CENTRO DE INVESTIGACIÓN Y DE
ESTUDIOS AVANZADOS DEL
INSTITUTO POLITÉCNICO NACIONAL

Unidad Zacatenco
Departamento de Computación

Punto de Acceso Seguro en IPv6

Tesis presentada por

José Nefi Gamboa Castañeda

para obtener el grado de

Maestro en Ciencias en Computación

Director de tesis

Dr. Luis Gerardo de la Fraga

Ciudad de México, México **Noviembre de 2016**

Dedicado a mi amada esposa Yunn, quien me ha dado su apoyo incondicional en cada momento, desafío, desvelo, logro, meta, así como su sostén en cada una de mis crisis.

Agradecimientos

Agradezco al Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional y al Departamento de Computación de la misma institución por haberme aceptado y permitirme concluir mis estudios de maestría, así mismo al Consejo Nacional de Ciencia y Tecnología (CONACYT) por haberme apoyado con dos años de beca.

Agradezco al proyecto 168357 de Conacyt por su apoyo en la realización de este trabajo de tesis.

Agradezco al Dr. Luis Gerardo de la Fraga por instruirme, motivarme y enriquecerme con su conocimiento sobre los temas contenidos en esta tesis, así como la paciencia que tuvo conmigo en el desarrollo de la misma, con lo cual he podido culminar mis estudios de maestría.

Agradezco a los investigadores de éste departamento por los contenidos aprendidos en cada materia que cursé, así como la resolución de mis dudas a lo largo de mis estudios de maestría, al jefe de departamento y al coordinador por los excelentes seminarios que complementaron mi aprendizaje, así como al personal administrativo por facilitarme los trámites necesarios para la obtención de mi grado.

Agradezco a mis compañeros de la generación 2014, con quienes compartí estos dos años, además de aprender de ellos varios contenidos importantes tanto en clases como en mis proyectos, así como su ánimo en cada uno de ellos.

Agradezco a mi mamá Miriam y a mis suegros Noé y Aurora por darme su apoyo incondicional en éste período, así como a mi difunto papá José, quien siempre me motivó a seguir estudiando. Así mismo a mi hermana Abish, mis cuñados, a Lehi y a cada una de las personas que me han acompañado. Y sobretodo agradezco a Dios por el amor recibido por cada uno de ellos, así también por permitirme tener salud y claridad de mente para desempeñarme en forma correcta en cada examen, tarea, proyecto y trabajo desarrollado en cada uno de mis estudios y en la vida en general.

Resumen

Un punto de acceso (PA) es una computadora que conecta una red inalámbrica con otras redes, incluida Internet. En esta tesis se diseña y construye un PA seguro que usa el protocolo IPv6 para conectar a los usuarios de la red inalámbrica a Internet, que en la actualidad, en el año 2016, aún utiliza el protocolo IPv4. IPv6 es más seguro y eficiente que IPv4.

Dos características del PA seguro es que solo podrán acceder a Internet los usuarios que puedan autenticarse y que tengan sus computadoras registradas.

Así, en esta tesis se revisan los mecanismos de transición entre IPv6 e IPv4 (doble pila, túnel y traducción de los protocolos); también se examinan los protocolos de seguridad para el acceso a las redes inalámbricas de los estándares 802.11, 802.11i y 802.1X con el objetivo de identificar las fortalezas y debilidades que éstos presentan.

Para el diseño del PA seguro se hace uso de los protocolos de traducción de IPv6 a IPv4 (TDR64) y el Servicio de Nombres de Dominio de IPv6 a IPv4 (DNS64), con los cuales se proporciona a los usuarios de la red interna IPv6 un acceso nativo a las direcciones y nombres de la red externa en IPv4. Además, se incorpora el protocolo DHCPv6 para la asignación dinámica de direcciones IPv6 a los clientes.

En materia de seguridad, el PA seguro usa la autenticación con el algoritmo robusto WPA2; junto con éste se usa el protocolo de extensiones de autenticación con tarjeta de tokens genéricos, que permite mantener las contraseñas cifradas en el mismo PA. Obviamente, también se usa el servidor de autenticación Radius. Los usuarios de la red deben registrar las direcciones MAC (asociadas a sus tarjetas de red) de sus computadoras, y el acceso a estas computadoras registradas se habilita en el cortafuegos también incorporado al PA seguro.

Se presentan todas las configuraciones del software para la creación del PA seguro y las pruebas que se realizaron para demostrar su utilidad.

Abstract

An access point (AP) is a computer that connects a wireless network with other networks, including Internet. In this thesis, it is designed and constructed a secure AP that uses the IPv6 protocol to connect the wireless network users to Internet, which currently, in 2016, still uses the IPv4 protocol. IPv6 is more secure and efficient than IPv4.

Two features included in the secure AP are: Internet access is granted only to users who can be authenticated, and also those users must have their computers registered within the AP.

Thus, in this thesis it is reviewed the IPv6 to IPv4 transition mechanisms (double stack, tunneling and protocol translation); also, it is examined the security protocolos for the wireless network access of standards 802.11, 802.11i and 802.1X with the objective of identifing the strengths and weaknesses that they have.

For the secure AP design are used the IPv6 to IPv4 translation (NAT64) and the IPv6 to IPv4 Domain Names Service (DNS64) protocols, to provide a native access to IPv6 internal network users to the addresses and names of the external IPv4 network. Moreover, it is added the DHCPv6 protocol for dynamic IPv6 addresses assignment for clients.

For security, the secure AP uses the authentication with the robust WPA2 algorithm; along with the extensible authentication protocol with generic tokens card, which allows to keep the encrypted passwords within the AP. Obviously, also it is included the Radius authentication server. The network users must register their MAC addresses (associated to their network cards) of their computers, and the access for these registered computers are enabled in the firewall, also built in the secure AP.

At the end of this thesis, it is presented all software configurations for the development of the secure AP, and the set of tests performed to shown its utility.

Índice general

Índice de figuras

Índice de cuadros

Índice de listas

Acrónimos y abreviaturas

A Registros de Dirección

AAAA Registros de Dirección para IPv6

AD Datos de autenticación

AES Estándar de Cifrado Avanzado

ATA Accesorio de Tecnología Avanzada

BIOS Sistema Básico de Entrada y Salida

BOOTP Protocolo *Bootstrap*

CBC Encadenado de Bloques de Cifrado

CD Revisión Deshabilitada

CCMP Protocolo MAC en Modo Contador-CBC

CHAP Protocolo de Autenticación de Desafío con Apretón de Manos

CIM Código de Integridad de Mensaje

COM2 Puerto de Comunicación versión 2

DHCP Protocolo de Configuración Dinámica del Anfitrión

DHCPv6 Protocolo de Configuración Dinámica del Anfitrión en IPv6

DIN *Deutsches Institut für Normung*

DMA Acceso Directo a Memoria

DNSKEY Clave Pública del Sistema de Nombres de Dominios

DNSSEC Extensiones de Seguridad del Sistema de Nombres de Dominios

DoS Denegación de Servicio

DS Delegación de Firmante

EA Encabezado Autenticado

EAP Protocolo de Extensiones de Autenticación

EC Encapsulamiento de la Carga

ECP Puerto de Capacidad Mejorado

EDNS Extensión de DNS

EIDE IDE Mejorado

EPP Puerto Paralelo Mejorado

FMS *Flurer, Mantin and Shamir*

GTC Tarjeta de *Tokens* Genéricos

HE hombre de enmedio

HTTP Protocolo de Transferencia de Hiper-texto

IAID Identificador de Asociación de Identidad

IATA Asociación de Identidad para Direcciones Temporales

ICANN Corporativo de Internet para la Asignación de Nombres y Números

ICMP Protocolo de Mensajes de Control de Internet

ICMPv6 Protocolo de Mensajes de Control de Internet versión 6

IDE Dispositivo Electrónico Integrado

IPsec Protocolo de Internet Seguro

LEAP Protocolo de Extensiones de Autenticación de Bajo Peso

NAS Servidor de Acceso a la Red

NSEC Siguiente Seguro

PAP Protocolo de Autenticación de Contraseñas

PCI Interconexión de Componentes Periféricos

PEAP Protocolo de Extensiones Protegidas de Autenticación

PIO Entradas y Salidas Programadas

PMK Par de Claves Maestras

PPP Protocolo Punto a Punto

PSI Proveedor de Servicio de Internet

PSK Clave Pre-Compartida

PTW *Pyshkin, Tews, Weinmann*

RAM Memoria de Acceso Aleatorio

RR Registro de Recurso

RRSIG Firma de Registro de Recurso

SBC Computadora en una Sola Tarjeta

SHA Algoritmo de Digesto Seguro

SLAAC Autoconfiguración de Dirección Sin Estado

SOA Inicio de Autoridad

SSID Identificador de Conjunto de Servicios

SPP Puerto Paralelo Estándar

TCP Protocolo de Control de Transmisión

TDR64 Traductor de Direcciones de Red 6 a 4

TKIP Protocolo de Integridad de Claves Temporales

TLD Servidores Autoritativos y Recursivos de Nivel Superior

UDP Protocolo de Datagramas de Usuario

UTM Unidad de Transmisión Máxima

VI vector de inicialización

WPA Acceso Protegido WiFi

WEP Privacidad Equivalente a Cableado

XOR O exclusiva

Capítulo 1

Introducción

En la actualidad, las redes de área local inalámbricas han alcanzado gran popularidad debido a la reducción de costos y disponibilidad que los medios inalámbricos ofrecen. En ellas se concede el acceso a los usuarios que desean hacer uso de la red a través de un punto de acceso inalámbrico, el cual consiste en el hardware que permite la comunicación entre los clientes inalámbricos y las redes alámbricas. La red alámbrica a su vez entabla la comunicación entre el punto de acceso inalámbrico y la red exterior, que puede ser Internet u otras redes.

En esta tesis llamaremos "punto de acceso" al software y hardware que garantizará el acceso a Internet de una red inalámbrica local. En forma general, el punto de acceso consta de una computadora, el punto de acceso inalámbrico y todo el software que garantiza la seguridad y acceso a Internet de los usuarios de la red inalámbrica.

En este trabajo se quieren explorar los beneficios de las redes en IPv6, su seguridad y rendimiento, con la construcción de una puerta de acceso de una red inalámbrica a Internet. IPv6 simplifica la cabecera de los datagramas y por ello mejora el rendimiento de la red. En seguridad, se asigna una dirección única para cada cliente protegiendo su identidad gozando de mayores opciones para asignar su dirección en contraste con las direcciones privadas en IPv4 [61].

Aún con los beneficios que ofrece el protocolo IPv6, es necesario incorporar un mecanismo de traducción entre la red interna en IPv6 y la red exterior. Esto se debe a que en el 2016, Internet funciona con el protocolo IPv4. Hay sitios que ya tienen una dirección IPv6. Por ejemplo `www.google.com` tiene la dirección 2607:f8b0:4012:805::200e. Sin embargo, la mayoría de los servidores (web en el ejemplo) aún funcionan con IPv4. Así mismo hay que permitir la conectividad entre los usuarios de la red a esos sitios en forma transparente.

El punto de acceso se ilustra en la figura 1.1. En ella se muestra una computadora con una interfaz conectada hacia el mundo exterior en IPv4, la otra interfaz estará en IPv6 y conectada a un punto de acceso inalámbrico. Habrá que añadir alguna forma de comunicación con los protocolos IPv6 de la red local inalámbrica con la red exterior en IPv4. Para la seguridad debe existir un servidor de autenticación, que podría ser externo a la computadora mostrada en la figura 1.1.

También será necesario un cortafuegos para poder autenticar a los usuarios a través de

las direcciones de hardware. Este cortafuegos no está mostrado en la figura 1.1.

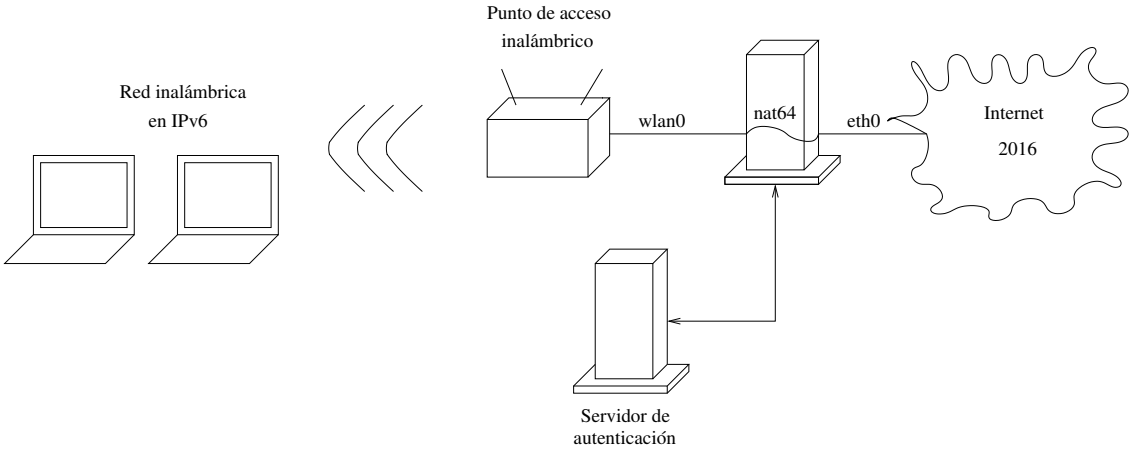

Figura 1.1: Esquema de desarrollo de un punto de acceso seguro para una red interna segura en IPv6 con acceso a la red mundial actual (en IPv4).

El funcionamiento del punto de acceso tendría tres escenarios:

- Usuario no autorizado: Intenta acceder a la red, pero al intentar ingresar el nombre de usuario y contraseña, no se le concede el acceso, porque no posee esos datos válidos para la red.
- Usuario autorizado con dispositivo desconocido: Después de ingresar su usuario y contraseña, se le concede el acceso a la red local, pero el cortafuegos le impide su acceso a la red exterior.
- Usuario autorizado con dispositivo conocido: Después de ingresar su usuario y contraseña, y con un equipo autorizado para usar la red, el usuario de la red inalámbrica tiene acceso a la red externa.

1.1. Planteamiento

Debido a los ataques constantes que se realizan sobre las redes inalámbricas, en donde los intrusos intentan obtener acceso a las redes para tener acceso a Internet y afectar a los usuarios conectados, es necesario ofrecer a los usuarios un servicio de autenticación robusto en su acceso, un mecanismo para evitar el acceso en el caso de compromiso de contraseñas y el uso de IPv6 para mejorar la calidad de conexión de los clientes. El punto de acceso debe contar con algún software que permita la conectividad entre IPv6 e IPv4, y también algún otro software que permita un acceso transparente a los usuarios.

Debido a la evolución de las redes hacia el protocolo IPv6, es necesario desarrollar herramientas que permitan a los usuarios migrar al nuevo protocolo. Así mismo, los usuarios

tienen mayores beneficios en materia de seguridad que los puntos de acceso actuales en IPv4. Se requiere un mecanismo de autenticación robusto multifactor, que no solo proporcione seguridad por medio de una contraseña. Entonces se agregará al punto de acceso el acceso a la red por medio de la dirección de hardware de la tarjeta de red del usuario. De esta forma, aún si un atacante roba una contraseña de algún usuario, no podrá usar la red desde un dispositivo desconocido que no haya sido registrado.

1.2. Objetivos

1.2.1. Objetivo general

Construir un punto de acceso seguro para que los usuarios en una red inalámbrica en IPv6 puedan hacer uso de los recursos actuales de Internet, de modo que la red inalámbrica utilice el protocolo IPv6 y los usuarios de la red tengan la capacidad de acceder a los servidores en IPv4 de la red global.

1.2.2. Objetivos particulares

- Estudiar los protocolos de seguridad existentes en el estándar 802.11i para definir un modelo seguro para la autenticación de los usuarios en la red inalámbrica.
- Estudiar IPv6 para conocer los beneficios de usar este protocolo.
- Examinar los mecanismos de transición de IPv4 a IPv6 para elegir un mecanismo que permita a los usuarios tener acceso a la red global actual (2016).
- Diseñar una puerta que permita a los usuarios acceder a la red en forma segura y usando IPv6.
- Construir la puerta para mostrar su funcionamiento.
- Construir un cortafuegos para redes en IPv6 con netfilter en GNU/Linux.
- Desarrollar las pruebas necesarias del punto de acceso para comprobar su funcionamiento correcto.

1.3. Organización de la tesis

Haciendo una descripción general de la organización de la tesis, empezamos explicando en el capítulo 2 los conceptos básicos de redes IPv6, empezando en la sección 2.1 con el modelado de direcciones y las ventajas de IPv6 sobre su predecesor. En la sección sección 2.2 se describen los mecanismos de transición entre los protocolos IPv4 e IPv6, tales como doble

pila, túnel y traducción de direcciones IPv6 en IPv4, para permitir la navegación entre ambos protocolos incompatibles; también se describe el servicio de nombres de dominio entre esta configuración con ambos protocolos. La sección 2.3 trata los mecanismos de asignación dinámica de nombres, tales como la asignación dinámica de direcciones y la autoconfiguración de ellas.

Después nos profundizamos en los mecanismos de seguridad, empezando con la sección 2.4, que contiene los servicios de seguridad actuales del estándar 802.11i para la autenticación de los usuarios y las extensiones de seguridad 802.1X que hacen uso de un servidor de autenticación para conceder el acceso a los clientes de la red.

El capítulo 3 consiste en el diseño y construcción de la puerta de enlace, en donde se propone el punto de acceso, así como sus componentes de forma más detallada. En la sección 3.2 se muestra la configuración de cada componente esencial para la creación de la puerta de enlace en forma física. En la sección 3.3 se presentan las pruebas de funcionamiento del punto de acceso creado, probando así cada una de las configuraciones hechas en el capítulo anterior. En la sección 3.4 se describe un portal cautivo realizado con el paquete wifidog. Un portal cautivo es otra forma de proporcionar seguridad en la puerta, el cual también hace uso de las reglas del cortafuegos para el filtrado de paquetes.

Finalmente, en el capítulo 4, concluimos con los resultados de esta tesis, así como algunas sugerencias de trabajos a futuro que podrían realizarse para mejorar los resultados obtenidos.

Capítulo 2

Conceptos Básicos

Para construir el modelo del punto de acceso, se hace uso de varios protocolos tanto a nivel superior, en la capa de aplicaciones, así como también de protocolos de nivel inferior. En este capítulo se describirán brevemente los protocolos utilizados en el desarrollo del punto de acceso seguro. Se describirá IPv6 y los mecanismos de traducción de IPv4 a IPv6. También se describirán los mecanismos de seguridad del estándar 802.11i, los cuales se consideran los conceptos primordiales para garantizar la autenticación de los usuarios de la red inalámbrica.

El objetivo de este capítulo no es dar una explicación completa de cómo funcionan cada uno de los protocolos configurados para la puerta de enlace, sino más bien para introducir al lector a su funcionamiento básico.

2.1. IPv6

La versión 6 del protocolo IP está conformada por direcciones con longitud de 128 bits, permitiendo así en teoría la increíble cantidad de 2^{128} direcciones IPv6 [17] (de modo que cada usuario en el planeta podría tener cómodamente algunos millones de anfitriones con direcciones IPv6 diferentes).

A diferencia de IPv4, en IPv6 no se permite la fragmentación de paquetes en nodos intermedios, dejando la responsabilidad de envío de paquetes de dimensiones completas y correctas al nodo fuente, lográndose así que su cabecera sea más simple y eficiente para direccionar (que en IPv4) proporcionando una cabecera de longitud fija de 40 bytes, que permite al software de los enrutadores optimizar el análisis sintáctico de los encabezados de IPv6. Además, se obtiene una reducción en los campos necesarios tal y como se muestra en la figura 2.1.

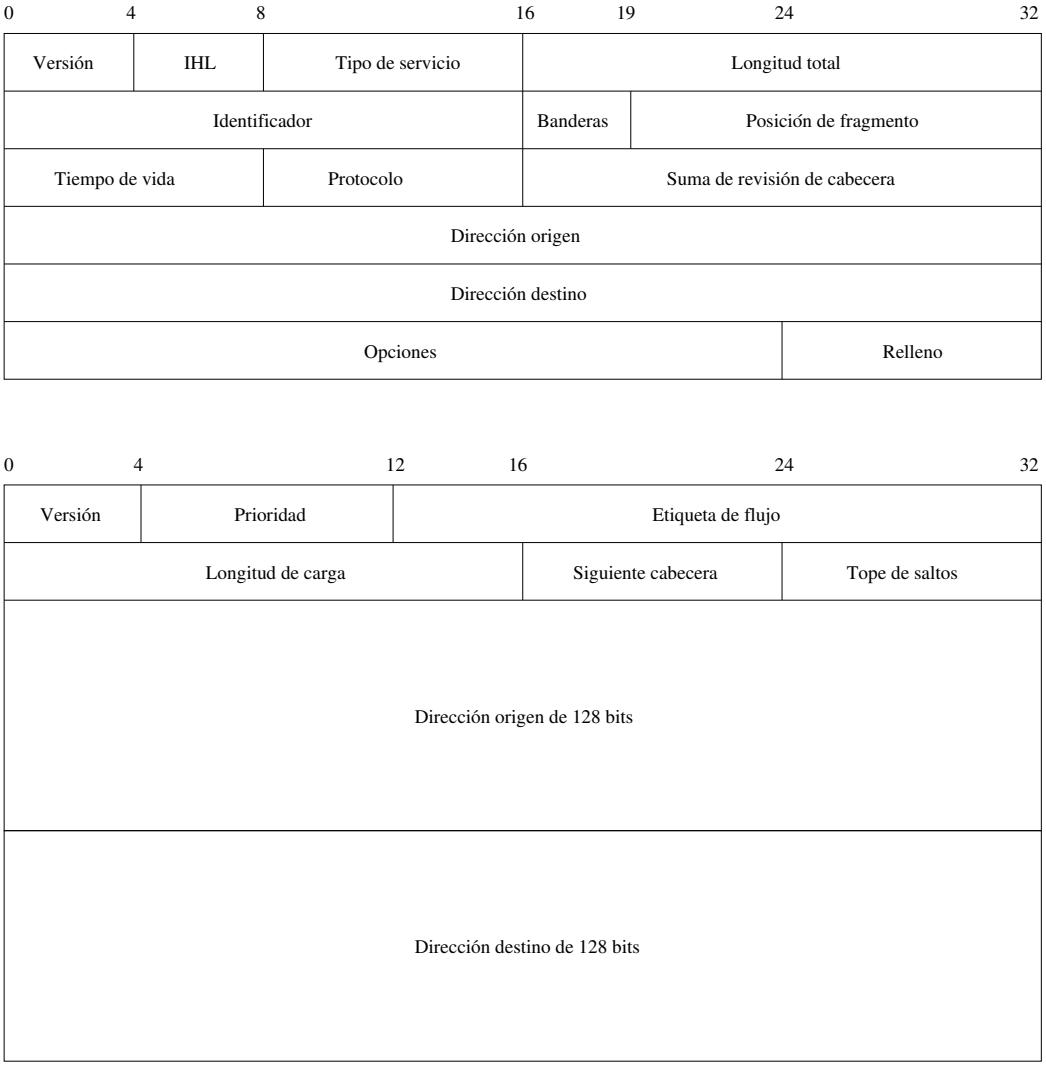

Figura 2.1: Formato de la cabeceras IPv4 (arriba) e IPv6 (abajo). Se puede apreciar la simplificación de campos con respecto a IPv4 [41].

El uso del protocolo IPv6 tiene muchas ventajas sobre su predecesor IPv4, así como nuevas características añadidas, entre las cuales podemos mencionar las siguientes:

2.1.1. IPsec

Para que la transmisión de datagramas de IP sea segura, es necesario proveer autenticación (asegurar la identidad de la fuente y destino), integridad (que los datos sean enviados y recibidos sin alteraciones), confidencialidad (que las partes involucradas sean las únicas con acceso al contenido enviado) y no repudio (seguridad en el destinatario de parte de quien se la envió). Estos objetivos se logran mediante el Protocolo de Internet Seguro (IPsec).

Definidos los aspectos de seguridad en el RFC 1825 [11], IPsec en el RFC 4301 [35] y RFC

4305 [27], se debe establecer una asociación de seguridad para la transmisión de datagramas del protocolo IP. Existen dos modos de operación: modo transporte (en donde los datos transferidos son cifrados y la cabecera IP no se modifica) y modo túnel (en donde tanto los datos como las cabeceras del mensaje son cifrados).

IPsec además proporciona seguridad a nivel paquete, entre los cuales existen los siguientes protocolos:

- Encabezado Autenticado (EA): Almacena información de autenticación para los datagramas, para proveer integridad, autenticación y no repudio. Principalmente se compone de:
  - Un algoritmo de autenticación.
  - La clave usada por el algoritmo de autenticación.
- Encapsulamiento de la Carga (EC): Encapsula un datagrama completo dentro de los protocolos de capa superior (Ejemplos: Protocolo de Control de Transmisión (TCP), Protocolo de Datagramas de Usuario (UDP) y Protocolo de Mensajes de Control de Internet (ICMP)). Al cifrado se añade una cabecera de texto en claro para el nuevo cifrado EC. Básicamente se necesita [29]:
  - Un algoritmo de cifrado.
  - La clave empleada por el algoritmo de cifrado.
  - Sincronización criptográfica (incluyendo su longitud) o campo de vector de inicialización para el algoritmo de cifrado.
  - Un algoritmo de autenticación que incluye la transformación de EC (si está en uso).

Además, se tienen los siguientes requisitos de seguridad para la autenticación de cabecera y EC:

- Tiempo de vida de la clave o tiempo para cambiar la clave.
- Tiempo de vida para la asociación de seguridad.
- Dirección origen de la asociación de seguridad.
- Nivel de sensibilidad para los datos protegidos.

Un último protocolo se define en el *intercambio de claves de internet* que se define en el RFC 2409 [32] y RFC 4109. Proporciona una *asociación de seguridad* al proporcionar una clave compartida para ambas partes con el objetivo de proteger la comunicación mediante un cifrado.

Para obtener esta clave se utiliza el algoritmo de generación de claves de Diffie-Hellman; debido a que para cada par de clientes se tiene la necesidad de crear una clave secreta

conocida únicamente entre ellos. Se sigue el siguiente procedimiento para evitar los escuchas o terceros no deseados:

Llámese Alicia y Beto las partes que quieren convenir en una clave común a través de un canal público (inseguro). Previamente ambos convinieron en un grupo cíclico $C = \langle g \rangle$ sobre el cual generarán las claves. Después ambos eligen un valor aleatorio $x \in \mathbb{Z}^+$ y $y \in \mathbb{Z}^+$ respectivamente, con el cual calculan g^x (Alicia) y g^y (Beto) y se lo envían recíprocamente. Finalmente $k = \left(g^y\right)^x$, es la clave común compartida entre ambos a través del medio inseguro.

