



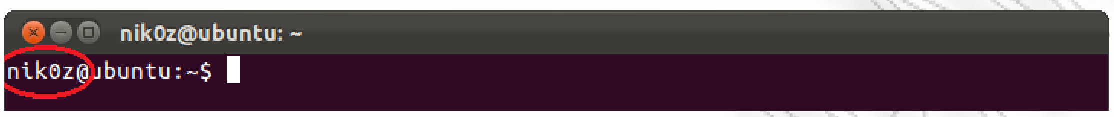
Picture 67:TRAVELDRIVE usb flash drive is inserted to out PC



```
nik0z@ubuntu:~$ pamusb-conf --add-device=TRAVELDRIVE
Please select the device you wish to add.
* Using "Memorex TD Classic 003B (Memorex_TD_Classic_003B_076A1892905E-0:0)" (on
ly option)

Which volume would you like to use for storing data ?
* Using "/dev/sdb1 (UUID: E6DD-E84F)" (only option)

Name            : TRAVELDRIVE
Vendor          : Memorex
Model           : TD Classic 003B
Serial          : Memorex_TD_Classic_003B_076A1892905E-0:0
UUID            : E6DD-E84F

Save to /etc/pamusb.conf ?
[Y/n] Y
```

5) We add a user

pamusb-conf --add-user=nik0z //use the exact name of the user we want to //authenticate the system is case sensitive //Nik0z≠nik0z



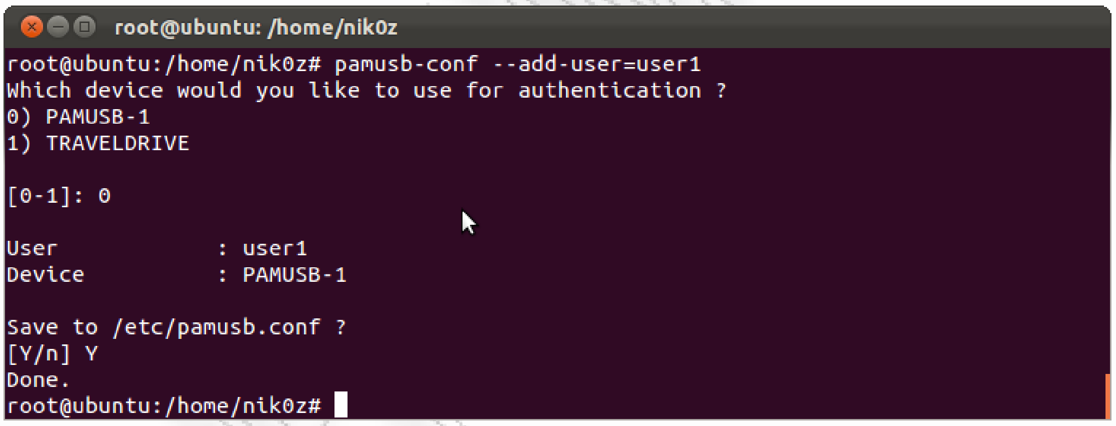
```
nik0z@ubuntu: ~
nik0z@ubuntu:~$
```

Picture 68: OS user nik0z

6) We can add another user if we want to the same device

pamusb-conf --add-user= user1

As we can see in the above screenshot after we run the command we can select in witch device we want to add our user (I have already installed two usb devices TRAVELDRIVE and PAMUSB-1) here we select **0** the flash drive PAMUSB-1 and then we select **Y** in order to write the new information into pamusb.conf file.



```
root@ubuntu: /home/nik0z
root@ubuntu:/home/nik0z# pamusb-conf --add-user=user1
Which device would you like to use for authentication ?
0) PAMUSB-1
1) TRAVELDRIVE

[0-1]: 0

User             : user1
Device           : PAMUSB-1

Save to /etc/pamusb.conf ?
[Y/n] Y
Done.
root@ubuntu:/home/nik0z#
```

Picture 69: Adding a user to device PAMUSB-1

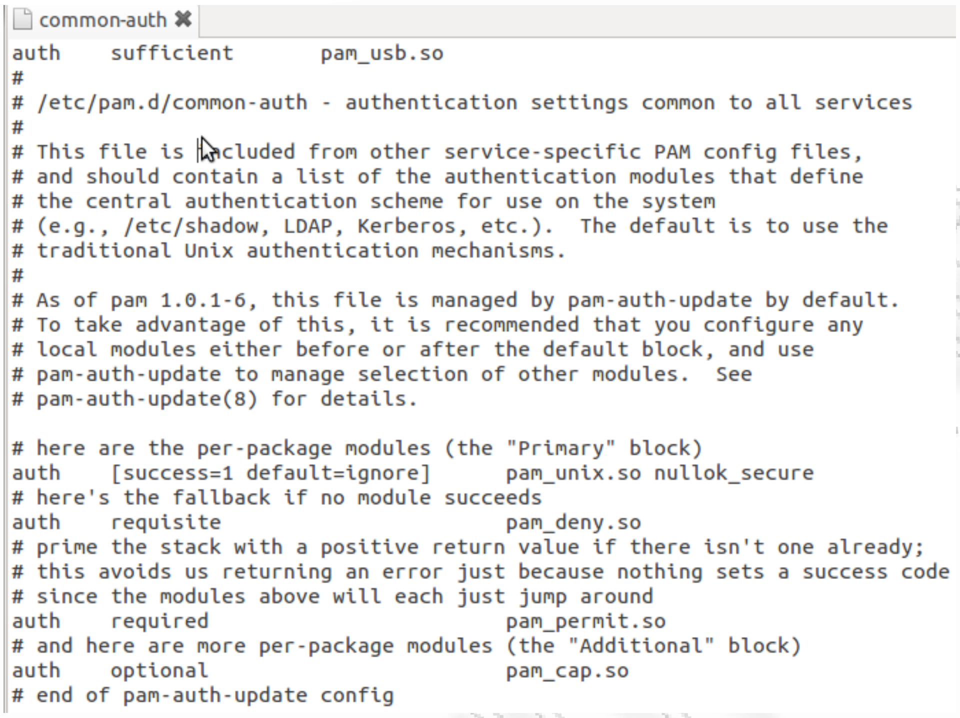
7) Now modify the common-auth file in order to start using the usb flash to authenticate ourselves while entering the system.

gedit /etc/pam.d/common-auth

8) Then we write inside

auth sufficient pam_usb.so

as in the following picture



```
auth    sufficient      pam_usb.so
#
# /etc/pam.d/common-auth - authentication settings common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of the authentication modules that define
# the central authentication scheme for use on the system
# (e.g., /etc/shadow, LDAP, Kerberos, etc.).  The default is to use the
# traditional Unix authentication mechanisms.
#
# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
auth    [success=1 default=ignore]      pam_unix.so nullok_secure
# here's the fallback if no module succeeds
auth    requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
auth    required                        pam_permit.so
# and here are more per-package modules (the "Additional" block)
auth    optional                        pam_cap.so
# end of pam-auth-update config
```

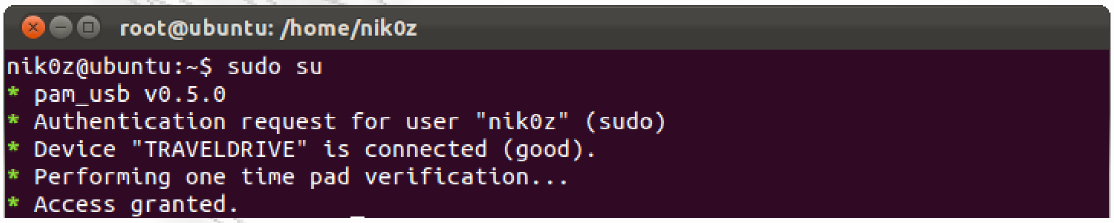
Picture 70: Common-auth Configuration File

9) Check if the usb works

we open a new terminal and simply run the command

sudo su //we can disable this feature from the /etc/pamusb.conf file in the services //section by adding the following

```
<service id="su">
        <!-- Disable pam_usb for "su" ("su" will ask for a password as usual) -->
        <option name="enable">false<option>
    </service>
```

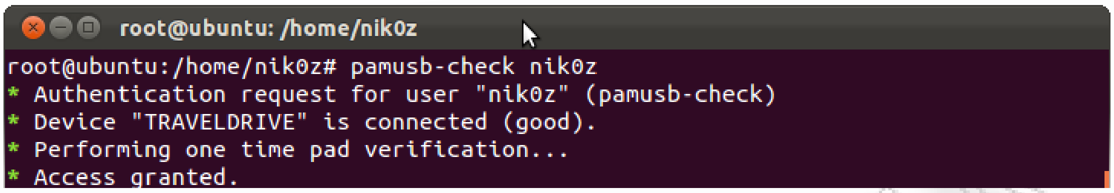


```
nik0z@ubuntu:~$ sudo su
* pam_usb v0.5.0
* Authentication request for user "nik0z" (sudo)
* Device "TRAVELDRIVE" is connected (good).
* Performing one time pad verification...
* Access granted.
```

Picture 71: Successful authentication via usb TRAVELDRIVE

or alternatively we run

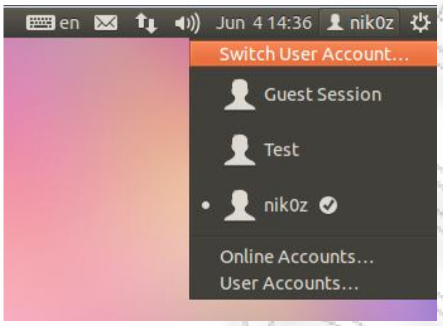
pamusb-check nik0z //instead of nik0z we use whatever username we have created



```
root@ubuntu:/home/nik0z# pamusb-check nik0z
* Authentication request for user "nik0z" (pamusb-check)
* Device "TRAVELDRIVE" is connected (good).
* Performing one time pad verification...
* Access granted.
```

