

Diseño e implementación de un sistema de autenticación, autorización y acceso a una red inalámbrica vía FreeRADIUS y Active Directory

Jorge Luque Alcalá
Ingeniería Informática

Consultor: Helena Rifà Pous

18 de Junio de 2004

Agradecimientos

A mi esposa Elena, ella sabe por qué.

A todos aquellos que en algún momento han compartido su sabiduría conmigo.

Resumen

La Universitat Oberta de Catalunya posee una infraestructura de red compuesta por un FrontEnd con sistema operativo LINUX que gestiona los accesos externos al sistema, y una intranet plataformada con sistemas operativos Windows y controlada por máquinas con sistemas operativos Windows 2000 Server con funcionalidades de controladores de dominio.

Como consecuencia de la evolución positiva de las tecnologías de comunicaciones inalámbricas, y una vez alcanzado un nivel óptimo de seguridad y de caudal de transmisión, la UOC se plantea incorporar en su infraestructura actual un acceso inalámbrico para sus clientes.

El objetivo del presente proyecto es el estudio e implementación de una solución segura de conectividad inalámbrica a la infraestructura actual, aprovechando en la máxima medida posible todos los sistemas ya implementados sin que sufran graves cambios estructurales.

El proyecto está estructurado en cuatro fases. En la primera fase se analizan todas las tecnologías disponibles actualmente que permitan securizar una red inalámbrica. En la segunda fase se estudia e implementa un acceso inalámbrico seguro a una red gestionada por máquinas con sistema operativo Windows 2000 Server y la funcionalidad de controladoras de dominio implementada. La tercera fase se asemeja a la segunda, diferenciándose en el sistema operativo implementado en las máquinas servidoras, en este caso se corresponde con el sistema operativo LINUX.

La cuarta fase está dedicada a la integración de un sistema de acceso inalámbrico seguro en la infraestructura actual de la UOC (Frontend con LINUX e Intranet con Windows 2000 Server), siendo compartida entre las dos plataformas la gestión de autenticación y autorización de acceso al sistema de los clientes inalámbricos.

Índice de contenidos

Resumen	iii
1 Introducción	1
1.1 Justificación del proyecto	1
1.2 Objetivo del proyecto	2
1.3 Enfoque y método seguido	3
1.4 Planificación del proyecto	4
1.5 Calendario de trabajo	9
2 Seguridad en las redes inalámbricas	10
2.1 Introducción	10
2.2 IEEE 802.11	11
2.3 WEP	14
2.4 Problemas del WEP	15
2.4.1 Manejo de claves	15
2.4.2 Cifrado	16
2.4.3 Integridad	16
2.5 Sistemas de seguridad adicionales	17
2.5.1 802.1x	17
2.5.2 EAP	18
2.5.3 VPN	20
2.5.4 WPA	21
2.6 802.11i	23
2.7 RADIUS	23
2.8 Conclusión	25
3 WLAN segura con MS Windows 2000 Server	26
3.1 Active Directory	26
3.1.1 Sitio	27

3.1.2	Controlador de dominio	28
3.2	DHCP	29
3.3	Configurar Active Directory	31
3.4	Servidor de Certificados	34
3.5	IAS (<i>Internet Authentication Service</i>)	39
3.6	Configurar el punto de acceso	42
3.7	Configurar cliente inalámbrico	44
3.8	Conclusión	47
4	WLAN segura con LINUX	48
4.1	DHCP	48
4.2	DNS	49
4.3	OpenSSL	51
4.4	LDAP	54
4.4.1	DB4 - Berkeley DB	54
4.4.2	Cyrus - SASL	56
4.4.3	OpenLDAP	56
4.5	FreeRADIUS	60
4.5.1	Módulo SQL	65
4.6	Configurar el punto de acceso	66
4.7	Configurar cliente inalámbrico	67
4.8	Errores de ejecución	71
4.9	Conclusión	72
5	WLAN segura con MS Windows 2000 Server y LINUX	73
5.1	DHCP	73
5.2	DNS	74
5.3	LDAP – Active Directory	75
5.3.1	AD4Unix	75
5.3.2	Cliente LDAP	79

Índice de contenidos

iii

5.4 Cortafuegos	80
5.5 Servidor VPN-IPSec	81
5.6 Conclusión	83
6 Conclusiones	84
Bibliografía	87
Glosario de acrónimos	89
Anexos	93
Anexo 1 – Configuración DHCP en Windows	93
Anexo 2 – Configuración cliente inalámbrico con MS Windows XP	98
Anexo 3 – Archivo configuración DHCP <code>/etc/dhcpd.conf</code>	101
Anexo 4 – Archivo configuración DNS <code>/etc/named.conf</code>	102
Anexo 5 – Archivos definición de zonas DNS	103
Anexo 6 – Archivo configuración del OpenSSL <code>openssl.cnf</code>	106
Anexo 7 – Scripts creación de certificados	111
Anexo 8 – Archivo configuración del OpenLDAP <code>slapd.conf</code>	113
Anexo 9 – Archivo configuración clientes OpenLDAP <code>lapd.conf</code>	116
Anexo 10 – Carga inicial de datos en OpenLDAP	117
Anexo 11 – Archivo configuración FreeRADIUS <code>eap.conf</code>	119
Anexo 12 – Archivo de mapeo FreeRADIUS-LDAP <code>ldap.attrmap</code>	124
Anexo 13 – Configuración cortafuegos	126
Anexo 14 – Archivo configuración IPSec <code>ipsec.conf</code>	132

Índice de figuras

2.1	Conexión en modo “ad hoc”	12
2.2	Conexión en modo infraestructura	13
2.3	Inclusión de un cortafuego en una WLAN	14
2.4	Arquitectura IEEE 802.1x	17
2.5	Red Privada Virtual (VPN)	20
2.6	Arquitectura general	24
3.1	Un dominio en varios sitios	28
3.2	Varios dominios en un sitio	29
3.3	Instalación servidor DHCP	30
3.4	Crear nuevo grupo en AD	31
3.5	Definir nuevo grupo en AD	32
3.6	Añadir un usuario a un grupo en AD	32
3.7	Cambiar propiedades de un usuario	33
3.8	Instalar el Servidor de Certificados	35
3.9	Definir el tipo entidad emisora de certificados	35
3.10	Datos entidad emisora de certificados raíz	36
3.11	Pantalla para solicitar un nuevo certificado de usuario	37
3.12	Crear nueva plantilla de certificados	37
3.13	Selección de la plantilla “Sesión autenticada”	38
3.14	Modificar propiedades de las plantillas de certificado	38
3.15	Registrar IAS en AD	40
3.16	Agregar cliente al servidor IAS	41
3.17	Definir tipo de autenticación utilizada en el acceso	42
3.18	Configuración de red del AP	43
3.19	Definición de la seguridad en el AP	44
3.20	Elección del tipo de solicitud de certificado	46
3.21	Entrega del certificado solicitado	46

4.1	Aspecto LDAP Browser/Editor	59
4.2	Editar usuario con LDAP Browser/Editor	59
4.3	Cliente EAP-TTLS para Windows	67
4.4	Configuración “User Info” del cliente “Odyssey”	68
4.5	Configuración “Authentication” del cliente “Odyssey”	69
4.6	Configuración del protocolo EAP-TTLS en el cliente “Odyssey”	69
4.7	Configuración propiedades de red del cliente “Odyssey”	70
5.1	Atributos de usuarios Linux en Active Directory	77
5.2	Consulta sobre AD con cliente LDAP	78
5.3	Firewall Builder	80
5.4	Protocolos permitidos	80
a1.1	Añadir nuevo ámbito al servidor DHCP	93
a1.2	Asignar nombre al nuevo ámbito DHCP	94
a1.3	Rango de direcciones IP y máscara que proporciona el servidor DHCP	94
a1.4	Puerta de enlace predeterminada que proporciona el servidor DHCP	96
a1.5	Nombre de dominio primario y servidores DNS que ofrecerá el DHCP	96
a2.1	Propiedades de la conexión de red inalámbrica	98
a2.2	Configuración de la asociación inalámbrica	99
a2.3	Definición del tipo de autenticación en la conexión	100

Capítulo 1

Introducción

1.1 - Justificación del Proyecto

El auge de las redes inalámbricas de área local (*WLAN, Wireless Local Area Network*) ha provocado que muchas empresas y entidades se vean en la pseudo-obligación y/o deseo de hacer uso de ellas en un intento de obtener los beneficios que comporta el uso de la nueva tecnología. Entre otros muchos beneficios, no hay que pasar por alto el valor añadido que le da a la empresa el hecho de ofrecer a sus clientes acceso inalámbrico a su red.

Lógicamente, dichas empresas y/o entidades están dispuestas a ofrecer el nuevo servicio siempre y cuando no les ocasione graves trastornos en su infraestructura de red, y no suponga un agujero de seguridad en el sistema principal. El objetivo de las empresas es implantar el servicio inalámbrico con los mínimos cambios posibles en su infraestructura de red, y con un mínimo impacto en sus sistemas de seguridad.

Si la empresa que desea implantar un sistema de comunicaciones inalámbrico ya posee dispositivos que controlen los accesos por cable, el impacto que ocasionará en las infraestructuras será mínimo, pero en temas de seguridad se deberán tener muy en cuenta la gran cantidad de vulnerabilidades del sistema, convirtiendo la seguridad en uno de los puntos más importantes a considerar cuando se implementa una WLAN.

Dentro del mundo empresarial está muy extendido el uso de sistemas operativos de red de la familia Microsoft Windows, llámense Windows NT Server, Windows 2000 Server o el más avanzado, Windows 2003 Server. De los tres, el más usado actualmente es el sistema operativo Windows 2000 Server que integra el servicio de directorio propietario de Microsoft llamado “Active Directory”.

En los últimos años, se está extendiendo entre las empresas de todo el mundo el uso del sistema operativo Linux, aunque la mayoría de ellas no están sustituyendo un sistema operativo por otro, sino que comienzan a hacer uso de algunos de los servicios de red que ofrece el sistema operativo Linux. Esta mezcla de sistemas provoca que se deban crear y/o adaptar una serie de protocolos que permitan la comunicación entre ambos.

En este proyecto se diseña e implementa la interconexión entre un servidor RADIUS (*Remote Authentication Dial In User Service*) que se ejecuta en una máquina con sistema operativo Linux, y un servicio de directorio (Active Directory) que se ejecuta en una máquina con sistema operativo Microsoft Windows 2000 Server.

1.2 - Objetivo del Proyecto

El objetivo del proyecto es diseñar e implementar una red segura con un acceso externo inalámbrico; con una particularidad que le distingue de un típico acceso inalámbrico a una red: los procesos de autorización, autenticación y acceso no se ejecutan en un entorno con un único sistema operativo, algunos procesos se ejecutan en una máquina con SO Linux y otros en una máquina con SO Microsoft Windows, con el consiguiente intercambio de información entre ambos sistemas.

La máquina con SO Linux ejecuta un servidor RADIUS cuyo cliente es el punto de acceso inalámbrico; en concreto, se utiliza el servidor FreeRADIUS que se encarga de autorizar y autenticar el acceso a la red de todos aquellos usuarios que intentan conectarse vía el punto de acceso inalámbrico. Para poder realizar los procesos de autorización y autenticación, el servidor RADIUS se apoya en un servidor LDAP que se ejecuta en la misma máquina.

Si el usuario que se desea conectar a la red está autorizado a hacerlo, el servidor RADIUS le proporciona las credenciales del usuario al servicio Active Directory que se ejecuta en una máquina con SO Microsoft Windows, siendo éste el que se encarga de gestionar el acceso del usuario a los recursos de la red, proporcionándole al usuario todo su perfil (privilegios y atributos).

Dado que el control de la seguridad en las comunicaciones inalámbricas es uno de los temas más importantes a la hora de implementar una infraestructura inalámbrica, en este proyecto se ha realizado un estudio de todas las posibles formas de securizar las comunicaciones inalámbricas, implementando en el desarrollo final los protocolos EAP-TLS y EAP-TTLS; además, se ha habilitado una red privada virtual (VPN) y un cortafuegos para conseguir una mayor seguridad en las comunicaciones entre el cliente inalámbrico y la red física de servidores, evitando a su vez el acceso de intrusos.

1.3 - Enfoque y método seguido

El enfoque y método seguido para la realización de este proyecto ha consistido en dividirlo en cuatro fases, sirviendo las tres primeras como mesa de pruebas para la obtención de conocimientos y así poder abordar la cuarta fase, que coincide con los objetivos propuestos en el proyecto, con plenas garantías de éxito.

En la primera fase se realiza un estudio de todas las posibles formas de securizar una comunicación inalámbrica.

En la segunda fase se utiliza el servidor RADIUS propio de Microsoft (IAS-Internet Authentication Service) y el servicio Active Directory también de Microsoft para gestionar la autorización, autenticación y gestión de acceso de los usuarios. En esta

fase se analizan y estudian ambos servicios, lo que proporciona un conocimiento exhaustivo de los protocolos que utilizan.

La tercera fase consiste en utilizar los mismos servicios de autenticación, autorización y gestión de acceso pero en un entorno Linux, sin utilizar ningún servicio de Windows.

Una vez finalizadas las tres primeras fases ya se poseen todas las pruebas y conocimientos necesarios para abordar la cuarta fase, consistente en la fusión de ambos sistemas, Linux y Windows.

1.4 - Planificación del Proyecto

Dado que el proyecto se divide en cuatro fases bien diferenciadas, se ha efectuado una planificación que se adapta a dichas etapas. Se ha realizando una división de cada fase en una serie de tareas y subtareas que permiten efectuar un adecuado seguimiento del trabajo, facilitando de esta forma alcanzar el objetivo propuesto.

- **TAREA 1:** Fase 1. Estudio de todas las posibilidades existentes a la hora de securizar una comunicación inalámbrica.
 - **Subtarea 1.1** Estudio de todas las posibilidades de securización de una comunicación inalámbrica.
Duración: 15 horas
Objetivo: Obtener todos los conocimientos necesarios a nivel de seguridad en las comunicaciones para poder implementar un sistema inalámbrico inviolable.
 - **Subtarea 1.2** Documentación sobre la seguridad.

Duración: 3 horas
Objetivo: Documentar toda la información obtenida en la búsqueda de los diferentes sistemas de segurización de una red inalámbrica.
Hito: Documentación sobre seguridad en redes inalámbricas.

- **TAREA 2:** Fase 2. Implementación de un sistema de control de acceso con Microsoft Windows 2000 Server

- **Subtarea 2.1** Análisis y estudio de los servicios y protocolos propietarios de Microsoft.

Duración: 30 horas
Objetivo: Analizar y estudiar toda la infraestructura con Microsoft Windows 2000 Server.

- **Subtarea 2.2** Instalación y configuración de Windows 2000 Server (SO, Active Directory y IAS)

Duración: 15 horas
Objetivo: Preparar una máquina con todo el hardware y software necesario para poder realizar las pruebas de autenticación, autorización y gestión de acceso de usuarios.

- **Subtarea 2.3** Instalación y configuración del punto de acceso y el cliente inalámbrico.

Duración: 12 horas
Objetivo: Instalar y configurar un punto de acceso en la red, y configurar un cliente con red inalámbrica para realizar las pruebas de conectividad.

- **Subtarea 2.4** Documentar el estudio y las pruebas con MS Windows 2000 Server.
Duración: 3 horas
Objetivo: Documentar todas las pruebas de conectividad, protocolos, manual de configuración, etc.
Hito: Documentación sobre el análisis, configuración y pruebas de una infraestructura de gestión de acceso con MS Windows.

- **TAREA 3:** Fase 3. Implementación de un sistema de control de acceso con Linux Fedora Core 1
 - **Subtarea 3.1** Análisis y estudio de los servicios y protocolos de utilizados por Linux.
Duración: 30 horas
Objetivo: Analizar y estudiar toda la infraestructura con Linux.

 - **Subtarea 3.2** Instalación y configuración de Linux (SO, FreeRADIUS, OpenLDAP)
Duración: 15 horas
Objetivo: Preparar una máquina con todo el hardware y software necesario para poder estudiar y realizar las pruebas de autenticación, autorización y gestión de acceso de usuarios.

 - **Subtarea 3.3** Instalación y configuración del punto de acceso.
Duración: 3 horas
Objetivo: Instalar y configurar un punto de acceso en la red, capaz de comunicarse con un servidor RADIUS de Linux.

- **Subtarea 3.4** Documentar las pruebas con Linux.
 - Duración: 3 horas
 - Objetivo: Documentar todas las pruebas de conectividad, protocolos, manual de configuración, etc.
 - Hito: Documentación sobre el análisis, configuración y pruebas de una infraestructura de gestión de acceso con Linux.

- **TAREA 4:** Fase 4. Unión de las dos plataformas: Windows y Linux
 - **Subtarea 4.1** Análisis y estudio de los servicios y protocolos de utilizados.
 - Duración: 42 horas
 - Objetivo: Analizar y estudiar toda la infraestructura con los dos sistemas operativos interconectados.

 - **Subtarea 4.2** Interconexión en la misma red de los tres elementos utilizados en el sistema: Punto de acceso, máquina con SO Linux y máquina con SO Windows 2000 Server.
 - Duración: 15 horas
 - Objetivo: Preparar toda la infraestructura para abordar el objetivo principal del proyecto.

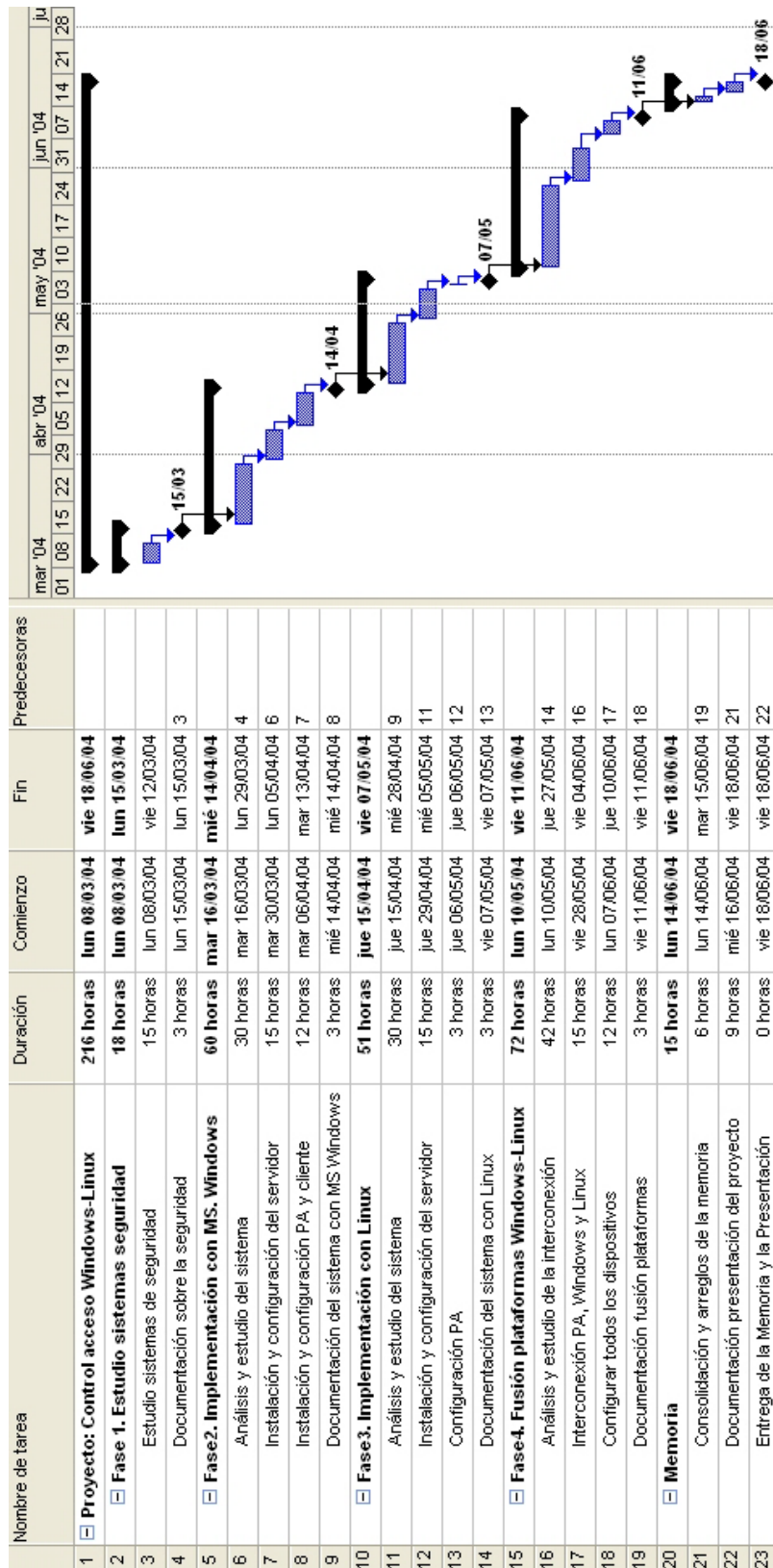
 - **Subtarea 4.3** Configurar todos los dispositivos del sistema.
 - Duración: 12 horas
 - Objetivo: Configurar todos los dispositivos del sistema de forma que permita una correcta interacción entre ellos, incluyendo la instalación de un firewall, y un servidor VPN.

- **Subtarea 4.4** Documentar toda la configuración y pruebas.
 - Duración: 3 horas
 - Objetivo: Documentar todas las pruebas de conectividad, protocolos, manual de configuración, etc.
 - Hito: Documentación sobre el análisis, configuración y pruebas de una infraestructura de gestión de acceso con un servidor RADIUS en Linux, y el servicio Active Directory de Microsoft.

- **TAREA 5:** Arreglos finales de la memoria y documentación de la presentación del proyecto.
 - **Subtarea 5.1** Consolidación y arreglos finales de la memoria del proyecto.
 - Duración: 6 horas
 - Objetivo: Consolidación del documento final del proyecto.
 - Hito: Memoria del proyecto.

 - **Subtarea 5.2** Preparar la presentación del proyecto.
 - Duración: 9 horas
 - Objetivo: Diseño e implementación de la presentación virtual del proyecto.
 - Hito: Presentación virtual del proyecto.

1.5 – Calendario de trabajo



Capítulo 2

Seguridad en las redes inalámbricas

El presente capítulo no pretende ser un estudio completo y exhaustivo sobre la seguridad en las redes inalámbricas, ni abarcar todas sus vertientes, solo intenta ser un compendio de todos aquellos conceptos básicos y necesarios para poder comprender el resto del proyecto.

“Cuando una persona habla en un espacio abierto, solo hay que escucharla y entender su idioma para comprender lo que está diciendo.”

2.1 - Introducción

La libertad que proporcionan los dispositivos inalámbricos es su principal ventaja y su principal inconveniente. Al contrario que en el caso de las redes cableadas tradicionales, un intruso no necesita tener acceso físico a nuestro edificio u oficina para intentar asaltar la red interna. Las señales de radio que utilizan los dispositivos de red inalámbricos circulan con total libertad a través del aire, incluso atravesando paredes, y por lo tanto están al alcance de cualquiera que tenga capacidad para interceptarlas. Un asaltante puede intentar entrar en nuestra red desde cualquier punto al que llegue nuestra señal de radio. Además de todo esto, la interceptación de paquetes de datos para su análisis nos pasará inadvertida ya que no hay modo de saber si alguien lo está haciendo. La idea es muy similar a la transmisión de información que realiza una emisora de televisión, la diferencia es que a la emisora de televisión le interesa que mucha gente reciba y decodifique su señal, pero si estamos hablando de transmisión de datos entre instalaciones militares, o estamos

realizando una transacción bancaria, lo que menos interesa es que alguien pueda captar nuestra señal y la pueda decodificar.

Estos riesgos no implican que no se deban utilizar las redes inalámbricas, solo hay que conocerlos y tomar todas las medidas oportunas para atajarlos; una vez conseguido este objetivo se puede afirmar que una red inalámbrica es tan segura como cualquier otro sistema de comunicación.

2.2 – IEEE 802.11

La familia de protocolos 802.11, definidos como estándar industrial por el IEEE (*Institute of Electrical and Electronics Engineers*), son los más utilizados en los sistemas de comunicaciones inalámbricos para redes de datos. Existen actualmente en el mercado diferentes sistemas operativos que soportan nativamente el estándar 802.11b –más conocido como Wi-Fi-- que utiliza la banda de radiofrecuencia de 2,4GHz., alcanzando velocidades de transmisión de hasta 11Mbps con radios de acción de hasta 400 metros en condiciones óptimas. Esta es la variante más utilizada en la actualidad de la familia de protocolos 802.11, aunque poco a poco va ganando mercado el estándar 802.11g con una velocidad de transmisión de 54Mbps.

Los elementos fundamentales dentro de una red inalámbrica Wi-Fi son el punto de acceso (*AP, Access Point*), que centraliza el servicio de acceso a la red inalámbrica, semejante a un Hub o Switch en una red cableada, y los nodos inalámbricos o estaciones, que son los distintos dispositivos que se conectan a la red inalámbrica utilizando algún tipo de adaptador de red sin cables.

Dentro de las redes inalámbricas existen dos modos de operación diferentes dependiendo del tipo de conectividad existente entre los distintos nodos que forman la red:

- **Modo “ad hoc” o IBSS (*Independent Basic Service Set*)**

Esta topología, tal y como se muestra en la figura 2.1, se caracteriza por no tener un punto de acceso (AP) encargado de centralizar y coordinar las comunicaciones, sino que los nodos se comunican directamente entre sí (peer-to-peer).

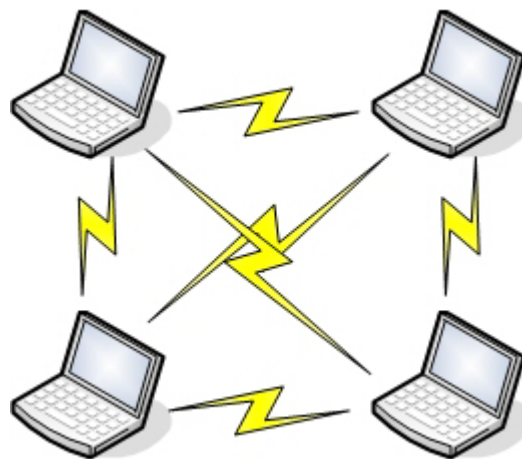


Figura 2.1: Conexión en modo “ad hoc”.

Es equivalente al modo entre iguales de las redes locales cableadas. El área de cobertura está limitada al alcance de cada estación individual.

- **Modo infraestructura o BSS (*Basic Service Set*)**

En esta topología hay como mínimo un Punto de Acceso encargado de centralizar las comunicaciones, las estaciones inalámbricas no se pueden comunicar entre sí, y todo el tráfico debe pasar de forma obligatoria por el AP.

La mayoría de redes inalámbricas instaladas en las empresas utilizan el modo infraestructura con uno o más puntos de acceso, actuando estos como un HUB en una LAN, redistribuyen los datos hacia todas las estaciones inalámbricas. Si existe más de un punto de acceso en la red, cada uno puede actuar como repetidor o puente entre redes inalámbricas, y al conjunto se le denomina ESS (*Extended Service Set*).

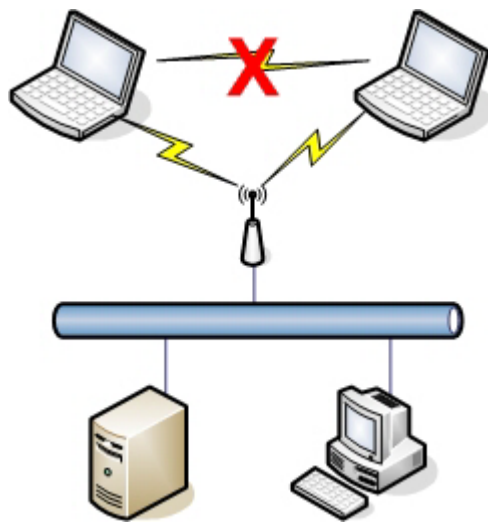


Figura 2.2: Conexión en modo infraestructura.

Viendo la figura 2.2 se observa que el punto de acceso actúa de puente entre la red inalámbrica y otras redes cableadas, permitiendo el acceso transparente a éstas de los nodos inalámbricos. Teniendo en cuenta los problemas de seguridad que se explicarán a lo largo de este capítulo, un primer elemento a incluir en el sistema es un dispositivo cortafuegos encargado de regular el tráfico entre el punto de acceso y la red local cableada (Figura 2.3).

Los dos temas más importantes a controlar en la seguridad de las redes inalámbricas de área local (WLAN) son la autenticación de usuario y el cifrado de datos. Dado que cualquiera puede interceptar los paquetes de datos que se transmiten a través del espectro electromagnético, es necesario cifrarlos para que los interceptores no puedan interpretar su significado. El protocolo 802.11b provee un mecanismo para

cifrar los datos conocido como WEP (*Wired Equivalent Privacy*), pero como se verá más adelante, posee multitud de problemas convirtiéndolo en un sistema poco seguro. Por otra parte, como cualquiera con un dispositivo adecuado, y dentro del área de cobertura puede intentar conectarse a nuestro punto de acceso, es especialmente importante conseguir un método fiable de autenticación o restricción de acceso.

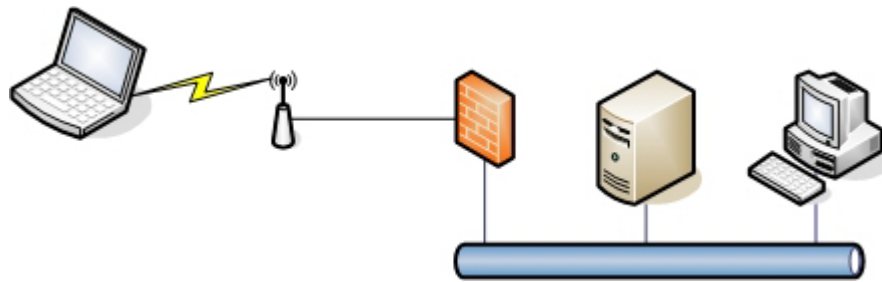


Figura 2.3: Inclusión de un cortafuegos en una WLAN.

2.3 – WEP

El estándar 802.11 para redes inalámbricas incluye un protocolo de privacidad equivalente al cableado –WEP (*Wired Equivalent Privacy*)–, usado para proteger comunicaciones a nivel de enlace contra escuchas indeseadas y ataques de otra naturaleza. WEP proporciona dos elementos críticos para una arquitectura de comunicaciones segura: autenticación y confidencialidad. WEP usa un mecanismo de cifrado de clave compartida basado en el algoritmo RC4. La clave que el cliente usa para cifrar los datos a transmitir debe ser la misma que usa el punto de acceso. El estándar 802.11 especifica usar claves de 40 bits, aunque hay dispositivos que implementan claves de hasta 104 bits para incrementar la seguridad. El cifrado de datos proporciona confidencialidad en las transmisiones entre los dispositivos inalámbricos.

Cuando una estación se intenta asociar a un punto de acceso, la estación debe autenticarse frente al punto de acceso. En el proceso de asociación, la estación y el punto de acceso negocian el tipo de autenticación que utilizarán, que puede ser “abierta” o “compartida”:

- **Autenticación abierta:** como su propio nombre indica, la entrada a la red está totalmente expedita a cualquiera que conozca el SSID (*Service Set Identifier*) del punto de acceso.
- **Autenticación compartida:** es un mecanismo parecido a la clásica autenticación desafío/respuesta en la que el AP envía un desafío a la estación cliente, la cual lo devuelve cifrado con la clave WEP para comprobar que comparten la misma clave.