La seguridad en éste protocolo radica en que cualquier intruso debe resolver el problema de logaritmo discreto para reconstruir la clave o bien calcular $z = g^{xy}$. Este ataque se puede realizar de forma pasiva (basta con escuchar los mensajes g^x y g^y).

2.1.2. Representación de direcciones en IPv6

Las direcciones IPv6 son identificadores de 128 bits de los cuales existen tres tipos [33]:

- Difusión única: Proporciona un identificador para una sola interfaz. Es decir, un paquete enviado a una dirección de difusión única es enviado a la interfaz identificada por dicha dirección.
- Difusión a cualquiera: Un identificador pertenece a un conjunto de interfaces. Por lo tanto, cuando un paquete es enviado a una dirección con difusión a cualquiera en realidad se envía a alguna de las interfaces identificadas por esa dirección.
- Multidifusión: Un identificador pertenece a un conjunto de interfaces, pero cuando un paquete se envía a una dirección de multidifusión, en realidad se envía a todas las interfaces identificadas por esa dirección. Para IPv6 no existe la difusión, es decir el envío a todas las direcciones dentro disponibles, ya que se sustituye por un enlace de multidifusión. Así también, en teoría, no existen direcciones prohibidas (por ejemplo la terminación en ceros para una red en IPv4) dentro de un campo.

2.1.3. Modelado de direcciones

Las direcciones IPv6 son asignadas *a las interfaces (no a los nodos)* y se componen al menos de una dirección de difusión única con enlace local y puede contener varias direcciones IPv6, ya sean de difusión única, difusión a cualquiera o de multidifusión y de algún alcance anfitrión, enlace local o global.

La representación de las direcciones sigue una de tres variantes:

- Un conjunto hexadecimal de 8 secciones de 16 bits, establecidos entre las amplitudes 0000:0000:0000:0000:0000:0000:0000:0000 y FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF; se pueden quitar los ceros a la izquierda de cada sección, y también simplificar con un solo cero si la sección consta de cuatro ceros. Por ejemplo la dirección 2001:0DF8:0000:0007:0000:0000:0000:0001 se simplifica en 2001:DF8:0:7:0:0:0:1

- Se pueden eliminar ceros con :: (doble dos puntos) si existen secciones consecutivas de ceros, por ejemplo 2001:DF8:0:7:0:0:0:1 en 2001:DF8:0:7::1 (usándose :: una sola vez en cada dirección) [33], [10].
- Cuando se combinan direcciones IPv6 con IPv4, se pueden indicar las primeras seis secciones de la dirección IPv6 y en vez de escribir las últimas dos secciones, se escribe la dirección IPv4 en forma normal, es decir dividida en octetos (8 bits), cada sección de la zona IPv4. Por ejemplo la dirección 64:ff9b::808:808 se puede escribir como 64:ff9b::8.8.8.8.

En cuanto a la representación de las direcciones IPv6 con sus prefijos, se realiza como en IPv4 de la forma: [dirección IPv6]/[longitud del prefijo] en donde la longitud del prefijo está dentro del rango indicado. Por ejemplo: 2001:DF8:0:7::/64 es válido, porque el prefijo es de 64 bits. Sin embargo, 2001:DF8:0:7:54::/64 no es válido, porque la longitud del prefijo excede la cantidad de bits indicados (aquí la longitud del prefijo sería (5 secciones)(16 bits) = 80 bits.)

2.1.4. Identificación de tipos de direcciones IPv6

Los tipos de direcciones son la interpretación de direcciones IPv6 según la composición de bits que estas tienen. Esta se da de la siguiente forma [33]:

- Sin especificar: 128 bits de ceros, es decir, ::/128.
- Retroalimentación: 127 bits de ceros y el último bit 1, es decir, ::1/128.
- Multidifusión: Prefijo: 11111111, por lo tanto en notación IPv6 FF00::/8.
- Enlace local (difusión única): Prefijo: 1111111010000000, por lo tanto en notación IPv6 FE80::/10.
- Global: Cualquier otro prefijo y dirección IPv6.

2.1.5. Otras ventajas de IPv6 sobre IPv4

Se muestran además de la cuestión de seguridad las siguientes ventajas:

- Eliminación de la suma de revisión de cabecera: Reduce el trabajo que tiene que realizar el enrutador para transferir los paquetes, obteniendo un beneficio en rendimiento y costos.
- Eficiencia en el manejo de sitios: Todas los sitios pueden alojar muchas direcciones, las cuales son arrendadas y cuando alguna caduca, ésta ya no es utilizada más, de modo que un sitio puede ser reasignado a direcciones nuevas funcionando de forma equivalente.

- Soporte de conectar y usar: Cuando un cliente se conecta a la red IPv6, la nueva máquina se puede comunicar inmediatamente con otros nodos debido a que se le asigna una dirección local IPv6 de forma automática. La autoconfiguración es manejada en dos formas: El modo *sin estado*, en donde las direcciones son asignadas mediante el prefijo de subred anexando un número de red dentro del rango permitido para formar una dirección IPv6 de 128 bits; el otro modo de configuración es en *estado completo* y aquí los nodos utilizan el mecanismo de Protocolo de Configuración Dinámica del Anfitrión en IPv6 (DHCPv6) para obtener del servidor una dirección y otras configuraciones de red.

2.2. Mecanismos de transición a IPv6

Para hacer la migración al protocolo IPv6 se puede utilizar un mecanismo de transición que nos permita hacer uso del nuevo protocolo sin necesidad de dejar a un lado el protocolo IPv4. Esto nos permite hacer uso del nuevo protocolo manteniendo compatibilidad con las aplicaciones que utiliza el protocolo anterior. Se tiene el caso de los dos protocolos activados en paralelo, que se conoce como *doble pila*, o bien se puede utilizar algún mecanismo de transferencia de paquetes entre ambos protocolos incompatibles: el mecanismo *túnel* o el de *traducción IPv6 a IPv4*.

2.2.1. Doble pila

El mecanismo de transición de doble pila se define en el RFC 4213 [44] y contiene una implementación completa de ambas versiones del protocolo IP, tanto IPv4 como IPv6. Al realizar la implementación en ambos protocolos, cada nodo tiene acceso a ambos protocolos y es capaz de enviar y recibir paquetes tanto de IPv4 como IPv6.

Cada nodo hace uso de ambos protocolos según el tipo de dirección de entrada o salida, como se muestra en la figura 2.2.

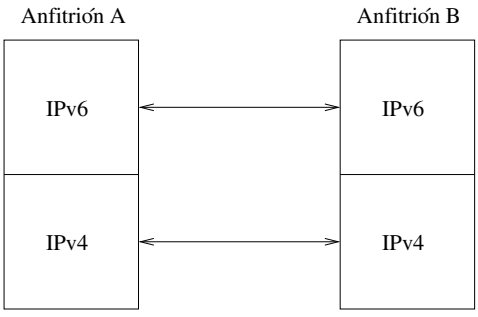

Figura 2.2: Estructura de dos nodos para el mecanismo doble pila. Nótese que existe comunicación directa entre los nodos, ya que poseen acceso a ambos protocolos.

Como cada nodo tiene acceso a los paquetes de ambos protocolos, deben existir mecanismos de asignación para direcciones IPv4 e IPv6, de los cuales podemos mencionar el Protocolo de Configuración Dinámica del Anfitrión (DHCP) para IPv4, DHCPv6 y la Autoconfiguración de Dirección Sin Estado (SLAAC) para IPv6.

Para la resolución de nombres en el modo doble pila se hace uso de los Registros de Dirección (A) para IPv4 y los Registros de Dirección para IPv6 (AAAA) [56], lo cual es posible gracias a la existencia de bibliotecas para interoperar directamente con los nodos que poseen ambos registros. Aún así, existe una preferencia de protocolo para comunicarse de acuerdo a la biblioteca (pudiendo ser IPv6 o IPv4 primero). Así mismo, si la aplicación solicita ambos registros, la biblioteca no realiza ningún filtro y los muestra.

La selección de dirección por defecto en IPv6 se expone en el RFC 3484 [24] y define un algoritmo de selección de dirección origen y otro algoritmo de selección de dirección destino de acuerdo a las siguientes reglas:

- Para selección de dirección origen:
    1. Misma dirección origen.
    2. Dirección de mayor alcance.
    3. Evitar direcciones obsoletas.
    4. Preferencia por direcciones de casa.
    5. Preferencia por dirección de la misma interfaz de salida.
    6. Preferencia por etiquetas que coincidan.
    7. Preferencia por direcciones públicas.
- Para selección de dirección destino:
    1. Evitar direcciones inutilizables.
    2. Dirección de alcance coincidente.
    3. Evitar direcciones obsoletas.
    4. Preferencia por direcciones de casa.
    5. Preferencia por dirección de mayor precedencia.
    6. Preferencia por etiquetas que coincidan.
    7. Preferencia por transporte nativo.
    8. Preferencia por menor alcance.
    9. Uso de mayor prefijo coincidente.
    10. En caso contrario, usar la primera dirección según el orden original.

Aún con la coexistencia de ambos protocolos en los nodos de la red, dos nodos con IPv4 e IPv6, respectivamente, no pueden comunicarse porque son dos protocolos distintos.

2.2.2. Túnel

El encapsulamiento de paquetes a través de un túnel para IPv6 es otro mecanismo de transición entre IPv4 e IPv6. Se define en el RFC 2473 [20] y consiste en una técnica que establece un enlace virtual entre dos nodos en IPv6 con el objetivo de transmitir paquetes. Se puede visualizar como un enlace punto a punto en el cual IPv6 funciona como un protocolo en la capa de enlace.

Su funcionamiento básicamente se describe como sigue: Dados dos nodos A y D, ambos en IPv6, y cuyas entradas al túnel son los nodos virtuales B (para el nodo A) y C (para el nodo B), si el nodo A quiere enviar un paquete al nodo D, A lo envía al nodo virtual B, B lo encapsula y lo envía al nodo C, C recibe el paquete y lo desencapsula para luego enviarlo al nodo destino D. Este proceso de encapsulamiento y desencapsulamiento puede ser en sentido unidireccional o bidireccional, siendo este último el mayormente utilizado. El diagrama bidireccional se muestra en la figura 2.3.

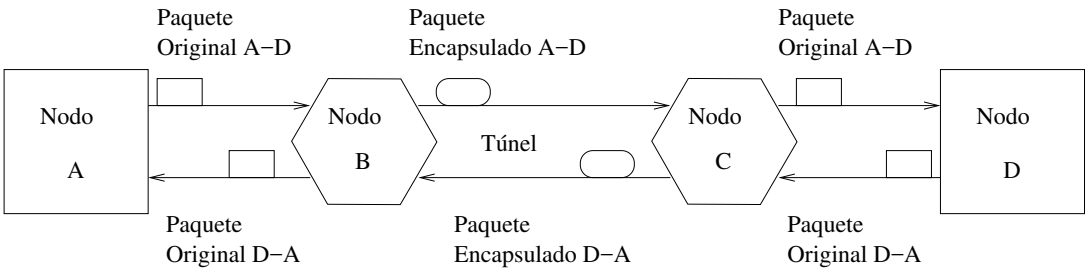

Figura 2.3: Proceso de envío bidireccional a través de un túnel. Los nodos A y D son nodos en IPv6, los nodos B y C encapsulan y desencapsulan respectivamente los paquetes para enviarlos a través del túnel en IPv4 [20].

Tanto para clientes anfitriones como para los enrutadores es posible encapsular los datagramas de IPv6 sobre la infraestructura existente en IPv4, de modo que es posible realizar la conexión entre enrutador-enrutador, enrutador-anfitrión, anfitrión-enrutador y anfitrión-anfitrión.

Debido a que los encabezados de IPv6 e IPv4 tienen diferente longitud, se realiza un encapsulamiento de los datagramas IPv6 a IPv4 como se ve en la figura 2.4.

El encapsulamiento de IPv6 utilizando IPv4 como capa nativa tiene un excedente de 20 bytes de su capacidad de la Unidad de Transmisión Máxima (UTM). Sin embargo, podría existir fragmentación en caso de paquetes de longitud superior, en particular con UTM de 64 kb. No obstante, se pueden implementar otros mecanismos consistentes de UTM estática o dinámica basados en la ruta de IPv4 hacia el punto final del túnel [44].

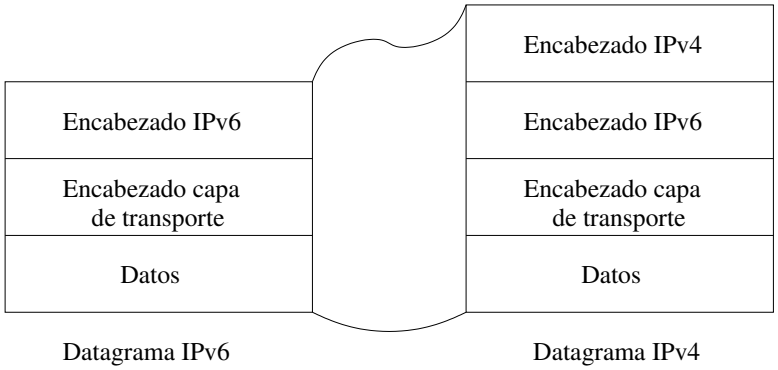

Figura 2.4: Proceso de encapsulado de un datagrama IPv6 en IPv4 [44].

UTM en túnel estático

Cuando se tiene la existencia de un túnel estático, la UTM tiene una interfaz de longitud fija, la cual por defecto se encuentra entre 1280 y 1480 bytes. Cada nodo tiene la capacidad de aceptar un paquete fragmentado cuya longitud después de ser reensamblado es mayor a 1500 octetos, siempre y cuando las UTMs fijas se encuentran en el rango de 1280 a 1480 bytes [44].

Para la selección de un buen túnel se debe obtener bajo las condiciones siguientes:

- Los paquetes sobre el protocolo IPv4 deben tener una ruta inferior de UTM y un valor muy alto que permita la fragmentación en IPv4.
- Un túnel usado para transportar los paquetes de IPv6 contiene un valor demasiado bajo para permitir la fragmentación en IPv6.

UTM en túnel dinámico

La fragmentación dentro del túnel puede reducirse cuando el nodo inicio del túnel puede rastrear el UTM para la ruta en IPv4 a través del túnel. En este caso, la capa IPv6 en este nodo puede verse como un túnel cuya capa de enlace tiene un UTM igual al UTM de la ruta en IPv4 menos el tamaño del encabezado del IPv4 encapsulado [44].

2.2.3. Traducción IPv6 a IPv4

El Traductor de Direcciones de Red 6 a 4 (TDR64), es un mecanismo de traducción de direcciones IPv6 a IPv4 y viceversa, el cual hace uso de información de configuración estática para hacer la traducción por medio de un algoritmo que anexa un prefijo definido para hacer la traducción y permitir la transferencia de paquetes en forma nativa dentro de cada protocolo. Se define en el RFC 6052 [12].

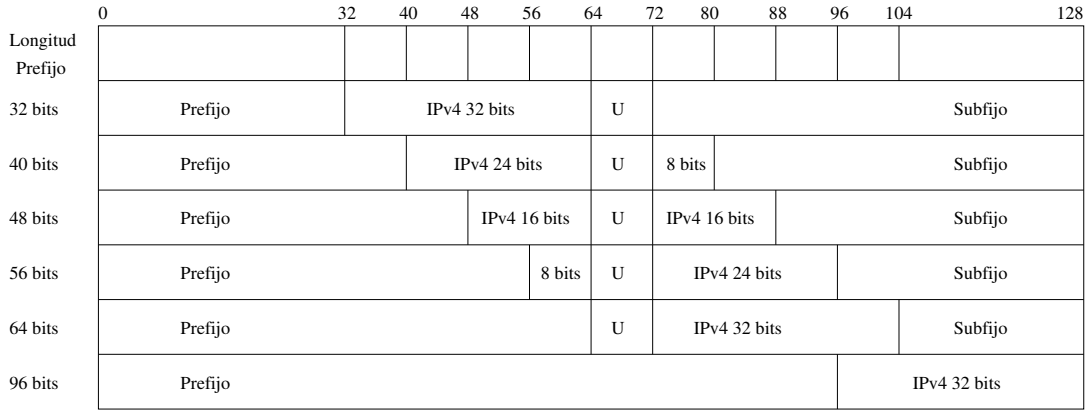

Figura 2.5: Esquema de composición de una dirección IPv6 de TDR64 (de 128 bits). Las direcciones TDR64 se constituyen de un prefijo de longitud variable, la dirección IPv4 por traducir, la cual se encuentra en su forma original (es decir embebida) y un sufijo.

Las direcciones IPv4 son traducidas en direcciones IPv6 utilizando el mismo formato de direcciones IPv6, compuesta por un prefijo de longitud variable, la dirección IPv4 embebida (sin cambios) y un sufijo de longitud variable, en donde algunas formas de estructurar la dirección IPv6 TDR64 se muestran en el esquema anterior.

En cuanto al prefijo, el usado por defecto es 64:ff9b::/96, perteneciente a la longitud de prefijo de 96 bits.

Algoritmos de traducción de direcciones

- Para la traducción de una dirección IPv4 en IPv6 [33]:
  - Concatenamos el prefijo, los 32 bits de la dirección IPv4 y el sufijo (si es necesario) para completar la dirección IPv6 de 128 bits.
  - Si la longitud de prefijo es inferior a 96 bits, se inserta el octeto nulo U en la posición apropiada según el esquema anterior.
- Para la traducción de una dirección IPv6 en IPv4:
  - Si el prefijo es de longitud de 96 bits, se extraen los últimos 32 bits de la dirección IPv6.
  - En el caso de otras longitudes de prefijos, removemos el octeto nulo U para obtener una secuencia de 120 bits, y después extraemos los 32 bits de la dirección IPv4 de acuerdo al esquema anterior.

2.2.4. Sistema de Nombres de Dominio IPv6-IPv4 (DNS64)

Este protocolo se encuentra definido en el RFC 6147 [13]. Es un mecanismo para sintetizar los registros de recursos (RR) $AAAA$ provenientes de registros RR A, se utiliza para entablar

comunicación entre un cliente sólo en IPv6 y un servidor sólo en IPv4. Un Registro de Recurso (RR) *AAAA* sintético consiste en el nombre original del RR *A* (el perteneciente a IPv4) pero conteniendo la dirección IPv4 traducida en IPv6 (generalmente el prefijo TDR64 unido con la dirección IPv4), de modo que la dirección IPv6 obtenida en conjunto de los parámetros de configuración conforman un registro para DNS64.

El contenido de un RR *AAAA* sintético es formado a partir de un RR *A* cuyos campos son los siguientes:

- NAME: Es el campo del nombre, obtenido del RR *A*.
- TYPE: Es el campo del tipo, para *AAAA* es 28.
- CLASS: Es el campo de clase original, es decir 1.
- TTL: Es el tiempo mínimo de vida del RR original *A* y el RR del Inicio de Autoridad (SOA) para el dominio buscado.
- RDLENGTH: Campo inicializado en 16.
- RDATA: Campo seleccionado para la representación IPv6 de la dirección IPv4 obtenida del campo RDATA del RR *A*.

Generación de la representación IPv6 de las direcciones IPv4.

Los siguientes aspectos son indispensables para la generación de las representaciones IPv6, sin importar tanto el algoritmo utilizado:

- El mismo algoritmo utilizado para la generación de las direcciones IPv6 a partir de las direcciones IPv4 debe ser utilizado tanto en DNS64 como en el traductor IPv6-IPv4.
- El algoritmo debe ser reversible, es decir, se debe poner obtener obtener la dirección IPv4 a partir de su representación en IPv6.
- La entrada para el algoritmo se resume en los componentes siguientes: la dirección en IPv4, el prefijo en IPv6 (usualmente 64:ff9b::/96, siguiendo la notación de 64::/n, con $n \leq 96$) usado para las representaciones en IPv6, y opcionalmente, un conjunto de parámetros configurados en DNS64 y TDR64.

DNS64 funciona en dos modos diferentes: autoritativo (en donde se provee el prefijo de TDR64 para que las consultas se efectúen en forma local con un RDATA apropiado) y recursivo (resuelve los nombres utilizando otros servidores DNS externos, para posteriormente hacer la sintetización).

DNS no protege por naturaleza de la falsificación de registros, principalmente debido a ataques de envenenamiento de caché de DNS, los cuales provienen de respuestas de DNS no autoritativos que pueden alterarse en forma local y enviarse a los clientes desprotegidos, provocando la obtención de un recurso no deseado debido al registro alterado; una solución

a este problema es usar el DNS seguro (DNSSEC).

DNSEC, aunque está especificado y resuelve el problema de falsificación de DNSs, no sabemos de algún sitio de red que lo use habitualmente, debido al nuevo problema del manejo seguro de las claves públicas.

DNSSEC

Las Extensiones de Seguridad del Sistema de Nombres de Dominios (DNSSEC) son especificaciones que previenen ataques a DNS a través de añadir autenticación e integridad de datos a DNS. Definido en el RFC 4033 [8], 4034 [9] y 4035 se basa en una cadena de confianza entre los dominios a nivel código de país y sus subdominios. DNSSEC incluye cuatro tipos de nuevos RR:

- Firma de Registro de Recurso (RRSIG): Se encarga de almacenar las firmas digitales asociadas al RR. El formato de su RDATA consiste en los campos: 16 bits para el tipo, 8 bits para el algoritmo, 8 bits para las etiquetas, 32 bits para el TTL original, 32 bits para la caducidad de la firma, 32 bits para el comienzo de la firma, 16 bits para la etiqueta de clave, el nombre del firmante y la firma.
- Clave Pública del Sistema de Nombres de Dominios (DNSKEY): Es un registro que contiene la clave pública, el cual tiene el siguiente formato: 16 bits de banderas, 8 bits del protocolo, 8 bits del algoritmo y la clave pública cuya longitud depende del algoritmo criptográfico de clave pública empleado. El octeto del algoritmo puede contener los siguientes valores:
  - 0: Reservado.
  - 1: RSA/MD5, sin firmado de zona.
  - 2: DH (Diffie-Hellman), sin firmado de zona.
  - 3: DSA/SHA1, con firmado de zona.
  - 4: ECC (Curvas elípticas).
  - 5: RSA/SHA1, con firmado de zona y el recomendado.
  - 252: Indirecto, sin firmado de zona.
  - 253: PRIVATEDNS, reservado para uso privado y nunca será asignado a algún algoritmo específico. El área de clave pública en el RR DNSKEY y el área de firma en el RR RRSIG comienza con un nombre de dominio cifrado y sin comprimir.
  - 254: PRIVATEOID, reservado para uso privado y nunca será asignado a algún algoritmo específico. El área de clave pública en el RR DNSKEY y el área de firma en el RR RRSIG comienza con un byte de longitud *unsigned* seguido de un identificador de objeto cifrado BER (ISO OID), el cual indica el algoritmo privado a utilizar.

Para generar claves de alta eficiencia se requiere una buena fuente generadora de números aleatorios (en hardware para acelerar la produccción de números), en donde se tengan grandes períodos para la generación de la sucesión seudoaleatoria.
Seguimos con la generación de claves con un algoritmo asimétrico y su firmado (por ejemplo RSA para la generación de claves y SHA1 para el digesto que constituye la firma digital).

- Delegación de Firmante (DS): Es insertado en la zona de corte para indicar que la zona delegada es firmada digitalmente y que reconoce la clave como válida para la zona delegada.

- Siguiente Seguro (NSEC): Se utiliza como prueba de autenticación de inexistencia de los nombres y tipos de DNS del propietario, de modo que hace uso del RR NXT para indicar el próximo nombre de dominio.

Y adicionalmente añade dos nuevos bits de cabecera en el mensaje:

- Revisión Deshabilitada (CD).

- Datos de autenticación (AD).

Debido a que los mensajes son más extensos por la adición de RR, DNSSEC requiere soporte EDNS0, así como para el bit de cabecera Extensión de DNS (EDNS) llamado DO (DNSSEC OK) para garantizar seguridad en la resolución de los RR indicados en los mensajes.

Con el objetivo de proveer autenticación, DNSSEC asocia firmas digitales con el conjunto de RR. Dichas firmas son almacenadas en RRSIG, de modo que una clave privada firme los datos de zona para autenticarla.

Las zonas de firmado se utilizan en los Servidores Autoritativos y Recursivos de Nivel Superior (TLD), los cuales son firmados por el Corporativo de Internet para la Asignación de Nombres y Números (ICANN), de modo que los DNS deben subir sus registros DS hacia los dominios padres (en donde DS identifica y autentifica la clave pública contenida en DNSKEY) [50].

Algoritmo DNS64

El sistema de nombres en el servidor DNS64 funciona en forma general del siguiente modo (sin afirmar que es el único modo):

1. DNS64 recibe la búsqueda de un RR de tipo $AAAA$, en donde utiliza la dirección IPv4 para realizar la búsqueda del servidor. Si existe el RR $AAAA$, DNS64 no include el RR sintético (es decir, no anexa el prefijo + dirección IPv4, cuando se encuentra una dirección IPv6 nativa).

2. Si no existe un RR $AAAA$, el servidor de nombres recursivo realiza la búsqueda de un registro A y el servidor de nombres la sintetiza en un registro $AAAA$ anexando el mismo

prefijo de TDR64. Ejemplo: la dirección 192.168.1.1 se sintetiza en 64:ff9b::192.168.1.1, con el mismo nombre de registro, para realizar la búsqueda de forma transparente entre IPv4 e IPv6.

2.3. Protocolos de configuración dinámica del anfitrión

Para asignar las direcciones en los clientes, se pueden hacer uso de los protocolos DHCP (para IPv4) y los protocolos DHCPv6 y SLAAC (para IPv6).

2.3.1. DHCP

Es un protocolo de transferencia de información hacia los anfitriones dentro de una red TCP/IP. Su función es asignar en forma automática las direcciones de red (las cuales son reutilizables) en el protocolo IPv4 y la configuración específica para los clientes participantes de la red por medio del Protocolo *Bootstrap* (BOOTP) siguiendo el modelo cliente-servidor. Se encuentra especificado en el RFC 2131 [25].

Servicios ofrecidos por DHCP

- Proveer almacenamiento persistente de los parámetros de red para los clientes de ella.
- Asignación de direcciones IP temporales y permanentes en la red para los clientes: Aquí el cliente solicita el uso de una dirección por un período de tiempo, en el cual dicha dirección no es reasignada hasta que el cliente la desocupe, notificando al servidor su liberación.

