FreeRADIUS

Introducción

Este manual explica la instalación y configuración de un servidor RADIUS IdP (identity provider) y SP (service provider) para participar en la jerarquía de los servidores RADIUS del servicio **eduroam**. Autenticación: EAP-TTLS + PAP; cifrado: WPA(2) + TKIP. El manual está previsto para Ubuntu server 8.04 con freeradius-2.1.6 (pero debería funcionar para todas las versiones freeradius 2.1.x).

En este ejemplo se usa un certificado intermediate (de Verisign). Antes de usarlo hay que añadir el certificado intermediate al certificado root de la CA (certificate authority) correspondiente. Si se usan certificados de RedIRIS este paso no hace falta.

Preparación del certificado

generación de un clave RSA:

```
$ openssl genrsa -out eduroam.key 102
```

generación petición csr (certificat request)

```
$ openssl req -new -nodes -key eduroam.key -out eduroam.cs
```

Este certificate request se envía a la CA. La CA genera el certificado.

El servidor freeradius trabaja una clave cifrada. Por eso hay que convertir la clave privada antes de que se pueda usarlo.

```
$ openssl rsa -in eduroam.key -des3 -out server_crypt.key
```

Hay que configurar esta clave en el fichero eap.conf (parámetro private_key_password).

Incluir el certificado intermediate en el certificado root de Verisign.

```
$ cat ca.pem cert_verisign_intermediate.pem > ca_bundle.pe
```

Instalación freeradius-server-2.1.x

- Baja freeradius-server-2.1.x.tar.gz http://freeradius.org/download.html
- Deshaz el fichero freeradius-server-2.1.x.tar.gz y cambia al directorio nuevo freeradius-server-2.1.x:

```
$ tar vzxf freeradius-server-2.1.x.tar.gz
$ cd freeradius-server-2.1._x
```

- Instala el servidor freeradius:

```
$ ./configure
$ make
$ sudo make install
$ sudo ldconfig
```

- Comprueba si han aparecido errores durante la compilación.

```
$ grep -ri error config.log | sort | uniq
En Ubuntu salen las líneas siguientes que se puede ignorar. Si aparecen más u otras notificaciones hay que
comprobar si se cumplen todas las dependencias del freeradius.
conftest.c:106: error: void value not ignored as it ought to be
conftest.c:110: error: 'struct utmpx' has no member named 'ut_xtime'
conftest.c:119: error: too many arguments to function 'ctime_r'
conftest.c:25: error: 'not' undeclared (first use in this function)
conftest.c:25: error: (Each undeclared identifier is reported only once
conftest.c:25: error: expected ';' before 'big'
conftest.c:25: error: for each function it appears in.)
conftest.c:34:22: error: resource.h: No such file or directory
conftest.c:51:21: error: winsock.h: No such file or directory
conftest.c:53:26: error: sys/security.h: No such file or directory
conftest.c:58:18: error: prot.h: No such file or directory
conftest.c:60:17: error: sia.h: No such file or directory
conftest.c:60:18: error: siad.h: No such file or directory
conftest.c:67:22: error: resource.h: No such file or directory
conftest.c:84:21: error: winsock.h: No such file or directory
conftest.c:86:26: error: sys/security.h: No such file or directory
conftest.c:8:28: error: ac_nonexistent.h: No such file or directory
conftest.c:91:18: error: prot.h: No such file or directory
conftest.c:93:17: error: sia.h: No such file or directory
conftest.c:93:18: error: siad.h: No such file or directory
conftest.cpp:19:28: error: ac_nonexistent.h: No such file or directory
/* Override any GCC internal prototype to avoid an error.
```

Arrancar el servidor freeradius

Para comprobar si la instalación ha ido bien, arranca el servidor radius en modo debug (genera mucha salida).

Cuando se arranca el servidor freeradius por primera vez, el servidor genera claves y certificados que se reemplazan durante la configuración con las claves y certificados que se han generado arriba.

```
$ radiusd -X -xx
```

Si el servidor arranca bien aparecen las líneas siguientes al final de la salida:

```
......

Wed May 20 15:55:11 2009 : Debug: Listening on authentication address * port 1812

Wed May 20 15:55:11 2009 : Debug: Listening on accounting address * port 1813

Wed May 20 15:55:11 2009 : Debug: Listening on proxy address * port 1814

Wed May 20 15:55:11 2009 : Debug: Ready to process requests.

Para el freeradius con CTRL-C
```