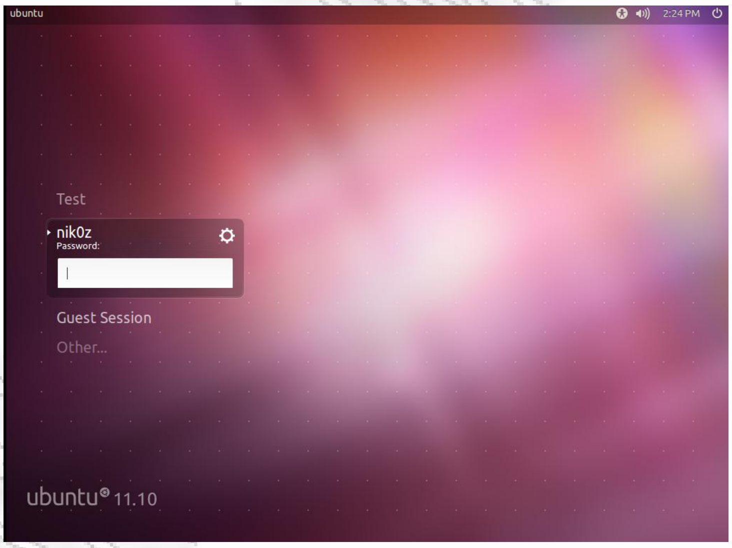
Picture 72: Successful authentication via usb TRAVELDRIVE

10) We remove our usb flash drive and we switch user account.



Picture 73: Switch User Account

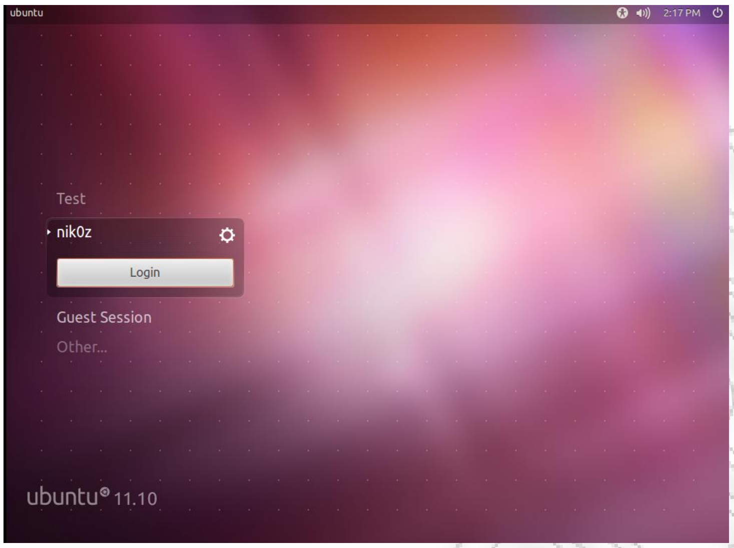
As we can see a password is needed to authenticate user nik0z



Picture 74: Password Request

We insert our usb flash drive that we have used to install user nik0z and wait a couple of seconds.

As we can see it no longer needs password to authenticate user nik0z, we simply press login.



Picture 75: No Password required

Success!

Two-Factor Authentication

If we want to use two-way authentication (Password and USB Flash Drive) we have to make the following change.

1) We simply change the common-auth file

gedit /etc/pam.d/common-auth

2) We write inside

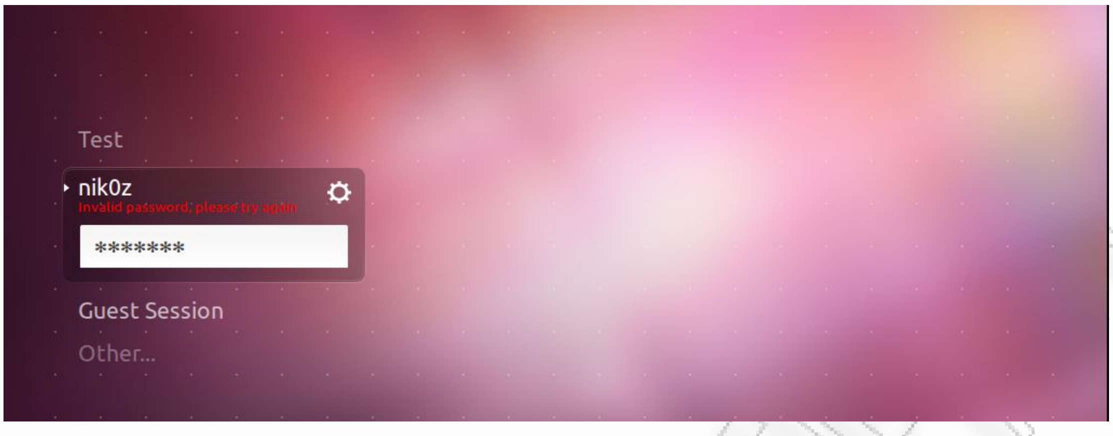
authentication required pam_usb.so

instead of *authentication sufficient pam_usb.so* .

This means that both password and usb flash drive will be needed in order to

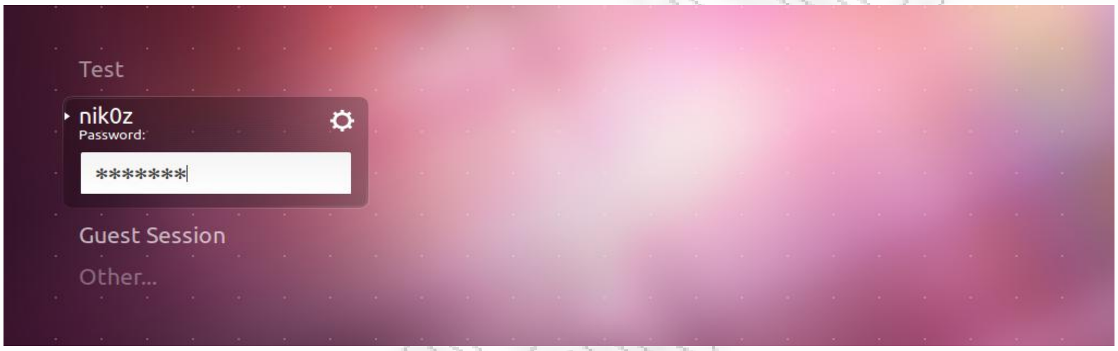
authenticate a user (Two-Factor authentication)

We unplug our flash drive and change user account once again. As we can see in the following picture despite the fact that the correct password is inserted the authentication is not completed.



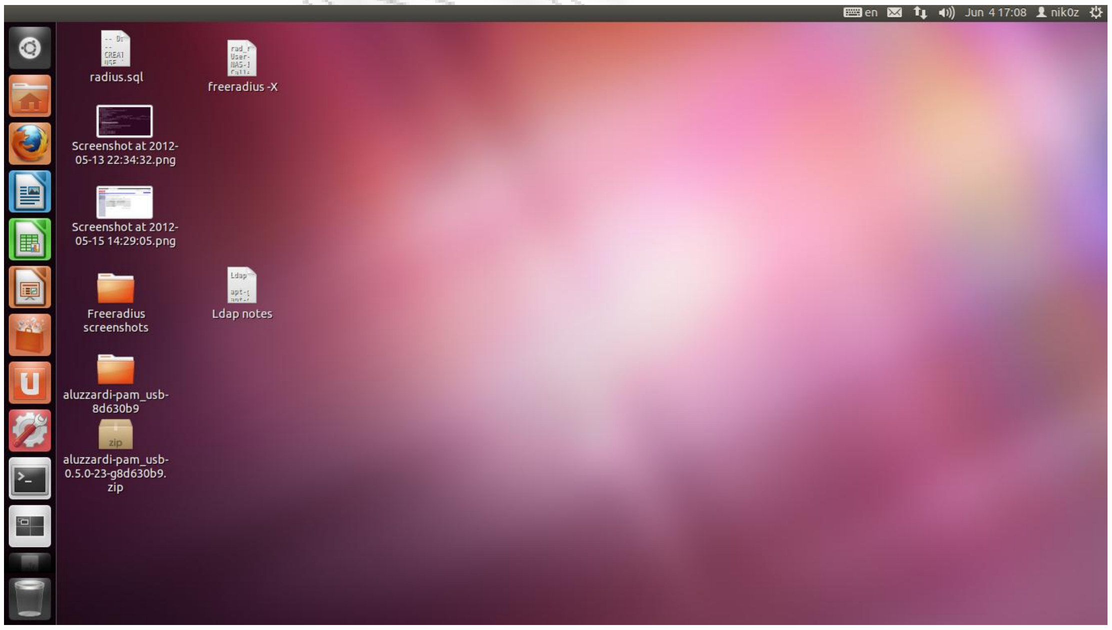
Picture 76: Authentication Failure without usb flash drive

3) We insert our usb drive wait a couple of seconds and type again the correct password



Picture 77: Renter Password while usb is plugged in

Then this time we will enter out OS


en
Jun 4 17:08
nik0z
radius.sql
freeradius -X
Screenshot at 2012-05-13 22:34:32.png
Screenshot at 2012-05-15 14:29:05.png
Freeradius screenshots
Ldap notes
aluzzardi-pam_usb-8d630b9
aluzzardi-pam_usb-0.5.0-23-g8d630b9.zip

Picture 78: Accessing our desktop by using two-factor authentication

Success!