Mensajes DHCP

Para la asignación de direcciones IP, DHCP usa mensajes con el formato BOOTP para enviar desde el cliente la solicitud al servidor y recibir de éste la respuesta deseada. El formato de los mensajes DHCP es el mostrado en la figura 2.6.

Agentes de retransmisión

Sirven para intercambiar mensajes entre servidor y clientes no conectados en el mismo enlace. Existen dos mensajes que se encargan de enviar: retransmisión hacia adelante y retransmisión de respuesta. El formato de cualquiera de estos dos mensajes es el mostrado en la figura 2.7.

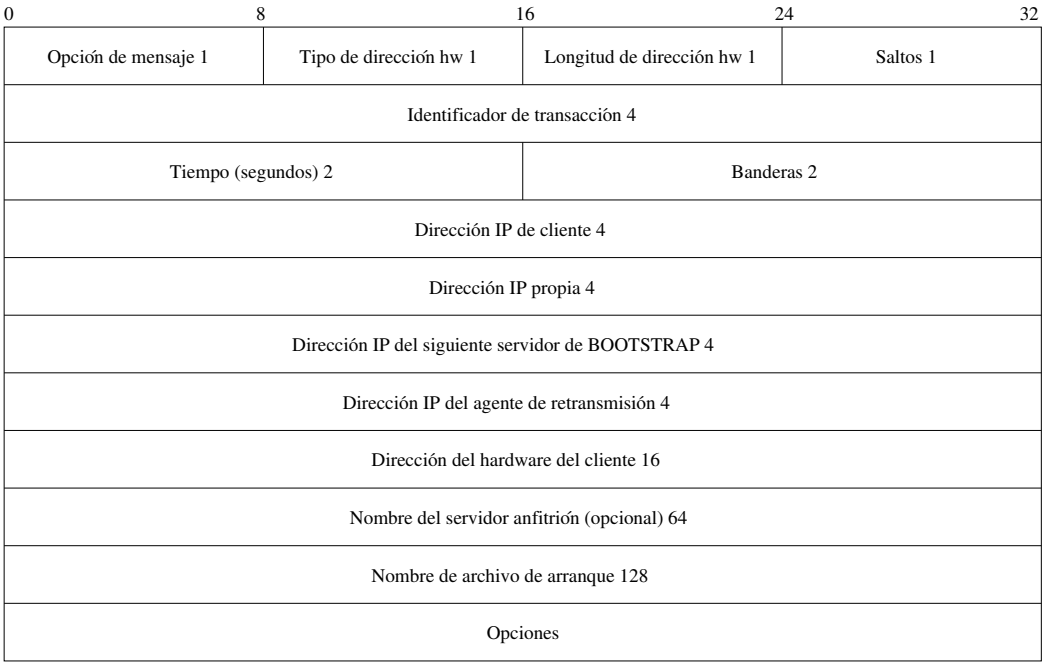

Figura 2.6: Formato de un mensaje DHCP, el número a la derecha se refiere a la cantidad de octetos que utiliza el campo descrito.

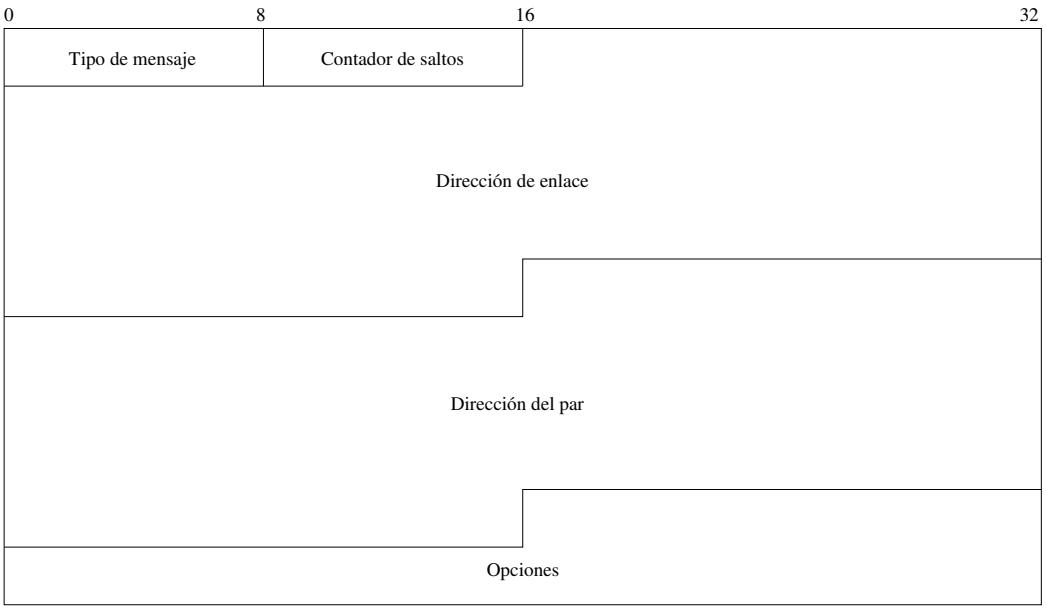

Figura 2.7: Formato de cabecera de un mensaje del agente de retransmisión.

2.3.2. DHCPv6

Este protocolo habilita el paso de parámetros de configuración a los servidores en una red IPv6 y se encuentra definido en el RFC 3315 [26]. DHCPv6 realiza el intercambio de mensajes utilizando el protocolo UDP (puerto 546 para los clientes y 547 para los servidores y agentes de retransmisión). Una vista general del funcionamiento del protocolo DHCPv6 para asignación de direcciones en los clientes se muestra en los siguientes pasos:

1. Se construye una asociación de identidad entre cliente y servidor, de modo que el cliente asocia un Identificador de Asociación de Identidad (IAID) propio a la solicitud de asignación de una dirección IPv6 a un servidor DHCPv6 para su identificación única.

2. El servidor selecciona direcciones para ser asignadas por medio de una asociación de identidad de acuerdo a las políticas de asignación del administrador.

3. El cliente podría solicitar la asignación de direcciones temporales y los servidores asignados a ellos. Dicha opción se llama Asociación de Identidad para Direcciones Temporales (IATA).

4. El cliente envía mensajes DHCPv6 para descubrir todos los servidores disponibles o un servidor individual, utilizando un enlace de multidifusión en la interfaz en donde la información de configuración fue solicitada.

5. El servidor valida las opciones del mensaje recibido del cliente, entre las cuales se encuentran la información y las políticas para responder al cliente. Si el servidor no está autorizado para responder al cliente, se desecha la solicitud. En caso contrario, el cliente cambia el tipo de mensaje como anuncio y copia el contenido de la transacción al mensaje de solicitud, el identificador de cliente y la opción de preferencias, entre otras opciones, las cuales son enviadas al cliente como un anuncio del enrutador, conteniendo las direcciones u otra información de configuración. Se utilizan los tipos de mensajes mencionados en la subsección mensajes DHCPv6.

6. Los clientes son los responsables de enviar mensajes de confianza, así que si un cliente DHCPv6 falla en la recepción de respuesta del servidor, el cliente debe retransmitir el mensaje. Así también los mensajes son descartados cuando las opciones esperadas no aparecen en el mensaje recibido.

7. Al obtener la información de configuración, el cliente obtiene su dirección de difusión única con enlace de alcance global para comunicarse.

Mensajes DHCPv6

Todos los mensajes DHCPv6 enviados entre clientes y servidor constan del mismo formato de cabecera y una sección variable en tamaño de opciones sin la necesidad de tener bits de relleno, como se aprecia en la figura 2.8.

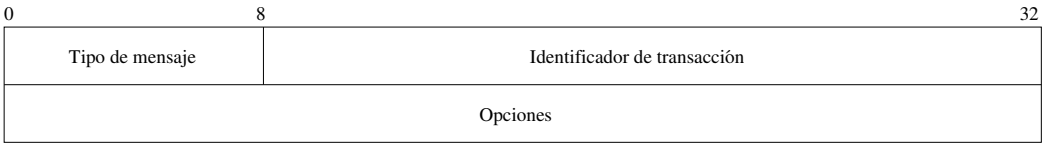

Figura 2.8: Formato de cabecera de un mensaje DHCPv6.

2.3.3. SLAAC

Definido en el RFC 4862 [57], proporciona la autoconfiguración de interfaces cuando una interfaz de multidifusión se encuentra habilitada mediante los siguientes pasos:

1. Cada nodo (tanto anfitriones como enrutadores) genera una dirección de enlace local para su interfaz; dicha dirección se compone al añadir un identificador de la interfaz al prefijo de enlace local FE80::

2. Antes de asignar la dirección al nodo, se verifica por medio de un mensaje *solicitud de vecino* si la dirección tentativa ya está en uso. Si algún nodo la posee, le devuelve un mensaje solicitud de vecino con el objetivo de evitar ambigüedad. Si esto ocurre, no procede la autoconfiguración.

3. En caso de ser única la dirección de enlace local, entonces se asigna dicha dirección a la interfaz, de modo que el nodo ya tiene conectividad vía IP con sus vecinos.

4. El nodo espera un mensaje anuncio del enrutador para determinar el tipo de autoconfiguración que el nodo puede tener, o bien indicar que no hay enrutadores presentes.

5. Al recibir el mensaje anuncio del enrutador, puede existir la información del prefijo para generar una o más direcciones globales. En este caso se podría usar tanto SLAAC o bien DHCPv6 para la obtención del prefijo global, y por lo tanto, la asignación de una dirección global única. En el caso de SLAAC se procede solo si la bandera configuración autónoma de la dirección está activada.

Antes de asignar cualquier dirección pasa por una prueba de unicidad para prevenir duplicidad de direcciones; además para acelerar el proceso de autoconfiguración, el nodo envía una solicitud a todos los enrutadores del grupo de multidifusión y también se puede reducir el tiempo de respuesta del enrutador trabajando en paralelo tanto en la generación de la dirección de enlace local como en la verificación de su unicidad [42].

2.4. Servicios de seguridad del estándar 802.11i

Una primera instancia en materia de seguridad para las redes inalámbricas se definió por el grupo de trabajo 802.11. Las especificaciones originales para las redes inalámbricas 802.11 consistieron de un Identificador de Conjunto de Servicios (SSID), filtrado de direcciones MAC

y un cifrado, de modo que en los primeros años de su aparición la Privacidad Equivalente a Cableado (WEP) fue ampliamente utilizada para protección de la contraseña de acceso a las redes inalámbricas.

Sin embargo, WEP ha presentado numerosos fallos de seguridad, debido a que existen solamente 2^{24} posibilidades de envío de tramas y, por consecuencia, los vectores de inicialización se agotan rápidamente y ocasiona que la detección de la clave de red sea fácil. Además de que no existen claves para los paquetes. De este modo se crean las siguientes debilidades en WEP [60]:

- No hay protección contra ataques de reconexión.
- Posibilidad de modificación y falsificación de paquetes.
- Cifrado RC4 débil.

Aprovechando estas debilidades, los *hackers* han desarrollado varios ataques para romper el sistema. Entre los más comunes podemos mencionar *Flurer, Mantin and Shamir* (FMS) (2001), Korek (2004), *Pyshkin, Tews, Weinmann* (PTW) (2007) y ChopChop (2008) de modo que las redes WLAN permanecieron vulnerables durante algunos años; e incluso la suite de Aircrack-ng proporciona herramientas para descifrar las claves (en particular para las redes WEP) utilizando estos ataques [7].

2.4.1. WPA

El Acceso Protegido WiFi (WPA) es una solución a los problemas de seguridad enfrentados por WEP. Introducido en 2002, es un sucesor del protocolo WEP y se creyó seguro hasta 2008, cuando se publicó un ataque recuperado por Michael [14]. Definido por el grupo de trabajo 802.11i, es un estándar de seguridad que proporciona las siguientes ventajas [29]:

- Autenticación: La proporciona un servidor de autenticación que incluye un protocolo robusto de autenticación. En este caso, una estación (cliente) solicita su autorización de acceso al servidor de autenticación, lo cual incluye su verificación de identidad.
- Gestión de claves: El servidor de autenticación proporciona diferentes esquemas de cifrado, para crear y distribuir las claves entre los clientes. Utiliza el algoritmo RC4 para cifrado, tal como su predecesor WEP. Sin embargo, la longitud de claves es de 128 bits con longitud de vector de inicialización (VI) de 48 bits.
- Privacidad en la transferencia de datos.

Con el objetivo de mejorar la seguridad en WPA, se proporciona un Código de Integridad de Mensaje (CIM) para proteger la integridad de los paquetes. El CIM consiste en un valor de 8 bytes calculado antes de cifrar y transmitir un mensaje, para detectar cualquier anomalía o modificación del paquete en el camino. En caso de presentarse, el enlace de los dispositivos comprometidos se deshabilita por una cantidad de tiempo (por lo general 60 segundos) y se

generan nuevas claves de sesión.

Existen algunos mecanismos de cifrado para WPA, como por ejemplo WPA-TKIP y WPA-PSK.

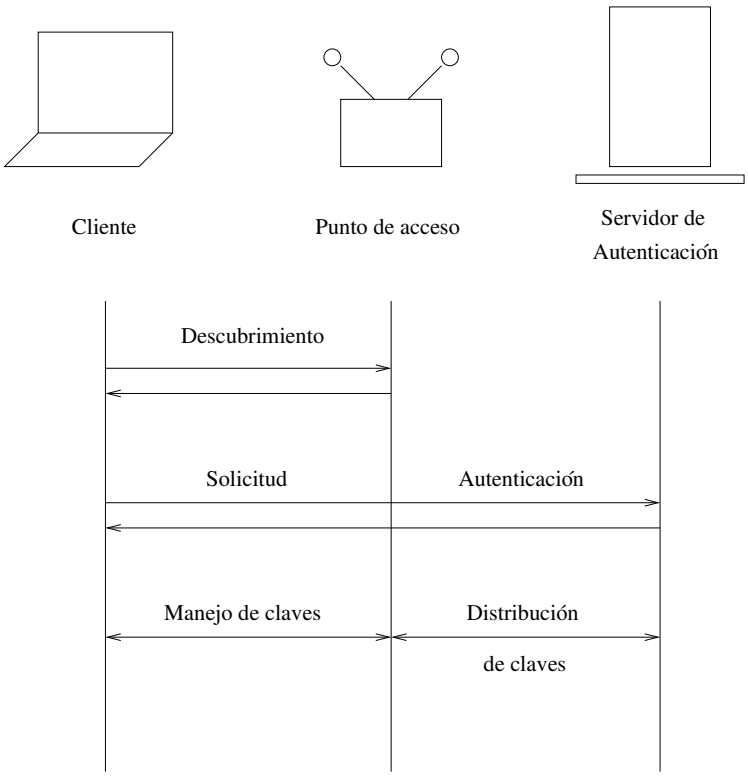

Figura 2.9: Estándar 802.11i, esquema de funcionamiento WPA [29].

WPA-TKIP

WPA-TKIP hace uso del Protocolo de Integridad de Claves Temporales (TKIP) para transmitir los datos, el cual hace cambios en los dispositivos configurados con antiguos sistemas de seguridad, por ejemplo WEP, para resolver el problema de reutilización de VI. Sin embargo, nunca fue una solución permanente y fue dado de baja en 2009.

WPA TKIP genera paquetes de 128 bits para ser transmitidos y utiliza un mecanismo para refrescar el cifrado y la clave de integridad. Las claves temporales cambian cada 10000 paquetes. El mecanismo de TKIP es el siguiente [60]:

- Se calcula el valor de una clave temporal de sesión y después se obtiene el digesto de la MAC del emisor, la clave temporal de sesión y 32 bits para el VI.
- Por cada paquete recibido, se obtienen los 16 bits menos significativos y con el resultado del paso anterior, obtenemos otro digesto, el cual tiene una longitud de 104 bits.

El cifrado se realiza en forma similar que WEP, con la diferencia que el campo de 24 bits de VI es reemplazado por los 16 bits de WPA menos significativos. Así, se obtiene un cifrado y descifrado con claves dinámicas de 128 bits en contraste con los 24 bits dinámicos concatenados con 40 o 104 bits. Evidentemente, esto incrementa la seguridad del sistema.

WPA-PSK

WPA Clave Pre-Compartida (PSK) es un modo especial que no necesita una infraestructura 802.1X. Este consiste en estaciones cuyas claves son precompartidas y es sencillo de configurar, de tal modo el usuario solo necesita introducir su contraseña para acceder. Sin embargo, el nivel de seguridad en WPA-PSK radica en la creación de claves robustas, pues en caso de no serlo, un atacante podría capturar los mensajes de autenticación y recuperar la contraseña una vez fuera de línea [60].

A pesar de las mejoras en seguridad al utilizar un servidor de autenticación, 802.11i es vulnerable a los atacantes, en particular si la red falla en algún punto (por ejemplo en la negociación de claves). La estación legítima se ve en la necesidad de enviar mensajes adicionales para retroceder. Sin embargo, si el tiempo de recuperación se vuelve excesivamente largo, significa que el atacante puede repetir el ataque una y otra vez obteniendo un ataque de Denegación de Servicio (DoS) satisfactorio [16].

2.4.2. WPA-2

Este protocolo hace uso del algoritmo Estándar de Cifrado Avanzado (AES) para cifrado y para la autenticación de mensajes con el objetivo de proporcionar un algoritmo de cifrado más robusto que el RC4 y que TKIP y mejora la seguridad.

Utiliza el método del Protocolo MAC en Modo Contador-CBC (CCMP), el cual hace uso de claves con longitud de 128 bits, el algoritmo AES en el modo Encadenado de Bloques de Cifrado (CBC) y MAC para calcular el CIM y VIs de longitud de 48 bits. A diferencia de RC4, utilizado en WEP y WPA, AES reduce los riesgos existentes al utilizar funciones digesto con WPA-TKIP. Además, WPA2 incluye un número de paquete y un *token* para evitar ataques de reconexión [7].

La fase de autenticación en WPA2 se maneja por medio de un Par de Claves Maestras (PMK) y a través del estándar de autenticación IEEE 802.1X. Así el proceso de cifrado consiste en dos etapas: el cifrado de los datos y la verificación de su integridad. A continuación se muestran cómo se realizan estos pasos [49]:

Proceso de cifrado AES

- El contador inicial es la entrada para AES, el cual cifra los datos produciendo un bloque de 128 bits.
- Se aplica una O exclusiva (XOR) a los primeros 128 bits de texto en claro, produciendo así el primer trama.

- Se repiten los pasos anteriores hasta finalizar el cifrado de los tramas.

Verificación de integridad del mensaje CIM

- Se introducen los 128 bits del block inicial junto con la clave de integridad de datos, en AES-CCMP para producir un bloque de 128 bits de longitud.
- Se aplica un XOR a los 128 bits obtenidos con los 128 bits del block de *payload* de IEEE 802.11.
- Se repiten los pasos anteriores, siempre con los remanentes 128 bits del block de *payload*. Dentro de los últimos 128 bits restantes de AES-CCMP, los 64 bits más significativos son el CIM descifrado.

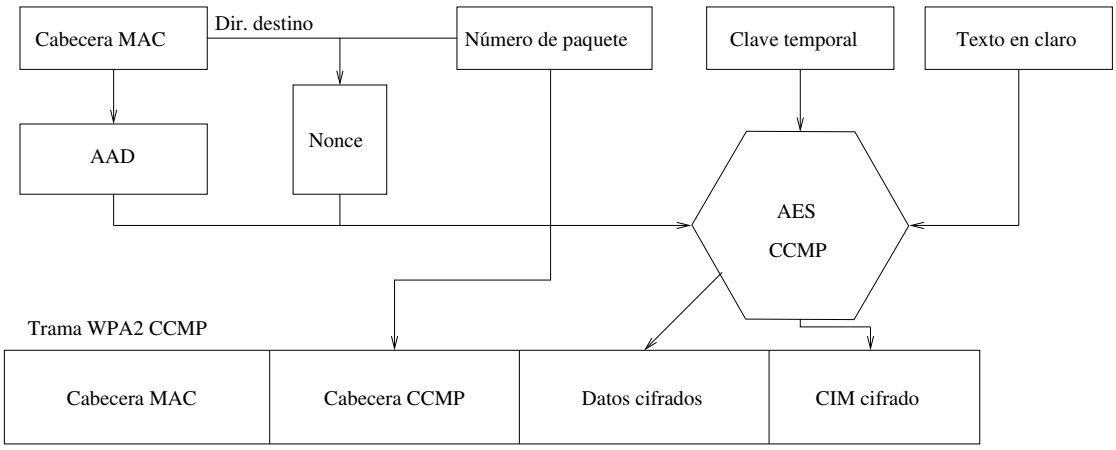

Figura 2.10: Funcionamiento de WPA2, creación de un trama por WPA2-CCMP [7].

En cuanto a la autenticación de mensajes, WPA2 soporta dos modos: *personal* (también conocida como WPA2-PSK, cuyo funcionamiento es similar a WPA-PSK) y *empresa*. En donde este último hace uso del estándar 802.X1 para la autenticación de usuarios.

Debido a la verificación de paquetes y a la autenticación de mensajes, WP2 asegura la integridad de paquetes. Así mismo, AES-CCMP proporciona mucha robustez en el cifrado de mensajes. Sin embargo, el costo de implementación se incrementa considerablemente debido a los recursos ocupados en el cifrado AES.

2.5. Extensiones de seguridad 802.1X

El grupo de trabajo IEEE 802.1X propuso en septiembre de 2003 una mejora a las autenticaciones locales. *Remote Authentication Dial-In User Service* (RADIUS), definido en el RFC 3580 [19], consiste en un servidor remoto que tiene la capacidad de hacer las funciones de AAA. Los parámetros principales que puede tener un servidor RADIUS son los siguientes:

- Nombre de usuario: El solicitante proporciona su identidad por medio de un mensaje de identidad, incluido en el atributo nombre de usuario, que el servidor de autenticación recibe como un mensaje RADIUS de solicitud de acceso.
- Contraseña de usuario: Debido a que IEEE 802.1X no soporta el Protocolo de Autenticación de Contraseñas (PAP) o el Protocolo de Autenticación de Desafío con Apretón de Manos (CHAP), no es utilizado.
- Dirección de Servidor de Acceso a la Red (NAS) / Dirección IPv6 de NAS: Es la dirección del puente o punto de acceso que hace la labor de servidor de autenticación.
- Puerto NAS: Número de puerto del puente, al cual, en caso de no existir, se le asigna un identificador de asociación cuya longitud es de 16 bits sin signo.
- Tipo de servicio: Tramado, solo autenticación o revisión de llamada.
- Otras opciones: Identificador de NAS, tipo de puerto NAS (Ethernet, Wireless, Anillo de Tokens o FDDI), mensaje EAP, identificador de puerto NAS, etc.

2.5.1. EAP

Protocolo de Extensiones de Autenticación (EAP) es un marco de trabajo para varios métodos de autenticación. Su funcionamiento permite trabajar con Protocolo Punto a Punto (PPP) y 802 sin necesitar una ip. Así mismo, tiene soporte a métodos individuales, sin permitir la fragmentación. Se define en el RFC 3748 [1].

El funcionamiento de la autenticación EAP es el siguiente:

1. El servidor de autenticación envía una solicitud para autenticar al cliente. Esta solicitud contiene un campo de tipo, en el cual se puede incluir la identidad, un desafío MD5, etc.
2. El cliente envía un paquete de respuesta, conteniendo el campo de tipo correspondiente a la solicitud.
3. Se repite el procedimiento cuantas veces sea necesario, ya que EAP no envía un nuevo paquete sin haber recibido el paquete anterior válido.
4. Mientras el cliente no sea autenticado, continúa la conversación, hasta que el servidor de autenticación determine que la autenticación es satisfactoria. En caso contrario, el servidor envía un fallo de EAP.

Debido a que EAP es un protocolo punto a punto, se lleva a cabo una autenticación en forma inversa, es decir que ambas partes del enlace se comportan como servidores de autenticación y clientes al mismo tiempo. Algunas implementaciones, métodos, protocolos de AAA y capas de enlace de EAP no funcionan y otros tienen un soporte diferente. Por ejemplo, algunos

métodos EAP pueden funcionar con autenticación asimétrica.

Por tal motivo, la falta de homogeneidad hace necesario explicar los tipos más comunes de protocolos de autenticación para EAP.

2.5.2. EAP-LEAP

Una primera solución al problema de autenticación es el Protocolo de Extensiones de Autenticación de Bajo Peso (LEAP), el cual es un marco de trabajo de autenticación, de fácil negociación para conexiones punto a punto, siendo el primero en su tipo en utilizar un método de autenticación con una clave precompartida estática.

Su funcionamiento se maneja por medio de transferencia de mensajes de autenticación entre cliente, el punto de acceso inalámbrico (*autenticador*) y el servidor de autenticación, los cuales siguen el flujo de la figura 2.11.

El mensaje de respuesta EAP-LEAP incluye la clave de sesión de cifrado, la cual es generada por parte del servidor de autenticación, realizando el digesto de toda la información intercambiada entre cliente y servidor, la cual consta del desafío del servidor, el desafío del cliente y el identificador del cliente. Esta clave es generada en ambas partes y es utilizada para el cifrado y descifrado de los datos transferidos entre cliente-servidor y viceversa.

La privacidad de datos al utilizar EAP-LEAP radica en el uso de claves de sesión con cifrado dinámico, generados por desafíos de parte tanto del cliente como el servidor. De este modo, se evitan ataques de *reconexión* a un bajo costo computacional. Sin embargo, existe la vulnerabilidad de que si se logra que los dos desafíos sean escuchados, un atacante podría utilizar un ataque de diccionario, de modo que podría utilizar un ataque estilo hombre de enmedio (HE) para controlar el canal, perdiendo así la conversación la integridad de datos y su privacidad.

Ante lo anterior, EAP-LEAP no tiene protección de la identidad del cliente, debido a que el identificador del cliente se envía como texto en claro [28].

2.5.3. EAP-MD5

El método EAP-MD5 funciona de forma análoga a CHAP y hace uso de la función MD5 como digesto para proporcionar autenticación para el cliente. Desafortunadamente, la autenticación proporcionada no es mutua (debido a que no proporciona autenticación del lado del servidor) [39].

La autenticación EAP-MD5 la exponen los mismos autores y funciona de la siguiente forma:

- El cliente envía un EAPoL-start al punto de acceso.
- El punto de acceso pregunta por el identificador del cliente.
- El cliente envía al punto de acceso su identificador, el cual es enviado al servidor RADIUS.