Configuración freeradius

Hay que adaptar unos cuantos ficheros de configuración:

nombre fichero	Componente Radius	descripción
`clients.conf`	SP	definición de las máquinas desde que el SP acepta requests
`eap.conf`	IdP	definición de la autenticación EAP

`proxy.conf`	SP	definición de los REALMs y de los servidores RADIUS para reenviar requests para non local REALMs
`users`	SP	definición usuario local `radius-test@DOMINIO.ORG`
`radiusd.conf`	SP	configuraciones generales
`modules/ldap`	IdP	configuración relacionado con la conexión al servidor LDAP
`modules/pap`	IdP	configuración relacionado con la autenticación PAP del túnel interno
`sites-available /eduroam`	SP/IdP	Configuración autenticación y autorización

Antes de hacer modificaciones en la configuración es necesario arrancar el servidor freeradius por primera vez con la configuración por defecto para verificar si es correcta la instalación.
Se puede encontrar un enlace para bajar los ficheros de configuración más abajo, en el capitulo "Configuración del servidor freeradius".

A continuación se muestra el contenido de estos ficheros.

`clients.conf`

El "shortname" sólo aparece en el fichero log. No tiene otra significación.

client 127.0.0.1 se usa para troubleshooting. Sin esta entrada no se puede ejecutar el comando #radtestá desde el mismo servidor.

client wlan-switch: definición de los clientes de la home organización. Podría ser un Switch o los Access Points (AP).

client rediris_flrs: definición del servidor RADIUS de RedIRIS. La contraseña se acuerda con RedIRIS.

client 127.0.0.1 {
ipaddr = 127.0.0.1
netmask = 32
secret = **CHANGE ME**
require_message_authenticator = no
shortname = loopback
nastype = other
}

client wlan-switch {
ipaddr = **IP Address wlan switch**
netmask = 32
secret = **CHANGE ME**
require_message_authenticator = no
nastype = other
shortname = wlan-switch
}

client rediris_flrs1 {
ipaddr = **IP Address Servidor RADIUS RedIRIS**
netmask = 32
secret = **CHANGE ME**
require_message_authenticator = no
shortname = rediris_flrs1
nastype = other
}

eap.conf

Definiciones relacionadas con el Extended Authentification Protocol (EAP).
Para (T)TLS hay que configurar el certificado del servidor.

eap {
default_eap_type = tls
timer_expire = 60
ignore_unknown_eap_types = no
cisco_accounting_username_bug = no
max_sessions = 2048

tls {
certdir = ${confdir}/certs
cadir = ${confdir}/certs
private_key_password = **CHANGE ME**
private_key_file = ${certdir}/**server_crypt.key**
certificate_file = ${certdir}/**server.pem**
CA_file = ${cadir}/**ca_bundle.pem**
dh_file = ${certdir}/dh
random_file = ${certdir}/random
cipher_list = "DEFAULT"
include_length = yes
check_crl = no
copy_request_to_tunnel = no
use_tunneled_reply = no

}

ttls {
default_eap_type = mschapv2
copy_request_to_tunnel = no
use_tunneled_reply = yes
}
}

proxy.conf

proxy server {
default_fallback = yes
}

home_server rediris_flrs1 {
type = auth+acct
ipaddr = **IP Address Servidor RADIUS RedIRIS**
port = 1812
secret = **CHANGE ME**
response_window = 20
zombie_period = 40
revive_interval = 60
status_check = status-server
check_interval = 30
num_answers_to_alive = 3
}

home_server_pool EDUROAM-FTLR {
type = fail-over
home_server = rediris_flrs1
}

realm **DOMINIO.ORG** {
nostrip
}

realm LOCAL {
nostrip
}

realm NULL {
nostrip
}

realm DEFAULT {
pool = EDUROAM-FTLR
nostrip
}

users

radius-test@**DOMINIO.ORG** Auth-Type := Reject
Reply-Message = "RADIUS OK"