LinOTP tool

Introduction

A **one-time password** (OTP) is a password that is valid for only one login session or transaction. OTPs avoid a number of shortcomings that are associated with traditional (static) passwords. The most important shortcoming that is addressed by OTPs is that, in contrast to static passwords, they are not vulnerable to replay attacks. This means that, if a potential intruder manages to record an OTP that was already used to log into a service or to conduct a transaction, he or she will not be able to abuse it since it will be no longer valid. On the downside, OTPs are difficult for human beings to memorize. Therefore they require additional technology in order to work. LinOTP is an open solution for strong two-factor authentication with One Time Passwords.

This new LinOTP 2 core modules and basic necessary components are licensed under the AGPLv3, so that you we are able to have a complete working open source solution. But LinOTP2 is also open as far as its modular architecture is concerned. LinOTP 2 aims to not bind you to any decision of the authentication protocol or it does not dictate you where your user information should be stored. This is achieved by its new, totally modular architecture.

LinOTP also provides a modular architecture to calculate OTP values. Thus many different OTP algorithms can be supported by LinOTP. LinOTP is originally based on the HMAC-OTP algorithm. This open algorithm is defined in RFC4226. The HMAC-OTP algorithm calculates an OTP value on the basis of a secret key that is shared between the LinOTP server and the user. This secret key is not put plainly in the users hand but is forged into a piece of hardware device - the OTP token. LinOTP supports several different hardware and software OTP tokens.

Installing the linOTP2 tool

Because LinOTP2 doesn't support the latest Ubuntu releases (11.04 and later) some problems might occur during the installation. We open a new terminal and type:

1) In order to get administrative permissions

sudo su

2) We update our repositories

apt-get update

apt-get upgrade

3) and we install all the following prerequisites in order to avoid as many installation bugs as possible.

apt-get install python-simplejson python-crypto python-docutils python-sqlalchemy python-pylons python-tempita python-weberror python-webob python-mako python-nose python-decorator python-formencode python-pastescript python-pastedeploy python-paste python-beaker python-webhelpers python-routes python-pygments python-ldap python-mysqldb python-m2crypto apache2 libapache2-mod-wsgi python-repoze.who pwgen mysql-server

(make sure you remember the mysql root password because we are going to need it later)

Mysql-server installation is recommended to achieve a more stable linotp installation. Using apache2 you get authentication and due to threading it is not possible to use SQLite with LinOTP running under apache2

4) Download the following packages:

linotp_2.2_all.deb

linotpuseridresolver_2.2_all.deb

libpam-linotp_2.2_i386.deb

linotpadminclientce_2.2-pre2_all.deb

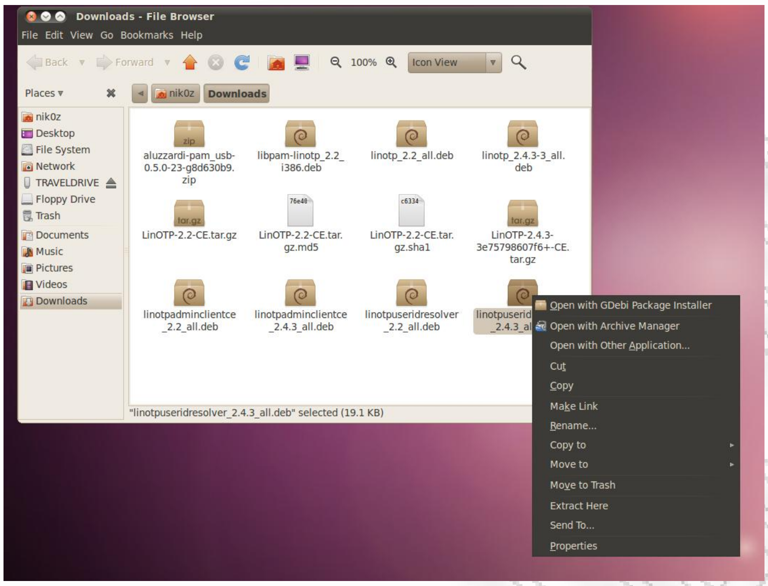
from here: http://www.linotp.org/index.php/download

5) Install the packages by issuing the command:

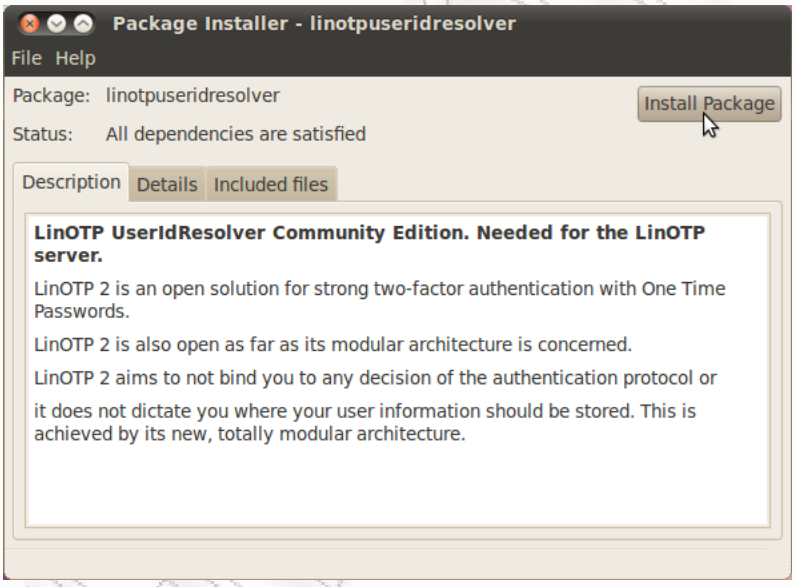
dpkg -i linotp_2.2_all.deb linotpuseridresolver_2.2_all.deb libpam-linotp_2.2_i386.deb linotpadminclientce_2.2_all.deb

(alternately you can choose to do it by using the gui of Ubuntu. You should go to the folder were these files were downloaded and >right click on linotpuseridresolver_2.2_all.deb >Open with GDebi Package Installer>Install package

Do the same for the *linotpadminclientce_2.2_all.deb* and *linotp_2.2_all.deb* in that sequence!


Downloads - File Browser
File Edit View Go Bookmarks Help
Back
Forward
100%
Icon View
Places
nik0z
Desktop
File System
Network
TRAVELDRIVE
Floppy Drive
Trash
Documents
Music
Pictures
Videos
Downloads
aluzzardi-pam_usb-0.5.0-23-g8d630b9.zip
libpam-linotp_2.2_i386.deb
linotp_2.2_all.deb
linotp_2.4.3-3_all.deb
LinOTP-2.2-CE.tar.gz
LinOTP-2.2-CE.tar.gz.md5
LinOTP-2.2-CE.tar.gz.sha1
LinOTP-2.4.3-3e75798607f6+-CE.tar.gz
linotpadminclientce_2.2_all.deb
linotpadminclientce_2.4.3_all.deb
linotpuseridresolver_2.2_all.deb
linotpuseridresolver_2.4.3_all.deb
Open with GDebi Package Installer
Open with Archive Manager
Open with Other Application...
Cut
Copy
Make Link
Rename...
Copy to
Move to
Move to Trash
Extract Here
Send To...
Properties
"linotpuseridresolver_2.4.3_all.deb" selected (19.1 KB)

Picture 79: Installing Linotp2



Package: linotpuseridresolver

Status: All dependencies are satisfied

LinOTP UserIdResolver Community Edition. Needed for the LinOTP server.

LinOTP 2 is an open solution for strong two-factor authentication with One Time Passwords.

LinOTP 2 is also open as far as its modular architecture is concerned.

LinOTP 2 aims to not bind you to any decision of the authentication protocol or

it does not dictate you where your user information should be stored. This is achieved by its new, totally modular architecture.

Picture 80: Package Installer

No matter which method you will use to install the packages, when installing linotp_2.2_all.deb you will be asked several questions, make the following configuration.

Do you want to run LinOTP 2 via Apache2?
-> Yes
Enter admin password for the new LinOTP admin account.
-> think of one
Do you want to create a self-signed certificate?
-> Yes
What SQL database do you want to use for the token database?
-> Mysql

database hostname:
-> localhost
Name of Token database
-> LinOTP2
database user
-> linotp2
database users password:
-> think of one
Enter my sql root password
Do you want to create the tables
-> yes

(Should any problem appears, run dpkg in debug mode and try to locate and fix the problem

dpkg --debug=3773 -i linotp_2.2_all.deb linotpuseridresolver_2.2_all.deb libpam-linotp_2.2_i386.deb linotpadminclientce_2.2_all.deb)

Then the encryption key /etc/linotp2/encKey will be created.

Restart the apache:

/etc/init.d/apache2 reload

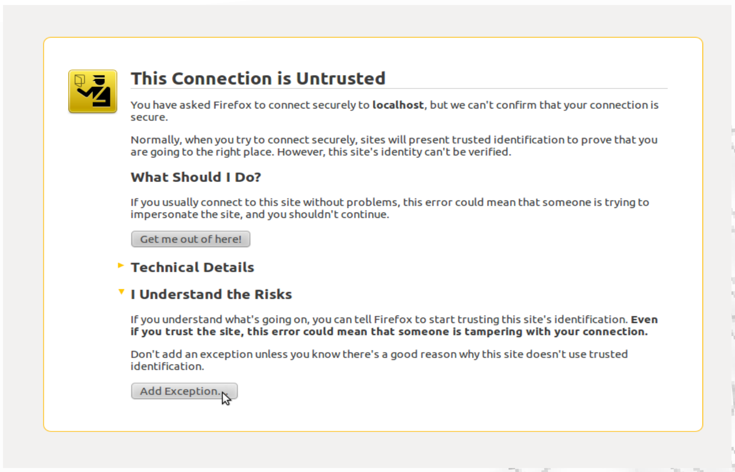
You will be able to see the management interface by pointing your browser to:

https://localhost/manage/index

Once you do that you will have to accept the certificate that was just created

We click on 'I understand the risks'

And 'Add Exception' as we can see in the following screenshot.