2.4 – Problemas del WEP

Desafortunadamente, la especificación WEP incluida en el estándar 802.11 no proporciona una seguridad equivalente a una red cableada. Son varios los problemas que tiene para considerarlo un protocolo seguro.

2.4.1 – Manejo de claves

El estándar WEP ignora completamente el uso de manejadores de claves. Esto causa problemas cuando el número de usuarios de la red inalámbrica crece. El uso de claves compartidas por todos los usuarios implica que cualquier usuario puede descifrar las comunicaciones de los demás, es decir, debe existir una confianza plena entre todos los usuarios de la red. Esto es una situación idílica, sobre todo cuando crece el número de usuarios.

Existe la posibilidad de realizar rotaciones entre un conjunto de claves preestablecidas. Este sistema no mejora en exceso la seguridad, ya que todos los usuarios son conocedores del conjunto de claves.

2.4.2 – Cifrado

El IEEE seleccionó un sistema de cifrado de 40 bits porque es el nivel más alto de encriptación que permiten las leyes internas de algunos países, como Estados Unidos, para poder exportar dispositivos de cifrado a otros países. Desafortunadamente, por un fallo en el algoritmo generador de claves, las longitudes efectivas de las claves sólo alcanzan un nivel de 22 bits. Con niveles de cifrado tan bajos es fácil romper la clave, como ya se demostró hace unos años. Actualmente existen en el mercado herramientas software gratuitas que permiten encontrar las claves de cifrado. Cabe indicar que cuando se realiza una “autenticación compartida”, el mensaje enviado por el punto de acceso viaja en texto plano, y la respuesta del nodo lo hace cifrado, la posesión de los dos mensajes facilita mucho el proceso de ruptura de la clave WEP.

2.4.3 – Integridad

El estándar WEP no verifica la integridad de los paquetes que se envían y reciben, basta que un atacante use una herramienta de análisis de tráfico para poder manipular un simple bit cualquiera en los paquetes en circulación por la red para causar problemas en las comunicaciones, e incluso llegar a impedir las, sería lo que se conoce como un ataque de denegación de servicio (DoS). El IEEE está trabajando en el protocolo 802.11i que implementa integridad de mensajes para evitar este problema.

2.5 – Sistemas de seguridad adicionales

Una vez visto los problemas de seguridad que conlleva el protocolo WEP, es necesario implementar nuevos sistemas de seguridad adicionales que permitan conseguir un auténtico nivel de seguridad en las redes WLAN, proporcionando integridad, autenticación y confidencialidad.

2.5.1 – 802.1x

El estándar IEEE 802.1x no fue diseñado inicialmente para redes inalámbricas, realmente se diseñó para redes cableadas y proporcionaba un nivel añadido de seguridad a las redes, evitando el acceso a los equipos de la red sin una autenticación previa; además, proporciona los mecanismos necesarios para la gestión y reparto de las claves de cifrado. En la figura 2.4 se muestra al arquitectura del estándar IEEE 802.1x.

El puerto de comunicaciones de los dispositivos que implementan el estándar 802.1x se encuentra por defecto cerrado, permitiendo exclusivamente el paso de tramas de autenticación, en el momento que el cliente se ha autenticado, el puerto de comunicaciones se abre permitiendo el paso de cualquier tipo de trama.

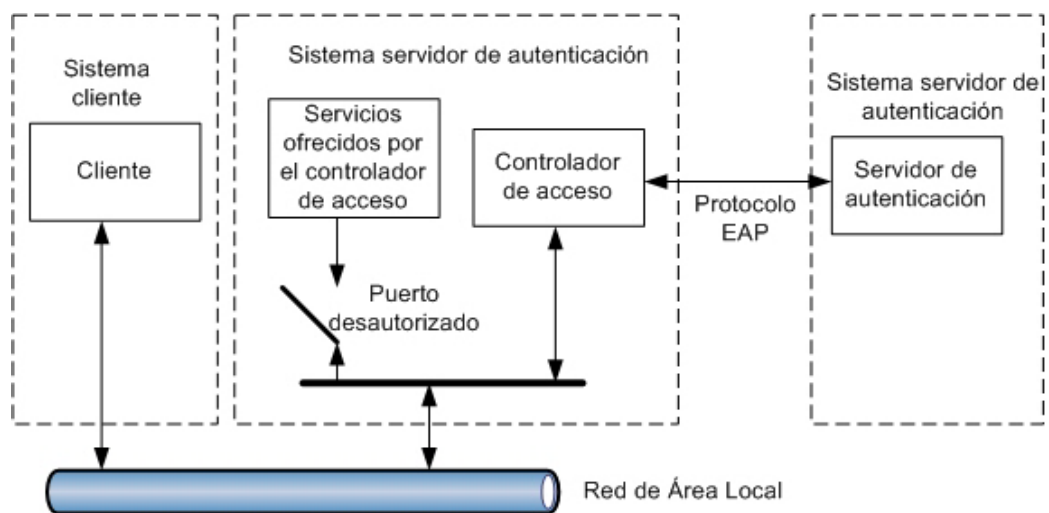


Figura 2.4: Arquitectura IEEE 802.1x.

El estándar 802.1x se basa en el protocolo EAP (*Extensible Authentication Protocol*) que le permite integrarse en sistemas externos de autenticación y autorización. El uso de este protocolo de seguridad conlleva la existencia de un servidor de autenticación de usuarios, normalmente un servidor RADIUS.

2.5.2 – EAP

El protocolo EAP (*Extensible Authentication Protocol*) se creó como una extensión al protocolo PPP (*Point-to-Point Protocol*) que permitiese el uso de cualquier tipo de sistema de autenticación para el acceso a redes. Al contrario que en PPP, con EAP el mecanismo de autenticación no se escoge durante la fase de establecimiento de la conexión punto a punto, sino que se negocia el sistema a emplear entre los nodos que se comunican. Este sistema permite el uso de cualquier tipo de método de autenticación (conocidos como tipos EAP), basta con instalar tanto en los clientes como en los servidores los controladores adecuados. Esta enorme flexibilidad permite dotar de diversos sistemas de autenticación a las conexiones (desafío-respuesta, certificados digitales, tarjetas inteligentes...), convirtiéndose en una tecnología fundamental para la seguridad de las conexiones.

EAP está soportado explícitamente en la capa de conexión de la especificación 802 del IEEE y por lo tanto está soportado por los dispositivos inalámbricos adheridos al estándar IEEE 802.11x. El protocolo 802.1x define como se debe usar EAP para autenticar a este tipo de dispositivos y de hecho es el único protocolo de autenticación que soporta.

Los tres tipos de autenticación EAP más utilizados son:

- **EAP-MD5 CHAP:** utiliza el mismo protocolo de desafío que PPP, pero empleando mensajes EAP. Se suele utilizar para validar credenciales a partir de nombres de usuario y contraseñas. Sin embargo, no es adecuado para las redes inalámbricas por varios motivos, pero fundamentalmente por que requiere que las contraseñas se almacenen de forma que se puedan descifrar,

algo que no está habilitado (ni conviene hacerlo) en cuentas de dominio de Windows, ni en UNIX/Linux. Cabe decir que el sistema operativo Microsoft Windows XP SP1 no lo soporta por los motivos expuestos.

- **EAP-TLS:** esta variante utiliza TLS (*Transport Level Security*) para habilitar la autenticación en entornos de seguridad basados en certificados digitales. Es obligatorio su uso en el caso de requerir tarjetas inteligentes, por ejemplo. El intercambio de mensajes EAP-TLS proporciona autenticación mutua (evita ataques de “hombre en el medio”) así como intercambio seguro de claves entre el servidor y los clientes. En la actualidad es tal vez el método de autenticación más seguro que existe. Es la elección más adecuada para redes inalámbricas por muchos motivos pero fundamentalmente porque usa certificados digitales, no depende de ninguna contraseña que el usuario deba conocer, no requiere intervención por parte del usuario, y utiliza infraestructura de clave pública de alta seguridad.
- **EAP-TTLS:** El EAP-TTLS (*Extensive Authentication Protocol-Tunneles Transport layer Security*) es una extensión de EAP-TLS, sólo requiere certificados en el servidor, lo que subsana una desventaja importante respecto a EAP-TLS, cuya gestión de certificados es mucho más tediosa y pesada. Con EAP-TTLS se elimina la necesidad de configurar certificados para cada cliente de la red inalámbrica. Además, EAP-TTLS autentica al cliente en el sistema con las credenciales ya existentes basadas en password, y encripta credenciales y password para garantizar la protección de la comunicación inalámbrica.

2.5.3 – VPN

Una VPN (*Virtual Private Network*) es una red privada que se extiende mediante un proceso de encapsulación y encriptación de datos a distintos puntos remotos mediante el uso de infraestructuras públicas de transporte. La figura 2.5 muestra gráficamente el concepto de VPN.

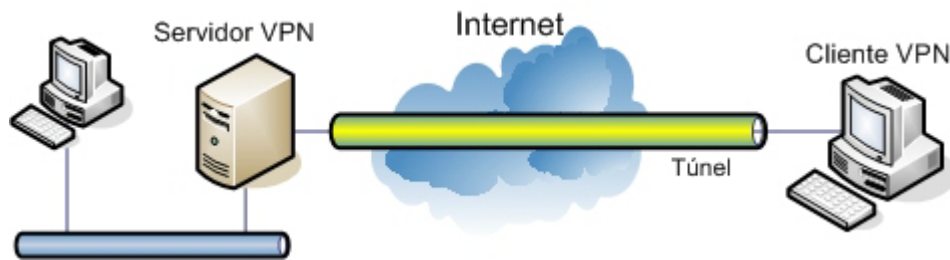


Figura 2.5: Red Privada Virtual (VPN)

¿Porqué utilizar redes privadas virtuales en las redes inalámbricas?, la circulación de ondas electromagnéticas por el aire se asemeja mucho al envío de datos por una red pública, cualquiera puede escuchar la comunicación, así que al igual que se utilizan redes privadas virtuales para transmitir información de forma segura entre nodos a través de Internet, se pueden utilizar redes privadas entre dispositivo inalámbricos que transmiten la información a través del aire.

Los datos transportados a través de una VPN pueden ser paquetes de un protocolo distinto al que maneja la red pública, es decir, en vez de enviar un paquete tal y como fue generado por el nodo emisor, el protocolo de tunneling (ya sea L2TP, IPSec, etc) encapsula el paquete con un encabezado (Header) adicional que pertenece al protocolo de transporte de la red sobre la que se establece la VPN. Los paquetes encapsulados son entonces enrutados sobre la red pública entre el servidor y el cliente VPN. A esta ruta lógica, a través de la cual viajan los paquetes encapsulados sobre la red pública, se le llama “túnel”.

Cuando los paquetes encapsulados llegan a su destino, el paquete es desencapsulado y reenviado a su destino final, con el encabezado del paquete original en caso de haber sufrido una transformación en el origen.

Algunos de los protocolos más utilizados para crear redes privadas virtuales son:

- **Point-to-Point Tunneling Protocol (PPTP):** permite que el tráfico IP, IPX o NetBEUI sea encriptado y encapsulado en encabezados IP para ser enviado a través de un red pública IP como Internet. Este protocolo fue creado por Microsoft, existiendo también una implementación para Linux.
- **Layer 2 Tunneling Protocol (L2TP):** permite que el tráfico IP o IPX sea encriptado y enviado sobre cualquier medio que soporte entrega de datagramas punto-a-punto, tales como IP, X.25, Frame Relay o ATM.
- **IP Security (IPSec) Túnel Mode:** permite que paquetes IP sean encriptados y encapsulados en encabezados IP para ser enviados a través de una red pública IP.

Cabe hacer la aclaración que no necesariamente los paquetes procesados por el protocolo de tunneling tienen que ser diferentes al protocolo de la red de transporte, tal es el caso de IP-sobre-IP y de IPSec, que encapsulan paquetes IP para posteriormente ser transportados sobre una red IP.

2.5.4 – WPA

Desde el punto de vista de la seguridad, WPA (*Wi-Fi Protected Access*) constituye un avanzado puente tecnológico entre los débiles sistemas de los primeros dispositivos y el futuro estándar 802.11i.

Aunque el protocolo 802.11i (ya conocido como WPA2) no es todavía un estándar, su desarrollo se encuentra en una fase muy avanzada. El protocolo WPA utiliza gran parte de la tecnología desarrollada para el estándar 802.11i como base para solventar los problemas de seguridad existentes en Wi-Fi. Incorpora autenticación mediante

802.1x y EAP además de un nuevo protocolo (TKIP, *Temporal Key Integrity Protocol*) y un sistema de verificación de integridad llamado Michael. WPA posee principalmente dos atribuciones: proveer de un sistema de acceso seguro y autenticación para usuarios, y proteger los datos transmitidos de una forma mucho más segura que WEP. Tal y como se ha visto anteriormente, los protocolos 802.1x y EAP sentaban las bases de un control de acceso seguro, sin embargo en el caso de Wi-Fi son un añadido externo que, por suerte, se puede usar con el protocolo para fortalecerlo. En el caso de WPA, ambos forman parte intrínseca de la especificación.

En el protocolo 802.1x la regeneración de claves de cifrado de paquetes es opcional, aparte de que no ofrecen elemento alguno para el cambio automatizado de las claves de cifrado globales o de multidifusión. En WPA la regeneración de claves de unidifusión y multidifusión es obligatoria. En el caso de claves establecidas con un único cliente (unidifusión) es el protocolo TKIP el encargado de hacerlo, manteniendo sincronizadas las claves entre cliente y punto de acceso. En el caso de claves de tráfico global (multidifusión) se incluye un sistema que permite al punto de acceso el envío seguro de la nueva clave a los clientes que estén conectados.

TKIP incrementa el tamaño de las claves de cifrado de 40 a 128 bits y sustituye las claves estáticas de WEP por claves dinámicamente generadas y distribuidas por el punto de acceso tras la autenticación de los usuarios. El sistema empleado elimina la predicción de las claves en la que se apoyan los posibles atacantes para romper WEP. Ni que decir tiene que esto hace casi imposible descifrar las comunicaciones; además, en WPA existe un nuevo algoritmo llamado Michael que se usa para el cálculo de códigos de integridad de mensaje (*MIC, Message Integrity Code*). Este MIC está pensado para evitar que un atacante capture paquetes, los modifique y los reenvíe. Michael define una avanzada función matemática que se calcula en cada paquete por parte del emisor y el receptor, incluyéndola el primero en el propio paquete a transmitir. Al recibirse el paquete en el destino se compara el MIC calculado con el contenido en éste y si no coinciden se asume que los datos han sido modificados y por consiguiente, descartando el paquete.

2.6 – 802.11i

El futuro estándar 802.11i aportará una gran ventaja respecto a los sistemas de seguridad vistos hasta ahora, no será necesario recurrir a un conjunto de elementos externos al estándar para poder implementar una WLAN segura, él mismo proporcionará todos los elementos necesarios para conseguir una WLAN tan segura como una LAN.

Además de los elementos y protocolos vistos en el apartado anterior, el estándar 802.11i utiliza el protocolo TSC (*TKIP Sequence Counter*) para evitar los ataques por repetición, los paquetes fragmentados y recibidos fuera de orden son descartados de forma automática. Otra novedad del estándar es el uso de métodos de cifrado basados en AES (*Advanced Encryption Standard*), concretamente utilizará el algoritmo CCM (*Counter mode with CBC-MAC*) que proporciona una perfecta privacidad, integridad y autenticación.

2.7 – RADIUS

RADIUS (*Remote Authentication Dial-In User Service*) es un protocolo que proporciona autenticación y autorización centralizada para acceso a redes de todo tipo. Aunque inicialmente se desarrolló para habilitar el acceso telefónico remoto, como su nombre indica, en la actualidad se usa para multitud de aplicaciones adicionales tales como servidores de redes privadas virtuales (VPN), acceso mediante ADSL, y por supuesto para redes inalámbricas.

Una infraestructura RADIUS está formada básicamente por los clientes que requieren el acceso, los clientes RADIUS y los servidores RADIUS. En el caso de las WLAN, el cliente RADIUS es el propio punto de acceso inalámbrico, que actúa

como un cliente RADIUS desde el punto de vista de la red interna, pero actúa como servidor de acceso desde la perspectiva de los clientes inalámbricos.

El servidor RADIUS concede o deniega el acceso de los clientes a la red interna utilizando para la autenticación y autorización la información contenida en cualquier tipo de base de datos que almacene información de usuarios, ya sean directorios de usuarios como OpenLDAP o el propio Active Directory de Microsoft Windows, como el sistema propio de UNIX para gestionar los usuarios.

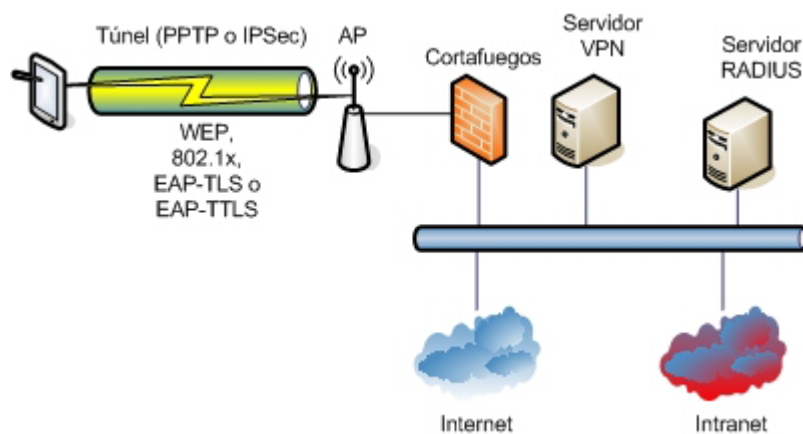


Figura 2.6: Arquitectura general

Tanto Windows 2000 Server como Windows 2003 Server incluyen de serie un servidor RADIUS, que se conoce como Servicio de Autenticación de Internet o IAS (*Internet Authentication Service*). Dentro del entorno Linux el servidor RADIUS más utilizado y conocido es FreeRADIUS.

La figura 2.6 muestra la arquitectura general del sistema, sin tener en cuenta los sistemas operativos utilizados.

2.8 – Conclusión

Vistas todas las posibilidades existentes a la hora de securizar una red inalámbrica, y el coste material y de mantenimiento que puede suponer realizar una implementación real, habrá que sopesar el nivel de seguridad que se desea alcanzar con el coste material y de mantenimiento que ello supone.

Dado que el presente proyecto está enfocado a implementar una red inalámbrica a la que podrán acceder cientos de usuarios, y que la red cableada a la que se conectará el punto de acceso posee datos y servicios críticos, hay que intentar implementar el máximo nivel de seguridad posible con las tecnologías actuales.

Lo ideal sería implementar el estándar IEEE 802.11i, pero todavía no se encuentra totalmente operativo, además de no existir apenas hardware que lo soporte. La segunda opción sería utilizar WPA, pero al igual que el estándar 802.11i, existe poco hardware y software que lo soporte, no es un sistema que se encuentre de forma nativa en los sistemas operativos, lo que implicaría que los cientos de posibles clientes de la red deberían adaptar sus dispositivos inalámbricos actualizando el software y firmware, situación nada aconsejable si el objetivo es facilitar el acceso a los usuarios.

La mejor solución y la más segura consiste en implementar un sistema utilizando el protocolo WEP, añadiendo los siguientes sistemas adicionales: protocolos 802.1x y EAP, con autenticación TLS o TTLS, un servidor de túneles (PPTP o IPSec), un servidor RADIUS y un cortafuego.

Capítulo 3

WLAN segura con MS Windows 2000 Server

Una vez vistas todas las características que debería cumplir una red inalámbrica para considerarla segura, en este capítulo se explica como implementarla utilizando exclusivamente los servicios que ofrece el sistema operativo Microsoft Windows 2000 Server.

Los requerimientos hardware necesarios son:

- Máquina con sistema operativo Microsoft Windows 2000 Server.
- Punto de acceso.
- Cliente con un dispositivo de red inalámbrico.

Por la extensión de la explicación y por estar fuera del alcance de este proyecto, se parte de la idea de que ya tenemos una máquina con el sistema operativo MS Windows 2000 Server instalado, con “Active Directory” implementado. Partiendo de esta base se explicará como configurar todos los servicios de red para implementar una red inalámbrica segura.

3.1 – Active Directory

A modo de resumen, y para un mejor entendimiento del sistema, se puede comentar qué es y para que sirve el “Active Directory”. Para entender el Active Directory de Microsoft Windows hay que comenzar por conocer qué es un servicio de directorio.

Un directorio es una fuente de información que se usa para almacenar información acerca de objetos. Un directorio telefónico almacena información sobre suscripciones de teléfono. En un sistema de archivos, el directorio almacena información acerca de archivos. Un servicio de directorio difiere de un directorio en que es la fuente de información de directorios y los servicios que hacen que la información esté disponible y al alcance de los usuarios.

Un servicio de directorio es uno de los componentes más importantes en un sistema distribuido. Los usuarios y administradores de una red, con frecuencia no saben el nombre exacto de los objetos en que están interesados, quizá solo conozcan uno o más atributos de los objetos que buscan. El servicio de directorio proporciona la funcionalidad de buscar objetos que concuerden con uno o más atributos. Un objeto es un conjunto nombrado y distinguible de atributos que representa a algo concreto, por ejemplo, un usuario, una impresora o una aplicación.

AD es el servicio de directorio incluido con el sistema operativo Windows 2000 Server. Amplia las características de los anteriores servicios de directorio basados en Windows, y agrega características completamente nuevas. AD es seguro, distribuido, particionado y replicado. Está diseñado para funcionar perfectamente en instalaciones de cualquier tamaño, desde sólo un servidor con algunos objetos, hasta miles de servidores y millones de objetos.

3.1.1 – Sitio

La estructura de red física de Active Directory se basa sobre una unidad conocida como sitio. Un sitio consta de una o más subredes con Protocolo Internet (IP) unidas entre sí mediante conexiones fiables de alta velocidad. Se deben establecer los sitios en base a una premisa: si bien muchas subredes pueden pertenecer a un único sitio, una única subred no puede abarcar varios sitios. La importancia del sitio es asegurar una transmisión de datos rápida y económica, especialmente en lo que se refiere a una replicación eficiente de los servicios de directorio. Es importante destacar que no

existe ninguna relación formal entre los límites de un sitio o un dominio. Un sitio puede tener varios dominios y un dominio puede incluir varios sitios (figura 3.1). Además, los sitios y los dominios no tienen que mantener el mismo espacio de nombres.

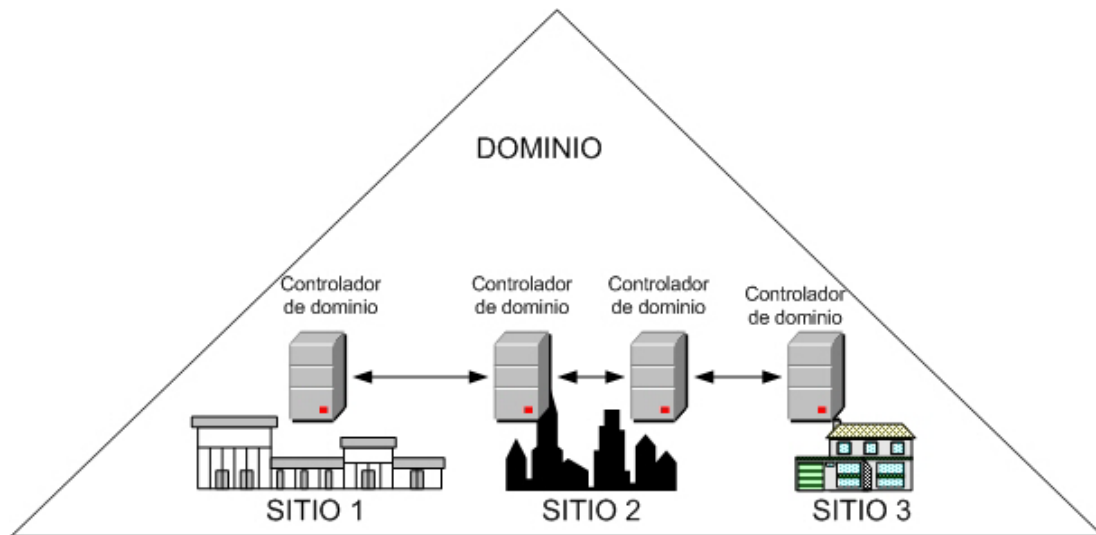


Figura 3.1: Un dominio en varios sitios.

3.1.2 – Controlador de dominio

Un controlador de dominio es un servidor que contiene una copia de Active Directory, todos los controladores de dominio son homólogos y mantienen versiones replicadas de Active Directory para el dominio. El controlador de dominio organiza todos los datos de objetos del dominio en un almacén de datos lógico y jerárquico, también autentica usuarios, proporciona respuestas a consultas acerca de objetos de la red y realiza la replicación de los servicios de directorio. La estructura física ofrece el medio para transmitir estos datos a través de sitios bien conectados.

Active Directory reemplaza el mecanismo utilizado en Windows NT como controlador principal de dominio (PDC) y sus correspondientes controladores de reserva. Ahora todos los controladores de dominio comparten una relación entre iguales con múltiples maestros que alojan copias de Active Directory. Otra gran

diferencia con respecto a Windows NT es que todos los controladores de dominio de Windows 2000 tienen capacidad de lectura y escritura en Active Directory. En las versiones anteriores, sólo el PDC tenía capacidad de lectura y escritura e iniciaba la replicación. Cualquier controlador de dominio de Active Directory puede iniciar el proceso de replicación cuando se agregan nuevos datos.

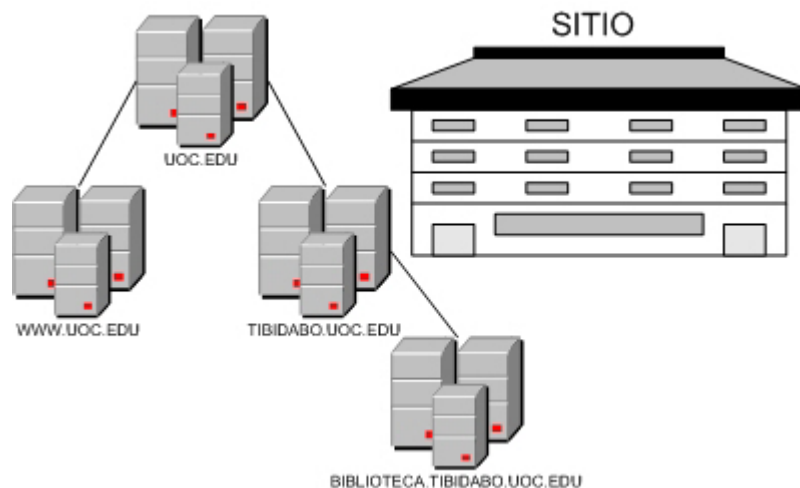


Figura 3.2: Varios dominios en un sitio.

Un dominio de Active Directory puede tener uno o más controladores de dominio que proporcionen replicación de la partición del directorio (figura 3.2).

3.2 – DHCP

Una vez tenemos una máquina con MS Windows 2000 Server y Active Directory hay que comenzar a instalar y configurar todos los servicios de red necesarios para implementar la red inalámbrica. Se recomienda comenzar instalando y configurando un servidor DHCP (*DynamicHost Configuration Protocol*).

DHCP es un protocolo utilizado para proporcionar toda la configuración de red a los clientes que se la soliciten (dirección IP, máscara de red, puerta de enlace, etc). Cada

cliente inalámbrico que se conecte a la red recibirá del servidor DHCP toda la configuración de red.

Si en la instalación del sistema operativo no se incluyó este servicio, habrá que instalarlo, para ello hay que dirigirse al “Asistente para componentes de Windows” y añadir el servicio “Protocolo de configuración dinámica de equipos (DHCP)” (figura 3.3).

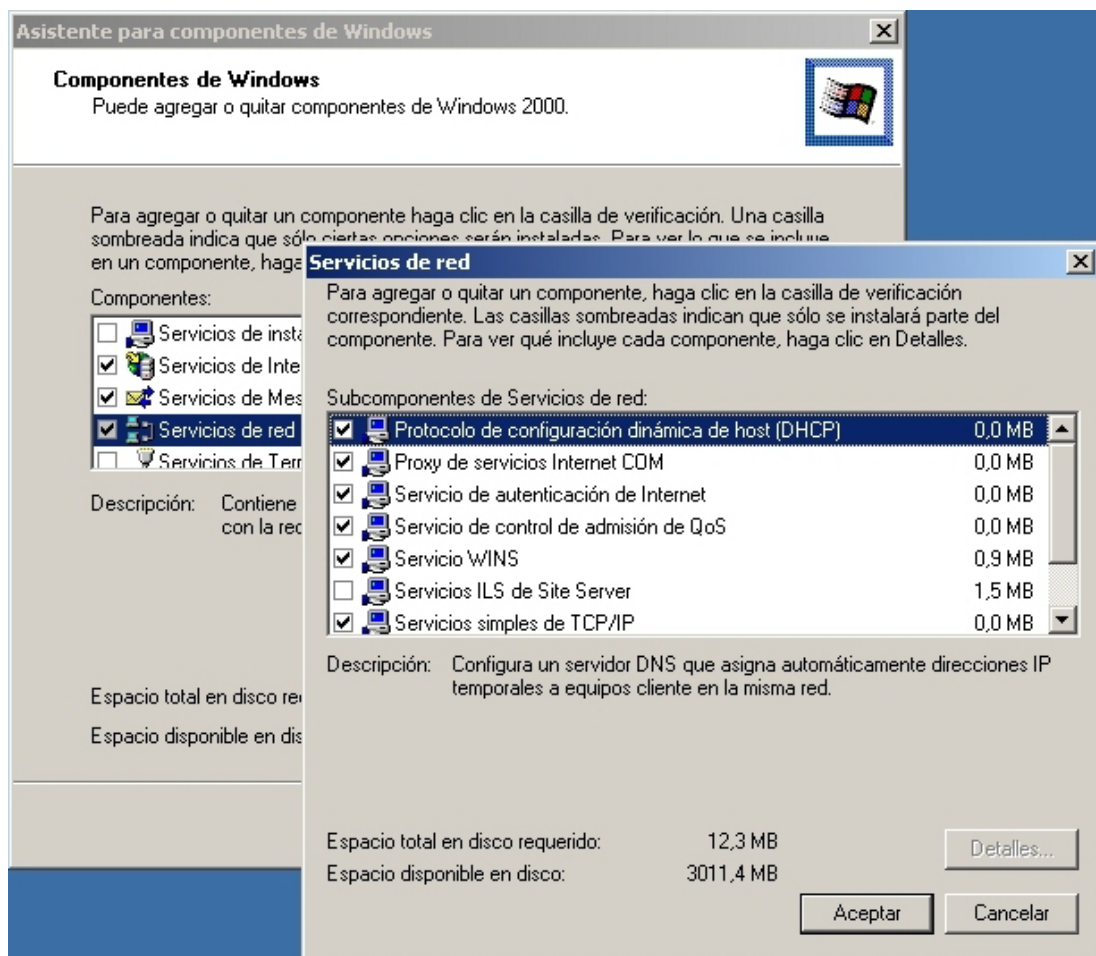


Figura 3.3: Instalación servidor DHCP.

