Servicio de movilidad académica eduroam

Taller de instalación y configuración.

Ing. Luis Manuel Castro Jiménez

Contenido

Servidor Radius Local

El presente material tiene como objetivo que el personal técnico encargado de la instalación y configuración de eduroam dentro de la unidad educativa, es necesario conocimientos en Linux, directorio activo y base de datos.

Requisitos

- ✓ Servidor Linux Debian (Squeeze, Wheezy o Jessie), se sugiere contar con la última versión actualizada (Jessie) con la finalidad de evitar huecos de seguridad, teniendo en cuenta que las versiones anteriores pueden tener mayor compatibilidad con herramientas que puedan llegar a utilizar los administradores. Se utiliza Debian por ser la convención para servidores dentro de eduroam.

- ✓ Contar con los repositorios habilitados, se encuentra la lista de configuración en el directorio /etc/apt/sources.list en este archivo se configuran las fuentes donde se actualiza Debian. La liga donde se localizan las fuentes oficiales para Debian es:

    - https://www.debian.org/mirror/list.es.html

- ✓ Se requieren 2 direcciones IPs públicas, en cada red debe tener permiso en los puertos UDP: 1812, 1813, 1814, 1830 y el puerto TCP 2083.

- ✓ Cada institución configurará su servidor RADIUS Local, el dominio a utilizar deberá ser del tipo **<Institución>.edu.mx** institución serán las siglas que se utilizan para definir cada academia.

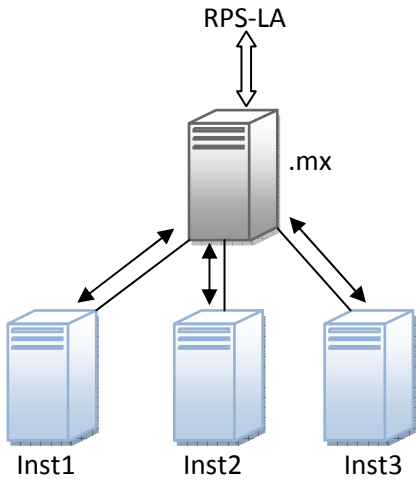


Figura 1.

En la figura 1 se muestra los niveles de Roaming Nacional que van de nivel local, que son las instituciones educativas, llegando al servidor federado nacional, este permite enlazar con los servidores de niveles superiores (regional, continental).

Instalación y configuración del Servidor RADIUS Local.

Instalación de software

Se requiere que el servidor cuente con todas las herramientas necesarias para una optima y acertada configuración.

1. Actualización del Sistema operativo.

```
#apt-get update && apt-get upgrade
```

2. Instalación de paquetes y librerías.

```
#apt-get install freeradius freeradius-ldap freeradius-mysql make pkg-config vim nmap mysql-server mysql-client
libssl-dev libgnutls-dev libsnmp-dev libmysqlclient-dev libldap-dev libtool libpcap0.8-dev gnutls-bin radsecproxy
```

3. Configuración y creación de certificados.

Creación de una entidad certificadora privada con firmas digitales autogeneradas, los certificados son emitidos para el servidor Radius.

1.1. Creamos una carpeta donde configuramos nuestros certificados, estos serán utilizado establecer el túnel seguro.

```
#mkdir /etc/eduroam-radsec
```

1.2. Ahora copiamos los certificados digitales y los configuramos:

```
#cp /usr/share/doc/freeradius/examples/certs/* /etc/eduroam-radsec/
#cd /etc/eduroam-radsec
#mkdir private newcerts
#touch index.txt
#echo '01' > serial
#ls /etc/eduroam-radsec/
```

El resultado motrado sera:

```
#bootstrap ca.cnf client.cnf Makefile server.cnf xpextensions index.txt private serial
```

1.3. Configuramos el archivo ca.cnf para la creación de CA privado.

```
#vi /etc/eduroam-radsec/ca.cnf
```

```
[ ca ]
default_ca              = CA_default

[ CA_default ]
dir                     = ./
certs                   = $dir
crl_dir                 = $dir/crl
database                = $dir/index.txt
new_certs_dir           = $dir
certificate             = $dir/ca.crt
serial                  = $dir/serial
crl                     = $dir/crl.crt
private_key             = $dir/ca.key
RANDFILE                = $dir/.rand
name_opt                = ca_default
cert_opt                = ca_default
```

```
default_days            = 3650
default_crl_days        = 30
default_md              = md5
preserve                = no
policy                  = policy_match

[ policy_match ]
countryName             = match
stateOrProvinceName     = match
organizationName  = match
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

[ policy_anything ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName  = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

[ req ]
prompt                  = no
distinguished_name      = eduroam
default_bits            = 2048
input_password          = <CLAVE>
output_password         = <CLAVE>
x509_extensions         = v3_req

[eduroam]
countryName             = MX
stateOrProvinceName     = <Ciudad>
localityName            = <Ciudad>
organizationName        = Institución
emailAddress            = tecnico@institucion.edu.mx
commonName              = Autoridad certificadora de <Institución>

[v3_ca]
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid:always,issuer:always
basicConstraints        = CA:true
extendedKeyUsage        = serverAuth, clientAuth

[v3_req]
basicConstraints        = CA:FALSE
subjectKeyIdentifier    = hash
extendedKeyUsage        = serverAuth, clientAuth
```

Cabe mencionar que el correo electrónico deberá ser el del responsable técnico de la institución que tiene su cargo la implementación del servicio de eduroam.

