Guide d’installation de FreeRadius avec EAP-TLS + MySQL

freeRADIUS OpenSSL MySQL

Installation de FreeRadius avec EAP-TLS + MySQL

I/ Introduction :

L'utilisation de EAP-TLS sur un réseau sans fil est une des méthodes les plus sûres pour sécuriser son réseau Wifi, mais son déploiement n'est pas des plus facile. Dans ce guide d'installation on commencera par l'installation de Freeradius + EAP-TLS puis on rajoutera le support d'une base de donnée MySQL ce qui permettra de réaliser un déploiement plus simple.

Pour l'installation, nous avons utilisé une Debian Sarge (fraîchement installé) avec un kernel 2.6.
Pour la compilation des différentes applications, vous aurez besoin des modules suivant :

```
apt-get install make gcc g++ wget openssl
```

II/ Installation et configuration de OpenSSL

Version utilisée : openssl-0.9.7g (dernière version stable)
On installe OpenSSL dans un autre répertoire, on l'utilisera pour générer les certificats. On installe une autre version de OpenSSL que celui du système qui nous servira à la génération des certificats.

II-1/ Installation :

Télécharger sur le site www.openssl.org, la version stable 0.9.7g,

```
tar zxvf openssl-0.9.7g.tar.gz
cd openssl-0.9.7g
./config --prefix=/usr/local/openssl-certgen shared
make && make install
```

OpenSSL se compile, cela dure plus ou moins longtemps suivant votre machine, vous devez avoir à la fin un message du type :

```
OpenSSL shared libraries have been installed in:
 /usr/local/openssl-certgen
If this directory is not in a standard system path for
dynamic/shared
libraries, then you will have problems linking and executing
applications that use OpenSSL libraries UNLESS:
[....]
See any operating system documentation and manpages about shared
libraries for your version of UNIX.  The following manpages may be
helpful: ld(1), ld.so(1), ld.so.1(1) [Solaris], dld.sl(1) [HP],
ldd(1), crle(1) [Solaris], pldd(1) [Solaris], ldconfig(8) [Linux],
chatr(1) [HP].
cp openssl.pc /usr/local/openssl-certgen/lib/pkgconfig
chmod 644 /usr/local/openssl-certgen/lib/pkgconfig/openssl.pc
```

Sinon reportez vous à la fin du guide dans l'annexe « Résolution des erreurs ».

II-2/ Configuration de OpenSSL

Il faut maintenant éditer le fichier de configuration de OpenSSL, ce fichier contient les différentes informations comme le nom de « l'entreprise », le pays, l'adresse e-mail, le nom du propriétaire du certificat...
Editer via votre éditeur de texte préféré (nous utiliserons nano) le fichier de configuration openssl.cnf

```
nano /usr/local/openssl-certgen/ssl/openssl.cnf
```

Vers le milieu du fichier se trouve les paramètres à modifier

```
[ req_distinguished_name ]
countryName             = Country Name (2 letter code)
countryName_default             = FR
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = Some-State

localityName                    = Locality Name (eg, city)
localityName_default            = St Malo

0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = IUT ST MALO

# we can do this but it is not needed normally :-)
#1.organizationName             = Second Organization Name (eg,
company)
#1.organizationName_default     = World Wide Web Pty Ltd

organizationalUnitName          = Organizational Unit Name (eg,
section)
#organizationalUnitName_default      = Departement GTR

commonName                      = Common Name (eg, YOUR name)
commonName_max                  = 64
commonName_default              = IUT ST MALO dpt GTR

emailAddress                    = Email Address
emailAddress_max                = 64
emailAddress_default            = iutsm@univ-rennes1.fr
```

Il faut modifier les lignes qui finissent par xxx_default, pour le nom *(commonName)* et l'e-mail, il n'y a pas ces lignes, vous pouvez les rajouter pour personnaliser votre configuration. Une fois que vous les avez modifiées, il suffit d'enregistrer et quitter.

L'installation de OpenSSL est terminée

II-3/ Génération des certificats :

Pour la génération des certificats, vous avez besoin des fichiers xpextensions, CA.root, CA.svr, CA.clt (voir fichiers joints)
Créez un nouveau dossier et rentrez dans le dossier

```
mkdir /root/certs
cd /root/certs
```

Mettre les fichiers dans ce dossier. Il ne faut surtout pas oublier le fichier xpextensions qui contient les OID pour la génération des certificats.

Génération du Certificat root

Le fichier CA.root permet de générer le certificat root qui est l'autorité de certification. Il permettra la génération des certificats clients (signature du certificat). Chaque utilisateur devra avoir son certificat et l'autorité de certification sur sa machine.