Una vez instalado el servicio, hay que configurarlo. En el Anexo 1 se muestra como se debe configurar.

3.3 – Configurar Active Directory

El siguiente paso es registrar en el AD todos aquellos usuarios que se conectarán al dominio a través de la red inalámbrica. Antes se debe crear un nuevo grupo que permita englobar y tratar a los usuarios inalámbricos como un conjunto.

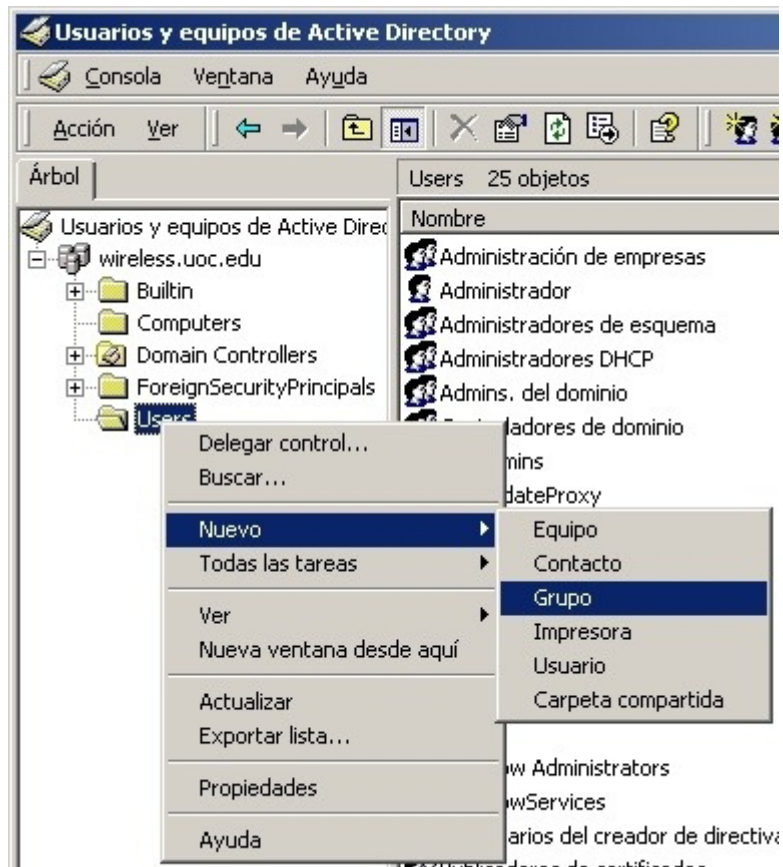


Figura 3.4: Crear nuevo grupo en AD.

Para realizar todas estas operaciones se debe acceder a las “Herramientas administrativas” y seleccionar el “Usuarios y equipos de Active Directory”. Dentro de la aplicación abierta hay que seleccionar la carpeta “Users” y con el botón derecho del ratón seleccionar la opción Nuevo Grupo (figura 3.4).

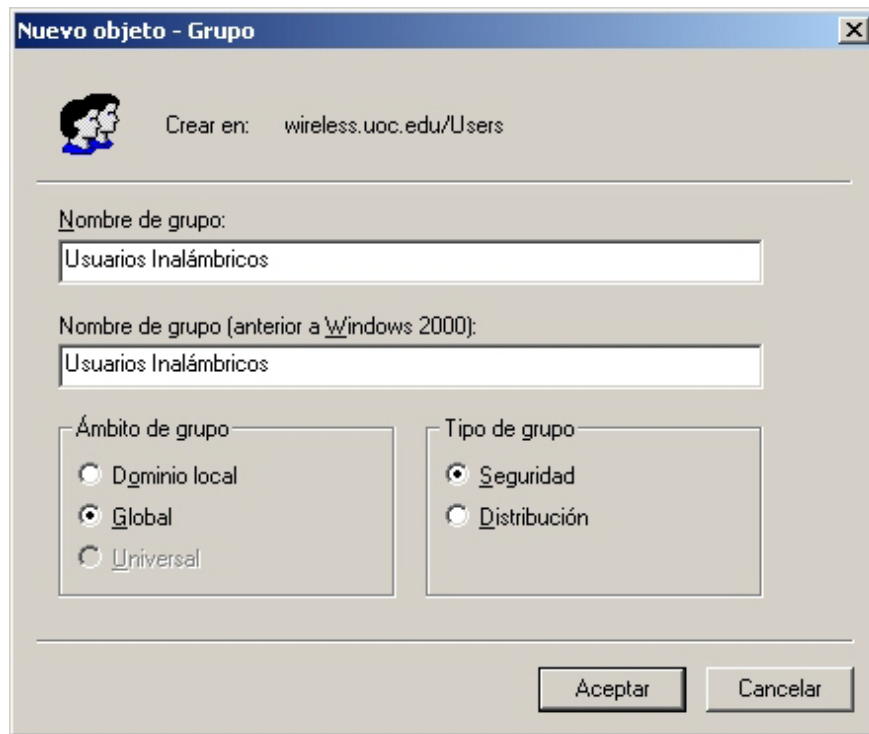


Figura 3.5: Definir nuevo grupo en AD.

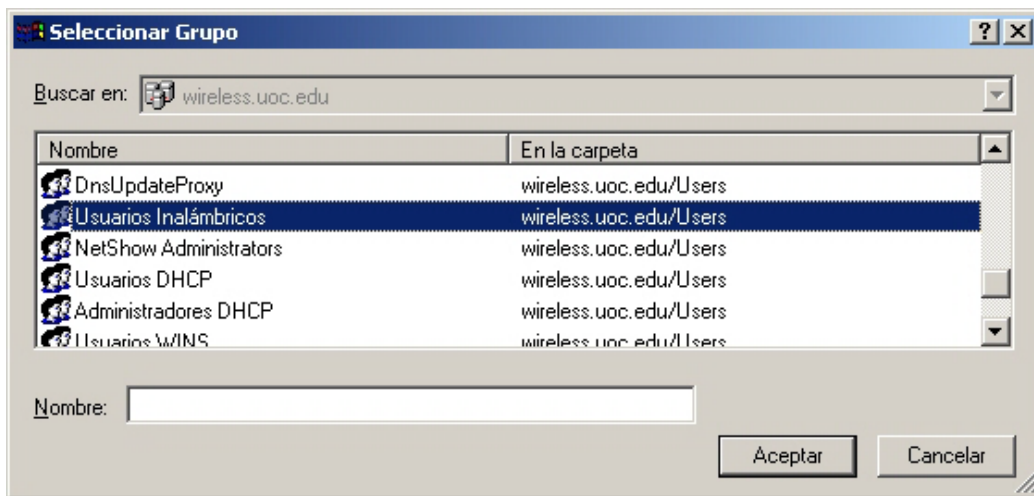


Figura 3.6: Añadir un usuario a un grupo en AD.

La aplicación nos mostrará una ventana (figura 3.5) donde se deben indicar los datos que identificarán al grupo. Se debe indicar el nombre del nuevo grupo, como siempre, es aconsejable asignar un nombre auto informativo, en nuestro caso se ha

escogido como nombre de grupo “Usuarios Inalámbricos”. Debemos indicar también que el ámbito del nuevo grupo a crear será “Global”, y del tipo “Seguridad”.

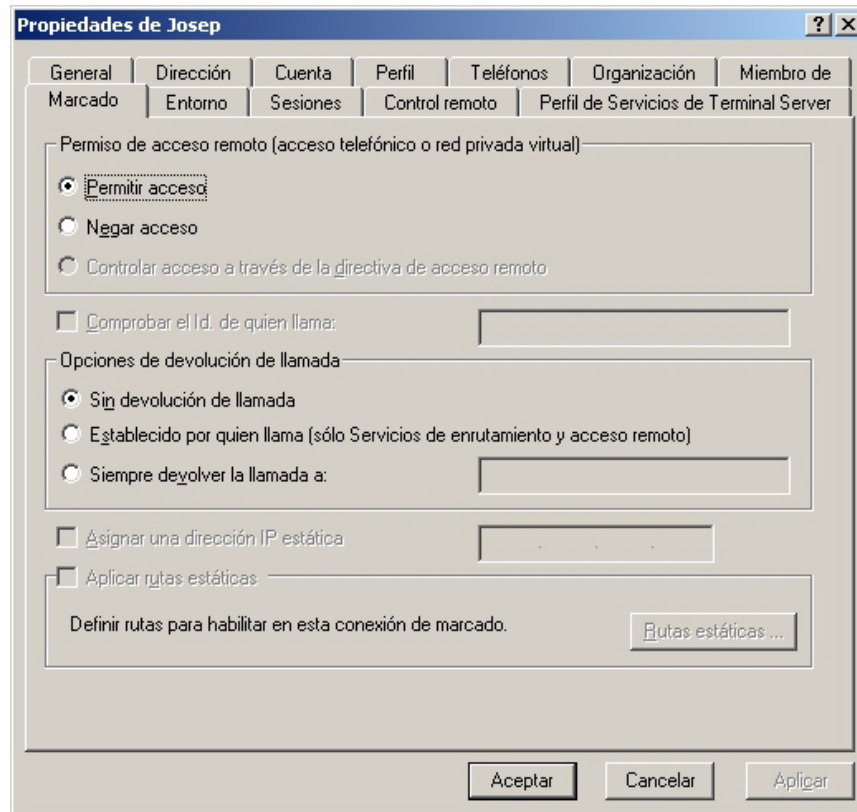


Figura 3.7: Cambiar propiedades de un usuario.

Utilizando el mismo método que para crear un Grupo, se debe crear un nuevo usuario escogiendo en esta ocasión la opción “Usuario” del menú contextual que aparecía al crear un nuevo grupo. Cuando aparezca una ventana solicitando los datos del nuevo usuarios se rellenan los campos con los datos del usuario que queremos crear (Nombre, Apellidos, contraseña, etc.). Una vez creado el usuario hay que añadirlo al grupo que hemos creado anteriormente, para ello se selecciona el usuario y pulsando el botón derecho del ratón seleccionamos la opción “Agregar miembros a un grupo...”; del listado de grupos que nos muestra el sistema, debemos escoger el que habíamos creado anteriormente, pulsando “Aceptar” para finalizar la operación (figura 3.6).

A continuación hay que darle permiso al usuario para que pueda realizar una conexión, escogemos la opción “Propiedades” en el menú contextual que aparece pulsando el botón derecho del ratón sobre el usuario. En la pestaña “Marcado” deberemos seleccionar las opciones “Permitir acceso” y “Sin devolución de llamada” (figura 3.7), es muy importante que estas opciones estén seleccionadas.

3.4 – Servidor de Certificados

Para poder utilizar métodos de autenticación tipo EAP-TLS entre el servidor y los clientes, es necesario que tanto el usuario como el servidor posean un certificado digital, para conseguirlos sin tener que solicitárselos a una entidad externa, hay que instalar el servidor de certificados de Windows 2000 Server. Para instalarlo hay que acceder al “Asistente para componentes de Windows” y seleccionar el componente “Servicios de Certificate Server” (figura 3.8).

Al pulsar el botón “Siguiente”, el sistema informará que a partir de ese momento no se podrá cambiar el nombre al servidor, si aceptamos el mensaje nos aparecerá una ventana donde el sistema pregunta que tipo de entidad certificadora queremos crear (figura 3.9), se debe seleccionar la opción “Entidad emisora raíz de la empresa”.

A continuación se deberán proporcionar todos los datos de la nueva entidad certificadora raíz tal y como se muestra en la figura 3.10, pero con los datos que correspondan.

Dado que el servidor de certificados interactúa con el servidor web, el sistema preguntará si puede parar dicho servidor para realizar los cambios que necesita, la respuesta debe ser afirmativa. Ya tenemos una entidad certificadora instalada en nuestro servidor.

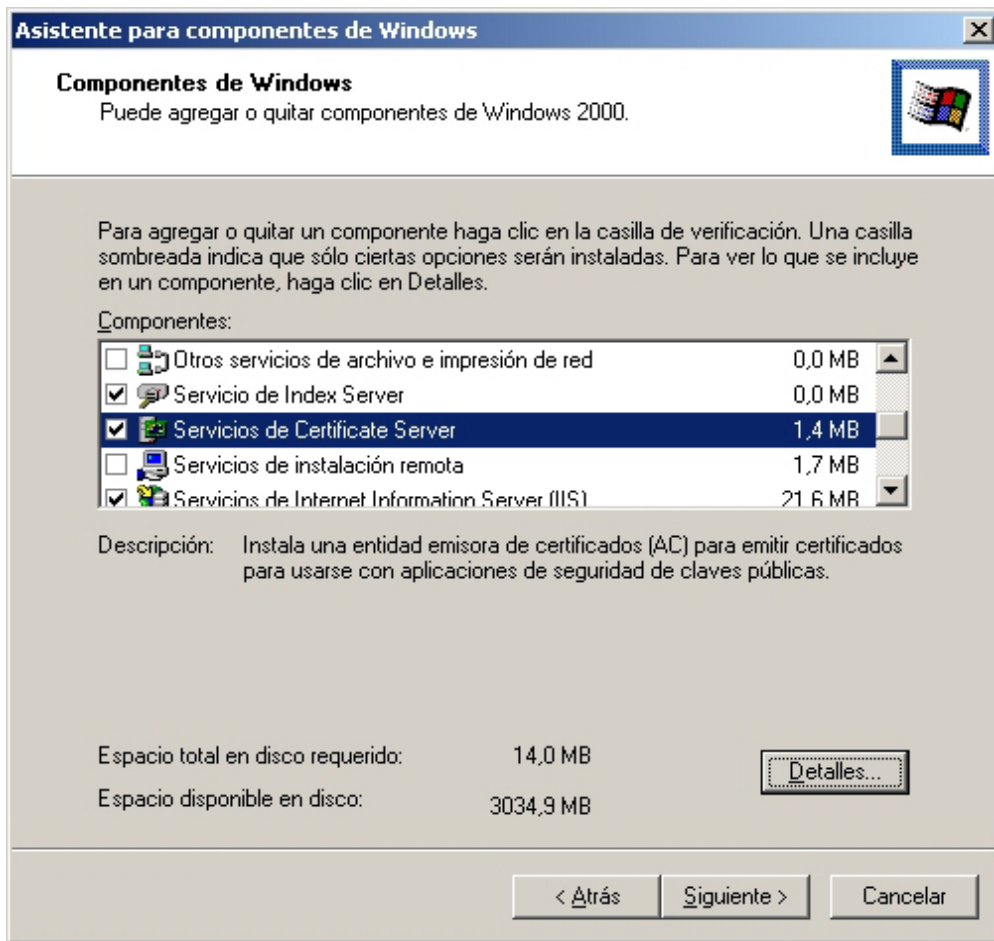


Figura 3.8: Instalar el Servidor de Certificados.

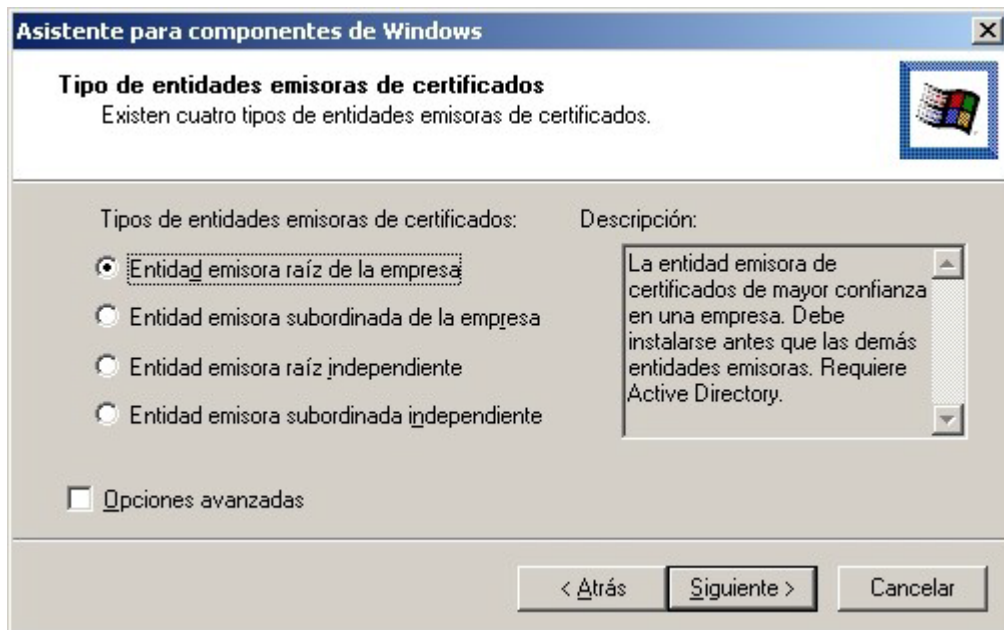


Figura 3.9: Definir el tipo entidad emisora de certificados.

Para comprobar que el servicio está correctamente instalado podemos acceder a la dirección: **http://<NombreNuestroServidor>/certsrv**, el servidor solicitará un nombre de usuario y contraseña del dominio (podemos utilizar el que habíamos creado anteriormente), y nos deberá aparecer una página como la mostrada en la figura 3.11.

The screenshot shows a window titled "Asistente para componentes de Windows" with the subtitle "Identificación de la entidad emisora de certificados". Below the subtitle is the instruction "Escriba la información para identificar esta entidad emisora de certificados". The form contains the following fields:

- Nombre de entidad emisora: UOC CA
- Organización: UOC
- Unidad organizativa: Proyecto FC
- Ciudad: Barcelona
- Estado o provincia: Barcelona País o región: ES
- Correo electrónico: (empty)
- Descripción de la entidad emisora: (empty)
- Válido durante: 2 Años Caduca: 05/05/2006 18:03

At the bottom of the window are three buttons: "< Atrás", "Siguiete >", and "Cancelar".

Figura 3.10: Datos entidad emisora de certificados raíz.

Desde esta pantalla se pueden solicitar certificados de usuario, necesarios para poder conectarse al servidor utilizando la autenticación EAP-TLS. Con este tipo de autenticación, el propio servidor necesita un certificado así que se puede aprovechar la conexión que hemos realizado de prueba para solicitar el certificado del servidor.

Ahora es necesario crear una nueva plantilla de certificado de seguridad para los usuarios inalámbricos, se accede a "Herramientas Administrativas" y se selecciona "Entidad emisora de certificados". Dentro del árbol de nuestra entidad certificadora se debe crear un nuevo "Certificado para emitir" en el apartado "Configuración de

directivas”, tal y como muestra la figura 3.12. En el listado que muestra el sistema se debe seleccionar “Sesión autenticada” (figura 3.13).



Figura 3.11: Pantalla para solicitar un nuevo certificado de usuario.

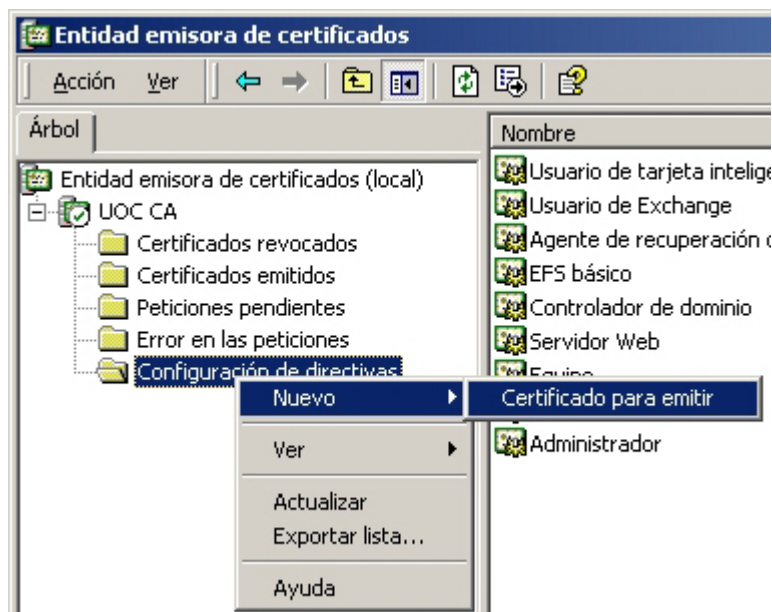


Figura 3.12: Crear nueva plantilla de certificados.

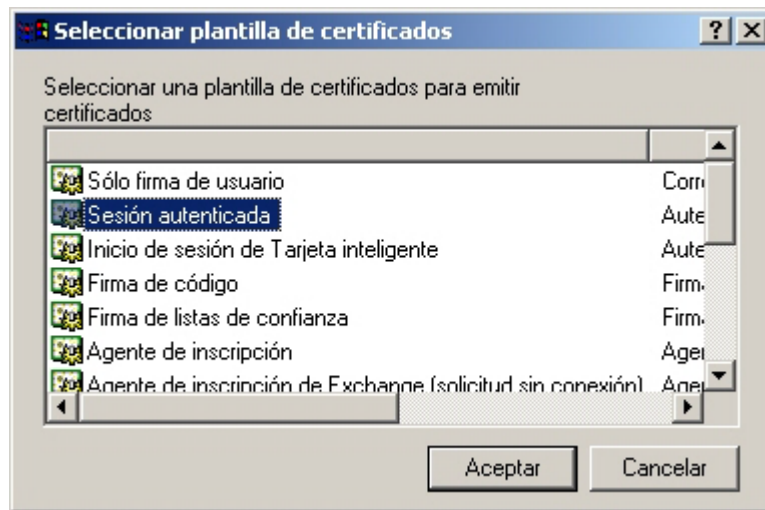


Figura 3.13: Selección de la plantilla “Sesión autenticada”.

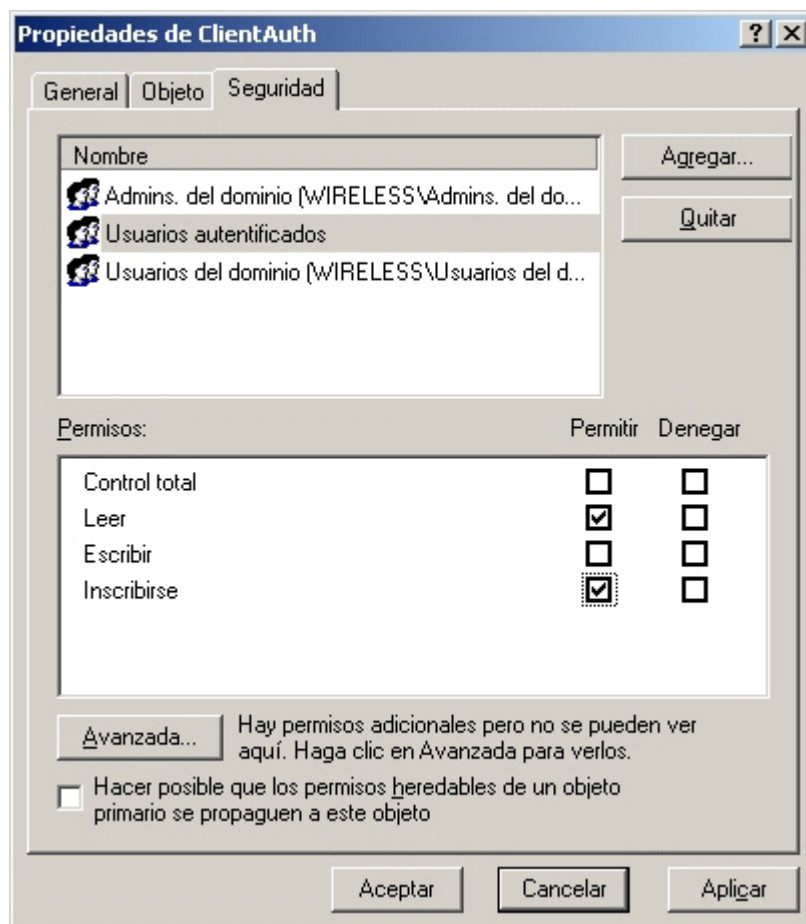


Figura 3.14: Modificar propiedades de las plantillas de certificado.

A continuación, en la aplicación “Sitios y servicios de AD” (“Herramientas Administrativas” → “Sitios y servicios de Active Directory”), desde el menú “Ver” seleccionamos la opción “Mostrar el nodo de servicios”. Nos situamos en el nodo “Services” → “Public Key Services” → “Certificate Templates” y en la ventana derecha aparecerá un listado de todas las plantillas soportadas por el sistema. Hay que modificar las propiedades de dos de ellas: “ClientAuth” y “User”. Al acceder a las propiedades de ambas, dentro de la pestaña Seguridad hay que marcar la opción permitir “Inscribirse” para el grupo “Usuarios autenticados” (figura 3.14).

3.5 – IAS (*Internet Authentication Service*)

Como ya se explicó en el apartado RADIUS, un punto de acceso hace las funciones de cliente de un servidor RADIUS. Si el punto de acceso se configura para que utilice un servidor RADIUS, le enviará todas las peticiones de conexión y será el servidor el que permita o deniegue el acceso al nuevo cliente, es por ello que hay que definir en el servidor IAS un cliente que será el punto de acceso.

Windows 2000 Server tiene un servidor RADIUS llamado IAS, para instalarlo hay que seguir el mismo procedimiento que para instalar los servicios de red instalados anteriormente. En el momento de la instalación, el sistema no realizará ninguna pregunta. Una vez instalado hay que configurarlo, para ello se accede como siempre al menú “Herramientas administrativas” y seleccionamos “Servicio de Autenticación de Internet”. Lo primero que hay que hacer es registrar el servicio en el Active Directory (figura 3.15), igual que se hizo con el servidor DHCP, para realizar la operación hay que pulsar el botón derecho del ratón sobre el servidor y seleccionar “Registrar servicio en Active Directory”.

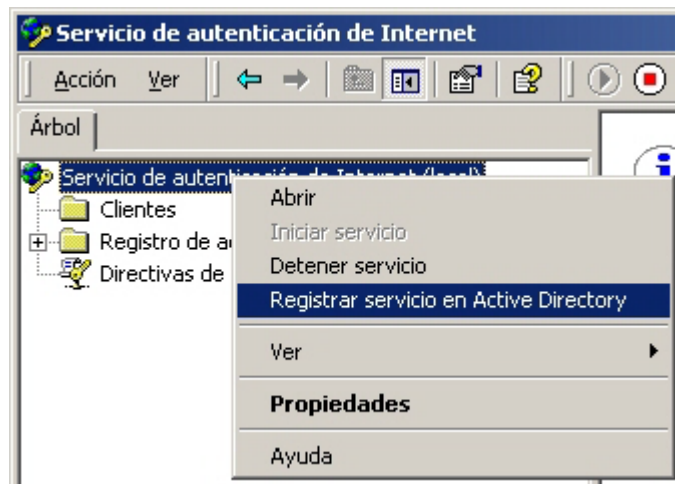


Figura 3.15: Registrar IAS en AD.

Una vez registrado, se debe crear el cliente del servicio IAS, es decir, el punto de acceso. Si nos situamos en la carpeta “Clientes” y pulsamos el botón derecho del ratón podemos seleccionar la opción “Nuevo cliente”.

Ahora se le debe asignar un nombre identificativo al cliente, en nuestro caso el nombre asignado es “AP_UOC”, en la siguiente ventana debemos indicar el nombre de red o dirección IP del punto de acceso, el protocolo utilizado que será “RADIUS Standard”, la clave compartida que servirá para cifrar toda comunicación entre el punto de acceso y el servidor IAS, y por último, se marca la casilla “El cliente debe enviar siempre el atributo de firma en la solicitud”, como se muestra en la figura 3.16.

Ahora hay que crear una directiva de acceso remoto que definirá los requisitos que debe cumplir un cliente para poder establecer la conexión; si se abre el menú contextual en el elemento “Directivas de acceso remoto” podemos seleccionar crear una nueva directiva. Como siempre, hay que asignarle un nombre, en nuestro caso “Acceso Inalámbrico”, a continuación se deben indicar las condiciones que debe cumplir un acceso para que se le permita la conexión, hay que añadir el atributo “Windows-Groups” dándole como valor el grupo de usuarios que habíamos creado inicialmente: “Usuarios Inalámbricos”, de esta forma le estamos indicando al

servidor que solo debe permitir el acceso a los clientes que pertenecen al grupo de usuarios “Usuarios Inalámbricos”.

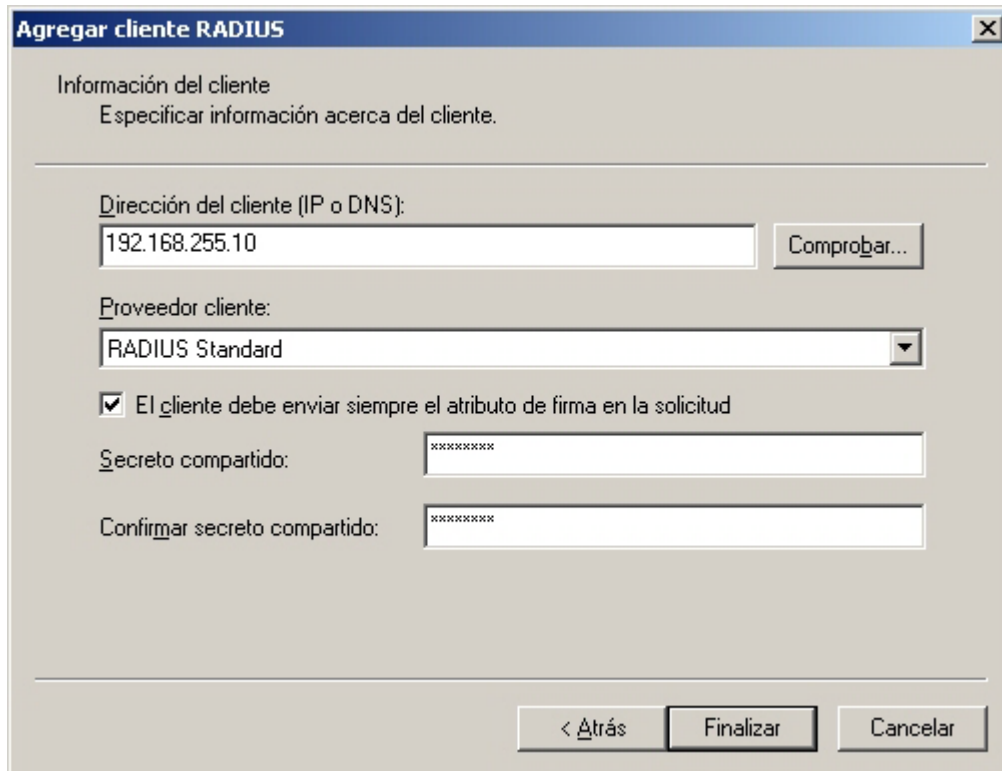


Figura 3.16: Agregar cliente al servidor IAS.

En la siguiente ventana, hay que indicar que se concede permiso de acceso remoto a este grupo de usuarios. A continuación, el sistema solicita si se desea modificar el perfil, hay que decir que sí, y en la nueva ventana seleccionar la pestaña “Autenticación”, en ella se seleccionará como único método de autenticación el protocolo EAP con “Tarjeta inteligente u otro certificado”, de esta forma forzamos a que las únicas conexiones permitidas se realizarán siempre que exista una autenticación EAP (figura 3.17).

