



## Proyecto WPA2-Enterprise, Radius, LDAP

I.E.S Gonzalo Nazareno

Fecha:

21/06/2013

Leonardo Bernal Bueno – leobernal91@gmail.com

Puedes copiar y modificar todos los contenidos,  
pero siempre respetando los términos de la licencia CC-BY-SA.



## Índice de contenido

1. Glosario.....	4
2. Introducción.....	4
3. Objetivos.....	5
Infraestructura.....	5
4. Métodos de Autenticación.....	6
PEAP.....	6
EAP-TTLS.....	6
EAP-TLS.....	7
LEAP.....	7
EAP-FAST.....	7
Tabla Comparativa.....	7
Método de autenticación elegido.....	8
5. Protocolos de autenticación.....	8
PAP.....	9
CHAP.....	9
MSCHAP / MSCHAPv2.....	9
Protocolo de autenticación elegido.....	10
5. Software usado.....	10
6. Hardware usado.....	11
7. Requisitos del sistema.....	11
8. Instalación de FreeRadius.....	12
Certificados.....	12
9. Configuración de freeradius.....	13
radiusd.conf.....	13
sites-available.....	15
Servidor virtual: iesgn.....	15
eap.conf.....	17
ldap.....	19
users.....	21
clients.conf.....	22
10. Arranque de freeradius.....	22

11. Freeradius Alta Disponibilidad.....	22
12. Configuración de puntos de acceso.....	24
13. Configuración de clientes.....	25
Linux.....	26
Windows.....	28
Android.....	36
Iphone.....	38
14. Comunicación cifrada.....	42
15. Vulnerabilidades.....	44
1. Protocolos de autenticación.....	44
2. Que se comprometa directamente la máquina.....	45
3. Rogue AP (con Backtrack o Kali Linux).....	45

## 1. Glosario

A continuación se citan algunas referencias de introducción que son consideradas básicas para entender el ámbito y alcance de este documento.

- **Confidencialidad:** sólo accederá a la información quien se encuentre autorizado.
- **Integridad:** la información se mantendrá íntegra, exacta y completa.
- **Disponibilidad:** los usuarios autorizados tendrán acceso a la información cuando lo requieran.
- **No repudio:** se hace referencia a la capacidad de afirmar la autoría de una mensaje o información, evitando que el autor niegue la existencia de su recepción o creación.

## 2. Introducción

Una de las opciones más seguras que permite controlar la autenticación de usuarios se puede realizar mediante la configuración de un servidor *Radius*. Radius o Remote Authentication Dial-In User Server, es un protocolo de autenticación para aplicaciones de acceso a la red o movilidad IP. **Podemos decir que más que un protocolo de autenticación, es un protocolo AAA (Authentication, Authorization, Administration).**

Actualmente la red inalámbrica de las aulas de informática del *IES Gonzalo Nazareno* usan como método de autenticación *WPA/WPA2-PSK* con una clave compartida.

Esta configuración no es la más adecuada, teniendo en cuenta que es relativamente fácil que un usuario ilícito consiga dichas credenciales usando diversos métodos, vulnerando así nuestra red.

En este documento se detallará de forma explícita la configuración e implantación de *WPA2-Enterprise*, *FreeRadius*, *OpenLDAP* usando como método de autenticación *EAP-TTLS/PAP* sobre *Debian Wheezy*.

De esta forma aprovecharemos nuestro servidor de usuarios centralizados para conectarnos a la red, aplicando así otro nivel de seguridad.

### 3. Objetivos

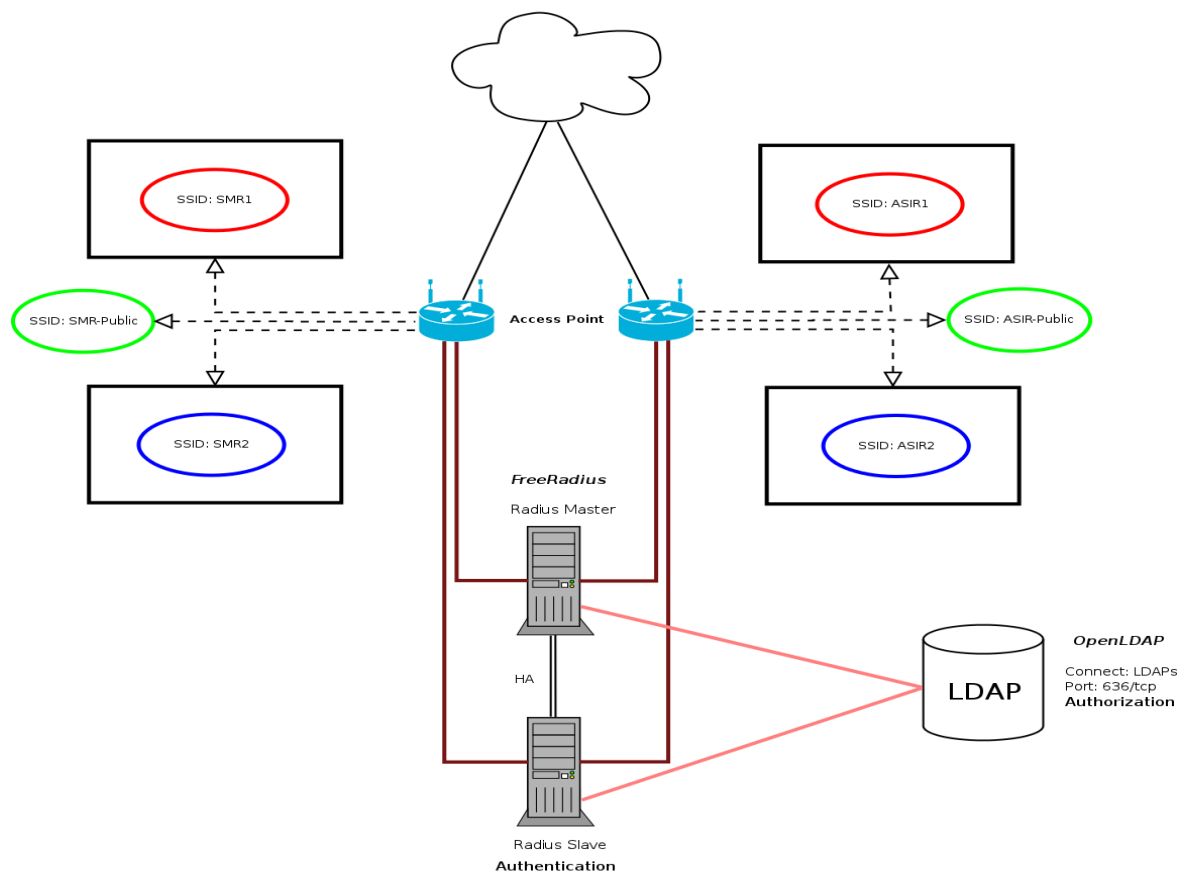
Montar una infraestructura de autenticación de redes inalámbricas basada en servidores Radius (autenticación) y servidores Ldap (autorización). Esto quiere decir que nuestros usuarios se conectarán a la red wifi usando su nombre de usuario y contraseña con el que están dados de alta en el servidor Ldap del centro.

#### Objetivos a cumplir:

- Estudio de diferentes métodos de autenticación y protocolos.
- Conexión y configuración Radius – Ldap
- Toda la comunicación cifrada.
- Configuración de puntos de acceso.
- Radius en alta disponibilidad.
- Documentación para los clientes. [1]
- Vulnerabilidades.

#### Infraestructura

A continuación, la infraestructura propuesta para la red inalámbrica del centro.



[1] Se redactará un documento con la configuración desde clientes Windows, Linux, Android y Iphone que estará disponible para todos los alumnos del centro.

## 4. Métodos de Autenticación

Existen diferentes tipos de autenticación en un servidor Radius, algunos son utilizados con mayor frecuencia principalmente por su soporte en plataformas *Windows*. En concreto los mecanismos de autenticación más conocidos y con los que normalmente se suele trabajar cuando se configura un servidor Radius son:

- PEAP
- EAP-TTLS
- EAP-TLS
- LEAP
- EAP-FAST

De los citados anteriormente quizás el más utilizado es PEAP, probablemente porque se encuentra soportado de forma nativa en sistemas *Windows*, aunque no es el más seguro.

### **PEAP**

Protected Extensible Authentication Protocol es un mecanismo de autenticación que se ha convertido en uno de los más extendidos, probablemente a causa de que se encuentra soportado de forma nativa en plataformas *Windows*, especialmente la versión *PEAP 0* la cual funciona con *EAP-MSCHAP-V2*, la versión 1 no se encuentra soportada de forma nativa en máquinas *Windows* por lo que no se ha vuelto tan popular, sin embargo es un mecanismo de autenticación robusto basado en la generación dinámica de tokens para los usuario autenticados.

Ahora bien, para que este mecanismo sea “seguro” es obligatorio que el servidor tenga instalados certificados que puedan ser validados por el cliente.

### **EAP-TTLS**

Se trata de un mecanismo muy robusto que permite que las comunicaciones se realicen en un túnel cifrado de comunicaciones y obliga a que el servidor se autentique con un certificado y opcionalmente permite el uso de certificados en el lado del cliente.

Se trata de una solución que no se encuentra tan extendida como *PEAP con EAP-MSCHAP-V2* debido a que no cuenta con soporte nativo para plataformas *Windows*.

## EAP-TLS

Es un mecanismo de autenticación muy similar al *EAP-TTLS* sin embargo tiene una diferencia que le hace ser, probablemente el mecanismo de seguridad más fuerte de todos los *EAP* que se encuentran disponibles. Esta diferencia es que *EAP-TLS* tiene como requerimiento **OBLIGATORIO** que tanto el cliente como el servidor utilicen certificados para realizar el proceso de establecimiento del túnel cifrado y posterior autenticación. Se trata del mecanismo más robusto que actualmente puede establecerse sobre un servidor Radius, no obstante tiene la dificultad de que en entornos con un número de clientes medio, la distribución de los certificados personales puede ser un proceso bastante delicado.

## LEAP

*LEAP* (Lightweight EAP) propiedad de Cisco. En este caso se utilizan también las contraseñas como método de autenticación en el servidor. Cuando se emplea *LEAP* las credenciales de usuario (nombre de usuario y contraseña) se envían sin cifrar, además requiere de una infraestructura Cisco para poder ser utilizado.

Por lo tanto, esta autenticación aunque ligera, previene de ataques *Man In The Middle* y de secuestro de sesión, pero sigue manteniendo riesgos de exposición de la identidad y de ataque de diccionarios.

## EAP-FAST

*EAP-FAST* es una propuesta de protocolo de Cisco System como reemplazo de *LEAP*. El protocolo fue diseñado para hacer frente a las debilidades de *LEAP* preservando al mismo tiempo la aplicación “ligera”. El uso de certificados es opcional en la parte del servidor y no soportado en el cliente. *EAP-FAST* utiliza unas credenciales de acceso protegidas (*PAC*) para establecer un túnel autenticado entre cliente-servidor.

### Tabla Comparativa

	Certificados en Servidor	Certificados en Clientes	Soporta LDAP	Compatibilidad Windows
PEAP	Obligatorio	Opcional	SI	SI
EAP-TTLS	Obligatorio	Opcional	SI	-
EAP-TLS	Obligatorio	Obligatorio	SI	-
LEAP	-	-	Sólo con MS-CHAP *	-
EAP-FAST	Opcional	-	SI	Sólo en Intel

\* *LEAP* sólo soporta conexiones con LDAP si sus clientes se autentican usando el protocolo *MS-CHAP*. Además como dato importante, las contraseñas deberán ser almacenadas en LDAP en texto plano debido a que las credenciales de usuario se envían sin cifrar.

