Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros de Telecomunicación

DISEÑO DE ESCENARIOS DE TRANSICIÓN A IPV6 UTILIZANDO LA HERRAMIENTA VNX: DNS, SERVICIOS WEB Y MECANISMOS DE TRANSICIÓN

TRABAJO FIN DE MÁSTER

Gloria Martín Martín

2011

Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros de Telecomunicación

Máster Universitario en
Ingeniería de Redes y Servicios Telemáticos

TRABAJO FIN DE MÁSTER

DISEÑO DE ESCENARIOS DE TRANSICIÓN A IPV6 UTILIZANDO LA HERRAMIENTA VNX: DNS, SERVICIOS WEB Y MECANISMOS DE TRANSICIÓN

Autor

Gloria Martín Martín

Director

David Fernández Cambronero

Departamento de Ingeniería de Sistemas Telemáticos

2011

Resumen

Cuando utilizamos Internet para cualquier actividad, ya sea correo electrónico, navegación web, descarga de ficheros, o cualquier otro servicio o aplicación, la comunicación entre los diferentes elementos de la red y nuestro propio ordenador o teléfono, utiliza un protocolo que denominamos Protocolo de Internet (IP, Internet Protocol).

En los últimos años, prácticamente desde que Internet tiene un uso comercial, la versión de este protocolo es la número 4 (IPv4).

Para que los dispositivos se conecten a la red, necesitan una dirección IP. Cuando se diseñó IPv4, casi como un experimento, no se pensó que pudiera tener tanto éxito comercial, y dado que sólo dispone de 2^{32} direcciones (direcciones con una longitud de 32 bits, es decir, 4.294.967.296 direcciones), junto con el imparable crecimiento de usuarios y dispositivos, implica que este número resulta claramente insuficiente.

A raíz de la entrega por parte del IANA de los últimos paquetes con direcciones de Internet IPv4 en Febrero de 2011, la necesidad de una transición al protocolo llamado a sustituirle, el protocolo IPv6, se hace cada vez más patente, especialmente en los ámbitos de docencia e investigación técnica que deben ser los pioneros en estos procesos.

En este Trabajo Fin de Máster vamos a desarrollar un escenario simulado que nos permita modelar la transición del protocolo de red IPv4 al protocolo IPv6 en una escuela de la Universidad Politécnica de Madrid (UPM en adelante), en concreto en la Escuela de Ingeniería Aeronáutica y del Espacio (EIAE en adelante), una escuela que conocemos y que demanda cada vez más la transición al nuevo protocolo de red para promover la investigación y mejorar las actividades docentes.

Utilizaremos, para realizar esta simulación, una herramienta desarrollada por el Departamento de Ingeniería de Sistemas Telemáticos de la Escuela Técnica Superior de Ingenieros de Telecomunicación llamada VNX/VNUML que nos permitirá generar el escenario completo con el único requisito de disponer de una máquina anfitriona relativamente potente en cuanto a memoria, capacidad de proceso y almacenamiento.

Finalmente cabe resaltar que esta memoria forma parte de un trabajo conjunto realizado con Rafael García García en el que se ha desarrollado el escenario completo,

por lo que ambos trabajos presentan una parte común (introducción y descripciones generales) y una parte específica que se describe en cada una de las dos memorias. Este Trabajo Fin de Master realiza la descripción de una parte del escenario completado en el Trabajo Fin de Máster de Rafael García García donde se describe el resto de servicios y configuraciones que lo componen.

Abstract

When we use the Internet for any activity, whether email, web browsing, downloading files, or any other service or application, communication between different network elements and our own computer or telephone, is using a protocol called Internet Protocol (IP, Internet Protocol).

In recent years, practically since the Internet has a commercial use, the version of this protocol is the number 4 (IPv4).

For devices to connect to the network they need an IP address. When IPv4 was designed, almost as an experiment, it was not thought that it could have so much commercial success, and since it only have 2^{32} addresses (addresses with a length of 32 bits, 4.294.967.296 addresses), along with the unstoppable growth of users and devices, means that this number is clearly insufficient.

Following the delivery by the IANA of the latest packages with IPv4 Internet addresses in February 2011, the need for a transition to the protocol designed to replace IPv4, the IPv6 protocol, it becomes increasingly evident, especially in the areas of teaching and technical research which are called to be the pioneers in these processes.

In this Master's Thesis we will develop a simulated scenario that will allow us to model the transition from IPv4 network protocol to IPv6 in a school of the Polytechnic University of Madrid (UPM), particularly in the College of Engineering Aeronautics and Space (EIAE) a school we know and demand increasingly a transition to the new network protocol to promote research and improve teaching activities.

We will use to perform this simulation, a tool developed by the Department of Telematic Systems Engineering of the School of Telecommunications Engineers named VNX/VNUML. With this tool we will be able to generate the whole scenario with the only requirement to have a relatively powerful host machine for memory, processing power and storage.

Finally it should be noted that this memory is part of a joint work with Rafael García García in which we have developed the complete scenario, so that both jobs have a common part (introduction and general description) and a specific part that is described in each of the two memories. This Master's Thesis makes the description of a

part of the scenario completed in the Master's Thesis of Rafael García García, which describes the other services and settings that compose it.

Índice general

Índice de figuras

Siglas

BIND: Berkeley Internet Name Domain.

DHCP: Dynamic Host Configuration Protocol.

DNS: Domain Name System.

DNS64: Domain Name System 6 to 4.

FTP: File Transfer Protocol.

IPv6: Significa Internet Protocol version 6.

IPv4: Significa Internet Protocol version 4.

HTTP: HyperText Transfer Protocol.

MAC: Media Access Control address.

NAT64: Network Address Translation 6 to 4.

UML: User mode Linux.

VLAN: Virtual Local Area Network.

VNUML: Virtualization Network User Mode Linux.

VNX: Virtual Networks over LinuX.

VSFTPD: Very Secure FTP Daemon.

XML: eXtensible Markup Language.

WWW: World Wide Web.

1 Introducción

El impulso del protocolo IPv6 se debe sobre todo a un motivo: la necesidad de más direcciones. Hoy en día hay millones de nuevos dispositivos como teléfonos móviles, PDAs, dispositivos de consumo, etc., algunos de los cuales necesitan más de una dirección IP. Además, se ha incrementado el número de usuarios incorporándose a Internet en países como China e India.

Inicialmente se buscaron soluciones que permitieran compartir direcciones utilizando mecanismos como NAT, PPP, etc., pero esto, aunque ha permitido el crecimiento de Internet, ha provocado problemas debido a la pérdida de conectividad extremo a extremo, lo que conlleva que algunas aplicaciones dejen de funcionar.

Finalmente se vio que la necesidad de un nuevo protocolo era ineludible, pues aún en el caso de una utilización más óptima de las direcciones IP, el número de ellas que proporciona IPv4 resulta insuficiente y el uso de NAT hace más costoso y complejo el desarrollo de servicios y aplicaciones y por lo tanto puede frenar la innovación en Internet.

Hoy en día IPv6 ya no es un protocolo en fase de diseño y experimentación y el trabajo técnico relacionado con este protocolo prácticamente está finalizado. Hoy todas las ventajas que introduce este nuevo protocolo (espacio virtualmente ilimitado de direcciones, seguridad a nivel de red, movilidad, multicast, etc.) aparecen disponibles ante un mundo que en el que se prevé que IPv6 genere una nueva ola de innovación en las aplicaciones y en la oferta de servicios ya que termina con la necesidad de direcciones compartidas.

La mayoría de sistemas operativos están preparados para el nuevo protocolo y la migración al mismo no debería suponer problemas al usuario final por eso, a pesar de que desde el punto de vista profesional existe aún cierto recelo a la introducción de esta nueva tecnología, la convergencia tecnológica a medio plazo hacia el nuevo protocolo IPv6 es inevitable.

La implantación de IPv6 no es, ni va a ser, inmediata. Además de todo el esfuerzo invertido en los últimos años en su desarrollo, ahora es necesario desplegar esta nueva tecnología en la red. Esto requiere actualizar el software y, en algunos casos, el hardware de millones de equipos y también conllevará cambios en las aplicaciones, procedimientos y herramientas de gestión de redes.

Hasta ahora han sido muy pocos los servicios y contenidos de Internet accesibles mediante IPv6, por lo que el tráfico IPv6 visible en la red es aún mínimo.

Es importante además que las Universidades y Centros de Investigación lideren el proceso de transición a esta nueva tecnología.

El objetivo de este proyecto es realizar, mediante una potente herramienta de virtualización como es VNX, un escenario simulado en el que realizar la transición de IPv4 al IPv6 en una Escuela de la UPM, tomando como objetivo la implantación de algunos de los servicios más relevantes que se ofrecen en la Universidad como pueden ser el servicio DNS, el servicio WEB, el servicio DHCP, etc.

A lo largo de este Trabajo Fin de Máster desarrollaremos un modelo en el que a través de varias redes, servidores y clientes instalaremos y configuraremos los diferentes servicios tanto en IPv4 como en IPv6. Así mismo realizaremos las pruebas pertinentes para comprobar el correcto funcionamiento de la red y los servicios con ambos protocolos.

También se estudiará uno de los mecanismos de transición para comunicar IPv4 e IPv6 durante el tiempo que necesiten coexistir. Este mecanismo será el mecanismo de los traductores de protocolos y en concreto se experimentará en el escenario propuesto con el traductor NAT64.

En este Trabajo Fin de Máster forma parte de un trabajo conjunto realizado con Rafael García García. En esta memoria se detallará una parte de los trabajos realizados, y el resto se explican en el Trabajo Fin de Máster de Rafael GarcíaGarcía. Ambos trabajos se presentan conjuntamente por formar parte del mismo objetivo.

2 IPv4 vs IPv6

En este apartado vamos a describir brevemente los protocolos de Internet versión 4 (IPv4) y versión 6 (IPv6) así como los factores que han provocado la necesaria implantación de IPv6 como futuro protocolo de Internet.

2.1 Protocolos de red

La comunicación en las redes de información es posible gracias a los protocolos de red. Un protocolo es un conjunto de reglas que controla o permite la comunicación entre dos o más dispositivos mediante la transmisión y recepción de paquetes de bits con una cierta estructura. En Internet, el protocolo más utilizado es el TCP/IP (Transmission Control Protocol/Internet Protocol).

2.2 IPv4

IP es un protocolo ideado para interconexión de redes heterogéneas mediante routers. Se trata de un protocolo de conexión no fiable, que no garantiza la entrega segura de los paquetes. Además, los paquetes que se transmiten a la red, aunque pertenezcan a un mismo mensaje original, pueden seguir caminos diferentes, por lo que pueden llegar desordenados e incluso duplicados. Deberá ser la capa de transporte, o incluso la propia aplicación, la que, en su caso, detecte y resuelva todas estas situaciones de error. Las unidades de información que se transmiten a nivel del protocolo IP se denominan paquetes IP o datagramas.

El protocolo IP tiene tres funciones básicas:

- **Direccionamiento**. IP debe proporcionar un conjunto global de direcciones que permitan identificar de forma unívoca a cada una de las máquinas conectadas a Internet. Estas direcciones se conocen con el nombre de direcciones IP y no deben confundirse con las direcciones físicas o MAC que se utilizan a nivel de la capa de control de acceso al medio en redes de área local.
- **Encaminamiento**. IP debe incorporar mecanismos de encaminamiento eficientes que permitan a todas las estaciones y routers de Internet encaminar correctamente los datagramas en función de su destino. Para poder llevar a cabo todas estas funciones, todos los datagramas que se transmiten a la red deben incluir las direcciones IP de las maquinas origen y destino.

- **Fragmentación**. Cuando un datagrama tiene que cruzar a través de una o varias redes en el camino hacia su destino, el protocolo IP debe encargarse de dividir el paquete en fragmentos de un tamaño aceptable por cada una de las redes que atraviesa. Igualmente, en el destino, el protocolo IP debe ser capaz de re-ensamblar los distintos fragmentos recibidos para formar el datagrama original.

IPv4 es la versión 4 del Protocolo IP. Esta fue la primera versión del protocolo que se implementó extensamente, y forma la base de Internet. Fue publicado en 1981 en la RFC 791.

Tiene la capacidad de ofrecer más de 4.200 millones de direcciones, pero no está capacitado para cubrir la demanda actual existente, no sólo por el número de direcciones que soporta, sino también por la forma en que agrupa los bits en su sistema de numeración network/host (que desperdicia direcciones y sufre excesiva sobrecarga en el routing).

2.2.1 Formato del datagrama IPv4

Un datagrama IP está formado por una cabecera IP y una zona de datos. La cabecera tiene un tamaño mínimo de 20 bytes y está formada por palabras de 32 bits (5 palabras como mínimo).

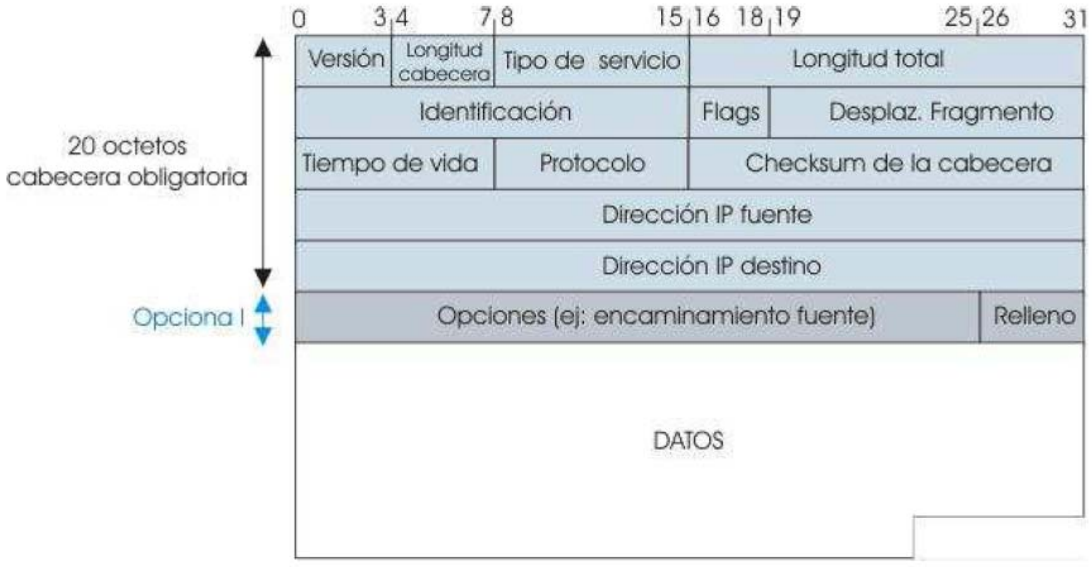

Figura 1. Estructura de un paquete IPv4

2.2.2 Limitaciones de IPv4

IPv4 usa direcciones de 32 bits, limitándose así a 2^{32} = 4.294.967.296 direcciones. Esto, que en un principio puede parecer más que suficiente, en realidad no lo es tanto.