On lance donc la génération du certificat

```
Debian:~/certs# ./CA.root
**********************************************************************
Creating self-signed private key and certificate
When prompted override the default value for the Common Name field
**********************************************************************

Generating a 1024 bit RSA private key
.............................................++++++
..++++++
writing new private key to 'newreq.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) [FR]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [IUT ST MALO]:
Organizational Unit Name (eg, section) [Département GTR]:
Common Name (eg, YOUR name) []:
Email Address []:
**********************************************************************
Creating a new CA hierarchy (used later by the ca command) with the
certificate
and private key created in the last step
**********************************************************************
**********************************************************************
Creating ROOT CA
**********************************************************************
MAC verified OK
```

A chaque question, tapez sur la touche Entrée. Il crée les fichier root.pem, root.p12, root.der et le dossier demoCA. Le fichier root.pem est utilisé par FreeRadius, et il faudra installer le fichier root.der sur chaque station cliente. En cas d'erreur se reporter à l'annexe « Résolution des erreurs ».

Génération du certificat serveur

Ce certificat sera installé sur la machine avec le serveur radius.
Il faut passer en paramètre le nom du fichier que l'on veut et à la question ***Common Name (eg, YOUR Name) []:*** on rentre le nom du serveur (dans notre exemple **serveur)**

```
Debian:~/certs# ./CA.svr serveur
**********************************************************************
Creating server private key and certificate
When prompted enter the server name in the Common Name field.
**********************************************************************

Generating a 1024 bit RSA private key
............++++++
.................++++++
writing new private key to 'newreq.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) [FR]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [IUT ST MALO]:
Organizational Unit Name (eg, section) [Département GTR]:
Common Name (eg, YOUR name) []:serveur
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /usr/local/openssl-certgen/ssl/openssl.cnf
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number:
            a8:db:b4:b4:1f:78:6e:3a
        Validity
            Not Before: Feb 14 16:35:22 2005 GMT
            Not After : Feb 14 16:35:22 2006 GMT
        Subject:
            countryName               = FR
            stateOrProvinceName       = Some-State
            organizationName          = IUT ST MALO
            organizationalUnitName    = Département GTR
            commonName                = serveur
        X509v3 extensions:
            X509v3 Extended Key Usage:
                TLS Web Server Authentication
Certificate is to be certified until Feb 14 16:35:22 2006 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
MAC verified OK
```

On se retrouve donc avec les fichiers serveur.pem, serveur.p12, serveur.der.

Génération du certificat client

Même chose que pour le certificat client, on passe en paramètre le nom du fichier et à la question ***Common Name***, mettre le nom de l'utilisateur (dans notre exemple **client**). Attention, on ne peut pas avoir 2 certificats avec le même nom d'utilisateur !

```
Debian:~/certs# ./CA.clt client
**********************************************************************
Creating client private key and certificate
When prompted enter the client name in the Common Name field. This is the same
 used as the Username in FreeRADIUS
**********************************************************************

Generating a 1024 bit RSA private key
.....................++++++
....++++++
writing new private key to 'newreq.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) [FR]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [IUT ST MALO]:
Organizational Unit Name (eg, section) [Département GTR]:
Common Name (eg, YOUR name) []:client
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /usr/local/openssl-certgen/ssl/openssl.cnf
DEBUG[load_index]: unique_subject = "yes"
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number:
            a8:db:b4:b4:1f:78:6e:3b
        Validity
            Not Before: Feb 14 17:20:56 2005 GMT
            Not After : Feb 14 17:20:56 2006 GMT
        Subject:
            countryName               = FR
            stateOrProvinceName       = Some-State
            organizationName          = IUT ST MALO
            organizationalUnitName    = Département GTR
            commonName                = client
        X509v3 extensions:
            X509v3 Extended Key Usage:
                TLS Web Client Authentication
Certificate is to be certified until Feb 14 17:20:56 2006 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
MAC verified OK
```

On se retrouve donc avec 3 fichiers client.pem, client.der, client.p12 on installera ce dernier sur la machine cliente

III/ Premier Installation de FreeRadius avec EAP-TLS (sans MySQL)

III-1/ Installation

Version utilisée : FreeRadius 1.0.2 (dernière version stable).

Avant l'installation de FreeRadius, il faut installer les bibliothèques suivantes afin d'éviter des erreurs pendant la compilation

- Libssl-dev pour le module EAP TLS
- Snmp pour le module SNMP
- Libltdl3-dev pour la libtool

```
apt-get install libssl-dev snmp libltdl3-dev
```

Télécharger la version stable 1.0.2 sur www.freeradius.org

```
tar zxvf freeradius-1.0.2.tar.gz
cd freeradius-1.0.2
```

Pour la configuration de compilation de FreeRadius, on utilise le paramètre *--sysconfdir=/etc/* qui placera tous les fichiers de configuration dans */etc/raddb*, le paramètre *--silent* permet d'afficher le debug minimum.