radiusd.conf

```
prefix = /usr/local
exec_prefix = ${prefix}
sysconfdir = ${prefix}/etc
localstatedir = ${prefix}/var
sbindir = ${exec_prefix}/sbin
logdir = ${localstatedir}/log/radius
raddbdir = ${sysconfdir}/raddb
radacctdir = ${logdir}/radacct

name = radiusd

confdir = ${raddbdir}
run_dir = ${localstatedir}/run/radiusd

db_dir = ${raddbdir}

libdir = ${exec_prefix}/lib

pidfile = ${run_dir}/${name}.pid

user = freerad
group = freerad

max_request_time = 30

cleanup_delay = 5

max_requests = 1024

listen {
type = auth
ipaddr = *
port = 1812
}

listen {
type = acct
ipaddr = *
port = 1813
}

hostname_lookups = no

allow_core_dumps = no

regular_expressions = yes
extended_expressions = yes

log {
destination = files
file = ${logdir}/radius.log
syslog_facility = daemon
stripped_names = no
auth = no
auth_badpass = no
auth_goodpass = no
}

checkrad = ${sbindir}/checkrad

security {
max_attributes = 200
reject_delay = 1
status_server = yes
}

proxy_requests = yes
$INCLUDE proxy.conf
$INCLUDE clients.conf

thread pool {
start_servers = 5
max_servers = 32
min_spare_servers = 3
max_spare_servers = 10
max_requests_per_server = 0
}
```

modules {
$INCLUDE ${confdir}/modules/
$INCLUDE eap.conf
$INCLUDE sql.conf
$INCLUDE sql/mysql/counter.conf
}

instantiate {
exec
expr
expiration
logintime
}

$INCLUDE policy.conf
$INCLUDE sites-enabled/

modules/ldap

En este ejemplo se incluye el uso del atributo "mail" como identificador y también, comentado, un uso más adaptado al esquema "eduPerson".

ldap {
server = "ldap://LDAP-SERVER.DOMINIO.ORG"
port = 389

1. server = "ldaps://LDAP-SERVER.DOMINIO.ORG"
2. port = 636
   identity="cn=**CHANGE ME**,dc=**DOMINIO**,dc=**ORG**"
   password="**CHANGE ME**"
   basedn = "ou=**CHANGE ME**,dc=**DOMINIO**,dc=**ORG**"
   filter = "(mail=%{User-Name})"
3. filter = "(eduPersonPrincipalName=%{User-Name})"
   base_filter = ""
   access_attr = "mail"
4. access_attr = "eduPersonPrincipalName"
   password_attribute = "userPassword"
   groupname_attribute = ""
   groupmembership_filter = ""
   ldap_connections_number = 5
   timeout = 4
   timelimit = 3
   net_timeout = 1
   tls {
   start_tls = no
   }
   dictionary_mapping = ${confdir}/ldap.attrmap
   edir_account_policy_check = no
   }

modules/pap

El parámetro "auto_header" es necesario para que discrimine de forma automática si las contraseñas se almacenan en el LDAP usando algún tipo de hash y "sepa" seleccionar el adecuado.

pap {
auto_header = yes
}

sites-available/eduroam

Reemplaza "DOMINIO.ORG" por tu dominio.

Nota: hay que crear un "simbolic link" de sites-available/eduroam -> sites-enabled/eduroam.

authorize {
auth_log
preprocess
suffix
eap {
ok = return
}
files

1. solo requests para el "home" organización se autentifica vía LDAP
   if ("%{User-Name}" =~ /@**DOMINIO.ORG**/) {
   ldap
   }
   }
   authenticate {

```
Auth-Type PAP {
pap
}
unix
Auth-Type LDAP {
ldap
}
eap
}
preacct {
preprocess
acct_unique
suffix
files
}
accounting {
detail
unix
radutmp
attr_filter.accounting_response
}
session {
radutmp
}
post-auth {
exec
Post-Auth-Type REJECT {
attr_filter.access_reject
}
}
pre-proxy {
}
post-proxy {
eap
}
```

Haz una copia de seguridad de la configuración original

$ tar vcf backupdir/raddb_orig.tar /usr/local/etc/raddb

Configuración del servidor freeradius

Se puede bajar un tarball con las configuraciones: Freeradius_eduroam_config.rar.

Deshaz el fichero freeradius_eduroam_configfiles.rar

$ unrar x freeradius_eduroam_configfiles.tar.gz

Cambia al directorio freeradius_eduroam_configfiles

$ cd freeradius_eduroam_configfiles

Adapta los ficheros de configuración (REALMs, contraseñas, DOMINIO...).
Después ejecuta el comando "egrep -r "DOMINIO|CHANGE|ORG" *" para verificar si no se ha olvidado cambiar ningún parámetro (el comando NO debe tener salida).