1.4. La selección de <clave secreta> de intercambio entre la institución y CUDI puede ser generada desde consola utilizando la herramienta mkpasswd. Esta clave proporcionará un alto nivel de seguridad para comunicarse entre servidores

```
#mkpasswd clave-aleatoria
2xghJGXyu6sds
```

1.5. Con OpenSSL para la creación del CA privado.

```
#openssl req -new -x509 -extensions v3_ca -keyout ca.key -out ca.crt -days 3650 -config ./ca.cnf
Durante la ejecución se verá lo siguiente.
Generating a 2048 bit RSA private key
.....
.....+++
.....+++
writing new private key to 'ca.key'
-----
```

1.6. Comprobamos que la autoridad certificadora fue creada correctamente.

```
#openssl x509 -in ca.crt -noout -text | grep Subject
```

Si esta correcta la configuración mostrará algo similar a:

```
Subject: C=MX, ST=Mexico, L=Mexico, O=Institución/emailAddress=tecnico@institucion.edu.mx,
CN=ftlr.cudi.edu.mx
Subject Public Key Info:
X509v3 Subject Key Identifier:
```

1.7. Creamos los archivos dh y random.

```
/etc/eduroam-radsec # openssl dhparam -out dh 1024
Generating DH parameters, 1024 bit long safe prime, generator 2
This is going to take a long time
.....+.....+.+.....+.....
/etc/eduroam-radsec #dd if=/dev/urandom of=./random count=10
10+0 registros leídos
10+0 registros escritos
5120 bytes (5.1 kB) copiados, 0.000402488 s, 12.7 MB/s
```

1.8. Configuramos el archive radius.cnf que solicitará el certificado digital en el Radius local, tomando como base el server.cnf.

```
#/etc/eduroam-radsec/cp server.cnf radius.cnf
#vim radius.cnf
```

1.8.1. La cuenta técnico@<institución>.edu.mx debe ser la cuenta del responsable en la institución.

1.8.2. Radius.<institución>.edu.mx este nombre debe coincidir con el Hostname del servidor de la institución. El Hostname se encuentra en el archivo /etc/hostname y en /etc/hosts .

```
[ ca ]
default_ca      = CA_default

[ CA_default ]
dir             = ./
certs           = $dir
crl_dir         = $dir/crl
database        = $dir/index.txt
new_certs_dir   = $dir
certificate     = $dir/radius.pem
serial          = $dir/serial
crl             = $dir/crl.pem
private_key     = $dir/radius.key
RANDFILE        = $dir/.rand
name_opt        = ca_default
```

```
cert_opt                = ca_default
default_days            = 3650
default_crl_days        = 30
default_md              = md5
preserve                = no
policy                  = policy_match

[ policy_match ]
countryName             = match
stateOrProvinceName     = match
organizationName        = match
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional


[ policy_anything ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

[req]
prompt                  = no
distinguished_name      = radius
default_bits            = 2048
input_password          = <clave secreta>
output_password         = <clave secreta>
x509_extensions         = v3_ca

[radius]
countryName             = MX
stateOrProvinceName     = <ciudad>
localityName            = <ciudad>
organizationName        = <Institución>
emailAddress            = técnico@<Institución>.edu.mx
commonName              = radius.<institución>.edu.mx

[v3_ca]
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid:always,issuer:always
basicConstraints        = CA:true
extendedKeyUsage        = serverAuth, clientAuth
```

1.9. Ejecutamos OpenSSL para la creación de una solicitud digital para Radius.

```
#openssl req -new -nodes -out radius.csr -keyout radius.key -config ./radius.cnf
Generating a 2048 bit RSA private key
.....+++
.....+++
writing new private key to 'radius.key'
-----
```

1.10. Se ejecuta nuevamente OpenSSL para la firma digital de la solicitud de certificado para el Radius. Renombramos el archivo radius.csr al de <institución>.csr.

```
/etc/eduroam-radsec #cp radius.csr <institución>.csr
```

1.11. Ejecutamos nuevamente OpenSSL para la firma digital de la solicitud de certificado para el Radius.

```
/etc/eduroam-radsec# openssl ca -out radius.cudi2.edu.mx.crt -config ./ca.cnf -infiles cudi.csr
Using configuration from ./ca.cnf
Enter pass phrase for .//ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4096 (0x1000)
        Validity
            Not Before: Mar  1 00:36:59 2016 GMT
            Not After : Feb 27 00:36:59 2026 GMT
        Subject:
            countryName               = MX
            stateOrProvinceName       = Mexico
            organizationName          = CUDI
            commonName                = radius.cudi.edu.mx
            emailAddress              = lcastro@cudi.edu.mx
Certificate is to be certified until Feb 27 00:36:59 2026 GMT (3650 days)
Sign the certificate? [y/n]:yes
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Configuración de claves entre servidor y realm local.

1. Proxy.conf

El servidor local y el servidor federado requieren establecer una conexión segura, configuramos el archivo proxy.conf, aquí encontramos lo necesario para conectar el servidor local con el servidor federado nacional (CUDI).

Para contar con un servicio de alta disponibilidad colocaremos un par de servidores en orden de prioridad, la configuración queda de la siguiente manera:

```
proxy server {
        default_fallback = yes
}
home_server ftlr {
        type            = auth+acct
        ipaddr          = 201.139.176.198
        port            = 1812,1813
        secret          = <Clave-secreta>
        response_windows        = 20
        zombie_period           = 40
        revive_interval         = 60
        status_check            = status-server
        check_interval          = 30
        num_answers_to_alive    = 3
}

home_server ftlr2 {
        type            = auth+acct
        ipaddr          = 148.243.81.203
        port            = 1812,1813
        secret          = <Clave-secreta>
        response_windows        = 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 = ftlr
        home_server = ftlr2
}

realm <institución>.edu.mx {
type = radius
authhost = LOCAL
accthost = LOCAL
}
realm LOCAL {
nostrip
}
realm null {
nostrip
}
realm "~.+$" {
        pool = eduroam-ftlr
        acc_pool= eduroam-ftlr
        nostrip
}
```

default_fallback = yes: Si una consulta de autenticación contiene un realm que no está explícitamente listado líneas abajo, entonces esto es reenviado a través del proxy al realm Default.
ipaddr = <ip-address> 201.139.176.198: IP del servicio Federado.
port = 1812, 1813: Estos puertos deben ser abiertos por el firewall para permitir que servidor Radius escuche las peticiones.
inst1.edu.xx: Reemplazar por el realm de su institución.

2. Client.conf

Una de las partes más importantes para interconectar a los servidores depende el archivo client.conf, aquí es necesaria la información entre el directorio activo y los puntos de acceso a través del servidor local.