### ***Método de autenticación elegido.***

Teniendo ahora una idea de algunos de los métodos de autenticación con los que trabaja *FreeRadius*, el método elegido es **EAP-TTLS** por los siguientes motivos:

- **EAP-TTLS** es un método de autenticación tunelado que implementa un sistema de dos túneles de seguridad: Uno se crea para el intercambio de credenciales y otro para el traspaso de la clave de cifrado de sesión con la que los puntos de acceso cifran el tráfico con el cliente que se conecta.
- Todo el tráfico circula totalmente cifrado, de manera que nos proporciona un sistema seguro de acceso a la red.
- La autenticación se realiza solo con certificados de servidor y no es necesario generar certificados para cada cliente nuevo que desee conectarse a la red. Esto facilita el acceso de nuevas máquinas a la red. De tal forma éste método está preparado para futuras necesidades, si se decide la implantación de certificados en el lado del cliente.
- A diferencia de otros tipos de EAP, tiene la capacidad de soportar una amplia variedad de métodos de autenticación interna, PEAP sólo puede autenticar a los clientes que utilizan el protocolo de autenticación por desafío mutuo de Microsoft conocida como MS-CHAPv2.

Por otro lado, **excepto Windows 8, las demás versiones de Microsoft no dan soporte nativo a EAP-TTLS/PAP**, para ello, la solución que se plantea (igual que ocurre en la red *Eduroam*) es la instalación del cliente **SecureW2**. Con una sencilla configuración, todos los equipos Windows podrán conectarse sin problema alguno.

**EAP-TTLS** no es vulnerable actualmente a ataques MITM ni de diccionarios.

## **5. Protocolos de autenticación**

Independientemente de los métodos de autenticación que hemos visto en el apartado anterior, para la comunicación interna de los servicios podemos usar diferentes protocolos de autenticación.

A continuación vamos a estudiar los protocolos soportados por EAP-TTLS:

- PAP
- CHAP
- MSCHAP
- MSCHAPv2



**PAP**

Password Authentication Protocol o PAP es un protocolo simple de autenticación para autenticar un usuario contra un servidor de acceso remoto o contra un proveedor de servicios de internet. PAP es un subprotocolo usado por la autenticación del protocolo PPP (Point to Point Protocol), validando a un usuario que accede a ciertos recursos. PAP transmite contraseñas o passwords en texto plano.

**CHAP**

CHAP o protocolo de autenticación por desafío mutuo, utiliza un algoritmo (MD5) para calcular un valor que sólo conocen el sistema de autenticación y el dispositivo remoto, conocido como secreto compartido. Con CHAP, el nombre de usuario y la contraseña siempre se envían cifrados.

Aunque la comunicación esté cifrada, actualmente el algoritmo MD5 está roto.

**MSCHAP / MSCHAPv2**

MS-CHAP, (Microsoft Challenge Handshake Authentication Protocol.).

Es un protocolo de autenticación de contraseñas de cifrado por desafío mutuo, de Microsoft, el cual es irreversible.

Básicamente la diferencia entre ambos protocolos es que la versión 2 de MSCHAP, Microsoft realizó mejoras importantes de seguridad. Proporciona autenticación mutua, claves de cifrado de datos iniciales más fuertes.

Actualmente, ambos protocolos son vulnerables.

	Clear-text	NTLM (ntlm_auth)	MD5 hash	Salted MD5 hash	SHA1 hash	Salted SHA1 hash
PAP	✓	✓	✓	✓	✓	✓
CHAP	✓	x	x	x	x	x
MSCHAP	✓	✓	x	x	x	x
MSCHAPv2	✓	✓	x	x	x	x

**NOTA:** Realizando algunas pruebas, usando como protocolo de autenticación MSCHAP, he observado cómo efectivamente no es compatible con el algoritmo de cifrado que están guardadas las contraseñas de LDAP (SSHA).

Mirando los log de freeradius podemos ver cómo exige usar el algoritmo NTLM, confirmando así los datos obtenidos en la tabla anterior.

```
[mschap] FAILED: No NT/LM-Password. Cannot perform authentication.  
[mschap] MS-CHAP-Response is incorrect.  
Failed to authenticate the user.
```

### ***Protocolo de autenticación elegido***

En nuestra infraestructura no tenemos más remedio que usar **PAP** debido a la compatibilidad con los algoritmos de cifrado de las contraseñas de LDAP.

PAP no es recomendable usarlo independiente, pero no hay ningún problema al usarlo junto a EAP-TTLS ya que éste cifra toda la comunicación cliente-servidor haciendo uso de túneles.

Mediante el uso de certificados digitales EAP-TTLS es capaz de autenticar a los usuarios a través del protocolo PAP de una manera segura.

En los siguientes apartados comprobaremos que efectivamente toda la comunicación está cifrada analizando la red con Wireshark.

## **5. Software usado**

A continuación se detallan los principales nombres, especificaciones y versiones del software que se usará:

- Sistema operativo: Debian Squeeze, Linux Kernel versión 3.2.0-4-amd64.
- Dos servidores FreeRadius 2.1.12
- OpenLDAP 2.4.31-1.
- SecureW2 EAP suite 2.0.4 para clientes Windows 7. [2]

[2] Para conectarnos a la red inalámbrica del centro desde clientes Windows, necesitamos el programa SecureW2 para autenticarnos usando *EAP-TTLS* (sólo en Windows. GNU/Linux, android y iPhone los incorporan de forma nativa).

## 6. Hardware usado

Los puntos de acceso que utilicemos, como requisito mínimo deberán tener soporte para 802.1X para que puedan configurarse como clientes de un servidor Radius.

- Punto de acceso Cisco Linksys WAP-4410N.

Al ser tecnología POE, incorporamos alimentación eléctrica directamente por la conexión Ethernet lo que simplifica la instalación y elimina la necesidad de un cableado adicional.

## 7. Requisitos del sistema

Los puertos usados por RADIUS son 1812/UDP para autenticación y 1813/UDP para administración de cuentas.

Necesitamos abrir dichos puertos en el firewall perimetral del centro en sentido Entrada-Salida.

```
# Acceso desde aulas a Papion Freeradius
```

```
iptables -A FORWARD -s $AULAS -d $PAPION -p udp --dport 1812 -j ACCEPT
iptables -A FORWARD -s $PAPION -d $AULAS -p udp --sport 1812 -j ACCEPT
iptables -A FORWARD -s $AULAS -d $PAPION -p udp --dport 1813 -j ACCEPT
iptables -A FORWARD -s $PAPION -d $AULAS -p udp --sport 1813 -j ACCEPT
```

Confirmamos que efectivamente las reglas se cargan correctamente.

```
# iptables -L -n -v
```

```
Chain FORWARD
```

6	174	ACCEPT	udp	--	*	*	172.22.0.0/16	192.168.2.2	udp dpt:1812
0	0	ACCEPT	udp	--	*	*	192.168.2.2	172.22.0.0/16	udp spt:1812
5	145	ACCEPT	udp	--	*	*	172.22.0.0/16	192.168.2.2	udp dpt:1813
0	0	ACCEPT	udp	--	*	*	192.168.2.2	172.22.0.0/16	udp spt:1813

Usamos la herramienta Netcat para probar que el servicio está escuchando.

-u: Para que netcat trabaje con protocolo UDP en vez de TCP

-v(verbose): Muestra informacion sobre la conexion.

```
leo@zeus:~$ nc -vu papion.gonzalonazareno.org 1812
```

```
Connection to papion.gonzalonazareno.org 1812 port [udp/radius] succeeded!
```

## 8. Instalación de FreeRadius

Podemos instalar freeradius desde código fuente (<http://freeradius.org>) o desde repositorios siendo más sencillo y rápido.

Los paquetes necesarios son freeradius y freeradius-ldap para realizar conexiones con nuestro servidor LDAP.

```
# aptitude update
```

```
# aptitude install freeradius freeradius-ldap
```

### **Certificados**

El método de autenticación EAP-TTLS no nos obliga a crear una autoridad de certificación, aunque sí vamos a necesitar tener un certificado de servidor. Para disponer de dicho certificado tenemos varias opciones:

1. Usar el certificado que se crea automáticamente al instalar freeradius.
2. Crear un certificado autofirmado.
3. Crear nuestra propia autoridad de certificación, con la que generamos dicho certificado.

Para nuestra infraestructura he decidido utilizar los certificados de freeradius. A efectos finales todas las opciones son iguales.

Para usar los certificados en otras aplicaciones, la instalación de freeradius en Debian almacena los certificados en /etc/ssl y crea enlaces a los mismo en el directorio /etc/freeradius/certs.

Vamos a restringir los permisos en sólo lectura para el propietario y grupo.

```
# chown root:freerad /etc/freeradius/certs/dh
```

```
# chown root:freerad /etc/ssl/private/ssl-cert-snakeoil.key
```

```
# chown root:freerad /etc/ssl/certs/ssl-cert-snakeoil.pem
```

```
# chown root:freerad /etc/ssl/certs/ca-certificates.crt
```

```
# chmod 440 /etc/freeradius/certs/dh
```

```
# chmod 440 /etc/ssl/certs/ca-certificates.crt
```

```
# chmod 440 /etc/ssl/certs/ssl-cert-snakeoil.pem
```

```
# chmod 440 /etc/ssl/private/ssl-cert-snakeoil.key
```

```
# ls -l /etc/freeradius/certs/
lrwxrwxrwx 1 root freerad 34 jun 12 16:32 ca.pem -> /etc/ssl/certs/ca-certificates.crt
-r--r----- 1 root freerad 245 jun 12 16:32 dh
lrwxrwxrwx 1 root freerad 38 jun 12 16:32 server.key -> /etc/ssl/private/ssl-cert-snakeoil.key
lrwxrwxrwx 1 root freerad 36 jun 12 16:32 server.pem -> /etc/ssl/certs/ssl-cert-snakeoil.pem
```

Podemos observar los permisos de los enlaces por defecto. Quien realmente restringe el acceso son los ficheros originales.

```
# ls -l /etc/ssl/certs/ca-certificates.crt
-r--r----- 1 root freerad 245341 mar 19 10:50 /etc/ssl/certs/ca-certificates.crt
```

## 9. Configuración de freeradius

Actualmente en versiones 2.x la configuración se almacena en `/etc/freeradius`. Una cuestión a comentar es que `freeradius` en sus últimas versiones es capaz de leer algunos ficheros de configuración en tiempo real, como por ejemplo el fichero `clients.conf`. En cualquier caso para que `freeradius` vuelva a leer los ficheros de configuración cuando hayamos hecho un cambio, no tenemos más que hacer un:

```
# /etc/init.d/freeradius reload
```

### ***radiusd.conf***

Este es el fichero principal de la configuración de radius. Los valores a tener en cuenta son los siguientes. El resto de parámetros se mantendrán por defecto.

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

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

Con estos últimos dos parámetros, estamos diciendo que no se almacenen en los log las peticiones de autenticación. Puede ser útil activarlos mientras estamos haciendo pruebas.

Desactivamos la configuración de Proxy

```
# PROXY CONFIGURATION
#
# proxy_requests: Turns proxying of RADIUS requests on or off.
#
# The server has proxying turned on by default. If your system is NOT
# set up to proxy requests to another server, then you can turn proxying
# off here. This will save a small amount of resources on the server.
#
# If you have proxying turned off, and your configuration files say
# to proxy a request, then an error message will be logged.
#
# To disable proxying, change the "yes" to "no", and comment the
# $INCLUDE line.
#
# allowed values: {no, yes}
#
proxy_requests = no
#$INCLUDE proxy.conf
```

Además comentamos los INCLUDE relacionados con otras bases de datos, ya que no vamos a almacenar nada en ellos.

```
# $INCLUDE sql.conf
# $INCLUDE sql/mysql/counter.conf
# $INCLUDE sqlippool.conf
```

### ***sites-available***

A partir de la versión 2 de *freeradius* es posible crear diferentes servidores virtuales, de una forma muy similar a como se crean apache.

En el directorio *sites-available* se encuentran los archivos de configuración de cada uno de los servidores virtuales que vayamos a crear. Y por supuesto, cada uno de estos servidores podría tener una configuración diferente.

En este directorio existe un fichero que define un servidor preconfigurado: el fichero *default*.

Creamos un enlace simbólico en */etc/freeradius/sites-enabled*.

Nuestro servidor virtual se llamará **iesgn**.

```
radius:/etc/freeradius/sites-available# cp default iesgn
```

```
radius:/etc/freeradius/sites-enabled# ln -s ../sites-available/iesgn
```

### ***Servidor virtual: iesgn***

Una vez creado el servidor virtual, como hemos dicho, pasamos a modificar su fichero de configuración para adaptarlo a nuestras necesidades. Nuestro servidor virtual se llama *iesgn*.