This Connection is Untrusted

You have asked Firefox to connect securely to **localhost**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

What Should I Do?

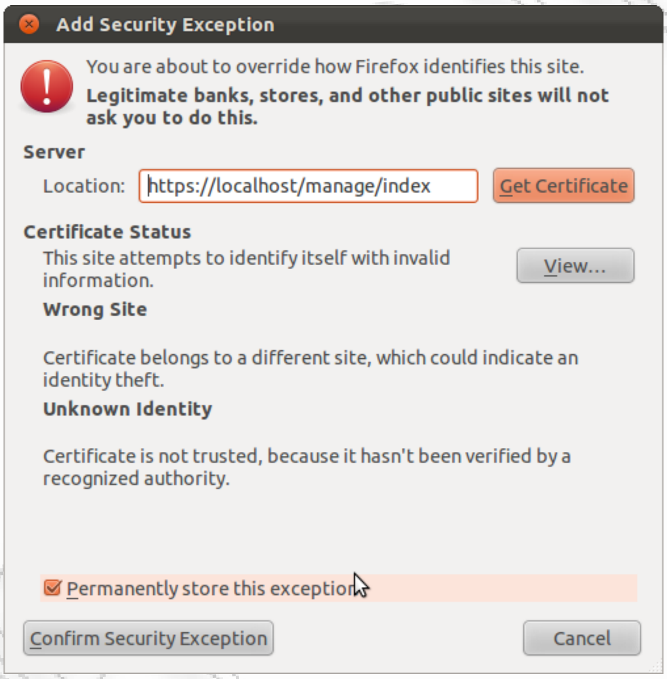
If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

If you understand what's going on, you can tell Firefox to start trusting this site's identification. **Even if you trust the site, this error could mean that someone is tampering with your connection.**

Don't add an exception unless you know there's a good reason why this site doesn't use trusted identification.

Picture 81: Add Exception

Then we simply 'Confirm the Security Exception'



You are about to override how Firefox identifies this site.
Legitimate banks, stores, and other public sites will not ask you to do this.

Server
Location: https://localhost/manage/index

Certificate Status
This site attempts to identify itself with invalid information.

Wrong Site

Certificate belongs to a different site, which could indicate an identity theft.

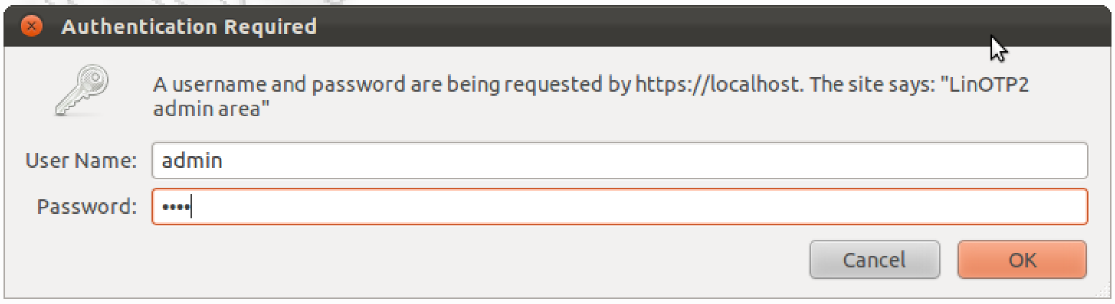
Unknown Identity

Certificate is not trusted, because it hasn't been verified by a recognized authority.

☑ Permanently store this exception

Picture 82: Confirm Security Exception

For username we type 'admin' and for password the password we used during the linopt2 installation.



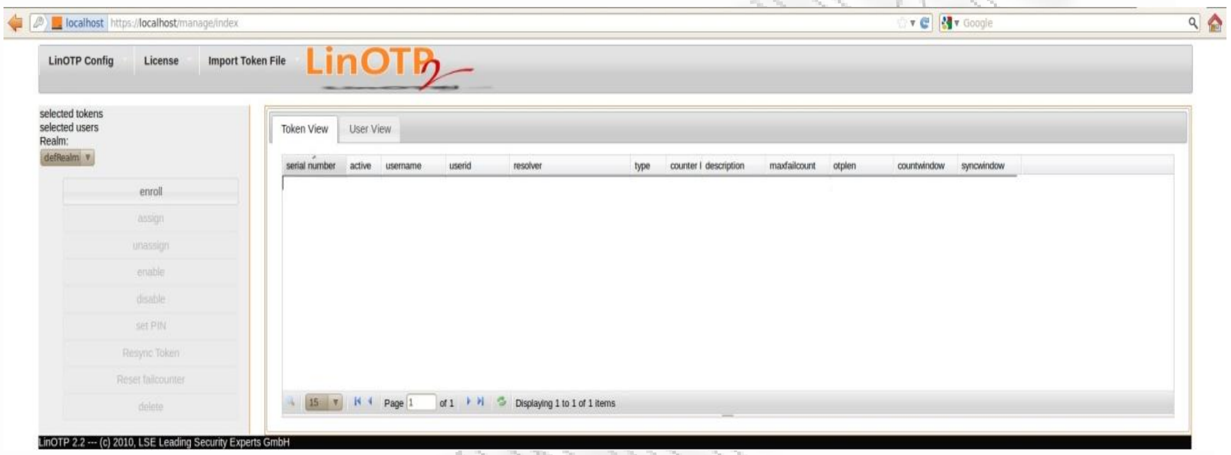
A username and password are being requested by https://localhost. The site says: "LinOTP2 admin area"

User Name: admin
Password: ••••

Picture 83: Insert admin credentials

If you get an "Internal Server error", assure, that the complete directory /etc/linotp2 is owned by the user linotp

If everything goes fine the linotp2 web user interface will appear.



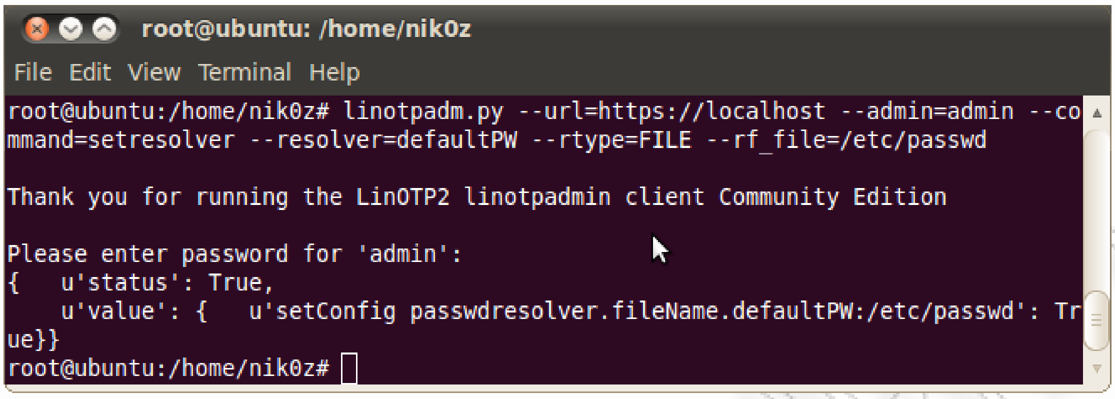
Picture 84: LinOTP2 WEB User Interface

Configuration

You may use the management web user interface or the command line client to setup your LinOTP server. You need to configure a UserIdResolver (a source were usernames/id's will be extracted from your system) and a default realm. You may do this by issuing the following commands:

linotpadmin.py --url=https://localhost --admin=admin --command=setresolver --resolver=defaultPW --rtype=FILE --rf_file=/etc/passwd

That reads the users from the /etc/passwd file. You will get a JSON feedback like this:



```
root@ubuntu:/home/nik0z# linotpadm.py --url=https://localhost --admin=admin --co
mmand=setresolver --resolver=defaultPW --rtype=FILE --rf_file=/etc/passwd

Thank you for running the LinOTP2 linotpadmin client Community Edition

Please enter password for 'admin':
{   u'status': True,
    u'value': {   u'setConfig passwdresolver.fileName.defaultPW:/etc/passwd': Tr
ue}}
root@ubuntu:/home/nik0z# 
```