Además hay que tener en cuenta que no todas las combinaciones están disponibles para el protocolo IP público. Hay una serie de combinaciones reservadas lo que hace que el número real de IP's disponibles no sea tan elevado.

En principio hay que asegurar una serie de combinaciones diferentes para las conexiones a Internet, por lo que a cada proveedor ISP (Internet Service Provider) se le asigna un determinado número de direcciones IP, asignándolas estos a su vez entre sus clientes. Para optimizar este número de conexiones los proveedores ISP recurren al sistema de IP dinámica. Este sistema hace posible que con un número limitado de IP's se atienda a un número bastante superior de usuarios, a condición de que el número de conexiones simultaneas no supere el número de IP's asignadas.

Además hay que tener en cuenta que este tipo de conexiones es cada vez más empleado. No sólo por ordenadores, sino también por dispositivos de otro tipo, tales como, por ejemplo, cámaras IP, comunicaciones de voz del tipo VoIP, teléfonos móviles, PDA, etc., lo que ha provocado, junto al cambio de hábitos en las conexiones (hemos pasado de conexiones por un corto periodo de tiempo cuando nos conectábamos por RTB a tener conectado el ordenador las 24 horas, o al menos 8 horas diarias) y el incremento en el número de usuarios (que prácticamente se duplica cada año desde 1.988), que el número de conexiones disponibles no sólo no sea exagerado, sino que sea claramente insuficiente.

Una de las consecuencias de este sistema es la necesidad de utilizar para conectarse a la red (Internet) un sistema que permita una sola IP por conexión (independientemente de los ordenadores que luego se conecten a través de esta conexión). Este sistema es el denominado NAT (Network Address Translation), y permite mediante un router tener una red interna (direcciones IP privadas) apuntando a una sola dirección de Internet (IP pública).

Esta serie de limitaciones han ayudado a estimular el impulso hacia IPv6.

2.3 IPv6

IPv6 o IPng (Next Generation Internet Protocol) es la nueva versión del protocolo IP. Ha sido diseñado por el IETF (Internet Engineering Task Force) para reemplazar de forma gradual a la versión actual, el IPv4.

2.3.1 Características principales

IPv6 surge, sobre todo, para resolver los problemas de direccionamiento y encaminamiento detectados en los años 90 en Internet.

IPv6 es una evolución de IPv4, no es una revolución ya que no aporta grandes novedades, sino que aprovecha los años de experiencia que aporta IPv4.

Las características más importantes que aporta IPv6 son las siguientes:

- **Mayor espacio de direcciones**. El tamaño de las direcciones IP cambia de 32 bits a 128 bits, para soportar más niveles de jerarquías de direccionamiento y más nodos direccionables.
- **Simplificación del formato de la cabecera**. Algunos campos de la cabecera IPv4 se quitan o se hacen opcionales**.**
- **Paquetes IP eficientes y extensibles**, sin que haya fragmentación en los routers, alineados a 64 bits y con una cabecera de longitud fija, más simple, que agiliza su procesado por parte del router.
- **Posibilidad de paquetes con carga útil** (datos) de más de 65.355 bytes.
- **Seguridad en el núcleo del protocolo**. El soporte de IPsec es un requerimiento del protocolo IPv6.
- **Capacidad de etiquetas de flujo**. Puede ser usada por un nodo origen para etiquetar paquetes pertenecientes a un flujo de tráfico particular, que requieren manejo especial por los routers IPv6, tal como calidad de servicio o servicios de tiempo real.
- **Autoconfiguración** de direcciones es más simple. Especialmente en direcciones Aggregatable Global Unicast, los 64 bits superiores son enviados por medio de un mensaje desde el router (Router Advertisement) y los 64 bits más bajos son obtenidos a partir de la dirección MAC (en formato EUI-64). En este caso, el largo del prefijo de la subred es 64, por lo que no hay que preocuparse más por la máscara de red. Además el largo del prefijo no depende en el número de los hosts por lo tanto la asignación es más simple.
- **Re-numeración y "multihoming"**, facilitando el cambio de proveedor de servicios.
- **Características de movilidad**, posibilitando a un nodo mantener la misma dirección IP, a pesar de su movilidad. Frente a IPv4 se mejora la eficiencia y seguridad.

- **Enrutamiento más eficiente en el backbone de la red**, debido a la jerarquía de direccionamiento basada en agregación. El encaminamiento es jerárquico basado en la agregación de rutas.
- **Calidad de servicio (QoS) y clase de servicio (CoS)**.
- **Capacidades de autenticación y privacidad.**
- **Multicast**: envío de un mismo paquete a un grupo de receptores.
- **Anycast**: envío de un paquete a un receptor dentro de un grupo.

2.3.2 Formato del paquete

El formato de la cabecera de un paquete IPv6 es el siguiente:

Figura 2. Formato cabecera IPv6

La longitud de esta cabecera es de 40 bytes, el doble que en el caso de IPv4, pero con muchas ventajas, al haberse eliminado campos redundantes.

Además, como ya hemos mencionado, la longitud fija de la cabecera, implica una mayor facilidad para su procesado en routers y conmutadores, incluso mediante hardware, lo que implica unas mayores prestaciones.

A este fin ayuda, como hemos indicado anteriormente, el hecho de que los campos están alineados a 64 bits, lo que permite que las nuevas generaciones de procesadores y microcontroladores, de 64 bits, puedan procesar mucho más eficazmente la cabecera IPv6.

- El **campo de versión**, que es igual a 6, lógicamente, tiene una longitud de 4 bits.
- **Clase de Tráfico**, también denominado Prioridad, o simplemente Clase. Tiene una longitud de 8 bits (1 byte).
- **Etiqueta de Flujo**, para permitir tráficos con requisitos de tiempo real. Tiene una longitud de 20 bits.

- **Longitud de la carga útil**: es la longitud de los propios datos, y puede ser de hasta 65.536 bytes. Tiene una longitud de 16 bits (2 bytes).
- **Siguiente cabecera**: dado que en lugar de usar cabeceras de longitud variable se emplean sucesivas cabeceras encadenadas, desaparece el campo de opciones. En muchos casos ni siquiera es procesado por los routers, sino tan sólo extremo a extremo. Tiene una longitud de 8 bits (1 byte). Este campo indica qué viene detrás de la cabecera y pueden ser paquetes TCP, cabeceras opcionales, etc.
- **Límite de saltos:** tiene una longitud de 8 bits (1 byte). No es necesario aumentarlo ya que las redes son jerárquicas y no se prevé que un paquete haga más de 256 saltos.

Las **cabeceras opcionales** se definen en el valor del campo "siguiente cabecera" e indica cual es la siguiente cabecera y así sucesivamente. Las sucesivas cabeceras, no son examinadas en cada nodo de la ruta, sino sólo en el nodo o nodos destinos finales. Hay una única excepción a esta regla: cuando el valor de este campo es cero, lo que indica opción de examinado y proceso "salto a salto" (hop-by-hop). Así tenemos, por citar algunos ejemplos, cabeceras con información de encaminado, fragmentación, opciones de destino, autenticación, cifrado, etc., que en cualquier caso, han de ser procesadas en el orden riguroso en que aparecen en el paquete.

Los tipos de cabeceras opcionales definidas son las siguientes:

- ***Hop by Hop Options***: esta información debe ser examinada en cada salto.
- ***Routing***: similar a la opción Source Route de IPv4.
- ***Fragmentation***: Segmentación y re-ensamblado.
- ***Authentication***: Firmas digitales.
- ***Security Encapsulation***: Cifrado.
- ***Destination Options***: Información examinada sólo en destino.

Sin entrar en más detalles, véanse a continuación los siguientes ejemplos gráficos del uso del concepto de las "cabeceras de extensión" (definidas por el campo "siguiente cabecera"), mecanismo por el que cada cabecera es "encadenada" a la siguiente y anterior (si existen):

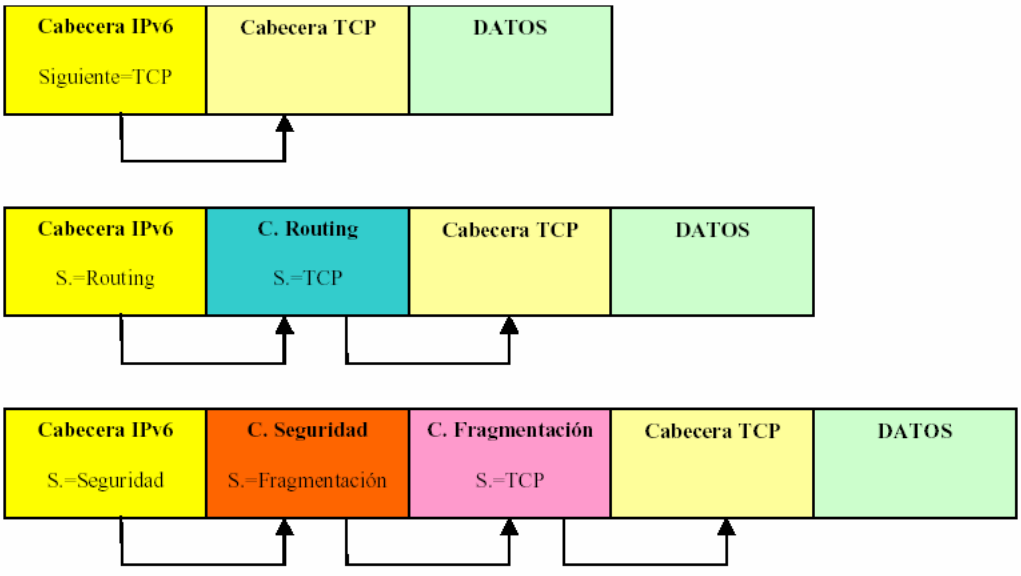

Figura 3. Cabeceras IPv6

La mejora del **formato del paquete IPv6** ofrece las siguientes ventajas:

- No hay limitaciones en el número de opciones.
- Mejora las prestaciones debido a la ordenación de la cabecera:
  - Cabeceras procesadas por los routers más hacia la derecha, hacia el exterior de la cabecera.
  - Cabeceras procesadas en destino, más hacia la izquierda, hacia el interior de la cabecera.

Las cabeceras opcionales están pensadas para facilitar el trabajo sobre todo a los routers ya que pueden saber qué tipo de opciones incluye la cabecera y ver si le interesa o no. Por ejemplo, a un router no le interesa una opción que sea una firma digital. La idea es que sólo analice lo que tiene que analizar porque le sea interesante para el proceso de encaminamiento de los paquetes.

2.3.3 Direccionamiento en IPv6

Modelo de direccionamiento

Las direcciones son de 128 bits e identifican interfaces individuales o conjuntos de interfaces. Al igual que en IPv4 los nodos se asignan a interfaces. De esta forma, podemos encontrar interfaces con una dirección asignada o interfaces con varias direcciones asignadas.

Cada máquina conectada a la red dispone de dos direcciones:

- **Dirección de ámbito local**: le permite comunicarse con los equipos que están en su misma red. Esta dirección se obtiene mediante la autoconfiguración.

- **Dirección de ámbito global:** le permite comunicarse con cualquier equipo conectado a la red Internet.

Las direcciones globales permiten a los sistemas conectarse a Internet, para ello se necesita obtener el prefijo global. Este prefijo lo obtienen los sistemas finales del router de la red ya que de forma periódica éste envía información del prefijo global. De esta forma, los equipos pueden conocer la dirección del router de la red y de la dirección global del sistema.

Las direcciones se clasifican en tres tipos:

- ***Unicast,*** que identifican a una sola interfaz. Un paquete enviado a una dirección unicast es entregado sólo a la interfaz identificada con dicha dirección. Es el equivalente a las direcciones IPv4 actuales. [RFC 2373][RFC 2374]

- ***Anycast,*** que identifican a un conjunto de interfaces (típicamente pertenecen a diferentes nodos). Un paquete enviado a una dirección anycast, será entregado a una (cualquiera) de las interfaces identificadas con la dirección del conjunto al cual pertenece esa dirección anycast (la más próxima, de acuerdo a las medidas de distancia del protocolo de encaminamiento). Nos permite crear, por ejemplo, ámbitos de redundancia, de forma que varias máquinas puedan ocuparse del mismo tráfico según una secuencia determinada (por el routing), si la primera "cae". [RFC 2526]

- ***Multicast*** que identifican un grupo de interfaces (por lo general pertenecientes a diferentes nodos). Cuando un paquete es enviado a una dirección multicast es entregado a todos las interfaces del grupo identificadas con esa dirección. En el IPv6 no existen direcciones broadcast, su funcionalidad ha sido mejorada por las direcciones multicast. [RFC 2375]

Vemos que desaparece la dirección Broadcast utilizada en IPv4 ya que lo educado es el empleo de Multicast, ya que sólo tiene que recibirlo la máquina que quiere recibirlo, el resto no deben recibir esos paquetes.