Una vez aceptemos los parámetros insertados, ya tenemos una nueva directiva de acceso remoto definida. Por defecto el sistema define una directiva que hay que borrar, solo hay que seleccionarla y pulsar la tecla suprimir. La directiva que acabamos de crear debe ser la única que exista en el servidor IAS.

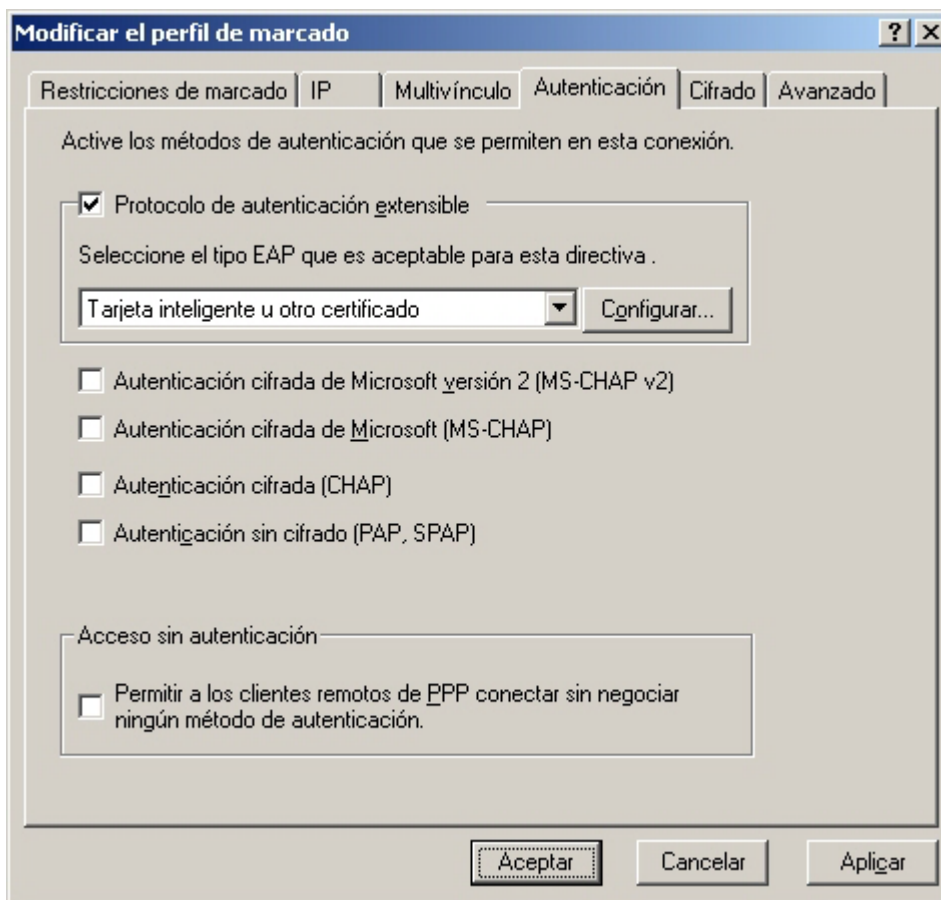


Figura 3.17: Definir tipo de autenticación utilizada en el acceso.

3.6 – Configurar el punto de acceso

El siguiente paso es configurar el punto de acceso, como es lógico pensar, dependiendo del AP variará su configuración, pero básicamente hay que indicarle las propiedades de red y de la seguridad. Aquí se mostrará como configurar el punto de acceso “Buffalo WBS-G54A-CB-2”.

El primer paso es asignarle una dirección IP al punto de acceso, que debe ser la misma que se le había indicado en el servicio IAS cuando se dio de alta al AP.

LAN side Ethernet settings

LAN side IP address ?	IP address:	<input type="text" value="192.168.255.10"/>
	Subnet mask:	<input type="text" value="255.255.255.0"/>

Note:

- After changing LAN side IP address, the current setup process cannot be continued. For the AirStation setup, change the PC's IP address and restart the configuration from the utility software.

DHCP server function

DHCP server function ?	<input type="radio"/> Use <input checked="" type="radio"/> Do not use	
Assigned IP address ?	From	<input type="text" value="192.168.255.20"/> total of
		<input type="text" value="9"/> PCs

Figura 3.18: Configuración de red del AP.

Como se puede apreciar en la figura 3.18, este AP tiene su propio servidor DHCP que hay que deshabilitar, ya tenemos un servidor DHCP en la red más fiable y configurable que el que tiene el punto de acceso.

Ahora hay que configurar el apartado de seguridad, en nuestro caso y para obtener un nivel máximo y en consonancia con la configuración establecida en el servidor, se habilitará el método WPA-TKIP para cifrar los datos entre el cliente y el punto de acceso. Siguiendo con la seguridad, se habilitará el protocolo 802.1x con autenticación EAP. Se debe configurar el servidor RADIUS al que enviará todas las peticiones de autenticación el punto de acceso, debe ser la dirección IP de nuestro servidor IAS; el puerto utilizado para la conexión del protocolo RADIUS, si no se ha modificado en el servidor, es el 1812 (figura 3.19). La clave compartida debe ser la misma que se puso en el servidor IAS cuando se dio de alta el punto de acceso.

Con esto acaba la configuración del punto de acceso, ahora solo queda configurar el cliente inalámbrico.

Wireless LAN security settings

Broadcast SSID Allow Deny

Data encryption Disabled

WEP WEP key

TKIP WPA-PSK (Pre-Shared key) (8 to 63 characters ASCII / 64 digits hexadecimal)
* If IEEE802.1x/EAP authentication is used, keep this field blank.

AES WPA Group Rekey Interval Sec

IEEE802.1x/EAP authentication (WPA) Do not authorize Authorize

RADIUS Authentication

RADIUS Server

RADIUS Port

RADIUS Key

Figura 3.19: Definición de la seguridad en el AP.

3.7 – Configurar cliente inalámbrico

Si el cliente tiene sistema operativo Windows XP SP1 solo habrá que asegurarse que tiene todos los parches de seguridad proporcionados por Microsoft instalados, ya que este sistema operativo es compatible con el protocolo WPA utilizado; si el sistema operativo fuese otro habrá que localizar las actualizaciones correspondientes que habiliten el sistema para poder utilizar el nuevo protocolo, Microsoft proporciona actualizaciones para las versiones 2000 de su sistema operativo.

Hay que asegurarse que el dispositivo inalámbrico instalado en el cliente soporta el protocolo WPA; si no fuese así, hay que actualizar el firmware del dispositivo (siempre y cuando el fabricante lo facilite), si no es posible actualizarlo, será imposible poderse conectar al sistema utilizando el protocolo WPA.

Una vez el cliente se encuentra en condiciones óptimas para poderse conectar al sistema, debemos instalarle un certificado de usuario para que se pueda validar ante el servidor, para ello bastará con solicitar un nuevo certificado a la entidad servidora de certificados que hay en el servidor, el método más sencillo consistirá en que el cliente haga una conexión con nuestro servidor web a la dirección **http://<NombreDelServidor>/certsrv**, lógicamente, deberá realizarla por medios físicos, sin utilizar el acceso inalámbrico.

Cuando se accede a la dirección indicada a través de un explorador web, aparecerá una pantalla como la mostrada en la figura 3.11, se debe seleccionar la opción “Seleccionar un certificado” y pulsar el botón “Siguiente” cargándose en el explorador la página mostrada en la figura 3.20. Aunque se puede escoger la opción “Solicitud avanzada”, si se escoge la que aparece por defecto “Solicitud de certificado de usuario” el proceso también es correcto.

En la siguiente página, el sistema informa que ya ha recogido toda la información necesaria y si se desea se puede enviar la solicitud al servidor, aunque da la posibilidad de acceder de nuevo a una definición experta de petición de certificado, al igual que antes, no es necesario así que ya se puede solicitar el certificado.

Tras unos mensajes del servidor, aparecerá una nueva página desde la que podremos instalar directamente el certificado generado por la entidad certificadora, tal y como se muestra en la figura 3.21.

Cuando pulsemos el link “Instalar este certificado” que aparece en la página, el sistema realizará una serie de preguntas sobre la instalación del mismo, se debe responder afirmativamente a todas ellas. Una vez finalizado el proceso, el cliente ya tendrá un certificado digital que le permitirá realizar conexiones inalámbricas con el servidor.

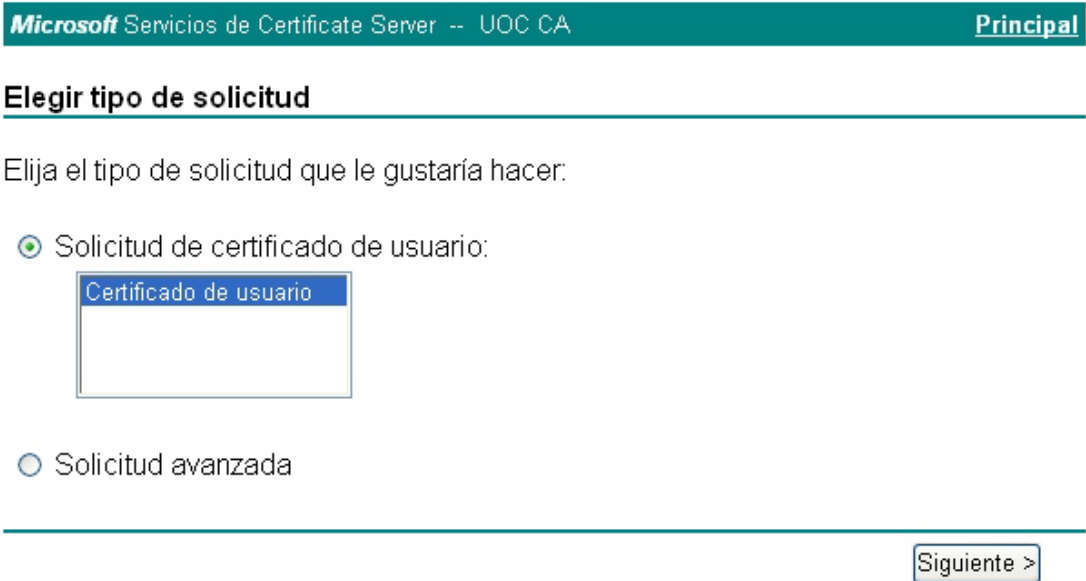


Figura 3.20: Elección del tipo de solicitud de certificado.

Por último, solo queda configurar la conexión de red que permita conectar el cliente con el servidor. Si el sistema operativo del cliente es Windows XP SP1 la conexión puede haber obtenido toda la configuración del propio punto de acceso y se autoconfigura, si no es así o si por cualquier motivo no se ha autoconfigurado habrá que seguir una serie de pasos para configurarla, tal y como se muestra en el Anexo 2.

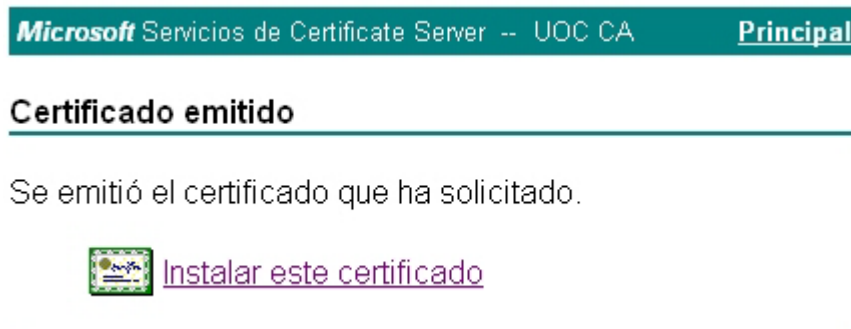


Figura 3.21: Entrega del certificado solicitado.

3.8 – Conclusión

En este capítulo se han visto todos los pasos a seguir para implementar una red inalámbrica segura utilizando exclusivamente los protocolos y servicios que ofrecen los sistemas operativos de MS Windows (Windows XP en el cliente y Windows 2000 Server en el servidor).

Se ha utilizado el protocolo WPA-TKIP para cifrar los datos entre el cliente y el punto de acceso. La autenticación del cliente se ha realizado mediante un certificado digital que es validado por el servidor utilizando para ello el protocolo EAP-TLS.

Se puede decir que se ha implementado una red en la que utilizando los estándares actuales más avanzados, se ha conseguido que las comunicaciones sean lo más seguras posible. En lo que respecta al acceso, se puede mejorar la seguridad utilizando cortafuegos y VPNs, pero esto se verá en el capítulo 5.

Capítulo 4

WLAN segura con LINUX

En este capítulo se explica como implementar una red inalámbrica segura utilizando una plataforma con sistema operativo LINUX. Los servicios necesarios son los mismos que en el montaje con Microsoft Windows 2000 Server, la diferencia estriba en el nombre y en la forma de instalarlos y configurarlos, por lo que en este capítulo no se repetirán las explicaciones de las funcionalidades de cada servicio comentadas en el capítulo anterior. También se da por hecho que el lector de este documento conoce los métodos y parámetros que se utilizan en LINUX para realizar funciones típicas de mantenimiento del sistema operativo (desempaquetado y compilación de código, sistema de arranque y parada automática de servicios, etc.), por lo que no se explicará con detalle por qué se utilizan ciertos comandos, ni qué significa cada uno de los parámetros utilizados, solo se indicará cuando y como hay que ejecutarlos.

Los requerimientos hardware necesarios son:

- Máquina con sistema operativo LINUX. Todo el montaje mostrado en este capítulo ha sido realizado en una máquina con sistema operativo LINUX Fedora Core 1, con kernel 2.4.22-1.2188.nptlsm
- Punto de acceso.
- Cliente con un dispositivo de red inalámbrico.

4.1 – DHCP

El servidor DHCP que se ha instalado es la versión `dhcp-3.0.1rc12-6.i386` de ISC (*Internet Software Consortium*) en su formato RPM (*Redhat Package Manager*).

Este paquete, al igual que otros muchos, puede ser descargado de cualquier repositorio de RPM's; concretamente, éste paquete ha sido descargado del servidor:

<http://download.fedora.redhat.com/pub/fedora/linux/core/development/i386/Fedora/RPMS/>

La instalación se realiza ejecutando el comando:

```
rpm -ihv dhcp-3.0.1rc12-6.i386.rpm
```

Una vez instalado el software en la máquina, hay que configurarlo. El primer paso consiste en modificar el archivo `/etc/dhcpd.conf` (si no existe, se crear uno nuevo), el cual contiene toda la configuración básica del servidor DHCP. Se debe eliminar o comentar todo el texto del archivo y añadir la información que se muestra en el Anexo 3; lógicamente, la configuración mostrada es la que se utiliza en este proyecto, pero cada implementación particular tendrá sus valores correspondientes.

El servicio DHCP no requiere ninguna otra configuración, así que lo único que falta es configurar el sistema operativo para que cuando se inicie en los niveles 3 y 5 de ejecución, se lance el proceso `/usr/sbin/dhcpd` que corresponde al ejecutable que inicia el servicio DHCP.

Cuando se ejecuta el servicio, se crea de forma automática el archivo `/var/lib/dhcp/dhcpd.leases` que contiene todas las concesiones de configuración realizadas por el servidor.

4.2 – DNS

El servidor DNS (*Domain Name Service*) utilizado es el que proporciona el propio sistema operativo Fedora Core 1. Su configuración es muy parecida a la del servidor

DHCP, se basa en la modificación de una serie de archivos. Se comienza modificando el archivo principal de configuración `/etc/named.conf`, eliminando o comentando las opciones que hay por defecto, y añadiendo las que se muestran en el Anexo 4.

Es importante, por temas de seguridad, que exclusivamente el usuario "named" tenga acceso de lectura y escritura al archivo `/etc/rndc.key`, y al directorio `/var/named` en el que se encuentran los archivos de definición de zonas. Para asegurar esta restricción de acceso hay que cambiar los permisos del archivo y directorio con el siguiente comando:

```
chown named:named /etc/rndc.key /var/named -R
```

Ahora hay que crear los archivos de definición de cada zona. El archivo `/var/named/named.root` (ver Anexo 5) es creado por el propio servidor, y contiene un listado con los principales servidores DNS de Internet. Cuando nuestro servidor DNS recibe una petición de resolución de nombre o de IP, y ésta no coincide con ningún registro definido en sus zonas de búsqueda, reenvía la petición a los servidores de Internet para que la resuelvan.

Los dos archivos de zonas que se han definido en la configuración del servidor DNS (búsqueda directa e inversa) se deben almacenar en el directorio `/var/named`, y sus nombres deben coincidir con los indicados en el archivo de configuración. Ambos archivos contendrán registros con los nombres de las máquinas del dominio y sus respectivas direcciones IP, conjuntamente con una serie de parámetros globales de la zona que definirán su comportamiento. Ambos archivos se pueden ver en el Anexo 5.

Ahora hay que configurar el servicio `/etc/named` para que se ejecute de forma automática cuando se inicie el sistema operativo en los niveles 3 y 5 de ejecución.

4.3 – OpenSSL

OpenSSL es un paquete software que proporciona una serie de librerías y programas que permiten crear una Infraestructura de Clave Pública (*PKI - Public Key Infrastructure*), es decir, podemos crear entidades certificadoras (CA), certificados de usuario, etc. Es necesario instalar este paquete software para poder cifrar la comunicación con el servidor OpenLDAP y para poder habilitar la autenticación de usuarios vía EAP-TLS y EAP-TTLS. La versión utilizada ha sido la **openssl-0.9.7d**, que puede descargarse desde el servidor:

<http://www.openssl.org/source/>

Para realizar la instalación hay que situarse en el directorio en el que se ha descargado el archivo y ejecutar los siguientes comandos:

```
tar zxvf openssl-0.9.7d.tar.gz
cd openssl-0.9.7d
./config --prefix=/usr/local/ --openssldir=/usr/local/openssl
make
make install
```

Los programas y las librerías ya están instaladas y preparadas para poderlas utilizar. Ahora puede ser un buen momento para crear una Infraestructura de Clave Pública ya que más adelante necesitaremos una entidad certificadora y los certificados de los clientes y del servidor para poder securizar las comunicaciones. Es aconsejable comenzar modificando el archivo de configuración `openssl.cnf` del software OpenSSL para adaptarlo a nuestras necesidades, en el Anexo 6 se muestra el archivo utilizado con los valores por defecto modificados para facilitar la creación de los certificados.

Para que los certificados creados con OpenSSL puedan ser reconocidos y utilizados en equipos con sistemas operativos Windows, es necesario crear un archivo con unos valores que serán utilizados en el momento de crear los certificados. El nombre del

archivo puede ser el que deseemos, siempre y cuando utilicemos el mismo nombre que hemos asignado al archivo en el momento de crear los certificados. Para identificar el archivo de forma fácil, se aconseja llamarlo “xpextensions”. A continuación se muestra el contenido que debe tener el archivo.

```
[ xpclient_ext]
extendedKeyUsage = 1.3.6.1.5.5.7.3.2
[ xpserver_ext ]
extendedKeyUsage = 1.3.6.1.5.5.7.3.1
```

Dado que lo normal es crear varios certificados de usuarios (uno por cada usuario que realice una conexión con el servidor), la mejor opción es construir unos scripts que permitan la creación automática de certificados. Aunque la entidad certificadora y el certificado del servidor solo se deben crear una única vez, también puede ser aconsejable preparar un script para crearlos. En el Anexo 7 se muestran los tres scripts: `CA.root`, `CA.svr` y `CA.clt` que permitirán crear la entidad certificadora, certificado del servidor y certificados de clientes, respectivamente. Es importante que durante la creación del certificado del servidor, cuando el script pregunta el “Common Name” se introduzca el nombre de la máquina en la que se instalará el certificado; igualmente, cuando se crean los certificados de usuario, el Common Name debe coincidir con el nombre de usuario definido posteriormente en la aplicación FreeRADIUS.

Una vez finalizada la ejecución de los tres scripts, en el directorio de ejecución tendremos nueve certificados, tres por cada entidad (tres formatos diferentes):

- CA: `root.p12`, `root.pem` y `root.der`
- Servidor: `<Nombre>.p12`, `<Nombre>.pem`, `<Nombre>.der`
- Cliente: `<Nombre>.p12`, `<Nombre>.pem`, `<Nombre>.der`

Como se puede apreciar en los scripts, todos los certificados se crean por comodidad con el mismo password: “secreto”. En una implementación operativa del sistema, los

password deberían cumplir con los mínimos criterios de seguridad exigidos para catalogarlos como seguros (longitud, alfanuméricos, no diccionario, etc.).

A partir de los certificados creados, hay que generar una serie de nuevos archivos que serán necesarios para configurar el servicio LDAP y RADIUS. Por ejemplo, si se desea configurar seguridad TLS en el servidor LDAP, hay que proporcionarle la clave privada del certificado del servidor desprotegida de password, de no ser así, cada vez que se inicie el servicio solicitará la clave de acceso al certificado, lo cual impide su ejecución de forma desatendida.

Para extraer la clave privada de un certificado eliminando el password, se debe ejecutar el siguiente comando del OpenSSL:

```
openssl rsa -in <CertificadoProtegido>.pem -out <ClavePrivada>.key
```

Mantener un archivo con la clave privada sin protección de password es muy inseguro, por lo que hay que asegurarse que exclusivamente el usuario “root” tiene acceso a dicho archivo.

Para extraer exclusivamente el certificado, sin la clave privada, basta con editar el archivo con extensión PEM y copiar en otro archivo la sección de texto que empieza con `-----BEGIN CERTIFICATE-----` y finaliza con `-----END CERTIFICATE-----`, incluyendo ambas líneas.

También será necesario un archivo con los parámetros del algoritmo Diffie-Hellman, el cual se puede generar con el siguiente comando:

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

Por ultimo, se necesitará un archivo con caracteres aleatorios. Existen muchos métodos para crear dicho archivo, aquí se muestra solo una posible forma de crearlo ejecutando el siguiente comando:

```
openssl rand -out random 4096
```

Utilizando los comandos y scripts mostrados anteriormente se obtienen todos los archivos necesarios para implementar seguridad de certificado en los servicios que se implementarán a continuación.

Para finalizar la instalación, hay que instalar en el servidor el certificado de la entidad certificadora y el del propio servidor siguiendo los pasos propios del sistema operativo para instalar certificados. Los archivos a utilizar son `root.der` como entidad certificadora, y el archivo `<NombreServidor>.p12` como certificado del servidor.

4.4 – LDAP

LDAP (*Lightweight Directory Access Protocol*) es el servicio de directorio más extendido y usado por todas las plataformas informáticas. En el apartado 3.1 se explica qué es un servicio de directorio, haciendo hincapié en el “Active Directory” de Windows, que no deja de ser un servicio LDAP adaptado por Microsoft. En el presente punto, se explica como implementar el servicio LDAP en un entorno LINUX.

4.4.1 – DB4 - Berkeley DB

LDAP debe almacenar la información de todos los objetos que componen el directorio en algún tipo de base de datos, en este proyecto se ha decidido utilizar la base de datos DB4 de Sleepycat Software, ya que es la más extendida como soporte de LDAP. La versión instalada es la `db-4.1.25` con soporte de encriptación, que se puede descargar desde el servidor:

<http://www.sleepycat.com/download/db/index.shtml>

Junto con el código fuente de la base de datos, hay que descargar dos parches que se encuentran en el mismo servidor: `patch.4.2.52.1` y `patch.4.2.52.2`.

La instalación se realiza ejecutando los siguientes comandos:

```
tar zxvf db-4.1.25.tar.gz
cd db-4.1.25
```

A continuación hay que copiar en el directorio en el que nos encontramos los dos parches descargados anteriormente, y continuar ejecutando los siguientes comandos:

```
patch -p0 <match.4.2.52.1
patch -p0 <match.4.2.52.2
cd build_unix
../dist/configure
make
make install
```

Por defecto, las utilidades de la base de datos habrán sido colocadas en el directorio `/usr/local/BerkeleyDB.4.1/bin/`. Para que sean accesibles de forma directa, sin modificar el PATH del sistema, hay que crear un link simbólico por cada una de las utilidades, a continuación se muestra como crear el primero, pero hay que repetir el proceso para cada una de ellas:

```
ln -s /usr/local/BerkeleyDB.4.1/bin/db_archive \
    /usr/local/bin/db_archive

ln -s /usr/local/BerkeleyDB.4.1/bin/db_archive /usr/bin/db_archive
```

También hay que crear un link a las librerías e includes de la base de datos para que el sistema las pueda localizar:

```
ln -s /usr/local/BerkeleyDB.4.1/include/db.h /usr/include/db.h
ln -s /usr/local/BerkeleyDB.4.1/include /usr/local/include/db4
ln -s /usr/local/BerkeleyDB.4.1/lib /usr/local/lib/db4
```

Para finalizar la instalación de la base de datos, hay que añadir la línea `/usr/local/BerkeleyDB.4.1/lib` al archivo `/etc/ld.so.conf` y ejecutar el comando: `ldconfig -vv` que actualizará el archivo `/etc/ld.so.cache`.

4.4.2 – Cyrus - SASL

SASL (*Simple Authentication and Security Layer*) es un protocolo seguro de autenticación simple necesario para una correcta funcionalidad del servicio LDAP. La versión utilizada es la `cyrus-sasl-2.1.14` que se puede obtener en el siguiente servidor:

<http://ftp.andrew.cmu.edu/pub/cyrus/OLD-VERSIONS/sasl/>

La instalación se realiza ejecutando los siguientes comandos:

```
tar zxvf cyrus-sasl-2.1.14.tar.gz
cd cyrus-sasl-2.1.14
./configure --with-bdb-libdir=/usr/local/BerkeleyDB.4.1/lib \
            --with-bdb-incdir=/usr/local/BerkeleyDB.4.1/include \
            --with-openssl=/usr/local/openssl
```

El hecho de haber utilizado la base de datos DB4 obliga a crear un link simbólico que haga accesibles las librerías instaladas:

```
ln -s /usr/local/lib/sasl2 /usr/lib/sasl2
```

Por último, hay que asegurarse que el directorio `/usr/local/include/sasl` tiene permiso de lectura para todo el mundo.

4.4.3 – OpenLDAP

Una vez instalada la base de datos y el protocolo SASL, ya se puede instalar el software núcleo del servicio de directorio OpenLDAP. La versión utilizada es la

openldap-2.1.21. Si se intenta instalar una versión superior a la **openldap-2.1.30**, se producirá un error de incompatibilidad con la base de datos.

Todas las versiones de este paquete software se pueden descargar desde el servidor:

<http://www.openssl.org>

Para instalar el paquete hay que seguir los mismos pasos que con las aplicaciones vistas anteriormente, la diferencia estriba en la definición de dos variables de entorno en el momento de configurar el código fuente.

```
tar zxvf openldap-2.1.21.tar.gz
cd openldap-2.1.21
env CPPFLAGS="-I/usr/local/include -I/usr/local/include/ssl \
             -I/usr/local/include/db4" \
    LDFLAGS="-L/usr/local/openssl/lib -L/usr/local/lib/db4" \
    ./configure --with-tls --with-cyrus-sasl \
               --enable-wrappers --enable-crypt --enable-bdb
make
make test
make install
```

La instrucción **make test** permite realizar un chequeo de la instalación, si no se produce ningún error, indicará que las tres aplicaciones instaladas funcionan correctamente. Si se produjese algún error, no se debería continuar con la instalación hasta que se resolviesen los posibles conflictos.

Ahora hay que configurar la aplicación y añadir los objetos de usuario a la base de datos. El archivo de configuración es el `/usr/local/etc/openldap/slapd.conf`. En el Anexo 8 se puede ver el archivo utilizado en este proyecto. El atributo "rootpw" es el password cifrado del usuario "Manager"; para poder obtener la representación textual de un password cifrado e incluirla en el archivo de configuración, se debe ejecutar el siguiente comando:

```
slappasswd -h {MD5} -s "PasswordManager"
```

Los certificados que se indican en el archivo de configuración, y que permitirán una conexión segura con el servidor, son los que se han creado anteriormente en el apartado 4.3 con la aplicación OpenSSL.

Existe otro archivo de configuración, el `/usr/local/etc/openldap/lapd.conf` que contiene la configuración que utilizarán los clientes del servicio LDAP (ver Anexo 9). En él se definen los parámetros que se utilizarán para realizar la conexión con el servidor.

Por último, hay que añadir al servidor LDAP todos aquellos objetos del sistema que se deseen (usuarios, dominios, organizaciones, etc.). Para realizar esta operación hay que crear un archivo con extensión `ldif` que contenga todos los datos de los objetos a añadir. En el Anexo 10 se puede ver el archivo utilizado para realizar la carga inicial del servidor. Para poder añadir los datos del archivo al servidor, es necesario que éste se esté ejecutando. El comando que lanza la ejecución del servidor es el mostrado a continuación:

```
/usr/local/libexec/slapd -d256
```

El parámetro `-d256` hace que el servidor se ejecute en modo depuración, lo cual es muy aconsejable para comprobar si se producen errores en la ejecución del servicio.

Para incluir los objetos mostrados en el Anexo 10 en el servidor LDAP se debe ejecutar el siguiente comando:

```
ldapadd -x -W -D 'cn=Manager,dc=wireless,dc=uoc,dc=edu' \  
-f MiArchivo.ldif -H "ldaps://radius.wireless.uoc.edu"
```

Existen aplicaciones gráficas, como LDAP Browser/Editor, que permiten la manipulación del servidor LDAP (añadir, modificar, eliminar, visualizar, etc). Desde esta aplicación también se puede realizar la carga inicial de datos. LDAP Browser/Editor se puede descargar desde el servidor:

<http://www.iit.edu/~gawojar/ldap/>

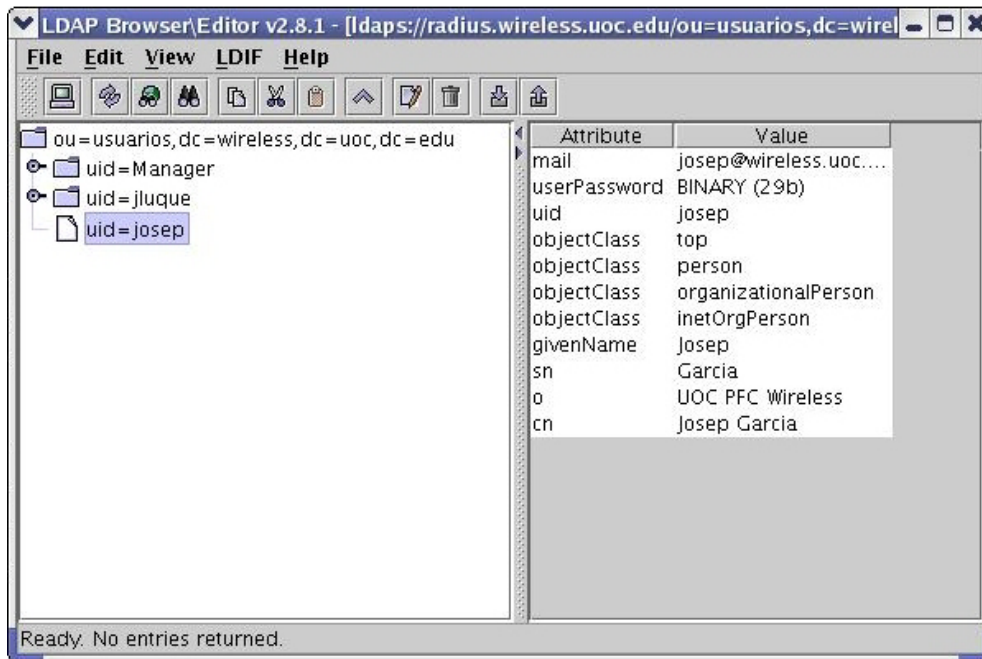


Figura 4.1: Aspecto de LDAP Browser/Editor

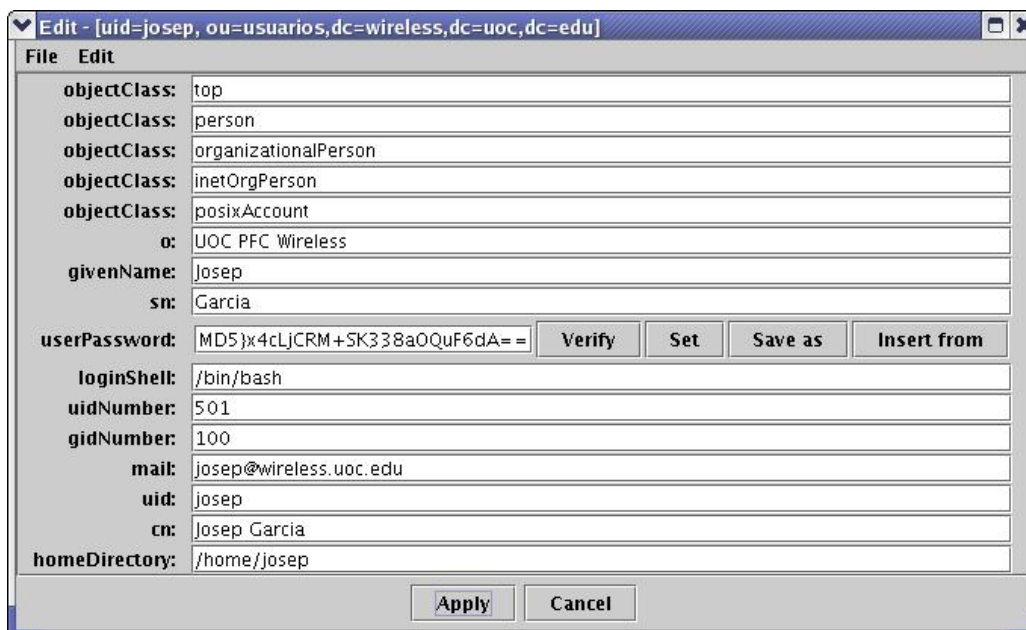


Figura 4.2: Editar usuario con LDAP Browser/Editor

La ventaja o desventaja de esta aplicación es que está escrita en Java, lo cual implica que se debe tener instalado el JRE de Java.