Copia los ficheros de configuración a /usr/local/etc/raddb:

$ sudo cp clients.conf eap.conf proxy.conf radiusd.conf users /usr/local/etc/raddb

$ sudo cp modules/ldap modules/pap /usr/local/etc/raddb/modules/

*$ sudo rm /usr/local/etc/raddb/sites-enabled/**

*$ sudo rm /usr/local/etc/raddb/sites-available/**

$ sudo cp sites-available/eduroam /usr/local/etc/raddb/sites-available/

Crea un "simbolic link":

$ ln -s /usr/local/etc/raddb/sites-available/eduroam /usr/local/etc/raddb/sites-enabled/eduroam

Copia el certificado CA, el certificado del servidor y la clave del servidor a /usr/local/etc/raddb/certs/ (y comprueba los nombres en el fichero eap.conf).

Comprueba si se ha creado el grupo "freerad" y el usuario "freerad" durante la instalación del freeradius. Si no, créalos.

$ grep freerad /etc/group
(si existe el grupo "freerad" sale una línea como "freerad:x:112:")

Si el grupo no existe, créalo con el comando

$ groupadd freerad

$ grep freerad /etc/passwd

(si existe el usuario "freerad" sale una línea como
"freerad:x:107:112::/usr/local/etc/raddb:/bin/false")
(la uid (107) y la gid (112) pueden variar)

Si el usuario no existe, créalo con el comando

$ useradd -d /usr/local/etc/raddb -g freerad -s /bin/false freerad

Abre el fichero /etc/passwd con tu editor preferido y comprueba si aparece la línea siguiente:

freerad:x:107:112::/usr/local/etc/raddb:/bin/false
(importante es /usr/local/etc/raddb y /bin/false. Si sale algo diferente, cámbialo.)

Comprueba si la gid es correcta:
gid del usuario tiene que ser igual en los ficheros passwd y group

/etc/passwd: freerad:x:107:112::/usr/local/etc/raddb:/bin/false
/etc/group: freerad:x:112:

Cambia el owner/group del directorio /usr/local/etc/raddb

$ sudo chown -R freerad:freerad /usr/local/etc/raddb

Crea un directorio /usr/local/var/log/radius si no existe y cambia su "owner" a "freerad"

$ mkdir /usr/local/var/log/radius

$ sudo chown -R freerad:freerad radius/

Arranca el servidor

Primero se debe arrancar el servidor freeradius en el modo "foreground + debug" para comprobar si el servidor arranca limpio

$ sudo radiusd -X -xx

El comando radtest

Se puede comprobar si funciona el servidor radius desde la línea de comando con el comando radtestá. El comando radtest se instala automáticamente junto con el servidor freeradius.

Comprueba si funciona la autenticación para tu organización:

$ radtest radius-test@DOMINIO.ORG XXXXXX localhost 10 YYYYYY

(reemplaza DOMINIO.ORG por tu dominio, XXXXXX por la contraseña del usuario radius-test@DOMINIO.ORG (normalmente testá) y YYYYYY por el "secret" que está configurado en el fichero clients.conf)

Sending Access-Request of id 143 to 127.0.0.1 port 1812
User-Name = "radius-test@DOMINIO.ORG"
User-Password = "XXXXXX"
NAS-IP-Address = A.B.C.D
NAS-Port = 10
rad_recv: Access-Accept packet from host 127.0.0.1 port 1812, id=143, length=20
Reply-Message = "RADIUS OK"

Comprueba si funciona la autenticación contra RedIRIS:

$ /usr/bin/radtest radius-test@rediris.es test 127.0.0.1 10 secret
Sending Access-Request of id 182 to 127.0.0.1 port 1812
User-Name = "radius-test@rediris.es"
User-Password = "test"
NAS-IP-Address = 255.255.255.255
NAS-Port = 10
rad_recv: Access-Reject packet from host 127.0.0.1:1812, id=182, length=31
Reply-Message = "RADIUS OK"

Nota: en principio, todos los miembros de Eduroam disponen de una cuenta radius-test. Asá se puede ejecutar el comando radtest contra cada miembro de Eduroam. Sólo hay que cambiar el dominio en el comando radtestá.

Haz que el servidor arranca durante el boot

Nota: este init-script está creado para Ubuntu Linux. Hay que adaptar el script para otras distribuciones Linux.

El init script está incluido en el tarball de las configuraciones (freeradius_eduroam_configfiles/rc/freeradius).

Copia el init-script freeradius-init a /etc/init.d/:

$ sudo cp rc/freeradius /etc/init.d/freeradius

Cambia el owner a root:

$ sudo chown root:root /etc/init.d/freeradius

Activa el init script:

$ sudo update-rc.d freeradius defaults (este comando es específico para Ubuntu; mira cómo está el comando en tu distribución).