On peut utiliser :

```
./configure --sysconfdir=/etc/ --silent --disable-shared
```

Ou :

```
./configure --sysconfdir=/etc/ --without-rlm_sql_iodbc --without-
rlm_eap_sim --without-rlm_eap_gtc --without-rlm_x99_token --without-
rlm_sql_unixodbc  --without-rlm_sql_oracle --without-rlm_ldap --
without-rlm_sql_postgresql --without-rlm_ippool --without-rlm_dbm --
without-rlm_counter --without-rlm_attr_rewrite --without-rlm_pam --
without-rlm_eap_ttls --without-rlm_radump --without-rlm_dbm --without-
rlm_eap_peap --without-rlm_krb5  --silent --disable-shared
```

Qui est plus rapide (on désactive les modules que l'on n'utilise pas)

Important : Il faut vérifier pendant la configuration qu'il n'y a pas d'erreur au niveau de EAP-TLS

```
config.status: creating Makefile
configure: configuring in ./types/rlm_eap_tls
configure: running /bin/sh './configure' --prefix=/usr/local  '--
prefix=/usr/local' '--sysconfdir=/etc/' '--enable-ltdl-install=no' '--
cache-file=/dev/null' '--srcdir=.' 'CFLAGS=-g -O2 -D_REENTRANT
-D_POSIX_PTHREAD_SEMANTICS -DOPENSSL_NO_KRB5 -Wall -D_GNU_SOURCE -g
-Wshadow -Wpointer-arith -Wcast-qual -Wcast-align -Wwrite-strings
-Wstrict-prototypes -Wmissing-prototypes -Wmissing-declarations
-Wnested-externs -W -Wredundant-decls -Wundef' --cache-file=/dev/null --
srcdir=.
checking for gcc... gcc
checking for C compiler default output file name... a.out
checking whether the C compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables...
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ANSI C... none needed
configure: creating ./config.status
```

Si il y a une erreur, reportez vous à l'annexe « Résolution des erreurs ».

On peut maintenant passer à la compilation et l'installation de FreeRadius

```
make && make install
```

III-2/ Installation des Certificats sur le serveur

Rendez vous dans le dossier de configuration de FreeRadius (/etc/raddb/), puis on efface les certificats par défaut de FreeRadius, après on copie notre certificat root et serveur dans le dossier certs. On finit par générer les fichiers random & dh avec la fonction date

```
radius: # cd /etc/raddb/certs/
radius:/etc/raddb/certs# rm -rf *
radius:/etc/raddb/certs# cp /root/certs/root.pem /etc/raddb/certs
radius:/etc/raddb/certs# cp /root/certs/serveur.pem /etc/raddb/certs
radius:/etc/raddb/certs# date > random
radius:/etc/raddb/certs# date > dh
```

Note : Les fichiers random et dh sont loin d'être des fichiers aléatoires. Reportez vous à l'annexe « Allons plus loin » pour voir comment générer de vrais fichiers aléatoires.

III-3/ Configuration de FreeRadius

On va maintenant s'occuper de la configuration de FreeRadius, les fichiers de configuration se trouvent dans /etc/raddb (comme nous l'avons précisé plus tôt via le –sysconfidir à la configuration de la compilation).
On va juste modifier les fichiers suivants :
eap.conf pour la configuration de EAP et des certificats
clients.conf pour la configuration des NAS (bornes Wifi) autorités à contacter le Radius.
users pour la configuration des utilisateurs autorités.
radius.conf le fichier principal de configuration de free radius.

Fichier *eap.conf*
On spécifie que l'on veut utiliser EAP-TLS et non MD5

Ligne 22 :

```
default_eap_type = tls
```

Après on configure EAP-TLS, il faut que l'on enlève les commentaires (les # devant) à partir de la ligne 122 et on modifie les chemins des certificats

```
tls {
      private_key_password = whatever
      private_key_file = ${raddbdir}/certs/serveur.pem

      certificate_file = ${raddbdir}/certs/serveur.pem
      CA_file = ${raddbdir}/certs/root.pem

      dh_file = ${raddbdir}/certs/dh
      random_file = ${raddbdir}/certs/random
      fragment_size = 1024

      include_length = yes

      #check_crl = yes

      check_cert_cn = %{User-Name}
      }
}
```

private_key_password est le mot de passe du certificat serveur (par défaut la valeur est whatever on peut le modifier en éditant le fichier CA.svr)
private_key_file et **certificate_file** est le chemin vers le certificat serveur.
CA_file est le chemin pour le certificat racine.
dh_file et **random_file** sont les chemins vers les fichiers aléatoires qu'on a générés précédemment.
check_cert_cn permet de vérifier que le nom d'utilisateur fournit par le client est le même que celui dans le certificat (utile car certain driver propose de choisir le nom d'utilisateur et le certificat ex : *Intel Proset* ou *Netgear Utility*).
check_crl est le seul paramètre qu'on laisse commenter, il permet de vérifier si le certificat n'a pas été révoqué.

