



Wheezy Stable

*free***RADIUS**

con usuarios registrados en



INDICE

Paginas

- 3 - 6 - Instalacion, configuracion
administración y prueba de LDAP.
- 7 - 10 - Instalacion, configuracion ,
administración y prueba de Freeradius
- 10 – 11 - Configuración del punto de acceso
(AP) para usar Radius.
- 12 – 15 -Prueba del funcionamiento del
servidor con error de credenciales.
- 16 - 18 - Configurando PEAP.
- 18 – 33 - Pruebas de las conexiones wifis
desde varios sistemas y modificando
de la configuracion de las conexiones.
Prueba del buen funcionamiento del
servidor.
- 33 - 38 - Problemas encontrados con sus
respectivos procedimientos para
solucionarlos.

Características del servidor

- Sistema operativo: Debian 7 Wheezy Stable 32b
- RAM: 1024MB
- HDD: 15GB
- Sistema de virtualización: Oracle VM VirtualBox

```
root@peasodebian:/home/usuario# uname -a
Linux peasodebian 3.2.0-4-686-pae #1 SMP Debian 3.2.41-2+deb7u2 i686 GNU/Linux
root@peasodebian:/home/usuario#
```

¿Qué es OpenLDAP?

OpenLDAP es una implementación libre y de código abierto del protocolo Lightweight Directory Access Protocol (LDAP) desarrollada por el proyecto OpenLDAP.

Está liberada bajo su propia licencia OpenLDAP Public License. LDAP es un protocolo de comunicación independiente de la plataforma.

Muchas distribuciones GNU/Linux incluyen el software OpenLDAP para el soporte LDAP. Este software también corre en plataformas BSD, AIX, HP-UX, Mac OS X, Solaris, Microsoft Windows (NT y derivados, incluyendo 2000, XP, Vista), y z/OS.

Instalación de OpenLDAP

Lo primero vamos a comprobar que nuestro equipo posee un FQDN definido correctamente:

```
root@peasodebian:/home/usuario# hostname --fqdn
peasodebian.azeroth.com
root@peasodebian:/home/usuario#
```

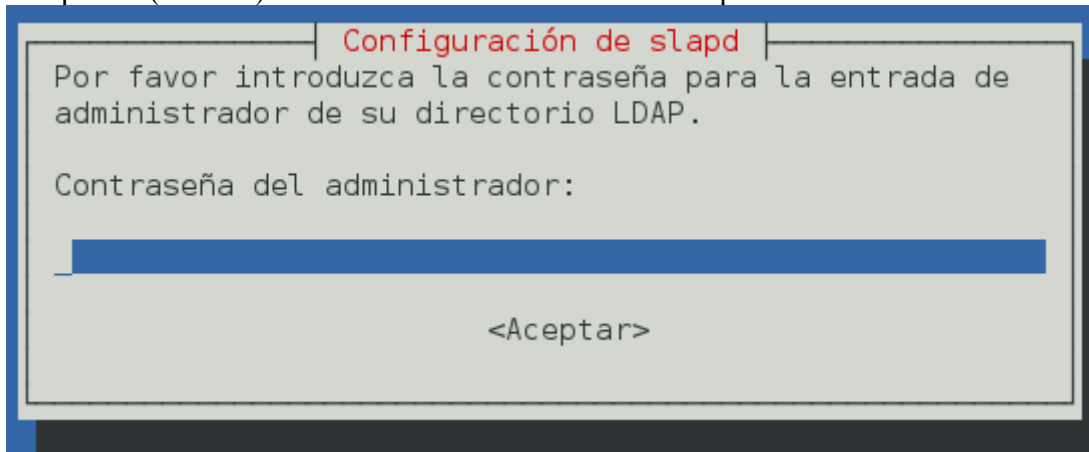
Observamos:

- Nombre del equipo: “peasodebian”.
- Dominio: “azeroth.com”.

Instalamos el paquete “slapd” (y las dependencias que nos proponga):

```
root@peasodebian:/home/usuario# aptitude install slapd
Se instalarán los siguiente paquetes NUEVOS:
  libodbc1{a} libslpl1{a} slapd
0 paquetes actualizados, 3 nuevos instalados, 0 para eliminar y 0 sin
actualizar.
Necesito descargar 2.031 kB de ficheros. Después de desempaquetar se
usarán 4.854 kB.
¿Quiere continuar? [Y/n/?] Y
```

Nos pedirá (2 veces) la contraseña de “Administrador” que tendrá el administrador de LDAP.



Ejecutamos el comando `#dpkg-reconfigure slapd`

- Nombre de dominio DNS: `servidorldap.azeroth.com`
- Nombre de la Organización: `Proyecto Gonzalo Carmona`
- Contraseña del administrador : “la password que queramos”
- Motor de base de datos a utilizar: `BDB`
- ¿Permitir el protocolo LDAPv2?: `No (salvo que sea necesario)`

Cuando acabe la instalación nos devolverá:

```
root@peasodebian:/home/usuario# slapcat
dn: dc=servidorldap,dc=azeroth,dc=com
objectClass: top
objectClass: dcObject
objectClass: organization
o: Proyecto Gonzalo Carmona
dc: servidorldap
structuralObjectClass: organization
entryUUID: c93138d4-5714-1032-8213-b77ded0e10c0
creatorsName: cn=admin,dc=servidorldap,dc=azeroth,dc=com
createTimestamp: 20130522101909Z
entryCSN: 20130522101909.694127Z#000000#000#000000
modifiersName: cn=admin,dc=servidorldap,dc=azeroth,dc=com
modifyTimestamp: 20130522101909Z
```

```
dn: cn=admin,dc=servidorldap,dc=azeroth,dc=com
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator
userPassword:: e1NTSEF9cE15M1VIN1QxZjg0UHJOVGdQZy9MYlhWdmxxeGFOTks=
structuralObjectClass: organizationalRole
entryUUID: c9328022-5714-1032-8214-b77ded0e10c0
creatorsName: cn=admin,dc=servidorldap,dc=azeroth,dc=com
createTimestamp: 20130522101909Z
entryCSN: 20130522101909.702509Z#000000#000#000000
modifiersName: cn=admin,dc=servidorldap,dc=azeroth,dc=com
modifyTimestamp: 20130522101909Z
```

Creamos dos unidades organizativas (People y Group)

Ejecutamos el comando “#slappasswd”, introducimos una contraseña dos veces y nos devolverá algo parecido a: {SSHA}IRI0WLbbmbyAt9s/DEYDL8OzYvvP+R/Q. Esto es la contraseña que tenemos que asignar en el campo “UserPassword” para introducir al usuario.

Creamos un archivo llamado “base.ldif” (en realidad da igual como se llame el archivo, es para organizarnos), con el siguiente contenido:

```
dn: ou=People,dc=servidorldap,dc=azeroth,dc=com
ou: People
objectClass: top
objectClass: organizationalUnit
```

```
dn: ou=Group,dc=servidorldap,dc=azeroth,dc=com
ou: Group
objectClass: top
objectClass: organizationalUnit
```

Paramos el servicio slapd (/etc/init.d/slapd stop).
Agregamos las nuevas entradas con el siguiente comando:
#slapadd -l base.ldif

Editamos el archivo base.ldif, borramos su contenido y agregamos lo siguiente:

```
dn: cn=pruebag,ou=Group,dc=servidorldap,dc=azeroth,dc=com
objectClass: posixGroup
objectClass: top
cn: pruebag
gidNumber: 2000
```

```
dn: uid=pruebau,ou=People,dc=servidorldap,dc=azeroth,dc=com
uid: pruebau
cn: Usuario de prueba
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: shadowAccount
userPassword: {SSHA}IRI0WLbbmbyAt9s/DEYDL8OzYvvP+R/Q
loginShell: /bin/bash
uidNumber: 2000
gidNumber: 2000
homeDirectory: /home/pruebau
gecos: Usuario de prueba
host: *
```

Volvemos hacer “#slapadd -l base.ldif”.

Administración de LDAP de forma gráfica

Una vez introducido lo básico del dominio, vamos a trabajar de una forma visual más agradable, ya que el principal propósito de este proyecto no es usar y enseñar cada uno de los comandos de OpenLDAP.

Instalamos el siguiente paquete desde el repositorio:

```
#aptitude install phpldapadmin
```

Instalamos otro paquete necesario:

```
#aptitude install php5
```

Después de las instalaciones reiniciamos los siguientes servicios:

```
#!/etc/init.d/slapt restart
```

```
#!/etc/init.d/apache2 restart
```



Accedemos a la interfaz web con “<http://localhost/phpldapadmin>” Nos logueamos:



¿Qué es FreeRadius?

Es un protocolo de autenticación y autorización para aplicaciones de acceso a la red o movilidad IP. Utiliza el puerto 1812 UDP para establecer sus conexiones.