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

```
        checkval
        expiration
        logintime
    }

authenticate {
    Auth-Type LDAP {
        ldap
    }
    # Allow EAP authentication.
    eap
}

preacct {
    preprocess
    acct_unique
    suffix
    files
}

accounting {
    detail
    unix
    radutmp
    attr_filter.accounting_response
}

session {
    radutmp
}

post-auth {
    exec
    Post-Auth-Type REJECT {
        # log failed authentications in SQL, too.
        attr_filter.access_reject
    }
}
```



```
pre-proxy {  
}  
post-proxy {  
}
```

### ***eap.conf***

Este es uno de los ficheros más importantes, donde vamos a definir el tipo de autenticación EAP-TTLS. Indicaremos los certificados de servidor que se usarán para establecer la comunicación radius.

El directorio confdir es /etc/freeradius

```
eap {  
    default_eap_type = tls  
    timer_expire     = 60  
    ignore_unknown_eap_types = no  
    cisco_accounting_username_bug = no  
    max_sessions = 4096  
#    md5 {  
#    }  
#    leap {  
#    }  
#    gtc {  
#        auth_type = PA P  
#    }  
    tls {  
        certdir = ${confdir}/certs  
        cadir = ${confdir}/certs  
        private_key_password = whatever  
        private_key_file = ${certdir}/server.key  
        certificate_file = ${certdir}/server.pem  
        CA_file = ${cadir}/ca.pem  
        dh_file = ${certdir}/dh  
        random_file = /dev/urandom
```

```
#      CA_path = ${cadir}
#      cipher_list = "DEFAULT"
#      make_cert_command = "${certdir}/bootstrap"
#      ecdh_curve = "prime256v1"
#      cache {
#          enable = no
#          lifetime = 24 # hours
#          max_entries = 255
#      }
#      verify {
#      }
#      ocsp {
#          enable = no
#          override_cert_url = yes
#          url = "http://127.0.0.1/ocsp/"
#      }
#  }
#  ttls {
#      default_eap_type = tls
#      copy_request_to_tunnel = no
#      use_tunneled_reply = no
#      include_length = yes
#  }
#  peap {
#      default_eap_type = mschapv2
#      copy_request_to_tunnel = no
#      use_tunneled_reply = no
#  }
#  mschapv2 {
#  }
}
```

## ldap

El fichero ldap nos permite configurar el acceso de freeradius con el servidor ldap del centro. Se encuentra dentro del directorio `/etc/freeradius/modules/`

Tenemos que configurar nuestro servidor Radius para que se conecte a ldap usando conexiones seguras cifrando los datos con SSL por el puerto 636 tcp.

Daremos de alta un usuario en openLdap con permisos de sólo lectura para realizar las consultas apropiadas. Nombre de usuario `"radius"` contraseña `"root$root"`.

Vamos a utilizar conexiones SSL seguras a ldap, pero no vamos a hacer uso de tls. En el apartado tls de este fichero tendremos que indicar: `start_tls = no`.

Por otra parte, para usar ldaps tendremos que copiar la clave pública del servidor ldap a un lugar seguro de la máquina freeradius. El certificado se encontrará en `/etc/ldap/ssl/` con permisos de sólo lectura para su propietario, es decir, para el usuario root.

```
# scp root@192.168.1.200:/etc/ldap/ssl/publica.pem /etc/ssl/.
```

```
# chown root:freerad /etc/ssl/publica.pem
```

```
# chmod 440 /etc/ssl/publica.pem
```

```
ldap {
    port = "636"
    server = "ldap.example.com"
    identity = "cn=radius,dc=ldap,dc=example,dc=com"
    password = "root$root"
    basedn = "dc=ldap,dc=example,dc=com"
    filter = "(uid=%{%{Stripped-User-Name}:-%{User-Name}})"
#    base_filter = "(objectclass=radiusprofile)"

    ldap_connections_number = 5
    timeout = 4
    timelimit = 3
    net_timeout = 1
    tls {
        start_tls = no
        cacertfile    = /etc/ldap/ssl/publica.pem
        require_cert  = "demand"
    }
}
```

```
dictionary_mapping = ${confdir}/ldap.attrmap
password_attribute = clearPassword
edir_account_policy_check = no
# groupname_attribute = radiusGroupName
# groupmembership_filter = "(dialupAccess=%u)"
# groupmembership_attribute = radiusGroupName
# keepalive {
#     idle = 60
#     probes = 3
#     interval = 3
# }
}
```

Además voy a incluir la configuración de freeradius para realizar autenticaciones contra LDAP sin cifrar.

```
[...]
ldap {
    port = "389"
    server = "ldap.example.com"
    identity = "cn=radius,dc=ldap,dc=example,dc=com"
    password = "root$root"
    basedn = "dc=ldap,dc=example,dc=com"
    filter = "(uid=%{%{Stripped-User-Name}}:-{%User-Name}})"
    #base_filter = "(objectclass=radiusprofile)"

    ldap_connections_number = 5
    timeout = 4
    timelimit = 3
    net_timeout = 1

    tls {
        start_tls = no
    }

    dictionary_mapping = ${confdir}/ldap.attrmap
```

```
password_attribute = clearPassword
edir_account_policy_check = no
# groupname_attribute = radiusGroupName
# groupmembership_filter = "(dialupAccess=%u)"
# groupmembership_attribute = radiusGroupName
# keepalive {
#     idle = 60
#     probes = 3
#     interval = 3
# }
}
[...]
```

## **users**

En el fichero `/etc/freeradius/users` especificamos las reglas de control de acceso, similar a los ficheros ACL de Cisco.

- La entrada `DEFAULT` representa cualquier usuario.
- Las entradas del fichero `users` son procesadas en orden desde el principio hasta el final del archivo.
- Si una entrada contiene el item `Fall-Through = No`, el procesamiento del fichero se detiene y no se procesarán más entradas.
- Si una entrada contiene el item `Fall-Through = Yes`, se continúa el procesamiento de la siguiente entrada en orden.
- Si la solicitud de acceso no coincide con ninguna de las entradas, la solicitud será rechazada.

Además podemos definir un control de horarios para que el servidor sólo permita conexiones dentro de esos rangos.

Considero que para nuestra infraestructura, este fichero es irrelevante, debido a que el encargado de controlar el acceso a internet es el Proxy/caché. Todos los parámetros se mantendrán por defecto.

## ***clients.conf***

En el fichero `clients.conf` definimos los clientes que van a tener acceso a nuestro servidor radius: nuestros puntos de acceso.

Se configurarán las ip de los puntos de accesos de las aulas, un secreto compartido el cual configuraremos en los puntos de acceso y un alias o nombre corto para identificar rápidamente cada cliente.

```
client 172.22.100.11 {  
    secret = EsUnaClaveSecreta  
    shortname = ap2asir  
}  
  
client 172.22.100.13 {  
    secret = EsUnaClaveSecreta  
    shortname = ap2smr  
}
```

## **10. Arranque de freeradius**

Como todos los servicios, freeradius dispone de un script en `/etc/init.d/freeradius` con las opciones que ya sabemos (arrancar, parar, reiniciar, cargar o forzar la carga de la configuración).

Además podemos arrancar freeradius en modo debug en lugar de arrancarlo en modo daemon, por ejemplo para comprobar si se está aplicando la configuración que pretendíamos o para ver si reciba y procesa bien las peticiones de los clientes.

Para ello, paramos el daemon y luego ejecutamos:

```
# /etc/init.d/freeradius stop  
# freeradius -X
```

## **11. Freeradius Alta Disponibilidad**

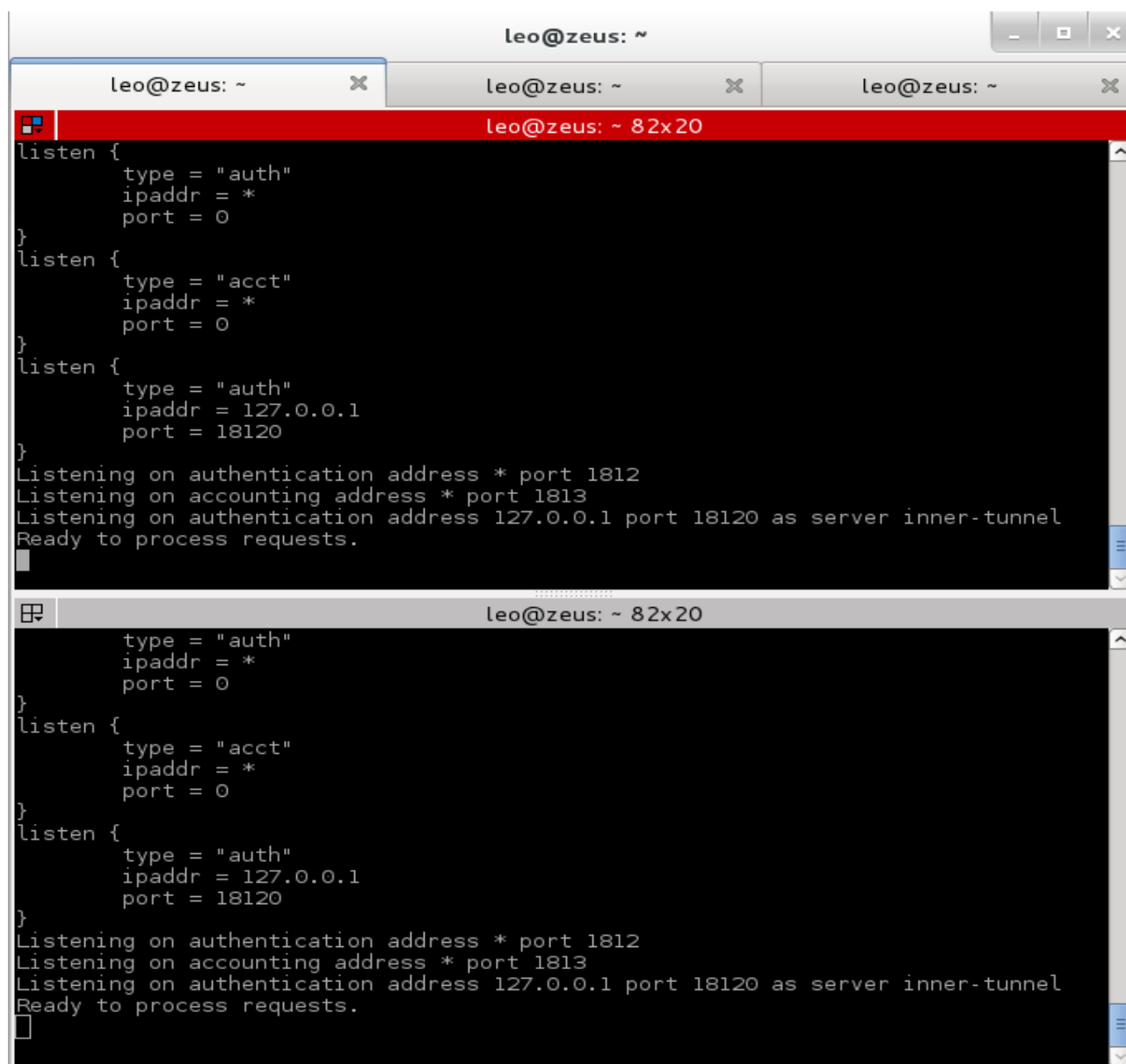
Para montar freeradius en alta disponibilidad vamos a hacer uso de una característica de nuestro punto de acceso; **Primary Radius - Secondary Radius**.

Habr  que definir la ip de cada servidor, el puerto por el que escucha peticiones (1812/udp) y una clave compartida, pudiendo ser diferentes para cada servidor. Es recomendable usar como m nimo una contrase a de 16 bits con may sculas y min sculas.

Cuando le llegue una petici n a nuestro punto de acceso,  l intentar  conectar con el servidor primario, si  ste no respondiera pasar  la petici n al secundario. Si esto ocurriera, nuestro Access Point registrar  a nuestro servidor secundario como favorito y a partir de entonces todas las peticiones ser n redirigidas a  l.