En la figura 4.1 se observar el aspecto de la aplicación LDAP Browser/Editor una vez cargados los datos. En la figura 4.2 se muestra el formulario que permite editar los atributos de un usuario.

Al igual que con todos los servicios vistos hasta ahora, la instalación finaliza habilitando la ejecución automática del servicio en los niveles 3 y 5 del sistema operativo.

4.5 – FreeRADIUS

FreeRADIUS es el servidor RADIUS más utilizado en plataformas LINUX. Se ha instalado la última versión snapshot que hay en el servidor:

```
ftp://ftp.freeradius.org/pub/radius/CVS-snapshots/freeradius-snapshot-20040520.tar.gz
```

La instalación es un poco diferente a la de los servicios anteriores ya que requiere realizar un paso intermedio que permite la compilación del código fuente con las funcionalidades EAP-TLS y EAP-TTLS habilitadas. La instalación comienza con los siguientes comandos:

```
tar zxvf freeradius-snapshot-20040520.tar.gz
cd freeradius-snapshot-20040520
./configure --sysconfigdir=/etc
cd src/modules/rlm_eap/types/rlm_eap_tls
```

Ahora hay que modificar el archivo `Makefile` que se encuentra en el directorio en el que nos encontramos. Se debe cambiar el parámetro que indica la ubicación del software OpenSSL, tal y como se muestra en el siguiente recuadro:

```
# Generated automatically from Makefile.in by configure.
TARGET = rlm_eap_tls
SRCS = rlm_eap_tls.c eap_tls.c cb.c tls.c mppe_keys.c
RLM_CFLAGS = $(INCLTDL) -I../.. -I/usr/local/openssl/include
HEADERS = eap_tls.h
RLM_INSTALL =
RLM_LDFLAGS += -L/usr/local/openssl/lib
RLM_LIBS += -lssl -lcrypto

$(STATIC_OBJS): $(HEADERS)

$(DYNAMIC_OBJS): $(HEADERS)

RLM_DIR=../..
include ${RLM_DIR}../rules.mak
```

Se continúa la instalación desplazándonos al directorio que contiene el código que habilita EAP-TTLS y realizando las mismas modificaciones en el archivo `Makefile` del directorio actual.

```
cd ../rlm_eap_tls
```

Una vez realizadas las modificaciones, ya se puede continuar con la instalación

```
make
make install
```

La configuración básica del servidor FreeRADIUS es relativamente simple, solo son necesarios modificar cuatro archivos. Al haber utilizado la opción `--sysconfdir=/etc` en la instalación, los archivos de configuración se encontrarán en el directorio `/etc/raddb`.

Se puede comenzar modificando el archivo `clients.conf`; en él se definen los clientes del servidor RADIUS, en nuestro caso, el cliente de FreeRADIUS es el punto de acceso, así que hay que añadir las siguientes líneas al archivo `clients.conf`:

```
client 192.168.255.4 { #Dirección IP del punto de acceso
    secret = secreto #Password que debe enviar el AP
    shortname = AP_WLAN #Nombre descriptivo del AP
}
```

El archivo `eap.conf`, mostrado en el Anexo 11, contiene la configuración de los módulos del FreeRADIUS que permitirán realizar una autenticación vía EAP-TLS o EAP-TTLS.

El valor del parámetro `fragment_size` depende del modelo/tipo de punto de acceso que utilicemos, en nuestro caso, el punto de acceso utilizado es el modelo “Buffalo AirStation G54” y el valor empleado para el parámetro `fragment_size` es 1750. A veces, este dato no es fácilmente localizable, así que se deberán realizar pruebas hasta conseguir una correcta comunicación entre el punto de acceso y FreeRADIUS.

En el archivo `radiusd.conf` hay que modificar los modos de autorización y autenticación que se pueden utilizar, quedando de la siguiente forma:

```
authorize {
    preprocess
    eap
    suffix
    files
    ldap
}

authenticate {
    unix
    authtype LDAP {
        ldap
    }
    eap
}
```

En el mismo archivo `radiusd.conf`, hay que configurar el módulo `ldap` para que se pueda conectar al servicio LDAP instalado anteriormente. Se debe indicar la ubicación de servidor, la identidad del usuario con permiso de consulta, los certificados del servidor y de la entidad certificadora creados en este mismo capítulo,

el filtro de búsqueda, y sobre todo el archivo de mapeo de atributos del LDAP y FreeRADIUS.

```
ldap {
    server = "radius.wireless.uoc.edu"
    port = 636
    identity = "cn=Manager,ou=usuarios,dc=wireless,dc=uoc,dc=edu"
    password = secreto
    basedn = " ou=usuarios,dc=wireless,dc=uoc,dc=edu "
    filter = "(uid=%{Stripped-User-Name:-%{User-Name}})"

    start_tls = yes
    tls_cacertfile = /etc/certificados/root.pem
    tls_cacertdir = /etc/certificados
    tls_certfile = /etc/certificados/radius.wireless.uoc.edu.pem
    tls_keyfile = /etc/certificados/radius.wireless.uoc.edu.key
    tls_randfile = /etc/certificados/random
    tls_require_cert = "demand"
    access_attr = "msNPAllowDialin"

    dictionary_mapping = ${raddbdir}/ldap.attrmap
    ldap_connections_number = 5

    timeout = 4
    timelimit = 3
    net_timeout = 1
    access_attr_used_for_allow = yes
}
```

En el Anexo 12 se muestra el archivo `ldap.attrmap` que se utiliza como regla de mapeo entre el diccionario de atributos del FreeRADIUS y el conjunto de atributos del directorio LDAP. Es un archivo que el administrador del sistema puede adaptar a sus necesidades modificando el mapeo o incluso añadiendo nuevos atributos, siempre y cuando los haya definido antes de usarlos en el archivo `dictionary`, o que ya existan en alguno de los múltiples diccionarios que instala la aplicación FreeRADIUS en el directorio `/usr/local/share/freeradius/dictionary`. Todos estos atributos se corresponden con las variables de entorno compartidas entre los clientes inalámbricos y el servidor FreeRADIUS.

La operativa seguida por un servidor FreeRADIUS en los procesos de autorización y autenticación es la siguiente:

- Autorización: El servidor intenta localizar las credenciales suministradas por el cliente en alguno de los repositorios indicados en el archivo de configuración (eap, files, ldap, etc.); si las localiza en alguno de ellos, se prosigue con el proceso de autenticación.
- Autenticación: El proceso de autenticación se realiza exclusivamente con un método, el que se haya definido en la configuración del usuario en el archivo `users`. Si se define que un usuario debe autenticarse vía EAP, el servidor FreeRADIUS no intentará realizar una autenticación vía LDAP, por lo tanto, no leerá los atributos que definen el perfil del usuario en el directorio LDAP.

Para finalizar la configuración básica del servidor FreeRADIUS se debe modificar el archivo `users`, añadiendo el tipo de autenticación que se debe utilizar con cada usuario (LDAP, EAP, System, etc.). Es aconsejable añadir la opción `DEFAULT`, que indicará el tipo de autenticación que se realizará con todos aquellos usuarios que no están explícitamente definidos.

```
"josep" Auth-Type := LDAP  
  
DEFAULT Auth-Type := EAP
```

Como siempre, hay que configurar el servicio para que se ejecute de forma automática, pero en este caso se debe crear un script que permita ejecutar el servicio con unas variables de entorno que definan la ubicación de las librerías criptográficas instaladas en el servidor. El script, al que llamaremos "run-radiusd", debe ser como se muestra a continuación, siempre y cuando se haya instalado el paquete OpenSSL en el directorio que se indicó cuando se explicaba la instalación del paquete criptográfico.

```
#run-radiusd  
#!/bin/sh -x  
  
LD_LIBRARY_PATH=/usr/local/openssl/lib  
LD_PRELOAD=/usr/local/openssl/lib/libcrypto.so
```



```
export LD_LIBRARY_PATH LD_PRELOAD
/usr/local/radius/sbin/radiusd $@
```

Ahora hay que hacer ejecutable el script, y como siempre, configurar el sistema para que ejecute el comando al iniciarse el sistema en los niveles 3 y 5 de ejecución. Para convertir el script en ejecutable se debe lanzar el siguiente comando:

```
chmod 700 ./run-radiusd
```

Si se desea ejecutar el servicio en modo depuración, para analizar su comportamiento, se debe ejecutar el script con los parámetros `-X` y `-A`.

```
run-radiusd -X -A
```

4.5.1 – Módulo SQL

FreeRADIUS posee un módulo de consultas a bases de datos que permite ser configurado para conectarse a MySQL, Postgres, MS-SQL, Oracle, e incluso a cualquier tipo de base de datos a través de ODBC.

Este módulo puede ser ejecutado después de que un usuario ha sido autenticado, permitiendo realizar una consulta a una base de datos para obtener todo el perfil del usuario autenticado. De forma automática, este módulo envía todo el perfil leído de la base de datos al usuario autenticado.

```
post-auth {
    # Get an address from the IP Pool.
    #main_pool

    #
    # If you want to have a log of authentication replies,
    # un-comment the following line, and the 'detail reply_log'
    # section, above.
    #reply_log

    #
```

```
# After authenticating the user, do another SQL query.
#
# See "Authentication Logging Queries" in sql.conf
sql
#
# Access-Reject packets are sent through the REJECT sub-
# section of the post-auth section.
#
# Post-Auth-Type REJECT {
#     insert-module-name-here
# }
}
```

Dependiendo del tipo de base de datos al que deseamos que se conecte, habrá que modificar y adaptar el archivo de configuración correspondiente, e indicarlo en el archivo de configuración radiusd.conf.

```
# Include another file that has the SQL-related configuration.
# This is another file only because it tends to be big.
#
# The following configuration file is for use with MySQL.
#
# For PostgreSQL, use:      ${confdir}/postgresql.conf
# For MS-SQL, use:         ${confdir}/mssql.conf
# For Oracle, use:         ${confdir}/oraclesql.conf
#
$INCLUDE ${confdir}/sql.conf
```

4.6 – Configurar el punto de acceso

La configuración del punto de acceso es idéntica a la realizada en el capítulo anterior, la única diferencia estriba en la dirección IP del servidor Radius al que debe realizar las peticiones de autorización y autenticación. En la figura 3.19 se veía que el parámetro “RADIUS Server” se había configurado con el valor 192.168.255.3, ahora habrá que asignarle la dirección IP del nuevo servidor Radius, que en nuestro caso es 192.168.255.2.

4.7 – Configurar cliente inalámbrico

La configuración del cliente no cambia con respecto a la implementada en el capítulo 3, punto 3.7. El usuario no se ve afectado por los cambios realizados en el servidor, da igual que el servidor RADIUS se esté ejecutando en una plataforma Windows, como que se esté ejecutando en LINUX, los protocolos utilizados son los mismos.

La novedad en este capítulo es que FreeRADIUS implementa la autenticación EAP-TTLS, cosa que no hacía Windows 2000 Server. Si se desea utilizar este tipo de autenticación, se debe instalar el protocolo EAP-TTLS en el cliente. Actualmente, no hay ningún sistema operativo que lo implemente de serie, así que hay que acudir a software de terceros para poder implementar el protocolo.

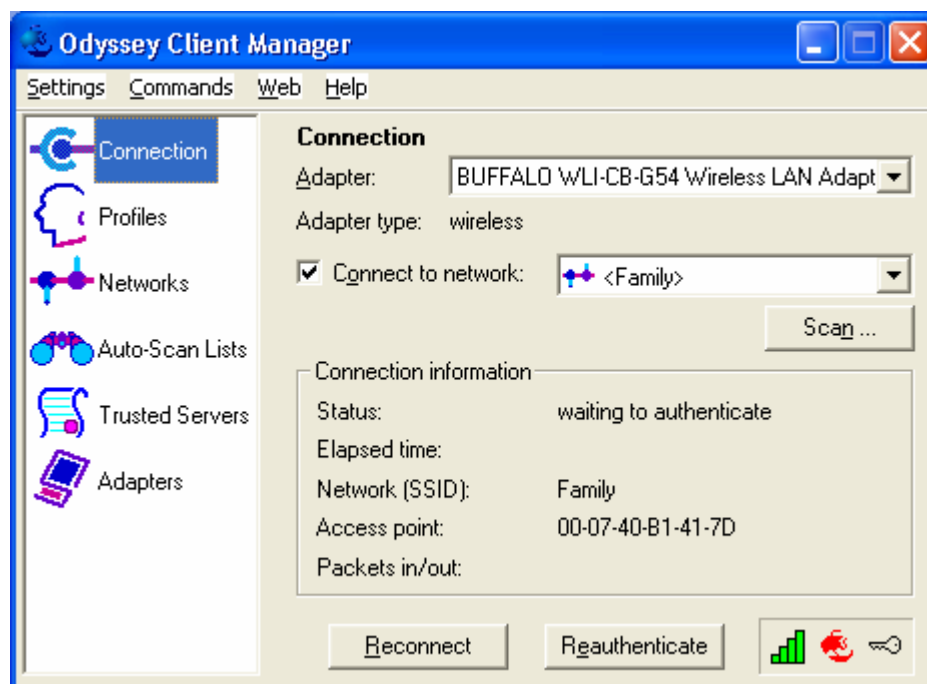


Figura 4.3: Cliente EAP-TTLS para Windows

De los diferentes tipos de protocolos EAP existentes para autenticar usuarios, Windows XP solo implementa EAP-TLS y PEAP, si se desea instalar EAP-TTLS se debe acudir a un software ajeno a Windows. Para realizar las pruebas en este proyecto, se ha utilizado el cliente “Odyssey” de “Funk Software, Inc” con una licencia de prueba de 30 días, dicho cliente se puede obtener en el siguiente servidor:

<http://www.funk.com>

La instalación es idéntica a la de cualquier software de Windows, solo hay que seguir las indicaciones del wizard de instalación.

Una vez instalado el cliente de red, se requiere una pequeña configuración que defina los parámetros de conexión del cliente EAP-TTLS. En la figura 4.3 se muestra el aspecto del cliente Odyssey. El primer punto a configurar es el perfil de usuario que utilizará la aplicación. Si se accede a la opción “Profiles” aparecerá una ventana donde se definen todos los parámetros del usuario. En la figura 4.4 se muestra la configuración “User Info” establecida para realizar las pruebas.

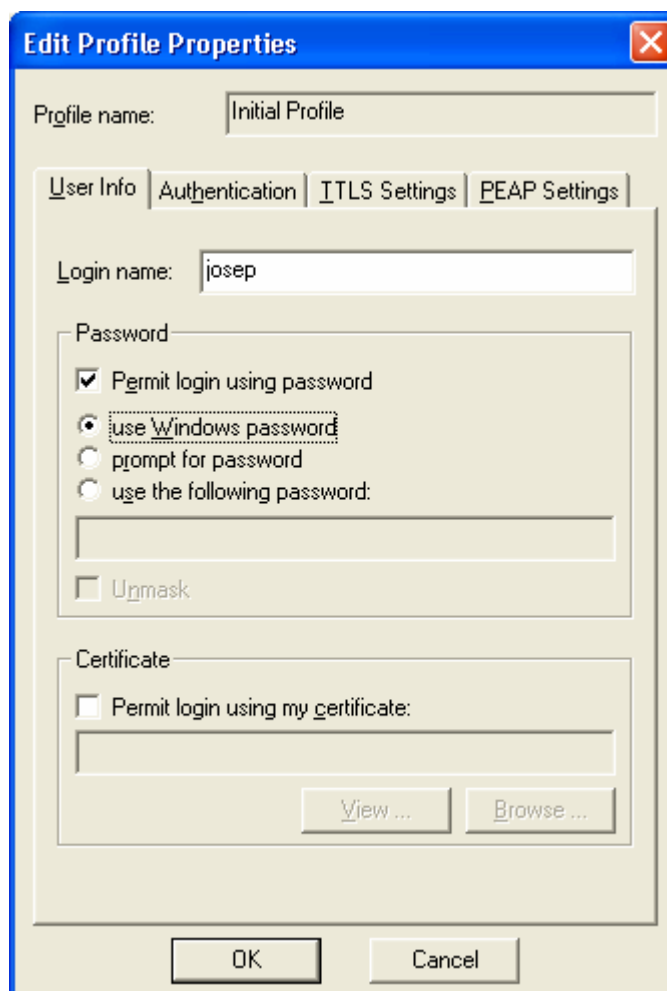


Figura 4.4: Configuración “User Info” del cliente “Odyssey”

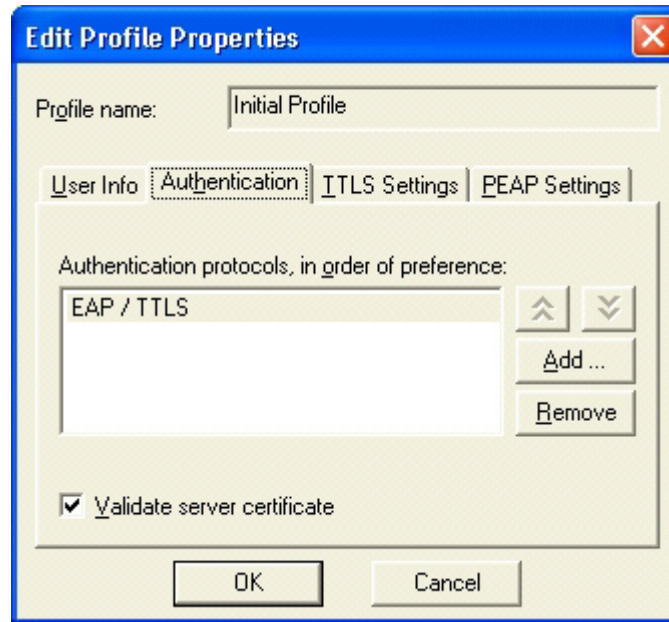


Figura 4.5: Configuración “Authentication” del cliente “Odyssey”

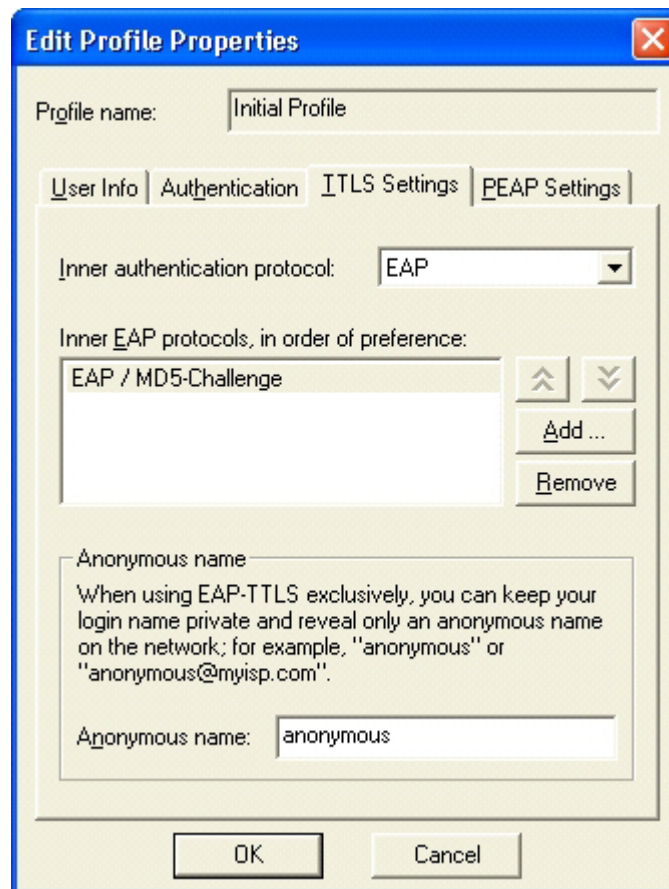
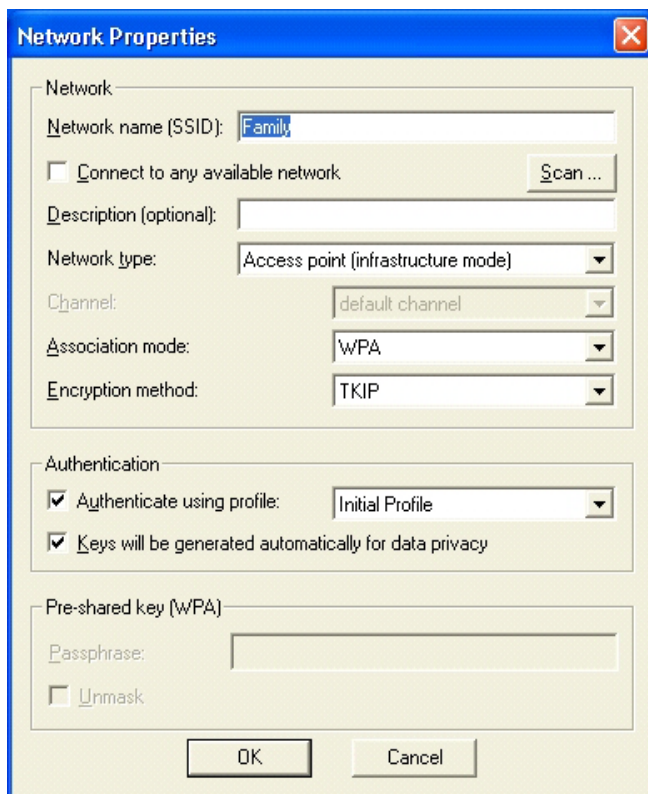


Figura 4.6: Configuración del protocolo EAP-TTLS en el cliente “Odyssey”



En la pestaña “Authentication” se define el tipo de autenticación que utilizará el cliente de red. En la figura 4.5 se muestra el tipo de autenticación escogida; lógicamente, se ha optado por el protocolo EAP-TTLS, ya que éste era el motivo de instalar el cliente “Odyssey”.

Por último, en la figura 4.6 se muestra la configuración establecida para el protocolo EAP-TTLS.

Figura 4.7: Configuración propiedades de red del cliente “Odyssey”

Con este paso finaliza la configuración del perfil de usuario, ahora hay que configurar las propiedades de red accediendo a la opción “Networks” mostrada en la figura 4.3. En la figura 4.7 se muestra la configuración de red establecida, teniendo en cuenta la configuración de cifrado definida en el punto de acceso.

La utilización del protocolo de autenticación EAP-TTLS tiene la ventaja de no tener que instalar un certificado digital en el cliente para conseguir una autenticación segura. En la configuración realizada en el archivo `eap.conf` del servidor FreeRADIUS en el punto anterior, se había dejado habilitada la posibilidad de autenticar vía EAP-TTLS, así que solo hay que realizar una conexión con el cliente Odyssey utilizando autenticación TTLS, y FreeRADIUS aceptará y validará al usuario.

4.8 – Errores de ejecución

Si se han seguido los pasos descritos en este capítulo, y se ha utilizado el mismo sistema operativo, se verá que se genera un error de “Violación de Segmento” en el programa FreeRADIUS cuando se utiliza autenticación vía LDAP. Es un defecto (bug) del programa cuando utiliza las librerías del protocolo SASL. En la siguiente dirección se puede comenzar a seguir el hilo del bug:

http://bugs.freeradius.org/show_bug.cgi?id=73

Una de las soluciones planteadas indica que actualizando el software de LDAP a su última versión, se resuelve el problema. Cuando se intenta realizar esta operación, aparecen incompatibilidades con al base de datos Berkeley, y ésta ya se encuentra instalada en su última versión.

En este proyecto no se ha encontrado una solución viable al problema, sobre todo por cuestiones de tiempo, pero se podrían realizar las siguientes pruebas para resolver el problema:

- Utilizar otro tipo de base de datos como soporte de LDAP (lmdb, Postgres, MySQL, etc.) que permita instalar versiones más avanzadas del paquete OpenLDAP
- El problema se produce porque hay dos librerías (`libsasl.so` y `libsasl2.so`) que contienen una misma función (nombre idéntico), pero que realizan operaciones diferentes. Sería cuestión de analizar las dos librerías e intentar modificarlas.
- Por último, y como medida drástica, se podría realizar la misma implementación pero en una distribución diferente de LINUX.

4.9 – Conclusión

En este capítulo, al igual que en el capítulo anterior, se han visto todos los pasos a seguir para implementar una red inalámbrica segura, pero en este caso se ha utilizado una plataforma con sistema operativo LINUX. En este capítulo se ha añadido un nuevo elemento no utilizado en el capítulo anterior, se ha implementado autenticación EAP-TTLS, tanto en el servidor como en el cliente.

Se ha producido un problema software no solventado con el software FreeRADIUS y OpenLDAP. El servidor FreeRADIUS genera un error de “Violación de Segmento” cuando utiliza las librerías de SASL. Es un defecto del programa FreeRADIUS. En el apartado 4.8 de este capítulo se indican posibles caminos a seguir para solventar dicho defecto.

Al igual que en el capítulo anterior, no se han implementado ciertas medidas de seguridad como la utilización de un cortafuegos y una red privada virtual, las cuales se dejan para el último capítulo en el que se fusionan las dos plataformas y se implementan todos los elementos de seguridad comentados en la introducción del proyecto.

Capítulo 5

WLAN segura con MS Windows 2000 Server y LINUX

Llegados a este punto, a excepción del cortafuegos y la red privada virtual, ya están instalados y operativos todos los servicios necesarios para abordar la última fase del proyecto en la que se crean las interacciones entre las dos plataformas. A excepción de los dos servicios indicados anteriormente, el resto de servicios solo requieren una pequeña modificación en su configuración para que funcionen en perfecta conjunción con el resto de servicios.

5.1 – DHCP

El servidor DHCP instalado en la plataforma Windows no requiere ninguna modificación, ya que se utiliza para proporcionar configuraciones de red a los clientes de la Intranet. Por cuestiones de seguridad, todos los clientes que acceden a la red a través del punto de acceso inalámbrico reciben una dirección de red diferente a la que posee la Intranet. Es el propio servidor LINUX el que enrutará las conexiones de los clientes autenticados hacia la Intranet.

Este cambio en las asignaciones de red obliga a realizar modificaciones en la configuración del servidor DHCP instalado en la plataforma LINUX. Únicamente se debe modificar el rango de direcciones que asigna y las opciones globales de red, tal y como se muestra en el siguiente recuadro. Las opciones no mostradas no deben ser cambiadas:

```
option broadcast-address 172.16.0.255;
option routers 172.16.0.1;
option netbios-name-servers 192.168.255.3;
option domain-name-servers 172.16.0.1;
option ip-forwarding on;
subnet 172.16.0.0 netmask 255.255.255.0 {
    range 172.16.0.3 172.16.0.254;
}
```

5.2 – DNS

El cambio realizado en el servidor DHCP obliga a realizar modificaciones en el servidor DNS de la plataforma LINUX. Se debe añadir una nueva zona de búsqueda inversa que permita almacenar las nuevas direcciones que sirve el DHCP.

En el archivo `named.conf` se debe añadir la definición de la nueva zona:

```
zone "0.16.172.in-addr.arpa" {
    type master;
    allow-update { "localnets"; };
    file "0.16.172.in-addr.arpa.zone";
};
```

Lógicamente, hay que crear el nuevo archivo de zona “0.16.172.in-addr.arpa.zone” con la siguiente información:

```
$ORIGIN .
$TTL 3600 ; 1 hour
0.16.172.in-addr.arpa IN SOA      root.wireless.uoc.edu.
radius_w.wireless.uoc.edu. (
                           37      ; serial
                           3600     ; refresh (1 hour)
                           900      ; retry (15 minutes)
                           3600000  ; expire (5 weeks 6 days 16
hours)
                           3600     ; minimum (1 hour)
                           )
                           NS       radius_w.wireless.uoc.edu.
$ORIGIN 0.16.172.in-addr.arpa.
1 PTR radius_w.wireless.uoc.edu.
2 PTR ap.wireless.uoc.edu.
```

Como se puede ver en el nuevo archivo de zona, el punto de acceso (ap) tiene una dirección de red perteneciente a la red de los clientes inalámbricos. Para que el punto de acceso se pueda comunicar con el servidor FreeRADIUS es necesario definir una nueva interfaz de red en el servidor LINUX con una dirección IP que pertenezca a la misma red del punto de acceso. En nuestro caso particular, se ha creado un interfaz con la dirección 172.16.0.1/24. En el resto de archivos de zonas hay que realizar las modificaciones correspondientes que definan las nuevas direcciones de cada uno de los elementos de red del sistema.

5.3 – LDAP – Active Directory

Conseguir que Active Directory de Microsoft se pueda utilizar como servicio de autenticación de un sistema LINUX requiere realizar ciertas modificaciones en el AD. Antes de proseguir con la explicación hay que ser consciente de que Microsoft no da soporte al uso de autenticaciones contra AD que no procedan de una máquina con sistema operativo de Microsoft. El autor de este proyecto ha realizado los cambios explicados a continuación en el AD sin ningún tipo de problema ni fallo, lo cual no significa que no pudiesen producirse en algún escenario concreto, es por ello que la responsabilidad de aplicar el siguiente procedimiento en una plataforma Microsoft corre a cargo de la persona que lo realice, y nunca sobre el autor de este proyecto.