Cuando se realiza la conexión con un ISP mediante módem, DSL, cablemódem, Ethernet o Wi-Fi, se envía una información que generalmente es un nombre de usuario y una contraseña. Esta información se transfiere a un dispositivo Network Access Server (NAS) sobre el protocolo PPP, quien redirige la petición a un servidor RADIUS sobre el protocolo RADIUS. El servidor RADIUS comprueba que la información es correcta utilizando esquemas de autenticación como PAP, CHAP o EAP. Si es aceptado, el servidor autorizará el acceso al sistema del ISP y le asigna los recursos de red como una dirección IP, y otros parámetros como L2TP, etc.

Una de las características más importantes del protocolo RADIUS es su capacidad de manejar sesiones, notificando cuando comienza y termina una conexión, así que al usuario se le podrá determinar su consumo y facturar en consecuencia; los datos se pueden utilizar con propósitos estadísticos.

RADIUS fue desarrollado originalmente por Livingston Enterprises para la serie PortMaster de sus Servidores de Acceso a la Red(NAS), más tarde se publicó como RFC 2138 y RFC 2139. Actualmente existen muchos servidores RADIUS, tanto comerciales como de código abierto. Las prestaciones pueden variar, pero la mayoría pueden gestionar los usuarios en archivos de texto, servidores LDAP, bases de datos varias, etc. A menudo se utiliza SNMP para monitorear remotamente el servicio. Los servidores Proxy RADIUS se utilizan para una administración centralizada y pueden reescribir paquetes RADIUS al vuelo (por razones de seguridad, o hacer conversiones entre dialectos de diferentes fabricantes)....

RADIUS es extensible; la mayoría de fabricantes de software y hardware RADIUS implementan sus propios dialectos.

Instalación de FreeRadius

Instalaremos el paquete “freeradius-ldap” ya que el nos instalará todo lo necesario tambien.

```

root@peasodebian:/home/usuario# aptitude search freeradius-ldap
p   freeradius-ldap - LDAP module for FreeRADIUS server
root@peasodebian:/home/usuario# aptitude install freeradius-ldap
Se instalarán los siguiente paquetes NUEVOS:
  freeradius{a} freeradius-common{a} freeradius-ldap freeradius-utils{a}
  libdbi-perl{a} libfreeradius2{a} libnet-daemon-perl{a} libplrpc-perl{a}
0 paquetes actualizados, 8 nuevos instalados, 0 para eliminar y 0 sin actualizar
.
Necesito descargar 2.241 kB de ficheros. Después de desempaquetar se usarán 5.87
7 kB.
¿Quiere continuar? [Y/n/?] █

```

Consultamos los paquetes que tenemos instalados referente a FreeRadius:

```

root@peasodebian:/home/usuario# dpkg -l *freeradius*
Deseado=Desconocido/Instalar/Eliminar/Purgar/Retener
| Estado=No/Instalado/Config-files/Desempaquetado/Medio-conf/Medio-inst/espera-d
isparo/pendiente-disparo
|/ Err?=(ninguno)/Requiere-reinst (Estado,Err: mayúsc.=malo)
||/ Nombre          Versión          Arquitectura Descripción
+++-----
ii  freeradius         2.1.12+dfsg-   i386          high-performance and highly confi
ii  freeradius-com     2.1.12+dfsg-   all          FreeRADIUS common files
un  freeradius-krb     <ninguna>      (no hay ninguna descripción disp
ii  freeradius-lda     2.1.12+dfsg-   i386          LDAP module for FreeRADIUS server
un  freeradius-mys     <ninguna>      (no hay ninguna descripción disp
un  freeradius-pos     <ninguna>      (no hay ninguna descripción disp
ii  freeradius-uti     2.1.12+dfsg-   i386          FreeRADIUS client utilities
ii  libfreeradius2     2.1.12+dfsg-   i386          FreeRADIUS shared library
root@peasodebian:/home/usuario# █

```


Habilitar autenticación LDAP en FreeRadius

Modificamos el fichero “/etc/freeradius/modules/ldap”.

En el fichero cambiaremos lo siguiente:

```
ldap {
    server = "127.0.0.1"
    identity = "cn=admin,dc=servidorldap,dc=azeroth,dc=com"
    password = momaso
    basedn = "dc=servidorldap,dc=azeroth,dc=com"
    filter = "(uid=%{%{Stripped-User-Name}:-%{User-Name}})"
    #base_filter = "(objectclass=radiusprofile)"
```

Explicamos que es cada campo:

- server= Nombre del servidor LDAP, o en su defecto IP.
- identity= Usuario con privilegios en el ldap y dominio de búsqueda.
- password= Password de este usuario
(es posible suprimir estos dos campos)
- basedn= le decimos donde buscar en el ldap

Editamos el fichero “/etc/freeradius/sites-enabled/default” :

```
authorize {
    #
    # The ldap module will set Auth-Type to LDAP if it has not
    # already been set
    ldap
    ....
}

authenticate {
    # Uncomment it if you want to use ldap for authentication
    #
    # Note that this means "check plain-text password against
    # the ldap database", which means that EAP won't work,
    # as it does not supply a plain-text password.
    Auth-Type LDAP {
        ldap
    }
```

Tenemos que configurar una “shared key” que usara el router/AP para autenticarse con el servidor FreeRadius. Editamos el fichero “/etc/freeradius/clients.conf”:

```
GNU nano 2.2.6 Fichero: clients.conf

secret = gonzalocarmona
```

Debian Wheezy Stable + OpenLDAP + FreeRadius

Reiniciamos el servicio:

```
root@peasodebian:/etc/freeradius/sites-enabled# /etc/init.d/freeradius re
start
[ ok ] Stopping FreeRADIUS daemon: freeradius.
[ ok ] Starting FreeRADIUS daemon: freeradius.
root@peasodebian:/etc/freeradius/sites-enabled#
```

Usaremos la herramienta “radtest” para comprobar el funcionamiento:

```
root@peasodebian:/etc/freeradius/sites-enabled# radtest usuario usuario 127.0.0.1 1812 testing123
Sending Access-Request of id 176 to 127.0.0.1 port 1812
  User-Name = "usuario"
  User-Password = "usuario"
  NAS-IP-Address = 127.0.1.1
  NAS-Port = 1812
  Message-Authenticator = 0x00000000000000000000000000000000
rad_recv: Access-Accept packet from host 127.0.0.1 port 1812, id=176, length=20
root@peasodebian:/etc/freeradius/sites-enabled#
```

En esta captura anterior vemos que acepta la conexión con un usuario local.

Probamos con un usuario del LDAP:



```
root@peasodebian:/home/usuario# radtest pruebas usuario 127.0.0.1 1812 testing123
Sending Access-Request of id 60 to 127.0.0.1 port 1812
  User-Name = "pruebas"
  User-Password = "usuario"
  NAS-IP-Address = 127.0.1.1
  NAS-Port = 1812
  Message-Authenticator = 0x00000000000000000000000000000000
rad_recv: Access-Accept packet from host 127.0.0.1 port 1812, id=60, length=20
root@peasodebian:/home/usuario#
```

Demostramos que el usuario es SOLO de LDAP y no un usuario del sistema:

```
root@peasodebian:/home/usuario# cat /etc/shadow | grep -i usuari*
usuario:$6$MopRLPFi$cd7wR8.JBGCviBV2/dvbpBCKo9V8ch3Ngcj0CrGbjhrU2lV24/UCPmXXiMUvtPE5nD/BL83t9of/
VG81JTFCH.:15847:0:99999:7:::
root@peasodebian:/home/usuario# cat /etc/shadow | grep -i prueba*
root@peasodebian:/home/usuario#
```

Configuración del punto de acceso (AP) para usar Radius

Accedemos al menú de configuración del AP.

En la sección “Wireless” nos encontramos (en este caso es un Linksys con firmware modificado DD-WRT):

The screenshot shows the DD-WRT web interface for configuring the wireless interface w10. The top navigation bar includes Setup, Wireless (selected), Services, Security, Access Restrictions, NAT / QoS, Administration, and Status. Below this, the 'Wireless' section has sub-tabs: Basic Settings, Radius, Wireless Security, MAC Filter, Advanced Settings, and WDS. The 'Basic Settings' tab is active, showing the 'Physical Interface w10 - SSID [Sputnik-Managed] HWAddr [00:22:6B:61:34:75]'. The configuration fields are as follows:

- Wireless Mode: AP (dropdown)
- Wireless Network Mode: Mixed (dropdown)
- Wireless Network Name (SSID): PruebasRadius (text input)
- Wireless Channel: Auto (dropdown)
- Wireless SSID Broadcast: ☒ Enable ☐ Disable
- Sensitivity Range (ACK Timing): 2000 (text input, Default: 2000 meters)
- Network Configuration: ☐ Unbridged ☒ Bridged

On the right side, there is a 'Help' section with a 'more...' link. The 'Wireless Network Mode' section explains that if you wish to exclude Wireless-G clients, you should choose 'B-Only' mode. A note states that when changing wireless mode, some advanced parameters are susceptible to modification ('Afterburner', 'Basic Rate' or 'Frame Burst'). The 'Sensitivity Range' section explains that it adjusts the ack timing, with 0 disabling ack timing completely for Broadcom firmwares, while on Atheros-based firmwares it turns into auto ack timing mode.