Para los usuarios todos estos cambios son irrelevantes.

La configuraci n del Radius secundario es exactamente la misma que la del primario. A continuaci n vamos los dos servidores escuchando en modo debug.



The image shows two terminal windows from a user named 'leo' on a machine named 'zeus'. The top window has a red title bar and displays the configuration for the Radius server. It shows three 'listen' blocks: one for authentication on port 1812, one for accounting on port 1813, and one for authentication on 127.0.0.1 port 18120 as an inner-tunnel server. The bottom window has a grey title bar and displays the same configuration, but with the status messages at the bottom indicating that the server is ready to process requests.

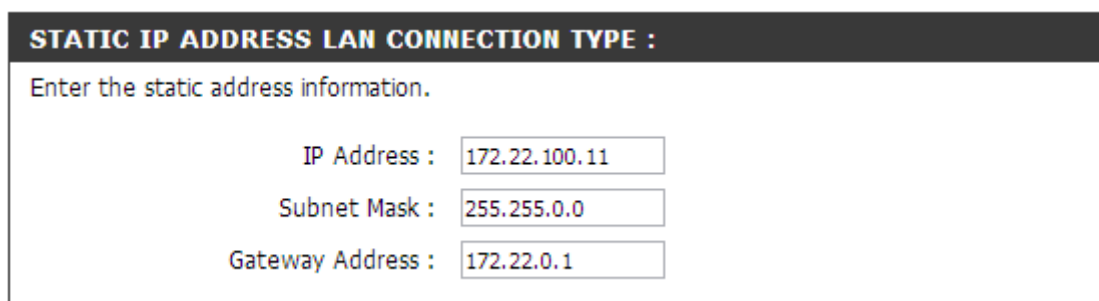
```
leo@zeus: ~  
leo@zeus: ~ 82x20  
listen {  
    type = "auth"  
    ipaddr = *  
    port = 0  
}  
listen {  
    type = "acct"  
    ipaddr = *  
    port = 0  
}  
listen {  
    type = "auth"  
    ipaddr = 127.0.0.1  
    port = 18120  
}  
Listening on authentication address * port 1812  
Listening on accounting address * port 1813  
Listening on authentication address 127.0.0.1 port 18120 as server inner-tunnel  
Ready to process requests.  
  
leo@zeus: ~ 82x20  
type = "auth"  
ipaddr = *  
port = 0  
}  
listen {  
    type = "acct"  
    ipaddr = *  
    port = 0  
}  
listen {  
    type = "auth"  
    ipaddr = 127.0.0.1  
    port = 18120  
}  
Listening on authentication address * port 1812  
Listening on accounting address * port 1813  
Listening on authentication address 127.0.0.1 port 18120 as server inner-tunnel  
Ready to process requests.
```

En el siguiente apartado veremos la configuración de los puntos de accesos para que respondan en alta disponibilidad.

## 12. Configuración de puntos de acceso

La configuración del punto de acceso puede variar dependiendo del modelo.

Lo primero que se debe hacer es reservar una ip estática de nuestro rango, máscara de red y puerta de enlace.



**STATIC IP ADDRESS LAN CONNECTION TYPE :**

Enter the static address information.

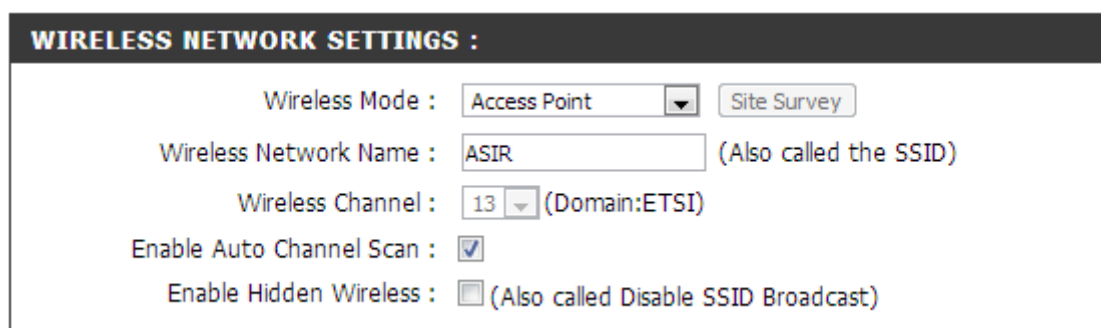
IP Address : 172.22.100.11

Subnet Mask : 255.255.0.0

Gateway Address : 172.22.0.1

Es recomendable cambiar el nombre de usuario y contraseña para acceder a la configuración del dispositivo.

A continuación configuramos el SSID.



**WIRELESS NETWORK SETTINGS :**

Wireless Mode : Access Point

Wireless Network Name : ASIR (Also called the SSID)

Wireless Channel : 13 (Domain:ETSI)

Enable Auto Channel Scan : ☒

Enable Hidden Wireless : ☐ (Also called Disable SSID Broadcast)

Y por último, configuramos el AP como cliente de nuestros servidores freeradius.

Como ya sabemos la seguridad inalámbrica será WPA2-Enterprise, Ip, Puerto y Frase de paso (primario y secundario).



WIRELESS SECURITY MODE :	
Security Mode :	Enable WPA2 Wireless Security (enhanced) ▼
WPA2 :	
WPA2 requires stations to use high grade encryption and authentication.	
Cipher Type : AUTO ▼	
PSK / EAP : Enterprise ▼	
802.1X	
RADIUS Server 1 : IP	192.168.2.2
Port	1812
Shared Secret	EsUnaClaveSecreta
RADIUS Server 2 : IP	172.22.0.49
Port	1812
Shared Secret	EsUnaClaveSecreta

### 13. Configuración de clientes

Como hemos comentado en el documento, la autenticación se realiza solo con certificados de servidor y no es necesario generar certificados para cada cliente nuevo que desee conectarse a la red, para facilitar el despliegue.

De forma opcional los clientes podrán seleccionar la autoridad certificadora (CA) del servidor radius que podrán descargar desde el NAS de nuestro centro con el nombre **ca.pem**

## Linux

Esta configuración es válida para Debian y Ubuntu.

Vamos a conectarnos sin CA (ca.pem) teniendo en cuenta que veremos un mensaje avisándonos de ello.

Seleccionamos la red (ASIR o SMR),

- Wireless Security    WPA & WPA2 Enterprise
- Authentication        Tunneled TLS
- CA certificate        (Ninguno)
- Inner Authentication   PAP
- Username              mi\_usuario
- Password              Contraseña

Debemos introducir el nombre de usuario y contraseña de LDAP (con los que accedemos a la plataforma).



**Authentication required by wireless network**

Passwords or encryption keys are required to access the wireless network 'ASIR'.

Wireless security: WPA & WPA2 Enterprise

Authentication: Tunneled TLS

Anonymous identity:

CA certificate: (Ninguno)

Inner authentication: PAP

Username: leo

Password: .....

☐ Ask for this password every time

☐ Show password

Cancelar Conectar

Veremos un aviso, diciéndonos que no estamos usando Autoridad Certificadora (CA). Como hemos comentado, su uso no es obligatorio, pero si recomendable.

Pulsamos sobre Ignore y en pocos segundos estaremos conectados a nuestra red inalámbrica.



Para conectarnos usando la Autoridad Certificadora del servidor (recomendado):

Seleccionamos la red (ASIR o SMR),

- Wireless Security      WPA & WPA2 Enterprise
- Authentication        Tunneled TLS
- CA certificate         ca.pem
- Inner Authentication   PAP
- Username              mi\_usuario
- Password              Contraseña

Debemos introducir el nombre de usuario y contraseña de LDAP (con los que accedemos a la plataforma Moodle).



**Authentication required by wireless network**

Passwords or encryption keys are required to access the wireless network 'ASIR'.

Wireless security: WPA & WPA2 Enterprise

Authentication: Tunneled TLS

Anonymous identity:

CA certificate: ca.pem

Inner authentication: PAP

Username: leo

Password: .....

☐ Ask for this password every time

☐ Show password

Cancelar Conectar

## Windows

Para conectarse a la red inalámbrica de nuestro centro (igual que ocurre en redes como Eduroam) necesitamos el programa SecureW2.

Esta configuración es válida para todas las versiones de Windows, excepto para Windows 8, el cual ya incorpora autenticación EAP-TTLS/PAP de forma nativa.

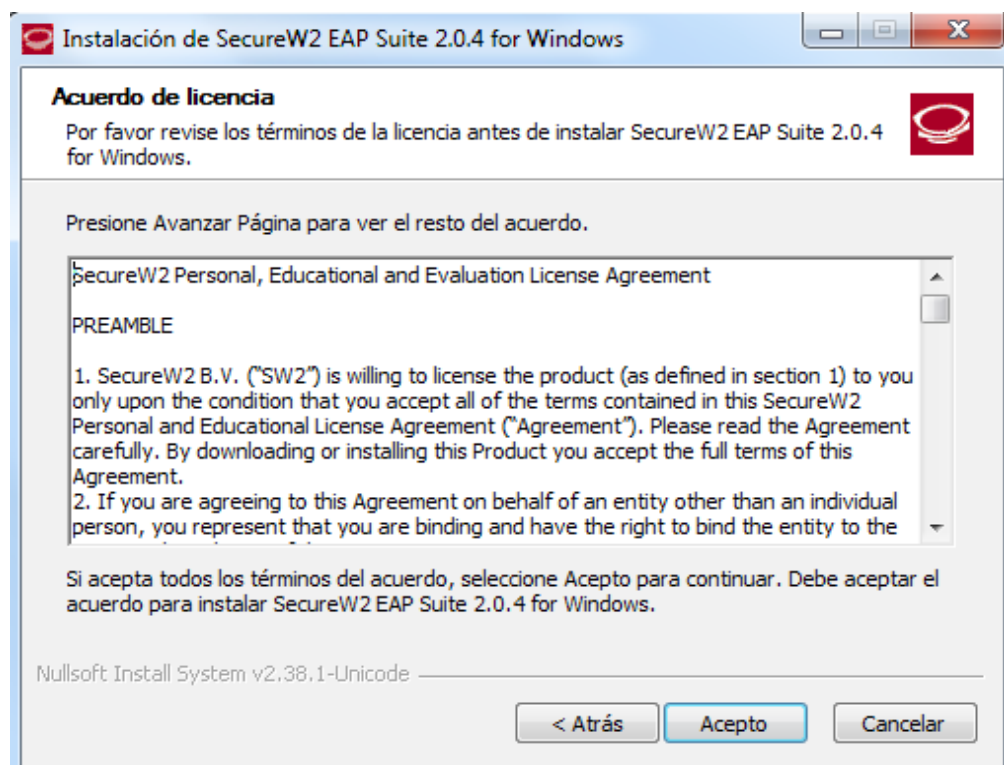
1. Lo primero será descargar SecureW2 e instalarlo.