Fichier *clients.conf*
Ce fichier permet de définir la liste des AP que l'on autorise à accéder au serveur Radius. Le serveur et l'AP partagent un secret (une clé) pour crypter les données.
Par défaut on autorise le localhost (127.0.0.1) avec comme secret : *testing123* (pour réaliser des tests en local)
Pour rajouter notre borne Wifi avec comme adresse IP 192.168.0.228

```
client 192.168.0.228 {
      secret      = bonjour
      shortname   = MaBorneWifi
      nastype     = other
}
```

On aura une clé partagée entre l'AP et le serveur qui sera bonjour et on lui donne le nom *MaBorneWifi* via shortname (utile pour le debug).
On a aussi la possibilité de déclarer un réseau entier, par exemple pour autoriser toutes les machines du réseau 192.168.23.0 à se connecter au radius

```
client 192.168.23.0/24 {…
```

Fichier *users*
Dans ce fichier, on défini la liste des utilisateurs qu'on autorise. On a précédemment géré le certificat pour l'utilisateur client, on ajoute donc à la fin du fichier.

```
"client" Auth-Type := EAP
```

On spécifie donc que l'utilisateur "client" peut s'authentifier avec la méthode EAP (donc EAP-TLS, EAP-PEAP, EAP-TTLS ...). Pour forcer un type, il faut utiliser l'attribut EAP-TYPE, par exemple si on veut que l'utilisateur ne fasse que de l'EAP-TLS, il faut mettre alors

```
"client" Auth-Type := EAP, EAP-Type := EAP-TLS
```

Pour chaque utilisateur, on doit ajouter une ligne dans ce fichier.
On peut utiliser une base de donnée pour gérer la liste des NAS autorisés (se reporter à l'annexe « Allons plus loin »)

Fichier *radiusd.conf*
C'est le fichier principal de configuration de FreeRadius, on a rien de spécial à configurer pour le moment.

III-4/ Lancement du daemon FreeRadius

Pour être sur que ça marche bien on lance le daemon avec le debug

```
radius2:~# radiusd -X -A &
Starting - reading configuration files ...
reread_config:  reading radiusd.conf
[...]
Listening on authentication *:1812
Listening on accounting *:1813
Listening on proxy *:1814
Ready to process requests.
```

Notre daemon s'est bien lancé ! Si votre daemon ne se lance pas, reportez vous à l'annexe « Résolution des erreurs ».

Pour arrêter le Radius, il suffit de taper

```
radius2:~# killall radiusd
```

Il faut à présent configurer la borne Wifi et les postes clients (se reporter aux autres guides d'installation).

Maintenant si vous vous connectez à la borne Wifi, ça marche ! (Vous pouvez aller voir les fichiers logs joints)

IV/ Installation de Freeradius avec EAP-TLS et MySQL

Maintenant que Freeradius et EAP-TLS marche, on va rajouter le support de MySQL à la place du fichier users car ça devient vite dur à gérer. Free radius supporte plusieurs bases de données MySQL, PostgreSQL, Oracle. Leur fonctionnement est le même (la structure des tables est identique).

IV-1/ Ajout du support de MySQL à FreeRadius

On commence par installer le serveur MySQL + le serveur Web (apache + php) pour la gestion via phpmyadmin

```
apt-get install mysql-server apache php4 php4-mysql phpmyadmin
```

On installe maintenant la bibliothèque « libmysqlclient12-dev »

```
apt-get install libmysqlclient12-dev
```

Il faut recompiler FreeRadius donc on lance un *make clean* et on recommence comme à la page 7(*./configure … make && make install*).
Vérifier que la configuration le module MySQL est bien pris en compte comme pour EAP-TLS.

```
config.status: creating Makefile
configure: configuring in ./types/rlm_sql_mysql
configure: running /bin/sh './configure' --prefix=/usr/local  '--
prefix=/usr/local' '--sysconfdir=/etc/' '--enable-ltdl-install=no' '--
cache-file=/dev/null' '--srcdir=.' 'CFLAGS=-g -O2 -D_REENTRANT
-D_POSIX_PTHREAD_SEMANTICS -DOPENSSL_NO_KRB5 -Wall -D_GNU_SOURCE -g
-Wshadow -Wpointer-arith -Wcast-qual -Wcast-align -Wwrite-strings
-Wstrict-prototypes -Wmissing-prototypes -Wmissing-declarations
-Wnested-externs -W -Wredundant-decls -Wundef' --cache-file=/dev/null --
srcdir=.
checking for gcc... gcc
checking for C compiler default output file name... a.out
checking whether the C compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables...
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ANSI C... none needed
```

Sinon se reporter à l'annexe « Résolution des erreurs »

Il faut qu'on modifie la configuration du serveur radius, on édite les fichier *radiusd.conf* et *sql.conf*

Fichier *sql.conf*

```
 # Database type
        # Current supported are: rlm_sql_mysql, rlm_sql_postgresql,
        # rlm_sql_iodbc, rlm_sql_oracle,
        # rlm_sql_unixodbc, rlm_sql_freetds
        driver = "rlm_sql_mysql"

        # Connect info
        server = "localhost"
        login = "root"
        password = ""

        # Database table configuration
        radius_db = "radius"
```

On utilise une base de données MySQL, donc on prend le driver rlm_sql_mysql. Il n'y a pas de mot de passe par défaut pour l'accès.