La estructura de las direcciones es la siguiente:

Figura 4. Estructura de las direcciones IPv6

La idea a la hora de diseñar el modelo de direcciones es separar en la dirección la parte que identifica "quién eres" y la parte que indica "dónde estás conectado". Teniendo esto en cuenta tenemos que:

- **Prefijo**: depende de la topología de la red e indica "dónde estás conectado".
- **Identificador de interfaz**: identifica a un nodo o lo que es lo mismo, "Quién eres".

Un ejemplo podría ser:

Figura 5. Ejemplo de dirección IPv6

De acuerdo con esta estructura se podría pensar que la dirección del identificador de interfaz estaría muy sobredimensionado ya que se reservan 64 bits para definirlo (no tiene sentido una red en la que haya 2^{64} equipos). Al reservar tantos bits se hizo pensando en:

- Facilitar la autoconfiguración a partir de direcciones MAC más algo más, ya que se piensa que en un futuro la dirección MAC de 48 bits se quedará pequeña. La máquina al arrancar se construye la dirección de la misma.
- Dificultar los ataques sistemáticos ("address/port scanning") ya que para escanear todas las máquinas posibles de una red (2^{64} equipos) se requiere mucho tiempo.

Además, en previsión de garantizar la privacidad de las máquinas cuando naveguen por Internet, se ha previsto una extensión a la autoconfiguración que permite configurar las máquinas (la parte de identificador de interfaz) con números aleatorios que varíen con el tiempo. De esta forma, aunque conozcan la dirección MAC de una máquina, no pueden saber la dirección IP de la misma. De esta forma se consigue:

- Garantizar la privacidad al navegar por Internet (se genera la dirección local de forma aleatoria y se cambia cada cierto tiempo).
- Para conexiones locales se hace uso de la dirección MAC para generar la dirección local.

2.4 Estrategias de transición a IPv6 (RFC1933)

La clave para la transición es la compatibilidad con la base instalada de dispositivos IPv4. Esta afirmación define un conjunto de mecanismos que los hosts y routers IPv6 pueden implementar para ser compatibles con host y routers IPv4.

Estos mecanismos permitirán usar infraestructuras IPv4 para IPv6 y viceversa, dado que se prevé que su uso será prolongado, e incluso indefinido en muchas ocasiones.

Los mecanismos de transición son un conjunto de mecanismos y de protocolos implementados en hosts y routers, junto con algunas guías operativas de direccionamiento designadas para hacer la transición de Internet al IPv6 con la menor interrupción posible.

Dichos mecanismos están diseñados para ser usados por hosts y routers IPv6 que necesitan inter-operar con hosts IPv4 y utilizar infraestructuras de enrutamiento IPv4. No obstante, IPv6 también puede ser usado en ambientes donde no se requiere interoperabilidad con IPv4. Nodos diseñados para esos ambientes no necesitan usar ni implementar estos mecanismos.

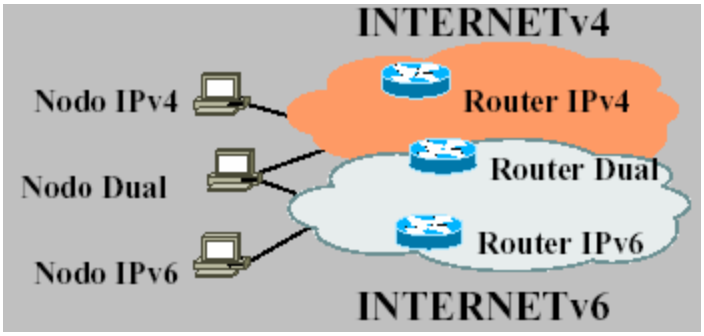

Figura 6. Transición de IPv6 a IPv4

La transición es muy compleja ya que se deben actualizar routers, hosts, aplicaciones, gestión y operación, formación del personal, etc. Conscientes de la complejidad de la transición, IPv6 se ha diseñado pensando en la transición.

Estos mecanismos son los siguientes:

- **Nodos duales o doble pila IPv6 –IPv4.**
- **Túneles.**
- **Traductores de protocolos.**

De estos tres mecanismos en este Trabajo Fin de Master utilizaremos dos de ellos: la pila dual de protocolos y los traductores de protocolos. Veamos con más detalle cada uno de estos mecanismos.

2.4.1 Nodos duales o doble pila IPv4 – IPv6 (Dual Stack)

El camino más lógico y evidente de transición es el uso simultáneo de ambos protocolos, en pilas separadas. Los dispositivos con ambos protocolos también se denominan "nodos IPv6/IPv4".

Figura 7. Pila Dual IPv4 e IPv6

Estos nodos se caracterizan por:

- El dispositivo tendrá una dirección en cada pila. Se pueden utilizar direcciones IPv4 e IPv6 relacionadas o no, y se pueden utilizar mecanismos manuales o automáticos para la asignación de las direcciones (cada una correspondiente al protocolo en cuestión).
- Sólo se duplica el nivel IP, no la pila completa.
- Se comunican utilizando:
    - IPv6 con nodos ya migrados.
    - IPv4 con nodos no migrados.

  De esta forma, un dispositivo con ambas pilas pueden recibir y enviar tráfico a nodos que sólo soportan uno de los dos protocolos (nodos sólo IPv4 o sólo IPv6). 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.

- Incluyen bibliotecas de DNS (resolver) capaces de tratar con registros A y AAAA. El DNS es quién conoce qué nodos han migrado (registros AAAA disponibles).
- Las aplicaciones deben ser conscientes de la dualidad:
    - Los clientes deben elegir entre v4 o v6 a la hora de conectarse.
    - Los servidores de DNS deben escuchar en v4 y v6.
- Ante una consulta al DNS sobre un nombre, el DNS podrá devolver la dirección IPv4 (registro A), la dirección IPv6 (registro AAAA), o ambas.
- En las redes pueden coexistir paquetes IPv4 e IPv6 ya que los protocolos de nivel inferior en los paquetes existe un campo que identifica al protocolo que transporta:
    - 0x0800 si el protocolo es IPv4.
    - 0x86dd si el protocolo es IPv6.
- Habrá tablas de encaminamiento para IPv4 y para IPv6.
- Hay muchos sistemas como Linux o Windows que ya lo soportan.

Este funcionamiento básico del funcionamiento de la pila dual de protocolos que hemos explicado se representa en la siguiente figura:

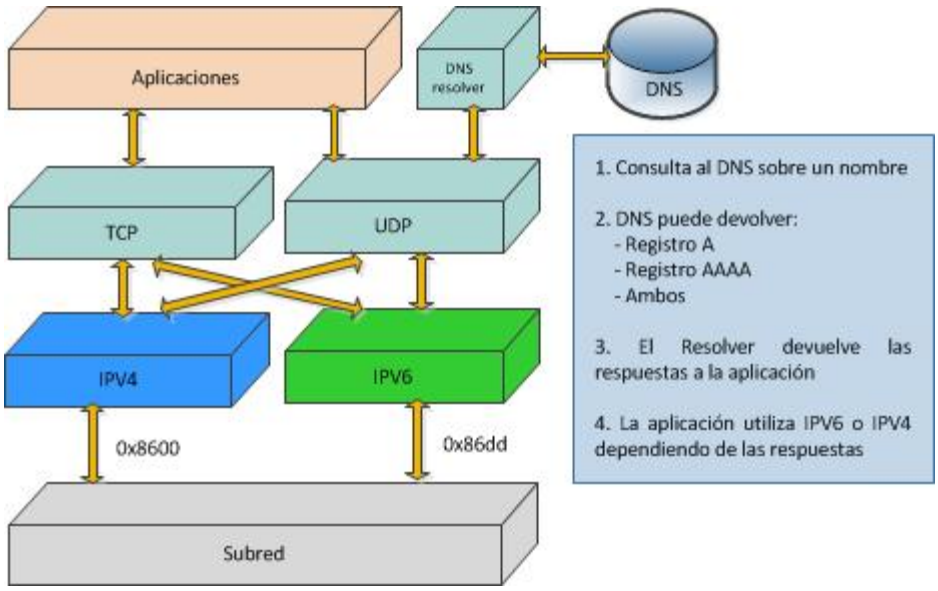

Figura 8. Funcionamiento de la Pila Dual

Se pueden emplear la dirección IPv4 (32 bits), anteponiéndole 80 bits con valor cero y 16 bits con valor 1, para crear una dirección IPv6 "mapeada desde IPv4".

Algunas recomendaciones a tener en cuenta en los nodos duales son:

- No es conveniente registrar en el DNS las direcciones IPv6 de un sistema hasta que el mismo esté configurado y accesible mediante IPv6 ya que provoca retardos debidos a "timeouts" de TCP al intentar las conexiones IPv6.
- No es conveniente activar IPv6 en un sistema si no tiene conectividad IPv6 ya que produce retardos e incluso problemas de accesibilidad a servidores IPv6.

En este Trabajo Fin de Master crearemos un escenario con varios servidores configurados con Pila Dual de protocolos para probar tanto el funcionamiento de los mismos como la conectividad entre las distintas redes usando IPv4 e IPv6 indistintamente.

2.4.2 Túneles

Los túneles proporcionan un mecanismo para utilizar las infraestructuras IPv4 mientras la red IPv6 está siendo implantada.

La RFC 2893 define la utilización básica de túneles como mecanismo para transportar paquetes IPv6 sobre redes IPv4.

Los datagramas IPv6 se encapsulan sobre datagramas IPv4 para atravesar redes que aún no han sido migradas.

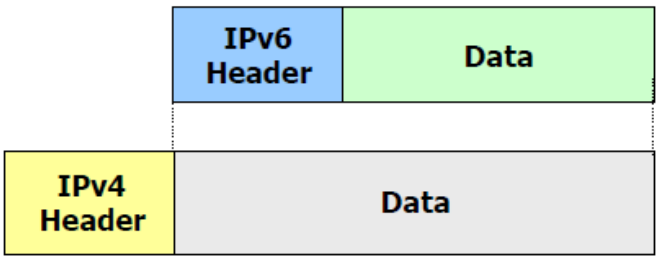

Figura 9. Encapsulación de datagramas

Los túneles se clasifican según el mecanismo por el que el nodo que realiza el encapsulado determina la dirección del nodo extremo del túnel.

- Router-to-router.

- Host-to-Router.
- Router-to-Host.
- Host-to-Host.

En los dos primeros casos (router to router y host to router), el paquete IPv6 es tunelizado a un router. El extremo final de este tipo de túnel, es un router intermedio que debe desencapsular el paquete IPv6 y reenviarlo a su destino final. En este caso, el extremo final del túnel es distinto del destino final del paquete, por lo que la dirección en el paquete IPv6 no proporciona la dirección IPv4 del extremo final del túnel. La dirección del extremo final del túnel ha de ser determinada a través de información de configuración en el nodo que realiza el túnel. Es lo que se denomina "túnel configurado", describiendo aquel tipo de túnel donde el extremo final del túnel es explícitamente configurado.

En los otros dos casos (host to host y router to host), el paquete IPv6 es tunelizado, durante todo el recorrido, a su nodo destino. El extremo final del túnel es el nodo destino del paquete, y por tanto, la dirección IPv4 está contenida en la dirección IPv6. Este caso se denomina "túnel automático".

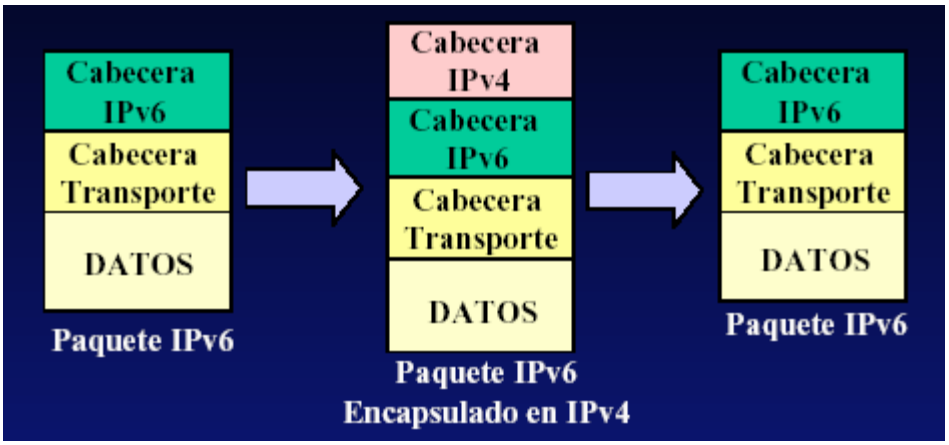

Figura 10. Túnel IPv6 encapsulado en IP4

2.4.3 Traductores de protocolos

Los traductores de protocolos se encargan de traducir datagramas IPv6 a IPv4 y viceversa. Permiten la comunicación entre sistemas sólo IPv4 y sistemas sólo IPv6. Traducen las cabeceras de los paquetes entre IPv4 e IPv6 (sólo los campos comunes).

El objetivo de las técnicas de traducción de protocolos es proveer de rutas transparentes a los nodos en las redes de IPv6 para comunicarlos con los nodos de redes IPv4 y viceversa.

Algunos mecanismos de traducción son NAT-PT, TCP-UDP Relay, Socks-based Gateway, Bump-in-the-Stack, Bump-in-the-API, etc...

El mecanismo de traducción que utilizaremos en este Trabajo Fin de Máster será NAT64. Se trata de un mecanismo para permitir a los clientes IPv6 comunicarse con servidores IPv4. El servidor NAT64 es el punto final de al menos una dirección IPv4 y un segmento de red de 32 bits IPv6.

El servidor NAT crea un mapeado NAT entre la IPv6 y la dirección IPv4 permitiendo su comunicación. Para ello se hace necesario el uso de sistemas DNS64.