[http://cdn.redeszone.net/download/soft/wifi/SecureW2\\_Windows7.zip](http://cdn.redeszone.net/download/soft/wifi/SecureW2_Windows7.zip)

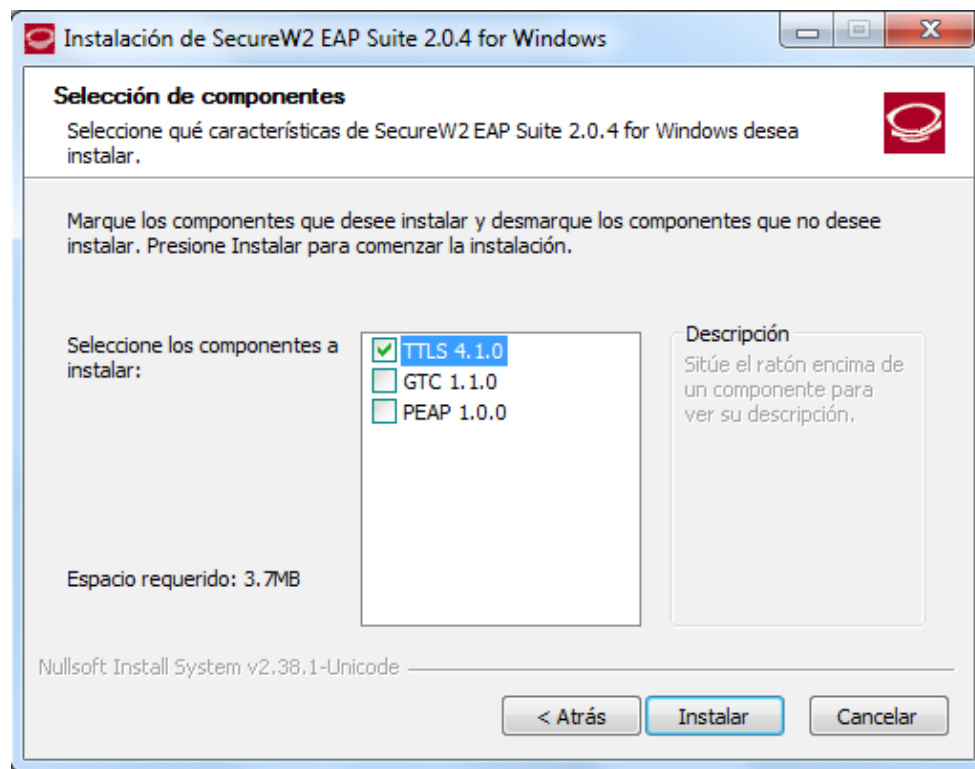
Descomprimos el ZIP y lo ejecutamos como Administrador.



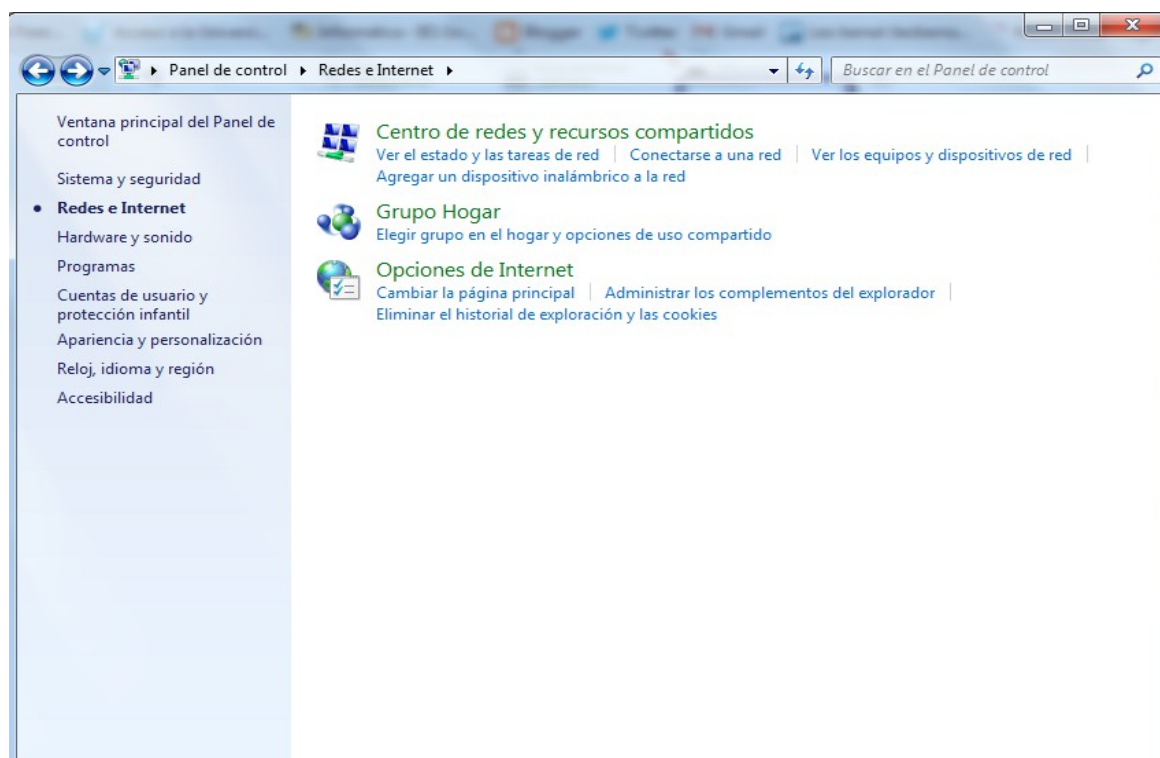
Aceptamos los términos de licencia y contrato.



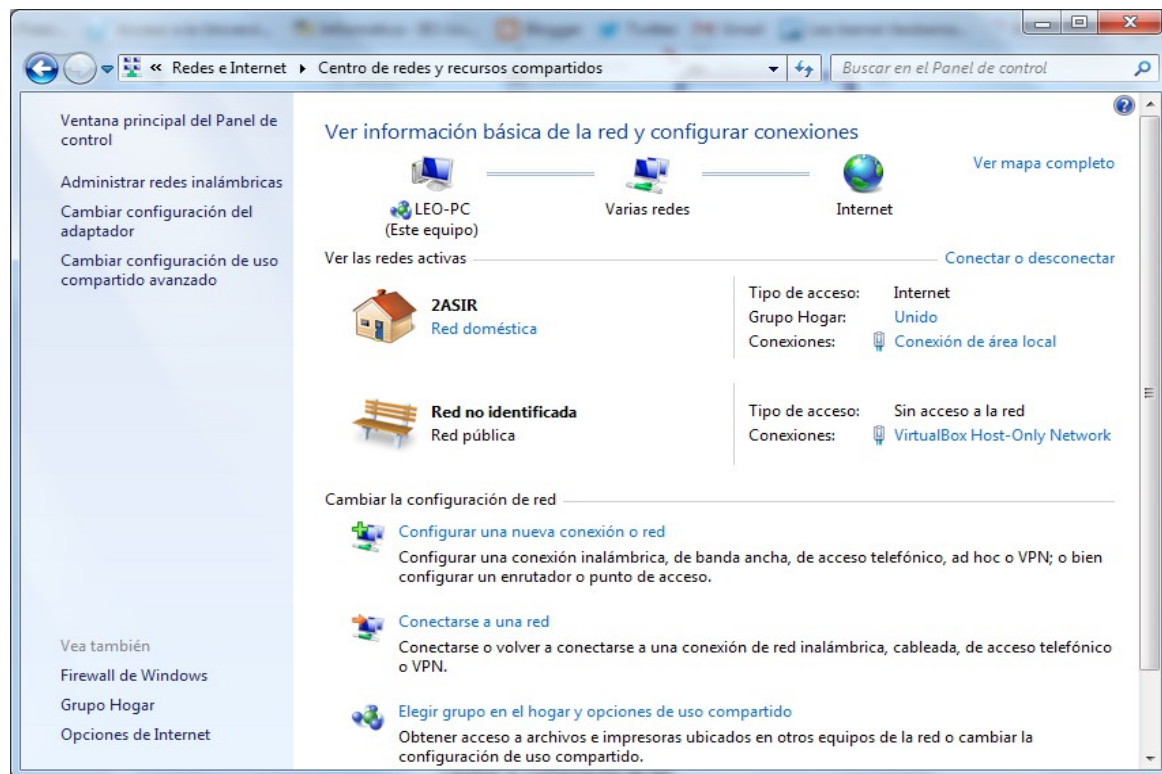
Seleccionamos el componente TTLS e instalamos.



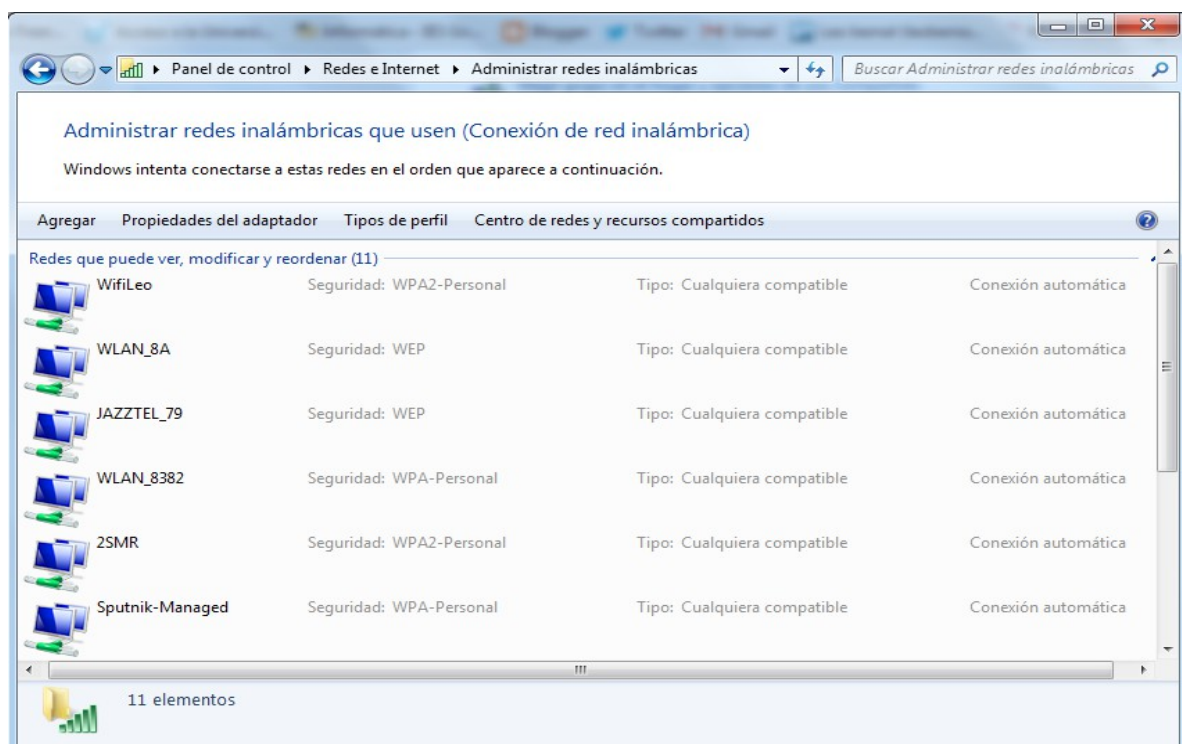
Una vez instalado nos pedirá reiniciar el equipo. Lo reiniciamos y luego, nos vamos a **Inicio/Panel de Control** y nos metemos en **Centro de Redes y Recursos Compartidos**.



## Ahora nos vamos a **Administrar Redes Inalámbricas**

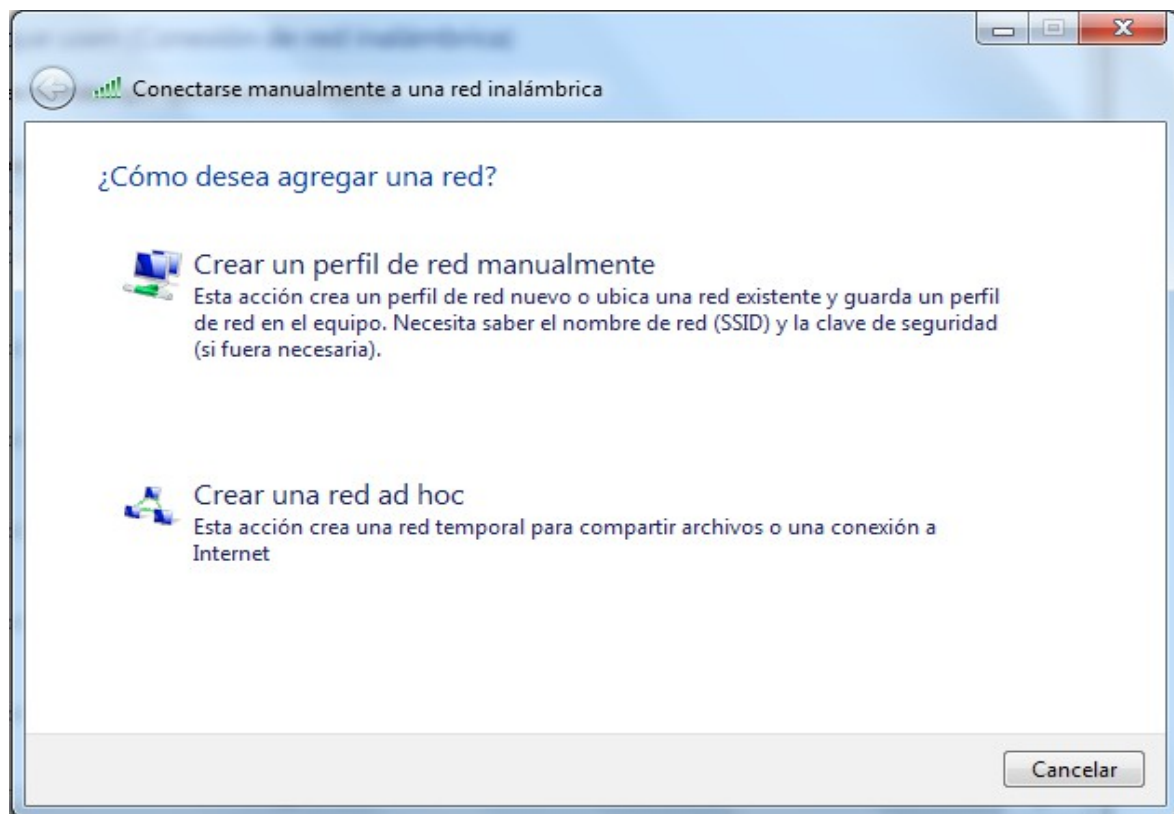


Nos aparecerán las redes guardadas que tenemos, pulsamos en **Agregar** para agregar una nueva conexión.