- El servidor envía un desafío al cliente a través del punto de acceso.
- El cliente responde el desafío con EAP-MD5 al servidor.
- Si la respuesta es válida, el servidor concede el acceso.

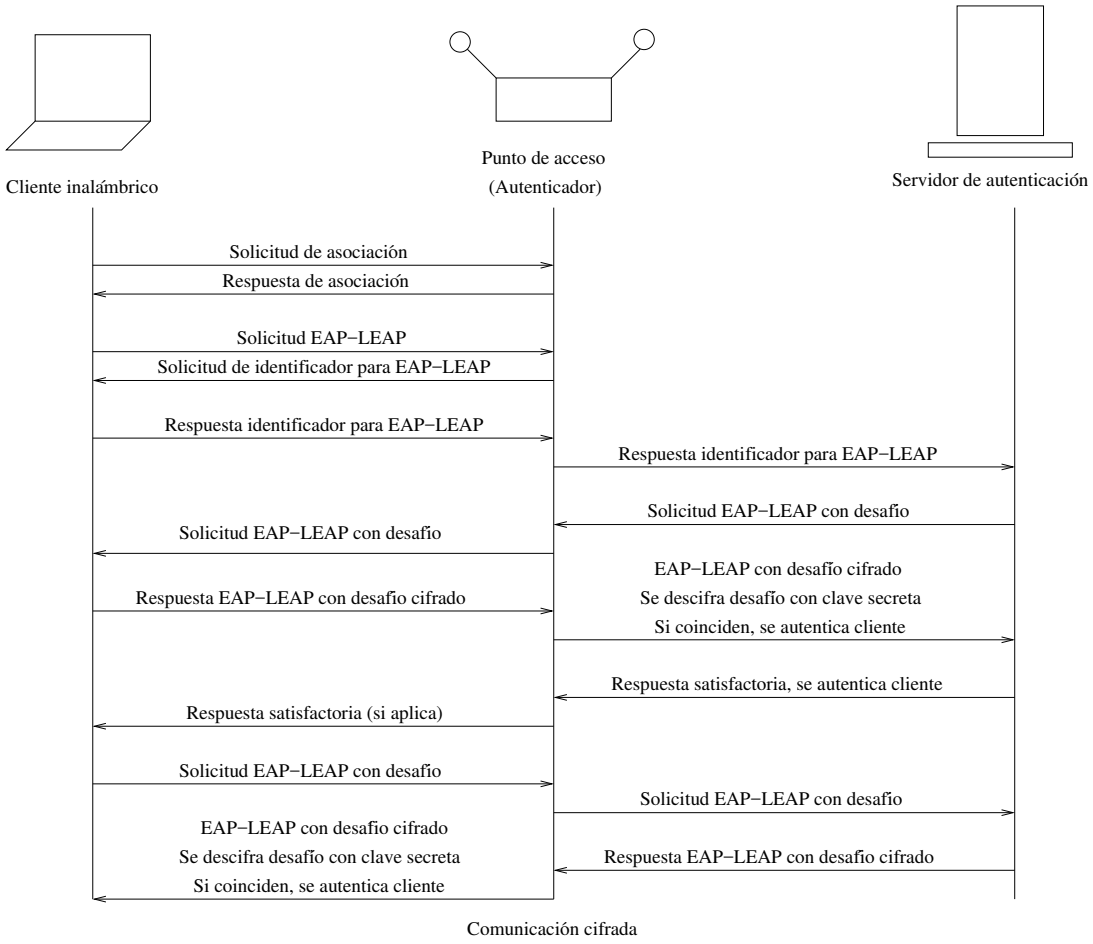

Figura 2.11: Flujo de mensajes para la autenticación EAP-LEAP entre cliente y servidor [28].

Debido a que el método no protege de escuchar por parte de un atacante, existen varios ataques que se pueden hacer abusando de la vulnerabilidad, tal y como lo indican Liu y Xie, así como un análisis de vulnerabilidad en EAP-MD5, como por ejemplo:

- Ataques HE.
- Ataques de diccionario (cuando las contraseñas son débiles) con *tablas arcoiris*.
- Inyección de paquetes utilizando la información del desafío.

El uso de tablas arcoiris, para precalcular todos los digestos de las posibles contraseñas de determinadas longitudes es una de las causas por las que el método se ha descontinuado en la práctica. Esto se debe a que en una PC ordinaria con una CPU de 2 GHz se pueden encontrar contraseñas menores a 5 caracteres. Aún si podemos involucrar paralelismo con mil procesadores, y tomando en cuenta un alfabeto de 95 caracteres, el ataque se concluiría en 18 segundos para contraseñas con longitud de 6 caracteres, 829 segundos para 7 caracteres y 1108 segundos para 8 caracteres [39]. Estos tiempos son muy razonables en la práctica.

Debido a que no se pueden generar claves dinámicas, el método se vuelve inseguro y por lo tanto descontinuado en la práctica. Además, se sabe que las funciones MD5 presentan colisiones para ciertas cadenas [52].

2.5.4. EAP-TLS

Definido en el RFC 2716 [2] y 5216 [53], EAP-TLS es un protocolo de autenticación mutua para dos clientes cuyo enlace es punto a punto (utilizando el protocolo PPP). El modelo que utiliza es cliente-servidor de acuerdo a los siguientes pasos:

- El servidor EAP recibe la identidad del cliente y responde con un paquete de inicio EAP-TLS.
- El cliente envía un paquete respuesta con el tipo EAP-TLS. En tal paquete se encapsulan los registros TLS y contiene un mensaje (hola) para iniciar un apretón de manos dentro del campo de datos.
- El servidor EAP responde con un paquete de solicitud EAP con tipo EAP-TLS. Este paquete contiene la encapsulación del mensaje hola del servidor, así como el certificado TLS, la clave de intercambio del servidor y la solicitud del certificado.
- Si el identificador del cliente es desconocido para el servidor o bien es nulo, el servidor establece una nueva sesión. De lo contrario, deberá coincidir con el del cliente.
- El cliente continúa la sesión (en caso de haberla iniciado antes) o bien el servidor EAP decidirá qué hacer.

De modo que la autenticación entre cliente y servidor se puede resumir bajo el procedimiento del cuadro 2.1.

Como un breve análisis de seguridad, al utilizar certificados digitales se verifica la autenticidad de los clientes en forma segura, así que es un buen método para autenticar. Sin embargo, el costo de implementación todavía es alto, aunque no tanto como con el uso de AES. Aún cuando la protección contra ataques activos del tipo HE sea otorgada, es vulnerable a los ataques de DoS. Además requiere un servidor RADIUS para la asociación mutua.

Autenticación EAP-TLS

Cliente	Servidor de autenticación (*authenticator*)
	< −Solicitud de identidad (EAP-Request)
Respuesta de identidad (EAP-Response)− >	
	< − Inicio de TLS, EAP-Request, EAP-Type=EAP-TLS
Mensaje del cliente (hola) EAP-TLS− >	
	< −Solicitud EAP-TLS, hola de servidor, certificado TLS, intercambio de clave de servidor, solicitud de certificado de cliente, hola de servidor hecho.
Mensaje EAP-TLS con: certificado TLS, intercambio de clave de cliente, verificación de certificado, especificación de cambio de cifra y finalización TLS− >	
	< − Solicitud EAP-TLS, cambio de especificación de la cifra, finalización de TLS
Respuesta EAP-TLS− >	
	< − Éxito, EAP-Success

Cuadro 2.1: Autenticación e intercambio de claves en EAP-TLS.

2.5.5. EAP-GTC

El método de EAP Tarjeta de *Tokens* Genéricos (GTC) es una implementación que hace uso de un *token* con la información de usuario para su autenticación, la cual puede ser leída por un usuario proveniente de un dispositivo en la cartera de *tokens*. La cartera de *tokens* puede hacer uso de una autenticación que incluye la resolución de un desafío como respuesta, de modo que es necesario que las contraseñas se encuentren cifradas en el servidor RADIUS a menos que se configure un túnel protegido con el servidor (para tener soporte a las contraseñas en claro dentro del servidor de autenticación) [1].

Mencionar el mecanismo en el RFC 3748, es una mejora al Protocolo de Extensiones Protegidas de Autenticación (PEAP) y CHAP al utilizar una respuesta que incluye un desafío del servidor de autenticación y un *token* de seguridad, aunque no se especifica un método para negociación [18].

La ventaja en utilizar EAP-GTC, en comparación con otros métodos de EAP es el al-

macenar contraseñas cifradas en el servidor de autenticación, protegiendo el servidor de autenticación mismo de ataques provenientes de usuarios internos de la red e incluso malos administradores. Este método se usó en el desarrollo del punto de acceso de esta tesis.

Capítulo 3

Diseño y construcción del punto de acceso

El objetivo de esta sección es especificar el modelo del punto de acceso (PA) para la red interna, que dará el servicio de internet a los clientes que usen esa red interna. El PA debe proporcionar todos los servicios necesarios para permitir el acceso a internet al cliente autenticado y la gestión de los mismos por parte del administrador.

Para hacer uso de la red actual en IPv4, los clientes en la red interna con IPv6 necesitan un mecanismo de transición. Como el mecanismo doble pila no permite la comunicación entre ambos protocolos y el túnel necesita un nodo para encapsular y desencapsular los paquetes, se ha elegido TDR64 para permitir la traducción de paquetes entre los dos protocolos, que además de permitir la transferencia de paquetes, permite una mejora en la calidad de servicio del PA.

El PA debe cumplir con las siguientes actividades:

- Asignar a los clientes autorizados una dirección IPv6 dentro de la red local, así como establecer una ruta hacia la puerta de enlace (en IPv6).
- Filtrar los paquetes recibidos de los clientes permitiendo solamente el tráfico autenticado por la puerta de enlace. De este modo, solamente los clientes autenticados con los dispositivos autorizados pueden hacer uso de la red.
- Realizar la traducción de paquetes IPv4 en IPv6, ya que los clientes solo tienen acceso a IPv6 y la red exterior se encuentra en IPv4.
- Levantar un servicio de dominio de nombres utilizando las direcciones reales exteriores IPv4 y convirtiéndolas en IPv6 para que los clientes puedan realizar consultas a éstas de forma nativa.
- Contar con un servidor web que muestre al usuario una página con las instrucciones para poder hacer uso de la red cuando el dispositivo no esté autorizado para usarla.

El PA se realizará en una computadora que debe contar con dos interfaces de red físicas: una alámbrica y otra cableada. Además, al usar el esquema TDR64 se tiene una interfaz virtual que se llama `nat64`. La interfaz cableada se presenta como puerta de enlace para IPv4 (la cual conecta hacia el exterior, es decir hacia internet) y la inalámbrica es la puerta de enlace para la red local IPv6. La interfaz virtual se encarga de la conexión entre interfaces por medio de una "caja negra" que realiza la traducción de direcciones de IPv4 a IPv6 por medio de TDR64 permitiendo así la comunicación nativa entre ambas interfaces físicas.

La figura 3.1 muestra el diseño construido del punto de acceso: todos los paquetes involucrados con la red exterior en IPv4, ya sean de entrada o de salida, son transferidos por medio de la interfaz alámbrica eth0. Los paquetes son traducidos por la interfaz `nat64` para que su transferencia por la interfaz `wlan0` sea en IPv6. El servicio DNS se encarga de las consultas en IPv4 mientras que la traducción de registros A en AAAA se realiza en la interfaz local del servidor (::1). La interfaz `wlan0` se encarga de transferir los paquetes traducidos en IPv6 hacia los clientes y la asignación de direcciones IPv6 a los clientes por medio de DHCPv6. La autenticación del usuario se realiza por medio del servidor RADIUS bajo el mecanismo EAP-GTC, el cual compara el digesto de la contraseña introducida por el usuario con la correspondiente en su registro, que en caso de coincidir, concede el acceso a la red local. El cortafuegos realiza el marcado de paquetes para los dispositivos autenticados con 2 para tener acceso a la red exterior y los desconocidos con 1, los cuales son redirigidos al servidor web con las instrucciones de registro.

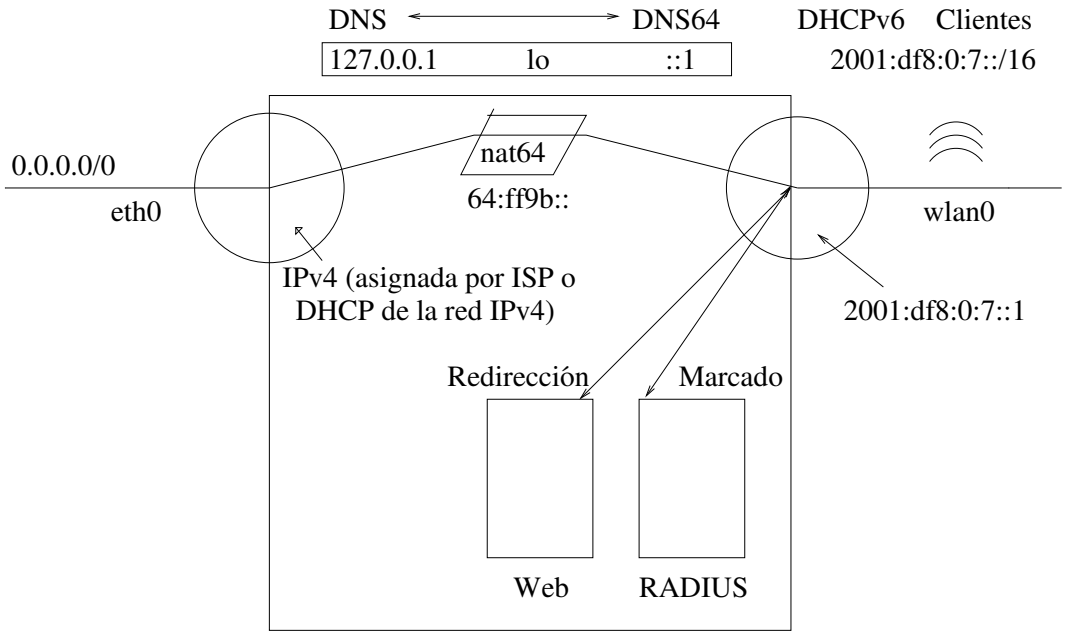

Figura 3.1: Punto de acceso seguro; contiene las interfaces físicas (`eth0` y `wlan0`), la interfaz virtual (`nat64`), la interfaz local (`lo`), las reglas de marcado y filtrado, así como los protocolos involucrados para la asignación de direcciones DHCPv6 y la resolución de nombres DNS64, así como el servidor de autenticación.

La interfaz en IPv4, llamada `eth0` en la figura 3.1, se conecta a la red exterior y contiene la IP pública asignada por el Proveedor de Servicio de Internet (PSI). Para esta interfaz se configura el servidor DNS, para hacer las búsquedas de los registros A de los nombres de dominio, previo a su conversión a registros AAAA, y éste es un servidor recursivo. Además, esta interfaz sirve como conexión de salida para la interfaz virtual TDR64 hacia el lado de IPv4.

Dentro de la interfaz en IPv6 se requiere un servicio de DHCPv6, encargado de la asignación dinámica de direcciones con enlace de difusión, las cuales permiten a los clientes locales la conectividad a la red IPv6. Aquí también tenemos un servidor de nombres DNS64 para convertir los registros de búsqueda A en AAAA, con el objetivo de conservar el mismo nombre de la dirección original en IPv4 a la dirección traducida en IPv6. También necesitamos un cortafuegos para el marcado y filtrado de paquetes IPv6 para redirigir al servidor web (dispositivos no autenticados) o a internet (clientes autenticados).

La interfaz virtual hace la función de TDR64 como mecanismo de traducción de direcciones IPv4 en IPv6, para que la comunicación entre interfaces sea de forma transparente. El funcionamiento de éste es sencillo: el cliente envía una solicitud a una dirección en IPv6, constituida por el prefijo 64:ff9b:: seguido de la dirección IPv4 del destinatario. Este prefijo se suprime al traducirse a IPv4 y así la dirección restante es manejada conforme al protocolo IPv4.

3.1. Configuración del punto de acceso

De acuerdo al diseño anterior, es momento de definir la configuración de los servicios para el funcionamiento del punto de acceso. Así mismo, se define el software mediante el cual funciona la configuración, de modo que el lector pueda replicar el modelo empleado.

Los servicios del PA se encuentran configurados en su totalidad en un sistema operativo a la medida, el cual funciona correctamente al utilizar la versión 3.11.x del núcleo de Linux. Estos servicios, al igual que los módulos del núcleo utilizados, compilan y funcionan correctamente al utilizar esa versión de Linux. La arquitectura empleada en la construcción del punto de acceso es x86_64. Sin embargo, no es la única arquitectura que se puede utilizar. Esto mismo sucede con los módulos del núcleo y los servicios del punto de acceso, de modo que es posible compilarlos utilizando otra versión del núcleo de Linux, pero tal vez se necesite modificar el código fuente de los módulos o paquetes, debido a las diferencias en las cabeceras que existen entre diferentes versiones del núcleo.

3.1.1. TDR64

Para el módulo de traducción IPv6 a IPv4 se compiló Ecdysis versión 22 de Abril de 2014 [58], el cual es una implementación libre (licencia GPLv3) de TDR64 para generar un módulo del núcleo. Esta implementación hace uso de las bibliotecas y macros de Netfilter [59] para llevar a cabo la traducción de direcciones.

La compilación del código fuente requiere una versión del núcleo de Linux mayor que o igual a 2.6.31 y menor que o igual a 3.13.10 para generar el módulo `nf_nat64.ko`, el cual se encarga de añadir un prefijo (en particular se eligió `64:ff9b::/96`) a las direcciones IPv4 para hacer su traducción a IPv6. Este procedimiento se hace por medio de una interfaz virtual nat64 que conecta ambas interfaces incompatibles.

Para iniciar el módulo, se lanza desde algún archivo de carga o manualmente, indicando el prefijo y las direcciones IPv4 e IPv6 para conectar la interfaz virtual `nat64`.

Suponiendo que la dirección IPv4 proporcionada por el ISP es `104.95.215.25`, este procedimiento se puede realizar mediante el script mostrado en el listado 3.1:

```
1  #!/bin/bash
2
3  # Dirección IPv4 pública, prefijo y longitud de prefijo
4  IPV4_ADDR="104.95.215.25"
5  PREFIX_ADDR="64:ff9b::"
6  PREFIX_LEN="96"
7
8  # Verificamos que la dirección IPv6 tenga alcance global
9  if [[$(ip -6 addr show scope global | grep inet6 | wc -l) == 0]]; then
10   echo "No existe dirección IPv6 con alcance global"
11   exit 1
12 fi
13
14 # Eliminamos módulo nf_nat64, en caso de existir
15 modprobe -r nf_nat64
16
17 # Iniciar módulo nf_nat64.ko
18 insmod /etc/ecdysis/nf_nat64.ko nat64_ipv4_addr=$IPV4_ADDR \
19 nat64_prefix_addr=$PREFIX_ADDR nat64_prefix_len=$PREFIX_LEN
20
21 # Habilitar interfaz virtual nat64
22 ifconfig nat64 up
23
24 # Agregar ruta al prefijo de nat64
25 ip -6 route add ${PREFIX_ADDR}/${PREFIX_LEN} dev nat64
26
27 # Habilitamos la transferencia de paquetes en ambas interfaces
28 sysctl -w net.ipv4.conf.all.forwarding=1
29 sysctl -w net.ipv6.conf.all.forwarding=1
30
31 #Fin
```

Listado 3.1: Script que inicia y configura el módulo TDR64

Al configurar TDR64, es posible ejecutar el comando ping6 hacia una dirección IPv4 al concatenar el prefijo de traducción con la dirección IPv4. Por ejemplo, la dirección IPv4 del dns de google es `8.8.8.8` y podemos ejecutar el comando `ping6 64:ff9b::8.8.8.8` (o bien con notación hexadecimal `ping6 64:ff9b::808:808`) desde cualquier cliente IPv6, eventualmente conectado al punto de acceso, obteniendo respuesta.

3.1.2. DNS64

Para la resolución de direcciones IPv4 a IPv6, hacemos uso del paquete `unbound`, en particular la versión 1.5.10. Se trata de un servidor recursivo que realiza las búsquedas en IPv4, y también funciona como un traductor de registros A en AAAA a través de su función DNS64. Para llevar a cabo la tarea de traducir hace uso del puerto 53 por defecto. Todas las opciones de configuración se indican por defecto en el archivo `/etc/unbound.conf`.

La traducción de registros A en AAAA requiere la asignación del prefijo `nat64` para asignarlo a los nombres traducidos por TDR64 y dar prioridad a los registros A en caso de existir servidores con doble pila. Estas opciones se muestran a continuación:

```
server:
  # Configuración de módulo para el servidor
  module-config: "dns64 validator iterator"

  # Prefijo IPv6 para uso de DNS64
  dns64-prefix: 64:ff9b::/96

  # Preferimos registros A en vez de AAAA en el caso de servidores con
  # doble pila
  dns64-synthall: yes
```

Listado 3.2: Opciones de `unbound` para configurar DNS64

La opción `validator` en la configuración del módulo `dns64` indica que se desea realizar la validación de éstos por medio de DNSSEC con el objetivo de evitar el envenenamiento del caché de DNS. Para ello se puede especificar el archivo de los certificados anclas de confianza y se incrementa la seguridad del servidor indicando qué clientes están autorizados para hacer búsquedas recursivas, es decir los clientes asignados por el DHCPv6 de la red.

Estas opciones se incluyen en el mismo archivo de configuración como se muestra en el listado 3.3. Además, especificamos en forma estática el nombre del servidor local (para uso del servidor web) con su dirección IPv6, el cual no se encuentra incluido en las búsquedas del servidor recursivo.

```
  # Asignamos en forma estática el nombre del servidor local cp64
  local-zone: "localdomain." static
  local-data: "cp64. IN AAAA 2001:df8:0:7::1"

```

```
5   # Se resuelven consultas locales
6   interface: ::1
7
8   # Resolver consultas udp y tcp
9   do-udp: yes
10  do-tcp: yes
11
12  # Solo los clientes asignados en la red local inalámbrica por
13  # DHCPv6 pueden hacer consultas recursivas
14  access-control: 2001:df8:0:7::/64 allow
15
16  # Solo consultas del conjunto de registros A y AAAA para evitar
17  # ataques de escuchas
18  harden-glue: yes
19
20  # Uso de dnssec para zonas con anclas de confianza obligatorio
21  harden-dnssec-stripped: yes
22
23  # Evitar consultas con firmado dnssec y nombres nxdomain
24  harden-below-nxdomain: yes
25
26  # Archivo de claves de confianza, descarga desde
27  # https://secure.isc.org/ops/dlv/dlv.isc.org.key
28  dlv-anchor-file: "/usr/local/etc/unbound/dlv.isc.org.key"
29
30  # Archivos válidos de claves de confianza
31  trusted-keys-file: /usr/local/etc/unbound/keys.d/*.key
32
33  # Clave de confianza raíz
34  auto-trust-anchor-file: "/usr/local/etc/unbound/root.key"
35
36  # Eliminar datos no firmados de los mensajes seguros
37  val-clean-additional: yes
38
39  # Evitar mensajes erróneos, es decir modo no permisivo
40  val-permissive-mode: no
```

Listado 3.3: Opciones de unbound para configurar DNSSEC y el nombre del servidor local

Unbound también tiene varias opciones para personalizar la configuración del servidor dns recursivo. Algunas de ellas son: cambio de puerto de búsqueda, establecer nombre de usuario, cambio de archivo de configuración, asegurar privacidad de direcciones, tamaño de caché, rango de puertos de búsqueda, mostrar estadísticas de registros encontrados o bien de errores, control remoto, zonas de transferencia, zonas prohibidas, etc.

Ejecución de DNS64

Para iniciar el servicio, se puede invocar desde el archivo de carga o bien por comandos. El orden de ejecución de DNS64 con DNSSEC debe incluir la generación de claves (si no se ha hecho anteriormente), la creación de certificados y el inicio del servidor. La creación de claves y certificados genera la clave pública (incluida en el certificado) y privada, así como su certificado, tanto para el cliente, como para el servidor, conteniendo sus firmas respectivamente. Tal procedimiento se puede resumir en el script descrito en el listado 3.4:

```
1  #!/bin/bash
2
3  # Generación del ancla de confianza raíz (root.key)
4  unbound-anchor
5
6  # Generación de clave privada del servidor (unbound_server.key),
7  # certificado del servidor (unbound_server.pem), clave privada del
8  # cliente (unbound_control.key) y certificado del cliente
9  # (unbound_control.key)
10 unbound-control-setup
11
12 # Inicio del servidor dns64
13 unbound -c unbound.conf
```

Listado 3.4: Script para generación de clave de confianza raíz, claves y certificados e inicio del servidor dns64

Para hacer pruebas del sistema de servicio de nombres, es suficiente hacer una consulta en el explorador del cliente o bien hacer `ping6` hacia un nombre de dominio real existente en la red IPv4 y debe haber respuesta, de forma que la traducción se hace en forma transparente para el usuario.

3.1.3. DHCPv6

Para configurar la asignación de direcciones IPv6 en los clientes, se hace uso de `dnsmasq` [47], en particular la versión 2.69, cuya función principal es proporcionar un servidor DNS ligero y un servidor DHCP. En nuestro caso en particular, la parte importante se encuentra en el uso de la función de asignación de direcciones DHCPv6.

El archivo de configuración utilizado es `/etc/dnsmasq.conf`, el cual consta de una configuración sencilla: asignar el prefijo para la amplitud de direcciones de los clientes IPv6, el manejo de la asignación de la dirección del cliente (en este caso por anuncios del enrutador en vez de la asignación sin estados), el servidor y la interfaz que se maneja para hacer la asignación de direcciones. El puerto utilizado es 5553, porque el puerto por defecto (53) ya se utiliza por unbound.

Es necesario también indicar que DHCPv6 es autoritativo, es decir que no existe otro servidor DHCPv6 en la red local y en forma opcional indicar la no consulta al archivo de direcciones de servidores `resolv.conf`, de modo que indicamos la dirección del servidor en el mismo archivo de configuración de `dnsmasq`.