5.3.1 – AD4Unix

AD4Unix es un plugin desarrollado por Maxim Batourine de la Universidad de Toronto, bajo licencia de libre uso tanto para aplicaciones privadas como comerciales, que añade un nuevo esquema a AD que permite el manejo de atributos propios de Unix/Linux desde la consola de administración de AD.

La aplicación está en formato .MSI (*Microsoft Installer*) y se puede descargar en el siguiente servidor:

<http://www.padl.com/download/MKSADPlugins.msi>

Antes de realizar la instalación de la aplicación, hay que modificar el AD para que permita actualizaciones de esquema en el controlador de dominio. Los pasos a seguir son los siguientes:

- Registrar en el sistema una librería dinámica ejecutando el siguiente comando desde una ventana de MS-DOS

```
regsvr32 c:\winnt\system32\schmmgmt.dll
```

- Crear un gestor de esquemas. Se debe ejecutar el comando MMC, y en el menú de consola, seleccionar la opción “Agregar o quitar complemento...”. A continuación se agrega el complemento “Esquema de Active Directory”.
- Se selecciona el complemento añadido y pulsando el botón derecho del ratón se selecciona “Cambiar el controlador de dominio”, en la nueva ventana que aparece hay que seleccionar nuestro dominio.
- Volviendo a seleccionar el complemento añadido y pulsando el botón derecho del ratón, se selecciona la opción “Maestro de Operaciones”. En la ventana que aparece, se debe seleccionar la casilla “Se puede modificar el esquema en este controlador de dominio”, y se pulsa Aceptar. Ahora ya se puede instalar la aplicación AD4Unix.

La instalación del plugin AD4Unix se realiza como cualquier software de Microsoft.

Una vez instalado el plugin, si se accede a las propiedades de un usuario de Active Directory, se observará una nueva pestaña llamada “Unix Setting” en la que se pueden modificar y añadir los valores de los atributos propios de los usuarios de

Linux (réplica de la información del archivo `/etc/passwd`), tal y como se muestra en la figura 5.1.

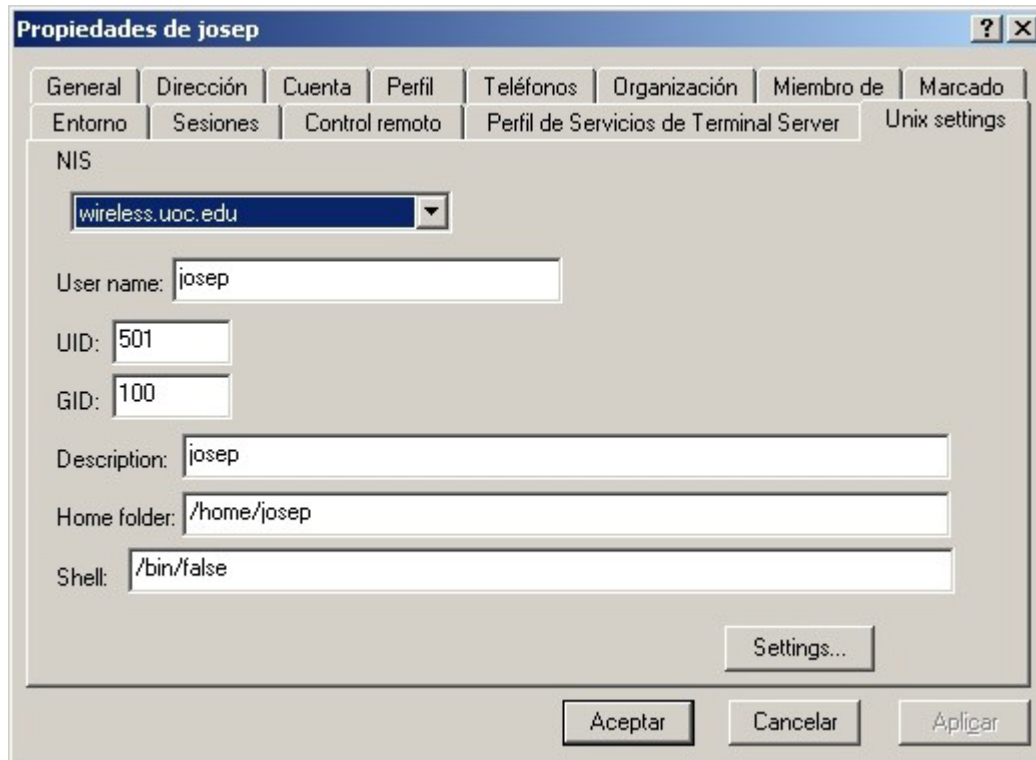


Figura 5.1: Atributos usuarios Linux en Active Directory

Como ya se ha comentado, con este plugin instalado se podría utilizar el servicio Active Directory como sistema de autenticación de usuarios que se presentan en máquinas Linux. En este proyecto, el servicio que realizará peticiones al Active Directory será el FreeRADIUS de Linux, obteniendo en la consulta el perfil del usuario inalámbrico, no importando que sea un cliente Linux o Windows.

Por defecto, Active Directory no permite consultas anónimas, para solventar el problema hay dos soluciones:

1. Modificar el Active Directory para que permita consultas anónimas, lo cual es muy inseguro.
2. Crear un usuario en el dominio con un perfil muy restringido, pero con permiso de lectura sobre el Active Directory.



Figura 5.2: Consulta sobre AD con cliente LDAP

La segunda opción es la seguida en este proyecto. Se ha creado en el dominio el usuario “consultaLDAP” con un perfil muy restringido. Este usuario será el utilizado por el cliente LDAP de Linux para realizar las consultas.

5.3.2 – Cliente LDAP

El cliente LDAP que realizará las consultas sobre Active Directory es el propio servidor FreeRADIUS. En el módulo `ldap` del archivo de configuración `radiusd.conf`, visto en el capítulo anterior, hay que realizar ciertas modificaciones para que las consultas las realice sobre Active Directory. A continuación se muestran dichas modificaciones:

```
server = "ad.wireless.uoc.edu" #Servidor con AD
identity = "cn=consultaLDAP,cn=Users,dc=wireless,dc=uoc,dc=edu"
password = secreto
basedn = "cn=Users,dc=wireless,dc=uoc,dc=edu"
```

Como se puede apreciar, las modificaciones realizadas implican un cambio en el servidor de consulta, la identidad del usuario que realiza la consulta, y el DN base de la búsqueda.

Si este cambio se realiza en el archivo `ldap.conf`, podremos utilizar la aplicación LDAP Browser/Editor para comprobar que las consultas funcionan correctamente (ver figura 5.2).

A partir de este momento, cualquier petición de autenticación recibida por FreeRADIUS procedente de clientes inalámbricos configurados con autenticación LDAP será enviada al servidor Active Directory para que la resuelva.

5.4 – Cortafuegos

El cortafuegos utilizado es el servicio “iptables” proporcionado por propio sistema operativo Linux instalado. Para configurarlo se ha utilizado la aplicación gráfica GNU “Firewall Builder” (ver figura 5.3), que se puede obtener en el servidor:

<http://www.fwbuilder.org>

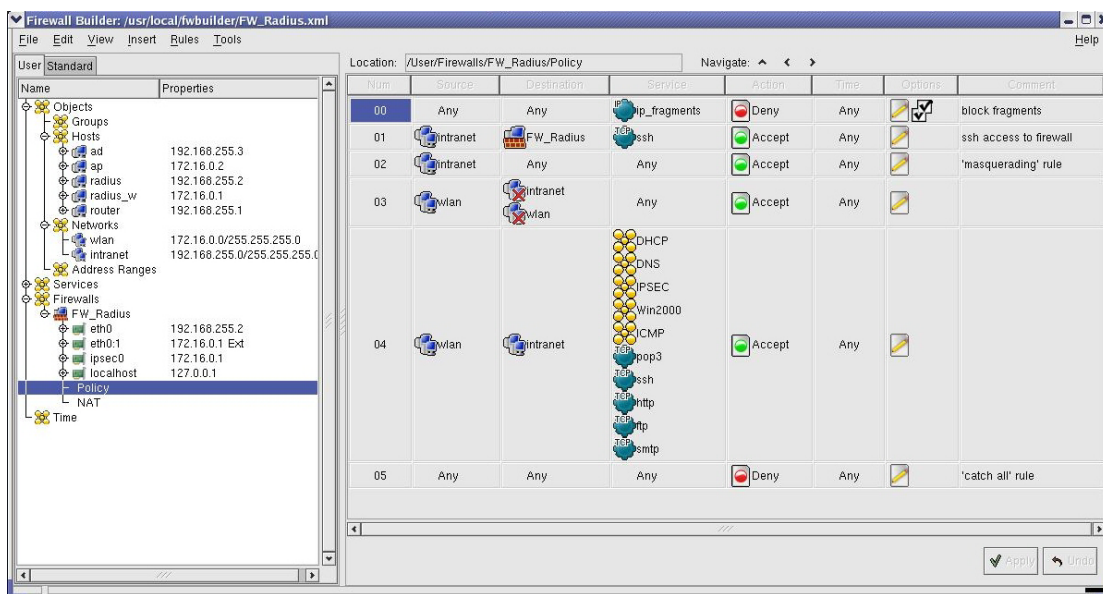


Figura 5.3: Firewall Builder

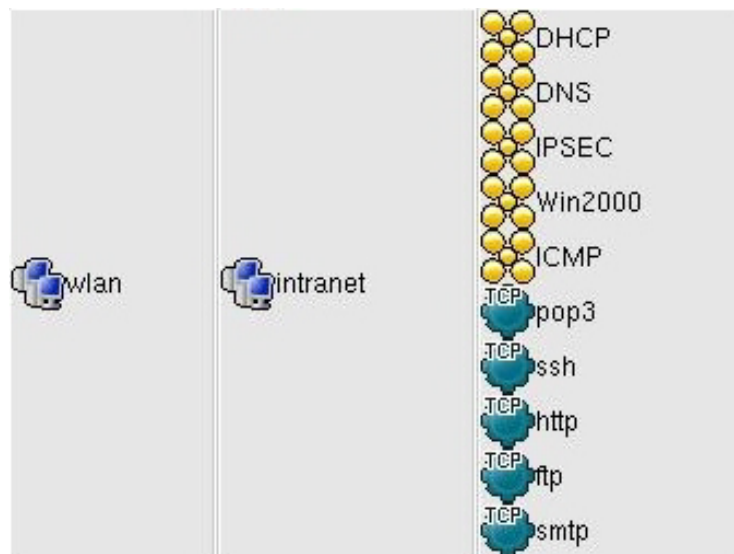


Figura 5.4: Protocolos permitidos

Esta aplicación permite definir todas las reglas que deseemos aplicar en el cortafuegos de una forma fácil y gráfica, generando a partir del diseño gráfico realizado todo el script que permite configurar el cortafuegos (ver Anexo 13). En principio, se ha configurado para que solo permita el tráfico entrante procedente de la red inalámbrica que coincida con los protocolos mostrados en la figura 5.4.

5.5 – Servidor VPN-IPSec

Para la implementación del servidor IPSec se ha utilizado el paquete Openswan (evolución de FreeS/WAN) en su versión 2.1.2-14 preparada para el kernel instalado en la máquina Linux. Los paquetes han sido descargados desde el servidor:

<http://www.openswan.org>

Los dos paquetes descargados del servidor han sido:

```
kernel-module-openswan-2.4.22-1.2188.nptlsmp-2.1.2-14.rhfc1.at.i686.rpm  
openswan-2.1.2-14.rhfc1.at.i386.rpm
```

El primero es un módulo que se integra en el kernel del sistema operativo, y el segundo es todo el conjunto de aplicaciones y scripts que permiten la implementación y administración del servidor VPN.

La instalación de ambos paquetes se realiza como cualquier otro paquete RPM visto en el presente proyecto:

```
rpm -ivh kernel-module-openswan-2.4.22-1.2188.nptlsmp-2.1.2-  
14.rhfc1.at.i686.rpm  
  
rpm -ivh openswan-2.1.2-14.rhfc1.at.i386.rpm
```

La configuración se realiza modificando dos archivos: `/etc/ipsec.conf` y `/etc/ipsec.secret`. El primero contiene la configuración del servidor, y el segundo contiene la referencia al certificado de la autoridad certificadora.

De los tres posibles tipos de autenticación que permite el servidor VPN (Clave compartida, Firma Digital RSA y Certificado X509), en este proyecto se ha utilizado el certificado X509 ya que tenemos una Infraestructura de Clave Pública implementada.

En el Anexo 14 está el archivo de configuración `/etc/ipsec.conf` utilizado, como se puede ver, solo se ha creado una conexión, la que corresponde al usuario con el que se están realizando las pruebas. En caso de existir diferentes usuarios habría que crear una conexión para cada uno de ellos.

La usar autenticación con certificado X509, el servidor requiere que los certificados se encuentren ubicados en el directorio `/etc/ipsec.d/certs` para que los pueda localizar.

El archivo `/etc/ipsec.secret` contiene la clave privada del certificado del servidor, junto con el password que permite leerlo.

```
: RSA radius.wireless.uoc.edu.key "secreto"
```

El inicio automático del servicio se realiza como cualquier otro servicio de Linux.

Para finalizar, solo hay que crear una nueva conexión de red en el cliente, teniendo en cuenta que el tipo de conexión debe ser VPN. Si el sistema operativo del cliente es Windows XP no hay que instalar nada nuevo, en caso de ser otro sistema operativo, habrá que descargar las actualizaciones que implementen el protocolo IPsec.

5.6 – Conclusión

En esta fase del proyecto se han integrado los servicios de las dos plataformas implementadas en los capítulos anteriores, añadiendo dos nuevos servicios que aumentan la seguridad general del sistema (Cortafuegos y VPN).

Se ha añadido un plugin a Active Directory que extiende su funcionalidad, permitiendo poder gestionar conjuntamente atributos propios de sistemas Unix/Linux y de Windows.

Capítulo 6

Conclusiones

El objetivo inicial del proyecto era la integración de un acceso externo inalámbrico a una infraestructura de red ya establecida. Dicha integración tenía dos condicionante:

1. El nuevo acceso inalámbrico no podía poner en peligro la seguridad general del sistema.
2. El control de acceso debía utilizar los recursos existentes en la infraestructura de red original, comportándose de forma equiparable a cualquier otro acceso por cable ya existente en la red.

El primer objetivo se ha cumplido en su totalidad. Dentro de los actuales estándares de seguridad establecidos para conexiones inalámbricas, se han utilizado los más avanzados.

- Encriptación WPA-TKIP entre los clientes inalámbricos y el punto de acceso a la red.
- Autenticación de usuarios con certificado de clave pública (EAP-TLS y EAP-TTLS).
- Red Privada Virtual (VPN) entre los clientes y la red cableada.
- Direccionamiento IP diferente del establecido en la red cableada para los clientes inalámbricos.
- Cortafuegos que impide el tráfico no deseado procedente de los clientes inalámbricos.

Hay una medida extra de seguridad que no se ha implementado por falta de tiempo y que permitiría aumentar la seguridad del sistema. Como ya se ha comentado, se ha utilizado un cortafuegos que impide el tráfico no deseado procedente de la red inalámbrica. Sería interesante que el cortafuegos tuviese reglas dinámicas, es decir, en principio no debería dejar pasar ningún tráfico procedente de la red inalámbrica, y a medida que el servidor RADIUS autentificase a un usuario, añadiese una regla que permitiese el tráfico controlado de dicho usuario. La funcionalidad de ejecutar scripts por parte del servidor FreeRADIUS está implementada, solo hay que habilitar el script, y preparar otro que elimine la regla del cortafuegos cuando el cliente finalice su sesión.

Respecto al segundo objetivo del proyecto, por culpa de un error (bug) de “Violación de Segmento” del programa FreeRADIUS (ver punto 4.8 del presente proyecto) que hizo perder mucho tiempo, no se han podido cubrir todas las expectativas a nivel de implementación, pero sí a nivel de análisis.

Se ha implementado la autenticación de clientes, tanto Linux como Windows, utilizando el FreeRADIUS como cliente LDAP del servicio Active Directory de Windows, el cual almacena tanto el perfil Windows como Linux de los clientes.

La comprobación total de este tipo de autenticación no ha sido posible por culpa del error software de FreeRADIUS comentado anteriormente. Además, este tipo de autenticación aunque permite transferir el perfil de usuario desde el servidor al cliente, disminuye la seguridad del sistema, ya que es más segura una autenticación EAP-TLS o EAP-TTLS que una autenticación LDAP.

Se ha estudiado, pero no implementado, el uso del módulo SQL de FreeRADIUS que permite la lectura del perfil de usuario de una base de datos después de que el usuario haya sido autenticado con el protocolo EAP-TLS o EAP-TTLS.

Dado que no se ha podido implementar el envío de perfiles a los clientes, no se ha podido comprobar la reacción de los clientes al recibir el perfil, ya que en ningún

momento lo han solicitado, es decir, cuando los clientes se conectan a la red inalámbrica, solicitan acceso a ella, pero no solicitan la unión a un dominio o login a una máquina.

Este proyecto deja abiertas diferentes líneas de trabajo que permitirían una mejora en la seguridad y una mejor integración del acceso inalámbrico en la infraestructura de red actual.

Bibliografía

- [1] Robbie Allen, Alistair G. Lowe-Norris. *Active Directory*. O'Reilly, 2nd Edition, April 2003
- [2] Jill Spealman. *Microsoft Windows 2000 Active Directory Services*. Microsoft Press, 2000
- [3] Brian Arkills. *LDAP Directories Explained: An Introduction and Analysis*. Addison Wesley, February 2003
- [4] Gerald Carter. *LDAP System Administration*. O'Reilly, March 2003
- [5] Timothy A. Howes Ph.D., Mark C. Smith, Gordon S. Good. *Understanding and Deploying LDAP Directory Services*. Addison Wesley, May 2003
- [6] Jonathan Hassell. *RADIUS*. O'Reilly, October 2002
- [7] Paul Heltzel. *Complete Home Wireless Networking: Windows XP Edition*. Prentice Hall PTR. June 2003
- [8] Linda McCarthy. *IT Security*. Prentice Hall PTR. February 2003
- [9] Cyrus Peikari, Seth Fogie. *Maximum Wireless Security*. Sams Publishing. December 2002
- [10] Anton Chuvakin, Cyrus Peikari. *Security Warrior*. O'Reilly. January 2004
- [11] Rob Flickenger. *Wireless Hacks*. O'Reilly. September 2003
- [12] Jon Edney, William A. Arbaugh. *Real 802.11 Security: Wi-Fi Protected Access and 802.11i*. Addison Wesley. July 2003
- [13] Tara M., Charles R. Elden. *Wireless Security and Privacy: Best Practices and Design Techniques*. Addison Wesley. September 2002
- [14] Mario Strasser. *DHCPv4 Configuration of IPsec Tunnel Mode HOWTO*. mast@gmx.net. August 2002
- [15] *How to Cheat at Securing Windows 2000 TCP/IP*. Syngress Publishing. 2003

- [16] Toni dIF. Díaz. *Autenticación e Integridad en redes Wireless*. <http://madridwireless.net>
- [17] Thomas Lee, Joseph Davies. *Microsoft Windows 2000 TCP/IP Protocols and Services. Technical Reference*. Microsoft Press. 2000
- [18] Roel Van Meer. *LDAP Implementation HOWTO*. Linvision BV. Revision 0.5, 2001
- [19] Harri Levanen, Bernard Freund, Hani Mansi. *Using LDAP for Directory Integration*. IBM Redbooks. December 2000
- [20] Paul Wolfe, Mike Erwin, Charlie Scoot. *Virtual Private Network*. O'Reilly 2nd Edition. 1999
- [21] *RADIUS for UNIX. Administrator's Guide*. Lucent Technologies. February 1999
- [22] *RFC 3580 – IEEE 802.1x Remote Authentication Dial In User Service (RADIUS) Usage Guidelines*. Network Working Group. September 2003
- [23] Adam Sulmicki. *HOWTO on EAP/TLS authentication between FreeRADIUS and XSupplicant*. <http://www.missl.cs.umd.edu/wireless/eaptls/>
- [24] Adam Sulmicki. *HOWTO: EAP/TLS Setup for FreeRADIUS and Windows XP Supplicant*. April 2002. <http://www.freeradius.org/doc/EAPTLS.pdf>
- [25] *Internet Authentication Service for Windows 2000*. <http://www.microsoft.com/technet/prodtechnol/windows2000serv/evaluate/featfunc/ias.msp>
- [26] Vladimir Vuksan. *DHCP mini-HOWTO*. Revision v4.12. October 2000. <http://www.tldp.org/HOWTO/DHCP/index.html>
- [27] Raymond McKay. *FreeRADIUS EAP/TLS – WinXP HOWTO*. Version 1.2. October 2002. <http://www.impossiblereflex.com/8021x/eap-tls-HOWTO.htm>
- [28] FreeS/WAN. *FreeS/WAN 2.4 documentation*. March 2003. http://www.freeswan.org/freeswan_trees/freeswan-2.04/doc/toc.html

Glosario de acrónimos

AD	<i>Active Directory</i> Directorio Activo
AP	<i>Access Point</i> Punto de Acceso
ADSL	<i>Asymmetric Digital Subscriber Line</i> Línea de Suscripción Digital Asimétrica
AES	<i>Advanced Encryption Standard</i> Estándar de Cifrado Avanzado
BSS	<i>Basic Service Set</i> Conjunto de Servicios Básicos
CA	<i>Certificate Authority</i> Autoridad Certificadora
CBC-MAC	<i>Cipher Block Chaining-Message Authentication Code</i> Bloques Cifrados Encadenados-Código Autenticador de Mensajes
CCM	<i>Counter mode with CBC-MAC</i> Modo Contador con CBC-MAC
CHAP	<i>Challenge-Handshake Authentication Protocol</i> Protocolo de Autenticación por Desafío-Respuesta
CRL	<i>Certificate Revocation List</i> Lista de Certificados Revocados
DB	<i>Data Base</i> Base de Datos
DES	<i>Digital Encryption Standard</i> Estándar de Encriptación Digital
DHCP	<i>Dinamyc Host Configuration Protocol</i> Protocolo de Configuración Dinámica de Equipos

DNS	<i>Domain Name Server</i> Servidor de Nombres de Dominio
EAP	<i>Extensible Authentication Protocol</i> Protocolo de Autenticación Extensible
EAPOL	<i>EAP Over Lan</i> EAP Sobre Red de Área Local
EAPOR	<i>EAP Over RADIUS</i> EAP Sobre RADIUS
EAPOW	<i>EAP Over Wireless</i> EAP Sobre Red de Área Local Inalámbrica
ESS	<i>Extended Service Set</i> Conjunto de Servicios Extendidos
IAS	<i>Internet Authentication Service</i> Servicio de Autenticación de Internet
IBSS	<i>Independent Basic Service Set</i> Conjunto de Servicios Básicos Independientes
IEEE	<i>Institute of Electrical and Electronics Enginneers</i> Instituto de Ingenieros Eléctricos y Electrónicos
IP	<i>Internet Protocol</i> Protocolo de Internet
IPSec	<i>IP Security</i> IP Segurizado
JRE	<i>Java Runtime Enviroment</i> Entorno de Ejecución de Java
L2TP	<i>Layer 2 Tunneling Protocol</i> Protocolo de Capa 2 Tunelizado
LAN	<i>Local Area Network</i> Red de Área Local

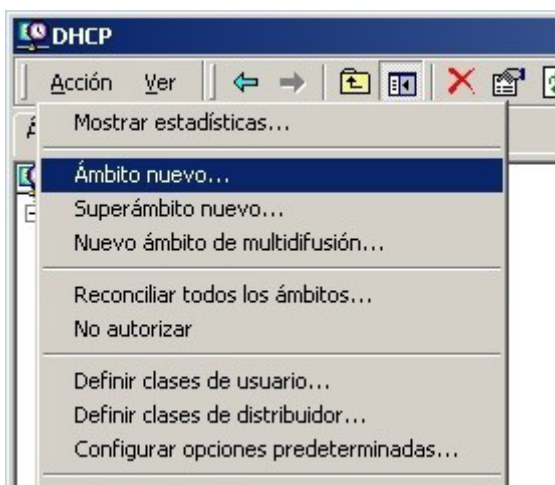
LDAP	<i>Lightweight Directory Access Protocol</i> Protocolo Ligero de Acceso a Directorio
LDIF	<i>LDAP Data Interchange Format</i> Formato de Intercambio de Datos del LDAP
MAC	<i>Medium Access Control</i> Control de Acceso al Medio
MIC	<i>Message Integrity Code</i> Código de Integridad de Mensajes
MS-CHAP	<i>MicroSoft Challenge-Handshake Authentication Protocol</i> Protocolo de Autenticación por Desafío-Respuesta de Microsoft
NAS	<i>Network Access Server</i> Servidor de Acceso a la Red
NAT	<i>Network Address Translation</i> Traducción de Direcciones de Red
OS	<i>Operating System</i> Sistema Operativo
PDC	<i>Primary Domain Controller</i> Controlador Primario de Dominio
PKI	<i>Public Key Infrastructure</i> Infraestructura de Clave Pública
PPP	<i>Point-to-Point Protocol</i> Protocolo de Punto-a-Punto
PPTP	<i>Point-to-Point Tunneling Protocol</i> Protocolo Punto-a-Punto Tunelizado
RADIUS	<i>Remote Authentication Dial-In User Service</i> Servicio de Autenticación de Usuarios de Acceso Remoto
RPM	<i>Redhat Package Manager</i> Gestor de Paquetes de Redhat

SASL	<i>Simple Authentication and Security Layer</i> Autenticación Simple y Capa Segura
SSID	<i>Service Set Identity</i> Identificación de Bloque de Servicio
SSL	<i>Secure Socket Layer</i> Capa Conectora Segura
TCP	<i>Transport Control Protocol</i> Protocolo de Control de Transporte
TKIP	<i>Temporal Key Integrity Protocol</i> Protocolo de Integridad de Clave Temporal
TLS	<i>Transport Layer Security</i> Capa de Transporte Segura
TSC	<i>TKIP Sequence Counter</i> Contador de Secuencia TKIP
TTLS	<i>Tunneled TLS</i> TLS Tunelizado
VPN	<i>Virtual Private Network</i> Red Privada Virtual
WEP	<i>Wired Equivalent Privacy</i> Privacidad Equivalente al Cable
WINS	<i>Windows Internet Naming Service</i> Servicio de Nombres de Internet de Windows
WPA	<i>Wi-Fi Protected Access</i> Acceso Wi-Fi Protegido
Wi-Fi	<i>Wireless Fidelity</i> Fidelidad Inalámbrica
WLAN	<i>Wireless Local Area Network</i> Red de Área Local Inalámbrica

Anexos

Anexo 1 – Configuración DHCP en Windows

Para configurar el servicio DHCP se debe que acceder al menú de “Herramientas administrativas” y seleccionar DHCP. Esta acción ejecutará una aplicación que permite configurar el protocolo / servicio DHCP. Lo primero que se debe hacer es crear un nuevo ámbito.

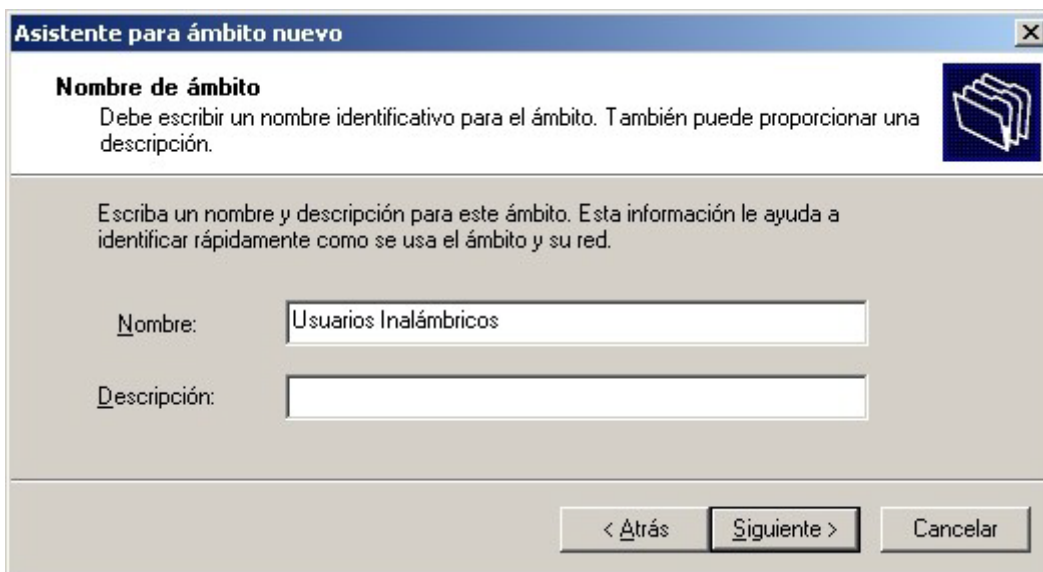


Dentro de la ventana que aparece al ejecutar la aplicación, se selecciona nuestra máquina y en el menú “Acción” se escoge la opción “Ámbito Nuevo”, tal y como muestra la figura a1.1. Esta acción desencadena la ejecución de un programa de ayuda que irá preguntando todos los parámetros necesarios para configurar el servidor DHCP.

Figura a1.1: Añadir nuevo ámbito al servidor DHCP.

El programa de ayuda irá solicitando la siguiente información:

- 1. Nombre del ámbito.** Es el nombre que le queremos dar al ámbito. Como es posible tener diferentes ámbitos, es importante asignarle un nombre identificativo, que al leerlo nos proporcione la suficiente información como para saber la finalidad del ámbito. En la figura a1.2 se observa que hemos llamado “Usuarios Inalámbricos” al nuevo ámbito.
- 2. Intervalo de direcciones IP.** Es el rango de direcciones IP que el servidor proporcionará a los clientes. Desde esta ventana (figura a1.3) también se indica cual es la máscara de red que será proporcionada a los clientes.



Asistente para ámbito nuevo

Nombre de ámbito
Debe escribir un nombre identificativo para el ámbito. También puede proporcionar una descripción.