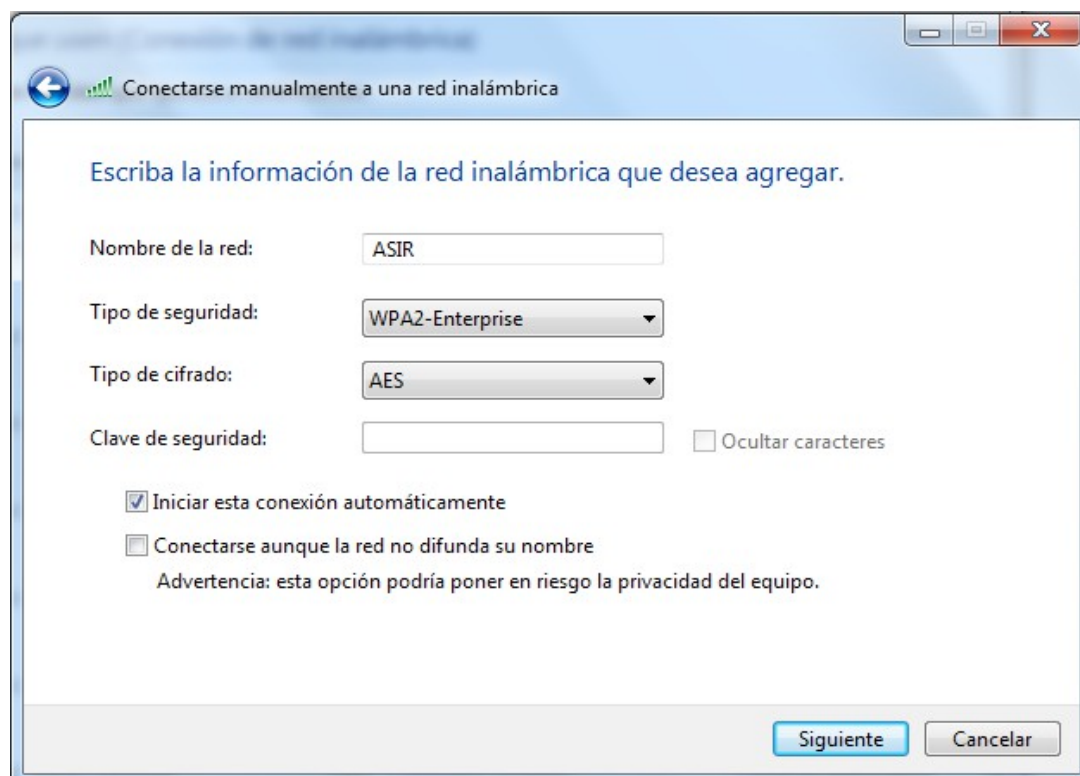




Ahora le damos a Crear un perfil de red manualmente

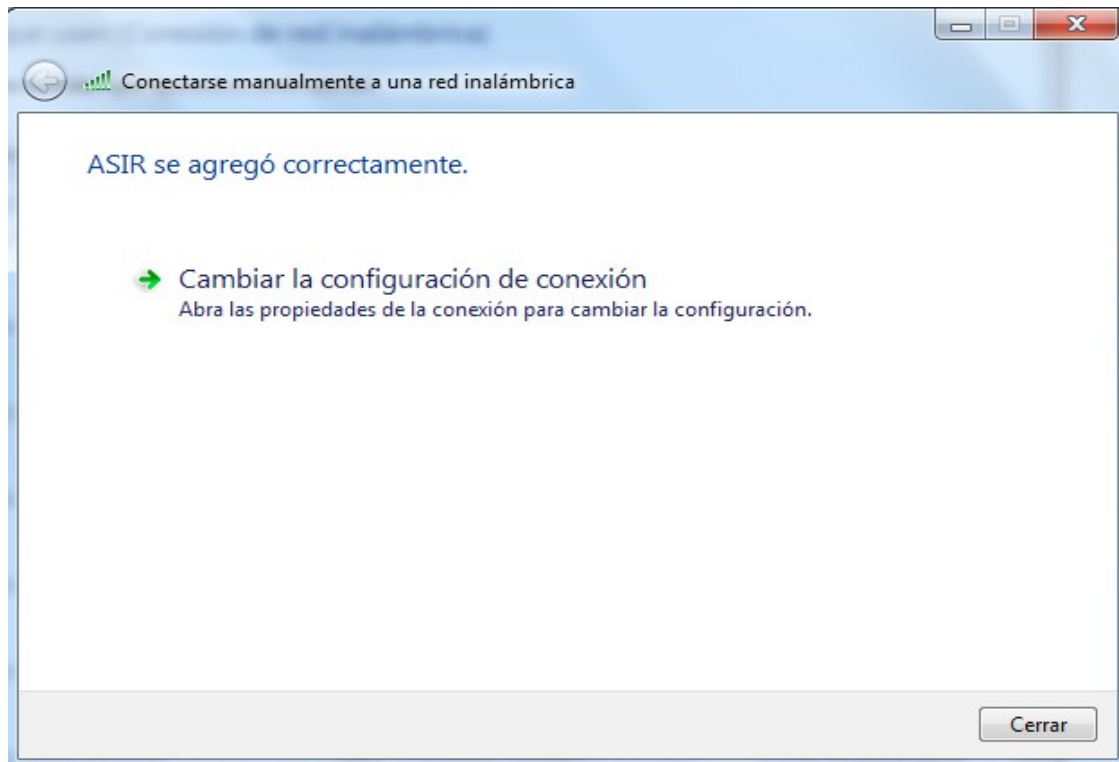


Ponemos los datos de la red.

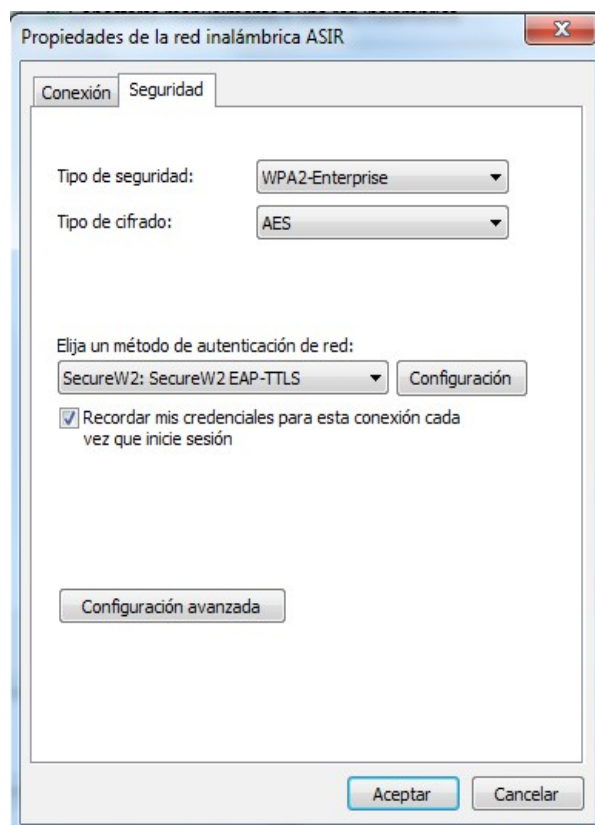




Pulsamos en siguiente y le damos a **Cambiar la configuración de la conexión** para configurar el SecureW2.



Elegimos el método de **autenticación de red (SecureW2)** y pulsamos en configurar.



Ahora en el perfil **Default** (no cambies el nombre) pulsamos en Configurar.



Y ahora ponemos todas las opciones como en las capturas:





Método de **autenticación PAP** (importante).



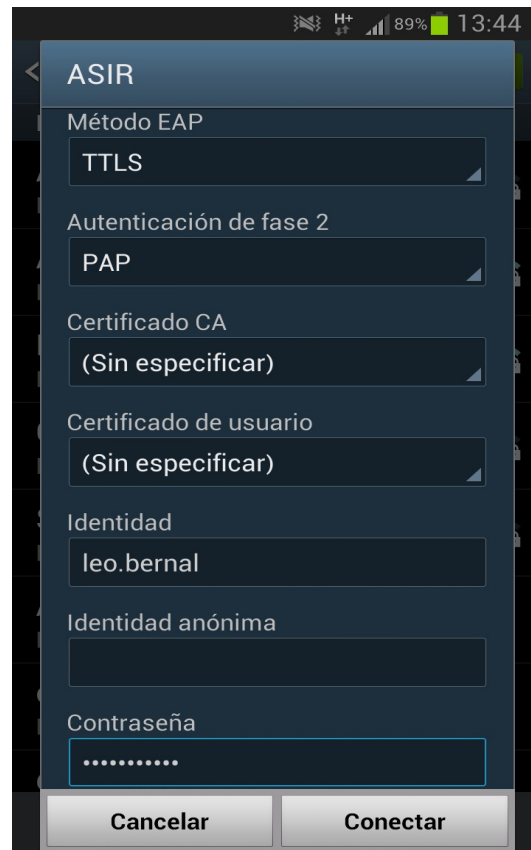
Ahora deseccionamos la casilla **Pedir credenciales de usuario**, e introducimos nombre de usuario y contraseña de LDAP. Aceptamos todas la configuración, seleccionamos el SSID (ASIR o SMR) y en pocos segundo nuestro equipo se conectará a la red.



The image shows the SecureW2 authentication dialog box. It has a red header with the SecureW2 logo. Below the header, there are four tabs: 'Conexión', 'Certificados', 'Autenticación', and 'Cuenta de usuario'. The 'Cuenta de usuario' tab is selected. Inside this tab, there is a checkbox labeled 'Pedir credenciales de usuario' which is unchecked. Below this, there are three input fields: 'Usuario:' with the text 'leo', 'Contraseña:' with masked characters '\*\*\*\*\*', and 'Dominio:' which is empty. At the bottom of the tab, there is another checkbox labeled 'Usar esta cuenta para entrar al ordenador' which is also unchecked. At the bottom of the dialog box, there are three buttons: 'Avanzado', 'Aceptar', and 'Cancelar'.

## Android

Android admite autenticación EAP-TTLS/PAP de forma nativa. Para conectarnos a nuestra red, sólo tenemos que seleccionar el SSID (ASIR o SMR) y marcar las siguientes opciones, introduciendo nuestro nombre de usuario y contraseña:



Veremos nuestro dispositivo conectado correctamente.



## **Ipnone**

Ipnone admite autenticación EAP-TTLS/PAP de forma nativa, pero tenemos que realizar la configuración desde nuestro Pc o Mac.