Fichier *Radiusd.conf*

Dans **authorize {**
Il faut enlever le # devant sql et commenter files

```
authorize {
      [...]
      #
      #  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
```

IV-

2/ Installation de la base de données

Nous utilisons le prompt MySQL, mais vous pouvez utiliser un gestionnaire de base de données connu : PhpMyAdmin. Il faut simplement copier les commandes en gras.

```
radius2:~/certs# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
mysql> CREATE DATABASE radius;
Query OK, 1 row affected (0.00 sec)
mysql> use radius
Database changed
mysql> source /root/freeradius-
1.0.2/src/modules/rlm_sql/drivers/rlm_sql_mysql/db_mysql.sql
```

Vous avez maintenant 8 nouvelles tables. Voilà un petit descriptif des tables, on utilisera juste les tables *radgroupcheck* et *usergroup*.

```
mysql> SHOW TABLES;
+------------------+
| Tables_in_radius |
+------------------+
| nas              |
| radacct          |
| radcheck         |
| radgroupcheck    |
| radgroupreply    |
| radpostauth      |
| radreply         |
| usergroup        |
+------------------+
8 rows in set (0.00 sec)
```

Table **nas** qui permet de remplacer le fichier client.conf en stockant la liste des NAS autorisé
Table **radacct** qui stocke les informations que retourne le NAS quand on fait de l'accounting
Table **radcheck** qui permet de vérifier une option d'un utilisateur (par exemple le mot de passe quand on utilise PEAP ou TTLS)
Table **radgroupcheck** même chose que radcheck mais pour une option de groupe.
Table **radgroupreply** qui permet de retourner une option de groupe (un numéro de Vlan par ex)
Table **radpostauth** qui stocke chaque authentification réussie.
Table **radreply** qui permet de retourner une option pour l'utilisateur.
Table **usergroup** qui permet de faire la liaison entre le nom d'utilisateur et son groupe.

Quand on utilise FreeRadius avec une base de données, la gestion des utilisateurs est un peu différente, chaque utilisateur est rattaché un groupe. Ce qui fait qu'il y a les options de groupe et les options pour l'utilisateur.

On va donc rajouter l'utilisateur *Client* qui va appartenir au groupe *WifiSecu*

```
mysql> INSERT INTO usergroup VALUES ('','Client','WifiSecu');
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM usergroup;
                  +----+----------+-----------+
                  | id | UserName | GroupName |
                  +----+----------+-----------+
                  |  1 | Client   | WifiSecu  |
                  +----+----------+-----------+
1 row in set (0.00 sec)
```

On va maintenant rajouter les options du groupe qui sont que les utilisateurs du groupe doivent utiliser EAP pour s'authentifier.

```
mysql> INSERT INTO radgroupcheck VALUES ('','WifiSecu','Auth-Type',':=','EAP');
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM radgroupcheck;
                     +----+-----------+-----------+----+-------+
                     | id | GroupName | Attribute | op | Value |
                     +----+-----------+-----------+----+-------+
                     |  1 | WifiSecu  | Auth-Type | := | EAP   |
                     +----+-----------+-----------+----+-------+
1 row in set (0.00 sec)
```

Maintenant que la configuration est terminée, on relance le daemon Radius

V/ Annexes

V-1/ Allons plus loin...

Voila quelques modifications à réaliser pour augmenter la sécurité de votre serveur radius.
Créer un groupe et un utilisateur pour le rattacher au daemon FreeRadius (modification du fichier *radiusd.conf* vers les lignes 110.)

On peut aussi créer un nouvel utilisateur dans MySQL avec des droits uniquement sur la base **radius** (penser à modifier fichier *sql.conf*)

Remplacement des fichiers dh et random :

Dans le paragraphe Installation des Certificats sur le serveur, pendant la configuration de EAP-TLS, on crée 2 fichiers (dh & random) avec des donnés aléatoires, pour générer ces fichiers on a utilisé la fonction date, qui est loin d'être aléatoire.
Pour le fichier dh, on utilise la fonction **dh (Diffie-Hellman)** OpenSSL.
Il faut se mettre dans le dossier où se trouvent les fichiers et taper la commande suivante

```
openssl dhparam -check -text -5 512 -out dh
```

On obtient donc

```
radius2:~# cd /etc/raddb/certs/
radius2:/etc/raddb/certs# openssl dhparam -check -text -5 512 -out dh
Generating DH parameters, 512 bit long safe prime, generator 5
This is going to take a long time
........+.+.....................................................
......+.............+.....................................................
.....................................................................+..+.....
.+.........+.....+......+........+.......+...+..+.......+..+...........
..........+......+.+............+.............................+.....+
.+.....................................................................
...................+............+.....................................
.............................................+.........................
..............+.................+..................................+..
.....................+..+.....................................................
..................+.............+.....................................
.......+.............+..+......+.......................+....+........
.................+.........................+................+.......
..............................+....+.+........+.................+.....
.........+.........................+.......+.........................+
.....................................................+................
...+.....................................+.............................
.............................................+.........................
.........................++*++*++*++*++*++*
DH parameters appear to be ok.
```

On a généré un fichier de 512bits
Pour le fichier random, on utilise le morceau de code suivant qu'on enregistre dans le dossier */etc/raddb/certs/random.c*

```
#include <stdio.h>
#include <openssl/rand.h>
// you will need to compile it with openssl lib
// $ gcc -lcrypto
main (void) {
unsigned char buf[100];
if (!RAND_bytes(buf, 100)) {
// the usual md5(time+pid)
}
printf("Random : %s\n", buf);
}
```