Un sistema DNS64 se utiliza para compatibilizar ambos sistemas de resolución de nombres. Si el usuario utiliza IPv6, cuando acceda a una web desde el navegador solicitará un registro IPv6 (AAAA). Es decir, a qué IPv6 corresponde un nombre de dominio, en caso de que esa web todavía no se haya adaptado al nuevo protocolo, DNS64 hará una petición IPv4 automáticamente (registro A), a la que sí responderán los servidores DNS, para luego traducir esa dirección a IPv6 y pasarla al usuario. La traducción a una dirección IPv6 consiste en embeber la dirección IPv4 en la dirección IPv6 (por ejemplo 64:ff9b::/96, véase RFC 6052, RFC 6146). La dirección IPv6 destino se construye utilizando el rango anterior de 96 bits más los 32 bits de la dirección IPv4 con la que desea comunicarse, enviando los paquetes a la dirección resultante.

Por otro lado, NAT64 traduce las direcciones IPv6 del lado del usuario en IPv4 para acceder a Internet en caso de hacerse necesario. Utiliza las mismas tablas que el NAT tradicional para que desde una red privada se acceda a Internet, aunque con adaptación a IPv6. Después de resolver el nombre en un acceso web por DNS64, se hace la petición web a esa dirección IPv6 a través de NAT64, que ya se encargará de volver a traducir a IPv4 para acceder finalmente al servidor.

El principio de funcionamiento puede pensarse como un router con al menos dos interfaces, una de ellas está conectada a la red IPv4 y la otra a la IPv6. El router lleva a cabo las traducciones necesarias para transferir paquetes de IPv6 a IPv4 y viceversa (teniendo en cuenta que esta traducción no es simétrica ya que el espacio de direcciones de IPv6 es de 2^{128} frente a 2^{32} del IPv4).

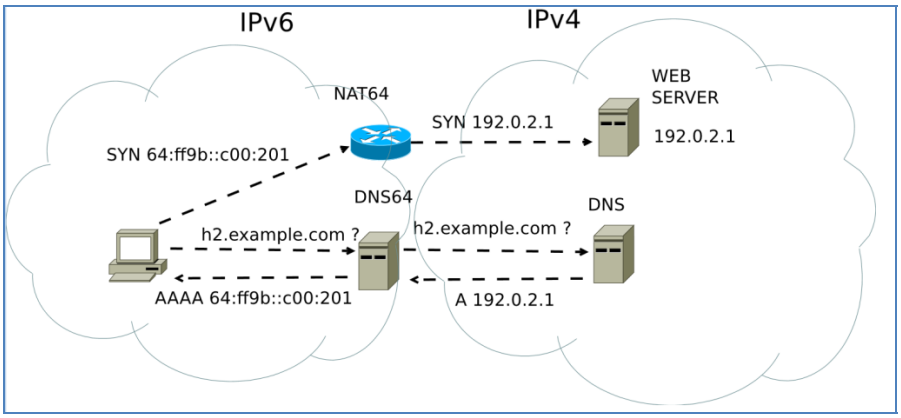

Figura 11. Funcionamiento del Traductor NAT64

3 Virtualización y VNX

La reducción de precios en el hardware y la presencia de productos software de virtualización cada vez más asequibles ha conseguido que la virtualización, antes reservada a grandes compañías, pueda hoy ser utilizada casi por cualquier persona u organización. Bien utilizada, la virtualización puede poner a nuestra disposición más opciones en el despliegue de sistemas, costes menores y un mayor control sobre nuestra estructura.

3.1 Virtualización

Desde un punto de vista genérico, puede definirse la virtualización como una técnica (normalmente con un software asociado) que permite encapsular una unidad de proceso (ya sea un programa, un sistema operativo o incluso un equipo completo, dependiendo de a qué profundidad se sitúe el nivel de virtualización) para su ejecución dentro de un entorno en un equipo anfitrión que emula el entorno real de forma trasparente.

Esto implica que, en disposición de una máquina lo suficientemente potente que actúe como equipo anfitrión, es posible ejecutar simultáneamente un sistema de "máquinas virtuales" que se comporten de forma equivalente al mismo sistema implementado con máquinas reales. El grado de similitud entre la implementación virtual del sistema y la implementación con equipos reales puede tomarse como una medida de la bondad de la técnica de virtualización empleada: lo deseable es que el sistema emulado se comporte lo más transparentemente posible, idealmente exactamente igual que el sistema real.

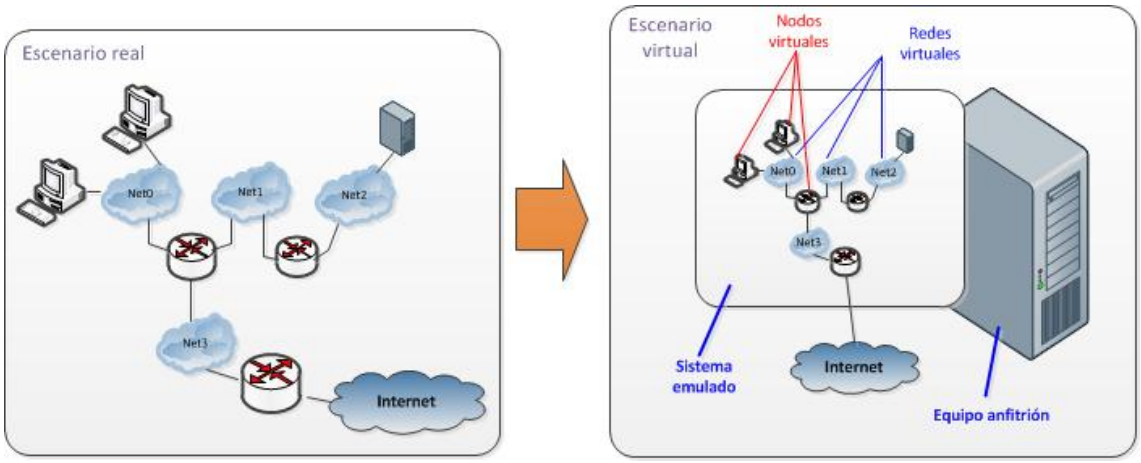

Figura 12. Sistema Real (izquierda) y sistema virtualizado (derecha)

Las ventajas de disponer de sistemas virtualizados frente a sistemas físicos son las siguientes:

- **Ahorro de costes**: Sólo es necesario adquirir una máquina física que funcione como máquina anfitriona, aunque más potente, y en ella ir creando sistemas en el gestor de máquinas virtuales. También permite ahorro en el coste de mantenimiento y en el de personal, además de ahorrar espacio.
- **Crecimiento más flexible**: Instalar un nuevo servidor es mucho más sencillo y rápido frente a hacerlo con un servidor físico.
- **Administración simplificada**: Desde la consola del gestor de máquinas virtuales podemos aumentar o reducir los recursos para una determinada máquina, reiniciarla, instalar parches o simplemente borrarla en caso de problemas.
- **Centralización de tareas de mantenimiento**: Podemos realizar copias de seguridad de todas las máquinas virtuales a la vez, programar actualizaciones y otras actividades desde el gestor de máquinas virtuales. También podemos centralizar otras funciones.
- **Disminución de los tiempos de parada**: Una ventaja importante, solucionar problemas o realizar copias de seguridad son tareas que se realizan en mucho menos tiempo. Por ejemplo, se puede clonar una máquina y seguir dando servicio mientras se realiza mantenimiento de la máquina virtual de producción como puede ser una actualización.
- **Mejor gestión de recursos**: Se puede aumentar la memoria o almacenamiento de la máquina huésped para aumentar los recursos de todas las máquinas virtuales a la vez, por lo que se aprovecha mucho mejor las inversiones en hardware.

- **Balanceo de recursos**: Es posible asignar un grupo de servidores físicos para que proporcionen recursos a las máquinas virtuales y asignar una aplicación que haga un balanceo de los mismos, otorgando más memoria, recursos de la CPU, almacenamiento o ancho de banda de la red a la máquina virtual que lo necesite.

Sin embargo, es difícil conseguir una transparencia completa usando virtualización. Esto se debe a que la virtualización introduce un nivel de proceso adicional (el que traduce las llamadas al sistema de la máquina virtual al sistema anfitrión) que supone un overhead debido al consumo de recursos. Para conseguir un rendimiento equivalente al del sistema real es necesario utilizar hardware de mayor potencia en la máquina anfitriona. En todo caso, esto es algo que depende de lo eficiente que sea la técnica de virtualización que se emplee.

Si bien la teoría en la que se basan las técnicas de virtualización modernas es bastante antigua [1], el auge de este tipo de soluciones no se ha producido hasta hace relativamente poco tiempo, motivadas principalmente por la baja relación *potencia del hardware/precio* a la que hemos llegado en nuestros días. Algunas de soluciones para ejecutar máquinas virtuales sobre un equipo anfitrión son: Xen [2], VMware [3], UML (User Mode Linux) [4].

Combinando las máquinas virtuales con el uso de redes virtuales emuladas en la máquina anfitrión es posible crear Escenarios de Red Virtuales incluso con conexiones externas. Algunas herramientas de gestión de máquinas virtuales son: GNS3 [5], Netkit [6], Marionnet [7], VNX/VNUML [8], etc.

3.2 VNX/VNMUL

VNX/VNUML es una herramienta de software libre formada por dos componentes: un lenguaje simple y descriptivo basado en XML (eXtended Markup Language [9]) que permite al usuario definir el sistema emulado en un fichero de texto; y un parser (escrito en Perl [10]) de ese lenguaje que se encarga de procesar el fichero, construir la simulación y gestionarla, ocultando los detalles complejos al usuario.

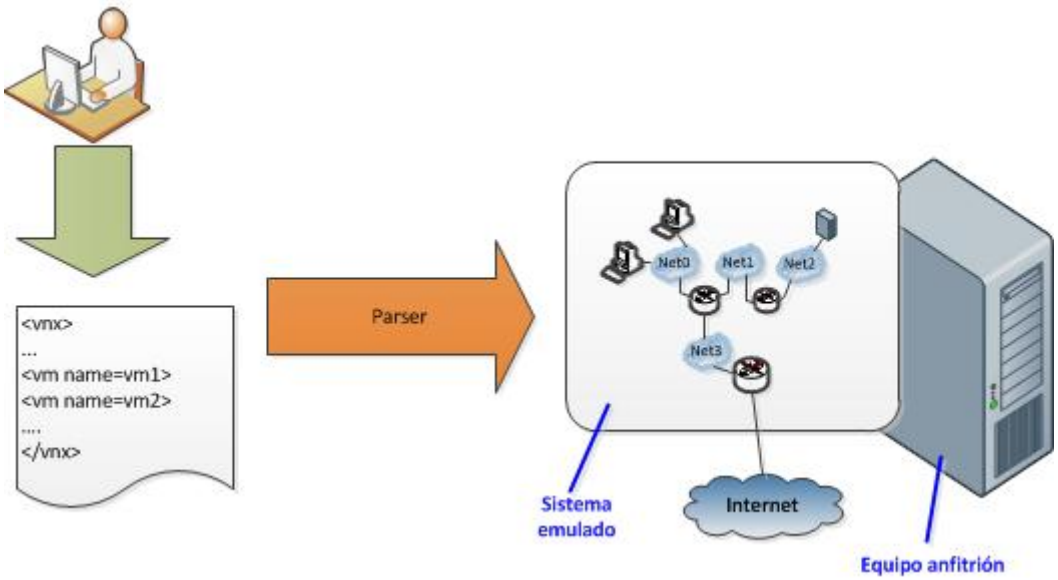

Figura 13. Funcionamiento básico de VNX

XML es el lenguaje base para la descripción de escenarios de simulación. Propuesto por el W3C en 1998 tras el auge del lenguaje HTML (Hypertext Markup Language) en la web, XML surge como un estándar abierto para el intercambio de información por Internet basado en etiquetas de texto. Comparado con otras alternativas, esta orientación a etiquetas hace que el procesado del lenguaje no sea muy eficiente, pero sí muy sencillo y muy intuitivo para el usuario, características por las cuales se ha elegido en VNX para representar las descripciones de escenario. XML tiene a su vez un conjunto de estándares asociados, de los que se hace uso en VNX: DTD (Document Type Definition), para la definición de la estructura de etiquetas; y DOM (Document Object Model), utilizado por el parser para procesar los datos de la simulación.

Como sistema de virtualización VNUML (Virtual Networks User Mode Linux) utiliza UML. UML es una modificación de las fuentes del núcleo de Linux [11] que permiten su ejecución como proceso de usuario encima del núcleo convencional de Linux (el que ejecuta la máquina anfitriona). Cada uno de los procesos UML tiene asociados sus propios recursos (espacio de memoria, procesos, sistemas de ficheros, dispositivos de red, etc.) y, en definitiva, constituye una máquina virtual dentro de la cual otros procesos se ejecutan. La funcionalidad del núcleo UML es exactamente la misma que la de un núcleo convencional de Linux (de hecho, el proceso de compilación es análogo), por lo que la transparencia es (salvo por la pérdida de rendimiento debido al overhead de virtualización) total.

UML no sólo proporciona la manera de crear máquinas virtuales, sino que existen mecanismos para la interconexión entre ellas mediante redes virtuales. Desde el punto de vista de la máquina virtual, se utiliza una interfaz de red de forma transparente. En

el entorno anfitrión, la red virtual interconecta los distintos terminales utilizando el módulo de túneles del núcleo. Es de destacar como la interconexión se produce en modo bridge, a nivel 2, y, a todos los efectos, es como si las máquinas interconectadas a una misma red virtual estuvieran físicamente conectadas a un mismo segmento. También es posible que la máquina anfitriona intervenga en la simulación o interconexión de máquinas virtuales con equipos externos a través del interfaz físico (lo cual, es importante a la hora de integrar sistemas no-Linux en la simulación), de forma completamente transparente o a través de VLANs (Virtual Local Area Network [12]).