1. Lo primero será descargar e instalar el software “*Utilizada de configuración de iPhone*” en nuestro equipo, a partir del siguiente enlace:

<http://support.apple.com/kb/DL1466>

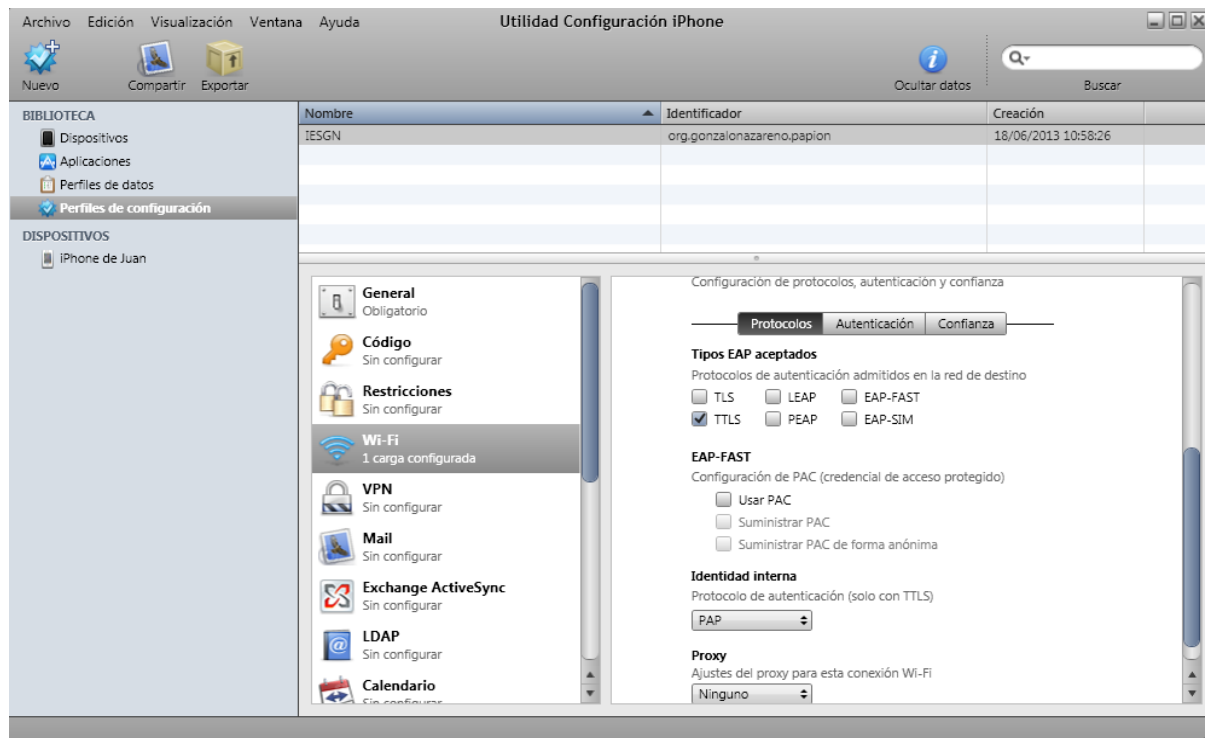
2. Una vez instalado, conecte su iPhone/iPod a su Pc o Mac por cable y comprobamos que sincroniza correctamente con la aplicación que acabamos de instalar.

3. Su dispositivo aparecerá reconocido en el menú de la izquierda de la ventana de Utilidad de configuración de iPhone.

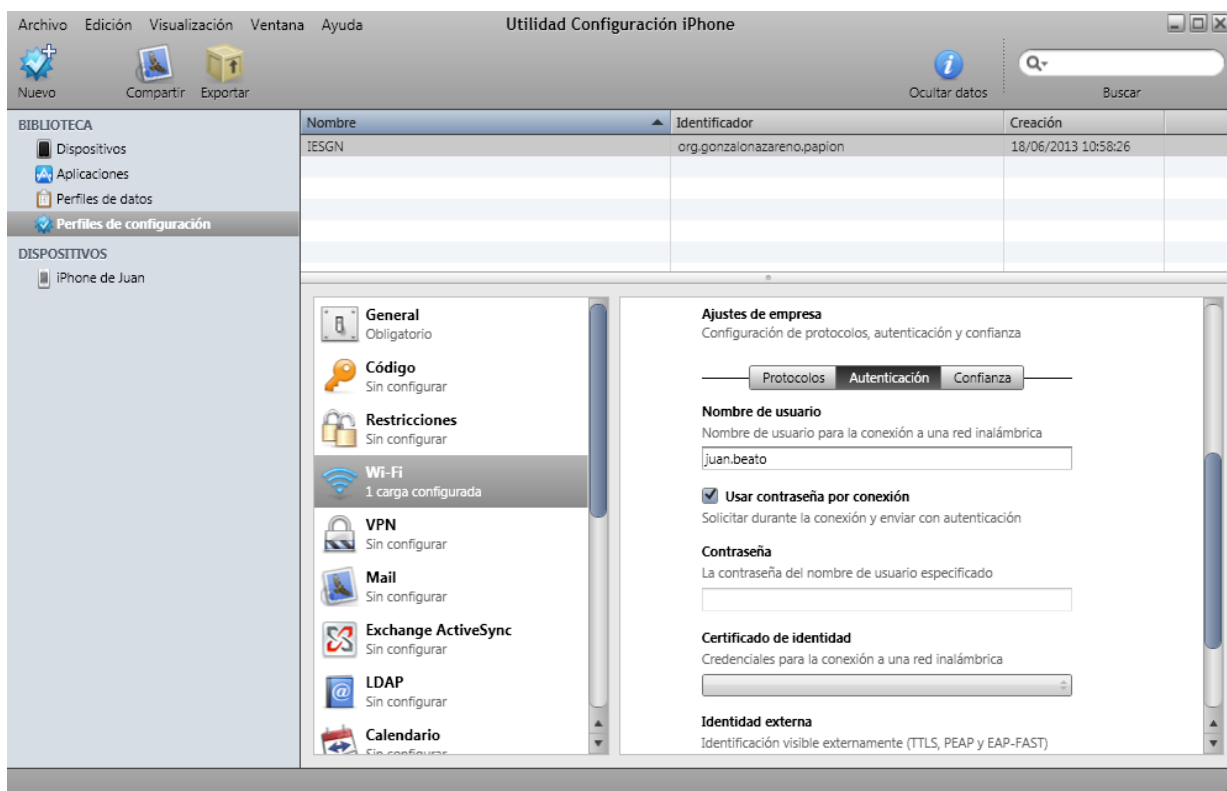


4. Seleccione la pestaña “*Perfiles de configuración*” y en el menú desplegable marquen “*Wifi*” y añadan las siguientes opciones para configurar los protocolos.

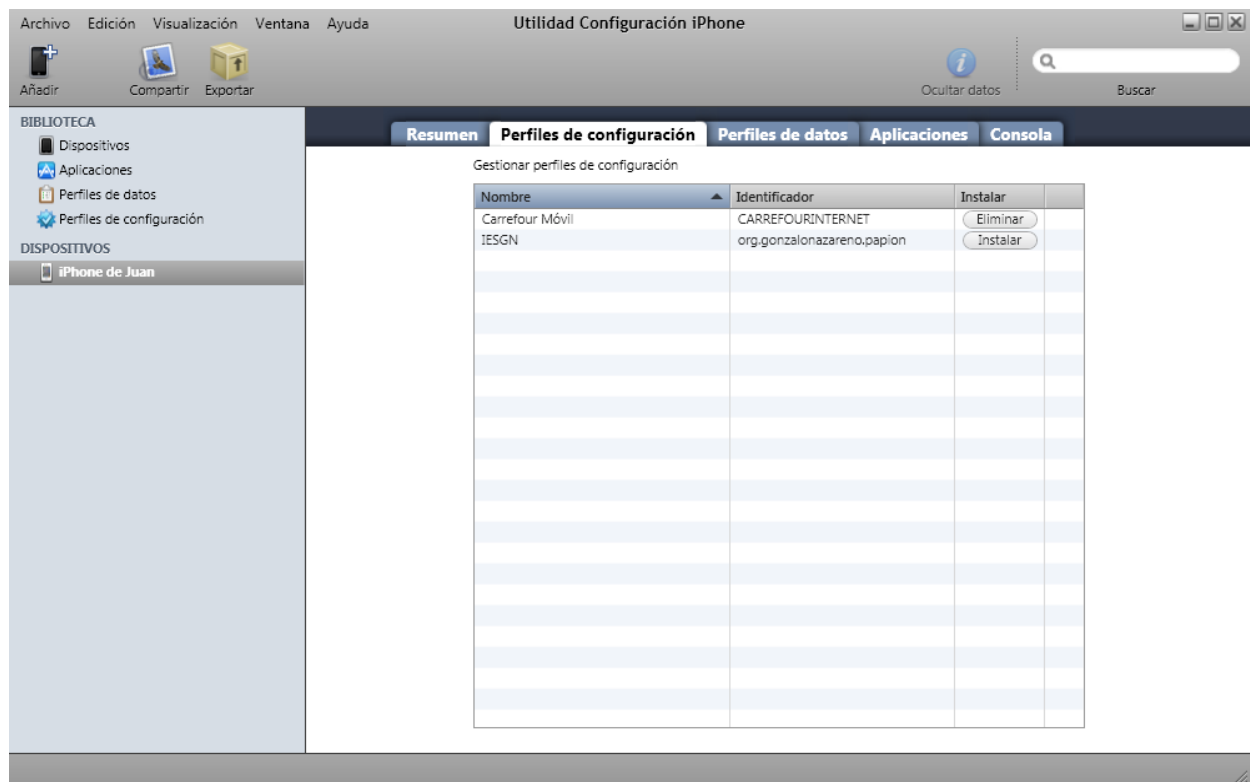
**IMPORTANTE:** Ten en cuenta que debemos introducir un nombre a nuestro perfil (*IESGN*) y un identificador obligatorio (*org.gonzalonazareno.papion*).



5. En “Autenticación” introducimos nuestro nombre de usuario de LDAP y marcamos “Usar contraseña por conexión”.



6. Ahora pulsamos sobre nuestro dispositivo y veremos el perfil que acabamos de crear (*IESGN*). Pulsamos “*Instalar*”.



7. En la pantalla de su iPhone/iPod aparecerá la siguiente ventana. Pulsamos “*Instalar*” para añadir nuestro perfil.





8. Una vez instalado el perfil, seleccionamos el SSID (*SMR* en este caso) e introducimos nuestro nombre de usuario y contraseña de LDAP.



9. En pocos segundos nuestro dispositivo se conectará a la red.

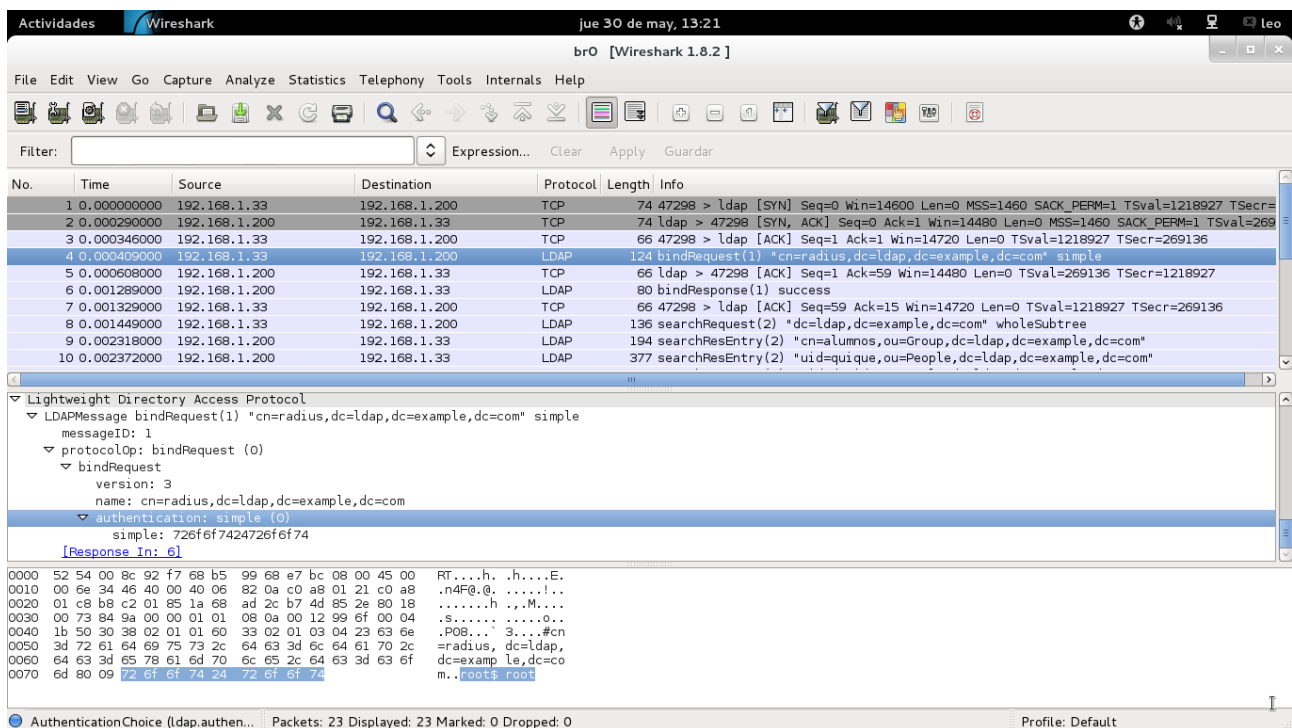


## 14. Comunicación cifrada

Se han realizado varias pruebas con el sniffer *Wireshark* para confirmar que efectivamente toda la comunicación está cifrada. Para ello se ha analizado la red sin cifrar y luego cifrada para comparar así los resultados.