At the bottom, there is an 'Add' button for virtual interfaces and a row of three buttons: Save, Apply Settings, and Cancel Changes.

Hemos configurado:

- Wireless Mode: AP (trabaja como un punto de acceso).
- SSID: “PruebasRadius” (Es un simple nombre, valdría “cualquiera”).

Ahora nos dirigimos a la pestaña “Wireless Security”:

The screenshot shows the Mikrotik WinBox interface with the 'Wireless Security' tab selected for the 'wlan0' interface. The configuration is as follows:

- Physical Interface:** wlan0 SSID [Sputnik-Managed] HWAddr [00:22:6B:61:34:75]
- Security Mode:** RADIUS (selected from a dropdown)
- MAC Format:** aabbcc-ddeeff (selected from a dropdown)
- Radius Auth Server Address:** 192 . 168 . 180 . 162
- Radius Auth Server Port:** 1812 (Default: 1812)
- Radius Auth Shared Secret:** gonzalocarmona (with an 'Unmask' checkbox checked)

Buttons at the bottom: Save, Apply Settings. A 'Help' link is also present.

Hemos modificado:

- Security Mode: RADIUS.
- Radius Auth Server Address: 192.168.180.162 (es la IP que tiene actualmente el servidor radius, cuando todo se compruebe es necesario poner las IP estáticamente).
- Radius Auth Shared Secret: gonzalocarmona (es la shared key que usarán el AP y el servidor radius para autenticarse).

Vamos a intentar conectarnos, en este caso desde un dispositivo móvil:

The left screenshot shows the Android 'Wi-Fi' settings. The 'Wi-Fi' toggle is turned on. Under 'Seleccione una red...', there are two networks: 'JAZZTEL_1CAD' (checked with a blue checkmark) and 'PruebaRadius'. Below this, the 'Preguntar al conectar' toggle is turned off. A note at the bottom states: 'Se accederá automáticamente a las redes conocidas. Si no hay ninguna red conocida disponible, deberá seleccionar una manualmente.'

The right screenshot shows the login screen for the 'PruebaRadius' network. The title is 'Introduzca la contraseña de "PruebaRadius"'. There are 'Cancelar' and 'Conectarse' buttons. The 'Nombre de usuario' field contains 'pruebau' and the 'Contraseña' field is masked with dots. A QWERTY keyboard is visible at the bottom.

Aquí vemos el certificado generado por el servidor:



SIN VERIFICAR significa que este certificado no está emitido por una entidad de certificación autorizada. Este certificado se crea automáticamente durante la instalación de freeradius.

Lo aceptamos para iniciar sesión:



Según podemos comprobar con el comando “radtest” desde el equipo si podemos autenticarnos con un usuario del ldap vía freeradius:

```
root@peasodebian:/etc/freeradius# radtest usuario usuario localhost 1812 testing321
Sending Access-Request of id 55 to 127.0.0.1 port 1812
  User-Name = "usuario"
  User-Password = "usuario"
  NAS-IP-Address = 127.0.1.1
  NAS-Port = 1812
  Message-Authenticator = 0x00000000000000000000000000000000
rad_recv: Access-Accept packet from host 127.0.0.1 port 1812, id=55, length=20
root@peasodebian:/etc/freeradius#
```

Pero a la hora de hacerlo por conexión Wifi da error de credenciales, y vemos en el modo “debug” de freeradius que el paquete es descartado.

```
rad_recv: Access-Request packet from host 192.168.180.1 port 3107, id=50, length=123
  User-Name = "usuario"
  NAS-IP-Address = 192.168.180.1
  NAS-Port = 0
  Called-Station-Id = "00-22-F7-22-B1-78"
  Calling-Station-Id = "14-5A-05-7D-59-23"
  Framed-MTU = 1400
  NAS-Port-Type = Wireless-802.11
  EAP-Message = 0x0201000c0175737561726966
  Message-Authenticator = 0x100167c7de998c0c05d6bd4e07c9170e
# Executing section authorize from file /etc/freeradius/sites-enabled/default
+- entering group authorize {...}
++- entering policy filter_username {...}
+++? if (User-Name =~ /^ /)
? Evaluating (User-Name =~ /^ /) -> FALSE
+++? if (User-Name =~ /^ /) -> FALSE
+++? if (User-Name =~ / $$/)
? Evaluating (User-Name =~ / $$/) -> FALSE
+++? if (User-Name =~ / $$/) -> FALSE
+++? if (User-Name != "%{tolower:%{User-Name}}")
  expand: %{User-Name} -> usuario
  expand: %{tolower:%{User-Name}} -> usuario
? Evaluating (User-Name != "%{tolower:%{User-Name}}") -> FALSE
```



```
[ldap] performing user authorization for usuario
[ldap] expand: %{Stripped-User-Name} ->
[ldap] ... expanding second conditional
[ldap] expand: %{User-Name} -> usuario
[ldap] expand: (uid=%{%{Stripped-User-Name}:-%{User-Name}}) -> (uid=usuario)
[ldap] expand: dc=azeroth,dc=com -> dc=azeroth,dc=com
[ldap] ldap_get_conn: Checking Id: 0
[ldap] ldap_get_conn: Got Id: 0
[ldap] attempting LDAP reconnection
[ldap] (re)connect to 192.168.180.101:389, authentication 0
[ldap] setting TLS Require Cert to never
[ldap] bind as cn=admin,dc=azeroth,dc=com/momaso to 192.168.180.101:389
[ldap] waiting for bind result ...
[ldap] Bind was successful
[ldap] performing search in dc=azeroth,dc=com, with filter (uid=usuario)
[ldap] No default NMAS login sequence
[ldap] looking for check items in directory...
[ldap] userPassword -> Password-With-Header == "usuario"
[ldap] looking for reply items in directory...
[ldap] user usuario authorized to use remote access
[ldap] ldap_release_conn: Release Id: 0
++[ldap] returns ok
```

```
++[expiration] returns noop
++[logintime] returns noop
ERROR: No authenticate method (Auth-Type) found for the request: Rejecting the u
ser
Failed to authenticate the user.
Login incorrect: [usuario/<no User-Password attribute>] (from client 192.168.180
.1 port 0 cli 14-5A-05-7D-59-23)
Using Post-Auth-Type Reject
# Executing group from file /etc/freeradius/sites-enabled/default
+- entering group REJECT {...}
[attr_filter.access_reject] expand: %{User-Name} -> usuario
attr_filter: Matched entry DEFAULT at line 11
++[attr_filter.access_reject] returns updated
Delaying reject of request 2 for 1 seconds
Going to the next request
Waking up in 0.9 seconds.
Sending delayed reject for request 2
Sending Access-Reject of id 52 to 192.168.180.1 port 3107
Waking up in 4.9 seconds.
```

Solucionando el problema, configurando PEAP

“PEAP” es un protocolo que encapsula al “Protocolo Extensible de Autenticación” (EAP) dentro de una “capa” de seguridad cifrada y autenticada de transporte (TLS)

“PEAP” se creó con el objetivo de corregir las carencias de “EAP”. “EAP” asumió un canal de comunicación protegido, como esto proporcionaba una seguridad física, no se necesitó ninguna instalación adicional para la protección de “EAP”.

En este proyecto se ha utilizado PEAP por los siguientes motivos:

- Es seguro.
- Fácil de implementar.
- No necesita instalaciones adicionales.

PEAP no está trabajando solo, aquí se convina con EAP-MSCHAP, por lo que podríamos en permitir la autenticación con bases de datos como Microsoft NT y Microsoft Active Directory.

Vamos aplicarlo, arreglando la conectividad.

Editamos el fichero `/etc/freeradius/sites-enable/default`

- En `Authorize {}` dejamos descomentado:
 - `preprocess`
 - `auth_log`
 - `chap`
 - `mschap`
 - `suffix`
 - `eap { ok=return }`
 - `ldap`
 - `expiration`
 - `logintime`
 - `pap`
- `authenticate {}`
 - `Auth-Type MS-CHAP { mschap }`
 - `Auth-Type LDAP { ldap }`
 - `eap`
- `preacct {}`
 - `preprocess`
 - `acct_unique`
 - `suffix`
 - `IPASS`
 - `nrdomain`
 - `files`
- `accounting {}`
 - `detail`
 - `radutmp`
 - `exec`
 - `attr_filter.accounting_response`
- `session {}`
 - `radutmp`
- `post-auth {}`

- main_poll
- ldap
- exec

El resto lo dejamos por defecto

Editamos el fichero /etc/freeradius/sites-enable/default

- server inner-tunnel {
 - listen {
 - ipaddr = 127.0.0.1
 - port = 18120
 - type = auth
 - }
 - authorize {
 - mschap
 - suffix
 - update control { Proxy-To-Realm := LOCAL }
 - eap { ok = return }
 - ldap
 - expiration
 - logintime
 - pap
 - authenticate {
 - Auth-Type PAP { pap }
 - Auth-Type CHAP { chap }
 - Auth-Type MS-CHAP { mschap }
 - unix
 - Auth-Type LDAP { ldap }
 - eap
 - }
 - session {
 - radutmp
 - }
 - post-auth {
 - ldap
 - Post-Auth-Type REJECT { attr_filter.access_reject }

Lo demas lo dejamos por defecto.