El archivo de configuración `dnsmasq.conf` se muestra en el listado 3.5. Para la ejecución del servicio de `dnsmasq` se puede hacer desde el archivo de carga o por comando con la instrucción única `dnsmasq`, comando que se encarga de iniciar el servidor por medio de la configuración encontrada en `dnsmasq.conf`.

```
# Definimos usuario y grupo
user=root
group=root

# Dirección que dnsmasq escucha, en este caso es local
listen-address=::1

# Para que no interfiera el puerto con unbound, lo cambiamos
port=5553

# Las consultas de servidores de nombres se realiza aquí y se indica
# con la instrucción server
no-resolv
server=2001:df8:0:7::1

# Interfaz utilizada por el servidor
interface=wlan0

# Tratamos las interfaces del servidor como una sola
bind-interfaces

# Habilitamos los anuncios del enrutador
enable-ra

# Establecemos la amplitud de direcciones con el prefijo, el modo de
# asignación (anuncios del enrutador) y vigencia de dirección
dhcp-range=2001:df8:0:7::,ra-only,infinite

# Indicamos la dirección del servidor DHCPv6
dhcp-option=option6:dns-server,[2001:df8:0:7::1]

# DHCPv6 autoritativo
dhcp-authoritative
```

Listado 3.5: Configuración del servidor DHCPv6 por `dnsmasq`

Para probar el protocolo es suficiente conectar un cliente a la red y, al detectarlo, el servidor le asigna una dirección IPv6 con el prefijo asignado por la configuración de `dnsmasq`.

3.1.4. Red inalámbrica

La configuración de red depende totalmente del punto de acceso. Para ello es necesario implementar un mecanismo en el espacio de usuario encargado de proporcionar a los usuarios la conexión a la red, así como su obtención de datos para autenticación. Para ello, utilizamos el paquete `hostapd` [40], versión 2.6, con soporte para Linux y el controlador `mac80211`, que es el manejador de tramas para el protocolo 802.11.

El uso de `mac80211` también implementa las llamadas del API `cfg8011`, las cuales sirven para el registro de la red y su configuración. Finalmente, `cfg80211` (contenida en el espacio del núcleo) se maneja en conjunto a la interfaz interespacio `nl80211` (cabecera de la interfaz pública de enlace de red) para reemplazar las extensiones inalámbricas (*Wireless-Extensions*) al hacer uso del uso la familia de sockets Netlink (en vez de ioctl) para la comunicación interprocesos [15].

La comunicación de `hostapd` con el hardware a través de sus controladores se muestra en la figura 3.2.

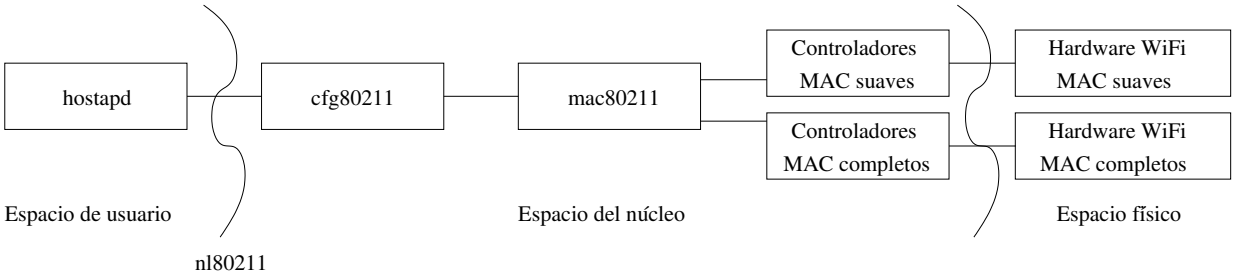

Figura 3.2: Comunicación de los controladores que depende de `mac80211`, utilizado por `hostapd`.

Para configurar el punto de acceso, creamos el archivo `/etc/hostapd.conf`, el cual debe contener de inicio el mecanismo para permitir a los usuarios su conexión a la red. Ésta configuración se compone del SSID de la red (el cual en particular es `cp64`), el nombre de la interfaz que utiliza el punto de acceso inalámbrico (`wlan0`), la banda que utiliza para conexión (g, a 2.4GHz), el canal por utilizar y el soporte al protocolo 802.11n. Esto se muestra en el listado 3.6.

```
1 # Interfaz utilizada por el punto de acceso
2 interface=wlan0
3
4 # Interfaz de comunicación entre el espacio de usuario (hostapd) y
5 # el espacio de núcleo (cfg80211 y el controlador mac80211)
```

```
6  driver=nl80211
7
8  # Nombre del id de conjunto de servicios del punto de acceso
9  ssid=cp64
10
11 # Modo de hardware (banda 2.4 GHz)
12 hw_mode=g
13
14 # Canal (cambiar para evitar interferencia con otros canales)
15 channel=10
16
17 # Soporte al protocolo 802.11n
18 ieee80211n=1
19
20 # Hacer disponible el nombre del punto de acceso a los clientes
21 ignore_broadcast_ssid=0
22
23 # Soporte a QoS
24 wme_enabled=1
```

Listado 3.6: Configuración del punto de acceso inalámbrico para permitir el acceso de los clientes a la red

Para autenticar a los usuarios, `hostapd` también proporciona el manejo de claves utilizando el estándar 802.1X, el cual consta de un algoritmo de autenticación (en este caso WPA2) y un mecanismo para autenticación (EAP). De modo que al hacer uso de WPA2-EAP `hostapd`, se realiza la solicitud al servidor de autenticación RADIUS (en conjunto con el secreto compartido) de una respuesta de autenticación para el cliente, según los datos ingresados por éste. En caso de ser correcta la respuesta de RADIUS, `hostapd` permite el acceso del usuario a la red.

La configuración incluye la dirección y puerto del servidor RADIUS (en este caso es `2001:df8:0:7::1` con puerto 1812), así como el secreto compartido con RADIUS. Esta configuración se muestra en el listado 3.7.

```
1  # Uso del estándar 802.1X
2  ieee8021x=1
3
4  # Versión de EAPoL
5  eapol_version=2
6
7  # Uso del algoritmo WPA2
8  wpa=2
9
10 # Mecanismo de autenticación
```

```
11 wpa_key_mgmt=WPA-EAP
12
13 # Algoritmo de manejo de claves
14 wpa_pairwise=CCMP TKIP
15
16 # Servidor RADIUS independiente de hostapd
17 eap_server=0
18
19 # Dirección de hostapd
20 own_ip_addr=2001:df8:0:7::1
21
22 # Dirección de radius (servidor local)
23 auth_server_addr=::1
24
25 # Puerto de radius
26 auth_server_port=1812
27
28 # Secreto compartido con radius
29 auth_server_shared_secret=testing123
```

Listado 3.7: Configuración de los algoritmos empleados, así como la información enviada a RADIUS para autenticación de los clientes. También se incluyen en `hostapd.conf`

Para probar el servidor de autenticación, un cliente se intenta conectar a la red, y antes de acceder aparece una ventana de autenticación, en donde el cliente introduce su información de autenticación. Si esta información es correcta, se le concede el acceso. De lo contrario, sigue apareciendo la ventana de autenticación indefinidamente o bien se muestra una ventana de rechazo.

Para iniciar el servicio, debemos desactivar el modo avión (en caso de existir), iniciar el servicio hostapd, iniciar la interfaz de red `wlan0` y añadir una ruta fija. Este procedimiento se puede simplificar con el script mostrado en el listado 3.8.

```
1  #!/bin/bash
2
3  # Desbloquear el adaptador inalámbrico en caso de estarlo
4  # por ejemplo, interfaz en modo avión
5  rfkill unblock all
6
7  # Iniciar el servicio hostapd de acuerdo a su configuración
8  hostapd /[ruta_configuración]/hostapd.conf -B
9
10 # Iniciar interfaz inalámbrica
11 ifconfig wlan0 up
12
```

```
13 # Asignar dirección estática y ruta en IPv6, para la interfaz
14 #inalámbrica
15 ip -6 addr add 2001:df8:0:7::1/16 dev wlan0
16 route -A inet6 add default gw 2001:df8:0:7::1 dev wlan0
```

Listado 3.8: Inicio del punto de acceso hostapd. Se inicia el servicio, así como las interfaz estática con la regla de enrutamiento, para IPv6

El servicio `hostapd` se inicia y se puede proteger con reglas para filtrar el tráfico entre la interfaz virtual `nat64` y `wlan0`, reglas que se exponen en la sección del cortafuegos.

3.1.5. Servidor de autenticación

El servidor RADIUS es manejado por medio de Freeradius [5]. La configuración de freeradius consta de dos partes: la configuración del servidor y la configuración del cliente. Ambas son requeridas porque en el mismo equipo se encuentran el servidor de autenticación y el cliente de autenticación local.

La configuración del servidor se maneja en el archivo `radiusd.conf` y la configuración del cliente se maneja en `clients.conf`. Las opciones de configuración son algo extensas para explicar en una sección, por lo tanto explicamos solo las opciones de configuración necesarias para la autenticación WPA2 con EAP-GTC tanto en el servidor RADIUS como en el cliente.

Configuración del servidor de autenticación

Para configurar el servidor, es necesario indicar la dirección que el servidor escucha. Esto se refiere a las direcciones en el protocolo IPv6, el número de puerto (1812) y el tipo de paquetes por escuchar, (en particular, necesitamos los paquetes de autenticación), tal como se muestra en el listado 3.9.

```
1  listen {
2    # El tipo de paquetes escuchados son de autenticación
3    type = auth
4
5    # Escuchamos las solicitudes de las direcciones IPv6
6    ipv6addr = ::
7
8    #  Número de puerto para escuchar las solicitudes
9    port = 1812
10 }
```

Listado 3.9: Configuración básica de autenticación en el servidor RADIUS, indicada en `radiusd.conf`.

En forma adicional, en el archivo `eap.conf` especificamos el modo de autenticación. Al establecer la sesión EAP, en el túnel se necesita un algoritmo. Para PEAP especificamos el tipo de *tokens* GTC, para permitirnos cifrar las contraseñas en el servidor.

La configuración de EAP es algo extensa, por lo tanto partiendo del archivo de configuración eap.conf resultante de la compilación de freeradius versión 2.17, mostramos en el listado 3.10 las modificaciones para hacer uso del método de autenticación cifrada EAP-GTC.

```
1  # Esta configuración se invoca desde la configuración de módulos,
2  # ubicada en radiusd.conf, así la inclusión del archivo de configuración
3  # eap.conf se hace con $INCLUDE eap.conf
4  eap {
5      #  Para efecto de simplificación solo mostramos la configuración
6      #  perteneciente a EAP-GTC (Generic Token Card)
7      gtc {
8        # El modo de autenticación PAP permite almacenar contraseñas
9        # cifradas en vez de en claro
10       auth_type = PAP
11     }
12
13     # El túnel de sesión PEAP necesita un tipo por defecto en forma
14     # separada que el módulo para EAP no en túnel. El tipo elegido es
15     # EAP-GTC
16     peap {
17       default_eap_type = gtc
18
19       #  Evitamos la solicitud en el túnel
20       copy_request_to_tunnel = no
21       use_tunneled_reply = no
22
23       # Solicitud enviada a través de un servidor virtual
24       virtual_server = "inner-tunnel"
25     }
26   }
```

Listado 3.10: Configuración básica para hacer uso del método de autenticación EAP-GTC en el archivo de configuración `eap.conf`

Configuración del cliente

La configuración del cliente RADIUS se realiza en el archivo `clients.conf`, el cual se invoca desde el archivo de configuración del servidor `radiusd.conf`. La configuración del cliente consta de la dirección del cliente, que al tratarse de un cliente local en IPv6 (en la misma tarjeta) su dirección es ::1, además se incluye el secreto y si el nombre del NAS. El listado

3.11 muestra la configuración del cliente RADIUS.

```
client nat64ipv6 {
  # Dirección IPv6 del cliente. Como el cliente y servidor se
  # encuentran en el mismo equipo, se elige el local en IPv6
  ipv6addr = ::1

  # Secreto compartido entre el NAS y RADIUS
  secret = testing123

  # Opcional para Freeradius > 2.0, nombre de cliente
  shortname = nat64ipv6

  # NAS local
  nastype = other
}
```

Listado 3.11: Configuración del cliente RADIUS, su dirección es local (::1) y se localiza en el archivo `clients.conf`.

Finalmente configuramos la lista de usuarios, la cual se realiza en el archivo users, el cual puede incluir una lista de contraseñas externas por medio de `$INCLUDE [archivo_contraseñas]`. Por medio de las ventajas de EAP-GTC, podemos agregar el digesto de las contraseñas, con el objetivo de proteger el servidor mismo de ataques. Al utilizar el módulo `rlm_pap` tenemos varias alternativas para realizar el digesto de nuestra contraseña, por ejemplo Crypt (UNIX), MD5, Algoritmo de Digesto Seguro (SHA), entre otros [22].

Para obtener la contraseña utilizando el algoritmo Crypt utilizando el comando `radcrypt --des [contraseña]`. Y el digesto lo escribimos dentro de la lista de contraseñas siguiendo el formato `Crypt-Password := "digesto\_producto\_de\_radcrypt"`.

Para aumentar la seguridad, podemos hacer uso de SHA, para el cual no se han encontrado todavía colisiones (2016), utilizando el script del listado 3.12 para la obtención del digesto.

```
# coding: utf8
import sys
import hashlib
import base64

n = len( sys.argv )
if n != 2:
  print "Args: contraseña"
  sys.exit(1)

h = hashlib.sha1()
```

```
h.update( sys.argv[1] )
print base64.b64encode( h.digest() )
```

Listado 3.12: Script `passha.py` para obtener el digesto SHA para las contraseñas.

Si el texto `abcd` es el resultado de ejecutar el script de python con la contraseña en claro como argumento, en el archivo de configuración se agrega `SHA-Password := "abcd"`. De esta forma, se autentica el cliente, cuando su nombre de usuario y el digesto de su contraseña coinciden con los de la lista de usuarios de la red inalámbrica interna.

Para ejecutar el servidor, es suficiente escribir radiusd, lo cual puede hacerse desde el archivo de carga o como comando y esperar las solicitudes de los clientes en modo oculto. En caso de querer mostrar el programa como hilo principal, escribimos `radiusd -X`.

3.1.6. Servidor web

Con el objetivo de proveer a los dispositivos no autenticados un portal con las instrucciones de acceso al sistema, podemos implementar un servidor web ligero con una única página que contenga esas instrucciones y además que tenga compatibilidad para el protocolo IPv6.

El paquete `mini_httpd` [3] es un servidor web ligero que cumple con ese objetivo. La versión que vamos a utilizar es 1.25 y sin compilar la instancia a SSL para simplificar las características de éste, cuya función particular es mostrar solamente una página web.

La configuración del servidor `mini_httpd` es mucho más sencilla que el tradicional servidor Apache [54], y se hace mediante el archivo `mini_httpd.conf`. En este archivo definimos un puerto (preferentemente distinto al 80 para hacer la redirección más transparente al usuario), la dirección del anfitrión y la ruta del directorio raíz para los archivos web principalmente. Esta configuración se muestra en el listado 3.13.

```
# Número de puerto para iniciar el servidor web
port=5280

# Usuario
user=root

# Dirección del anfitrión
host=2001:df8:0:7::1

# Ruta del directorio raíz de los archivos web
data_dir=/var/www/html

# Archivo de bitácora, identificador de proceso y conjunto de caracteres
logfile=/var/log/mini-httpd.log
pidfile=/var/run/mini-httpd.pid
charset=iso-8859-1
```

Listado 3.13: Configuración del servidor web ligero `mini_httpd`, compatible con IPv6. Configuración contenida en `mini_httpd.conf`.

Una vez configurado el servidor web, es necesario incluir el archivo index.html en el directorio raíz de datos, archivo que consta de las instrucciones para que los usuarios cuyos dispositivos no han sido registrados, puedan ingresar a la red. Este archivo es muy simple como incluir el mensaje de texto "Usted se ha autenticado con un dispositivo anónimo, con privilegios limitados. Póngase en contacto con el administrador del sistema para tener acceso a la red".

Para iniciar el servidor web, se realiza por medio de la instrucción única `mini_httpd -C /[ruta_archivo_configuración]/mini_httpd.conf`, ya sea desde el archivo de carga o en la línea de comandos. Con esto se abre el archivo por defecto de red (`index.html`) al intentar abrir el servidor en el puerto indicado en la configuración.

3.1.7. Cortafuegos

El filtrado de paquetes es un elemento fundamental para la seguridad en las redes, ya que intercepta los paquetes provenientes de fuentes desconocidas para impedir lograr su propósito, que generalmente puede ser no deseado. El paquete por excelencia para el filtrado de tráfico en IPv6 es `ip6tables`, el cual es parte de la suite de Netfilter 6 y se encuentra en el espacio de usuario. La versión del núcleo de Linux mínima para uso de `ip6tables` es 2.6.32 y la versión 3.6.0, para permitir el NAT, enmascarado y marcado de paquetes en IPv6 [59].

Las reglas para el filtrado de paquetes se realizan en primer lugar para el tráfico entre la interfaz `wlan0` y la interfaz virtual `nat64`, con el objetivo de proteger la identidad del tráfico saliente de la red interna por medio del enmascarado en IPv6 y la transferencia de paquetes entre ambas interfaces. Estas reglas las podemos ver en el listado 3.14.

```
1  # Limpiamos tablas
2  ip6tables -F
3  ip6tables -X
4  ip6tables -F -t nat
5  ip6tables -F -t mangle
6
7  # Política de transferencia de paquetes para interfaz local: aceptar
8  ip6tables -A INPUT -i lo -j ACCEPT
9  ip6tables -A OUTPUT -o lo -j ACCEPT
10
11 # Enmascarar tráfico saliente a la interfaz virtual nat64
12 ip6tables -t nat -A POSTROUTING -o nat64 -j MASQUERADE
13
14 # Permitir el tráfico entre interfaces wlan0 y nat64
15 ip6tables -A FORWARD -i nat64 -o wlan0 -j ACCEPT
16 ip6tables -A FORWARD -i wlan0 -o nat64 -j ACCEPT
```

Listado 3.14: Reglas de filtrado de paquetes entre la interfaz interna (`wlan0`) y la interfaz virtual (`nat64`), se enmascara el tráfico saliente para proteger la identidad de los clientes.

Dado a que todos los servicios se encuentran en el lado IPv6 de la red, no es necesario aplicar reglas en la interfaz IPv4 (es decir establecer reglas entre la interfaz `nat64` y la interfaz conectada al exterior en IPv4). Sin embargo, el objetivo de esta sección no es medir el nivel de seguridad al implementar un esquema de cortafuegos doble, ya que al aplicar un único cortafuegos en IPv6 se obtiene un buen nivel de seguridad.

Para proteger la red de accesos por parte de dispositivos desconocidos, hacemos uso del marcado y redirección de paquetes de `ip6tables` (`MARK` y `REDIRECT`). Sin embargo, estas reglas las veremos en la sección de autenticación por MAC.

La ejecución de las reglas solo necesitan tener iniciados los módulos de red en IPv6 y haber iniciado previamente las interfaces. Para probar su funcionamiento, se debe mostrar que la transferencia de paquetes sigue las reglas establecidas por el cortafuegos.

3.1.8. Autenticación MAC

Además de la autenticación de los usuarios con sus contraseñas a través de EAP-GTC, podemos incrementar la seguridad al autenticar los dispositivos. Para autenticar un dispositivo necesitamos la dirección MAC de éste, con la cual creamos reglas para permitir su acceso a internet a través del cortafuegos cuando se trata de un dispositivo conocido. En caso contrario, se crean reglas que impiden el acceso del usuario y redirige a una ventana que informa al usuario con dispositivo desconocido cómo poder registrarse. En este caso, el registro debe ser a través de un administrador.

Las reglas para autorizar los dispositivos a utilizar a internet o redirigir su acceso a la página web local son las siguientes:

- Todos los paquetes de entrada son marcados con 1.
- Los paquetes de los clientes autorizados por el administrador (es decir, cuyos dispositivos son conocidos) son marcados con 2.
- Los paquetes marcados con 1 son redirigidos a la página web local con las instrucciones para acceder al sistema, mientras que a los paquetes marcados con 2 se les concede el acceso a internet.

Estas reglas de filtrado, las cuales hacen uso del paquete `ip6tables`, se pueden construir de acuerdo al listado 3.15.

```
1 # Todos los paquetes entrantes son marcados con 1, sin importar si éstos
2 # provienen de una fuente conocida o de una desconocida.
3 ip6tables -A PREROUTING -i wlan0 -t mangle -j MARK --set-mark 1
```

```
4
5  # Redirigimos los paquetes que permanecen marcados con 1, los cuales
6  # son enviados al puerto donde se encuentra el servidor web.
7  ip6tables -t nat -A PREROUTING -p tcp -m mark --mark 1 --dport 80 \
8    -j REDIRECT --to-port 5280
9  ip6tables -t nat -A PREROUTING -p tcp -m mark --mark 1 --dport 443 \
10   -j REDIRECT --to-port 5280
11
12 # Los clientes cuyos dispositivos son autorizados a través de su
13 # dirección MAC, se marcan con 2. Éstos paquetes son transmitidos a su
14 # destino en forma normal
15 ip6tables -A PREROUTING -t mangle -m mac --mac-source \
16   ac:29:3a:a0:a2:2e -j MARK --set-mark 2
17 ip6tables -A PREROUTING -t mangle -m mac --mac-source \
18   99:88:77:66:55:44 -j MARK --set-mark 2
```

Listado 3.15: Reglas de filtrado para marcar los paquetes de los dispositivos de acuerdo al nivel de autorización otorgado por el administrador.

De este modo, para llevar a cabo la transmisión, el dispositivo tiene que ser registrado por medio de su MAC, o por lo contrario no tiene acceso a la red externa. De tal forma, se obtienen por medio de la autenticación tres condiciones:

- El cliente autenticado con un dispositivo conocido, tiene acceso a la red externa.
- El cliente autenticado con un dispositivo desconocido, tiene acceso al servidor web local, que contiene las instrucciones de registro y sin acceso a la red externa.
- El cliente no autenticado, no tiene acceso a la red externa ni al servidor web local.

El administrador tiene dos papeles importantes: agregar el digesto de las contraseñas de los usuarios en la lista de usuarios del servidor de autenticación (archivo `users`) y crear las reglas en el cortafuegos incluyendo las direcciones MAC de los clientes seguros en éste, para evitar su redirección a la página de instrucciones y así poder hacer uso de la red. Para hacer uso de estas dos funciones se creó el programa `registraUsuario`.

La invocación de `registraUsuario` consta de tres argumentos: la MAC del cliente, el nombre de usuario y la contraseña en claro. El formato de la MAC consta de 6 conjuntos de 8 bits con valores hexadecimales y separados por :, es decir, XX:XX:XX:XX:XX:XX, con XX entre 00 y FF, inclusive. La validación de la dirección MAC se realiza por medio de la función indicada en el listado 3.16.

```
1  /* Validar dirección MAC */
2  /* 1-> No hexadecimal, 0 -> OK */
3  int validaMAC(char *mac)
4  {
```

```
5   int i;
6   for (i=0;i<17;i++)
7   {
8     if (i%3 != 2 && !isxdigit(mac[i]))
9       return 1;
10    if (i%3 == 2 && mac[i] != ':')
11      return 1;
12  }
13  if (mac[17]!= '\0')
14    return 1;
15  return 0;
16 }
```

Listado 3.16: Validación de dirección MAC

Para obtener el digesto de la contraseña original, hacemos uso de la aplicación `radcrypt` utilizando la función para obtener el digesto, indicada en el listado 3.17.

```
1  /* Función para obtener el digesto de la contraseña original */
2  char *exec_radcrypt(char * pass)
3  {
4    FILE *crypt;
5    char radcrypt[MAXBUFF];
6    char *passhash=(char *)malloc(sizeof(char)*15);
7    sprintf(radcrypt, "radcrypt --des %s\n", pass);
8    crypt=popen(radcrypt,"r");
9    if(!crypt)
10   {
11     printf ("Error al abrir radcrypt\n");
12     exit(1);
13   }
14   /* Obtenemos digesto */
15   if (!fgets(passhash, 14, crypt))
16   {
17     printf("Error al recuperar digesto de la contraseña\n");
18     pclose(crypt);
19     exit(1);
20   }
21   pclose(crypt);
22   return passhash;
23 }
```

Listado 3.17: Función para obtener el digesto del password original, utilizando el algoritmo Crypt de Unix

Para añadir los usuarios, tanto en la lista de contraseñas como sus direcciones MAC con el marcado en 2, agregamos las instrucciones utilizando la función `ipt_rad`, cuyos argumentos son los mismos que en los argumentos para la invocación del programa en la línea de comandos; función mostrada en el listado 3.18.

```
/* Función para crear la regla de admisión de la MAC y su
   correspondiente usuario y contraseña */
int ipt_rad(char * mac, char * user, char * hash)
{
  FILE *ipt,*rad;
  rad=fopen("/usr/local/etc/raddb/usuarios", "a+");
  ipt=fopen("/etc/mark.sh", "a+");
  if(ipt!=NULL && rad!=NULL)
  {
    /* Agregamos regla de marcado 2 (acceso a internet) al usuario por
    registrarse */
    fprintf(ipt,"ip6tables -A PREROUTING -t mangle -m mac --mac-source
    %s -j MARK --set-mark 2\n",mac);
    fclose(ipt);
    /* Agregamos nombre de usuario y digesto de la contraseña a la lista
    de usuarios de Freeradius */
    fprintf(rad,"%s Crypt-Password := \"%s\"\n",user,hash);
    fclose(rad);
  }
  else
  {
    /* Errores al agregar archivos */
    if (rad!=NULL)
    {
      printf("Error: No existe /etc/mark.sh, saliendo\n");
      fclose(rad);
      exit(1);
    }
    else
    {
      printf("Error: No existe /usr/local/etc/raddb/usuarios, saliendo
      \n");
      if (ipt!=NULL) fclose(ipt);
      exit(1);
    }
  }
  return 0;
}
```

Listado 3.18: Función para escribir el nombre de usuario y contraseña en la lista de usuarios en Freeradius, y agregar la regla de marchado 2 para las direcciones MAC de los usuarios.

Finalmente, tenemos la función principal, que realiza la invocación de la validación de argumentos, y la invocación de las función del registro de la MAC, nombre de usuario y contraseña, según corresponda. Esto se muestra en el listado 3.19.

```
1  /* Función para crear la regla de admisión de la MAC y su
2     correspondiente usuario y contraseña */
3  int main(int argc, char *argv[])
4  {
5    char passhash[15];
6
7    if (argc!=4)
8    {
9      fprintf(stderr, "Sintaxis: %s <MAC cliente> <Usuario> <Password>\n",
10     argv[0]);
11     return 1;
12   }
13   strcpy(passhash, exec_radcrypt(argv[3]));
14   if (validaMAC(argv[1])!=0)
15   {
16     printf("Error: MAC debe tener el formato XX:XX:XX:XX:XX:XX,
17     00<=XX<=FF\n");
18     return 1;
19   }
20   ipt_rad(argv[1], argv[2], passhash);
21   return 0;
22 }
```

Listado 3.19: Función principal de `registraUsuario`.