```
client localhost {
        ipaddr          = 127.0.0.1
        secret          = <clave test>
        require_message_authenticator = no
        shortname       = localhost
        nastype         = other
}
#Federado principal
client 201.139.176.198 {
        secret          = <clave-compartida>
        shortname       = org-FTLR1-Mx
}
#Federado secundario
client 148.243.81.203 {
        secret          = <clave-compartida>
        shortname       = org-FTLR2-Mx
}
```

```
#Como se realiza la configuración de los clientes
client <nombre_institución_remota> {
      ipaddr = <IP_RADIUS_REMOTO>
      netmask =32
      require_message_authenticator=no
      secret = <secreto>
      shortname = org-<INST>
}

#Configuración de equipos de puntos de acceso
client localhost1 {
       ipaddr = xxx.xxx.xxx.xxx
       secret = <clave-secreta-local>
       shortname = AP1
}

# Se puede colocar toda una subred para identificar
client  AP-Edificio-org {
       ipaddr = xxx.xxx.0.0
       netmask = 12
       secret = <clave-1>
       shortname = AP2-edificio
}

# Se puede también colocar una subred desde una IP específica.
client  AP-Auditorio{
       ipaddr = zzz.zzz.zzz.100
       netmask = 24
       secret = <clave-2>
       shortname = AP3-auditorio
}
```

ipaddr=127.0.0.1, 127.0.1.1; IP del localhost
secret=<secreto-localhost>; Clave secreta del localhost
ipaddr=<IP_RADIUS_REMOTO>; Dirección IP del servidor Radius remoto
secret=<secreto>; Clave secreta compartida con el servidor Radius remoto
shortname=org-<INST>: Nombre de la institución o ubicación del dispositivo.

Configuración de cuentas para acceso en el Radius Local

1. Texto plano.

Radius nos permite configurar cuenta de prueba en texto plano, así como habilitar el uso de LDAP, editaremos el archivo users, aquí se encuentra la información para validad usuarios.

```
#vim /etc/freeradius/users

DEFAULT
User-Name = `%{User-Name}`,
Fall-Through = yes

user Cleartext-Password := "pass"

#DEFAULT Auth-Type = LDAP
#                Fall-Through = 1
#DEFAULT Auth-Type = SQL
```

```
#               Fall-Through = 1
```

User Cleartext-Password:= “pass” : Se utiliza para tener usuarios en texto plano.

#DEFAULT Auth-Type = LDAP
Fall-Through = 1:
Descomentar en caso que todos los usuarios estén almacenados en un servidor LDAP.

#DEFAULT Auth-Type = SQL
Fall-Through = 1:
Descomentar en caso que todos los usuarios estén almacenados en un servidor SQL.

Las configuraciones que se realicen dentro del servidor Radius es necesario reiniciar el demonio, en el caso de debían es del siguiente modo:

```
# invoke-rc.d freeradius stop
O
# service freeradius stop
```

invoke-rc.d es el comando preferido para programas de desarrollador paquetes ', de acuerdo a la página de comando man.

service tiene un única opción --status-all, que consulta el estado de todos los demonios disponibles.

Mientras service es el comando orientado al usuario, mientras que invoke-rc.d está ahí para otros usos.

Para revisar el funcionamiento de y validación de la configuración del Radius Local utilizaremos el freeradius en modo depuración:

```
#freeradius -X
```

El método para validar la cuenta eduroam es la siguiente:

```
radtest user@<Institución>.edu.mx pass 127.0.0.1 0 <secreto-localhost>
```

<secreto-localhost>: Es la clave secreta configurada en el archivo clients.conf

Radtest es una interfaz para radclient. Genera una lista de pares atributo / valor en base a los argumentos de línea de comandos, y se alimenta de estos en radclient.

Si la prueba es positiva, se recibirá un mensaje en el cual se visualiza el texto **Accept-accept**.

2. Instalación y configuración para validar a través de LDAP.

Se deben instalar los siguientes paquetes:

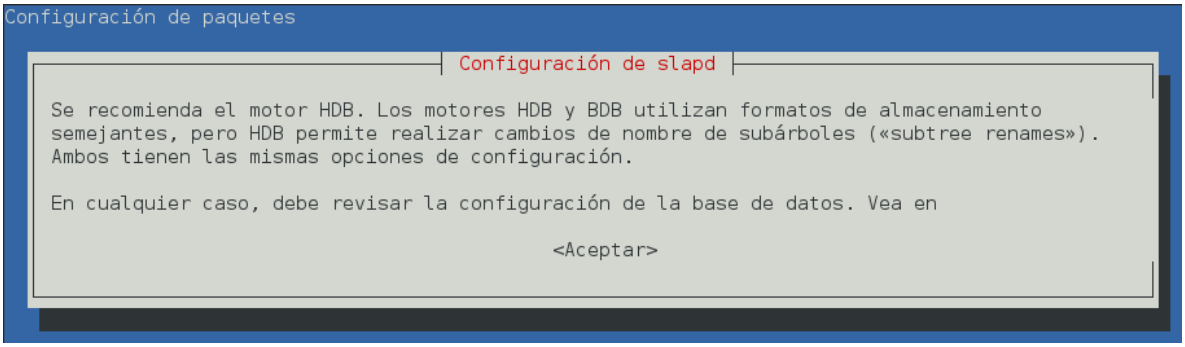
```
#apt-get install apache2 slapd ldap-utils phpldapadmin libapache2mod-php5
```

Se requiere cambiar los valores por defecto de LDAP por lo que ejecutamos el siguiente comando:

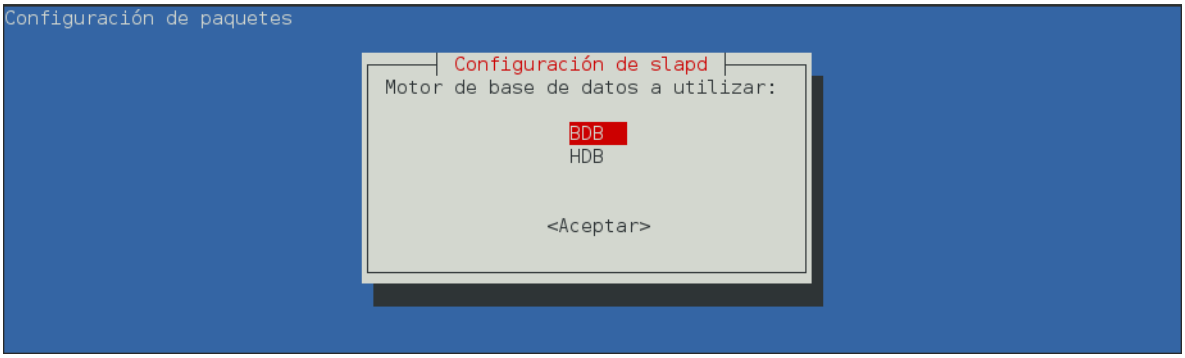
```
#dpkg-reconfigure slapd
```

Aparecerá la ventana de configuración en la cual ingresaremos la nueva configuración para LDAP de nuestra institución,

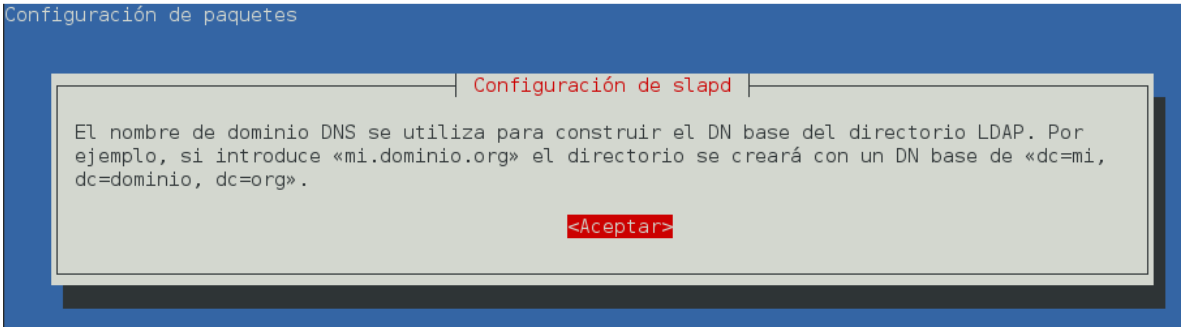
¿Desea omitir la configuración del servidor OpenLDAP? **No**



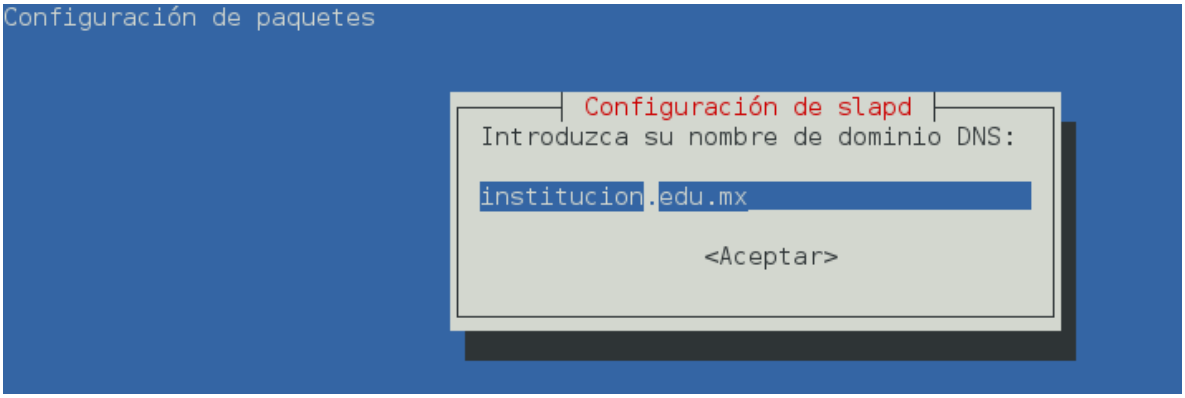
Para el caso de de eduroam utilizaremos un BDB.



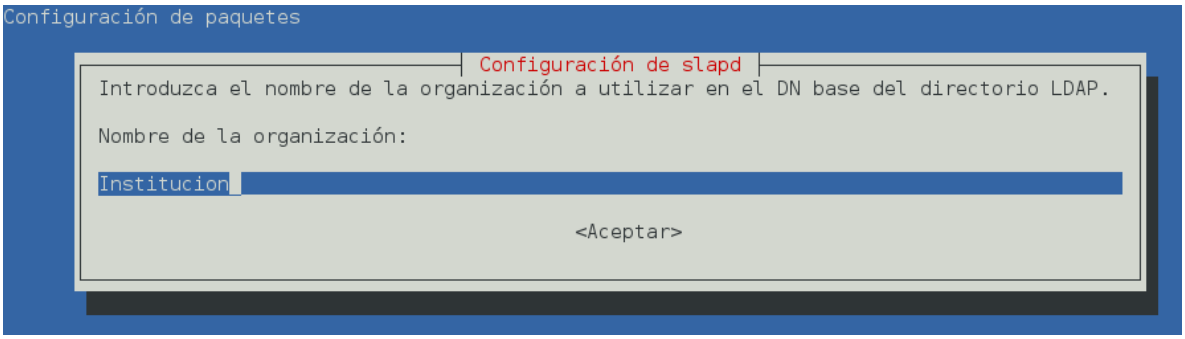