Editamos el fichero “/etc/freeradius/eap.conf

- eap{
 - default_eap_type = mschapv2
 - timer_expire = 60
 - ignore_unknown_eap_types = no
 - cisco_accounting_username_bug = no
 - max_sessions = 4096
 - md5{}
 - leap{}
 - gtc{ auth_type = PAP }
 - 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"
-
-
- }

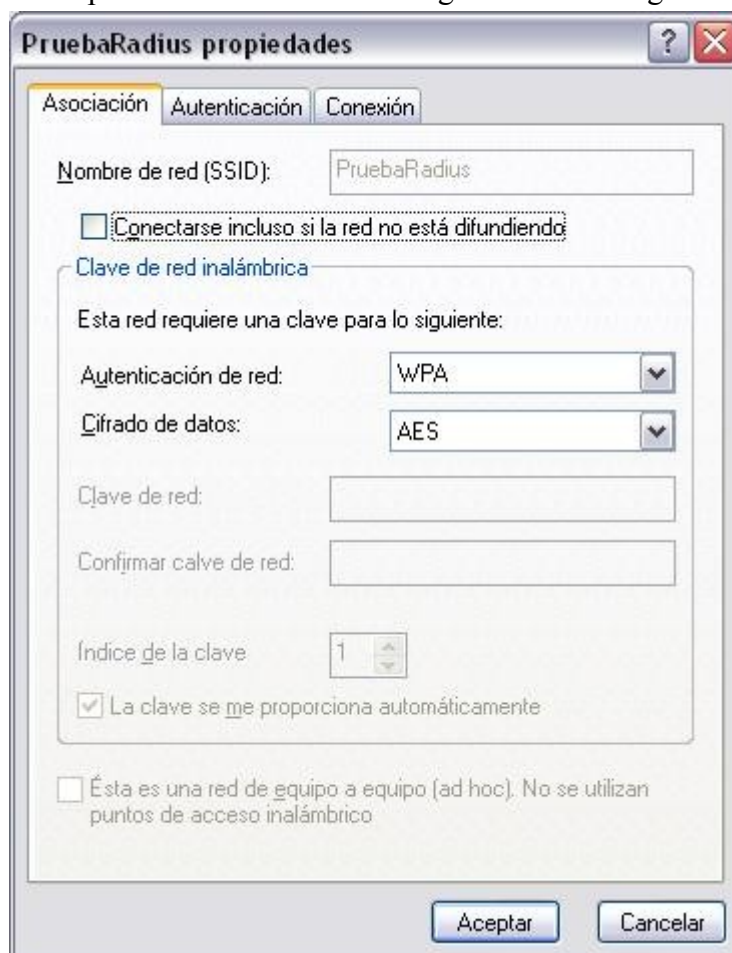
El resto lo dejamos por defecto.

Probamos que funciona logueandonos en esta ocasión desde un Windows XP:

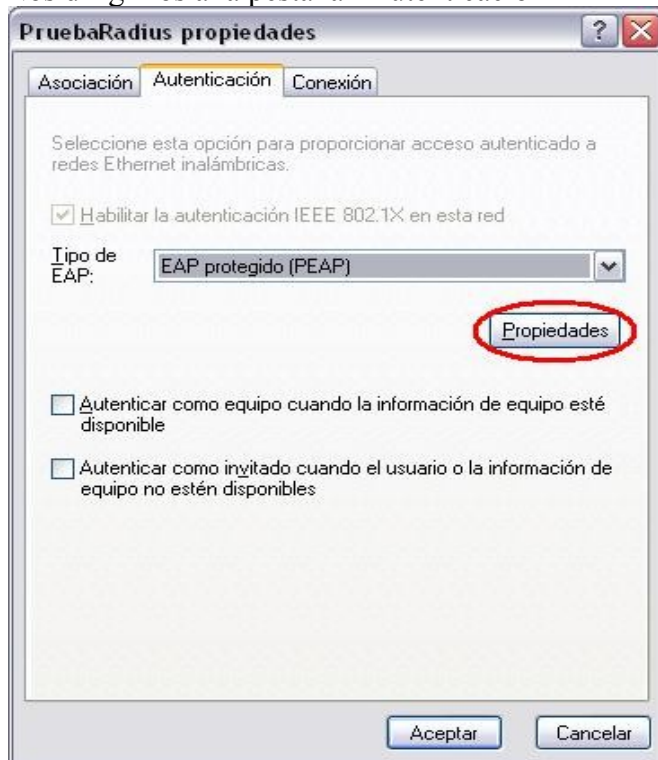
- Buscamos la red “PruebaRadius” en este caso. Accedemos a las propiedades de la conexión.



- En la pestaña “Asociación” configuramos de la siguiente forma:



- Nos dirigimos a la pestaña “Autenticación”



Seleccionamos EAP (PEAP) ya que así hemos configurado el server.

- En la ventana anterior pinchamos en “Propiedades”:



En “Metodo de autenticación” seleccionamos “EAP-MSCHAP v2).

- Pinchamos en “Aceptar” hasta salir de todas las ventanas, se conectará a la red y probamos a ver si tenemos “ping”.

```

Adaptador Ethernet Conexión de área local      :
        Estado de los medios. . . .: medios desconectados
Adaptador Ethernet Conexiones de red inalámbricas      :
        Sufijo de conexión específica DNS :
        Dirección IP. . . . . : 192.168.180.101
        Máscara de subred . . . . . : 255.255.255.0
        Puerta de enlace predeterminada : 192.168.180.1

```

La prueba la he hecho con FreeRadius en modo “debug” (#freeradius -X, con el servicio parado) para poder ver detalladamente los pasos que se realizan para la conexión.

- Primero vemos como el cliente manda la petición:

```

rad_recv: Access-Request packet from host 192.168.180.1 port 3077, i
length=166
        User-Name = "prueba"
        NAS-IP-Address = 192.168.180.1
        NAS-Port = 0
        Called-Station-Id = "00-22-F7-22-B1-78"
        Calling-Station-Id = "00-25-D3-59-F2-02"
        Framed-MTU = 1400
        NAS-Port-Type = Wireless-802.11
        EAP-Message = 0x020900261900170301001bb185e2448f4c0eb644ead3
c039509d55b080a2370471519a
        State = 0x4292c886459bd1da8febf189f9255e2c
        Message-Authenticator = 0x79101f9ec4b696dfb776929317a0750e

```

- Entramos en la fase de “Autorización”:

```

# Executing section authorize from file /etc/freeradius/sites-enabled/inne
r-tunnel
+- entering group authorize {...}
++[mschap] returns noop
[suffix] No '@' in User-Name = "prueba", looking up realm NULL
[suffix] No such realm "NULL"
++[suffix] returns noop
++[control] returns noop
[eap] EAP packet type response id 8 length 6
[eap] No EAP Start, assuming it's an on-going EAP conversation
++[eap] returns updated
[ldap] performing user authorization for prueba
[ldap] expand: %{Stripped-User-Name} ->
[ldap] ... expanding second conditional
[ldap] expand: %{User-Name} -> prueba
[ldap] expand: (uid=%{%{Stripped-User-Name}:%{User-Name}}) -> (uid=prueb
a)
[ldap] expand: ou=People,dc=servidorldap,dc=azeroth,dc=com -> ou=People,d
c=servidorldap,dc=azeroth,dc=com
[ldap] ldap_get_conn: Checking Id: 0
[ldap] ldap_get_conn: Got Id: 0
[ldap] performing search in ou=People,dc=servidorldap,dc=azeroth,dc=com,
with filter (uid=prueba)

```

- Seguimos con la fase de “Autenticación”:

```
[ldap] ldap_release_conn: Release Id: 0
++[ldap] returns ok
++[expiration] returns noop
++[logintime] returns noop
[pap] WARNING: Auth-Type already set. Not setting to PAP
++[pap] returns noop
Found Auth-Type = EAP
# Executing group from file /etc/freeradius/sites-enabled/inner-tunnel
+- entering group authenticate {...}
[eap] Request found, released from the list
[eap] EAP/mschapv2
[eap] processing type mschapv2
[eap] Freeing handler
++[eap] returns ok
Login OK: [prueba] (from client radping port 0 via TLS tunnel)
```

- Procesos “post autenticación”:

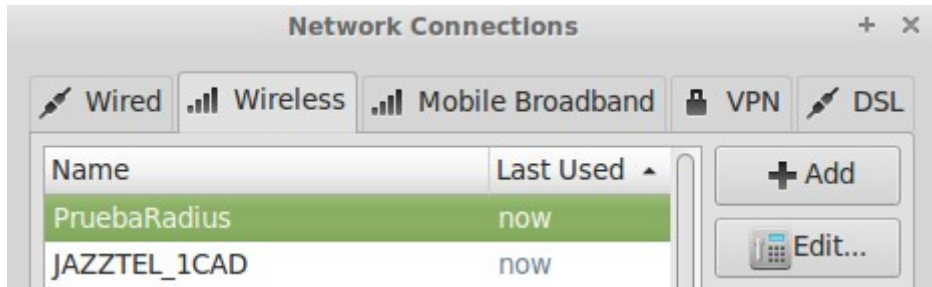
```
# Executing section post-auth from file /etc/freeradius/sites-enabled/inner-tunnel
+- entering group post-auth {...}
++[ldap] returns noop
} # server inner-tunnel
[peap] Got tunneled reply code 2
      MS-MPPE-Encryption-Policy = 0x00000001
      MS-MPPE-Encryption-Types = 0x00000006
      MS-MPPE-Send-Key = 0xa03b9af53ddceb31a93f097df77410f5
      MS-MPPE-Recv-Key = 0x59aaae6758ed2cb8744a41d27c260896
      EAP-Message = 0x03080004
      Message-Authenticator = 0x00000000000000000000000000000000
      User-Name = "prueba"
[peap] Got tunneled reply RADIUS code 2
      MS-MPPE-Encryption-Policy = 0x00000001
      MS-MPPE-Encryption-Types = 0x00000006
      MS-MPPE-Send-Key = 0xa03b9af53ddceb31a93f097df77410f5
      MS-MPPE-Recv-Key = 0x59aaae6758ed2cb8744a41d27c260896
      EAP-Message = 0x03080004
      Message-Authenticator = 0x00000000000000000000000000000000
      User-Name = "prueba"
[peap] Tunneled authentication was successful.
[peap] SUCCESS
```

Como vemos la autenticación fué satisfactoria.

- Devolvemos la respuesta al cliente:

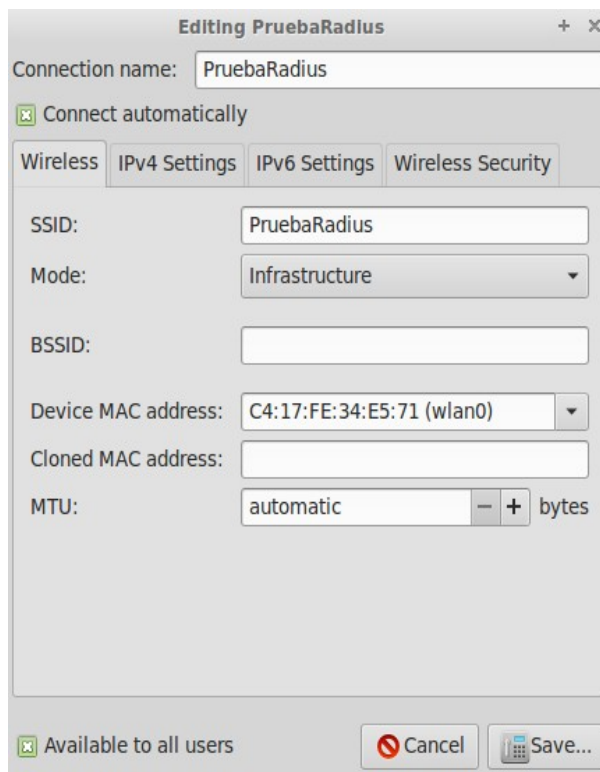
```
Sending Access-Challenge of id 170 to 192.168.180.1 port 3077
      EAP-Message = 0x010900261900170301001bc0c746cd65d53fc84e32dd83ccf9
fa397c5f681205a347d22d1f3c
      Message-Authenticator = 0x00000000000000000000000000000000
      State = 0x4292c886459bd1da8feb189f9255e2c
Finished request 40.
Going to the next request
Waking up in 4.8 seconds.
```

Conectamos desde un cliente Linux.



Edito la conexión para que veamos como la tengo configurada.

- Wireless:



- Wireless Security:



He probado que funciona con otro usuario que he creado para asegurarme que está todo correcto:

```
Sending Access-Accept of id 4 to 192.168.180.1 port 3077
  MS-MPPE-Recv-Key = 0xa65948d504ada32d35d9ced61f2d8a0c265426b3683534fc7
a8cb428c3c7c5e4
  MS-MPPE-Send-Key = 0x5af32747280792b029258eb2150422bdd6cd9928ccb7168b1
b4c2fbff80a2237
  EAP-Message = 0x030a0004
  Message-Authenticator = 0x00000000000000000000000000000000
  User-Name = "gonzalo"
Finished request 130.
Going to the next request
Waking up in 4.8 seconds.
Cleaning up request 121 ID 251 with timestamp +4981
```

Realizamos un “ifconfig” desde el cliente y nos devuelve:

```
eth0      Link encap:Ethernet  HWaddr 00:26:9e:f2:94:fa
          inet6 addr: fe80::226:9eff:fef2:94fa/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:993 errors:0 dropped:0 overruns:0 frame:0
          TX packets:883 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:551805 (551.8 KB)  TX bytes:93563 (93.5 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2742 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2742 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:293413 (293.4 KB)  TX bytes:293413 (293.4 KB)

wlan0     Link encap:Ethernet  HWaddr c4:17:fe:34:e5:71
          inet addr:192.168.180.102  Bcast:192.168.180.255  Mask:255.255.255.0
          inet6 addr: fe80::c617:feff:fe34:e571/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:22746 errors:0 dropped:0 overruns:0 frame:0
          TX packets:20790 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:20225152 (20.2 MB)  TX bytes:4183546 (4.1 MB)
```

Vemos que está en el segmento de red 192.168.180.0/24, por lo que confirma que estoy conectado a la red inalámbrica.

Tenía la intención de probar de nuevo el funcionamiento en el iPhone pero al que tengo acceso está actualizado a la Beta 1 de iOS7 y tiene algún bug por el cual al conectar y recibir el certificado del servidor el botón de aceptarlo no realiza ningún proceso.

Esto me ha ocupado tiempo mirando que no fuese problema de la configuración del servidor...

General Información	
Canciones	0
Videos	10
Fotos	614
Aplicaciones	8
Capacidad	13,6 GB
Disponible	10,4 GB
Versión	7.0 (11A4372q)
Operador	Carrier 14.5
Modelo	MC604Y/A

Lo probamos en otro dispositivo, en este caso un iPad (3ª generación, “New iPad”), iOS 6.1.1