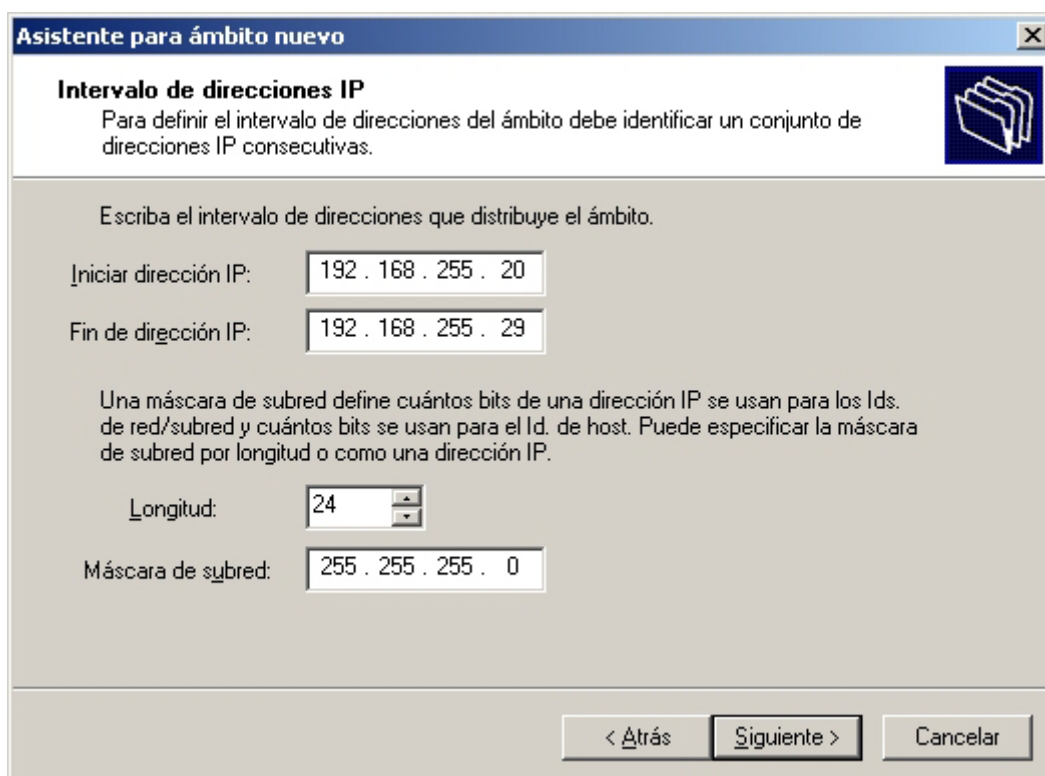
Escriba un nombre y descripción para este ámbito. Esta información le ayuda a identificar rápidamente como se usa el ámbito y su red.

Nombre:

Descripción:

< Atrás Siguiete > Cancelar

Figura a1.2: Asignar nombre al nuevo ámbito DHCP.



Asistente para ámbito nuevo

Intervalo de direcciones IP
Para definir el intervalo de direcciones del ámbito debe identificar un conjunto de direcciones IP consecutivas.

Escriba el intervalo de direcciones que distribuye el ámbito.

Iniciar dirección IP:

Fin de dirección IP:

Una máscara de subred define cuántos bits de una dirección IP se usan para los Ids. de red/subred y cuántos bits se usan para el Id. de host. Puede especificar la máscara de subred por longitud o como una dirección IP.

Longitud:

Máscara de subred:

< Atrás Siguiete > Cancelar

Figura a1.3: Rango de direcciones IP y máscara que proporciona el servidor DHCP.

3. **Agregar exclusiones.** Desde esta ventana se pueden configurar direcciones excluidas del rango, es decir, direcciones que pertenecen al rango pero que el servidor nunca ofrecerá a ningún cliente. En nuestro caso no excluirémos ninguna dirección.
4. **Duración de concesión.** Indica durante cuanto tiempo concede el servidor una dirección IP a un cliente. Si transcurrido dicho tiempo el cliente sigue conectado a la red, el servidor volverá a concederle la misma IP durante el mismo período; en caso contrario, el servidor habilitará la dirección IP para asignársela a cualquier otro nuevo cliente. Sabiendo que un cliente inalámbrico suele ser transitorio, una duración de concesión de 4 horas es un valor adecuado.
5. **Configurar opciones DHCP.** El programa de ayuda preguntará si queremos configurar ahora el resto de parámetros del servidor, responderemos afirmativamente.
6. **Enrutador (puerta de enlace predeterminada).** Indica cual será la puerta de enlace de la red, es decir, la dirección IP a la que enviarán los paquetes las máquinas clientes cuando quieren comunicarse con direcciones de red que no pertenecen a la suya. En nuestro caso hemos indicado la dirección de un router de acceso a Internet (figura a1.4).
7. **Nombre de dominio y servidores DNS.** Esta información servirá para indicarle a los clientes cual es el nombre de su dominio principal, y cuales son los servidores de DNS (*Domain Name Service*) de la red, en principio se debería indicar como primer servidor, el propio servidor de DNS del dominio, como segundo servidor de DNS se puede indicar uno perteneciente a Internet. En la figura a1.5 se muestran los dos servidores DNS definidos para el ámbito que se está definiendo.

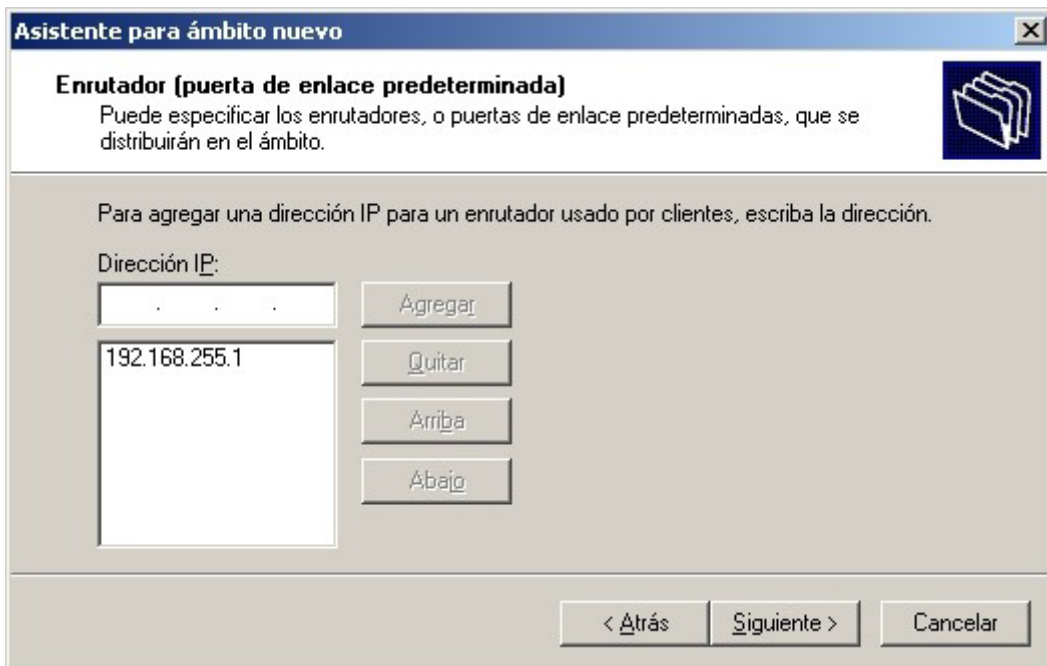


Figura a1.4: Puerta de enlace predeterminada que proporciona el servidor DHCP.

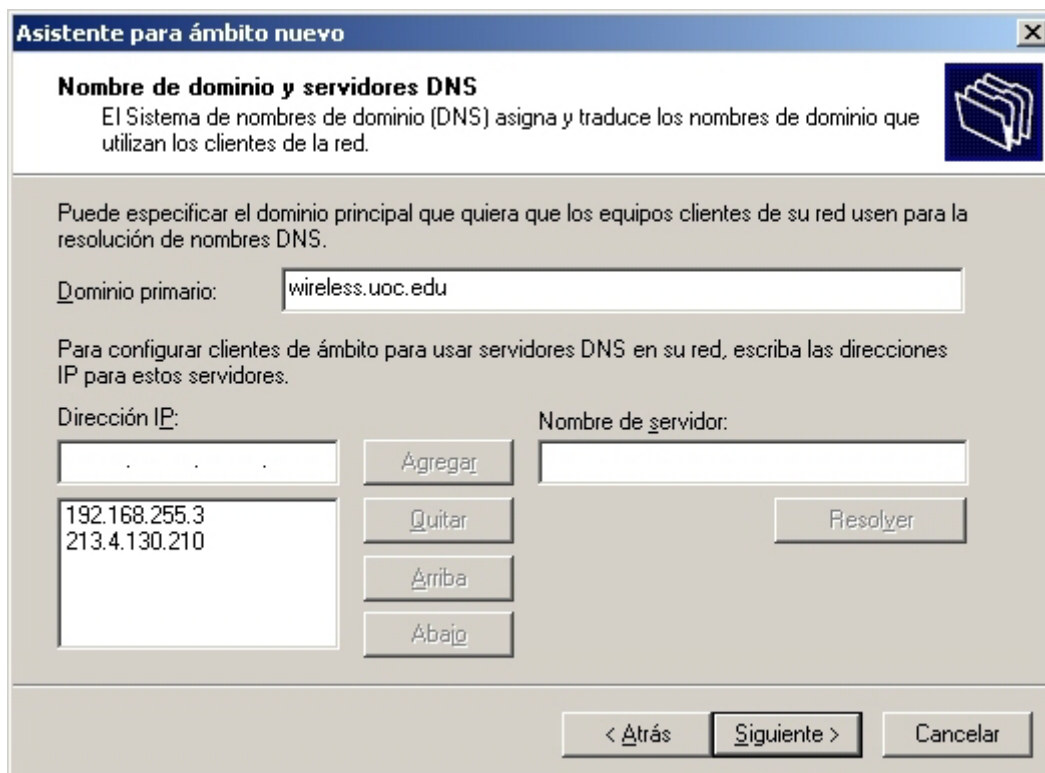


Figura a1.5: Nombre de dominio primario y servidores DNS que ofrece el DHCP.

8. **Servidores WINS.** Indicaremos la dirección del servidor WINS de la red, que coincidirá con la dirección IP de nuestro controlador de dominio (192.168.255.3).

9. **Activar ámbito.** Por último, se nos pregunta si queremos activar el ámbito, es decir, si queremos que comience a funcionar, responderemos afirmativamente.

Con este último paso finaliza la configuración inicial y básica del servidor DHCP, ahora solo falta registrarlo en el Active Directory. Como servicio de red de un dominio, debe estar registrado en el AD. Para registrarlo hay que acceder al menú “Acción” y seleccionar la opción “Autorizar”. A partir de este momento el servidor DHCP proporcionará configuraciones de red a cualquier cliente que lo solicite.

Anexo 2 – Configuración cliente inalámbrico con MS Windows XP

Para configurar un cliente inalámbrico con sistema operativo Windows XP se debe acceder a las propiedades de la conexión de red inalámbrica y seleccionar la pestaña “Redes Inalámbricas” (figura a2.1).

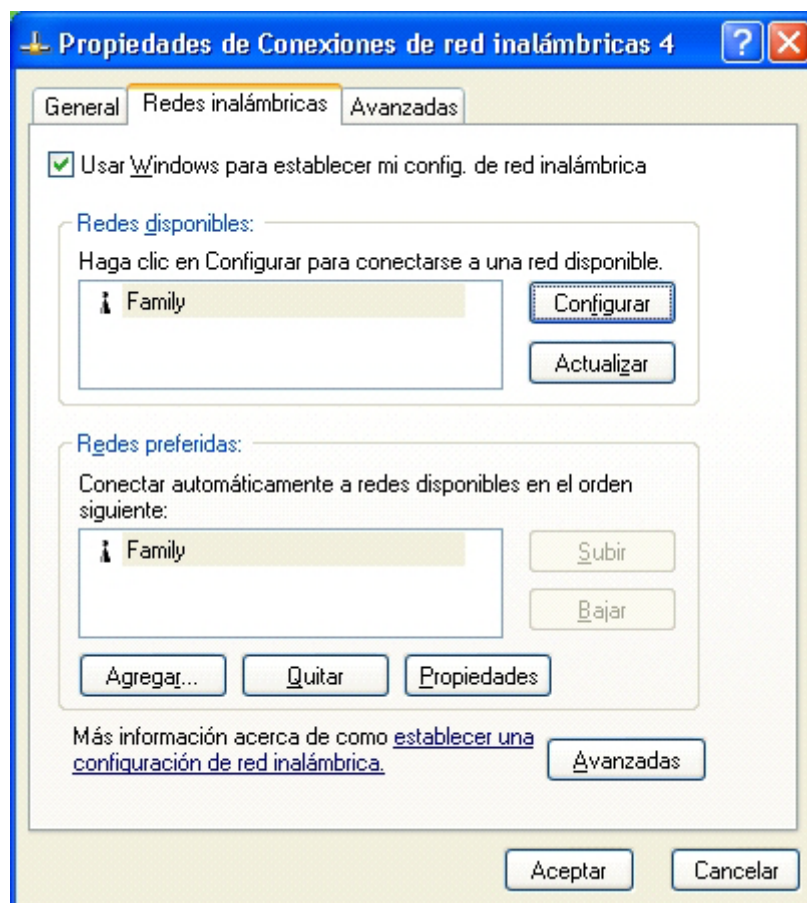


Figura a2.1: Propiedades de la conexión de red inalámbrica.

Si el punto de acceso y el dispositivo inalámbrico del cliente funcionan correctamente aparecerá dentro de la ventana “Redes disponibles” el SSID (identificador) del punto de acceso, hay que tener presente que el hecho de que el punto de acceso proporcione su propio SSID dependerá de la configuración de éste, tal y como se veía en la figura 3.19, habíamos dejado seleccionada la opción “Allow Broadcast SSID”, si se hubiese marcado la opción “Deny” sería necesario conocerlo

de antemano para poder conectarse con el punto de acceso, ya que habría que especificarlo explícitamente en la conexión inalámbrica.

A continuación se debe pulsar el botón “Configurar”, apareciendo la ventana que se muestra en la figura a2.2:

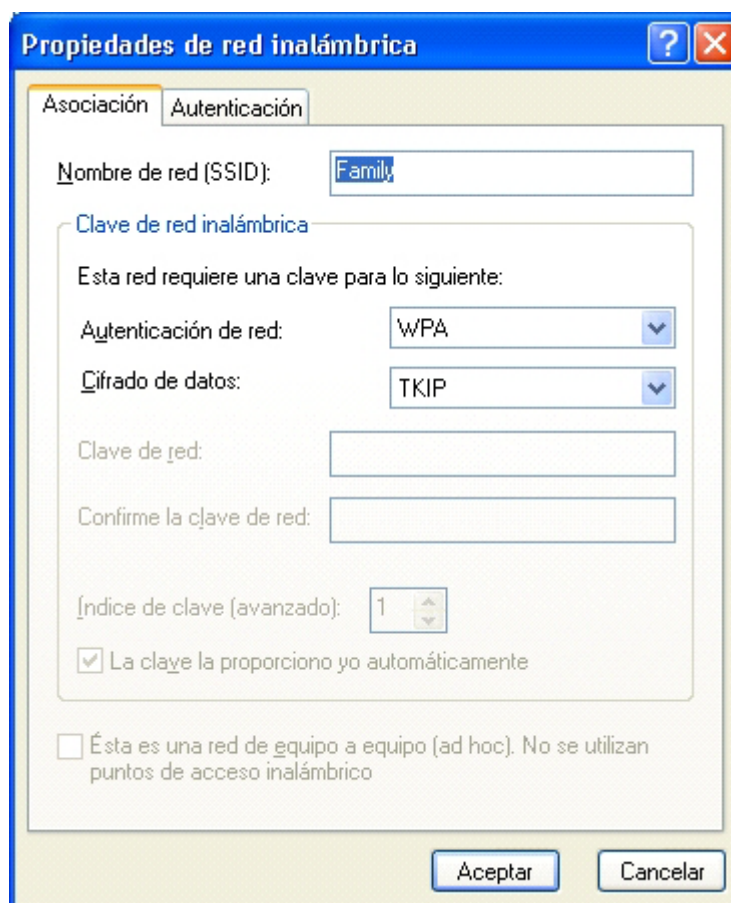


Figura a2.2: Configuración de la asociación inalámbrica.

En esta ventana se indica el protocolo que se debe utilizar para realizar la asociación, en nuestro caso, ya que punto de acceso lo permite, se utilizará autenticación de red WPA con cifrado de datos TKIP, tal y como se ha definido en el punto de acceso. Si se selecciona la pestaña “Autenticación” se puede definir el tipo de autenticación que utilizará el cliente frente al servidor, tal y como se vio al configurar el servidor, hay que seleccionar “Habilitar control de acceso a la red mediante IEEE 802.1x con el tipo de EAP “Tarjeta inteligente u otro certificado” (ver figura a2.3).

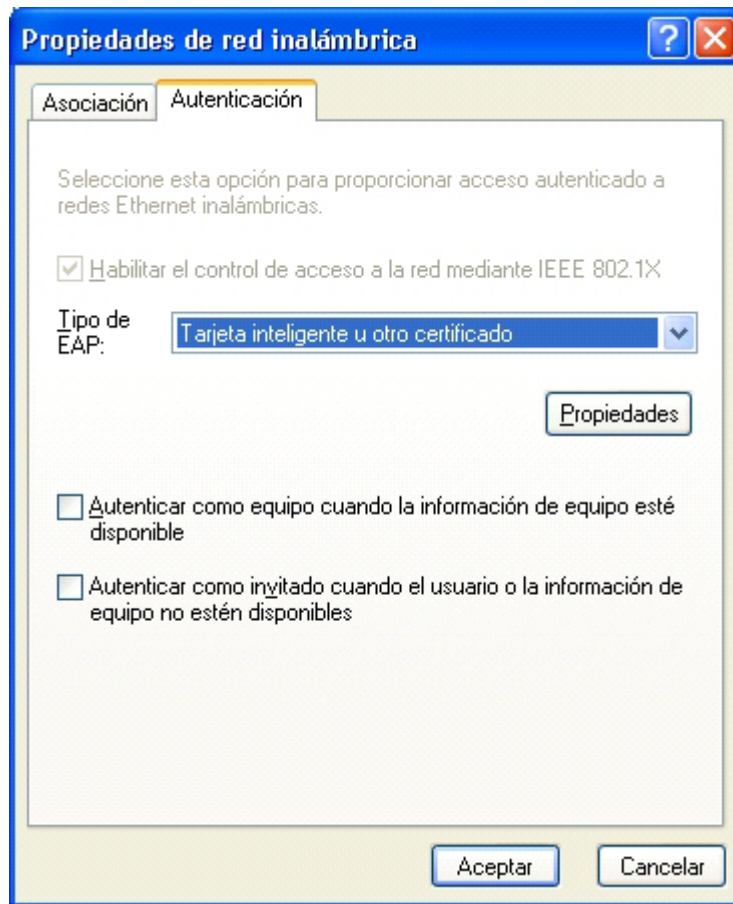


Figura a2.3: Definición del tipo de autenticación en la conexión.

Si aceptamos todas las variaciones realizadas, el cliente se debería conectar con el servidor sin problemas, y sin realizar ninguna otra petición de datos.

Anexo 3 – Archivo configuración DHCP /etc/dhcpd.conf

```
## /etc/dhcpd.conf ##
# Método utilizado por el servidor DHCP para actualizar
# el servidor DNS.
ddns-update-style interim;

# Habilita que el servidor DHCP actualice al DNS.
ddns-updates on;

# Tiempo de vida de las actualizaciones que realice el DHCP sobre
# el servidor DNS. 3600 segundos
ddns-ttl 3600;

# Indica que este servidor es autoritativo para el ámbito definido.
authoritative;

# Clave definida en el servidor DNS /etc/named.conf. Se usa para
# asegurar la autenticidad de las máquinas permitidas para
# actualizar los registros del DNS.
include "/etc/rndc.key";

# Número de segundos de concesión de IP a las máquinas.
default-lease-time 7200;

# Duración máxima de concesión de IPs en segundos.
max-lease-time 14400;

# Duración mínima de concesión de IPs en segundos.
min-lease-time 3600;

# Nombre del servidor DHCP.
server-name "radius.wireless.uoc.edu";

# Parámetros generales para todos los ámbitos definidos.
option subnet-mask 255.255.255.0; #Máscara de red
option broadcast-address 192.168.255.255; #Dirección de broadcast
option routers 192.168.255.1; #Puerta de enlace de la red
option netbios-name-servers 192.168.255.2; #Servidor WINS
option domain-name-servers 192.168.255.2; #Servidor DNS
option domain-name "wireless.uoc.edu"; #Nombre del dominio

# No permitir en reenvío de peticiones a otros servidores.
option ip-forwarding off;

# Rango de direcciones que ofrece el servidor.
subnet 192.168.255.0 netmask 255.255.255.0 {
    range 192.168.255.20 192.168.255.29;
}
```

Anexo 4 – Archivo configuración DNS /etc/named.conf

```
## /etc/named.conf ##
# Definición de las opciones generales del servidor
options {
    # Directorio raíz que contiene los archivos de zonas
    directory "/var/named";

    # Definición de quién puede realizar actualizaciones automáticas
    # del servidor. "localnets" representa a cualquier cliente de
    # las redes accesibles por el servidor
    allow-query { "localnets"; };
};

# Archivo de claves compartido con el servidor DHCP para asegurar
# su autenticación
include "/etc/rndc.key";

# Definición de la zona "hint". Esta zona representa a todas
# aquellas que no están definidas en el servidor. Cuando un cliente
# realiza una petición de resolución y no aparece en las zonas
# definidas, el servidor la reenvía a esta zona
zone "." {
    type hint;
    # Archivo de definición de zona
    file "named.root";
};

# Definición de zona para búsquedas directas
zone "wireless.uoc.edu" {
    # Este servidor es el principal de esta zona
    type master;

    # Puede ser actualizado por cualquier equipo de su red
    allow-update { "localnets"; };

    # Archivo de definición de zona
    file "wireless.uoc.edu.zone";
};

# Definición de zona para búsquedas inversas
zone "255.168.192.in-addr.arpa" {
    type master;
    allow-update { "localnets"; };
    file "255.168.192.in-addr.arpa.zone";
};
```

Anexo 5 – Archivos definición de zonas DNS

Archivo `/var/named/named.root`

```

; This file holds the information on root name servers needed to
; initialize cache of Internet domain name servers
; (e.g. reference this file in the "cache . <file>"
; configuration file of BIND domain name servers).
;
; This file is made available by InterNIC
; under anonymous FTP as
;   file           /domain/named.root
;   on server      FTP.INTERNIC.NET
; -OR-            RS.INTERNIC.NET
; last update:    Jan 29, 2004
; related version of root zone: 2004012900
;
; formerly NS.INTERNIC.NET
;
.           3600000   IN   NS       A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET. 3600000           A       198.41.0.4
;
; formerly NS1.ISI.EDU
;
.           3600000           NS       B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET. 3600000           A       192.228.79.201
;
; formerly C.PSI.NET
;
.           3600000           NS       C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET. 3600000           A       192.33.4.12
;
; formerly TERP.UMD.EDU
;
.           3600000           NS       D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET. 3600000           A       128.8.10.90
;
; formerly NS.NASA.GOV
;
.           3600000           NS       E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET. 3600000           A       192.203.230.10
;
; formerly NS.ISC.ORG
;
.           3600000           NS       F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET. 3600000           A       192.5.5.241
;
; formerly NS.NIC.DDN.MIL
;
.           3600000           NS       G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET. 3600000           A       192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
;
.           3600000           NS       H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET. 3600000           A       128.63.2.53

```

```

;
; formerly NIC.NORDU.NET
;
.           3600000      NS      I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.  3600000      A      192.36.148.17
;
; operated by VeriSign, Inc.
;
.           3600000      NS      J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.  3600000      A      192.58.128.30
;
; operated by RIPE NCC
;
.           3600000      NS      K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.  3600000      A      193.0.14.129
;
; operated by ICANN
;
.           3600000      NS      L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.  3600000      A      198.32.64.12
;
; operated by WIDE
;
.           3600000      NS      M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.  3600000      A      202.12.27.33
; End of File

```

Archivo /var/named/wireless.uoc.edu.zone

```

$ORIGIN .
$TTL 3600 ; 1 hora
wireless.uoc.edu IN SOA      radius_w.wireless.uoc.edu.
                          root.wireless.uoc.edu. (
                              1          ; Número serie
                              3600       ; Refresco (1 hora)
                              900        ; Reintento (15 minutos)
                              3600000    ; Expira (1000 horas)
                              3600       ; Mínimo (1 hora)
                              )
                          NS      radius_w.wireless.uoc.edu.

$ORIGIN wireless.uoc.edu.
router      A      192.168.255.1
radius      A      192.168.255.2
ap          A      192.168.255.10

```


Archivo /var/named/255.168.192.in-addr.arpa.zone

```
$ORIGIN .
$TTL 3600 ; 1 hora
255.168.192.in-addr.arpa      IN SOA      root.wireless.uoc.edu.
                               radius.wireless.uoc.edu. (
                               2          ; Número serie
                               3600       ; Refresco (1 hora)
                               900        ; Reintento (15 minutos)
                               3600000    ; Expira (1000 horas)
                               3600       ; Mínimo (1 hora)
                               )
                               NS      radius.wireless.uoc.edu.

$ORIGIN 255.168.192.in-addr.arpa.
1          PTR      router.wireless.uoc.edu.
2          PTR      radius.wireless.uoc.edu.
10         PTR      ap.wireless.uoc.edu.
```

Anexo 6 – Archivo de configuración del OpenSSL

`/usr/local/openssl/ssl/openssl.cnf`

```
# OpenSSL example configuration file.
# This is mostly being used for generation of certificate requests.
#

# This definition stops the following lines choking if HOME isn't
# defined.
HOME = .
RANDFILE = $ENV::HOME/.rnd

# Extra OBJECT IDENTIFIER info:
# oid_file = $ENV::HOME/.oid
oid_section = new_oids

# To use this configuration file with the "-extfile" option of the
# "openssl x509" utility, name here the section containing the
# X.509v3 extensions to use:
# extensions =
# (Alternatively, use a configuration file that has only
# X.509v3 extensions in its main [= default] section.)

[ new_oids ]

# We can add new OIDs in here for use by 'ca' and 'req'.
# Add a simple OID like this:
# testoid1=1.2.3.4
# Or use config file substitution like this:
# testoid2=${testoid1}.5.6

#####
[ ca ]
default_ca = CA_default # The default ca section

#####
[ CA_default ]

dir = ./demoCA # Where everything is kept
certs = $dir/certs # Where the issued certs are kept
crl_dir = $dir/crl # Where the issued crl are kept
database = $dir/index.txt # database index file.
new_certs_dir = $dir/newcerts # default place for new certs.

certificate = $dir/cacert.pem # The CA certificate
serial = $dir/serial # The current serial number
crl = $dir/crl.pem # The current CRL
private key = $dir/private/cakey.pem # The private key
RANDFILE = $dir/private/.rand # private random number file

x509_extensions = usr_cert # The extensions to add to the cert

# Extensions to add to a CRL. Note: Netscape communicator chokes on
# V2 CRLs so this is commented out by default to leave a V1 CRL.
# crl_extensions = crl_ext
```

```
default_days = 365 # how long to certify for
default_crl_days= 30 # how long before next CRL
default_md = md5 # which md to use.
preserve = no # keep passed DN ordering

# A few difference way of specifying how similar the request should
# look. For type CA, the listed attributes must be the same, and the
# optional and supplied fields are just that :-)
policy = policy_match

# For the CA policy
[ policy_match ]
countryName = match
stateOrProvinceName = match
organizationName = match
organizationalUnitName = optional
commonName = supplied
emailAddress = optional

# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
# types.
[ policy_anything ]
countryName = optional
stateOrProvinceName = optional
localityName = optional
organizationName = optional
organizationalUnitName = optional
commonName = supplied
emailAddress = optional

#####
[ req ]
default_bits = 1024
default_keyfile = privkey.pem
distinguished name = req distinguished name
attributes = req_attributes
# The extensions to add to the self signed cert
x509_extensions = v3_ca
# Passwords for private keys if not present they will be prompted
# for
# input_password = secret
# output_password = secret

# This sets a mask for permitted string types. There are several
# options.
# default: PrintableString, T61String, BMPString.
# pkix : PrintableString, BMPString.
# utf8only: only UTF8Strings.
# nombstr : PrintableString, T61String (no BMPStrings or
# UTF8Strings).
# MASK:XXXX a literal mask value.
# WARNING: current versions of Netscape crash on BMPStrings or
# UTF8Strings
# so use this option with caution!
string_mask = nombstr

# The extensions to add to a certificate request
```

```
# req_extensions = v3_req

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

stateOrProvinceName = State or Province Name (full name)
stateOrProvinceName_default = Barcelona

localityName = Locality Name (eg, city)
localityName_default = Barcelona

0.organizationName = Organization Name (eg, company)
0.organizationName_default = UOC

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

organizationalUnitName = Organizational Unit Name (eg, section)
organizationalUnitName_default = UOC Acceso Inalambrico

commonName = Common Name (eg, YOUR name)
commonName_max = 64
commonName_default = Nombre maquina

emailAddress = Email Address
emailAddress_max = 40
emailAddress_default = email@wireless.uoc.edu

# SET-ex3 = SET extension number 3

[ req_attributes ]
challengePassword = A challenge password
challengePassword_min = 4
challengePassword_max = 20

unstructuredName = An optional company name

[ usr_cert ]

# These extensions are added when 'ca' signs a request.

# This goes against PKIX guidelines but some CAs do it and some
# software requires this to avoid interpreting an end user
# certificate as a CA.

basicConstraints=CA:FALSE

# Here are some examples of the usage of nsCertType. If it is
# omitted the certificate can be used for anything *except* object
# signing.

# This is OK for an SSL server.
# nsCertType = server

# For an object signing certificate this would be used.
```

```
# nsCertType = objsign

# For normal client use this is typical
# nsCertType = client, email

# and for everything including object signing:
# nsCertType = client, email, objsign

# This is typical in keyUsage for a client certificate.
# keyUsage = nonRepudiation, digitalSignature, keyEncipherment

# This will be displayed in Netscape's comment listbox.
nsComment = "OpenSSL Generated Certificate"

# PKIX recommendations harmless if included in all certificates.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always

# This stuff is for subjectAltName and issuerAltname.
# Import the email address.
# subjectAltName=email:copy

# Copy subject details
# issuerAltName=issuer:copy

#nsCaRevocationUrl = http://www.domain.dom/ca-crl.pem
#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName

[ v3_req ]
# Extensions to add to a certificate request
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment

[ v3_ca ]
# Extensions for a typical CA
# PKIX recommendation.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer:always

# This is what PKIX recommends but some broken software chokes on
# critical extensions.
#basicConstraints = critical,CA:true
# So we do this instead.
basicConstraints = CA:true

# Key usage: this is typical for a CA certificate. However since it
# will prevent it being used as an test self-signed certificate it
# is best left out by default.
# keyUsage = cRLSign, keyCertSign

# Some might want this also
# nsCertType = sslCA, emailCA

# Include email address in subject alt name: another PKIX
```

```
# recommendation
# subjectAltName=email:copy
# Copy issuer details
# issuerAltName=issuer:copy

# DER hex encoding of an extension: beware experts only!
# obj=DER:02:03
# Where 'obj' is a standard or added object
# You can even override a supported extension:
# basicConstraints= critical, DER:30:03:01:01:FF

[ crl_ext ]
# CRL extensions.
# Only issuerAltName and authorityKeyIdentifier make any sense in a
# CRL.

# issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always,issuer:always
```

Anexo 7 – Scripts creación de certificados

Script: CA.root

```
#!/bin/sh
SSL=/usr/local/openssl
export PATH=${SSL}/bin/:${SSL}/ssl/misc:${PATH}
export LD_LIBRARY_PATH=${SSL}/lib
rm -rf demoCA
echo "*****"
echo "Creando Entidad Certificadora"
echo "*****"
echo
openssl req -new -x509 -keyout newreq.pem -out newreq.pem -passin
pass:secreto -passout pass:secreto

openssl pkcs12 -export -in demoCA/cacert.pem -inkey newreq.pem -out
root.p12 -cacerts -passin pass:secreto -passout pass:secreto

openssl pkcs12 -in root.p12 -out root.pem -passin pass:secreto -
passout pass:secreto

openssl x509 -inform PEM -outform DER -in root.pem -out root.der

rm -rf newreq.pem
```