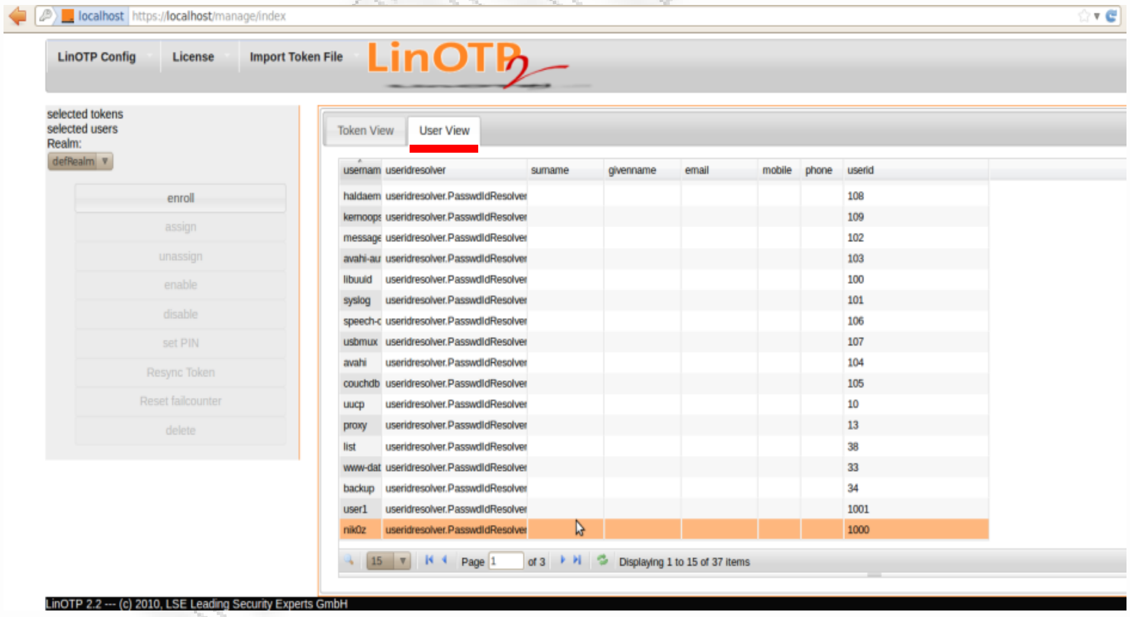
Picture 85: Terminal Screenshot

Now you need to add this resolver to your default realm:

linotpadmin.py --url=https://localhost --admin=admin --command=setrealm --realm=defRealm --resolver=useridresolver.PasswdIdResolver.IdResolver.defaultPW

linotpadmin.py --url=https://localhost --admin=admin --command=setdefaultrealm --realm=defRealm

You can either check it in the web UI



usernam	useridresolver	surname	givenname	email	mobile	phone	userid
haldaem	useridresolver.PasswdIdResolver						108
kernoops	useridresolver.PasswdIdResolver						109
message	useridresolver.PasswdIdResolver						102
avahi-au	useridresolver.PasswdIdResolver						103
libuuid	useridresolver.PasswdIdResolver						100
syslog	useridresolver.PasswdIdResolver						101
speech-c	useridresolver.PasswdIdResolver						106
usbmux	useridresolver.PasswdIdResolver						107
avahi	useridresolver.PasswdIdResolver						104
couchdb	useridresolver.PasswdIdResolver						105
uucp	useridresolver.PasswdIdResolver						10
proxy	useridresolver.PasswdIdResolver						13
list	useridresolver.PasswdIdResolver						38
www-dat	useridresolver.PasswdIdResolver						33
backup	useridresolver.PasswdIdResolver						34
user1	useridresolver.PasswdIdResolver						1001
nik0z	useridresolver.PasswdIdResolver						1000

15 Page 1 of 3 Displaying 1 to 15 of 37 items

LinOTP 2.2 --- (c) 2010, LSE Leading Security Experts GmbH

Picture 86: Users List

or issue the command

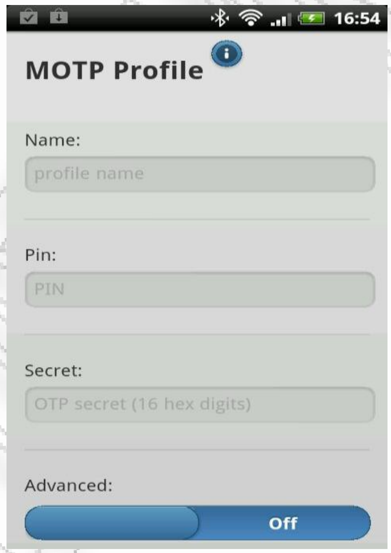
linotpadmin.py --url=https://localhost --admin=admin --command=listuser

Now we are ready to assign tokens to our users.

Enroll OTP Token

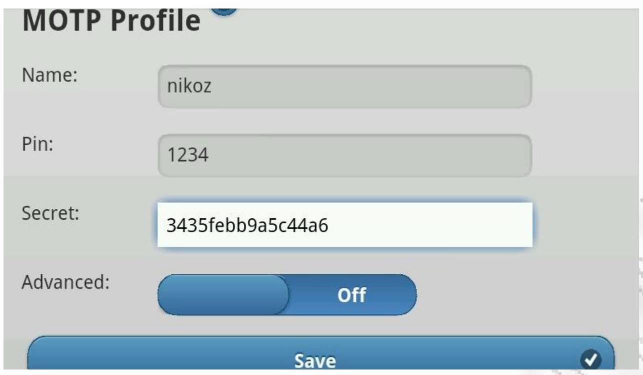
You may want to enroll an OTP. In this example we enroll a motp token on our mobile phone (HTC Desire S android-2.3.5).This token will be generated to our phone every time we want to authenticate to the system. We used the app **mOTP token** which can be found downloaded and installed to your phone from the android market *'Google Play'*. We create our user by filling the name the pin we want (the pin will be inserted every time we would want to generate a new One Time Password) and the secret value of our choice (16 hex digits) like in the following screenshots.

This is the main page where we create new Profile.



Picture 87: Main Page of android application MOTP

We create the usename, we add the pin we want (which is going to be asked every time we wish to generate a new OTP in our phone) and finally the secret value-seed which is going to be used in combination with our pin to create the One Time Password. Then we simply save our settings.


MOTP Profile

Name: nikoz

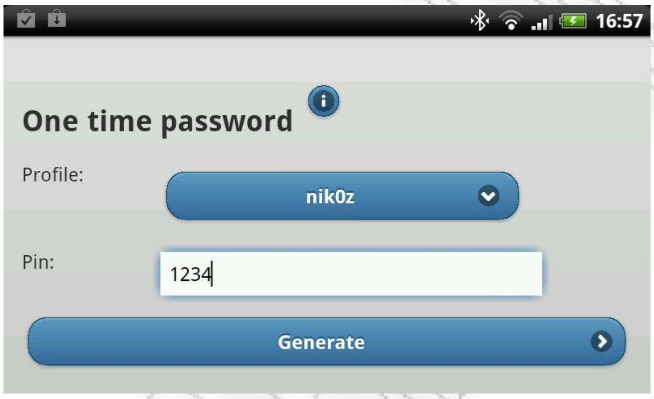
Pin: 1234

Secret: 3435febb9a5c44a6

Advanced: Off

Picture 88: Insert Pin and Secret Value

We then we will face the following tab were the pin that we have chosen will be asked and then we just click generate.



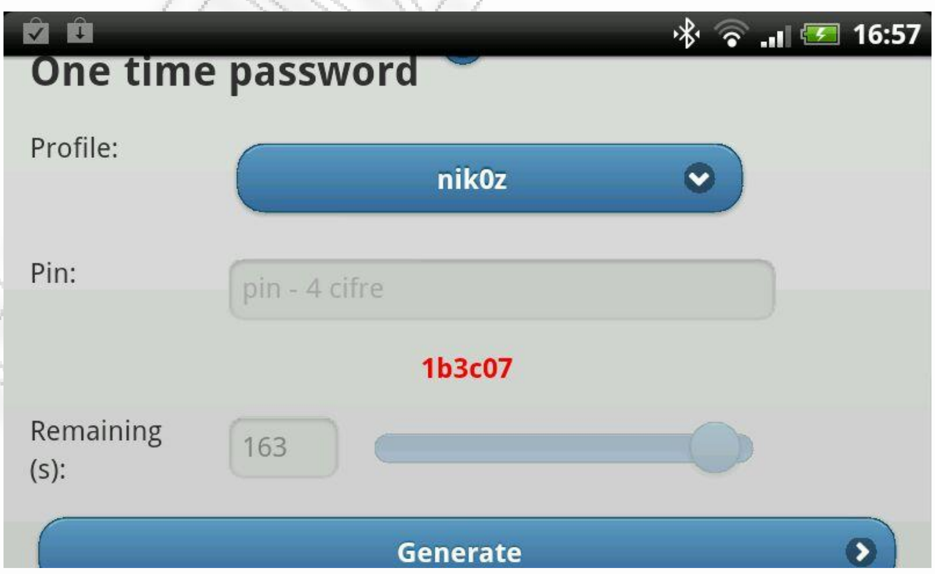
One time password

Profile: nik0z

Pin: 1234

Picture 89: Generate a OTP

Our OTP is the six digits 1b3c07 as we can see in the following screenshot and it is valid for a finite time (in this case for the next 163 seconds).



One time password

Profile: nik0z

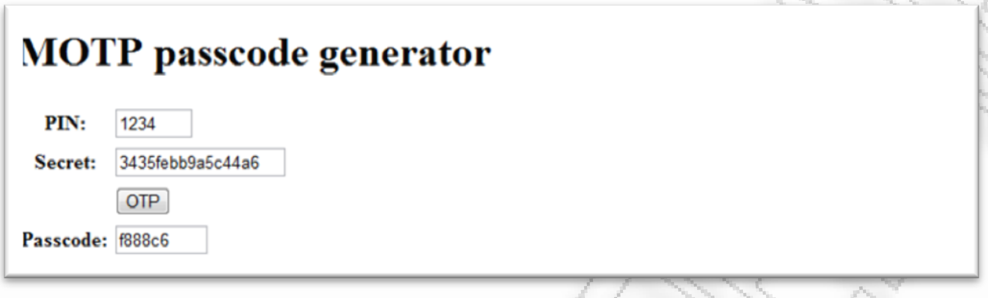
Pin: pin - 4 cifre

1b3c07

Remaining (s): 163

Picture 90: The OTP is generated

If we don’t want to use a mobile device and we just want to test if our configuration works properly we can use the following url: http://motp.sourceforge.net/motp_js.html, were we can insert the pin and the secret we want to use and generate the One-Time-Password(OTP) as we can see in the following screenshot as we did with the phone application.