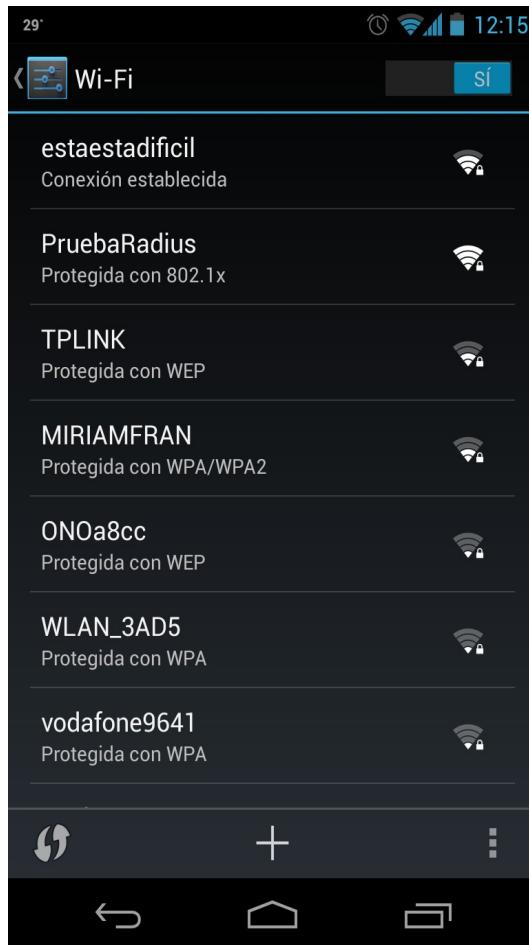


Debian Wheezy Stable + OpenLDAP + FreeRadius



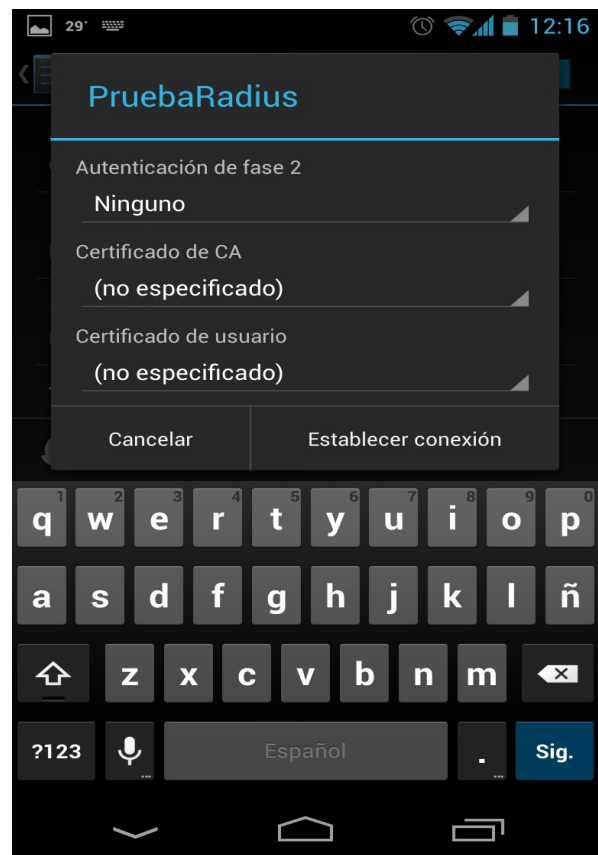
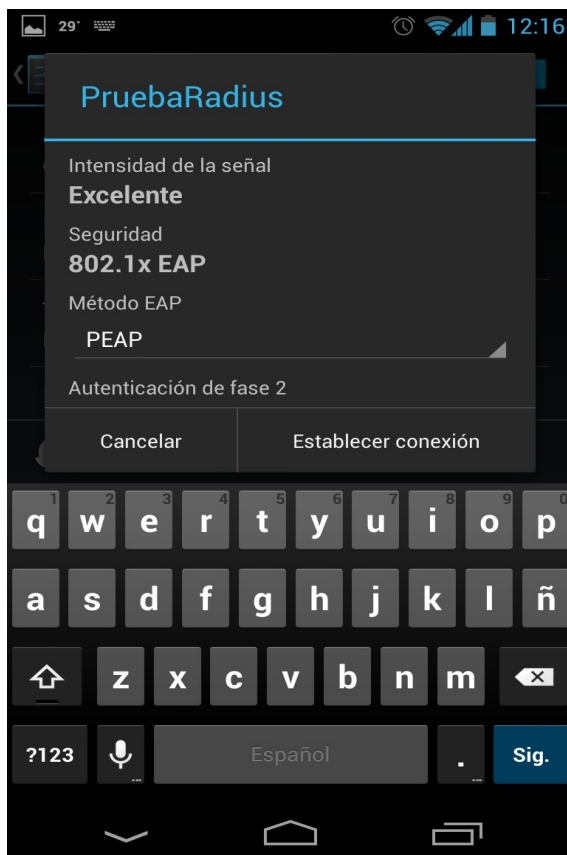
Ahora toca un dispositivo Android, en mi caso un Nexu 4, fabricado por LG

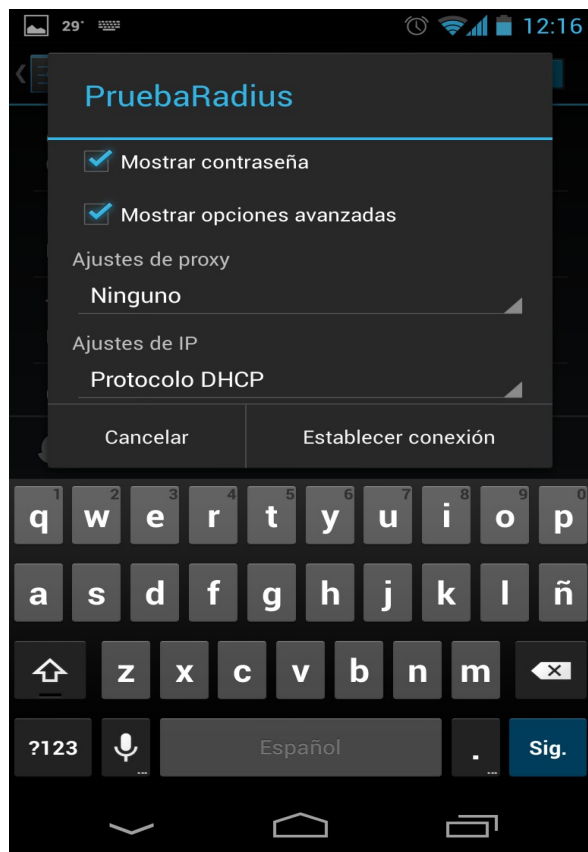
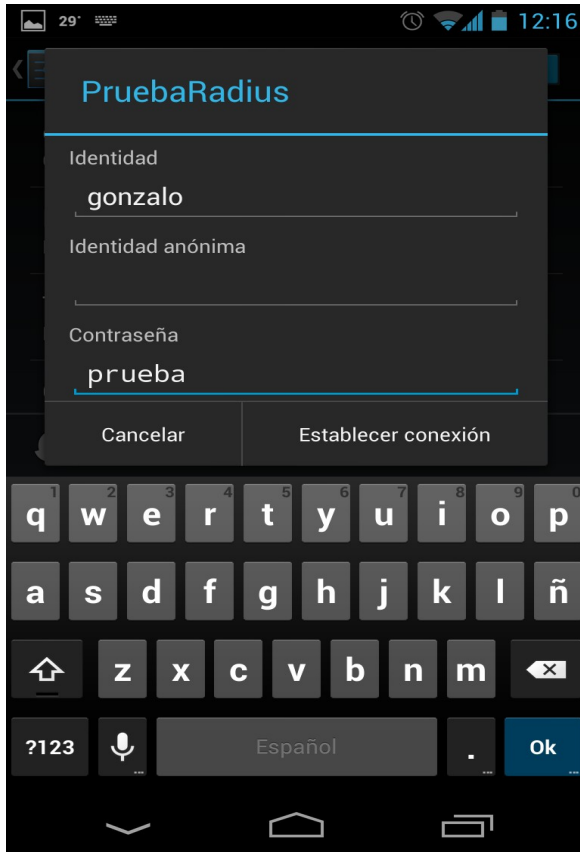
- Nos dirigimos a la configuración de redes inalámbricas:



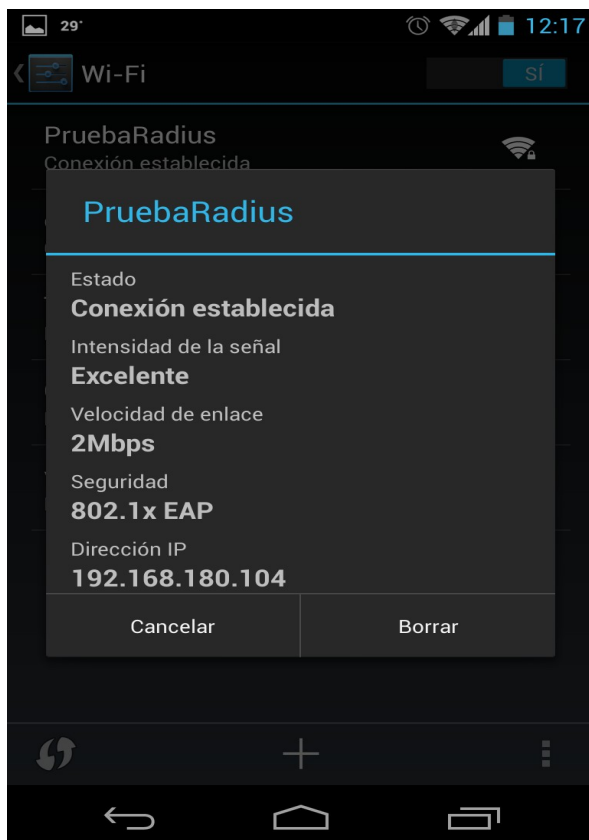
Seleccionamos “PruebaRadius” como venimos haciendo siempre.

- Vemos los detalles de la conexión:





Conectamos a la red:



Configuración del cliente en Windows7

- Editamos las propiedades de la conexión Wifi “PruebaRadius”

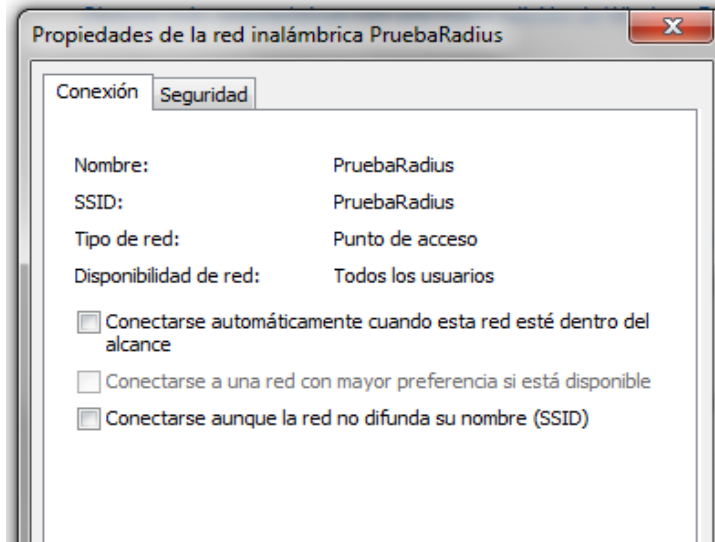
Ver información básica acerca del equipo

Edición de Windows

Windows 7 Home Premium

Copyright © 2009 Microsoft Corporation. Reservados todos los derechos.