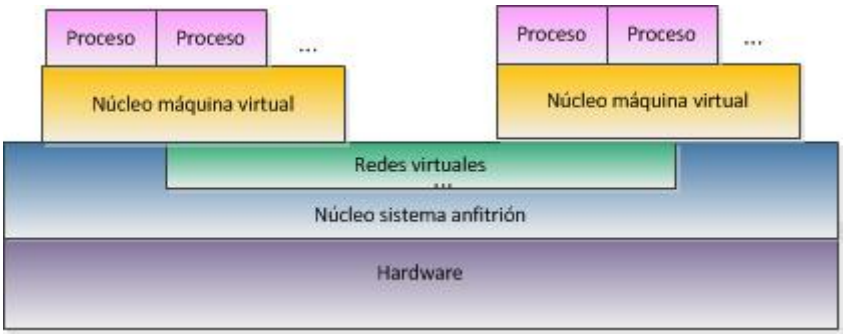

Figura 14. Virtualización con VNX

VNX/VNUML también proporciona facilidades en la gestión de las máquinas virtuales. En todas ellas existe una interfaz de red reservada, que permite una conexión directa con la máquina anfitriona, con el objetivo de realizar operaciones de gestión a través de la misma (por ejemplo, a través de SSH) [13]. Existen otros mecanismos de gestión de las máquinas virtuales, como el uso de consolas directas en las mismas (por ejemplo, un xterm) o a bajo nivel a través de un socket UNIX que comunica directamente con el núcleo UML.

VNX se distribuye con licencia libre GPL y con vocación de herramienta de uso público para investigadores y docentes. Está basada en VNUML del que mantiene muchas características e incorpora algunas nuevas funcionalidades como:

- Integración de libvirt (acceso estándar a virtualización de Linux).
- Autoconfiguración para Windows XP, Windows 7, Linux (Ubuntu 9.10/10.04/11.04), FreeBSD (8.1) y Fedora 14.
- Integración Dynamips (CISCO).
- Integración Olive (Juniper).
- Funcionalidad de gestión individual de máquinas.

3.3 Funcionamiento de VNX

El funcionamiento de VNX/VNUML se refleja en los pasos indicados en la siguiente figura:

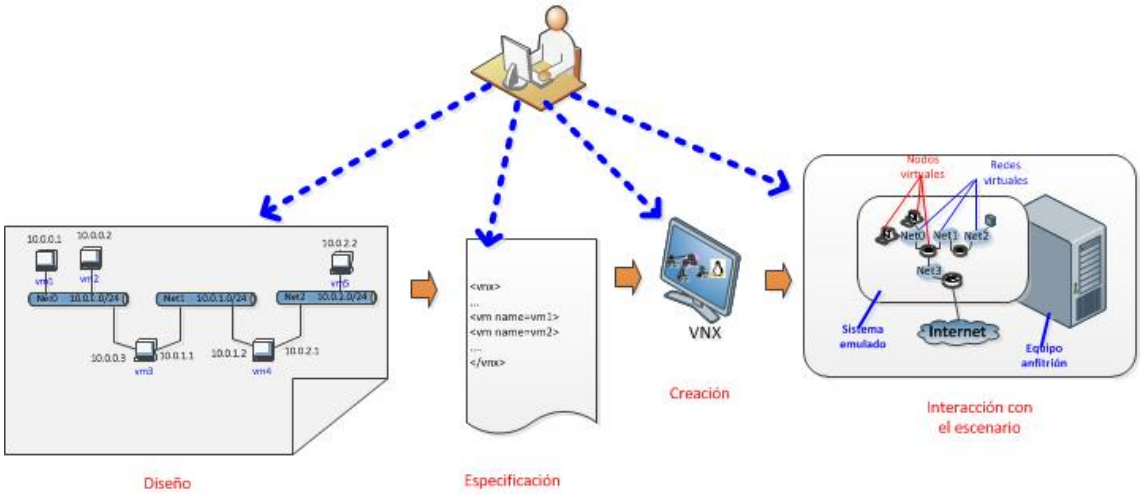

Figura 15. Funcionamiento de VNX

En primer lugar, el usuario diseña el escenario que pretende simular, de manera off-line. A continuación especifica el escenario en lenguaje VNX/VNUML usando XML. Una vez escrito, VNX procesa la especificación y crea el escenario virtual que ya podemos arrancar. En este escenario podemos ejecutar comandos en las máquinas virtuales a través de VNX o directamente sobre las mismas.

Podemos concluir que la utilización de técnicas de virtualización y, en particular, herramientas de virtualización de escenarios como VNX en infraestructuras de experimentación permite:

- Un ahorro de costes, tanto de equipamiento como esfuerzo invertido en su gestión y configuración.
- Mejor compartición de infraestructuras de experimentación.
- Crear y reutilizar escenarios complejos mediante un esfuerzo razonable.
- Concentrar el esfuerzo en el sistema o servicio a probar y no en la infraestructura de pruebas.

4 Migración a IPv6 en una escuela de la UPM (EIAE)

4.1 Introducción

Son muchas las motivaciones que pueden dar lugar a implantar el protocolo IPv6 en las redes de la EIAE. Anteriormente se han descrito algunas de las ventajas técnicas que ofrece IPv6.

La implantación del protocolo IPv6 se debe, además de las ventajas técnicas, a la demanda producida por parte del personal investigador de la EIAE como soporte para muchos de los proyectos de investigación en los que trabajan.

Pero la migración es un proceso largo y costoso que comienza con la implantación del protocolo de red en la infraestructura de comunicaciones de la EIAE, continuando con la modificación de los servicios ofrecidos actualmente en ella y finalizará con la instalación del protocolo en todos los dispositivos conectados a su infraestructura de comunicaciones.

La implantación del protocolo de red IPv6 lleva consigo una serie de requisitos que deben cumplirse para poderse llevar a cabo de forma eficaz la migración al mismo. Entre los muchos requisitos, podemos destacar los siguientes:

- Acceso de todos los usuarios a la nueva red.
- Acceso a los servicios de información de IPv6.
- Documentación y apoyo técnico a los usuarios.
- Servicio de gestión del direccionamiento IPv6.
- Servicio de gestión de la seguridad.

Todo esto teniendo en cuenta de que la red actual no debe sufrir durante el proceso de migración.

4.2 Modelo de direccionamiento

El modelo de direccionamiento se encarga de realizar una asignación de direcciones IPv6 a cada una de las redes que actualmente existen en la EIAE para dotarlas de conectividad IPv6. El modelo de direccionamiento también tiene en cuenta las posibles necesidades que puede haber en un futuro dentro de la EIAE [14].

El documento RIPE-267 define la política de asignación de direcciones en Europa. En dicho documento, propone asignar un prefijo /32 a los proveedores de servicio de Internet que en menos de dos años asignen a sus usuarios 200 prefijos /48.

Dado que el proveedor de servicios de la UPM es RedIRIS y, de acuerdo con las anteriores recomendaciones, RedIRIS tiene delegado por RIPE el prefijo **2001:0720::/32**.

De acuerdo con el plan de asignación de direcciones establecido por RedIRIS, a cada Universidad se le asigna un prefijo /48. De acuerdo con esto, el prefijo que RedIRIS ha asignado a la UPM es **2001:0720:041c::/48**.

Dado que RedIRIS ha delegado a UPM el prefijo **2001:0720:041c::/48**, esto permite utilizar 16 bits para la Universidad, esto hace un total de 65.536 redes posibles de las que ha asignado a la EIAE el prefijo **2001:0720:041c:4000::/54**.

La EIAE está compuesta por personal docente e investigador, personal de administración y servicios y alumnos organizados en departamentos o adscritos a Servicios Centrales. Las necesidades de direcciones no son iguales en todos los departamentos de la EIAE. Por este motivo es necesario dividir las direcciones IPv6 entre los departamentos según sus necesidades. Así, podemos clasificarlos en tres tipos:

- Un **departamento de tipo 1** es aquel que bien por su mayor número de personal adscrito al mismo, o bien por su número de sistemas, tiene más necesidades de direcciones IPv6. Dentro de este tipo se incluye los Servicios Centrales.
- Un **departamento de tipo 2** es aquel que tiene menos necesidades de direcciones IPv6 que un departamento de tipo 1 pero más que uno de tipo 3.
- Un **departamento de tipo 3** es aquel que menos necesidades de direcciones IPv6. Dentro de este tipo se pueden incluir los proyectos de investigación desarrollados en la EIAE y las cátedras Universidad-Empresa.

De acuerdo con esto y considerando la estructura departamental de la EIAE, en la que la mayor parte de los departamentos son pequeños o no disponen de un gran número de máquinas, el plan de direccionamiento podría ser el representado en la figura 4:

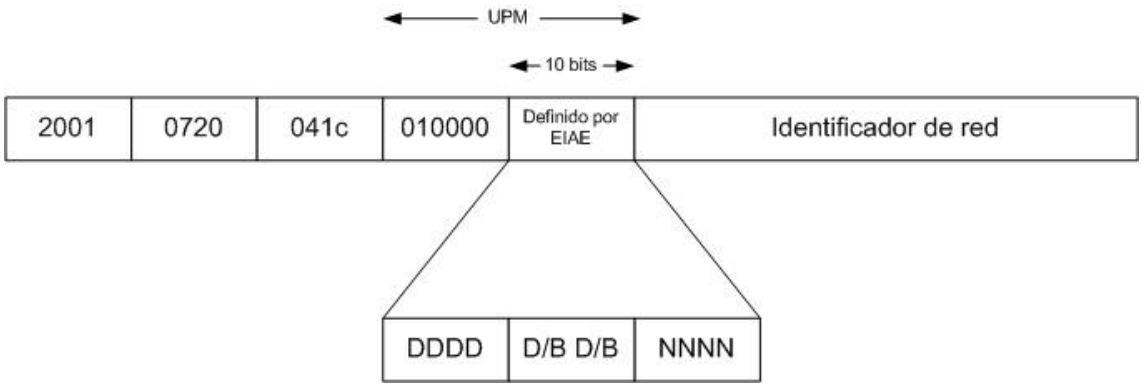

Figura 16. Plan de direccionamiento IPv6 en la EIAE

Donde:

- **D**: representa al identificador de departamento.
- **B**: Representa al identificador de bloque.
- **N:** Representa el identificador de red (dentro de una dirección de bloque asignado a un departamento).

De acuerdo con este plan de direccionamiento, habría:

- Hasta un máximo de 4 departamentos del tipo 1 a los que se les asignará inicialmente dos bloques de 16 redes, quedando otros dos bloques de reserva para futuros usos.
- Hasta un máximo de 16 departamentos del tipo 2 a los que se les asignará inicialmente un bloque de 16 redes, quedando otro bloque de reserva para futuros usos.
- Hasta un máximo de 16 departamentos del tipo 3 a los que se les asignará un bloque de 16 redes.

Tipo de Departamento	Formato de direcciones	Nº Bloques asignados	Nª de bloques de Reserva	Longitud del prefijo
1	DDDDBBNNNN	2	2	/58
2	DDDDDBNNNN	1	1	/59
3	DDDDDDNNNN	1	0	/60

Tabla 1. Reparto por tipos de departamentos en la EIAE

Un posible reparto de las direcciones IPv6 dentro de la EIAE podría ser el siguiente:

Tipo de Departamento	Nombre Departamento	Formato de direcciones	Nº Bloques asignados	Nª de bloques de Reserva	Longitud del prefijo	Ejemplo
1	Departamento-1	0000BBNNNN	2	2	/58	2001:720:41c:4000::/58
1	Departamento-2	0001BBNNNN	2	2	/58	2001:720:41c:4040::/58
1	Departamento-3	0010BBNNNN	2	2	/58	2001:720:41c:4080::/58
1	Departamento-4	0011BBNNNN	2	2	/58	2001:720:41c:40c0::/58
2	Departamento-5	01000BNNNN	1	1	/59	2001:720:41c:4100::/59
2	Departamento-6	01001BNNNN	1	1	/59	2001:720:41c:4120::/59
2	Departamento-7	01010BNNNN	1	1	/59	2001:720:41c:4140::/59
2	Departamento-8	01011BNNNN	1	1	/59	2001:720:41c:4160::/59
2	Departamento-9	01100BNNNN	1	1	/59	2001:720:41c:4180::/59
2	Departamento-10	01101BNNNN	1	1	/59	2001:720:41c:41a0::/59
2	Departamento-11	01110BNNNN	1	1	/59	2001:720:41c:41c0::/59
2	Departamento-12	01111BNNNN	1	1	/59	2001:720:41c:41e0::/59
2	Departamento-13	10000BNNNN	1	1	/59	2001:720:41c:4200::/59
2	Departamento-14	10001BNNNN	1	1	/59	2001:720:41c:4220::/59
2	Departamento-15	10010BNNNN	1	1	/59	2001:720:41c:4240::/59
2	Departamento-16	10011BNNNN	1	1	/59	2001:720:41c:4260::/59
2	Departamento-17	10100BNNNN	1	1	/59	2001:720:41c:4280::/59
2	Departamento-18	10101BNNNN	1	1	/59	2001:720:41c:42a0::/59

2	Departamento-19	10110BNNNN	1	1	/59	2001:720:41c:42c0::/59
2	Departamento-20	10111BNNNN	1	1	/59	2001:720:41c:42e0::/59
3	Departamento-21	110000NNNN	1	0	/60	2001:720:41c:4300::/60
3	Departamento-22	110001NNNN	1	0	/60	2001:720:41c:4310::/60
3	Departamento-23	110010NNNN	1	0	/60	2001:720:41c:4320::/60
3	Departamento-24	110011NNNN	1	0	/60	2001:720:41c:4330::/60
3	Departamento-25	110100NNNN	1	0	/60	2001:720:41c:4340::/60
3	Departamento-26	110101NNNN	1	0	/60	2001:720:41c:4350::/60
3	Departamento-27	110110NNNN	1	0	/60	2001:720:41c:4360::/60
3	Departamento-28	110111NNNN	1	0	/60	2001:720:41c:4370::/60
3	Departamento-29	111000NNNN	1	0	/60	2001:720:41c:4380::/60
3	Departamento-30	111001NNNN	1	0	/60	2001:720:41c:4390::/60
3	Departamento-31	111010NNNN	1	0	/60	2001:720:41c:43a0::/60
3	Departamento-32	111011NNNN	1	0	/60	2001:720:41c:43b0::/60
3	Departamento-33	111100NNNN	1	0	/60	2001:720:41c:43c0::/60
3	Departamento-34	111101NNNN	1	0	/60	2001:720:41c:43d0::/60
3	Departamento-35	111110NNNN	1	0	/60	2001:720:41c:43e0::/60
3	Departamento-36	111111NNNN	1	0	/60	2001:720:41c:43f0::/60

Tabla 2. Reparto de prefijos en la EIAE por departamentos

4.3 Estrategia de transición

Una vez que se ha definido el plan de direccionamiento IPv6 en la EIAE se debe comenzar con la transición a la nueva red.