MOTP passcode generator

PIN: 1234

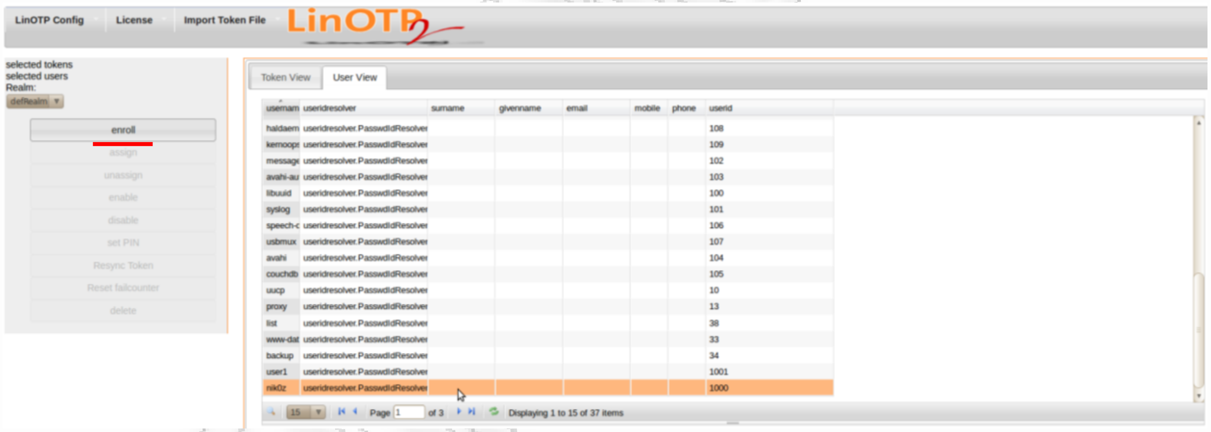
Secret: 3435febb9a5c44a6

OTP

Passcode: f888c6

Picture 91: MOTP passcode generator

Then we go back to our computer and create our user to our linotp server. This can be done either by using the web user interface where we select the user we want and click ‘enroll’.



usernam	useridresolver	surname	givenname	email	mobile	phone	userid
haldaem	useridresolver.PasswdIdResolver						108
kernoops	useridresolver.PasswdIdResolver						109
message	useridresolver.PasswdIdResolver						102
avahi-au	useridresolver.PasswdIdResolver						103
libuuid	useridresolver.PasswdIdResolver						100
syslog	useridresolver.PasswdIdResolver						101
speech-c	useridresolver.PasswdIdResolver						106
usbmux	useridresolver.PasswdIdResolver						107
avahi	useridresolver.PasswdIdResolver						104
couchdb	useridresolver.PasswdIdResolver						105
uucp	useridresolver.PasswdIdResolver						10
proxy	useridresolver.PasswdIdResolver						13
list	useridresolver.PasswdIdResolver						38
www-dat	useridresolver.PasswdIdResolver						33
backup	useridresolver.PasswdIdResolver						34
user1	useridresolver.PasswdIdResolver						1001
nik0z	useridresolver.PasswdIdResolver						1000

15 Page 1 of 3 Displaying 1 to 15 of 37 items

Picture 92: Create a token to our Server

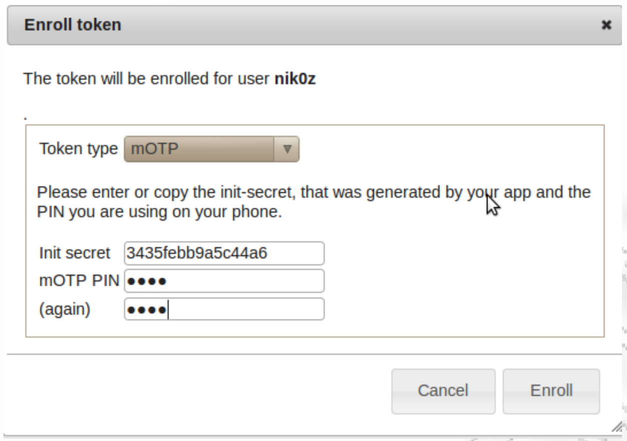
Then we select

Token Type : mOTP

Init secret : the same 16 hex digits we used on our phone application

mOTP : 1234 (the same we will every time to out phone)

and click ‘Enroll’.



The token will be enrolled for user **nik0z**

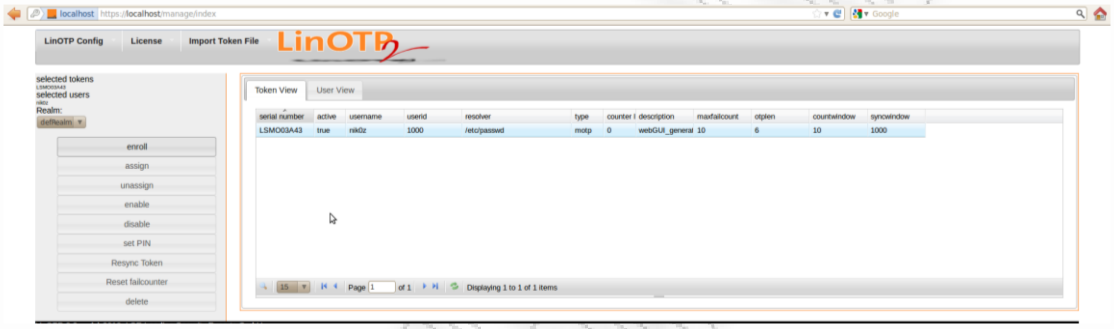
Token type: mOTP

Please enter or copy the init-secret, that was generated by your app and the PIN you are using on your phone.

Init secret	3435febb9a5c44a6
mOTP PIN	••••
(again)	••••

Picture 93: mOTP Token

As we can see our first token has been created.



selected tokens
LSMO03A43
selected users
nik0z
Realm: defRealm

serial number	active	username	userid	resolver	type	counter I	description	maxfailcount	otplen	countwindow	syncwindow
LSMO03A43	true	nik0z	1000	/etc/passwd	motp	0	webGUI_general	10	6	10	1000

Displaying 1 to 1 of 1 items

Picture 94: New server Token

Or alternatively with the use the following command in terminal to avoid all the web user interface procedure.

linotpadmin.py --url=https://localhost --admin=admin --command=inittoken –user=nik0z(whatever user you like to authenticate to your Ubuntu OS) **--type=motp --otpkey= 3435febb9a5c44a6**(the 16 hex digits we used in our mobile device) **--serial=1 –otppin=1234**(whatever pin we used in our mobile device)

This means, that you generated a mOTP token that gets assigned to the user "cornelius". The otpkey is the init-secret that was displayed on your phone. You may choose a serial (serial number) as you like to. The otppin is the PIN you will enter on the phone to generate an otp value.

TWO-factor Authentication

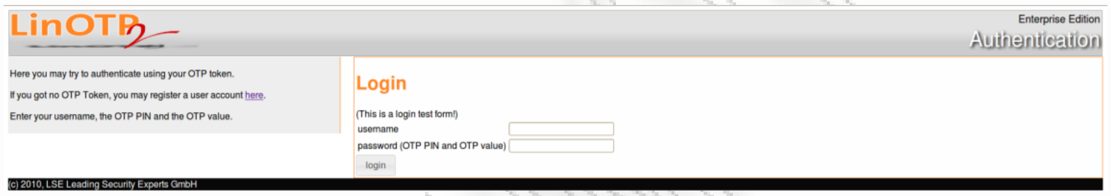
Optionally we may set an OTP PIN, which is a fixed-static password part that you need to enter in front (Depending on the parameter PrependPIN) of the OTP value.

By using this option we actually enable a two factor authentication procedure (something we know: static password and something we create in real time with our OTP generator: one time password from our phone)

With the following command our static password from now on will be 123.

linotpadmin.py --url=https://localhost --admin=admin --command=set --pin=123 --serial=1

Now we can test the token that we created. Insert to you browser the url: https://localhost/auth/index and you will enter the following page.



Here you may try to authenticate using your OTP token.
If you got no OTP Token, you may register a user account here.
Enter your username, the OTP PIN and the OTP value.

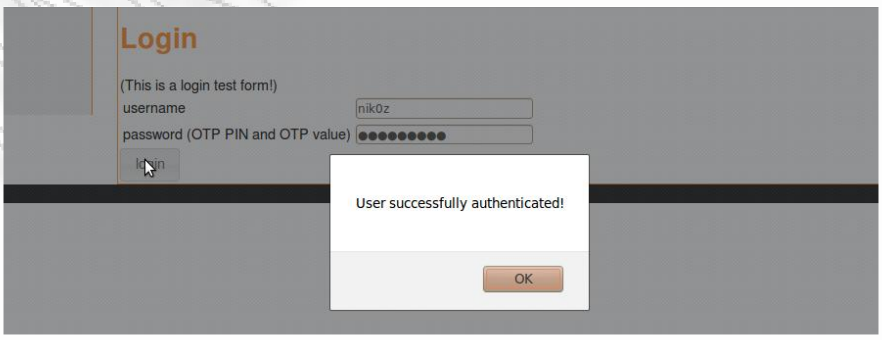
Login
(This is a login test form!)
username
password (OTP PIN and OTP value)
login

(c) 2010, LSE Leading Security Experts GmbH

Picture 95: LinOTP test page

Enter your username and as password the OTP PIN we just created (in our case the simple pin 123) and then the 6 digit OTP value you generated with your phone. We don’t have to delay to enter the OTP because the OTP value that is generated to our phone is based to epoch (time in seconds since 1/1/1970), thus it is synchronized to our computer, consequently the OTP is valid for a few seconds after and after some time it will no longer be valid so we should generate a new one. If we fail to authenticate, we might mistyped the secret value or most likely the time of our phone is out of sync.

If everything goes well we should see a message like this to appear.