Script: CA.svr <NombreCertificadoServidor>

```
#!/bin/sh
SSL=/usr/local/openssl
export PATH=${SSL}/bin/:${SSL}/ssl/misc:${PATH}
export LD_LIBRARY_PATH=${SSL}/lib
echo "*****"
echo "Creando Certificado del Servidor"
echo "Cuando pregunte el Common Name, se debe introducir"
echo "el nombre del servidor."
echo "*****"
echo
openssl req -new -keyout newreq.pem -out newreq.pem -passin
pass:secreto -passout pass:secreto

openssl ca -policy policy_anything -out newcert.pem -passin
pass:secreto -key secreto -extensions xpserver_ext -extfile
xpeextensions -infile newreq.pem

openssl pkcs12 -export -in newcert.pem -inkey newreq.pem -out $1.p12
-clcerts -passin pass:secreto -passout pass:secreto

openssl pkcs12 -in $1.p12 -out $1.pem -passin pass:secreto -passout
pass:secreto

openssl x509 -inform PEM -outform DER -in $1.pem -out $1.der
```

```
rm -rf newert.pem newreq.pem
```

Script: CA.clt <NombreCertificadoCliente>

```
#!/bin/sh
SSL=/usr/local/openssl
export PATH=${SSL}/bin/:${SSL}/ssl/misc:${PATH}
export LD_LIBRARY_PATH=${SSL}/lib
echo "*****"
echo "Creando Cretificado del cliente"
echo "Cuando pregunte el Common Name, se debe introducir"
echo "el nombre del cliente, debe coincidir con el indicado"
echo "en el FreeRADIUS."
echo "*****"
echo
openssl req -new -keyout newreq.pem -out newreq.pem -passin
pass:secreto -passout pass:secreto

openssl ca -policy policy_anything -out newcert.pem -passin
pass:secreto -key secreto -extensions xpclient_ext -extfile
xpextensions -infile newreq.pem

openssl pkcs12 -export -in newcert.pem -inkey newreq.pem -out $1.p12
-clcerts -passin pass:secreto -passout pass:secreto

openssl pkcs12 -in $1.p12 -out $1.pem -passin pass:secreto -passout
pass:secreto

openssl x509 -inform PEM -outform DER -in $1.pem -out $1.der

rm -rf newcert newreq.pem
```


Anexo 8 – Archivo de configuración del OpenLDAP

`/usr/local/etc/openldap/slapd.conf`

```
# Schemas
#
# Order matters for schema loading since some schema objects
# are dependant upon attribute types defined in other
# schemas. For example
#
#     locking.schema depends on nis.schema
#
include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/cosine.schema
include /usr/local/etc/openldap/schema/nis.schema
include /usr/local/etc/openldap/schema/inetorgperson.schema
include /usr/local/etc/openldap/schema/openldap.schema

#
# Process Info
#
pidfile /var/run/slapd.pid
argsfile /var/run/slapd.args

#
# Default password hash method used by Password Modify
# Extended Operation (RFC 3052) and ldappasswd.
#
password-hash {MD5}

#
# Allow LDAPv2 clients to bind. This is needed for such
# clients as am-utils since that automounter only speaks
# LDAPv2 for the moment. Many mail clients only speak LDAPv2
# as well.
#
allow bind_v2

#
# Access Control Policy
#
#
# Allow authenticated users to change own their password,
# anonymous users to authenticate (bind), and members of the
# LDAPaccess group to read userPasswords. Deny access to
# all others.
#
access to attr=userPassword
    by self write
    by anonymous auth
    by * none
```

```

access to dn=""
    by * read

access to *
    by self write
    by users read
    by anonymous auth

sasl-host      radius.wireless.uoc.edu
sasl-realm     wireless.uoc.edu
sasl-secprops  noplain,noanonymous

#
# TLS Security Configuration
#
TLSCipherSuite HIGH:MEDIUM, 3DES:SHA1:+SSL2
TLSCertificateFile /etc/certificados/radius.wireless.uoc.edu.pem
TLSCertificateKeyFile /etc/certificados/radius.wireless.uoc.edu.key
TLSCACertificateFile /etc/certificados/root.pem

#####
#          BDB BACKEND CONFIGURATION          #
#####

#
# Note: Berkeley Database configuration information is also
#       store in the DB_CONFIG file under the bdb directory,
#       /usr/local/var/openldap-data
#
#
# Base Database Type and Domain
#
database      bdb
suffix        "dc=wireless,dc=uoc,dc=edu"

#
# Distinguished Name allowed complete
# access to database backend.
#
rootdn        "cn=Manager,ou=usuarios,dc=wireless,dc=uoc,dc=edu"
rootpw        {MD5}4gGZTcqTIPyUM2YDsc/JcA==

#
# Database Directory
#
directory     /usr/local/var/openldap-data

#
# Set the entry cache size to 5000.
#
# This value is separate from the set_cachesize value set in
# the DB_CONFIG file under the bdb directory. That value
# should be set as well to optimize database caching for the
# Berkeley DB subsystem.
#
cachesize     5000

#

```

```
# Set transactional checkpoint (writing of changed data to
# to disk) to occur when either
#
# 512 Kilobytes of data have been written to the bdb sub-
# system.
#
# 720 Minutes have passed since the last checkpoint.
#
#
checkpoint          512      720

#
# Database Indexes
#
index  uid          eq
index  cn           pres,eq,sub
index  sn           pres,eq,sub
index  objectClass pres,eq
```

Anexo 9 – Archivo configuración clientes OpenLDAP

`/usr/local/etc/openldap/lapd.conf`

```
# $OpenLDAP: pkg/ldap/libraries/libldap/ldap.conf,  
#v 1.9 2000/09/04 19:57:01 kurt Exp $  
#  
# LDAP Defaults  
#  
host radius.wireless.uoc.edu  
base ou=usuarios,dc=wireless,dc=uoc,dc=edu  
uri ldap:/// ldaps:///  
  
ldap_version 3  
  
binddn cn=Manager,ou=usuarios,dc=wireless,dc=uoc,dc=edu  
bindpw secreto  
  
TIMELIMIT 25  
SIZELIMIT 12  
  
#  
# TLS Security Configuration  
#  
tls_checkpeer yes  
tls_cacert /etc/certificados/root.pem  
tls_request allow  
ssl start_tls
```

Anexo 10 – Carga inicial de datos en OpenLDAP

```
dn: dc=wireless,dc=uoc,dc=edu
objectclass: dcObject
objectclass: organization
o: UOC PFC Wireless
dc: wireless

dn: ou=usuarios,dc=wireless,dc=uoc,dc=edu
objectClass: organizationalUnit
ou: usuarios

dn: uid=josep,ou=usuarios,dc=wireless,dc=uoc,dc=edu
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
uid: josep
homeDirectory: /home/josep
loginShell: /bin/bash
uidNumber: 501
gidNumber: 100
userPassword: {MD5}x4cLjCRM+SK338aOQuF6dA==
cn: Josep Garcia
givenname: Josep
sn: Garcia
o: UOC PFC Wireless
mail: josep@wireless.uoc.edu

dn: uid=jluque,ou=usuarios,dc=wireless,dc=uoc,dc=edu
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
uid: jluque
homeDirectory: /home/jluque
loginShell: /bin/bash
uidNumber: 500
gidNumber: 100
cn: Jorge Luque
givenname: Jorge
sn: Luque
o: UOC PFC Wireless
userPassword: {MD5}FTe930JI6MMNobu4nYYNqg==
mail: jluque@wireless.uoc.edu

dn: uid=Manager,ou=usuarios,dc=wireless,dc=uoc,dc=edu
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: Manager
cn: Manager
givenname: Manager
```

```
sn: Manager  
o: UOC PFC Wireless  
userPassword: {MD5}4gGZTcqTIPyUM2YDsc/JcA==  
mail: Manager@wireless.uoc.edu
```

Anexo 11 – Archivo configuración FreeRADIUS: eap.conf

```
# Whatever you do, do NOT set 'Auth-Type := EAP'. The server
# is smart enough to figure this out on its own. The most
# common side effect of setting 'Auth-Type := EAP' is that the
# users then cannot use ANY other authentication method.
#
eap {
    # Invoke the default supported EAP type when
    # EAP-Identity response is received.
    #
    # The incoming EAP messages DO NOT specify which EAP
    # type they will be using, so it MUST be set here.
    #
    # For now, only one default EAP type may be used at a time.
    #
    # If the EAP-Type attribute is set by another module,
    # then that EAP type takes precedence over the
    # default type configured here.
    #
    default_eap_type = tls

    # A list is maintained to correlate EAP-Response
    # packets with EAP-Request packets. After a
    # configurable length of time, entries in the list
    # expire, and are deleted.
    #
    timer_expire      = 60

    # There are many EAP types, but the server has support
    # for only a limited subset. If the server receives
    # a request for an EAP type it does not support, then
    # it normally rejects the request. By setting this
    # configuration to "yes", you can tell the server to
    # instead keep processing the request. Another module
    # MUST then be configured to proxy the request to
    # another RADIUS server which supports that EAP type.
    #
    # If another module is NOT configured to handle the
    # request, then the request will still end up being
    # rejected.
    ignore_unknown_eap_types = no

    # Cisco AP1230B firmware 12.2(13)JA1 has a bug. When given
    # a User-Name attribute in an Access-Accept, it copies one
    # more byte than it should.
    #
    # We can work around it by configurably adding an extra
    # zero byte.
    cisco_accounting_username_bug = no

    # Supported EAP-types

    #
    # We do NOT recommend using EAP-MD5 authentication
    # for wireless connections. It is insecure, and does
```

```
# not provide for dynamic WEP keys.
#
#md5 {
#}

# Cisco LEAP
#
# We do not recommend using LEAP in new deployments. See:
# http://www.securiteam.com/tools/5TP012ACKE.html
#
# Cisco LEAP uses the MS-CHAP algorithm (but not
# the MS-CHAP attributes) to perform it's authentication.
#
# As a result, LEAP *requires* access to the plain-text
# User-Password, or the NT-Password attributes.
# 'System' authentication is impossible with LEAP.
#
# leap {
# }

## EAP-TLS
#
# To generate ctest certificates, run the script
#
#     ../scripts/certs.sh
#
# The documents on http://www.freeradius.org/doc
# are old, but may be helpful.
#
# See also:
#
# http://www.dslreports.com/forum/remark,9286052~mode=flat
#
# tls {
#     private_key_password = secreto
#     private_key_file =
#         /etc/certificados/radius.wireless.uoc.edu.pem
#
#     # If Private key & Certificate are located in
#     # the same file, then private_key_file &
#     # certificate_file must contain the same file
#     # name.
#     certificate_file =
#         /etc/certificados/radius.wireless.uoc.edu.pem
#
#     # Trusted Root CA list
#     CA_file = /etc/certificados/root.pem
#
#     dh_file = /etc/certificados/dh
#     random_file = /etc/certificados/random
#
#     #
#     # This can never exceed the size of a RADIUS
#     # packet (4096 bytes), and is preferably half
#     # that, to accomodate other attributes in
#     # RADIUS packet. On most APs the MAX packet
#     # length is configured between 1500 - 1600
#     # In these cases, fragment size should be
#     # 1024 or less.
```



```

#
fragment_size = 1750

# include_length is a flag which is
# by default set to yes If set to
# yes, Total Length of the message is
# included in EVERY packet we send.
# If set to no, Total Length of the
# message is included ONLY in the
# First packet of a fragment series.
#
include_length = yes

# Check the Certificate Revocation List
#
# 1) Copy CA certificates and CRLs to same directory.
# 2) Execute 'c_rehash <CA certs&CRLs Directory>'.
#    'c_rehash' is OpenSSL's command.
# 3) Add 'CA_path=<CA certs&CRLs directory>'
#     to radiusd.conf's tls section.
# 4) uncomment the line below.
# 5) Restart radiusd
#check_crl = yes

#
# If check_cert_cn is set, the value will
# be xlated and checked against the CN
# in the client certificate. If the values
# do not match, the certificate verification
# will fail rejecting the user.
#
#check_cert_cn = %{User-Name}
}

# The TTLS module implements the EAP-TTLS protocol,
# which can be described as EAP inside of Diameter,
# inside of TLS, inside of EAP, inside of RADIUS...
#
# Surprisingly, it works quite well.
#
# The TTLS module needs the TLS module to be installed
# and configured, in order to use the TLS tunnel
# inside of the EAP packet. You will still need to
# configure the TLS module, even if you do not want
# to deploy EAP-TLS in your network. Users will not
# be able to request EAP-TLS, as it requires them to
# have a client certificate. EAP-TTLS does not
# require a client certificate.
#
ttls {
# The tunneled EAP session needs a default
# EAP type which is separate from the one for
# the non-tunneled EAP module. Inside of the
# TTLS tunnel, we recommend using EAP-MD5.
# If the request does not contain an EAP
# conversation, then this configuration entry
# is ignored.
default_eap_type = md5
# The tunneled authentication request does

```

```

# not usually contain useful attributes
# like 'Calling-Station-Id', etc. These
# attributes are outside of the tunnel,
# and normally unavailable to the tunneled
# authentication request.
#
# By setting this configuration entry to
# 'yes', any attribute which NOT in the
# tunneled authentication request, but
# which IS available outside of the tunnel,
# is copied to the tunneled request.
#
# allowed values: {no, yes}
copy_request_to_tunnel = no

# The reply attributes sent to the NAS are
# usually based on the name of the user
# 'outside' of the tunnel (usually
# 'anonymous'). If you want to send the
# reply attributes based on the user name
# inside of the tunnel, then set this
# configuration entry to 'yes', and the reply
# to the NAS will be taken from the reply to
# the tunneled request.
#
# allowed values: {no, yes}
use_tunneled_reply = no
}

#
# The tunneled EAP session needs a default EAP type
# which is separate from the one for the non-tunneled
# EAP module. Inside of the TLS/PEAP tunnel, we
# recommend using EAP-MS-CHAPv2.
#
# The PEAP module needs the TLS module to be installed
# and configured, in order to use the TLS tunnel
# inside of the EAP packet. You will still need to
# configure the TLS module, even if you do not want
# to deploy EAP-TLS in your network. Users will not
# be able to request EAP-TLS, as it requires them to
# have a client certificate. EAP-PEAP does not
# require a client certificate.
#
peap {
# The tunneled EAP session needs a default
# EAP type which is separate from the one for
# the non-tunneled EAP module. Inside of the
# PEAP tunnel, we recommend using MS-CHAPv2,
# as that is the default type supported by
# Windows clients.
    default_eap_type = mschapv2
}

#
# This takes no configuration.
#
# Note that it is the EAP MS-CHAPv2 sub-module, not
# the main 'mschap' module.

```

```
#
# Note also that in order for this sub-module to work,
# the main 'mschap' module MUST ALSO be configured.
#
# This module is the *Microsoft* implementation of MS-CHAPv2
# in EAP. There is another (incompatible) implementation
# of MS-CHAPv2 in EAP by Cisco, which FreeRADIUS does not
# currently support.
#
mschapv2 {
}
}
```

Anexo 12 – Archivo de mapeo FreeRADIUS-LDAP:

ldap.attrmap

```
# Mapping of RADIUS dictionary attributes to LDAP directory
# attributes to be used by LDAP authentication and authorization
# module (rlm_ldap)
#
# Format:
#   ItemType          RADIUS-Attribute-Name          ldapAttributeName
#
# Where:
#   ItemType          = checkItem or replyItem
#   RADIUS-Attribute-Name = attribute name in RADIUS dictionary
#   ldapAttributeName = attribute name in LDAP schema
#
# If $GENERIC$ is specified as RADIUS-Attribute-Name, the line
# specifies a LDAP attribute which can be used to store any RADIUS
# attribute/value-pair in LDAP directory.
#
# You should edit this file to suit it to your needs.
#
```

checkItem	\$GENERIC\$	radiusCheckItem
replyItem	\$GENERIC\$	radiusReplyItem
checkItem	Auth-Type	radiusAuthType
checkItem	Simultaneous-Use	radiusSimultaneousUse
checkItem	Called-Station-Id	radiusCalledStationId
checkItem	Calling-Station-Id	radiusCallingStationId
checkItem	LM-Password	lmPassword
checkItem	NT-Password	ntPassword
checkItem	SMB-Account-CTRL-TEXT	acctFlags
checkItem	Expiration	radiusExpiration
replyItem	Service-Type	radiusServiceType
replyItem	Framed-Protocol	radiusFramedProtocol
replyItem	Framed-IP-Address	radiusFramedIPAddress
replyItem	Framed-IP-Netmask	radiusFramedIPNetmask
replyItem	Framed-Route	radiusFramedRoute
replyItem	Framed-Routing	radiusFramedRouting
replyItem	Filter-Id	radiusFilterId
replyItem	Framed-MTU	radiusFramedMTU
replyItem	Framed-Compression	radiusFramedCompression
replyItem	Login-IP-Host	radiusLoginIPHost
replyItem	Login-Service	radiusLoginService
replyItem	Login-TCP-Port	radiusLoginTCPPort
replyItem	Callback-Number	radiusCallbackNumber
replyItem	Callback-Id	radiusCallbackId
replyItem	Framed-IPX-Network	radiusFramedIPXNetwork
replyItem	Class	radiusClass
replyItem	Session-Timeout	radiusSessionTimeout
replyItem	Idle-Timeout	radiusIdleTimeout
replyItem	Termination-Action	radiusTerminationAction
replyItem	Login-LAT-Service	radiusLoginLATService
replyItem	Login-LAT-Node	radiusLoginLATNode

replyItem	Login-LAT-Group	radiusLoginLATGroup
replyItem	Framed-AppleTalk-Link	radiusFramedAppleTalkLink
replyItem	Framed-AppleTalk-Network	radiusFramedAppleTalkNetwork
replyItem	Framed-AppleTalk-Zone	radiusFramedAppleTalkZone
replyItem	Port-Limit	radiusPortLimit
replyItem	Login-LAT-Port	radiusLoginLATPort

Anexo 13 – Configuración cortafuegos

```
#!/bin/sh -x
#
log() {
    test -x "$LOGGER" && $LOGGER -p info "$1"
}

va_num=1
add_addr() {
    addr=$1
    nm=$2
    dev=$3

    type=""
    aadd=""

    L=`$IP -4 link ls $dev | grep "$dev:"`
    if test -n "$L"; then
        OIFS=$IFS
        IFS=" /:,<"
        set $L
        type=$4
        IFS=$OIFS

        L=`$IP -4 addr ls $dev to $addr | grep " inet "`
        if test -n "$L"; then
            OIFS=$IFS
            IFS=" /"
            set $L
            aadd=$2
            IFS=$OIFS
        fi
    fi
    if test -z "$aadd"; then
        if test "$type" = "POINTOPOINT"; then
            $IP -4 addr add $addr dev $dev scope global label
$dev:FWB${va_num}
            va_num=`expr $va_num + 1`
        fi
        if test "$type" = "BROADCAST"; then
            $IP -4 addr add $addr/$nm dev $dev brd + scope global label
$dev:FWB${va_num}
            va_num=`expr $va_num + 1`
        fi
    fi
}

getaddr() {
    dev=$1
    name=$2
    L=`$IP -4 addr show dev $dev | grep inet`
    test -z "$L" && {
        eval "$name=''"
        return
    }
    OIFS=$IFS
}
```

```
IFS=" /"
set $L
eval "$name=$2"
IFS=$OIFS
}

getinterfaces() {
NAME=$1
$IP link show | grep -E "$NAME[^\ ]*:" | while read L; do
OIFS=$IFS
IFS=":"
set $L
IFS=$OIFS
echo $2
done
}

LSMOD="/sbin/lsmmod"
MODPROBE="/sbin/modprobe"
IPTABLES="/sbin/iptables"
IP="/sbin/ip"
LOGGER="/usr/bin/logger"

INTERFACES="eth0 eth0 ipsec0 lo "
for i in $INTERFACES ; do
$IP link show "$i" > /dev/null 2>&1 || {
echo Interface $i does not exist
exit 1
}
done

echo 30 > /proc/sys/net/ipv4/tcp_fin_timeout

echo 1800 > /proc/sys/net/ipv4/tcp_keepalive_intvl

add_addr 192.168.255.2 24 eth0
$IP link set eth0 up
add_addr 172.16.0.1 24 eth0
$IP link set eth0 up
add_addr 172.16.0.1 24 ipsec0
$IP link set ipsec0 up
add_addr 127.0.0.1 8 lo
$IP link set lo up

$IPTABLES -P OUTPUT DROP
$IPTABLES -P INPUT DROP
$IPTABLES -P FORWARD DROP

cat /proc/net/ip_tables_names | while read table; do
```

```

$IPTABLES -t $table -L -n | while read c chain rest; do
    if test "X$c" = "XChain" ; then
        $IPTABLES -t $table -F $chain
    fi
done
$IPTABLES -t $table -X
done

MODULE_DIR="/lib/modules/`uname -r`/kernel/net/ipv4/netfilter/"
MODULES=`(cd $MODULE_DIR; ls *_conntrack_* *_nat_* | sed
's/\.\o.*$//; s/\.ko$//')`
for module in $(echo $MODULES); do
    if $LSMOD | grep ${module} >/dev/null; then continue; fi
    $MODPROBE ${module} || exit 1
done

log "Activating firewall script generated Sat Jun 12 15:59:23 2004
CEST by root"

#
# Rule 0(NAT)
#
#
$IPTABLES -t nat -A POSTROUTING -o eth0 -s 192.168.255.0/24 -j SNAT
--to-source 172.16.0.1
#
#

$IPTABLES -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

#
# Rule 0(eth0)
#
#
#
$IPTABLES -A INPUT -i eth0 -m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -i eth0 -m state --state NEW -j ACCEPT
$IPTABLES -A OUTPUT -o eth0 -m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -o eth0 -m state --state NEW -j ACCEPT
#
# Rule 0(lo)
#
# allow everything on loopback
#
$IPTABLES -A INPUT -i lo -j ACCEPT
$IPTABLES -A OUTPUT -o lo -j ACCEPT
#
# Rule 0(global)
#
# block fragments
#
$IPTABLES -N RULE_0

```



```

$IPTABLES -A OUTPUT -p all -f -j RULE_0
$IPTABLES -A INPUT -p all -f -j RULE_0
$IPTABLES -A FORWARD -p all -f -j RULE_0
$IPTABLES -A RULE_0 -j LOG --log-level 4 --log-prefix "RULE 0 --
DENY " --log-tcp-sequence --log-tcp-options --log-ip-options
$IPTABLES -A RULE_0 -j DROP
#
# Rule 1(global)
#
# ssh access to firewall
#
$IPTABLES -N RULE_1
$IPTABLES -A INPUT -p tcp -s 192.168.255.0/24 -d 192.168.255.2 --
destination-port 22 -m state --state NEW -j RULE_1
$IPTABLES -A INPUT -p tcp -s 192.168.255.0/24 -d 172.16.0.1 --
destination-port 22 -m state --state NEW -j RULE_1
$IPTABLES -A RULE_1 -j LOG --log-level 6 --log-prefix "RULE 1 --
ACCEPT " --log-tcp-sequence --log-tcp-options --log-ip-options
$IPTABLES -A RULE_1 -j ACCEPT
#
# Rule 2(global)
#
# 'masquerading' rule
#
$IPTABLES -N RULE_2
$IPTABLES -A INPUT -s 192.168.255.0/24 -m state --state NEW -j
RULE_2
$IPTABLES -A OUTPUT -s 192.168.255.0/24 -m state --state NEW -j
RULE_2
$IPTABLES -A FORWARD -s 192.168.255.0/24 -m state --state NEW -j
RULE_2
$IPTABLES -A RULE_2 -j LOG --log-level 6 --log-prefix "RULE 2 --
ACCEPT " --log-tcp-sequence --log-tcp-options --log-ip-options
$IPTABLES -A RULE_2 -j ACCEPT
#
# Rule 3(global)
#
#
#
#
$IPTABLES -N Cid40CAF72A.0
$IPTABLES -A INPUT -s 172.16.0.0/24 -m state --state NEW -j
Cid40CAF72A.0
$IPTABLES -A OUTPUT -s 172.16.0.0/24 -m state --state NEW -j
Cid40CAF72A.0
$IPTABLES -A FORWARD -s 172.16.0.0/24 -m state --state NEW -j
Cid40CAF72A.0
$IPTABLES -A Cid40CAF72A.0 -d 192.168.255.0/24 -j RETURN
$IPTABLES -A Cid40CAF72A.0 -d 172.16.0.0/24 -j RETURN
$IPTABLES -N RULE_3_3
$IPTABLES -A Cid40CAF72A.0 -m state --state NEW -j RULE_3_3
$IPTABLES -A RULE_3_3 -j LOG --log-level 6 --log-prefix "RULE 3 --
ACCEPT " --log-tcp-sequence --log-tcp-options --log-ip-options
$IPTABLES -A RULE_3_3 -j ACCEPT
#
# Rule 4(global)
#
#
#
#
$IPTABLES -N RULE_4

```

```

$IPTABLES -A OUTPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 3 -m state --state NEW -j RULE_4
$IPTABLES -A OUTPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
-m state --state NEW -j RULE_4
$IPTABLES -A OUTPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 3/1 -m state --state NEW -j RULE_4
$IPTABLES -A OUTPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 0/0 -m state --state NEW -j RULE_4
$IPTABLES -A OUTPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 8/0 -m state --state NEW -j RULE_4
$IPTABLES -A OUTPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 3/3 -m state --state NEW -j RULE_4
$IPTABLES -A OUTPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 11/0 -m state --state NEW -j RULE_4
$IPTABLES -A OUTPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 11/1 -m state --state NEW -j RULE_4
$IPTABLES -A OUTPUT -p tcp -m multiport -s 172.16.0.0/24 -d
192.168.255.0/24 --destination-ports
53,139,135,42,445,88,389,636,3268,3269,110,22,80,21,25 -m state --
state NEW -j RULE_4
$IPTABLES -A OUTPUT -p udp -m multiport -s 172.16.0.0/24 -d
192.168.255.0/24 --destination-ports 68,67,53,138,137,88 -m state
--state NEW -j RULE_4
$IPTABLES -A OUTPUT -p 50 -s 172.16.0.0/24 -d 192.168.255.0/24 -m
state --state NEW -j RULE_4
$IPTABLES -A OUTPUT -p 51 -s 172.16.0.0/24 -d 192.168.255.0/24 -m
state --state NEW -j RULE_4
$IPTABLES -A INPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24 -
-icmp-type 3 -m state --state NEW -j RULE_4
$IPTABLES -A INPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24 -
m state --state NEW -j RULE_4
$IPTABLES -A INPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24 -
-icmp-type 3/1 -m state --state NEW -j RULE_4
$IPTABLES -A INPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24 -
-icmp-type 0/0 -m state --state NEW -j RULE_4
$IPTABLES -A INPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24 -
-icmp-type 8/0 -m state --state NEW -j RULE_4
$IPTABLES -A INPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24 -
-icmp-type 3/3 -m state --state NEW -j RULE_4
$IPTABLES -A INPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24 -
-icmp-type 11/0 -m state --state NEW -j RULE_4
$IPTABLES -A INPUT -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24 -
-icmp-type 11/1 -m state --state NEW -j RULE_4
$IPTABLES -A INPUT -p tcp -m multiport -s 172.16.0.0/24 -d
192.168.255.0/24 --destination-ports
53,139,135,42,445,88,389,636,3268,3269,110,22,80,21,25 -m state --
state NEW -j RULE_4
$IPTABLES -A INPUT -p udp -m multiport -s 172.16.0.0/24 -d
192.168.255.0/24 --destination-ports 68,67,53,138,137,88 -m state
--state NEW -j RULE_4
$IPTABLES -A INPUT -p 50 -s 172.16.0.0/24 -d 192.168.255.0/24 -m
state --state NEW -j RULE_4
$IPTABLES -A INPUT -p 51 -s 172.16.0.0/24 -d 192.168.255.0/24 -m
state --state NEW -j RULE_4
$IPTABLES -A FORWARD -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 3 -m state --state NEW -j RULE_4
$IPTABLES -A FORWARD -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
-m state --state NEW -j RULE_4

```

```
$IPTABLES -A FORWARD -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 3/1 -m state --state NEW -j RULE_4
$IPTABLES -A FORWARD -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 0/0 -m state --state NEW -j RULE_4
$IPTABLES -A FORWARD -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 8/0 -m state --state NEW -j RULE_4
$IPTABLES -A FORWARD -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 3/3 -m state --state NEW -j RULE_4
$IPTABLES -A FORWARD -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 11/0 -m state --state NEW -j RULE_4
$IPTABLES -A FORWARD -p icmp -s 172.16.0.0/24 -d 192.168.255.0/24
--icmp-type 11/1 -m state --state NEW -j RULE_4
$IPTABLES -A FORWARD -p tcp -m multiport -s 172.16.0.0/24 -d
192.168.255.0/24 --destination-ports
53,139,135,42,445,88,389,636,3268,3269,110,22,80,21,25 -m state --
state NEW -j RULE_4
$IPTABLES -A FORWARD -p udp -m multiport -s 172.16.0.0/24 -d
192.168.255.0/24 --destination-ports 68,67,53,138,137,88 -m state
--state NEW -j RULE_4
$IPTABLES -A FORWARD -p 50 -s 172.16.0.0/24 -d 192.168.255.0/24 -
m state --state NEW -j RULE_4
$IPTABLES -A FORWARD -p 51 -s 172.16.0.0/24 -d 192.168.255.0/24 -
m state --state NEW -j RULE_4
$IPTABLES -A RULE_4 -j LOG --log-level 6 --log-prefix "RULE 4 --
ACCEPT " --log-tcp-sequence --log-tcp-options --log-ip-options
$IPTABLES -A RULE_4 -j ACCEPT
#
# Rule 5(global)
#
# 'catch all' rule
#
$IPTABLES -N RULE_5
$IPTABLES -A OUTPUT -j RULE_5
$IPTABLES -A INPUT -j RULE_5
$IPTABLES -A FORWARD -j RULE_5
$IPTABLES -A RULE_5 -j LOG --log-level 6 --log-prefix "RULE 5 --
DENY " --log-tcp-sequence --log-tcp-options --log-ip-options
$IPTABLES -A RULE_5 -j DROP
#
#
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Anexo 14 – Archivo configuración IPsec: ipsec.conf

```
# /etc/ipsec.conf - FreeS/WAN IPsec configuration file
# RCSID $Id: ipsec.conf.in,v 1.12 2004/01/20 19:37:13 sam Exp $

version      2.0    # conforms to second version of ipsec.conf
specification

# basic configuration
config setup
    interfaces="ipsec0=eth0:1"
    klipsdebug=none
    plutodebug=none
    uniqueids=yes

conn %default
    keyingtries=1
    compress=yes
    disablearrivalcheck=no
    authby=rsasig
    leftprotoport=17/0
    lefttrsasigkey=%cert
    rightprotoport=17/1701
    righttrsasigkey=%cert

conn josep
    right=%any
    rightcert=josep.pem
    rightid=@clientes
    left=172.16.0.1
    leftsubnet=192.168.255.0/24
    leftcert=radius.wireless.uoc.edu.pem
    auto=add
    pfs=yes

#Disable Opportunistic Encryption
include /etc/ipsec.d/examples/no_oe.conf
```