Puis on le compile avec gcc

```
gcc random.c -o random -lcrypto
```

Remplacement du fichier clients.conf par une base de données :

Depuis la version 1.0.1, on a la possibilité de remplacer le fichier qui permet de gérer les NAS par une table dans une base de données qui permet d'avoir une gestion plus facile des NAS (via une interface web par exemple)

On modifie la configuration du module sql.
Dans le fichier sql.conf à la fin du fichier, on enlève le commentaire devant *readclient=yes*

```
      #
      # Set to 'yes' to read radius clients from
the database ('nas' table)
      readclients = yes
}
```

Il faut maintenant rajouter notre NAS dans la base de données.

```
mysql> INSERT INTO nas VALUES
('','192.168.0.228','MaBorneWifi','other','1812','b
onjour','public','MonNas');
Query OK, 1 row affected, 1 warning (0.01 sec)
```

On a rajouté la même configuration que avec le fichier client.conf, un NAS d'adresse IP 1*92.168.0.228*, avec comme nom *MaBorneWifi* et comme clé partagée *bonjour*.

```
mysql> SELECT * FROM nas;
+----+---------------+-------------+-------+-------+---------+-----------+------------+
| id | nasname       | shortname   | type  | ports | secret  | community | description|
+----+---------------+-------------+-------+-------+---------+-----------+------------+
|  1 | 192.168.0.228 | MaBorneWifi | other |  1812 | bonjour | public    | MonNas     |
+----+---------------+-------------+-------+-------+---------+-----------+------------+
```

Une brève explication sur les différents champs de la table.
Champ **id** : Numéro unique qui s'auto-incrémente.
Champ **nasname** : Adresse IP du NAS (on peut spécifier une réseau en mettant 192.168.23.0**/24** par exemple)
Champ **shortname** : Nom de notre NAS
Champ **type** : le type de NAS pour l'utilisation dictionnaire de la marque par défaut **other**
Champ **ports** : le(s) port(s) utilisé(s) pour communiquer avec le NAS par défaut **1812**
Champ **secret** : la clé partagée entre le NAS et le serveur Radius.
Champ **community** : la communauté SNMP du NAS (pas obligatoire)
Champ **description** : une petite description du nas (par exemple : La borne Wifi a coté de la machine à café.)

Par contre, comme avec les fichiers de configuration, il faut que l'on relance le daemon FreeRadius, pour que les NAS ajoutés soit pris en compte.

Nous avons vu juste une partie des possibilités de FreeRadius, si vous êtes motivés vous pouvez remplacer la base de données MySQL par un annuaire LDAP ou encore rajouter le support PEAP et/ou TTLS.

V-2/ Résolution des erreurs

Important : Ne pas oublier de faire un make clean avant de refaire une configuration ou une compilation

A- ERREUR à la compilation de OpenSSL

Il

```
[....]
../include/openssl/lhash.h:184: error: erreur d'analyse syntaxique before
"FILE"
../include/openssl/lhash.h:185: error: erreur d'analyse syntaxique before
"FILE"
../include/openssl/lhash.h:186: error: erreur d'analyse syntaxique before
"FILE"
Dans le fichier inclus à partir de cryptlib.h:70,
          à partir de cryptlib.c:61:
../include/openssl/err.h:84:19: errno.h : Aucun fichier ou répertoire de ce
type
In file included from cryptlib.h:70,
                 from cryptlib.c:61:
../include/openssl/err.h:265: error: erreur d'analyse syntaxique before '*'
token
cryptlib.c: Dans la fonction « CRYPTO_thread_id »:
cryptlib.c:381: attention : implicit declaration of function `getpid'
cryptlib.c: Dans la fonction « OpenSSLDie »:
cryptlib.c:512: attention : implicit declaration of function `fprintf'
cryptlib.c:512: error: `stderr' undeclared (first use in this function)
cryptlib.c:512: error: (Each undeclared identifier is reported only once
cryptlib.c:512: error: for each function it appears in.)
cryptlib.c:515: attention : implicit declaration of function `abort'
make[1]: *** [cryptlib.o] Erreur 1
make[1]: Leaving directory `/root/openssl-0.9.7e/crypto'
make: *** [sub_all] Erreur 1
```

apt-get install libc6-dev

manque la bibliothèque « libc6-dev » donc installez la et relancez la compilation.