Login
(This is a login test form!)
username nik0z
password (OTP PIN and OTP value) ●●●●●●●●●
login

User successfully authenticated!
OK

Picture 96: Successful authentication

As we can see by the dots in the password field we used our pin (3 digits) + our OTP value (6 digits).

Authenticate to our Operating System Desktop

Now we will setup PAM to enable the ability to authenticate to our desktop via OTP. The security module /lib/security/pam_linotp.so is already installed to our system. We may now setup otp authentication. We choose to do it modular:

cp /etc/pam.d/common-auth /etc/pam.d/common-linotp

Just for safety we won't use the common-auth file but instead we will copy all its contents (backup) to a new file called common-linotp and we will make there all the necessary changes.

Run the command

gedit /etc/pam.d/common-linotp

and change the line:

auth [success=1 default=ignore] pam_unix.so nullok_secure

to

auth [success=1 default=ignore] pam_linotp.so debug nosslhostnameverify nosslcertverify url=https://localhost/validate/simplecheck realm=defrealm resConf=defaultPW

Please note, that when using "debug" option, many information - also the Password - will be written to the auth.log. Now we may use the common-linotp in any PAM configuration instead of common-auth where we like to, here we will change the following lines in /etc/pam.d/gdm in order to authenticate to our system using OTP authentication.

Run the command

gedit /etc/pam.d/gdm

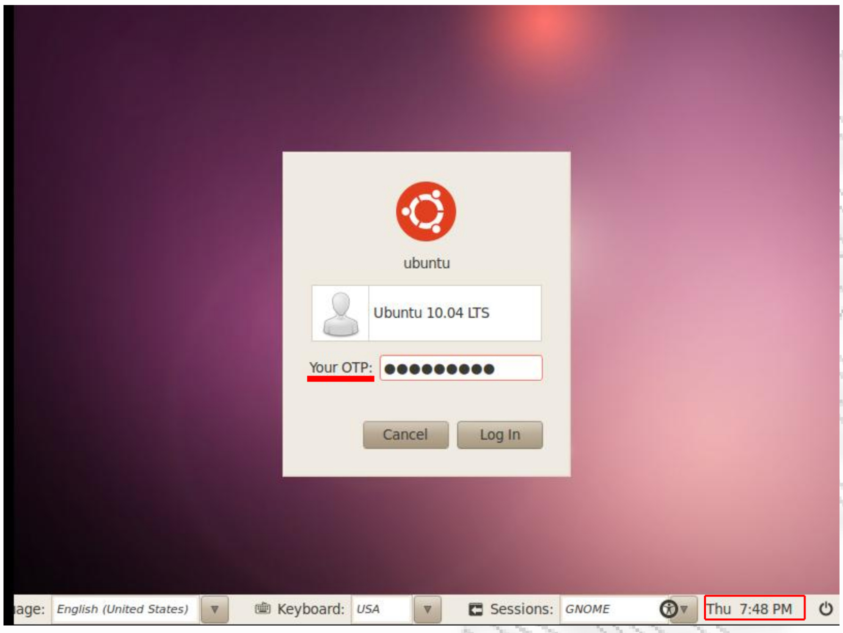
and change

@include common-auth

to

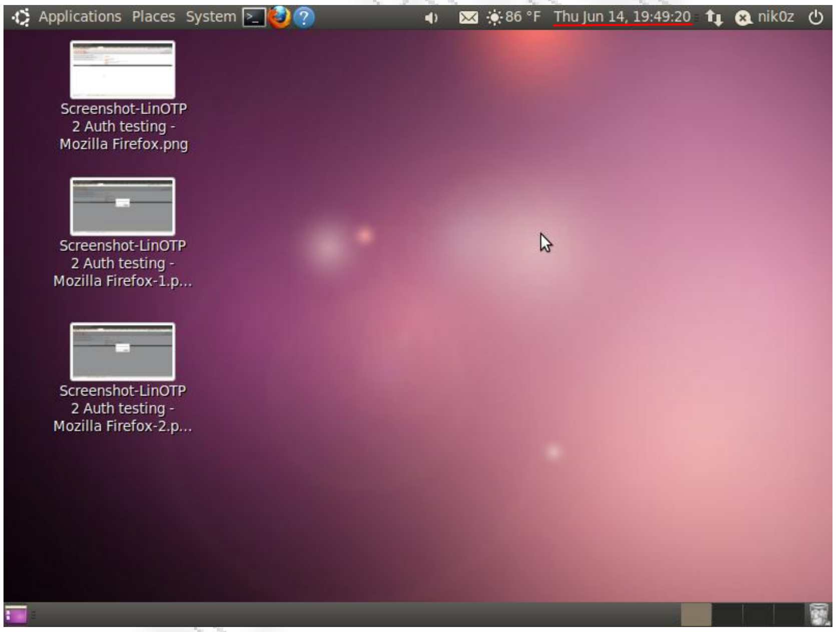
@include common-linotp

Now you will need to authenticate with OTP to our Gnome desktop! Note that we will also need to change /etc/pam.d/gnome-screensaver if we also want to unlock the desktop using OTP: You will not be asked for "Password" anymore but for "Your OTP", as we can see above.



Picture 97: OTP request

And for the proof of concept we login to our desktop


Applications Places System
86 °F Thu Jun 14, 19:49:20
nik0z
Screenshot-LinOTP 2 Auth testing - Mozilla Firefox.png
Screenshot-LinOTP 2 Auth testing - Mozilla Firefox-1.p...
Screenshot-LinOTP 2 Auth testing - Mozilla Firefox-2.p...

Picture 98: OS Desktop

If you fail to authenticate you may also take a look into /var/log/auth.log

Triple-factor authentication

Now we can implement a triple factor authentication by combining the linOTP2 tool with the PAM-USB module (Something we know:our pin, something we randomly create: OneTimePassword from our phone generator and finally something we possess: the USB-flash drive). As we did in the previous PAM-USB module section we download and install PAM-USB module from pamusb.org

Or if we are using Ubuntu 10.04 as we do here we simply open a terminal and run the command

sudo apt-get install libpam-usb pamusb-tools

and then the procedure is the same as follows:

in order to add a USB device

pamusb-conf --add-device=TRAVELDRIVE

to add a user we run the command

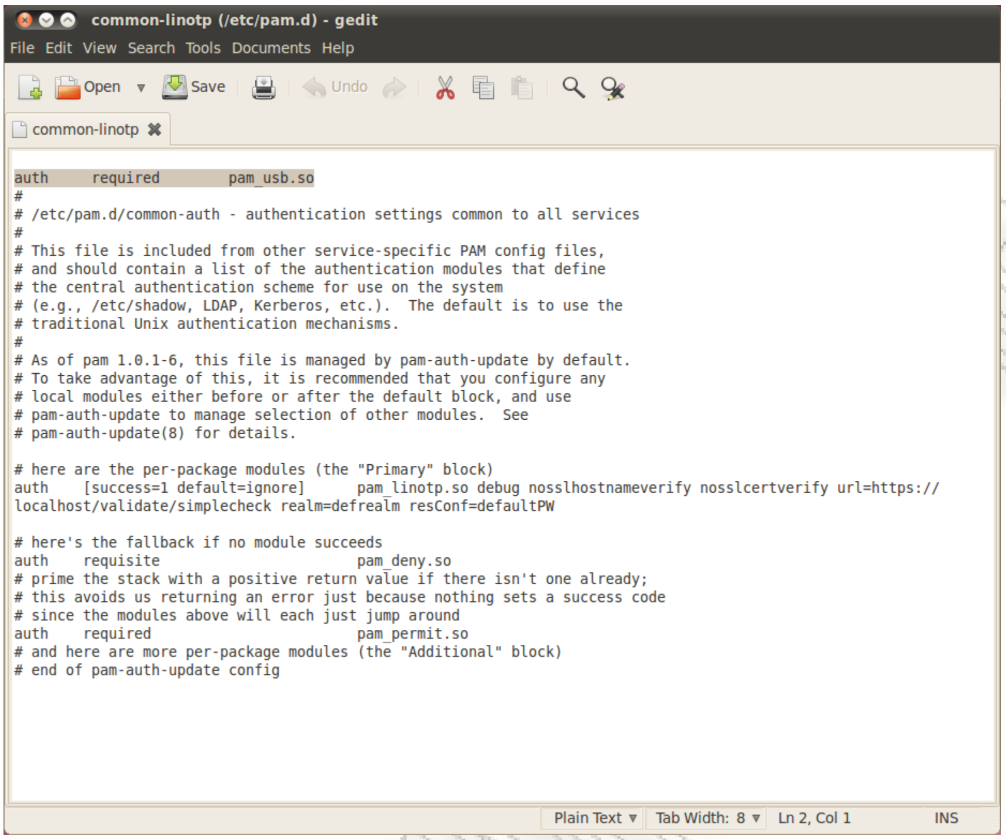
pamusb-conf --add-user=nik0z

and then we modify the common-auth file of our system. In this case we will modify the common-linotp file since we have replaced the common-auth file with the common-linop during this current configuration a few steps back. So we run the command