Seleccionamos el motor BDB y damos aceptar.



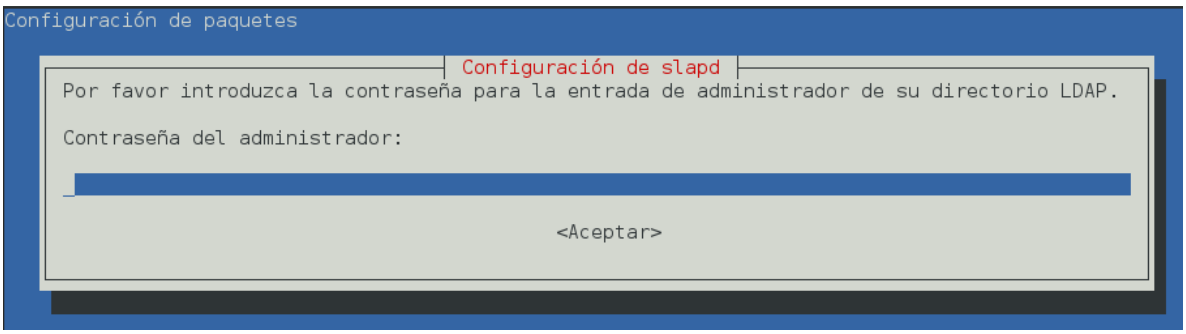
El dominio que se desea utilizar es el que tendrá que ser el mismo configurado en el Radius.



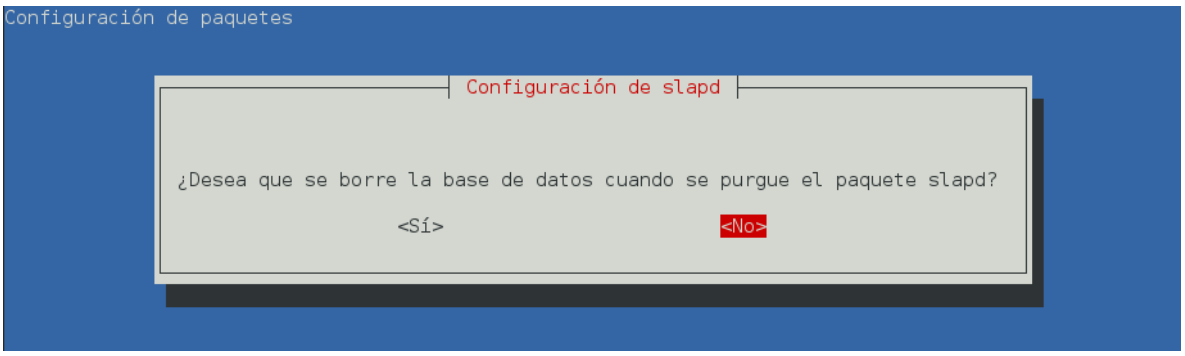
El ejemplo muestra el dominio Institución.edu.mx



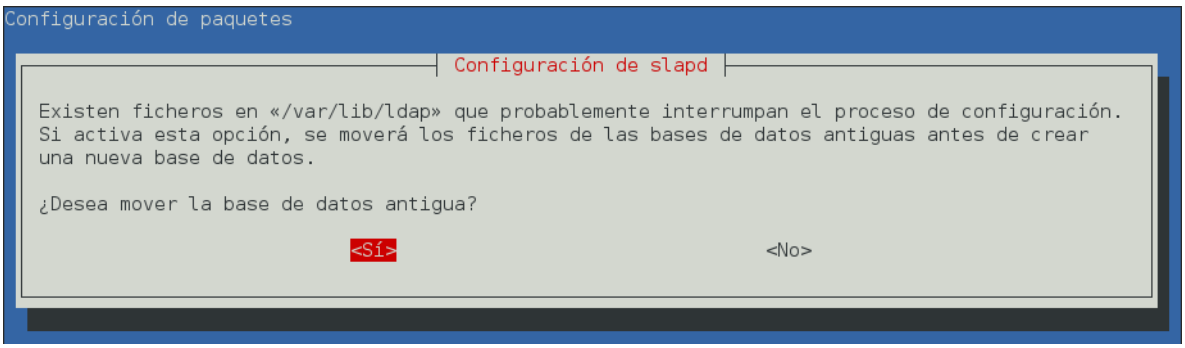
Nombre de la institución.

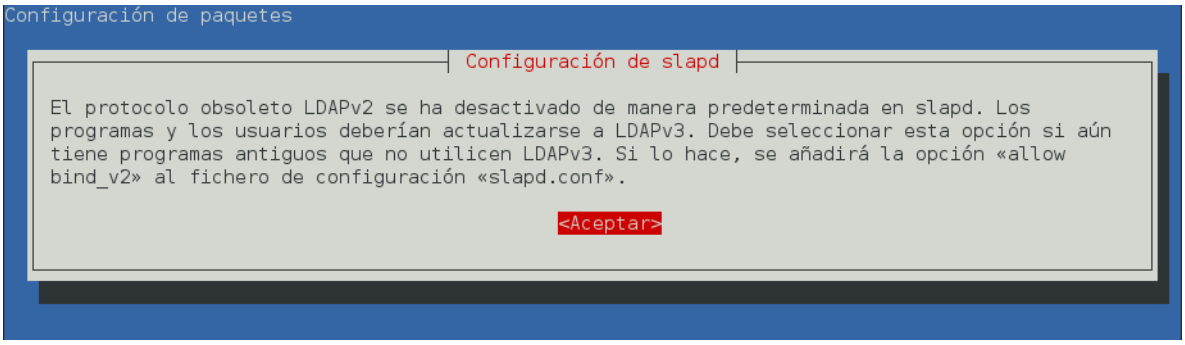


La contraseña deberá contener caracteres que aseguren un nivel alto de seguridad.

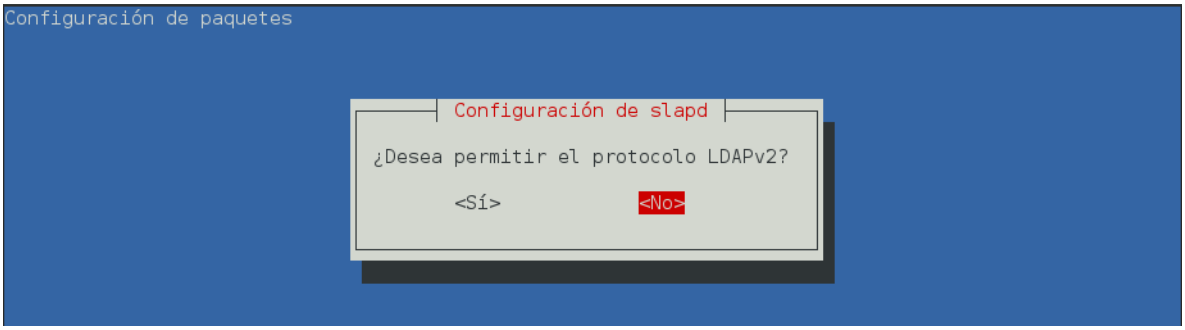


Se mantiene la base de datos, si ya se tiene una base de datos que se está migrando, se debe mantener.





El protocolo LDAPv2 presenta niveles de seguridad muy bajos por los que ya no se utiliza.



Seleccionamos NO y termina la configuración.

Recuerde guardar los cambios.

3. Configuración de usuarios

Para tener un orden en la creación de usuarios de acceso a eduroam utilizamos un grupo en LDAP llamado **"usuarios"**, para estos usuarios se crea un perfil en LDAP en modo texto.

Desde consola ingresamos al directorio /etc/ldap y creamos el directorio **ldif** el cuál contendrá la información de los usuarios a trasladar a LDAP acerca del usuario.