Empecemos con LDAP, permitiendo sólo el tráfico por el puerto 389/TCP.

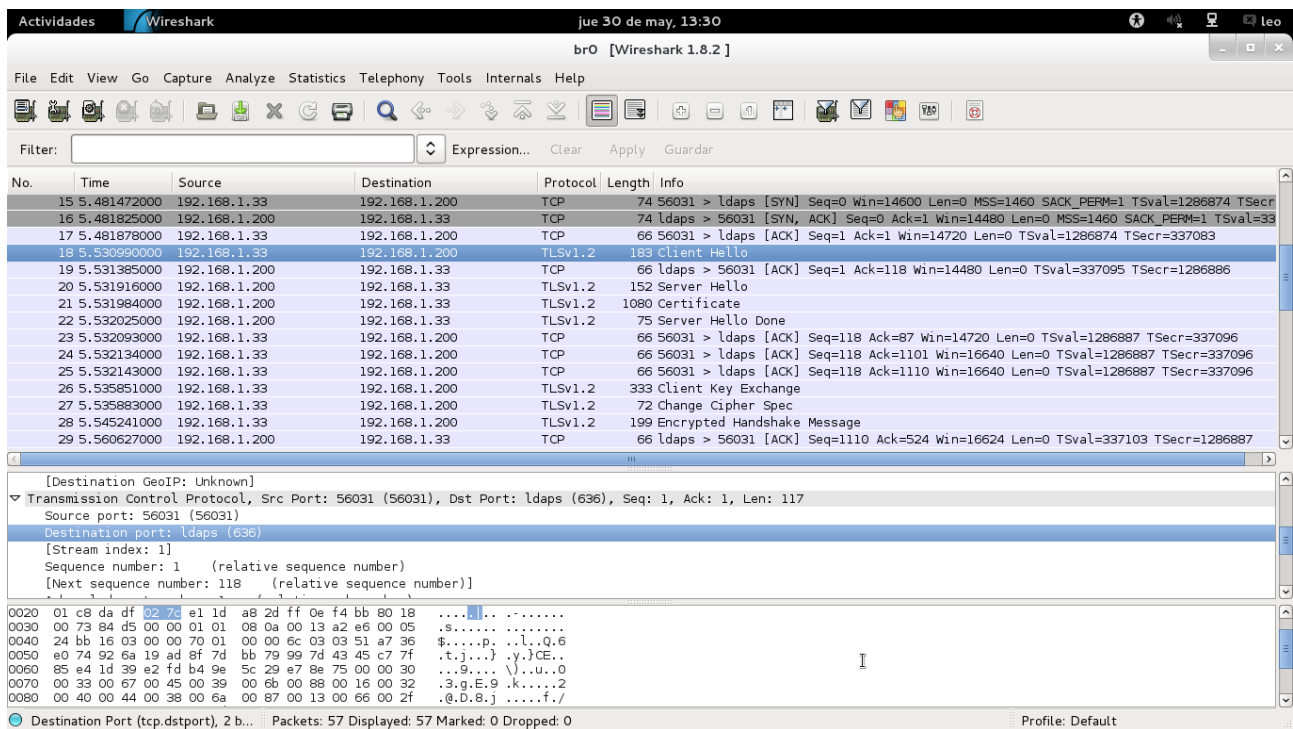
```
root@ldap:~# netstat -puttan | grep -i slapd
tcp        0      0 0.0.0.0:389          0.0.0.0:*            LISTEN      2542/slapd
tcp        0      0 192.168.1.200:389    192.168.1.33:47294    ESTABLISHED 2542/slapd
tcp6       0      0 :::389              :::*                  LISTEN      2542/slapd
root@ldap:~#
```



Podemos observar el tipo de autenticación simple del protocolo LDAP, siendo la contraseña de usuario “root\$root”.

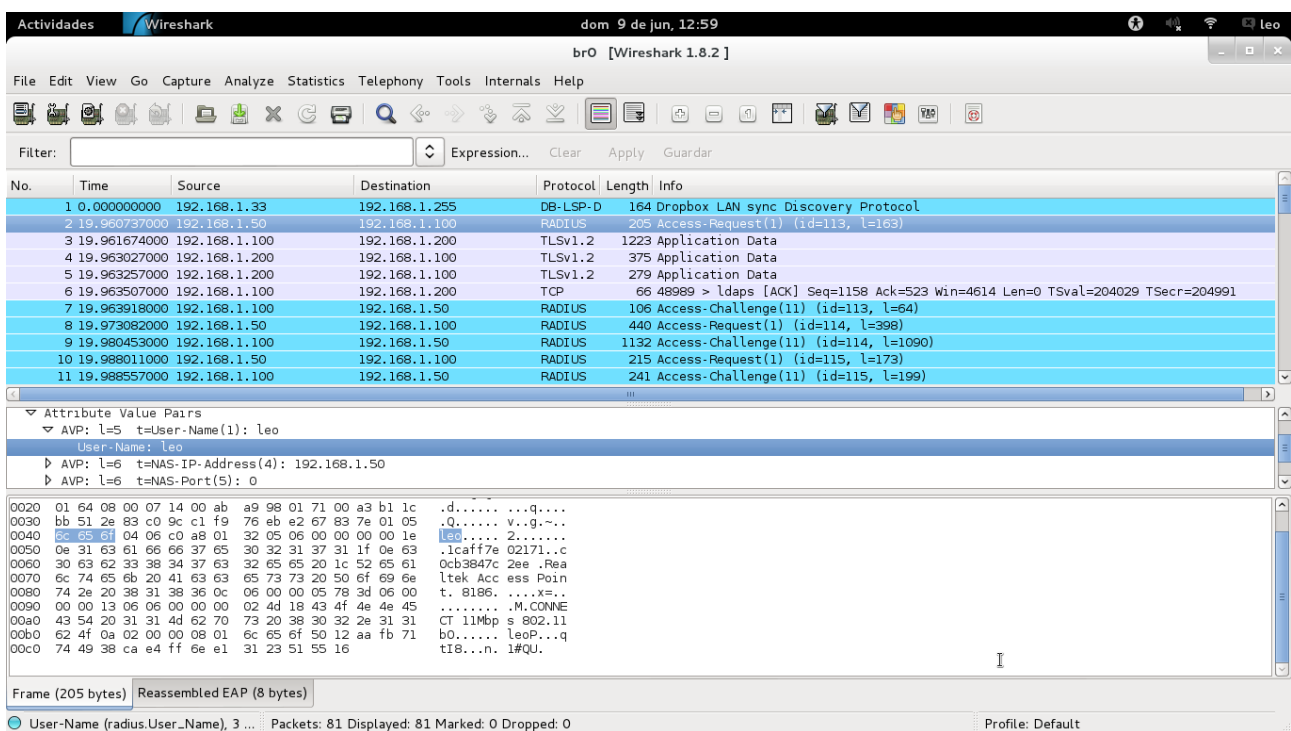
Ahora sólo permitimos la comunicación por el puerto 636/TCP, es decir, debemos hacer uso de certificados de servidor para realizar autenticaciones.

```
root@ldap:~# netstat -puttan | grep -i slapd
tcp        0      0 0.0.0.0:636          0.0.0.0:*            LISTEN      2668/slapd
tcp        0      0 192.168.1.200:636    192.168.1.33:56029    ESTABLISHED 2668/slapd
tcp6       0      0 :::636              :::*                  LISTEN      2668/slapd
root@ldap:~#
```



Como podemos ver, los datagramas son totalmente diferentes, la comunicación se realiza por LDAPS, identifica el certificado y cifra todas las credenciales “*Encrypted Handshake Messenger*”. Ningún mensaje reflejará las contraseñas en claro.

Por último, comprobamos los mensajes del servidor Radius:



La Ip origen corresponde al punto de acceso y la ip destino al servidor freeradius. Radius establece un túnel cifrado por el que viajan todas las contraseñas. Observamos que el único dato que circula sin cifrar es el nombre de usuario “leo”.

De esta forma hemos comprobado que aunque el método de autenticación que usa internamente sea PAP (sin cifrar) no nos afecta, debido a que freeradius establece túneles cifrados entre cliente-servidor gracias al método EAP-TTLS aportando a nuestra red un alto nivel de seguridad.

## **15. Vulnerabilidades**

La infraestructura EAP-TTLS/PAP presenta un escenario muy robusto, y complejo, poniendo realmente a prueba a un atacante ilícito, pero, nada es perfecto.

A continuación voy a comentar posibles vulnerabilidades o puntos débiles.

### **1. *Protocolos de autenticación.***

Podemos pensar que aunque se establezcan túneles cifrados entre el cliente y el servidor, si un atacante pudiera romper dicho túnel, lo tendría bastante fácil, debido a que los protocolos usados están rotos. De ahí a usarlos siempre acompañados de métodos de autenticación (TTLS).

- PAP viaja en texto plano.
- CHAP usa MD5 como algoritmo, actualmente roto.
- MS-CHAPv2 según la web “Una al Día” en una noticia de Septiembre de 2012, se desarrollaron métodos los cuales son capaces de descifrar cualquier contraseña MS-CHAP-v2 en menos de 24 horas.

Esto no son vulnerabilidades en sí, sino debilidades criptográficas de los protocolos, por tanto no serán solucionadas con actualizaciones.

Ahora bien, antes de llegar a este punto, ¿podemos romper un túnel cifrado con certificados de servidor? Es poco probable, aunque considero que gran parte de esta responsabilidad recae sobre los clientes, con ataques de ingeniería inversa, pudiéndoles engañar para confiar en autoridades certificadoras desconocidas.

## 2. Que se comprometa directamente la máquina.

Ni que decir tiene que si un atacante consigue acceder al servidor, aprovechando alguna vulnerabilidad no parcheada del propio sistema operativo podrá interceptar mensajes y realizar todo tipo de opciones.

## 3. Rogue AP (con Backtrack o Kali Linux)

Una de las aproximaciones más interesantes para el robo de información en redes WiFi es la técnica conocida como Rogue AP o suplantación del punto de acceso. La idea de esta técnica de ataque es conseguir que la víctima se conecte al equipo del atacante, que funciona como un punto de acceso legítimo, para que sea éste el que redirija el tráfico. Es una forma sencilla de realizar un ataque de *Man In The Middle* ya que al estar el atacante realizando funciones de AP va a poder interceptar absolutamente todas las comunicaciones.

Para que el ataque tenga efectividad, es necesario que la suplantación de un AP legítimo sea lo más real posible, por lo que se debe recrear un entorno de red con las mismas características en el Rogue AP a las del AP legítimo, copiando para ello el BSSID, ESSID y las configuraciones de seguridad de la red.