3.2. Pruebas del punto de acceso

Ahora que está construido el punto de acceso seguro en IPv6, es momento de hacer las pruebas para evaluar el producto creado.

Se ha creado un sistema operativo mínimo con la versión del núcleo de Linux 3.11.10, tomado de Fedora 20. La creación del sistema operativo mínimo contiene `grub2` y el archivo de carga para iniciar los servicios indicados en las secciones anteriores.

3.2.1. Pruebas de traducción TDR64 y DNS64

Para probar la traducción de direcciones y los nombres para un cliente IPv6, es necesario realizar las búsquedas en forma normal para la red actual en IPv4 y en forma transparente debe existir respuesta.

Para verificarlo hacemos dos búsquedas (con dirección y nombre) en cada caso. Una se realiza por medio del Protocolo de Mensajes de Control de Internet versión 6 (ICMPv6), es decir el comando ping6. La otra se realiza por medio del Protocolo de Transferencia de Hiper-texto (HTTP) a través de una página web.

Prueba TDR64

Antes de hacer la prueba en el cliente, mostramos en el servidor con el comando ifconfig las interfaces existentes en éste, en donde `eth0` es la interfaz conectada a IPv4, `wlan0` es la interfaz que conecta a la red interna en IPv6, `lo` es la interfaz local y `nat64` es la interfaz virtual encargada de la traducción. Las tomas de pantalla del servidor en donde se muestran estas interfaces se encuentran en las figuras 3.3 y 3.4.

```
eth0      Link encap:Ethernet  HWaddr 08:00:27:D2:4E:D8
          inet addr:172.16.214.129  Bcast:172.16.214.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3307 errors:0 dropped:0 overruns:0 frame:0
          TX packets:269 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:532945 (520.4 KiB)  TX bytes:75902 (74.1 KiB)

wlan0     Link encap:Ethernet  HWaddr 08:00:27:D3:06:4F
          inet6 addr: 2001:df8:0:7::2/16 Scope:Global
          inet6 addr: fe80::a00:27ff:fed3:64f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:28 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:3900 (3.8 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:13 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

Figura 3.3: Interfaces de red en el servidor al iniciar el sistema (`eth0`, `wlan0` y `lo`).

```
nat64     Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
-00
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Figura 3.4: Interfaces de red en el servidor al iniciar el sistema (`nat64`).

Después de la conexión de un cliente (en particular un cliente Ubuntu 14.04.1), hacemos la prueba de traducción de direcciones haciendo `ping6` sobre el prefijo `nat64` (`64:ff9b::`) con la dirección IPv4 real, se toma como ejemplo la dirección web de la descarga del módulo TDR64 ecdysis nat64 (`jazz-v4.viagenie.ca`), cuya dirección es `206.123.31.2` (y al concatenarse el prefijo da como resultado `64:ff9b::206.123.31.2` o bien en hexadecimal `64:ff9b::ce7b:1f02`) tal como se muestra en la figura 3.5.

```
root@jogam-VirtualBox:/home/jogam# ping6 64:ff9b::206.123.31.2
PING 64:ff9b::206.123.31.2(64:ff9b::ce7b:1f02) 56 data bytes
64 bytes from 64:ff9b::ce7b:1f02: icmp_seq=1 ttl=53 time=122 ms
64 bytes from 64:ff9b::ce7b:1f02: icmp_seq=2 ttl=53 time=124 ms
^C
--- 64:ff9b::206.123.31.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 122.930/123.957/124.984/1.027 ms
```

Figura 3.5: Respuesta a una consulta ICMPv6 nat64 de una dirección en la red externa IPv4.

Y haciendo una consulta `nat64` por HTTP desde el cliente a la misma dirección (para hacer una consulta en el navegador se hace entre corchetes la dirección ipv6), es decir `http://[64:ff9b::ce7b:1f02]`, también hay respuesta, como se muestra en la figura 3.6.

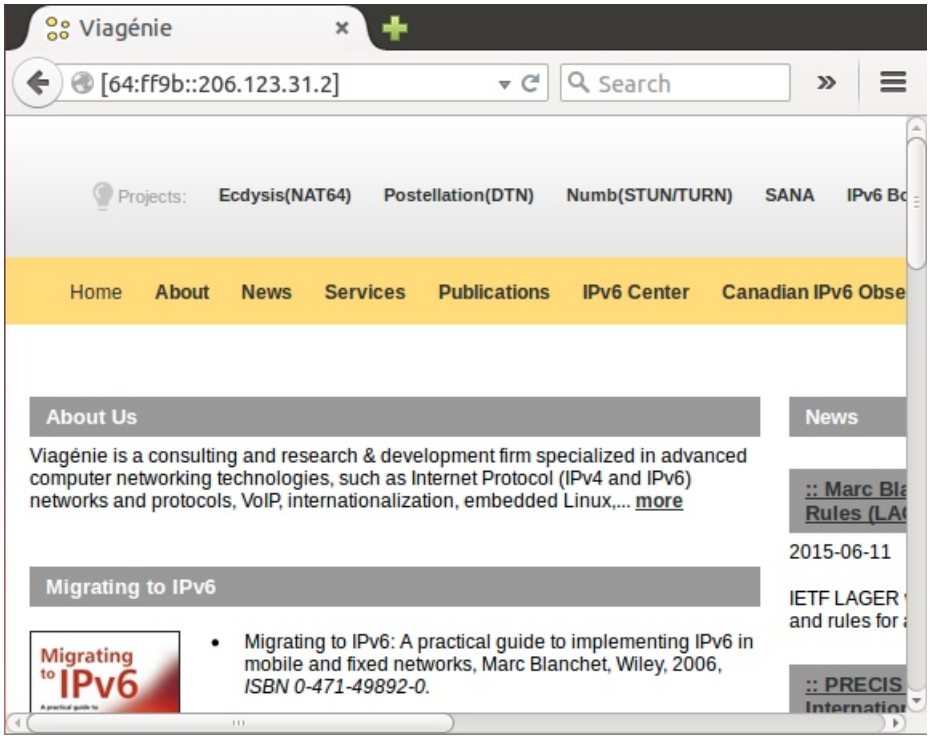

Figura 3.6: Respuesta a una consulta HTTP nat64 de una dirección en la red externa IPv4.

Prueba DNS64

Para probar la traducción de direcciones, al igual que en nat64 la hacemos por medio de ICMPv6 y HTTP, pero incluyendo el nombre original en IPv4. Haciendo la búsqueda en el cliente con la web de ecdysis (`jazz-v4.viagenie.ca`) también obtenemos respuesta a la búsqueda de los nombres en IPv6.

```
root@jogam-VirtualBox:/home/jogam# ping6 jazz-v4.viagenie.ca
PING jazz-v4.viagenie.ca(jazz.viagenie.ca) 56 data bytes
64 bytes from jazz.viagenie.ca: icmp_seq=1 ttl=53 time=131 ms
64 bytes from jazz.viagenie.ca: icmp_seq=2 ttl=53 time=135 ms
64 bytes from jazz.viagenie.ca: icmp_seq=3 ttl=53 time=136 ms
^C
--- jazz-v4.viagenie.ca ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 131.628/134.493/136.048/2.072 ms
```

Figura 3.7: Respuesta a una consulta ICMPv6 de un nombre de registro A convertido en AAAA conservando el mismo nombre.

Y haciendo uso del protocolo HTTP también tenemos respuesta, utilizando la misma dirección como ejemplo. También hay respuesta.

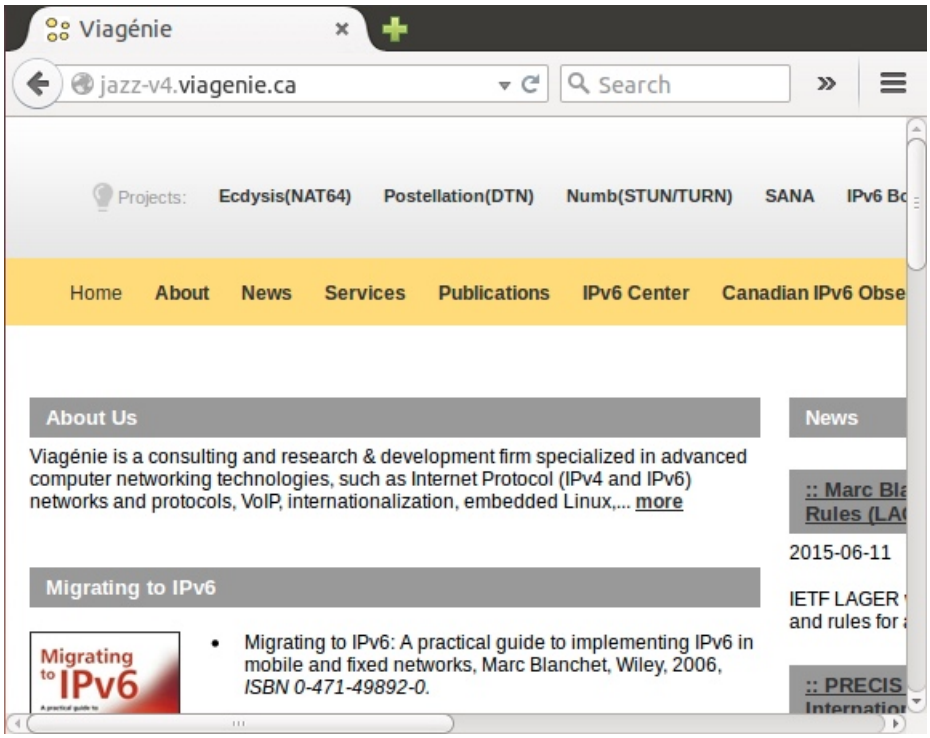

Figura 3.8: Respuesta a una consulta HTTP de un nombre de registro A convertido en AAAA conservando el mismo nombre.

Y así queda demostrado el funcionamiento del proceso de traducción de direcciones y nombres para nuestro punto de acceso.

3.2.2. Prueba de asignación de direcciones (DHCPv6)

La asignación dinámica de direcciones se realiza inmediatamente después de autenticarse el cliente. En este caso el cliente adquiere una dirección IPv6 con el prefijo indicado en la configuración de `dnsmasq`, que en este caso es `2001:df8:0:7::`. La asignación de la dirección por DHCPv6 se realiza una vez conectado el usuario a la red, sin importar si el dispositivo del usuario sea conocido o no, ya que el filtrado para esta condición se realiza por medio del cortafuegos.

Para probar el funcionamiento de DHCPv6 conectamos el cliente a la red interna y comprobamos la asignación de su dirección IPv6 por el comando ifconfig tal como se muestra en la figura 3.9.

```
wlan0     Link encap:Ethernet  HWaddr 08:00:27:96:48:6b
          inet6 addr: fe80::a00:27ff:fe96:486b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:46867 errors:0 dropped:0 overruns:0 frame:
0
          TX packets:84077 errors:0 dropped:0 overruns:0 carrie
r:0
          collisions:0 txqueuelen:1000
          RX bytes:8841302 (8.8 MB)  TX bytes:8516872 (8.5 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:45617 errors:0 dropped:0 overruns:0 frame:
0
          TX packets:45617 errors:0 dropped:0 overruns:0 carrie
r:0
          collisions:0 txqueuelen:0
          RX bytes:4165059 (4.1 MB)  TX bytes:4165059 (4.1 MB)
```

Figura 3.9: Interfaz `wlan0` del cliente antes de la conexión a la red. Nótese que no tiene conexión a la red de alcance global.

De este modo, el cliente tiene acceso a la red interna IPv6 y de acuerdo a las condiciones del administrador, puede tener acceso a la red exterior. La asignación de la dirección por DHCPv6 se muestra en la figura 3.10, en donde ahora tenemos la dirección IPv6 de alcance global, necesaria para conectarse al servidor.

```
root@jogam-VirtualBox:/home/jogam# ifconfig
wlan0     Link encap:Ethernet  HWaddr 08:00:27:96:48:6b
          inet6 addr: 2001:df8:0:7:ecbb:5822:84e8:ed8f/64 Scope
:Global
          inet6 addr: fe80::a00:27ff:fe96:486b/64 Scope:Link
          inet6 addr: 2001:df8:0:7:a00:27ff:fe96:486b/64 Scope:
Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:46868 errors:0 dropped:0 overruns:0 frame:
0
          TX packets:84115 errors:0 dropped:0 overruns:0 carrie
r:0
          collisions:0 txqueuelen:1000
          RX bytes:8841460 (8.8 MB)  TX bytes:8523725 (8.5 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:45675 errors:0 dropped:0 overruns:0 frame:
0
          TX packets:45675 errors:0 dropped:0 overruns:0 carrie
r:0
```

Figura 3.10: Interfaz `wlan0` después de la conexión a la red. Nótese la presencia de la conexión de alcance global `2001:df8:0:7:ecbb:5822:84e8:ed8f/64`, asignada por DHCPv6.

3.2.3. Prueba de autenticación con el servidor

Ahora vamos a realizar pruebas de autenticación. Sea un cliente Mac OS X versión 10.10.5, y quiere acceder a la red que ofrece nuestro punto de acceso. Gracias a hostapd, el cliente puede ver la red *cp64* en la lista de las redes disponibles WiFi mostrado en la figura 3.11.

Figura 3.11: Lista de conexiones WiFi disponibles. Nótese el SSID del punto de acceso cp64.

Cuando el cliente intenta conectarse a la red, se le solicitan sus datos de autenticación (usuario y contraseña), así como la red para aplicar el protocolo 802.1X, datos que son enviados al servidor RADIUS para obtener una respuesta de parte de éste, sea la autorización o rechazo del cliente, según sus datos. La ventana de autenticación se muestra en la figura 3.12.

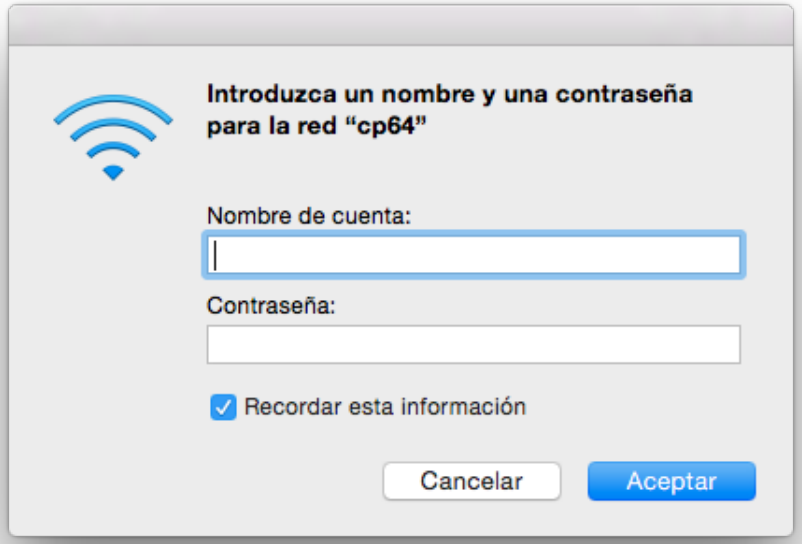

Figura 3.12: Ventana de autenticación. El usuario y contraseña son individuales y seleccionar 802.1X es automático.

En caso de ser correcta la autenticación (usuario y contraseña correctos), se le concede el acceso al usuario mostrando la conexión de éste como se ve en la figura 3.13, sin importar se trata de un dispositivo conocido o desconocido en esta parte de la autenticación. Si la información de autenticación es incorrecta, simplemente vuelve a aparecer la pantalla de autenticación hasta introducirse la información de autenticación correcta o cancelarse el intento de acceder.

La autenticación por 802.1X permite hacer uso del método EAP-GTC que protege la contraseña al almacenar el digesto de las contraseñas en el servidor de autenticación.

Como parte del control de usuario, se añade al usuario un contador con el tiempo de conexión a la red, aunque sin auditoría de parte del servidor, tal y como se muestra en la figura 3.13.

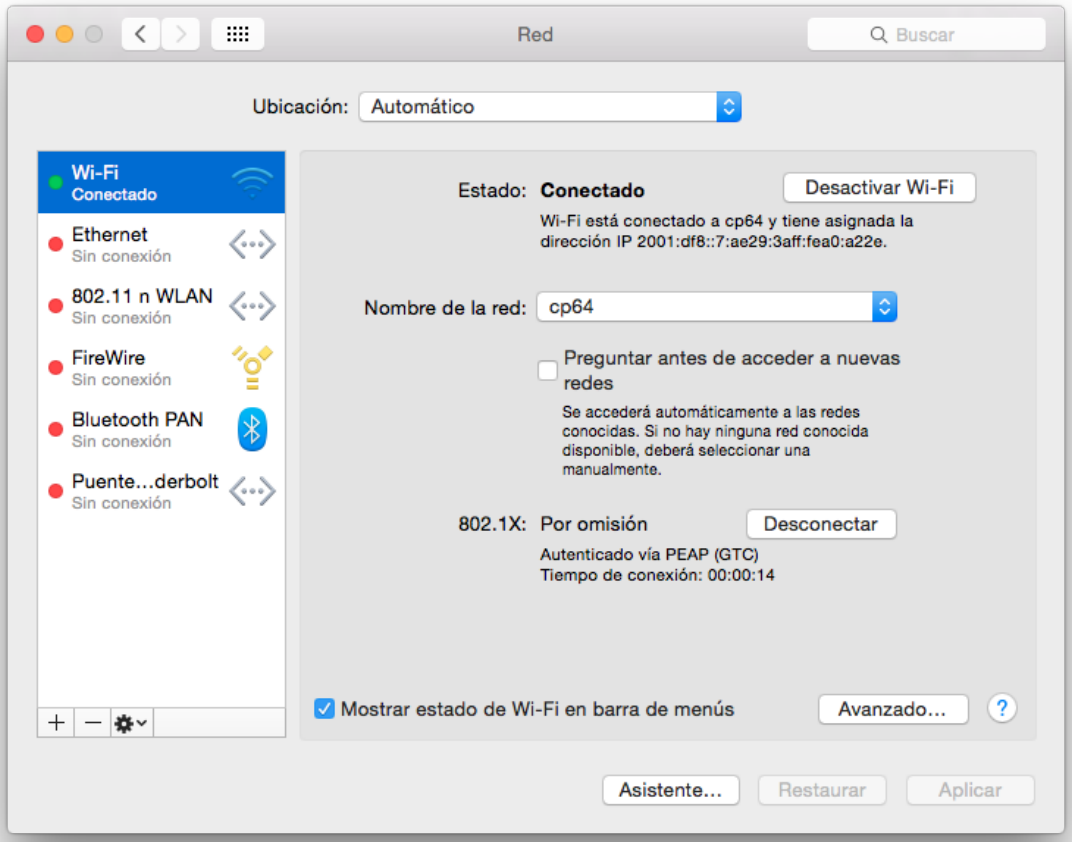

Figura 3.13: Ventana de conexión WiFi al punto de acceso cp64.

Es necesario notar que el usuario aún con dispositivo no registrado puede autenticarse con un nombre de usuario y contraseña válidos y así tener acceso a la red local, pero el acceso a la red exterior solo se logra por medio de la autenticación por MAC, la cual se realiza con un dispositivo registrado.

3.2.4. Pruebas de autenticación MAC

Para hacer la prueba de la autenticación MAC, se hace el marcado de los paquetes de acuerdo a las reglas creadas según los dispositivos de los usuarios.

Marcado y prueba de dispositivo desconocido

Haciendo `ifconfig wlan0` podemos ver la dirección MAC del dispositivo, la cual se encuentra en la sección `HWaddr`. Esta dirección se añade por DHCPv6 después de autenticarse el

usuario, sin importar si se trata de un dispositivo conocido o no.

```
usergj-6:~ josenefigamboacastaneda$ ifconfig wlan0
wlan0     Link encap: Ethernet  HWaddr 08:00:27:96:48:6b
          inet6 addr: 2001:df8:0:7:7017:a37b:511d:93b6/64 Scope:Global
          inet6 addr: fe80::a00:27ff:fe96:486b/64 Scope:Link
          inet6 addr: 2001:df8:0:7:a00:27ff:fe96:486b/64 Scope:Global
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:31 errors:0 dropped:0 overruns:0 frame:0
          TX packets:99 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4550 (4.5 KB)  TX bytes:14725 (14.7 KB)
```

Figura 3.14: Interfaz `wlan0`. Se aprecia la dirección MAC de la interfaz inalámbrica, la cual es 08:00:27:96:48:6b.

Para el caso de los dispositivos conocidos, los paquetes se marcan con 2 y se envían a su destino. Si los dispositivos son desconocidos, los paquetes se marcan con 1 y se redirigen a una pantalla con las instrucciones para su admisión en la red.

En este momento, el dispositivo del usuario es desconocido. En esta posición, si el usuario se logra autenticar en la red, esto solo es posible por medio de uno de los siguientes dos escenarios (de los cuales solo el primero es común): el usuario conocido intenta acceder desde otro dispositivo diferente al registrado por el administrador, o ha sido robado el nombre de usuario y contraseña del usuario (sea en claro, o bien, descifrándolo).

Las reglas del cortafuegos con el marcado según el tipo de dispositivo, se muestran en la tabla de marcado (mangle), por medio de la ejecución en el servidor del comando `ip6tables -L -t mangle`. El resultado de la consulta se muestra en la figura 3.15.

```
~ # ip6tables -L -t mangle
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
MARK       all      anywhere             anywhere            MARK set 0x1

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
~ # _
```

Figura 3.15: Tabla de marcado (`mangle`) de `ip6tables` en el servidor antes de agregar dispositivos conocidos. Nótese que todos los paquetes son marcados con 1 antes de su enrutamiento.

También tenemos las reglas de redirección, encargadas de enviar todos los paquetes marcados con 1 al servidor web local. En particular, se aplican en la redirección los protocolos web de la red exterior HTTP (puerto 80) y HTTPS (puerto 443). Las reglas aplicadas se muestran al ejecutar en el servidor el comando `ip6tables -L -t nat` tal como se ve en la figura 3.16.

```
~ # ip6tables -L -t nat
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
REDIRECT   tcp  --  anywhere             anywhere            mark match 0x1 tcp
 dpt:http redir ports 5280
REDIRECT   tcp  --  anywhere             anywhere            mark match 0x1 tcp
 dpt:https redir ports 5280

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
~ #
```

Figura 3.16: Tabla de traducción (nat) de `ip6tables` en el servidor, la cual contiene las reglas de redirección de paquetes marcados con 1 hacia el servidor web local (puerto 5280).

Estas reglas son aplicadas del lado del servidor. Del lado del usuario, con este panorama, cuando un usuario intenta acceder a la red, su dispositivo es desconocido y aún cuando se lograse autenticar a la red por medio de RADIUS, al intentar acceder a una página web exterior es redirigido al servidor local, debido a las reglas del cortafuegos.

El mensaje con las instrucciones para registrar un usuario se muestra en la figura 3.17. Este mensaje se encuentra en `index.html` dentro del directorio raíz para los datos de `mini_httpd`.

Figura 3.17: Mensaje del servidor local cuando se intenta acceder con un dispositivo no conocido, los paquetes con las consultas externas del cliente son marcados con 1 y redirigidos al puerto 5280, de modo que el cliente no tiene otra opción más que registrar su dispositivo con el administrador para hacer uso de la red.

Marcado y prueba de dispositivo conocido

Para ingresar un nuevo usuario a la red, se comunica con el administrador y si cumple las condiciones que impone éste, el administrador hace uso de la aplicación `registraUsuario` para agregar el usuario a la red.

Para hacer uso de la aplicación, es suficiente ejecutar ./`registraUsuario` < MAC_cliente >< Usuario >< Password_en_claro > en el directorio donde se encuentra la aplicación. Un ejemplo lo tenemos en la figura 3.18.

```
~ # ./addusu 08:00:27:96:48:6b yo misecretoindividual
```

Figura 3.18: Aplicación `registraUsuario`, el administrador agrega el usuario (yo), la contraseña (misecretoindividual) y la MAC del dispositivo (08:00:27:96:48:6b).

La aplicación añade automáticamente el usuario y digesto de la contraseña en el archivo usuario, dentro del directorio de freeradius, como se ve en la figura 3.19.

```
~ # cat usuarios
yo Crypt-Password := "eAY8yjEhGsX2Y"
~ # _
```

Figura 3.19: Nombre de usuario y digesto de la contraseña introducidos por el administrador a través de la aplicación `registraUsuario` para la autenticación RADIUS.

Además, `registraUsuario` crea una regla en el script `mark.sh`, para realizar el marcado 2 para la MAC del dispositivo ingresado, con el objetivo de permitir su acceso a internet. Esto se muestra en la figura 3.20.

```
~ # cat mark.sh
ip6tables -A PREROUTING -t mangle -m mac --mac-source 08:00:27:96:48:6b -j MARK
--set-mark 2

~ # _
```

Figura 3.20: Regla añadida al script `mark.sh`, para marcar los paquetes del dispositivo agregado con 2 y evitar su redirección al servidor local.

Así al ejecutarse el script, actualizamos las reglas de marcado de su tabla correspondiente en ip6tables, agregando el marcado 2 a los paquetes provenientes de la MAC ingresada, para evitar la redirección del dispositivo a la red local.

El resultado muestra la regla añadida a la tabla de marcado tal y como se ve en la figura 3.21.

```
~ # ip6tables -L -t mangle
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
MARK       all      anywhere             anywhere             MARK set 0x1
MARK       all      anywhere             anywhere             MAC 08:00:27:96:48
:6B MARK set 0x2

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
~ # _
```

Figura 3.21: Tabla de marcado con la regla añadida para marcar los paquetes provenientes de la MAC 08:00:27:96:48:6B con 2.

Como las reglas de la tabla de traducción (nat) son las mismas (figura 3.16), no hay ninguna regla de redirección para los paquetes marcados con 2. Por lo tanto, llegan a su destino de la red exterior sin problemas.

A lo anterior le llamamos autenticación por MAC, debido a que solamente los dispositivos cuya dirección MAC ha sido registrada por el administrador a través de la aplicación `registraUsuario` pueden utilizar la red exterior.

La figura 3.22 muestra el resultado de la consulta en el cliente autenticado con el dispositivo registrado por el administrador.