```
# mkdir /etc/ldap/ldif
```

```
# vim /etc/ldap/ldif/eduroam.ldif
```

Dentro creamos el archivo eduroam.ldif que da el perfil de la institución.

```
dn: ou=usuarios,dc=<Institución>,dc=edu,dc=mx
objectClass: top
objectClass: organizationalUnit
ou: usuarios
```

```
# vim /etc/ldap/ldif/user.ldif
```

Dentro creamos el archivo user.ldif que da el perfil del usuario de prueba.

```
dn: cn=test test,ou=usuarios, dc=<Institución>,dc=edu,dc=mx
givenName: test
sn: test
cn: test test
uid: test
mail: test@<Institucion>.edu.mx
userPassword: <clave en md5 usando slappaswd>
objectClass: inetOrgPerson
objectClass: top
```

Agregamos la estructura del directorio recién creada así como el usuario denominado "test" al directorio LDAP.

```
# ldapadd -x -w secret -D "cn=admin,dc=<nombre_institución>,dc=edu,dc=mx" -f eduroam.ldif
# ldapadd -x -w secret -D "cn=admin,dc=<nombre_institución>,dc=edu,dc=mx" -f user.ldif
```

4. Configuración del cliente LDAP para RADIUS.

Radius necesita reconocer cual es la base de datos que se requiere para conectar a los usuarios, se tiene que configurar el módulo LDAP.

```
# vim /etc/freeradius/modules/ldap
```

Se buscan las siguientes líneas para realizar la edición:

```
        …
ldap {
        ...
        server = 127.0.0.1
        basedn = "ou=usuarios,dc=<nombre_institución>,dc=edu,dc=<mx>"
        filter = "(uid=%{%{Stripped-User-Name}:-%{User-Name}})"
        base_fi = "(objectclass=radiusprofile)"
        ...
}
```

Se editara el servidor virtual *Default* de Radius

```
# vim /etc/freeradius/sites-enabled/default
```

Hay que descomentar (#) las líneas siguientes de LDAP

```
authorize {
...
        ldap
...
}
authenticate {
...
        Auth-Type LDAP {
        ldap
        }
...
}
```

El mismo procedimiento en inner-tunnel

```
authorize {
...
        ldap
...
}
authenticate {
...
        Auth-Type LDAP {
        ldap
        }
...
}
```

Al realizar estas configuraciones queda conectada la base de datos con LDAP, a continuación realizamos configuraciones para base de datos con SQL.

5. Configuración de un cliente MySQL para el servidor RADIUS.

Se requiere editar el archivo sql.conf

```
# vim /etc/freeradius/sql.conf
```

Editamos las siguientes líneas.

```
sql {
...
        database = "mysql"
        driver = "rlm_sql_${database}"
        server = 127.0.0.1
        port = 3306
        login = "eduroam"
        password = "eduroam"
        radius_db = "freeradius"
...
}
```

NOTA:
"eduroam": Como ejemplo usaremos: usuario= eduroam , y clave=eduroam
"freeradius": Base de datos del freeradius, donde se almacenará los eventos de conexión de usuarios.

Asignar los permisos correspondientes:

```
Pruebas de MySQL a nivel cliente

# mysql -u root -p
Enter password:
mysql> create user eduroam identified by 'eduroam';
Query OK, 0 rows affected (0.00 sec)
mysql> create database freeradius;
Query OK, 1 row affected (0.02 sec)
mysql> grant all privileges on freeradius.* to 'eduroam'@'127.0.0.1' identified by
'eduroam' with grant option;
Query OK, 0 rows affected (0.00 sec)
```

Este paso es importante, pues nos permite comprobar la validación del servidor Radius hacia una base de datos MySQL.

```
# mysql -h 127.0.0.1 -u eduroam -p freeradius
Enter password:
mysql> exit
```

Importar el esquema por defecto “schema.sql” hacia el servidor base de datos:

```
# cd /etc/freeradius/sql/mysql
# /etc/freeradius/sql/mysql# mysql -h 127.0.0.1 -u eduroam -p freeradius < schema.sql
Enter password:
root@test-eduroam:/etc/freeradius/sql/mysql#
```

Cree un usuario de pruebas en la base de datos freeradius

```
# mysql -h 127.0.0.1 -u eduroam -p freeradius
mysql> use freeradius;
mysql> insert into radcheck values (1,'user1','User-Password','==','pass1');
Query OK, 1 row affected (0.00 sec)
mysql> quit
```

NOTA:
1,'user1','User-Password','==','pass1': Usuario de prueba para la autenticación de usuarios, éstos se encontrarán en la tabla radcheck de la base de datos.

Cargamos el módulo de MySQL en el servidor Radius

```
# vim /etc/freeradius/radiusd.conf
```

```
#
...
        $INCLUDE sql.conf
...
#
```

Editamos los servidores virtuales del Radius

```
vim /etc/freeradius/sites-enabled/default
```

Las siguientes líneas serán habilitadas

```
# See “Authorization Queries” in sql.conf
        sql
#
...
# See “Authentication Logging Queries” in sql.conf
        sql
```

```
vim /etc/freeradius/sites-enabled/inner-tunnel
```

```
# See “Authorization Queries” in sql.conf
        sql
#
...
# See “Authentication Logging Queries” in sql.conf
        sql
```

Para finalizar, es necesario hacer unos cambios en el archivo “diaulup.conf”.

```
vim /etc/freeradius/sql/mysql/dialup.conf
```

```
Descomentar la línea
sql_user_name = "%{%{Stripped-User-Name}:-%{%{User-Name}:-DEFAULT}}"
Comentar la línea
#sql_user_name = "%{User-Name}"
```

Se reinician los demonios.

```
# service freeradius restart
# service mysql restart
```

6. Configuración de logs de monitoreo para eduroam.

Editamos el archivo radiusd.conf

```
# vim /etc/freeradius/radiusd.conf
```

```
...
log {
        destination = syslog
        file = ${logdir}/radius.log
        syslog_facility = local1
        stripped_names = yes
        auth = yes
        auth_badpass = yes
        auth_goodpass = yes
        msg_goodpass = "Usuario Aceptado %{User-Name}"
        msg_badpass = "Usuario Rechazado"
}
...
```

Nota:
destination = files: Usaremos como destino el archivo radius.log por defecto para los logs de autenticaciones.

syslog_facility = local1: Tomaremos como referencia la facility “local1” según
http://wiki.freeradius.org/guide/Syslog-HOWTO

Además editamos el archivo rsyslog.conf

```
# vim /etc/rsyslog.conf
```

Adicionar las siguientes líneas.