Service Pack 1



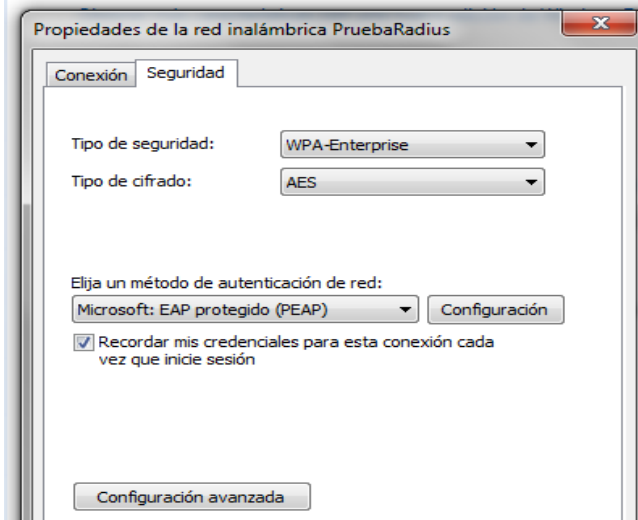
- En la pestaña “Seguridad”:

Edición de Windows

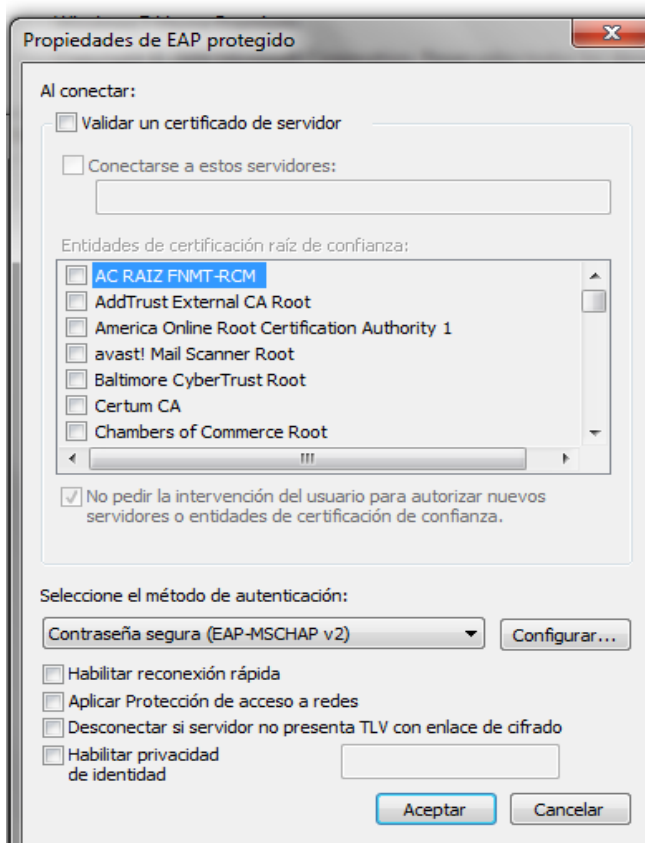
Windows 7 Home Premium

Copyright © 2009 Microsoft Corporation. Reservados todos los derechos.

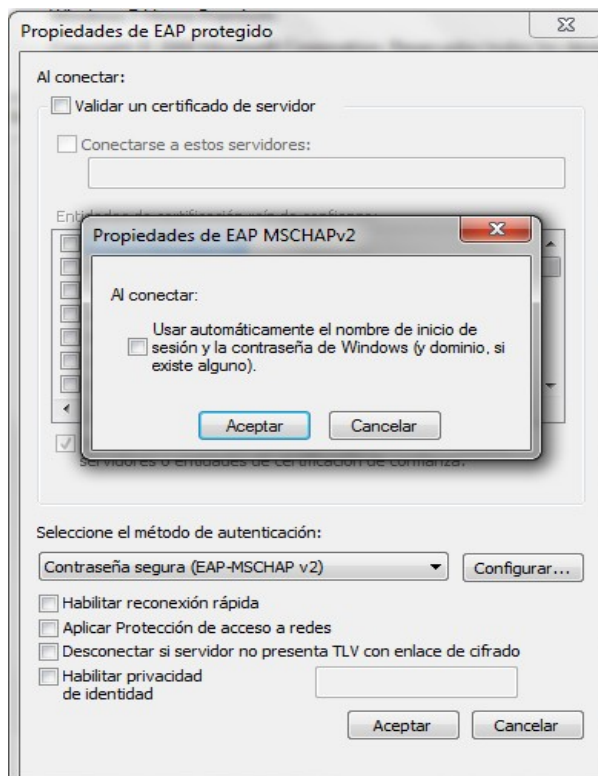
Service Pack 1



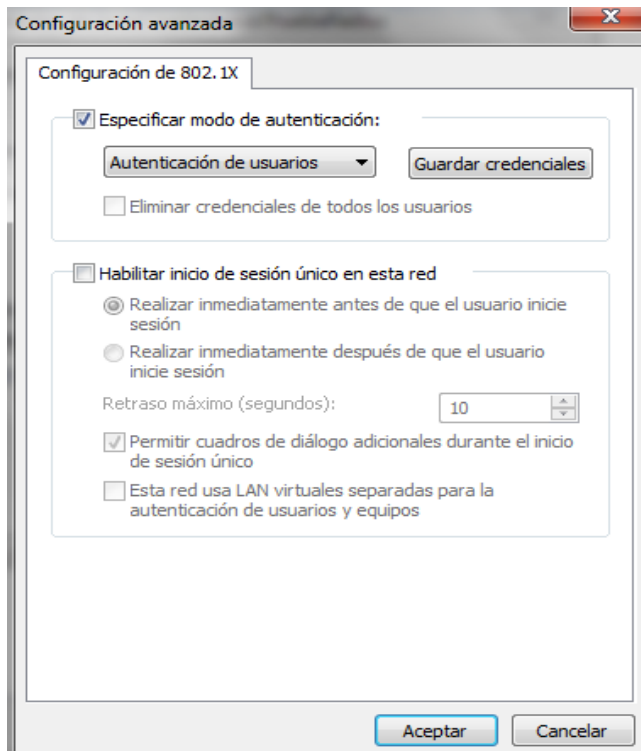
- Pinchamos en “Configuración” y editamos la configuración para que quede así:



- Pinchamos en “Configurar”:



- Salimos de las ventanas hasta la que contenía la pestaña “Seguridad” y pinchamos en “Configuración avanzada” y lo dejamos como aparece en la captura:




Conectamos:

[Ver las redes activas](#)

[Conectar o desconectar](#)



PruebaRadius
Red pública

Tipo de acceso: Sin acceso a Internet
Conexiones:  Conexión de red inalámbrica (PruebaRadius)

Dispositivos probados que no funcionan con Radius

- PlayStation Vita (Modelo Wifi&3G).
- Nintendo 3DS

Problemas ocurridos durante el proyecto

- Flash corrupted Linksys WRT54GL v1.1

El router/AP poseía un firmware NO oficial, en concreto un dd-wrt chillispot.

En principio por un mal flasheo e instalación del firmware durante su funcionamiento quedó “bricked”, de tal forma que no podía hacerle “ping”, conectarme por “tftp” o recuperarlo puenteando la patilla 16 de la flash a tierra.

Para recuperarlo monté un cable jtag.

Componentes necesarios:

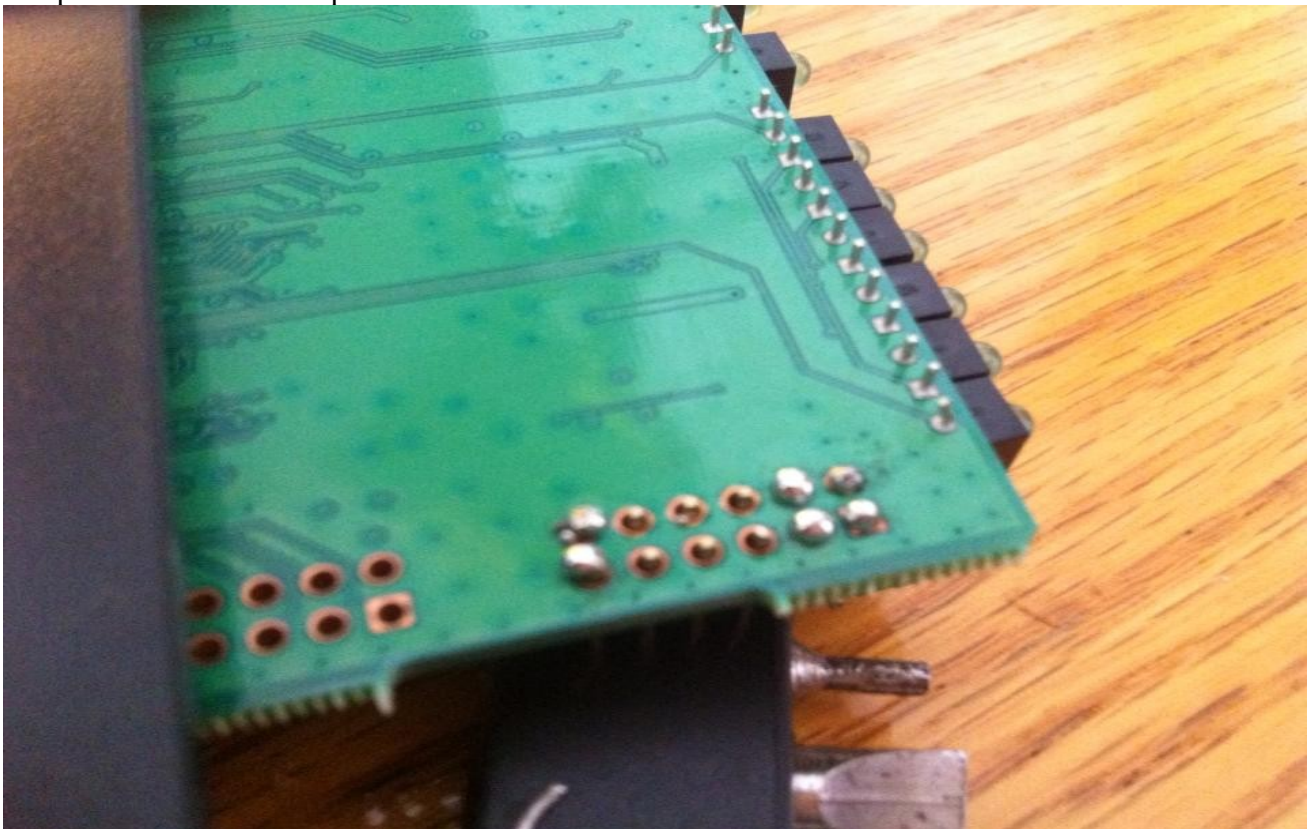
- 4 resistencias del 100 Ohm.
- 1 conector DB25 macho.
- 1 conector de 12 pines para el puerto jtag.
- Cable plano.

Colocamos los pines como vemos en la parte superior derecha.





Aquí vemos la parte inferior, donde se soldarán los pines.
Empezamos a soldar los pines.



Conectamos el cable al puerto jtag y al pc.



Descargamos el Tjtag v3.

Guardamos el “giveio.sys” en System32/drivers.

Abrimos el “loaddrv.exe”. Le indicamos la ruta al “giveio.sys”, pinchamos en “install”, reiniciamos el equipo, volvemos abrir “loaddrv.exe” e iniciamos el servicio.

Copiamos el .cfe en el firectorio donde está el “tjtag”.

Ahora abrimos “simbolos del sistema” y nos dirigimos a la carpeta donde está el “Tjtag”.

Renombramos el ejecutame para que resulte más facil, en mi caso a tjtagv3.

Ejecutamos los siguientes comandos:

- tjtagv3 -erase:cfe
- tjtagv3 -erase:nvram
- tjtagv3 -erase:kernel
- tjtagv3 -flash:cfe

Cuando termine el proceso:

```

C:\WINDOWS\system32\cmd.exe
[ 99% Flashed] 1fc3ff40: ffffffff ffffffff ffffffff ffffffff
[ 99% Flashed] 1fc3ff50: ffffffff ffffffff ffffffff ffffffff
[ 99% Flashed] 1fc3ff60: ffffffff ffffffff ffffffff ffffffff
[ 99% Flashed] 1fc3ff70: ffffffff ffffffff ffffffff ffffffff
[ 99% Flashed] 1fc3ff80: ffffffff ffffffff ffffffff ffffffff
[ 99% Flashed] 1fc3ff90: ffffffff ffffffff ffffffff ffffffff
[ 99% Flashed] 1fc3ffa0: ffffffff ffffffff ffffffff ffffffff
[ 99% Flashed] 1fc3ffb0: ffffffff ffffffff ffffffff ffffffff
[ 99% Flashed] 1fc3ffc0: ffffffff ffffffff ffffffff ffffffff
[ 99% Flashed] 1fc3ffd0: ffffffff ffffffff ffffffff ffffffff
[ 99% Flashed] 1fc3ffe0: ffffffff ffffffff ffffffff ffffffff
[ 99% Flashed] 1fc3fff0: ffffffff ffffffff ffffffff ffffffff
Done <CFE.BIN loaded into Flash Memory OK>

=====
Flashing Routine Complete
=====
elapsed time: 589 seconds

*** REQUESTED OPERATION IS COMPLETE ***

C:\Documents and Settings\Administrador\Escritorio\De-Bricking\Tjtag v3.0.1 <De-Brick Program Newer Version>\Windows 32 users XP Vista 7 etc>

```

Ahora apagamos y encendemos el router quitando el cable de alimentación.
 Configuramos nuestra interfaz de red con una IP estática (192.168.1.2) con el “gateway” 192.168.1.1.
 Probamos hacer “ping” a la “192.168.1.1”.

Descargamos el firmware de nuestro router de la web de linksys.
 Abrimos ahora “tftp” de linksys, indicamos:

- IP: 192.168.1.1
- Password: admin
- File: firmware.bin

```

C:\WINDOWS\system32\cmd.exe
Paquetes: enviados = 3, recibidos = 3, perdidos = 0
<0% perdidos>,
Tiempos aproximados de ida y vuelta en milisegundos:
Mínimo = 0ms, Máximo
Control-C
^C
C:\Documents and Settings\Administrador\Escritorio\De-Bricking\Tjtag v3.0.1 <De-Brick Program Newer Version>\Windows 32 users XP Vista 7 etc>
Configuración IP de Windows
Adaptador Ethernet Conexión
Sufijo de conexión
Dirección IP...
Máscara de subred
Puerta de enlace
Adaptador Ethernet Conexión
Estado de los medios
C:\Documents and Settings\Administrador\Escritorio\De-Bricking\Tjtag v3.0.1 <De-Brick Program Newer Version>\Windows 32 users XP Vista 7 etc>

```

Upgrade Firmware Version 1.255

Server : 192.168.1.1

Password : *****

File : C:\Documents and Settings\Administrador\Escritorio\De-Bricking\Tjtag v3.0.1 <De-Brick Program Newer Version>\Windows 32 users XP Vista 7 etc> \firmware.bin

When upgrade fails, the program will retry 99 times.

Firmware was upgraded successfully!

Retry (1/99) ...

Upgrade Close

Accedemos desde el navegador de internet a la 192.168.1.1.

LINKSYS® by Cisco Versión del Firmware: v4.30.16

Enrutador de banda ancha Wireless-G WRT54GL

Configuración

Configuración | Inalámbrica | Seguridad | Restricciones de acceso | Aplicaciones & Juegos | Administración | Estado

Configuración básica | DDNS | Clonación de direcciones MAC | Enrutamiento avanzado

Idioma
Seleccione su idioma: Español

Configuración de Internet
Tipo de conexión a Internet: Configuración automática - DHCP

Configuración opcional (necesario para algunos ISP)

Nombre del enrutador: WRT54GL

Nombre de host:

Nombre de dominio:

MTU: Automático

Tamaño: 1500

Configuración de red
IP del enrutador: Dirección IP local: 192 . 168 . 1 . 1

Configuración automática - DHCP: este valor se utiliza principalmente con operadores de cable.

Nombre de host: Introduzca el nombre de host proporcionado por su ISP.

Nombre de dominio: Introduzca el nombre de dominio proporcionado por su ISP.

Dirección IP local: Es la dirección del enrutador.

Máscara de subred: Es la máscara de subred del enrutador.

Configuración de “phpldapadmin” para reconocer el dominio

En la ventana de logueo puede que nos aparezca “cn=admin,dc=example,dc=com”.

Aunque lo cambiemos y nos logueemos con el nuestro puede aparecer “No hay base para example.com”.

Esto es un problema que puede parecer “evidente” y tal, pero que me ha dado algún que otro quebradero de cabeza.

Vamos a solucionarlo:

- Editamos el fichero “/etc/phpldapadmin/config.php
- Modificamos la línea: `$ldapservers->SetValue($i,'server','base',array('dc=example,dc=com'));` por lo que necesitamos, en mi caso quedaría `$ldapservers->SetValue($i,'server','base',array('dc=servidorldap,dc=azeroth,dc=com'));`.
- Modificamos: `$ldapservers->SetValue($i,'login','dn','cn=admin,dc=example,dc=com');` por `$ldapservers->SetValue($i,'login','dn','cn=admin,dc=servidorldap,dc=azeroth,dc=com');`.

Agradecimientos

- Profesores IES Gonzalo Nazareno, por la ayuda a realizar este proyecto.
- <http://freeradius.1045715.n5.nabble.com> Donde he podido consultar errores y que gracias a las preguntas de antiguos miembros de la Web he podido entender muchos de los problemas que he tenido.
- Wikipedia. Para conocer en que consisten ciertos aspectos técnicos.