La transición es compleja ya que se deben actualizar routers, hosts, aplicaciones, gestión y operación, formación del personal, etc. Además es un proceso largo ya que durante la migración de IPv4 a IPv6 deben coexistir ambos sistemas por un tiempo que puede llegar a ser indefinido. Como ya hemos comentado la clave para la transición es la compatibilidad con la base instalada de dispositivos IPv4 lo que supone un conjunto de mecanismos que los hosts y routers IPv6 pueden implementar para ser compatibles con host y routers IPv4.

Las consideraciones a tener en cuenta desde el punto de vista de la transición a IPv6 son las siguientes:

- Todos los dispositivos actualmente conectados a la red de la EIAE dispondrán de doble pila IPv4 – IPv6. Esta técnica es una solución más fácil de implementar en entornos corporativos como el que queremos tratar.

  Ambos protocolos convivirán en el mismo segmento de red (dual-stack network). Esto implica que los dispositivos de red, independientemente de que utilicen IPv4 o IPv6 a nivel de red, comparten la misma red física y por tanto, el mismo dominio de broadcast y la misma vlan. Esto supone que los sistemas operativos de los equipos conectados a las redes van a utilizar ambas pilas de protocolos en paralelo.

  Para configurar la dirección IPv6 de los dispositivos conectados a la red de la EIAE se utilizarán métodos de autoconfiguración. En el presente trabajo hemos utilizado el método de autoconfiguración stateful utilizando para ello un servidor DHCPv6 para, además de ofrecer la dirección IPv6, ofrecer otras informaciones como el servidor dns con el que tiene que configurarse, el dominio al que pertenecen, etc.

- Aunque actualmente no es necesario debido a que todos los sistemas conectados a la red de la EIAE disponen de direccionamiento IPv4 público, y una vez desplegado el direccionamiento IPv6, todos los sistemas dispondrían de doble pila, se va a implementar un método de translación de direcciones llamado NAT64.

La estrategia a seguir en el despliegue de IPv6 en la EIAE es la siguiente:

1. Poner la doble pila IPv4-IPv6 en los routers y en los servidores.
2. Configurar los servicios para IPv6.
3. Extender la doble pila a los usuarios finales.
4. Eliminar IPv4 en los sistemas de los usuarios finales cuando todos tengan instalado IPv6.
5. Eliminar IPv4 de los routers y de los servidores.

5 Prototipos

5.1 Simulador

Como hemos comentado en apartados anteriores el simulador elegido para realizar este Trabajo Fin de Master es VNX/VNUML. En el apartado 3 hemos descrito su funcionamiento básico. Vamos a describir brevemente un poco del lenguaje que utiliza y que usaremos para describir nuestro escenario:

La sintaxis básica es la siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<vnx>
<global>
<version>1.92</version>
<scenario_name>Escenario-Proyecto</scenario_name>
</global>
<net name="Net0" mode="virtual_bridge" external="eth0"/>
<net name="Net1" mode="uml_bridge" />
<net name="Net2" mode="uml_bridge" />
<net name="Net3" mode="uml_bridge" />
<vm> … </vm>
<vm>…  </vm>
…
</vnx>
```

Toda la definición del escenario debe estar englobada entre las etiquetas <vnx> y </vnx>. Las definiciones globales van entre las etiquetas <global> y </global>. Las redes se definen con la etiqueta <net> y las máquinas virtuales se definen entre las

etiquetas <vm> y </vm>. Dentro de estas etiquetas se definen los parámetros de cada una de las máquinas virtuales:

```
<vm name="cliente-xp1" type="libvirt" subtype="kvm" os="windows">
<filesystem type="cow">/usr/share/vnx/filesystems/root_fs_winxp</filesystem>
<mem>256M</mem>
<if id="1" net="Net2">
<ipv4>10.0.2.2/24</ipv4>
</if>
<route type="ipv4" gw="10.0.2.1">10.0.2.1/24</route>
</vm>
<vm name="srv-www" type="libvirt" subtype="kvm" os="linux">
<filesystem type="cow">/usr/share/vnx/filesystems/root_fs_ubuntu</filesystem>
<mem>256M</mem>
<if id="1" net="Net3">
<ipv4>10.0.3.2/24</ipv4>
</if>
<route type="ipv4" gw="10.0.0.1">10.0.0.1/24</route>
</vm>
```

La etiqueta <filesystem> define el sistema de ficheros, la etiqueta <mem> define la cantidad de memoria asignada a la máquina virtual, la etiqueta <if> nos permite configurar una dirección IP y la etiqueta <route> poner una ruta por defecto. A modo de ejemplo algunos de los comandos más usados en VNX son:

- Arranque del escenario:

```
# vnx –f escenario.xml --create
```

- Acceso a consolas:

```
# vnx –f escenario.xml --console –M vm1
```

- Ejecución de comandos:

```
# vnx –f escenario.xml –execute start
```

- Re-arranque de una máquina virtual:

```
# vnx –f escenario.xml --reboot –M vm1
```

- Parada del escenario:

```
# vnx –f escenario.xml --shutdown
```

```
# vnx –f escenario.xml --destroy
```

5.2 Descripción del escenario

El escenario creado pretende simular la red y los posibles servicios ofrecidos por una escuela de la UPM y probar sobre ella diferentes servicios tanto con IPv4 como con IPv6.

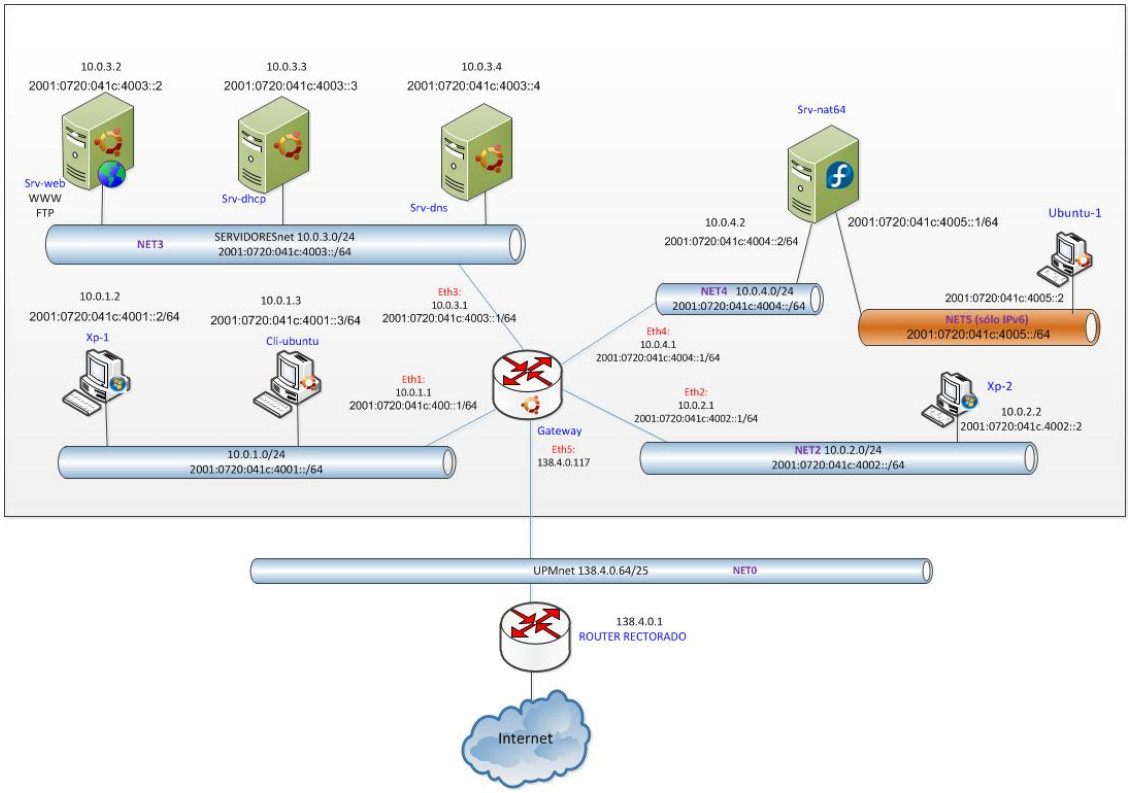

Figura 17. Escenario de simulación

Los sistemas utilizados en esta simulación son los siguientes:

- **Gateway**: es el router principal de la escuela.
- **Srv-web**: es el servidor web y ftp de la escuela.
- **Srv-dns**: es el servidor dns de la escuela.
- **Srv-dhcp**: es el servidor dhcp de la escuela.
- **Srv-nat64**: es el servidor NAT64 de la escuela.
- **Xp-1**: es un cliente con sistema operativo Windows XP.
- **Cli-ubuntu**: es un cliente con sistema operativo Ubuntu.
- **Xp-2**: es un cliente con sistema operativo Windows XP.
- **Ubuntu-1**: es un cliente con sistema operativo Ubuntu.

Todos los equipos están conectados a las siguientes redes:

Nombre	Dirección IPv4	Dirección IPv6	Descripción
NET0	192.168.1.0/24	-	Red de conexión a la red de UPM
NET1	10.0.1.0/24	2001:720:41c:4001::/64	Red de Clientes
NET2	10.0.2.0/24	2001:720:41c:4002::/64	Red de Clientes
NET3	10.0.3.0/24	2001:720:41c:4003::/64	Red de Servidores
NET4	10.0.4.0/24	2001:720:41c:4004::/64	Red para la translación IPv4 – Ipv6
NET5	-	2001:720:41c:4005::/64	Red con sistemas configurados sólo con IPv6

Tabla 3. Redes utilizadas en la simulación

Las características de todos los sistemas utilizados en la virtualización es la siguiente:

Nombre	Dirección IPv4	Dirección IPv6	Sistema Operativo
Gateway	192.168.1.2 10.0.1.1 10.0.2.1 10.0.3.1 10.0.4.1	2001:720:41c:4001::1 2001:720:41c:4002::1 2001:720:41c:4003::1 2001:720:41c:4004::1	Ubuntu 10.10
Srv-web	10.0.3.2	2001:720:41c:4003::2	Ubuntu 10.10
Srv-dhcp	10.0.3.3	2001:720:41c:4003::3	Ubuntu 10.10
Srv-dns	10.0.3.4	2001:720:41c:4003::4	Ubuntu 10.10

Srv-nat64	10.0.4.2	2001:720:41c:4004::2 2001:720:41c:4005::1	Fedora 14
Xp-1	10.0.1.2	2001:720:41c:4001::2	Windows XP SP3
Xp-2	10.0.2.2	2001:720:41c:4002::2	Windows XP SP3
Cli-ubuntu	10.0.1.3	2001:720:41c:4001::3	Ubuntu 10.10
Ubuntu-1	-	2001:720:41c:4005::2	Ubuntu 10.10

Tabla 4. Sistemas utilizados en la simulación

El fichero XML que permite crear este escenario es el siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
Trabajo Fin de Master
AUTORES: Rafael García García
   Gloria Martín Martín
-->
<vnx xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="/usr/share/xml/vnx/vnx-1.95.xsd">
 <global>
  <version>1.92</version>
  <scenario_name>Escenario</scenario_name>
  <automac/>
  <vm_mgmt type="none" />
  <vm_defaults exec_mode="mconsole">
 <console id="0" display="yes"/>
     <console id="1" display="no"/>
  </vm_defaults>
 </global>

 <net name="Net0" mode="virtual_bridge" external="eth0"/>
 <net name="Net1" mode="virtual_bridge" />
 <net name="Net2" mode="virtual_bridge" />
 <net name="Net3" mode="virtual_bridge"  />
 <net name="Net4" mode="virtual_bridge"  />
```

```
    <net name="Net5" mode="virtual_bridge"  />
    <!-- NODOS -->

     <vm name="Xp-1" type="libvirt" subtype="kvm" os="windows">
       <filesystem type="cow">/usr/share/vnx/filesystems/root_fs_winxp</filesystem>
       <mem>256M</mem>
       <if id="1" net="Net1">
       </if>
     </vm>

      <vm name="Cli-ubuntu" type="libvirt" subtype="kvm" os="linux">
       <filesystem type="cow">/usr/share/vnx/filesystems/root_fs_ubuntu</filesystem>
       <mem>256M</mem>
       <if id="1" net="Net1">
       </if>
   <filetree seq="redes" root="/root/cli-ubuntu">/root/cli-ubuntu/</filetree>
   <exec seq="red" type="verbatim" mode="system">cp /root/cli-ubuntu/interfaces.
/etc/network/interfaces</exec>
   <exec seq="reinicio-red" type="verbatim" mode="system">/etc/init.d/networking restart</exec>
     </vm>

      <vm name="Ubuntu-1" type="libvirt" subtype="kvm" os="linux">
       <filesystem type="cow">/usr/share/vnx/filesystems/root_fs_ubuntu</filesystem>
       <mem>256M</mem>
       <if id="1" net="Net5">
        <ipv6>2001:0720:41c:4005::2/64</ipv6>
       </if>
              <route type="ipv6" gw="2001:0720:41c:4005::1">2001:0720:41c:4005::1/64</route>
       <filetree seq="redes" root="/root/ubuntu-1">/root/ubuntu-1/</filetree>
   <exec seq="red" type="verbatim" mode="system">cp /root/ubuntu-1/interfaces.
/etc/network/interfaces</exec>
   <exec seq="reinicio-red" type="verbatim" mode="system">/etc/init.d/networking restart</exec>
         <exec seq="rutadefecto" type="verbatim" mode="system">route -A inet6 add default gw
2001:720:41c:4005::1</exec>
     </vm>

     <vm name="Xp-2" type="libvirt" subtype="kvm" os="windows">
       <filesystem type="cow">/usr/share/vnx/filesystems/root_fs_winxp</filesystem>
       <mem>256M</mem>
       <if id="1" net="Net2">
```

```
        </if>
      </vm>

      <vm name="Srv-web" type="libvirt" subtype="kvm" os="linux">
        <filesystem
type="direct">/usr/share/vnx/filesystems/root_fs_ubuntu_www.qcow2</filesystem>
        <mem>256M</mem>
        <if id="1" net="Net3">
         <ipv4>10.0.3.2/24</ipv4>
         <ipv6>2001:0720:41c:4003::2/64</ipv6>
        </if>
        <route type="ipv4" gw="10.0.3.1">10.0.3.1/24</route>
                <route type="ipv6" gw="2001:0720:41c:4003::1">2001:0720:41c:4003::1/64</route>

          <filetree seq="apache" root="/root/apache">/root/apache/</filetree>
          <filetree seq="paginas" root="/root/ipv6">/root/ipv6/</filetree>
          <filetree seq="ftp" root="/root/ftp">/root/ftp/</filetree>
          <exec seq="copia-1" type="verbatim" mode="system">cp /root/apache/default.
/etc/apache2/sites-available/default</exec>
          <exec seq="copia-2" type="verbatim" mode="system">cp /root/apache/ports.conf
/etc/apache2/ports.conf</exec>
          <exec seq="copia-3" type="verbatim" mode="system">cp /root/apache/hosts.
/etc/hosts</exec>
          <exec seq="copia-4" type="verbatim" mode="system">cp /root/apache/default_ssl.
/etc/apache2/sites-available/default-ssl</exec>
        <exec seq="copia-5" type="verbatim" mode="system">a2enmod ssl</exec>
        <exec seq="copia-6" type="verbatim" mode="system">a2ensite default-ssl</exec>
          <exec seq="copia-7" type="verbatim" mode="system">cp -a /root/ipv6/
/var/www/</exec>
          <exec seq="copia-8" type="verbatim" mode="system">chmod -R o+rx
/var/www/ipv6</exec>
          <exec seq="copia-9" type="verbatim" mode="system">/etc/init.d/apache2 restart</exec>

    <exec seq="copia-ftp" type="verbatim" mode="system">cp /root/ftp/vsftpd.conf
/etc/vsftpd.conf</exec>
    <exec seq="reinicio-ftp" type="verbatim" mode="system">/etc/init.d/vsftpd restart</exec>

        <exec seq="dns" type="verbatim" mode="system">echo "nameserver 10.0.3.4" >
/etc/resolv.conf</exec>
        <exec seq="rutadefecto" type="verbatim" mode="system">route -A inet6 add default gw
2001:720:41c:4003::1</exec>
```

```
    </vm>

    <vm name="Srv-dhcp" type="libvirt" subtype="kvm" os="linux">
      <filesystem
type="direct">/usr/share/vnx/filesystems/root_fs_ubuntu_dhcp.qcow2</filesystem>
      <mem>256M</mem>
      <if id="1" net="Net3">
       <ipv4>10.0.3.3/24</ipv4>
       <ipv6>2001:0720:41c:4003::3/64</ipv6>
      </if>
      <route type="ipv4" gw="10.0.3.1">10.0.3.1/24</route>
            <route type="ipv6" gw="2001:0720:41c:4003::1">2001:0720:41c:4003::1/64</route>
        <exec seq="dns" type="verbatim" mode="system">echo "nameserver 10.0.3.4" >
/etc/resolv.conf</exec>
      <exec seq="rutadefecto" type="verbatim" mode="system">route -A inet6 add default gw
2001:720:41c:4003::1</exec>
    </vm>

    <vm name="Srv-dns" type="libvirt" subtype="kvm" os="linux">
      <filesystem
type="direct">/usr/share/vnx/filesystems/root_fs_ubuntu_dns.qcow2</filesystem>
      <mem>256M</mem>
      <if id="1" net="Net3">
       <ipv4>10.0.3.4/24</ipv4>
       <ipv6>2001:0720:41c:4003::4/64</ipv6>
      </if>
      <route type="ipv4" gw="10.0.3.1">10.0.3.1/24</route>
            <route type="ipv6" gw="2001:0720:41c:4003::1">2001:0720:41c:4003::1/64</route>
        <exec seq="dns" type="verbatim" mode="system">echo "nameserver 10.0.3.4" >
/etc/resolv.conf</exec>
      <exec seq="rutadefecto" type="verbatim" mode="system">route -A inet6 add default gw
2001:720:41c:4003::1</exec>
    </vm>

    <vm name="Srv-nat64" type="libvirt" subtype="kvm" os="linux">
      <filesystem type="direct">/usr/share/vnx/filesystems/root_fs_fedora-14-v01-
nat64.qcow2</filesystem>
      <mem>256M</mem>
      <if id="1" net="Net4">
       <ipv4>10.0.4.2/24</ipv4>
```

```
          <ipv6>2001:0720:41c:4004::2/64</ipv6>
        </if>
        <if id="2" net="Net5">
          <ipv6>2001:0720:41c:4005::1/64</ipv6>
        </if>
        <route type="ipv4" gw="10.0.4.1">10.0.4.1/24</route>
                <route type="ipv6" gw="2001:0720:41c:4004::1">2001:0720:41c:4004::1/64</route>
          <exec seq="dns" type="verbatim" mode="system">echo "nameserver 10.0.4.2" > /etc/resolv.conf</exec>
    <exec seq="rutaipv4" type="verbatim" mode="system">route add -net 0.0.0.0 netmask 0.0.0.0 gw 10.0.4.1</exec>
    <exec seq="rutaipv6" type="verbatim" mode="system">route -A inet6 add ::0/0 gw 2001:720:41c:4004::1</exec>
    <exec seq="lanzadns64" type="verbatim" mode="system">/root/nat64/ecdysis-bind-9.7.2-P2D20101117/bin/named/named -c /etc/named.conf</exec>
    <exec seq="lanzanat64" type="verbatim" mode="system">/root/nat64/ecdysis-nf-nat64-20101117/nat64-config.sh</exec>
      </vm>

      <vm name="Gateway" type="libvirt" subtype="kvm" os="linux">
        <filesystem type="direct">/usr/share/vnx/filesystems/root_fs_ubuntu_router.qcow2</filesystem>
        <mem>256M</mem>
        <if id="1" net="Net1">
          <ipv4>10.0.1.1/24</ipv4>
          <ipv6>2001:0720:41c:4001::1/64</ipv6>
        </if>
        <if id="2" net="Net2">
          <ipv4>10.0.2.1/24</ipv4>
          <ipv6>2001:0720:41c:4002::1/64</ipv6>
        </if>
        <if id="3" net="Net3">
          <ipv4>10.0.3.1/24</ipv4>
          <ipv6>2001:0720:41c:4003::1/64</ipv6>
        </if>
        <if id="4" net="Net4">
          <ipv4>10.0.4.1/24</ipv4>
          <ipv6>2001:0720:41c:4004::1/64</ipv6>
        </if>
      <if id="5" net="Net0">
```

```
          <ipv4 mask="255.255.255.128">138.4.0.117</ipv4>
          </if>
          <route type="ipv4" gw="138.4.0.1">default</route>

            <forwarding type="ip" />
        <exec seq="dns" type="verbatim" mode="system">echo "nameserver 138.4.2.10" > /etc/resolv.conf</exec>
    <exec seq="salida" type="verbatim" mode="system">iptables -t nat -A POSTROUTING -o eth5 -j MASQUERADE</exec>
    <exec seq="ruta1" type="verbatim" mode="system">route -A inet6 add 2001:720:41c:4005::/64 gw 2001:720:41c:4004::2 dev eth4</exec>
    <exec seq="ruta2" type="verbatim" mode="system">route -A inet6 add 64:FF9B::/96 gw 2001:720:41c:4004::2 dev eth4</exec>
        <exec seq="arranquerelay" type="verbatim" mode="system">/etc/init.d/dhcp3-relay start</exec>
        <exec seq="arranqueradvd" type="verbatim" mode="system">/etc/init.d/radvd start</exec>
       </vm>

    <host>
        <hostif net="Net0">
          <ipv4 mask="255.255.255.128">138.4.0.116</ipv4>
        </hostif>
        <physicalif name="eth0" type="ipv4" ip="138.4.0.1" mask="255.255.255.128" gw="138.4.0.1"/>
        <route type="ipv4" gw="138.4.0.1">default</route>
      </host>

    </vnx>
```

Los servicios que se prueban en este escenario, tanto para IPv4 como para IPv6, son los siguientes:

- Configuración del protocolo.
- Encaminaniento.
- Servicio DHCP.
- Firewall.
- Servicio DNS.
- Servicio FTP.
- Servicio WEB.
- Servicio NAT64.

En la presente memoria se describen la instalación, configuración y pruebas realizadas para comprobar los siguientes servicios:

- Servicio DNS.
- Servicio FTP.
- Servicio WEB.
- Servicio NAT64.

Mientras que la configuración del protocolo, el encaminamiento, el Servicio DHCP y el Firewall se describen en la memoria realizada por Rafael García García titulada "Diseño de escenarios de transición a IPv6 utilizando la herramienta VNX: Encaminamiento y asignación de direcciones"

5.3 Servicios

5.3.1 DNS

Introducción

DNS es la abreviatura para Sistema de Nombres de Dominio (Domain Name System), un método para asignar nombres a equipos y servicios de red que se organiza en una jerarquía de nombres de dominio. El Servicio de Nombres de Dominio se utiliza en las redes TCP/IP, como Internet, para asociar equipos y servicios con nombres sencillos. Cuando un usuario escribe un nombre DNS en una aplicación, los servicios DNS traducirán el nombre a otra información asociada con el mismo, como una dirección IP.

BIND es el servidor de DNS de mayor popularidad en la actualidad. Ha sido portado para múltiples plataformas basadas en Unix (incluyendo a Linux) así como para Windows.

Las versiones de BIND han evolucionado continuamente tal y como lo ha hecho el DNS en sí y hoy las versiones modernas de BIND soportan IPv6.

Cada distribución de Linux proporciona diferentes paquetes. En el caso de la distribución Ubuntu, el servidor BIND se empaqueta bajo el nombre bind.

Instalación

Este servicio se instala en el servidor "**Srv-dns**" y para realizar la instalación de BIND ejecutamos el siguiente comando:

```
# apt-get install bind9
```

Configuración

Una vez instalado BIND, el demonio encargado de brindar el servicio se llama named y permite implementar todo tipo de servidores mediante el fichero /etc/bind/named.conf.

Para configurar el servicio debemos editar el fichero y poner lo siguiente:

```
// Filename:        named.conf
// Domain:          .es

options {
        directory "/etc/bind";
   port 53;
   pid-file "named.pid";
   listen-on { any; };
   listen-on-v6 port 53 { any; };
   notify yes;
recursion yes;
allow-query {10.0/16; 2001:720::/48;};
};

// ROOT.DB file
zone "." {
        type hint;
        file "db.root";
};

zone "localhost.localdomain" IN {
        type master;
        file "localhost";
        allow-update { none; };
};

// Inverse Resolution for IPv4 loopback
zone "1.0.0.127.in-addr.arpa" {
        type master;
        file "named.loopback";
};
```

```
// Inverse Resolution for IPv6 loopback
zone "1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.arpa"
{
        type master;
        file "named.loopback";
};

zone "0.in-addr.arpa" IN {
        type master;
        file "named.empty";
        allow-update { none; };
};

// Inverse Resolution for 10.0.0.0/8
zone "10.in-addr.arpa" {
        type master;
        file "db.10";
};

// Inverse Resolution for 10.0.1.0/24
zone "1.0.10.in-addr.arpa" {
        type master;
        file "db.10.0.1";
};

// Inverse Resolution for 10.0.2.0/24
zone "2.0.10.in-addr.arpa" {
        type master;
        file "db.10.0.2";
};

// Inverse Resolution for 10.0.3.0/24
zone "3.0.10.in-addr.arpa" {
        type master;
        file "db.10.0.3";
};

// Inverse Resolution for 2001:720:041c::/48
zone "c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa" {
```

```
        type master;
        file "db.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa";
};

// Inverse Resolution for 2001:0720:041c:4000::/64
zone "4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa" {
        type master;
        file "db.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa";
};

// Inverse Resolution for 2001:720:041c:4001::/64
zone "1.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa" {
        type master;
        file "db.1.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa";
};

// Inverse Resolution for 2001:0720:041c:4002::/64
zone "2.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa" {
        type master;
        file "db.2.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa";
};

// Inverse Resolution for 2001:0720:041c:4003::/64
zone "3.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa" {
        type master;
        file "db.3.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa";
};

// Direct Resolution
zone "escuela.upm.es" {
   type master;
   file "db.escuela.upm.es";
};
```

Este fichero fundamentalmente describe las zonas de los dominios que controlará BIND. Para cada una de estas zonas se indica, entre otras características, el nombre del fichero que almacena sus datos. Es en estos ficheros donde se colocan todos los registros de recursos del DNS y se conocen como ficheros ``db'' (database files). En sistemas Ubuntu dichos ficheros se agrupan en el directorio /etc/bind/.