```
…
$ModLoad imudp
$UDPServerRun 514
…
        local1.=notice   /root/radius-notice.log
        local1.=err      /root/radius-err.log
        local1.=info     /root/radius-fticks.log
…
```

Pruebas de Logs.

Radius cuenta con una herramienta que permite comprobar la conectividad y validación de usuarios como test de autenticación tanto a servidores como a “localhost”

```
# radtest user@<institución>.edu.mx pass 127.0.1.1 0 <clave-localhost>
```

Para mantener visualizando los últimos ingresos del logs.

```
# tail –f /root/radius-notice.log
```

El resultado será como el ejemplo que a continuación se muestra.

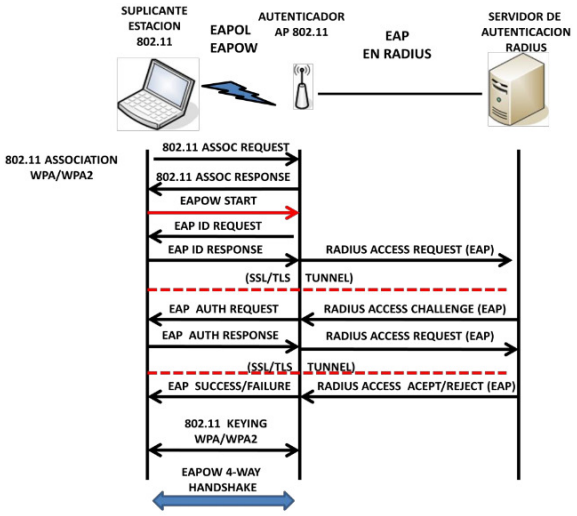
```
May  2 14:30:55 eduroam-local freeradius[20012]: Login OK: [usuario@<institución>.edu.mx/<no User-Password attribute>] (from client <AP-Local> port 0 cli <MAC address>) Usuario Aceptado usuario@<institución>.edu.mx
```

El archive radius-notice.log guardará toda la información de las conexiones que son realizadas al servidor o pasan a través de él.

Glosario

Proveedor de Identidad (IdP)	Organización del usuario – Suministra la información relacionada con las identidades – Suministra el servidor RADIUS que autentica al usuario – Responsable de autenticar al usuario
Proveedor de Servicio (SP)	Organización visitada – Suministra la infraestructura de red (puntos de acceso, VLANs, acceso a Internet, entre otros) – Responsable de la autorización del usuario
Suplicante	Es un software (a veces es parte del sistema operativo o como un programa separado) que usa el protocolo IEEE 802.1X para enviar la información de solicitud de autenticación usando EAP. Los suplicantes son instalados y operan en dispositivos de cómputo de usuarios finales (Notebooks, PDA, teléfonos celulares con Wi-Fi habilitado, entre otros).
Access Point (autenticador)	Son dispositivos de acceso LAN inalámbrico conforme al estándar IEEE 802.11 y necesitan tener la capacidad IEEE 802.1X. Deben tener la capacidad de reenviar las solicitudes de acceso desde un suplicante al servidor RADIUS del Proveedor de Servicio (red visitada), para dar acceso a red luego de una correcta autenticación, permitiendo la asignación de usuarios a una VLAN específica basada en la información recibida desde el servidor RADIUS. Además los access point intercambian material clave (vectores de inicialización, claves públicas y sesiones, etc.) con sistemas de clientes para impedir sesiones hijacking.
Switches	Necesitan ser capaces de reenviar las solicitudes de acceso que viene de un suplicante al servidor RADIUS del Proveedor de Servicio, para permitir el acceso a red tras una apropiada autenticación y posiblemente asignar usuarios a VLANs específicas basadas en la información recibida del servidor RADIUS.
Protegido Protocolo de autenticación extensible (PAEP)	EAP protegido, o simplemente PAEP, es un método para transmitir de forma segura la autenticación de la información, incluyendo contraseñas, más de las redes LAN inalámbricas. Ha sido desarrollado conjuntamente por Microsoft, RSA Security y Cisco. Es un estándar abierto IETF. PEAP no es un protocolo de cifrado; al igual que con otros tipos de EAP sólo autentica un cliente en una red. PEAP utiliza certificados de clave pública solamente del lado del servidor para autenticar a los clientes mediante la creación de un túnel encriptado SSL / TLS entre el cliente y el servidor de autenticación, que protege el subsiguiente intercambio de información de autenticación de la inspección casual. PEAP es similar en diseño a EAP-TTLS, que sólo requiere un certificado PKI del lado del servidor para crear un túnel seguro TLS para proteger la autenticación del usuario. A partir de mayo de 2005, había dos subtipos del PAEP certificados para el estándar WPA y WPA2 actualizada. Ellos son: • PEAPv0 / EAP-MSCHAPv2 • PEAPv1 / EAP-GTC Tipos PEAPv0 / EAP-MSCHAPv2 es la forma más común de PEAP en uso, y lo que se conoce generalmente como PEAP.

	Detrás de EAP-TLS, PEAPv0 / EAP-MSCHAPv2 es la segunda norma EAP más amplio apoyo en el mundo. Hay cliente y servidor implementaciones de la misma de diferentes proveedores, incluido el apoyo en todas las versiones más recientes de Microsoft, Apple y Cisco. PEAPv1 / EAP-GTC fue creado por Cisco como una alternativa a PEAPv0 / EAP-MSCHAPv2. Se permite el uso de un protocolo de autenticación interna que no sea de Microsoft MSCHAPv2. A pesar de que Microsoft co-inventó el estándar PEAP, Microsoft nunca se ha añadido soporte para PEAPv1 en general, lo que significa PEAPv1 / EAP-GTC no tiene soporte nativo de Windows OS. PEAP-EAP-TLS Microsoft admite otra forma de PEAPv0 (que Microsoft llama PEAP-EAP-TLS) que Cisco y otros software de servidor de terceros y el cliente no apoyan. PEAP-EAP-TLS requiere un certificado digital en el cliente se encuentra en el disco duro del cliente o de una tarjeta inteligente más seguro. PEAP-EAP-TLS es muy similar en funcionamiento al original EAP-TLS, pero ofrece un poco más de protección, debido al hecho de que las partes del certificado de cliente que están sin encriptar en EAP-TLS se cifran en PEAP-EAP-TLS. Desde algunos clientes de terceros y servidores soportan PEAP-EAP-TLS, los usuarios probablemente deberían evitar a menos que sólo tienen la intención de utilizar los clientes y servidores de escritorio de Microsoft.