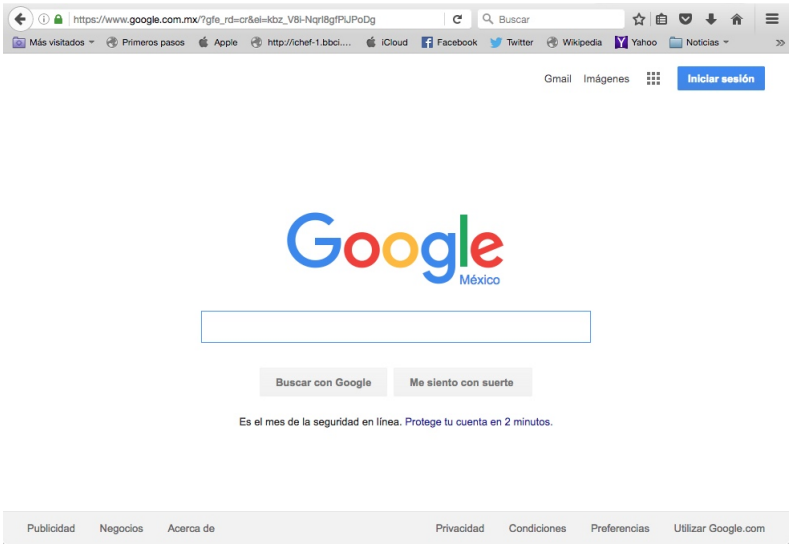

Figura 3.22: Resultado de un acceso exitoso a una página web después de registrar al usuario. Los paquetes del cliente son marcados con 2 y no aplica la redirección al servidor web local.

3.3. Portal cautivo con Wifidog

Otra forma de autenticar los usuarios en las redes se puede hacer por medio de un portal cautivo. Un portal cautivo consiste en un portal web que forza a los usuarios a autenticarse antes de poder acceder a la red. Así como en la autenticación por MAC, hace uso de reglas en el cortafuegos que redireccionan al usuario no autenticado al sitio web local. En el servidor web, el usuario llena una forma con su nombre de usuario y contraseña que se la envía al servidor de autenticación. Si la autenticación del usuario tiene éxito, los paquetes de red procedentes de su computadora se marcan de forma distinta en el cortafuegos para permitirle el acceso a los servicios de red.

La ventaja de usar portales cautivos es permitir al administrador establecer otras políticas, tales como un período de vigencia en el uso de la red o establecer niveles de autorización a la red, así como personalizar el acceso a la red de los usuarios, tanto en la interfaz como en su uso.

Actualmente (2016) existen varias opciones de software libre que funcionan como portales cautivos, tales como nocat (sitio descontinuado), wifidog [37], chillispot [34], pepperspot [51] y coovachilli [21]. Siendo los primeros dos manejados por reglas de marcado en el cortafuegos. Los últimos son manejados por medio reglas con el uso de un servidor de autenticación RADIUS.

Desafortunadamente, de los portales cautivos anteriores, sólo pepperspot en la actualidad tiene compatibilidad con el protocolo IPv6. Sin embargo, las cabeceras de las versiones recientes del núcleo de Linux tienen problemas de compatibilidad con la última versión de pepperspot (0.4), proyecto discontinuado desde 2011 y con un problema sin parche desde junio de 2015.

En el caso de IPv4, el portal cautivo wifidog, con licencia GPL y escrito en C, ha mostrado ser muy estable, debido a su facilidad de tratar diferentes clases de clientes a través del uso de diferentes clases de marcas para los paquetes de éstos (prueba, conocido, servidor de autenticación caído, cliente bloqueado y sin marca). Esto facilita la tarea de auditar a los usuarios, así como el establecimiento de diferentes cadenas de paquetes, según su origen (salida o entrada), hacia donde se dirige el paquete (hacia internet, al enrutador, al servidor de autenticación o globales), por el tipo de paquete (conocido, desconocido, para validación, bloqueado, de confianza) e incluso si el servidor de autenticación ha caído. Esta ventaja es usada para las tablas de marcado, filtrado y/o traducción de paquetes y depende enteramente de la MAC de los clientes.

3.3.1. Funcionamiento básico de Wifidog

Wifidog funciona bajo el esquema de autenticación con servidor de autenticación, el cual no es necesario que sea RADIUS. El proceso de autenticación entre un cliente que quiere hacer uso de la red y el servidor se muestra en la figura 3.23 [55] y es el siguiente:

- El cliente previamente conectado a la red privada (procedimiento independiente de Wifidog) realiza su solicitud inicial.

- Las reglas del cortafuegos del punto de acceso redirigen la solicitud a un puerto local de éste, obteniendo así una respuesta para el cliente de redirección HTTP que contiene el identificador del punto de acceso.
- El cliente envía su solicitud al servidor de autenticación.
- El servidor de autenticación responde al usuario con una pagina web de ingreso.
- El cliente proporciona su nombre y contraseña.
- Para obtener una autenticación satisfactoria, el cliente se redirige por medio de HTTP hacia el servidor web del punto de acceso, redirección que incluye una prueba (*token*).
- El cliente se conecta al punto de acceso y envía su prueba.
- El punto de acceso solicita la validación de la prueba hacia el servidor de autenticación.
- El servidor de autenticación confirma la prueba.
- El punto de acceso envía una redirección al cliente para obtener una página satisfactoria, proveniente del servidor de autenticación.
- El servidor de autenticación notifica al cliente que la solicitud fue exitosa.

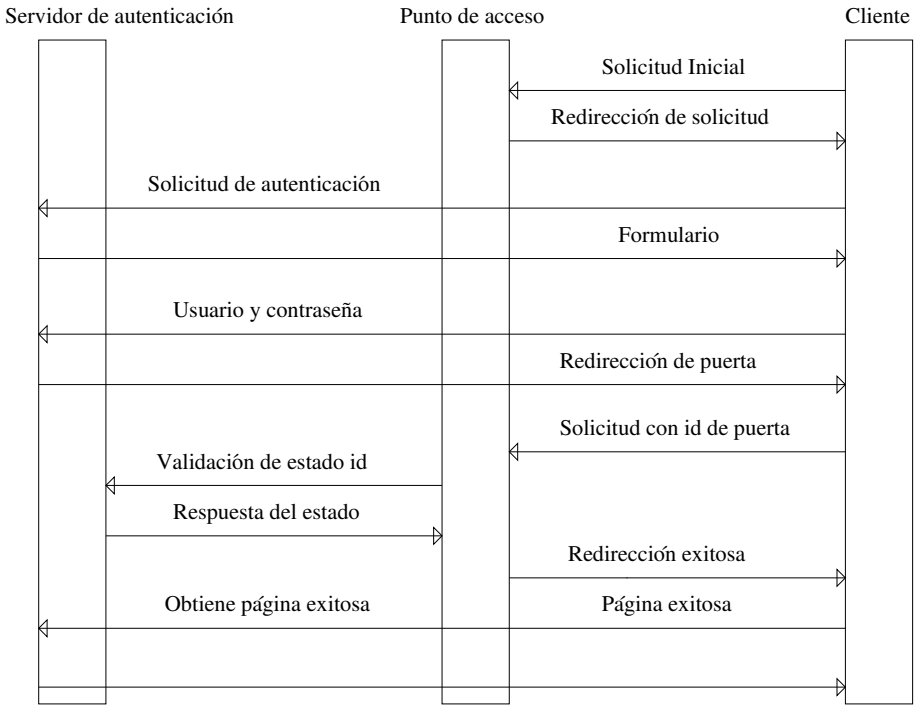

Figura 3.23: Funcionamiento de wifidog. Se observan tres componentes importantes: el cliente, el punto de acceso, encargada de transferir las solicitudes y el servidor de autenticación, que acepta o rechaza solicitudes, de acuerdo a la información contenida en éste [37].

3.3.2. Ventajas de un portal cautivo sobre un punto de acceso

Por medio del uso de los portales cautivos se añaden algunas ventajas a la autenticación adquirida por medio de un punto de acceso seguro:

- Registro de usuarios: Por medio del portal web, se puede incluir en la misma página un sistema de registro de usuarios, el cual permita a los propios usuarios crear su propio nombre y contraseña sin requerir del registro por parte de un administrador. El procedimiento de registro hace uso del correo electrónico del usuario para validar su identidad y se le concede un período de gracia (15 minutos) para validar su correo.

- Auditoría: Permite mostrar estadísticas de los usuarios y del uso de la red, caso imposible para el punto de acceso seguro. Algunas de estas estadísticas son: los consumidores que hacen mayor uso del ancho de banda, usuarios más frecuentes, usuarios con mayor movilidad, informe de usuarios que hacen uso de la red, informes individuales para los usuarios, nodos más populares, estado de la red, estado del nodo, informe de registro de usuarios, informe de registro interno, de errores, etc.

- Personalización: Es posible adaptar el portal cautivo a las necesidades específicas de una empresa, aplicar autenticación de usuarios multifactor, establecer condiciones de registro, definir una página web a la medida, cambiar el idioma del portal, etc.

3.3.3. Desventajas del uso de portales cautivos

Debido al uso de la dirección MAC para autenticación, Wifidog es vulnerable a ataques de suplantación de dirección MAC, ya que no involucra un mecanismo de autenticación del estándar 802.1X. Sin embargo, al combinarse con otros métodos de autenticación proporciona un medio de autenticación robusta, además de las ventajas mostradas en las sección anterior.

Capítulo 4

Conclusiones

De acuerdo a los resultados adquiridos, el punto de acceso solo permite el acceso a la red a los usuarios cuyo nombre y contraseña son correctos utilizando el mecanismo EAP-GTC, con la mejora de que aún cuando se permite el acceso a la red a los usuarios autenticados correctamente, solamente podrán hacer uso de ella si la dirección MAC de su dispositivo ha sido añadida a las reglas del cortafuegos, evitando la redirección de las consultas web al servidor local. De modo que para hacer uso de la red externa es necesario autenticarse y tener el dispositivo registrado, tarea que realiza el administrador.

Se diseñó y construyó un punto de acceso para clientes inalámbricos en IPv6. El punto de acceso usa el mecanismo de traducción de direcciones de IPv4 a IPV6, el servicio de nombres DNS64 y la asignación de direcciones a los clientes con DHCPv6. Para la autenticación de los clientes se usa el servidor RADIUS con el mecanismo EAP-GTC.

El uso del protocolo IPv6 proporciona a los clientes varios beneficios que el protocolo IPv4 no tiene o limita, tales com permitir a los clientes tener varios anfitriones cómodamente gracias a la amplitud de direcciones que el protocolo permite; transferir los paquetes como multidifusión en vez de difusión a todos, evitando así que los paquetes sean recibidos por parte de posibles intrusos; la existencia de un enlace local, para comunicarse entre anfitriones en forma local y la reducción de campos en el encabezado IPv6, optimizando el rendimiento en la transferencia de paquetes.

La traducción de direcciones TDR64 es un mecanismo de transición entre redes IPv6 e IPv4 que permite a los usuarios hacer consultas en la red actual IPv4 desde la red interna en IPv6, permitiendo así la navegación nativa y el disfrute de los beneficios de las redes IPv6. Además, con la traducción de nombres DNS64, los registros AAAA conservan el mismo nombre que su homónimo A, para que la traducción sea transparente para los usuarios.

El diseño del punto de acceso permitió analizar los protocolos que permiten comunicarse el usuario con el punto de acceso en forma segura (802.1X, con el mecanismo EAP-GTC), el diseño del servidor de autenticación (RADIUS) con contraseñas cifradas, el servidor web, las reglas del cortafuegos para el filtrado y marcado de paquetes, las interfaces de red físicas: wlan0 para la red inalámbrica y eth0 para la red exterior, la asignación de direcciones dinámicas (DHCPv6), TDR64, DNS64 y la agregación de usuarios para la autenticación

MAC, para llevar a cabo la construcción de un punto de acceso seguro.

Al haber estudiado los protocolos existentes en el IEEE 802.11i y 802.1X, hemos visto que el mecanismo de seguridad EAP-GTC ofrece un buen nivel de seguridad, debido al uso de un servidor de autenticación que permite la verificación de identidad del cliente. Además, se protege la contraseña del servidor de autenticación por medio del almacenado del digesto de la contraseña en el archivo de configuración del servidor RADIUS.

La construcción del punto de acceso seguro en forma física cumplió con todas las expectativas de funcionamiento en materia de seguridad, accesibilidad y disponibilidad, permitiendo a los usuarios en el 100 % de los casos con autenticación satisfactoria tener acceso a la red, cifra similar en el rechazo debido a una autenticación no satisfactoria. Esto se logra gracias al uso de la autenticación multifactor, en donde los mecanismos de autenticación 802.1X y por dirección MAC funcionan en forma correcta, así como la asignación dinámica de direcciones, la traducción de direcciones y de nombres.

A pesar de que las partes individuales del punto de acceso y el uso de protocolos ya existe, la implementación del punto de acceso TDR64 con autenticación multifactor es nuevo.

Finalmente, el objetivo general de construir un punto de acceso seguro en una red inalámbrica en IPv6 para hacer uso de los recursos actuales de internet se cumple y su implementación es efectiva para permitir el acceso seguro a los usuarios de la red interna. El punto de acceso construido podría ser parte de la migración mundial que se está haciendo al protocolo IPv6.

4.1. Trabajo a futuro

Como proyectos a futuro que surgen del trabajo de esta tesis se encuentran los siguientes:

- La implementación del punto de acceso inalámbrico seguro en una sola tarjeta, por ejemplo Raspberry Pi 2, ya que en la tesis se desarrolló para una arquitectura x86_64. El problema actual para llevarlo a cabo es que el módulo nat64 no soporta las macros de las bibliotecas de Netfilter para las nuevas versiones del núcleo de Linux. Para compilar nat64 se necesitan versiones del núcleo de Linux anteriores a 3.13.x.

- La implementación de Wifidog en IPv6. Con eso podría existir un portal cautivo estable en IPv6 que posea los beneficios de seguridad con IPv6 en conjunto con la auditoría y registro del usuario autónomo. Es posible implementar Wifidog en IPv6 que use también el mecanismo TDR64, en analogía con el punto de acceso construido en esta tesis.

Apéndice A

Punto de Acceso Seguro en una SBC

Una Computadora en una Sola Tarjeta (SBC) es una computadora simple, de costo bajo y constituida con los componentes esenciales para el funcionamiento de una computadora de propósito general: un microprocesador, memoria Memoria de Acceso Aleatorio (RAM), controladores empotrados para video, red, disco duro, etc.

Un ejemplo de SBC lo tenemos en la tarjeta PCM-5823. Esta es una tarjeta con arquitectura i586, y tiene las siguientes especificaciones [4]:

- Procesador: NS GC1-300/(1.8, 2.0, 2.2)V empotrado.
- Sistema Básico de Entrada y Salida (BIOS): Concede 256 Kb de Memoria Flash.
- Circuito Integrado Auxiliar: NS CX5530.
- Interfaz Dispositivo Electrónico Integrado (IDE) mejorada: Soporta hasta dos dispositivos IDE Mejorado (EIDE), autodetectables por BIOS, con el modo de transferencia 3 o 4 de Entradas y Salidas Programadas (PIO), Modo Ultra Acceso Directo a Memoria (DMA)33 Accesorio de Tecnología Avanzada (ATA)-4 con 33 Mb/segundo.
- Puerto serial: RS-232/422/485.
- Puerto paralelo: Uno con soporte al modo Puerto Paralelo Estándar (SPP)/Puerto Paralelo Mejorado (EPP)/Puerto de Capacidad Mejorado (ECP).
- Puerto infrarrojo: Compartido con Puerto de Comunicación versión 2 (COM2), con taza de transferencia de hasta 115 kbps.
- Conector de teclado y ratón: Mini-*Deutsches Institut für Normung* (DIN) con soporte a los teclados estándar PC/AT y ratón PS/2.
- Interfaz USB: Dos con compilación 1.1.
- Circuito Integrado Auxiliar de la interfaz dual de ethernet: Realtek RTL 8139.

- Interfaz ethernet: Interconexión de Componentes Periféricos (PCI) 10/100 Mbps, compatible con estándar 802.3.
- Conexiones ethernet: Dos con RJ-45.
- Disco de estado sólido: Socket con soporte para una Memoria *Compact Flash*.

Los componentes se encuentran ubicados en la tarjeta como se muestra en la figura A.1.

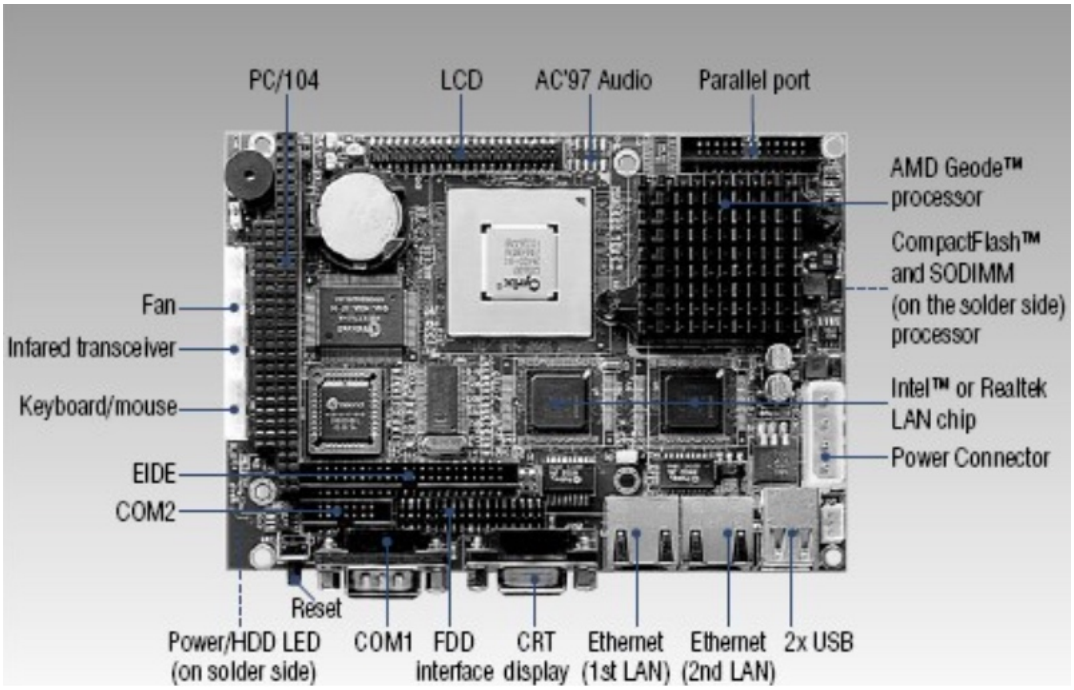

Figura A.1: Ubicación de los componentes de la tarjeta SBC PCM 5823 [31].

A.1. Construcción del sistema operativo para PCM-5823

Debido a la incompatibilidad entre las arquitecturas x86_64 y i586, es necesario hacer una compilación cruzada para transferir el sistema operativo hacia la tarjeta i586. Esta compilación se realiza por medio de buildroot [30].

Esta es una herramienta que genera un sistema operativo Linux, tanto el núcleo, los módulos del núcleo y el sistema de archivos, usando la compilación cruzada. Se utilizó la versión de Agosto de 2016.

A.1.1. Estructura del directorio de buildroot

Buildroot realiza la compilación de los paquetes por medio de los archivos de configuración de estos para crear los archivos binarios en la arquitectura deseada. La configuración de los paquetes se encuentra en el directorio `package`, en donde los paquetes hacen uso de los archivos: `nombre_paquete.mk` que incluye la configuración de compilación y `Config.in` para mostrar el paquete en el menú de paquetes de buildroot.

La estructura de los directorios de los paquetes es la siguiente: El directorio `dl` contiene las descargas de los paquetes en formato comprimido. El directorio de construcción `output/build` contiene los archivos necesarios para el proceso de compilación, provenientes de la descompresión de éstos. El directorio `output/host` contiene las bibliotecas y cabeceras necesarias para la compilación cruzada, dentro de la arquitectura del anfitrión. El directorio `output/target` contiene el sistema de archivos y los binarios en la arquitectura destino, que en particular es i586. El directorio `output/images` contiene las imágenes de disco, de grub, del núcleo, del sistema de archivos, etc; provenientes de la arquitectura destino.

La configuración de la cadena de herramientas (*toolchain*) por defecto para buildroot, contiene las características y parámetros para la construcción. Esta se encuentra en el directorio `toolchain` y contiene las herramientas para compilar el núcleo y sus cabeceras; la biblioteca `uClibc`, que es una biblioteca de funciones generales de peso bajo; binutils, para construir los archivos binarios y las opciones del compilador gcc.

También existe la configuración predefinida para diferentes tarjetas de fabricantes comunes: intel, qemu, raspberrypi, atmel, technologic, etc. Esta configuración se encuentra en el directorio `board`. La forma de aplicar ésta configuración por defecto es por medio de `make nombre_tarjeta_defconfig` antes de compilar el sistema.

A.2. Proceso de compilación

El proceso de compilación cruzada se realiza a través de reglas establecidas en los archivos de configuración. La configuración de los paquetes se sugiere realizarse en modo gráfico utilizando el comando `make menuconfig`, que permite tener un ambiente gráfico para elegir las opciones, aunque también se puede hacer en modo de línea de comandos con `make config`.

Para crear un Linux empotrado se requiere compilar el núcleo, la biblioteca uClibc, la biblioteca busybox y además los paquetes complementarios junto con sus bibliotecas para el uso específico de la tarjeta.

A.2.1. Opciones generales de compilación

Al ejecutar el comando `make menuconfig` se muestran las opciones generales de compilación para el sistema empotrado que incluiremos en la SBC, mostrado en la figura A.2.

Para el caso de la construcción del punto de acceso en la tarjeta PCM-5823, elegimos las siguientes opciones de configuración:

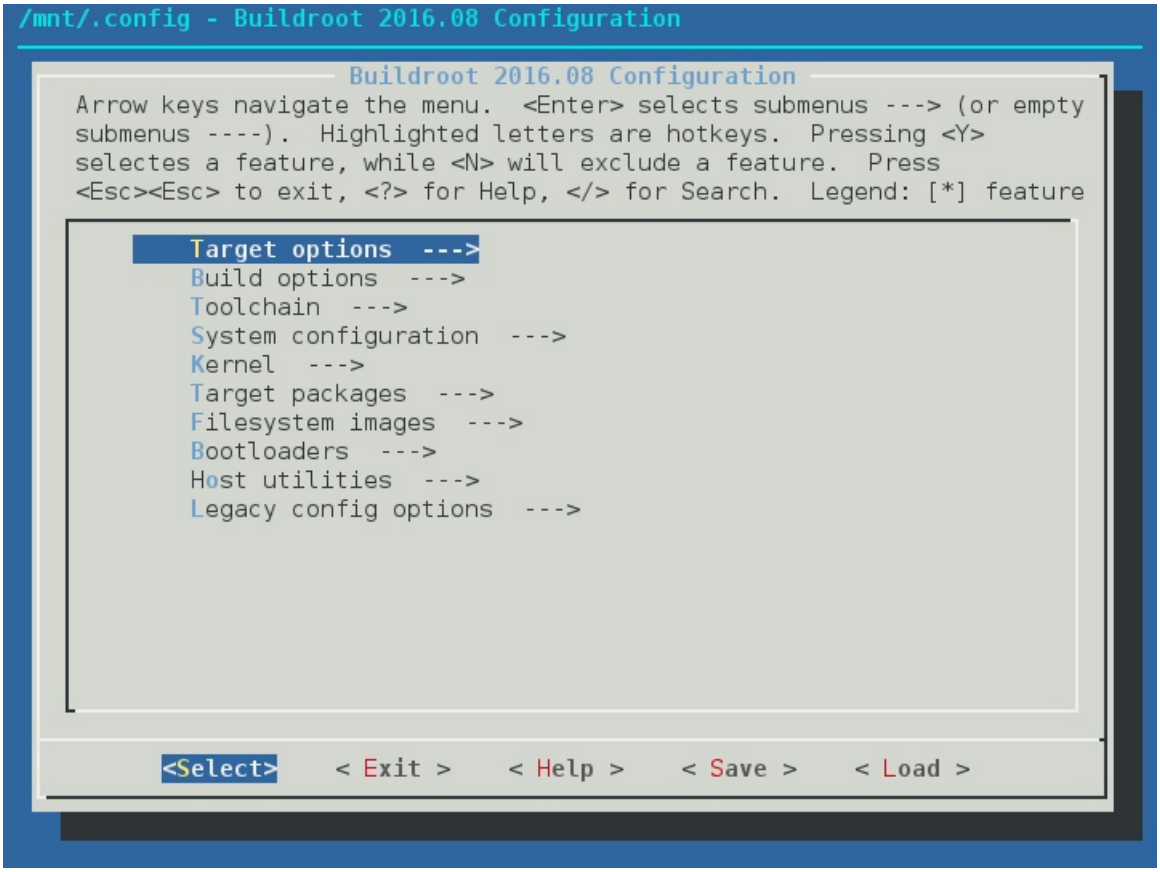

Figura A.2: Menú principal de buildroot.

- Opciones de tarjeta: Especificar la arquitectura de tarjeta i386, con la variante i586.
- Opciones de construcción: Bibliotecas compartidas.
- Cadena de herramientas: Especificar tipo de cadena de herramientas de buildroot. Cabeceras del núcleo especificadas manualmente, con la versión 3.11.10. La biblioteca C es uClibc. Binutils versión 2.25.1 y gcc versión 4.9.x.
- Núcleo: Versión 3.11.10, en formato bzImage y compresión gzip.
- Paquetes destino:
  - Manejadores de hardware: rfkill.
  - Bibliotecas: openssl (con binario), libcurl (con binario), libnl, libpcap y libevent.
  - Aplicaciones de red: dnsmasq, hostapd, iproute2 e iptables.

Estos paquetes no son necesarios para la construcción de un sistema operativo mínimo. No obstante, son necesarios para el punto de acceso seguro.

A.2.2. Adición de paquetes externos

Para añadir un paquete externo, es decir, no incluido en las opciones de configuración de buildroot, es necesario añadir la configuración del paquete en el directorio `package`, la cual consta de los archivos `Config.in` y `nombre_paquete.mk`. Los paquetes añadidos por esta forma son mini_httpd, unbound, freeradius-server, insertarUsuario y el módulo nat64.

La configuración del archivo `Config.in` en cada caso es sencilla, solo añadimos el nombre del paquete, la activación de la opción y opcionalmente una ayuda breve. Por ejemplo en la aplicación `insertarUsuario`, contiene el archivo de configuración del listado A.1.

```
config BR_PACKAGE_ADDUSU
  bool "addusu"
  help
    Agregar usuario para autenticarse con WPA2-EAP y MAC

    by Nefi Gamboa
```

Listado A.1: Archivo de configuración `Config.in` para el paquete insertarUsuario.