Arranca el servidor freeradius:

$ sudo /etc/init.d/freeradius start

Monitorización de los servidores RADIUS con Nagios

Nagios ([es un sistema open source de monitorización de redes, servicios y aplicaciones ampliamente utilizado.

Con el Nagios plug-in siguiente se comprueba si funciona la autenticación de usuarios de la home organización y de otra organización (RedIRIS).

#!/bin/sh

1. Marc Uebel <muebel@uoc.edu> 20090625

1. Chequeo si funcionan los servidores RADIUS para
2. el servicio EDUROAM

1. Requisitos: programa "radtest" (viene con freeradius)

1. Usage: check_radius_eduroam.sh servername
2. (reemplaza "servername" por el nombre del servidor RADIUS
3. que se debe comprobar)

1. a. i. 1. a. Pon tu REALM aquí #######

```
REALM=dominio.org

if ! $1)]; then
echo
echo usage: check_radius_eduroam.sh servername
echo
exit 3;
fi

SERVER=$1

STATE_OK=0
STATE_CRITICAL=2

STATE_OK_MESSAGE="RADIUS EDUROAM OK"
STATE_CRITICAL_LOCAL_MESSAGE="No funciona la autenticación de la cuenta radius-test@$REALM contra $SERVER"
STATE_CRITICAL_REDIRIS_MESSAGE="No funciona la autenticación de la cuenta radius-test@rediris.es contra $SERVER"

/usr/bin/radtest radius-test@$REALM test $SERVER 10 hola123 2>&1| grep "RADIUS OK" >/dev/null 2>&1
STAT1=$?
/usr/bin/radtest radius-test@rediris.es test $SERVER 10 hola123 2>&1| grep "RADIUS OK" >/dev/null 2>&1
STAT2=$?
```

if FreeRADIUS; then
echo $STATE_CRITICAL_LOCAL_MESSAGE
if FreeRADIUS; then
echo $STATE_CRITICAL_REDIRIS_MESSAGE
fi
exit $STATE_CRITICAL
else
if FreeRADIUS; then
echo $STATE_CRITICAL_REDIRIS_MESSAGE
exit $STATE_CRITICAL
fi
echo $STATE_OK_MESSAGE
exit $STATE_OK
fi

Depuración de conexiones en freeradius

Depurar en freeradius cuando el servidor ya se encuentra en producción, pudiera parecer una tarea imposible, pero no lo es. Eso sí, debemos olvidarnos de lanzar el servidor con la opción

```
-X
```

. A continuación explicaremos como depurar conexiones de usuarios con la utilidad

```
radmin
```

si bien dicha utilidad puede utilizarse para muchas cosas más, como saber el número de usuarios que se han conectado, o añadir clientes sin necesidad de reiniciar el servidor.

Como requisito para poder depurar "en caliente", necesitaríamos en primer lugar tener activado (si no lo teníamos ya) el

```
control-socket
```

en freeradius. Esta opción se consigue teniendo en el directorio

```
sites-enabled
```

el fichero

```
control-socket
```

que podemos enlazar desde el directorio

```
sites-available
```

. En principio no debería ser necesario modificar el contenido de este fichero, simplemente tener el fichero o enlazar a el mismo en ese directorio. Como unica modificación, si queremos, podríamos elegir el lugar donde se creará el socket unix que utilizará la utilidad

```
radmin
```

que ahora veremos.

Una vez activado el

```
control-socket
```

y reiniciado nuestro freeradius, podemos invocar a radmin de la siguiente manera (adaptando rutas a nuestro caso particular):

```
/usr/sbin/radmin -d /etc/raddb/
```

Nos aparecerá una shell, en la que podemos introducir la siguiente secuencia de comandos para depurar una conexión concreta de un

```
usuario@dominio.tld
```

:

```
# Fichero en el que guardaremos la salida
debug file myoutputfile.log

# Condición de depuración (hay que escapar parentesis y arroba):
debug condition '\(User-Name == usuario\@dominio.tld\)'
```

Y ya está. Cada vez que una petición con ese par valor/atributo pase por nuestro servidor, quedará reflejada en el fichero

```
myoutputfile.log
```

, que podremos encontrar en el directorio especificado en

```
radiusd.conf
```

usando la orden

```
logdir
```

.

NOTA IMPORTANTE: Viendo las opciones de depuración pudiéramos estar tentados de cambiar el nivel de depuración con

```
debug level
```

. Si hacemos esto estaremos cambiando el nivel de depuración general del servidor, y no se tendrá en consideración la condición especificada antes, por tanto si queremos realizar una depuración concreta, mejor no especificar un nivel de depuración.