B-1 Erreur de génération de certificat

```
Debian:~/certs# ./CA.root
bash: ./CA.root: Permission non accordée
```

Problème au niveau des droits sur le fichier, pour corriger le problème

On tape la commande suivante

```
chmod 777 CA.root
```

B-2

```
Debian:~/certs# ./CA.root
**********************************************************************
Creating self-signed private key and certificate
When prompted override the default value for the Common Name field
**********************************************************************

Generating a 1024 bit RSA private key
...++++++
...++++++
writing new private key to 'newreq.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]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:
Email Address []:
**********************************************************************
Creating a new CA hierarchy (used later by the ca command) with the
certificate
and private key created in the last step
**********************************************************************

./CA.root: line 22: CA.pl: command not found
**********************************************************************
Creating ROOT CA
Error opening input file demoCA/cacert.pem
==========
demoCA/cacert.pem: No such file or directory
Error opening input file root.p12
root.p12: No such file or directory
Error opening Certificate root.pem
430:error:02001002:system library:fopen:No such file or
directory:bss_file.c:278:fopen('root.pem','r')
430:error:20074002:BIO routines:FILE_CTRL:system lib:bss_file.c:280:
unable to load certificate
```

Éditer le fichier CA.root (et les autres fichiers) et vérifier le chemin SSL.

```
#!/bin/sh
SSL=/usr/local/openssl-certgen/
export PATH=${SSL}/bin/:${SSL}/ssl/misc:${PATH}
export LD_LIBRARY_PATH=${SSL}/lib
```

Il doit correspondre au préfixe de la configuration de la compilation de OpenSSL.

C-2

Vous avez déjà un certificat avec le même ***Common Name,*** il est impossible d'avoir 2 certificats pour le même nom d'utilisateur. Relancer en changeant le nom d'utilisateur.

```
Certificate is to be certified until Feb 14 16:46:10 2006 GMT (365
days)
Sign the certificate? [y/n]:y
failed to update database
TXT_DB error number 2
No certificate matches private key
656:error:0D07207B:asn1 encoding routines:ASN1_get_object:header too
long:asn1_lib.c:140:
unable to load certificate
657:error:0906D06C:PEM routines:PEM_read_bio:no start
line:pem_lib.c:637:Expecting: TRUSTED CERTIFICATE
```

D-1 FreeRadius

Au lancement du make, vous avez le message d'erreur suivant :

```
Debian:/root/freeradius-1.0.2# make
make[1]: Entering directory `/root/freeradius-1.0.2'
Making all in libltdl...
make[2]: Entering directory `/root/freeradius-1.0.2/libltdl'
make[2]: *** Pas de règle pour fabriquer la cible « all ». Arrêt.
make[2]: Leaving directory `/root/freeradius-1.0.2/libltdl'
make[1]: *** [common] Erreur 1
make[1]: Leaving directory `/root/freeradius-1.0.2'
make: *** [all] Erreur 2
```

Il vous manque la bibliothèque « libltdl3-dev » il suffit de l'installer et de relancer la configuration (donc ./configure & après le make)

```
apt-get install libltdl3-dev
```

D-2 Erreur avec EAP-TLS

Erreur pendant la configuration de la compilation du FreeRadius (./configure...)

```
config.status: creating Makefile
configure: configuring in ./types/rlm_eap_tls
configure: running /bin/sh './configure' --prefix=/usr/local  '--
prefix=/usr/local' '--sysconfdir=/etc/' '--disable-shared' '--
enable-ltdl-install=no' '--cache-file=/dev/null' '--srcdir=.'
'CFLAGS=-g -O2 -D_REENTRANT -D_POSIX_PTHREAD_SEMANTICS
-DOPENSSL_NO_KRB5 -Wall -D_GNU_SOURCE -g -Wshadow -Wpointer-arith
-Wcast-qual -Wcast-align -Wwrite-strings -Wstrict-prototypes
-Wmissing-prototypes -Wmissing-declarations -Wnested-externs -W
-Wredundant-decls -Wundef' --cache-file=/dev/null --srcdir=.
checking for gcc... gcc
checking for C compiler default output file name... a.out
checking whether the C compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables...
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ANSI C... none needed
configure: WARNING: silently not building rlm_eap_tls.
configure: WARNING: FAILURE: rlm_eap_tls requires:  OpenSSL.
configure: creating ./config.status
```

Il vous manque la bibliothèque « libssl-dev » il suffit de l'installer et de relancer la configuration

```
apt-get install libssl-dev
```

D-3

```
 eap: ignore_unknown_eap_types = no
 eap: cisco_accounting_username_bug = no

[1]+  Erreur de segmentation  radiusd -X -A
```

Vous utilisez le module eap mais à la compilation/configuration, le module n'a pas été compilé, reportez vous à « erreur avec EAP-TLS ».

E-1

```
rlm_eap: Failed to link EAP-Type/peap: /usr/local/lib/rlm_eap_peap.a:
invalid ELF header
radiusd.conf[9]: eap: Module instantiation failed.
radiusd.conf[1864] Unknown module "eap".
[1]+  Exit 1                  radiusd -X -A
```

Vous avez oublié de rajouter *–disable-shared* à la fin du *./configure*, relancer un *./configure...* puis relancer une compilation/installation (ne pas oublier le *make clean*)

E-2 Erreur avec snmp

Au début de la configuration de la compilation, Vous avez le Warning suivant :

```
appending configuration tag "CXX" to libtool
appending configuration tag "F77" to libtool
configure: WARNING: snmpget not found -
Simultaneous-Use and checkrad.pl may not work
 WARNING: snmpwalk not found - Simultaneous-Use and
checkrad.pl may not work
```