La compilación de los paquetes externos se realiza de acuerdo a las opciones mostradas en `nombre_paquete.mk`. Por ejemplo, el archivo `insertarUsuario.mk` del paquete insertarUsuario, tiene el contenido del listado A.2.

```
# Versión, origen, método y ubicación
ADDUSU_VERSION = 0.1
ADDUSU_SOURCE = addusu.tar.gz
ADDUSU_SITE_METHOD = local
ADDUSU_SITE = /fosil/buildroot-2016.08/output/build/addusu
# Incluimos las variables de compilación destino
ADDUSU_MAKE_OPTS = CC="$(TARGET_CC)" LD="$(TARGET_LD)" LDFLAGS="$(TARGET
_LDFLAGS)"

# Opciones de construcción
define ADDUSU_BUILD_CMDS
  $(ADDUSU_MAKE_ENV) $(MAKE) $(ADDUSU_MAKE_OPTS) -C $(@D)
endef

# Opciones de instalación
define ADDUSU_INSTALL_TARGET_CMDS
  $(ADDUSU_MAKE_ENV) $(MAKE) -C $(@D) DESTDIR=$(TARGET_DIR)
endef

$(eval $(generic-package))
```

Listado A.2: Archivo de compilación para el paquete externo `insertarUsuario`.

De forma similar creamos éstos dos archivos para el resto de paquetes. Es necesario tener el directorio descomprimido para cada caso si se quiere utilizar la opción de sitio local.

A.2.3. Configuración del núcleo

Las opciones de configuración del núcleo se eligen al escribir en la línea de comandos `make linux-menuconfig`, ventana principal mostrada en la figura A.3.

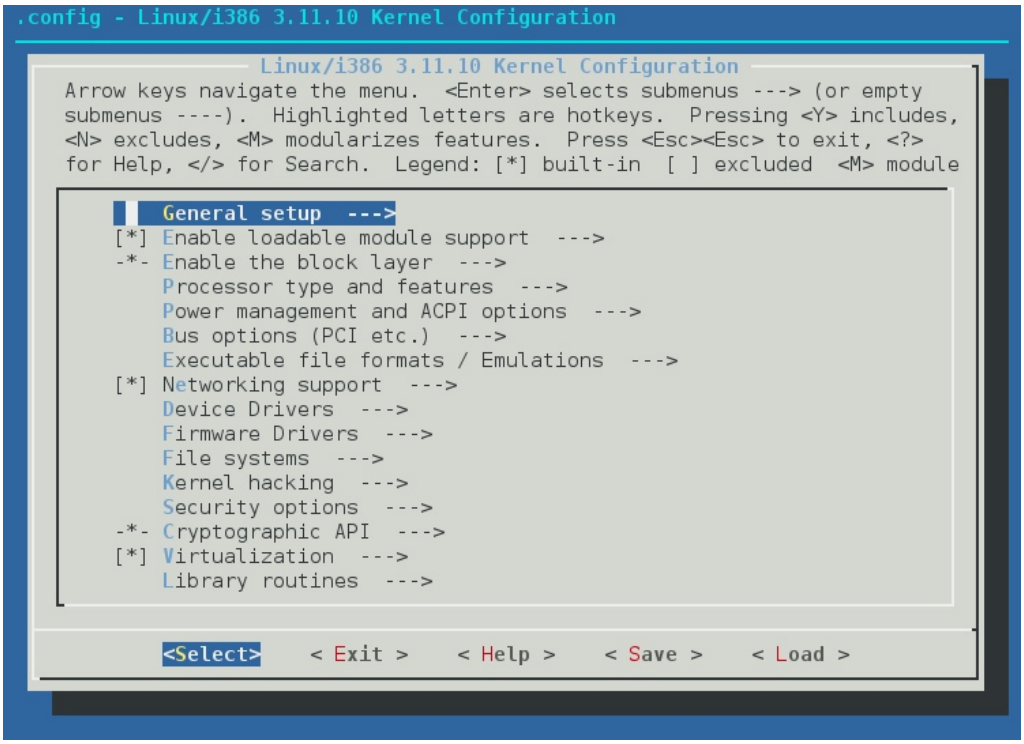

Figura A.3: Menú principal para la compilación del núcleo de buildroot.

Las opciones necesarias del núcleo para compilarse con el objetivo de crear el punto de acceso en IPv6 son:

- Opciones del canal: Soporte para PCI.
- Soporte de red:
  - Opciones de red: Activar protocolo IPv6, paquete de filtrado Netfilter (Incluyendo los módulos de IPv6: IP6tables, NAT y filtrado. También incluir el marcado y redirección de paquetes).
  - Red inalámbrica: Activar los módulos cfg80211, nl80211 y mac80211.
- Controladores de dispositivos: Necesitamos los controladores para el hardware de la SBC PCM-5823, los cuáles son los siguientes:

- Soporte ATA: Incluir el soporte para el circuito integrado auxiliar Cyrix/National CS5530 MediaGX, así como el soporte para el circuito integrado auxiliar IDE genérico con PCI.
- Controladores ATA serial y paralelo: Soporte para SFF ATA, con soporte BMDMA con PATA CS5530. También el soporte para ATA genérico.
- Soporte para dispositivo de red: Soporte para TUN/TAP universal. Soporte para controladores ethernet: Dispositivos Realtek, con soporte para el adaptador rápido de PCI Realtek-8139, así como Realtek 8169. También el soporte para la tarjeta inalámbrica Atheros, con soporte a 802.11n USB de la comunidad de Linux AR9170, el cuál activa el controlador carl9170 para la tarjeta inalámbrica Netgear WN111v2.
- Soporte USB: Controlador OHCI HCD, para USB 1.1, con soporte a PCI. También Convertidor USB serial.

- API criptográfica: Incluir ARC4 y DES para el algoritmo Crypt.

A.2.4. Configuración de uClibc y Busybox

Para la configuración de la biblioteca de funciones generales uClibc, ejecutamos el comando `make uclibc-menuconfig`, en donde se muestra el menú de opciones de la figura A.4.

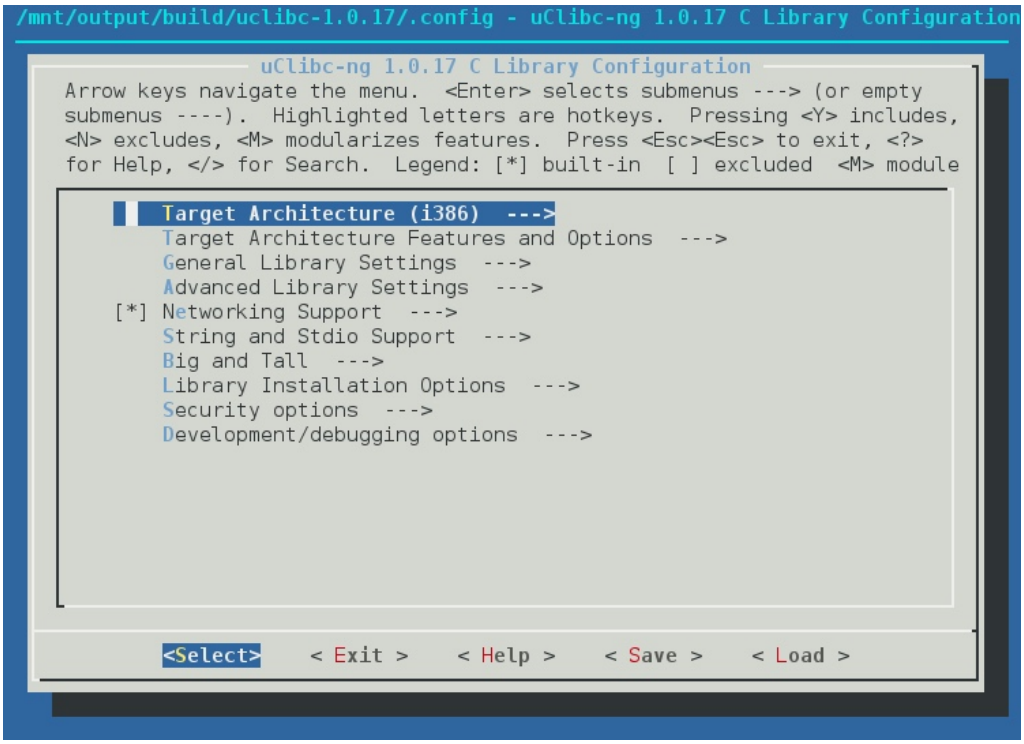

Figura A.4: Menú principal para la compilación de la biblioteca C ligera uClibc.

Para la biblioteca uClibc necesitamos activar las siguientes opciones de configuración:

- Soporte de red: Incluir el soporte a IPv6.
- Soporte de cadenas: Incluir datos de localidad.

Busybox es un conjunto de aplicaciones útiles de Linux [6], para la configuración de las opciones de compilación ejecutamos el comando `make busybox-menuconfig`, el cuál muestra el menu de la figura A.5.

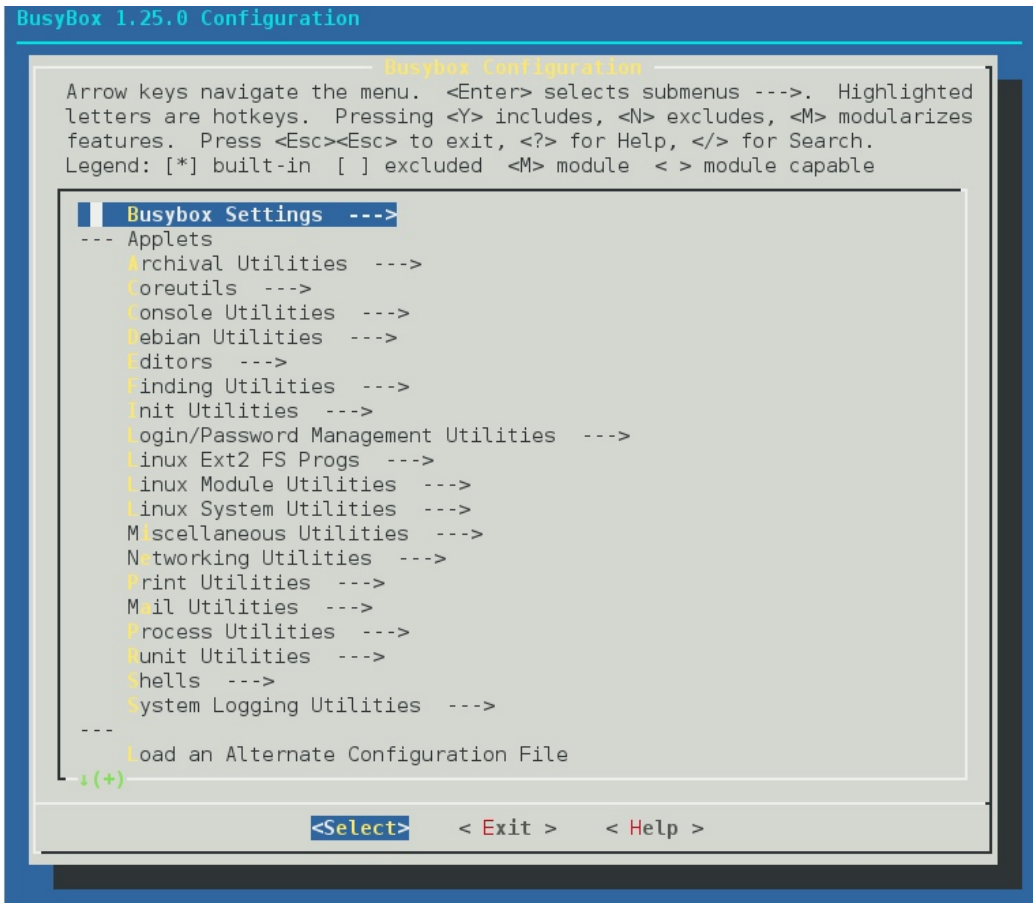

Figura A.5: Menú principal para la compilación de busybox.

Las opciones que faltan activar en busybox son las siguientes dentro de utilidades de red: ping6, soporte a IPv6 y preferencia por las direcciones IPv4 en consultas.

A.2.5. Compilación general

Después de configurar el núcleo, uClibc y busybox, creamos el sistema operativo al compilar con `make`. El resultado es el sistema de archivos del punto de acceso, ubicado en el directorio

`output/target`. Así como la generación del núcleo con los módulos construidos en el directorio `output/images`.

La SBC PCM-5823 posee un socket para introducir una *Compact Flash*, la cual funciona como disco duro de la tarjeta. Para transferir el sistema operativo a la tarjeta, es necesario hacer una partición que tenga al menos 64 Mb de espacio en disco y darle formato ext2 (puede ser con la utilidad fdisk o cfdisk) y hacer que sea arrancable.

Para crear el sistema de archivos, es suficiente copiar recursivamente el contenido de `output/target` en la tarjeta, así como copiar el núcleo `output/images/bzImage` en el directorio `/boot` de la CF. Al final debemos configurar nuestro punto de acceso de acuerdo a la información mostrada en el capítulo 3 de esta tesis.

En forma adicional necesitamos copiar el firmware para la tarjeta inalámbrica cuyo controlador es carl9170. Este se descarga desde https://wireless.wiki.kernel.org/en/users/drivers/carl9170 y se copia en el directorio `/lib/firmware` del punto de acceso.

A.2.6. Instalación de grub

Para iniciar el sistema, es necesario un archivo de arranque tal y como lo hace grub versión 2 [45]. El procedimiento para generar el archivo de arranque se hace por el comando `grub2-install /dev/sdb --boot-directory=/mnt/boot`. Aquí el primer argumento es el dispositivo CF y el segundo es el directorio `/boot` del punto de montaje para el dispositivo.

Debido al almacenamiento de contraseñas constante, es necesario hacer la carga desde el disco duro y no desde memoria. Para ello el archivo `/boot/grub2/grub.cfg` debe contener la información del listado A.3.

```
1 set default=0
2 set timeout=10
3 menuentry "NAT64 OS by Nefi Gamboa" {
4   echo 'Booting, wait a minute...'
5   set root='hd0,msdos1'
6   linux /boot/bzImage rw root=/dev/hdc1 rootwait selinux=0 \
7   acpi_osi=Linux quiet splash
8 }
```

Listado A.3: Contenido de `grub.cfg` para el inicio del sistema operativo desde el disco duro.

A.3. Evaluación del punto de acceso en la SBC PCM-5823

A pesar del éxito obtenido en la creación del punto de acceso en IPv6 para la arquitectura x86_64, en la SBC indicada previamente con arquitectura i586 existe un problema de estabilidad con el controlador `carl9170` [48]. Este problema es ocasionado debido al controlador

FUSB200, el cual consta de cuatro puntos finales (EP) para red (tres EP de salida y un EP de entrada) y dos EP para comandos WMI (uno de entrada y otro de salida).

Mientras que los cambios son aplicados en forma correcta para USB 3.0 (con un tiempo aproximado de 0.2 ms.), para USB 1.1 el controlador carl9170 se interrumpe y reinicia antes de aplicar los cambios, debido a que con este controlador, USB no puede trabajar con un tamaño de FIFO muy pequeño. Esto sucede porque EP3 y EP4 pueden acceder directamente y en particular EP4 tiene un mapeo FIFO de 64 bytes, pero se necesitan 512 bytes para la transferencia, pérdida ocasionada por la velocidad que en USB 2.0 es de 2 a 4 ms. y mucho menor para USB 1.1.

Una probable solución sería el uso de DMA (no usada por EP3 y EP4) y usar otros EPs (EP5, EP6 o EP1), sin embargo después de la carga del firmware de carl9170, solo trabajan EP3 y EP4, que irónicamente no usan DMA.

Sin utilizar la tarjeta de red inalámbrica, el funcionamiento en la SBC es exitoso dentro de la red cableada, proporcionando a los clientes los servicios de DHCPv6, DNS64, TDR64 y del cortafuegos, con la limitante de no hacer uso del servidor de autenticación (aún cuando si funciona) ni de hostapd.

Bibliografía

[1] Aboba, B; Blunk, L; et. al. *Extensible Authentication Protocol (EAP), RFC 3748*, Proposed Standard, Network Working Group, Standards Track, June 2004.

[2] Aboba, B; Simon, D. *PPP EAP TLS Authentication Protocol, RFC 2716*, Network Working Group, Experimental, October 1999.

[3] ACME Labs. *mini_httpd - small HTTP server*, https://acme.com/software/mini_httpd/, disponible al 10 de Octubre de 2016.

[4] Advantech. *PCM-5823, NS Geodude Single Board Computer with CPU SVGA/LCD, Dual Ethernet Interface Manual*, Part No. 2006582302, 3rd Edition, Printed in Taiwan, June 2001.

[5] Alan DeKok (Project Leader), Arran Cudbard-Bell, Phil Mayers and Matthew Newton. *The FreeRADIUS Technical Guide* Network RADIUS SARL. 2014. p.2, 9-10.

[6] Andersen, Erik. *BusyBox: The Swiss Army Knife of Embedded Linux*, 1999-2008, https://busybox.net/

[7] Angela, Ajah. *Evaluation of Enhanced Security Solutions in 802.11-Based Networks*, International Journal of Network Security & Its Applications (IJNSA), Vol. 6, No. 4, July 2014, p. 29-42.

[8] Arends; Austein; Larson; Massey; Rose. *DNS Security Introduction and Requirements, RFC 4033*, The Internet Society, Network Working Group, Standards Track, March 2005.

[9] Arends; Austein; Larson; Massey; Rose. *Resource Records for the DNS Security Extensions, RFC 4034*, The Internet Society, Network Working Group, Standards Track, March 2005.

[10] Ariganello, Ernesto y Barrientos S, Enrique. *Redes Cisco, guía de estudio para la certificación CCNP*, Alfaomega Ra-Ma, Primera Edición, Septiembre 2011, p.197−−208.

[11] Atkinson, R. *Security Architecture for the Internet Protocol, RFC 1825*, Naval Research Laboratory, Network Working Group, August 1985.

[12] Bao; Huitema; Bagnulo; Boucadair; Li. *IPv6 Addressing of IPv4/IPv6 Translators, RFC 6052*, Internet Engineering Task Force (IETF), Standards Track, ISSN: 2070-1721, October 2010.

[13] Bagnulo; Sullivan; Matthews; Beijnum. *DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers, RFC 6147*, Internet Engineering Task Force (IETF), Standards Track, ISSN: 2070-1721, April 2011.

[14] Beck, Martin. *Enhanced TKIP Michael Attacks*. TU-Dresden, Germany. February 25, 2010. pp.10

[15] Bert, Johannes. *About mac80211*, Linux Wireless, https://wireless.wiki.kernel.org /en/-developers/documentation/mac80211, June 2016. Disponible en Octubre de 2016.

[16] Bicakci, Kemal and Tavli, Bulent *Denial-of-Service attacks and countermeasures in IEEE 802.11 wireless networks*, Computer Standards & Interfaces, TOBB University of Economics and Technology, Electrical and Electronics Engineering Departament, Sogutozu Caddesi No. 43, Nov 2008.

[17] Bradner, Scott. *Status report on IPng*, Network World; Apr 17, 1995; 12, No. 16; ProQuest SciTech Collection pg. 23.

[18] Cam-Winget, N; Zhou, H. *Basic Password Exchange within the Flexible Authentication via Secure Tunneling Extensible Authentication Protocol (EAP-FAST), RFC 5421*, Network Working Group, Informational, March 2009.

[19] Congdon, P; Aboba, B; et. al. *IEEE 802.1X Remote Authentication Dial In User Service (RADIUS), Usage Guidelines, RFC 3580*, Network Working Group, Informational, September 2003.

[20] Conta, A; Deering, S. *Generic Packet Tunneling in IPv6 Specification, RFC 2473*, Network Working Group, Proposed Standard, December 1998.

[21] Coova.org. *Documento técnico de CoovaChilli*, https://coova.github.io, Disponible en Octubre de 2016.

[22] DeKok, Alan. *rlm_pap - FreeRADIUS Module*, https://freeradius.org/radiusd/man/rlm_pap.txt, 10 January 2015.

[23] Diffie, Whitfield and Hellman, Martin E. *New Directions in Cryptography*, Institute of Electrical and Electronics Engineers (IEEE). Transactions on Information Theory, vol. IT-22, no. 6, p644−−654, 11p. November 1976.

[24] Draves, R. *Default Address Selection for Internet Protocol version 6 (IPv6), RFC 3484*, Network Working Group, Proposed Standard, February 2003.

[25] Droms, R. *Dynamic Host Configuration Protocol, RFC 2131*, Network Working Group, Bucknell University, Standards Track, March 1997.

[26] Droms; Bound; Volz; Lemon; Perkins; Carney. *Dynamic Host Configuration Protocol for IPv6 (DHCPv6), RFC 3315*, Network Working Group, Standards Track, July 2003.

[27] Eastlake, D. *Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH), RFC 4305*, Motorola Laboratories, Network Working Group, December 2005.

[28] El-Nagar, Ahmed; El-Hafez, Ahmed; Elhnawy, Adel. *A novel EAP-moderate weight Extensible Authentication Protocol*, Computer Engineering Conference (ICENCO), 2011 Seventh International, 27-28 December 2011, p1-6.

[29] Eissa, M; Ali, Ihab; Abdel-Latif. *Wi-Fi protected access for secure power network protection scheme*, International Journal of Electrical Power & Energy Systems, Vol. 46, March 2013, p.414-424.

[30] Free electrons, *Buildroot user manual*, https://buildroot.org/docs, Disponible en Noviembre de 2016.

[31] Fuentes, Laura; De la Fraga, Luis G. *Puerta en una SBC*, Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, Departamento de Computación, Junio 2010, 7 pp.

[32] Harkins, D. and Carrel D. *The Internet Key Exchange (IKE), RFC 2409* , Cisco Systems, Network Working Group, Standards Track, November 1998.

[33] Hinden R. and Deering S, *IP Version 6 Addressing Architecture, RFC 4291*, Network Working Group, Standards Track, February 2006.

[34] Jakobsen, Jens. *Documento técnico de Chillispot*, http://www.chillispot.org/, Disponible en Octubre de 2016.

[35] Kent, Stephen and Seo K. *Security Architecture for the Internet Protocol, RFC 4301*, BBN Technologies, Network Working Group, December 2005.

[36] Kerrisk, Michael. *The Linux Programming Interface, A Linux and UNIX System Programming Handbook*, http://man7.org/linux/man-pages/man7/ipv6.7.html, March 29, 2015.

[37] Lenczner, Michael. *Wireless Portals with Wifidog.* Linux Journal. Oct 31, 2005. pp.5.

[38] Linux man page. *Netdevice - low-level access to Linux network devices*, https://linux.die.net/man/7/netdevice. Disponible Octubre 2016.

[39] Liu, Fanbao; Xie, Tao. *How to Break EAP-MD5*, School of Computer, National University of Defense Technology, Changsha, 410073, Hunan, P.R. China. 9pp.

[40] Malinen, Jouni. *hostapd: IEEE 802.11 AP, IEEE 802.1X/WPA/WPA2/EAP/RADIUS Authenticator*, https://w1.fi/hostapd/, 2013. Disponible en Octubre de 2016.

[41] Meehan, Tom. *IPv6 vs. IPv4 write tale of two protocols*, Electronic Engineeering Times; Aug 18, 1997; 967; ProQuest SciTech Collection pg. 82.

[42] Narten, T. *IEEE Internet Computing*, Jul/Aug 1999, Vol.3 Issue 4, p54−−62, 9p, 2 Charts. Publisher: IEEE.

[43] NLnet Labs. *Documento técnico de Unbound*, NLnet Labs foundation, http://unbound.net/, November 2007 - May 2016.

[44] Nordmark, E. and Gilligan, R. *Basic Transition Mechanisms for IPv6 Hosts and Routers, RFC 4213*, Network Working Group, Proposed Standard, October 2005.

[45] Okuji, Yoshinori. *Documento técnico de GNU GRUB*, Free Software Foundation, Inc. 2010. Resource http://www.gnu.org/software/grub/

[46] Postel, J. *Internet Protocol - DARPA Internet Program Protocol Specification, RFC 791*, USC/Information Sciences Institute, September 1981.

[47] Powers, Shawn. *DNSMasq, the Pint-Sized Super Daemon!*, Linux Journal, Issue 245, p.50−−57, September 2014.

[48] Rempel, Oleksij. *USB related issues*, Open ath9k-htc firmware, https://wireless.wiki.kernel.org/en/users/drivers/carl9170, disponible en Noviembre de 2016.

[49] Sarmiento, Oscar; Guerrero, Fabio; Argote, David. *Basic security measures for IEEE 802.11 wireless networks*, Ingeniería e Investigación, vol. 28, no. 2, Bogotá, Mayo-Agosto 2008.

[50] Schlyter, J. *DNS Security (DNSSEC) NextSECure (NSEC) RDATA Format, RFC 3845*, The Internet Society (2004), Network Working Group, Standards Track, August 2004.

[51] Sebastien Vincent, Thibault Vançon *Documento técnico de Pepperspot*, pepperspot.sourceforge.net, Disponible Octubre de 2016.

[52] Selinger, Peter. *MD5 Collision Demo*, http://www.mscs.dal.ca/ selinger/md5collision/, October, 2011.

[53] Simon, D; Aboba, B; Hurst, R. *The EAP-TLS Authentication Protocol, RFC 5216*, Network Working Group, Standards Track, March 2008.

[54] The Apache Software Foundation. *Apache HTTP Server Project*, https://httpd.apache.org/, 1997-2016. Disponible en Octubre 2016.

[55] The Wifidog Project. *Wifidog, A captive portal suite*, Wifidog Flow Diagram, http://dev.wifidog.org/wiki/doc/developer/FlowDiagram, Disponible Octubre 2016.

[56] Thompson, S; Huitema, C; et. al. *DNS Extensions to Support IP Version 6, RFC 3596*, Network Working Group, Draft Standard, October 2003.

[57] Thomson, Narten and Jinmei. *IPv6 Stateless Address Autoconfiguration, RFC 4862*, Network Working Group, Standards Track, September 2007.

[58] Viagenie. *Ecdysis: open-source implementation of a NAT64 gateway.* 2009−−2014. Resource http://ecdysis.viagenie.ca/

[59] Welte, Harald; Ayuso, Pablo. *The netfilter.org "iptables" project*, Netfilter firewalling, NAT and packet mangling for Linux, 1999−−2014. http://www.netfilter.org/projects/iptables/index.html

[60] Wong, Luis Carlos. An Overview of 802.11 Wireless Network Security Standards & Mechanisms, GIAC Security Essentials Certification (GSEC), Practical Assignment 1.4c, 21 October 2004.

[61] Xia, Haidong and Brustoloni, José. *Detecting and Blocking Unauthorized Access in Wi-Fi Networks.* Lecture Notes in Computer Science, University of Pittsburgh, Dept. Computer Science, Networking 2004, p. 795-806