gedit /etc/pam.d/common-linotp

and we add the first line as follows

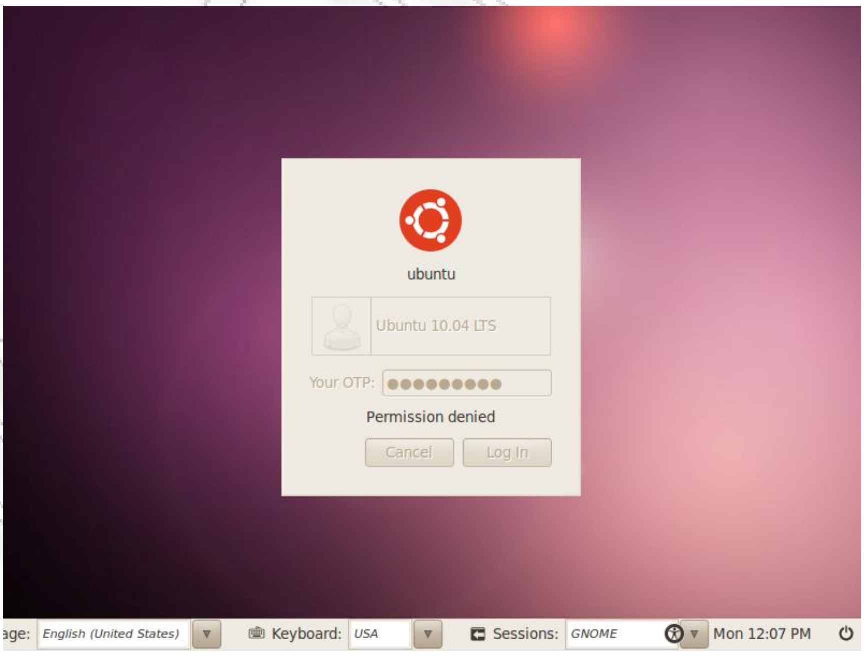
auth required pam_usb.so


common-linotp (/etc/pam.d) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
common-linotp
Plain Text Tab Width: 8 Ln 2, Col 1 INS

```
auth     required        pam_usb.so
#
# /etc/pam.d/common-auth - authentication settings common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of the authentication modules that define
# the central authentication scheme for use on the system
# (e.g., /etc/shadow, LDAP, Kerberos, etc.).  The default is to use the
# traditional Unix authentication mechanisms.
#
# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
auth    [success=1 default=ignore]      pam_linotp.so debug nosslhostnameverify nosslcertverify url=https://
localhost/validate/simplecheck realm=defrealm resConf=defaultPW

# here's the fallback if no module succeeds
auth    requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
auth    required                        pam_permit.so
# and here are more per-package modules (the "Additional" block)
# end of pam-auth-update config
```

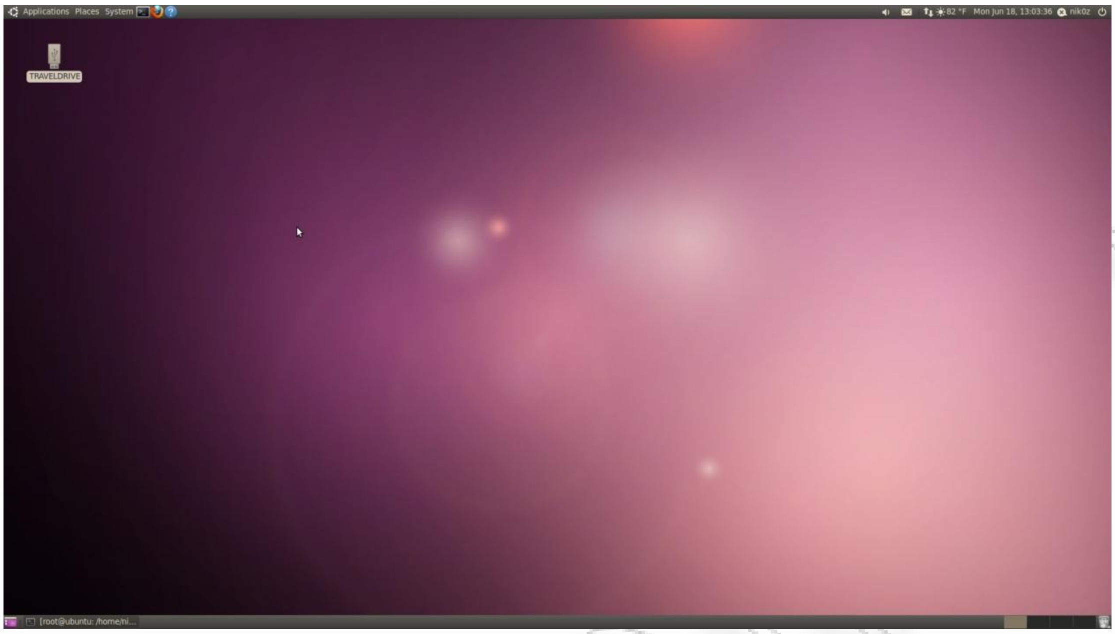
Picture 99: Common-linotp configuration file

We remove the USB flash drive from our computer and we try to login. As we can see the authentication fails (Permission Denied).



Picture 100: Permission Denied

Then we insert our USB flash drive and we type again our password and the OTP from our mobile device. The proof of concept, we login to our Desktop.


Applications Places System
82 °F Mon Jun 18, 13:03:36 nik0z
TRAVELDRIVE
[root@ubuntu: /home/ni...

Picture 101: Login Successfully to OS Desktop

Success!

References

Freeradius MySQL

- http://wiki.freeradius.org
- http://www.mydeveloperblog.com/linux-tutorial/radius/radius-servers-installation-guide-freeradius-ubuntu-mysql/
- http://www.mydeveloperblog.com/linux-tutorial/radius/radius-servers-configuration-guide-freeradius-ubuntu-mysql/
- https://help.ubuntu.com/community/phpMyAdmin
- http://www.youtube.com/watch?v=gSCgESg8UvY
- http://www.novell.com/coolsolutions/tools/14377.html
- https://help.ubuntu.com/community/UsingTheTerminal
- http://www.docstoc.com/docs/22211496/How-to-configure-FreeRadiusnet-to-work-with-Alcatel-Lucent-OmniSwitch
- http://linuxtechtutorials.blogspot.com/
- http://publib.boulder.ibm.com/infocenter/rsthelp/v8r0m0/index.jsp?topic=%2Fcom.ibm.rational.test.lt.doc%2Ftopics%2Ftcreatecertopenssl.html
- http://www.openssl.org/support/faq.html
- http://shib.kuleuven.be/docs/ssl_commands.shtml
- http://www.docstoc.com/docs/22211496/How-to-configure-FreeRadiusnet-to-work-with-Alcatel-Lucent-OmniSwitch
- http://www.dartmouth.edu/~pkilab/pages/EAP-TLSwFreeRadius.html
- http://forums.freebsd.org/showthread.php?t=28467
- http://blog.wains.be/2009/09/13/wpa2-freeradius-eap-tls/
- http://www.privacywonk.net/2010/10/security-how-to-wpa2-enterprise-on-your-home-network.php
- http://www.youtube.com/watch?v=fVRDbAxKcqw&feature=related
- http://shib.kuleuven.be/docs/ssl_commands.shtml

Freeradius LDAP

- http://wiki.freeradius.org/
- http://wiki.freeradius.org/LDAP
- http://www.novell.com/coolsolutions/feature/15359.html
- http://oss.sgi.com/LDP/HOWTO/LDAP-Implementation-HOWTO/radius.html

- http://linux-ubuntu1104.blogspot.com/
- http://www.howtoforge.com/wikid-openldap-freeradius-howto
- http://serverfault.com/questions/47815/setting-up-radius-ldap-for-wpa2-on-ubuntu
- http://ubuntuforums.org/showthread.php?t=1760830
- http://ubuntuforums.org/showthread.php?t=1976883
- http://www.youtube.com/watch?v=gSCgESg8UvY&feature=relmfu
- http://www.youtube.com/watch?v=Mw_jrGX635Y&feature=channel&list=UL
- http://www.youtube.com/watch?v=DM_UQVVVtoY

PAM/USB Module

- http://pamusb.org/
- http://csc560.wikispaces.com/PAM+USB
- http://www.youtube.com/watch?v=RDkK1quOfuE&feature=related
- http://www.youtube.com/watch?v=CCfwgqZlsoA
- https://github.com/aluzzardi/pam_usb/wiki/Install
- https://github.com/aluzzardi/pam_usb/wiki/Configuration
- https://github.com/aluzzardi/pam_usb/wiki/Getting-Started
- http://linux-software-news-tutorials.blogspot.gr/2011/03/login-in-linux-with-usb-memory.html
- http://ubuntuforums.org/showthread.php?t=17571
- http://linuxconfig.org/linux-authentication-login-with-usb-device
- http://en.gentoo-wiki.com/wiki/PAM_Authentication_using_USB_Devices
- http://www.novell.com/communities/node/3627/pam-pluggable-authentication-module-usb-authentication

LinOTP

- http://en.wikipedia.org/wiki/One-time_password
- http://www.linotp.org/index.php/
- http://www.linotp.org/index.php/download
- http://www.linotp.org/index.php/howtos/5-community/22-howto-install-linotp2-with-ubuntu-1010-to-secure-your-desktop-with-otp
- http://motp.sourceforge.net/
- http://motp.sourceforge.net/motp_js.html
- http://shearer.org/Debugging_Dpkg_Problems
- http://www.rcdevs.com/tokens/?type=software

- https://play.google.com/store/apps/details?id=ro.len.mobile
- https://play.google.com/store/apps/details?id=uk.co.bitethebullet.android.token&feature=search_result#?t=W251bGwsMSwxLDEsInVrLmNvLmJpdGV0aGVidWxsZXQuYW5kcm9pZC50b2tlbiJd
- https://play.google.com/store/apps/details?id=com.google.android.apps.authenticator2&feature=search_result#?t=W251bGwsMSwxLDEsImNvbS5nb29nbGUuYW5kcm9pZC5hcHBzLmF1dGhlbnRpY2F0b3IyIl0.
- http://www.androidpit.com/en/android/market/apps/app/net.marinits.android.droidotp/DroidOTP
- http://www.infosecisland.com/blogview/11813-One-Time-Passwords-are-Not-Secure-Enough.html