Il vous manque la bibliothèque « snmp » il suffit de l'installer et de relancer la configuration

```
apt-get install snmp
```

F- Erreur avec MySQL

Uniquement si vous utilisez MySQL avec FreeRadius :

Il

```
config.status: creating Makefile
configure: configuring in ./drivers/rlm_sql_mysql
configure: running /bin/sh './configure' --prefix=/usr/local  '--
prefix=/usr/local' '--sysconfdir=/etc/' '--disable-shared' '--enable-
ltdl-install=no' '--cache-file=/dev/null' '--srcdir=.' 'CFLAGS=-g -O2
-D_REENTRANT -D_POSIX_PTHREAD_SEMANTICS -DOPENSSL_NO_KRB5 -Wall
-D_GNU_SOURCE -g -Wshadow -Wpointer-arith -Wcast-qual -Wcast-align
-Wwrite-strings -Wstrict-prototypes -Wmissing-prototypes -Wmissing-
declarations -Wnested-externs -W -Wredundant-decls -Wundef' --cache-
file=/dev/null --srcdir=.
checking for gcc... gcc
checking for C compiler default output file name... a.out
checking whether the C compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables...
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ANSI C... none needed
checking how to run the C preprocessor... gcc -E
checking for mysql_config... no
checking for compress in -lz... yes
checking for mysql/mysql.h... no
configure: WARNING: mysql headers not found.  Use --with-mysql-include-
dir=<path>.
configure: WARNING: sql submodule 'mysql' disabled
configure: creating ./config.status
```

vous manque la bibliothèque « libmysqlclient12-dev » il suffit de l'installer et de relancer la configuration

```
apt-get install  libmysqlclient12-dev
```

G- Erreur au lancement du daemon radiusd

```
rlm_sql (sql): Could not link driver rlm_sql_mysql: rlm_sql_mysql.a:
cannot open shared object file: No such file or directory
rlm_sql (sql): Make sure it (and all its dependent libraries!) are in
the search path of your system's ld.
radiusd.conf[14]: sql: Module instantiation failed.
radiusd.conf[1766] Unknown module "sql".
[1]+  Exit 1                  radiusd -X -A
```

Vous utilisez le module sql mais à la compilation/configuration, le module n'a pas été compilé reportez vous à « erreur avec MySQL ».

H-

Vous avez bien installé la bibliothèque « libmysqlclient12-dev » mais vous avez l'erreur suivante pendant la compilation

```
`/root/freeradius-1.0.2/src/modules/rlm_sql/drivers/rlm_sql_mysql'
gcc  -g -O2 -D_REENTRANT -D_POSIX_PTHREAD_SEMANTICS -DOPENSSL_NO_KRB5
-Wall -D_GNU_SOURCE -DNDEBUG  -I../.. -I../../../../include
-I/root/freeradius-1.0.2/libltdl -c sql_mysql.c -o sql_mysql.o
sql_mysql.c:39:20: errmsg.h : Aucun fichier ou répertoire de ce type
sql_mysql.c:40:19: mysql.h : Aucun fichier ou répertoire de ce type
sql_mysql.c:47: error: erreur d'analyse syntaxique before "MYSQL"
sql_mysql.c:47: attention : pas de point virgule à la fin de la
structure ou de l'union
[....]
sql_mysql.c:397: attention : implicit declaration of function
`mysql_affected_rows'
make[10]: *** [sql_mysql.o] Erreur 1
make[10]: Leaving directory
`/root/freeradius-1.0.2/src/modules/rlm_sql/drivers/rlm_sql_mysql'
make[9]: *** [common] Erreur 1
make[9]: Leaving directory `/root/freeradius-
1.0.2/src/modules/rlm_sql/drivers'
make[8]: *** [static] Erreur 2
make[8]: Leaving directory `/root/freeradius-
1.0.2/src/modules/rlm_sql/drivers'
make[7]: *** [common] Erreur 1
make[7]: Leaving directory `/root/freeradius-1.0.2/src/modules/rlm_sql'
make[6]: *** [static] Erreur 2
make[6]: Leaving directory `/root/freeradius-1.0.2/src/modules/rlm_sql'
make[5]: *** [common] Erreur 1
make[5]: Leaving directory `/root/freeradius-1.0.2/src/modules'
make[4]: *** [all] Erreur 2
make[4]: Leaving directory `/root/freeradius-1.0.2/src/modules'
make[3]: *** [common] Erreur 1
make[3]: Leaving directory `/root/freeradius-1.0.2/src'
make[2]: *** [all] Erreur 2
make[2]: Leaving directory `/root/freeradius-1.0.2/src'
make[1]: *** [common] Erreur 1
make[1]: Leaving directory `/root/freeradius-1.0.2'
make: *** [all] Erreur 2
```

C'est un problème au niveau du Makelife, supprimer le dossier freeradius-1.0.2 et extraire à nouveau l'archive freeradius-1.0.2.taz.gz, puis il suffit de relancer un ./configure ... et compiler.