Estándar IEEE 802.1X	La IEEE 802.1X es una norma del IEEE para el control de acceso a red basada en puertos. Es parte del grupo de protocolos IEEE 802 (IEEE 802.1). Permite la autenticación de dispositivos conectados a un puerto LAN, estableciendo una conexión punto a punto o previniendo el acceso por ese puerto si la autenticación falla. Es utilizado en algunos puntos de acceso inalámbricos cerrados y se basa en el protocolo de autenticación extensible (EAP). 802.1X está disponible en ciertos conmutadores de red y puede configurarse para autenticar nodos que están equipados con software suplicante. Esto elimina el acceso no autorizado a la red al nivel de la capa de enlace de datos. Algunos proveedores están implementando 802.1X en puntos de acceso inalámbricos que pueden utilizarse para operarse como un punto de acceso cerrado, corrigiendo deficiencias de seguridad de WEP. Esta autenticación es realizada normalmente por un tercero, tal como un servidor de RADIUS. Esto permite la autenticación sólo del cliente o, más apropiadamente, una autenticación mutua fuerte utilizando protocolos como EAP-TLS.

	 Secuencia de operación de IEEE802.1x - Autenticación EAP
Temporal Key Integrity Protocol - TKIP Advanced Encryption Standard (AES)	Es un conjunto de algoritmos de seguridad que funcionan como un “envoltorio” para WEP. Fue diseñado para obtener la mayor seguridad posible en dispositivos WLAN antiguos equipados con WEP sin necesidad de actualizar el hardware. El problema del WEP original es que un atacante podría obtener tu clave “esnifiando” una cantidad relativamente pequeña del tráfico. TKIP resuelve dicho problema re-negociando una clave nueva cada pocos minutos (el atacante nunca tendría suficiente información para descubrirla). En la actualidad TKIP no es fiable ni eficiente para proteger un entorno WLAN. Por eso el estándar 802.11i especifica el protocolo AES como mecanismo adicional de seguridad. AES ofrece un mayor nivel de seguridad, pero requiere un hardware específico que no es compatible con los dispositivos que solo funcionan con WEP y con WPA. Utiliza bloques de cifrado de 128, 192 0 256 bits y es considerado el sistema de cifrado estrella. Es cierto que AES necesita más potencia de cálculo y eso repercute en el consumo de algunos dispositivos móviles. Pero AES no es sólo más seguro, sino también más eficiente ya que necesita meno ancho de banda. Sin duda es la mejor opción para los sistemas WLAN actuales. WEP y WPA utilizan TKIP. WPA2 es infinitamente más seguro y utiliza AES, pero también puede usar TKIP por retro-compatibilidad (así WPA2 podría aceptar conexiones WPA). Cuando configuramos nuestro router con WPA2, podemos seleccionar entre sólo AES o TKIP+AES. En esta última modalidad los dispositivos de red puedan usar WPA2, y los dispositivos que sólo puedan usar WPA se conectarán mediante WPA. La clave precompartida para WPA para ambos sería la misma. La mejor opción de configuración será WPA2/AES, sin TKIP.
OpenSSL	OpenSSL es una librería que se halla en todos los servidores linux con código abierto como es el caso de los servidores de eduroam. Consiste en un robusto paquete de herramientas de administración y bibliotecas relacionadas con la criptografía, que suministran funciones criptográficas a otros paquetes como OpenSSH y navegadores web (para acceso seguro a sitios HTTPS).

	OpenSSL sirve para cifrar los datos que se transmiten desde el servidor o al servidor con el fin de que si una cadena de datos fuese interceptada por un hacker mientras se transmite dichos datos son ilegibles ya que el hacker debería desencriptar esa información para poder sacar algún dato útil.
dh	Algoritmo de claves Diffie-Hellman – dh Estas funciones implementan el protocolo de acuerdo de claves Diffie-Hellman. El algoritmo dh proporciona la capacidad para comunicar dos partes a ponerse de acuerdo sobre un secreto compartido entre ellos. Su esquema de un acuerdo porque ambas partes añaden material utilizado para derivar la clave.
dhparam	Manipulación y generación de parámetros dh.
dd (Data Duplicator)	dd es una aplicación para copiar datos desde dispositivo a otro similar, copiando información de varias formas, además también se puede utilizar para formatear unidades, borrar información sensible, reparar en forma lógica unidades tanto en Discos y Memorias, dd tiene la ventaja de poseer una administración avanzada basada solo en Texto, para crear Backups, Disco de Instalación, Formateo y otros etc. La Sintaxis básica de dd es así: dd if=origen of=destino
DER (Distinguished Encoding Rules) para Windows Reglas de codificación distinguida	Abstract Syntax Notation One (ASN.1) define los siguientes conjuntos de reglas que gobiernan cómo se codifican y decodifican las estructuras de datos que se envían entre equipos. Reglas de codificación básica (BER) Canonical Encoding Rules (CER) Reglas de codificación distinguida (DER) Reglas de codificación compactada (PER) El conjunto de reglas original fue definido por la especificación BER CER y DER, se desarrollaron más tarde como subconjuntos especializados de BER. PER fue desarrollado en respuesta a las críticas sobre la cantidad de ancho de banda necesario para transmitir datos utilizando BER o sus variantes. PER proporciona un importante ahorro. DER fue creado para satisfacer los requisitos de la especificación X.509 para la transferencia segura de datos. La API de inscripción de certificados utiliza exclusivamente DER. Para obtener más información.

Apéndice de problemas y soluciones

Durante la creación de los certificados puede pasar que existan errores como:

failed to update database TXT_DB error number 2	En el archivo /etc/eduroam-radsec index.txt.attr Cambiar el estado a: unique_subject = no

Referencias

Protocolos de autenticación.
http://wiki.freeradius.org/protocol/EAP-PEAP

CISCO Tecnología Inalámbrica
http://www.cisco.com/web/LA/soluciones/comercial/proteccion_wireless.html

Seguridad en WIFI
http://borrowbits.com/2014/02/seguridad-en-wi-fi-que-cifrado-es-mejor-tkip-o-aes/

OpenSSL
https://www.openssl.org/docs/manmaster/crypto/dh.html