Como vemos en este fichero se define la zona "." cuyo fichero de zona root.hint, que contiene los servidores raíz y se encuentra almacenado en /etc/bind/db.root.

A continuación veremos el resto de ficheros de zona, es decir los ficheros que contienen la información relativa a las distintas zonas:

- Fichero **localhost** para la zona **localhost.localdomain:**

```
$TTL 1D
@       IN SOA  @ rname.invalid. (
                                        0       ; serial
                                        1D      ; refresh
                                        1H      ; retry
                                        1W      ; expire
                                        3H )    ; minimum
        NS      @
        A       127.0.0.1
        AAAA    ::1
```

- Fichero **named.loopback** para las zonas **1.0.0.127.in-addrr.arpa** y **1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.arpa:**

```
$TTL 1D
@       IN SOA  @ rname.invalid. (
                                                0       ; serial
                                                1D      ; refresh
                                                1H      ; retry
                                                1W      ; expire
                                                3H )    ; minimum
        NS      @
        A       127.0.0.1
        AAAA    ::1
        PTR     localhost.
```

- Fichero **named.empty** para la zona **0.in-addr.arpa:**

```
$TTL 3H
@       IN SOA  @ rname.invalid. (
                                                0       ; serial
                                                1D      ; refresh
                                                1H      ; retry
```

```
                                        1W      ; expire
                                        3H )    ; minimum
        NS      @
        A       127.0.0.1
        AAAA    ::1
```

- Fichero **db.10** para la resolución inversa de la zona **10.0.0.0/8**:

```
;
; Domain:   10.in-addr.arpa
; Filename: db.10
;
$TTL   3600

@       IN    SOA    Srv-dns.escuela.upm.es.   root.Srv-dns.escuela.upm.es. (
        1 ; serial
                    3600        ; refresh (1hour)
                    1800        ; retry (30 min)
                    604800       ; expire (7 days)
                    3600        ; minimum (1 hour)
                                )

                IN    NS    Srv-dns.escuela.upm.es.
```

- Fichero **db.10.0.1** para la resolución inversa de la zona **10.0.1.0/24**:

```
; Domain:   1.0.10.in-addr.arpa
; Filename: db.10.0.1
;
$TTL   3600

@       IN    SOA    Srv-dns.escuela.upm.es.   root.Srv-dns.escuela.upm.es. (
        1 ; serial
                    3600        ; refresh (1hour)
                    1800        ; retry (30 min)
                    604800       ; expire (7 days)
                    3600        ; minimum (1 hour)
                                )
                IN    NS    Srv-dns.escuela.upm.es.
; PTR records for name servers (addresses 10.0.1.X)
1       IN    PTR    Gateway.escuela.upm.es.
```

```
2       IN      PTR     Xp-1.escuela.upm.es.
3       IN      PTR     Cli-ubuntu.escuela.upm.es.
```

- Fichero **db.10.0.2** para la resolución inversa de la zona **10.0.2.0/24**:

```
; Domain:   2.0.10.in-addr.arpa
; Filename:  db.10.0.2
;
$TTL   3600

@       IN     SOA    Srv-dns.escuela.upm.es.   root.Srv-dns.escuela.upm.es. (
        1 ; serial
                       3600          ; refresh (1hour)
                       1800          ; retry (30 min)
                       604800         ; expire (7 days)
                       3600          ; minimum (1 hour)
                                     )
                   IN     NS     Srv-dns.escuela.upm.es.

; PTR records for name servers (addresses 10.0.2.X)
1       IN     PTR    Gateway.escuela.upm.es.
2       IN     PTR    Xp-2.escuela.upm.es.
```

- Fichero **db.10.0.3** para la resolución inversa de la zona **10.0.3.0/24**:

```
; Domain:   3.0.10.in-addr.arpa
; Filename:  db.10.0.3
;
$TTL   3600

@       IN     SOA    Srv-dns.escuela.upm.es.   root.Srv-dns.escuela.upm.es. (
        1 ; serial
                       3600          ; refresh (1hour)
                       1800          ; retry (30 min)
                       604800         ; expire (7 days)
                       3600          ; minimum (1 hour)
                                     )

                   IN     NS     Srv-dns.escuela.upm.es.

; PTR records for name servers (addresses 10.0.3.X)
```

```
1       IN     PTR    Gateway.escuela.upm.es.
2       IN     PTR    Srv-web.escuela.upm.es.
3       IN     PTR    Srv-dhcp.escuela.upm.es.
4       IN     PTR    Srv-dns.escuela.upm.es.
```

- Fichero **db.10.0.4** para la resolución inversa de la zona **10.0.4.0/24**:

```
; Domain:    4.0.10.in-addr.arpa
; Filename:  db.10.0.4
;
$TTL    3600

@       IN     SOA    Srv-dns.escuela.upm.es.   root.Srv-dns.escuela.upm.es. (
        1 ; serial
                       3600          ; refresh (1hour)
                       1800          ; retry (30 min)
                       604800         ; expire (7 days)
                       3600          ; minimum (1 hour)
                                     )
                    IN     NS     Srv-dns.escuela.upm.es.

; PTR records for name servers (addresses 10.0.4.X)
1       IN     PTR    Gateway.escuela.upm.es.
2       IN     PTR    Srv-nat64.escuela.upm.es.
```

- Fichero **db.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa** para la resolución inversa de la zona **2001:720:041c::/48**:

```
; Domain:    c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa
;          Inverse resolution for prefix 2001:720:04c1::/48
; Filename:  db.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa
;
$TTL    86400

@       IN     SOA    Srv-dns.escuela.upm.es.   root.Srv-dns.escuela.upm.es. (
        1 ; serial
                       3600          ; refresh (1hour)
                       1800          ; retry (30 min)
                       604800         ; expire (7 days)
                       3600          ; minimum (1 hour)
                                     )
```

```
        IN     NS     Srv-dns.escuela.upm.es.

; PTR records for name servers (addresses 2001:720:04c1::X)
$ORIGIN 0.0.0.0.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa.
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0                IN        PTR       Srv-dns.escuela.upm.es.

; Inverse delegation for ESCUELA.UPM.ES (2001:720:4c1:4001::/64)
1.0.0.4.1.c.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN     NS        Srv-dns.escuela.upm.es.
; Inverse delegation for ESCUELA.UPM (2001:720:4c1:4002::/64)
2.0.0.4.1.c.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN     NS        Srv-dns.escuela.upm.es.
; Inverse delegation for UPM (2001:720:4c1:4003::/64)
3.0.0.4.1.c.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN     NS        Srv-dns.escuela.upm.es.
; Inverse delegation for UPM (2001:720:4c1:4004::/64)
4.0.0.4.1.c.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN     NS        Srv-dns.escuela.upm.es.
; Inverse delegation for UPM (2001:720:4c1:4005::/64)
5.0.0.4.1.c.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN     NS        Srv-dns.escuela.upm.es.
```

- Fichero **db.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa** para la resolución inversa de la zona **2001:720:041c:4000:/48**:

```
;
; Domain:    c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa
;          Inverse resolution for prefix 2001:720:04c1::/48
; Filename:  db.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa
;
$TTL    86400
@       IN     SOA    Srv-dns.escuela.upm.es.   root.Srv-dns.escuela.upm.es. (
                    1 ; serial
                      3600          ; refresh (1hour)
                      1800          ; retry (30 min)
                      604800         ; expire (7 days)
                      3600          ; minimum (1 hour)
                                    )
IN     NS     Srv-dns.es.
; PTR records for name servers (addresses 2001:720:04c1::X)
$ORIGIN 0.0.0.0.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa.
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0                IN        PTR       Srv-dns.escuela.upm.es.
; Inverse delegation for ESCUELA.UPM.ES (2001:720:4c1:4001::/64)
1.0.0.4.1.c.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN     NS        Srv-dns.escuela.upm.es.
; Inverse delegation for ESCUELA.UPM (2001:720:4c1:4002::/64)
```

```
2.0.0.4.1.c.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN     NS       Srv-dns.escuela.upm.es.
; Inverse delegation for UPM (2001:720:4c1:4003::/64)
3.0.0.4.1.c.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN     NS       Srv-dns.escuela.upm.es.
; Inverse delegation for UPM (2001:720:4c1:4004::/64)
4.0.0.4.1.c.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN     NS       Srv-dns.escuela.upm.es.
; Inverse delegation for UPM (2001:720:4c1:4005::/64)
5.0.0.4.1.c.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN     NS       Srv-dns.escuela.upm.es.
```

- Fichero **db.1.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa** para la resolución inversa de la zona **2001:720:041c:4001:/48**:

```
; Domain:    1.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa
;          Inverse resolution for prefix 2001:720:04c1:4001::/64
; Filename:  db.1.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa
;
$TTL    86400

@       IN     SOA    Srv-dns.escuela.upm.es.   root.Srv-dns.escuela.upm.es. (
        1 ; serial
                       3600         ; refresh (1hour)
                       1800         ; retry (30 min)
                       604800        ; expire (7 days)
                       3600         ; minimum (1 hour)
                                    )
IN     NS     Srv-dns.escuela.upm.es.

; PTR records for name servers (addresses 2001:720:04c1:4001::X)
; Inverse delegation for UPM (2001:720:04c1:4001::/64)
1.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN     NS       Srv-dns.escuela.upm.es.

; db.1.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa. PTR records
; hosts under 2001:720:04c1:4001::/64
;
$ORIGIN 1.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa.
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0          IN        PTR       Gateway.escuela.upm.es.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0          IN        PTR       Xp-1.escuela.upm.es.
3.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0          IN        PTR       Cli-ubuntu.escuela.upm.es.
```

- Fichero **db.2.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa** para la resolución inversa de la zona **2001:720:041c:4002:/48**:

```
; Domain:    2.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa
;          Inverse resolution for prefix 2001:720:04c1:4002::/64
; Filename:  db.2.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa
;
$TTL    86400

@       IN      SOA     Srv-dns.escuela.upm.es.   root.Srv-dns.escuela.upm.es. (
        1 ; serial
                        3600          ; refresh (1hour)
                        1800          ; retry (30 min)
                        604800         ; expire (7 days)
                        3600          ; minimum (1 hour)
                                )
                    IN      NS      Srv-dns.escuela.upm.es.
; PTR records for name servers (addresses 2001:720:04c1:4002::X)
; Inverse delegation for UPM (2001:720:04c1:4002::/64)
2.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN      NS        Srv-dns.escuela.upm.es.

; db.2.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa. PTR records
; hosts under 2001:720:04c1:4003::/64
;
$ORIGIN 2.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa.
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0                 IN        PTR       Gateway.escuela.upm.es.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0                 IN        PTR       Xp-2.escuela.upm.es.
```

- Fichero **db.3.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa** para la resolución inversa de la zona **2001:720:041c:4003:/48**:

```
; Domain:    3.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa
;          Inverse resolution for prefix 2001:720:04c1:4003::/64
; Filename:  db.3.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa
;
$TTL    86400

@       IN      SOA     Srv-dns.escuela.upm.es.   root.Srv-dns.escuela.upm.es. (
        1 ; serial
                        3600          ; refresh (1hour)
```

```
                        1800        ; retry (30 min)
                        604800       ; expire (7 days)
                        3600        ; minimum (1 hour)
                                    )
        IN    NS    Srv-dns.escuela.upm.es.

; PTR records for name servers (addresses 2001:720:04c1:4003::X)
; Inverse delegation for UPM (2001:720:04c1:4003::/64)
3.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN    NS       Srv-dns.escuela.upm.es.

; db.3.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa. PTR records
; hosts under 2001:720:04c1:4003::/64
;
$ORIGIN 3.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa.
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0    IN      PTR     Gateway.escuela.upm.es.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0    IN      PTR Srv-web.escuela.upm.es.
3.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0    IN      PTR     Srv-dhcp.escuela.upm.es.
4.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0    IN      PTR     Srv-dns.escuela.upm.es.
```

- Fichero **db.4.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa** para la resolución inversa de la zona **2001:720:041c:4004:/48**:

```
; Domain:   3.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa
;         Inverse resolution for prefix 2001:720:04c1:4003::/64
; Filename:  db.4.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa
;
$TTL   86400

@          IN    SOA    Srv-dns.escuela.upm.es.   root.Srv-dns.escuela.upm.es. (
                        1 ; serial
                        3600        ; refresh (1hour)
                        1800        ; retry (30 min)
                        604800       ; expire (7 days)
                        3600        ; minimum (1 hour)
                        )

IN    NS    Srv-dns.escuela.upm.es.
; PTR records for name servers (addresses 2001:720:04c1:4004::X)
; Inverse delegation for UPM (2001:720:04c1:4004::/64)
4.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa.  IN    NS   Srv-dns.escuela.upm.es.
```

```
; db.4.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa. PTR records
; hosts under 2001:720:04c1:4004::/64
;
$ORIGIN 4.0.0.4.c.1.4.0.0.2.7.0.1.0.0.2.ip6.arpa.
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0    IN    PTR    Gateway.escuela.upm.es.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0    IN    PTR    Srv-nat64.escuela.upm.es.
```

- Fichero **db.escuela.upm.es** para la resolución directa de la zona **escuela.upm.es**:

```
;
; Domain:    .escuela.upm.es
; Filename:  db.escuela.upm.es
;

$TTL 300
@              IN      SOA     nat64-15.escuela.upm.es.          root.nat64-
15.escuela.upm.es. (
                                1           ; serial
                                3600        ; refresh (1hour)
                                1800        ; retry (30 min)
                                604800      ; expire (7 days)
                                3600    ; minimum (1 hour)
                                )
escuela.upm.es.        IN      NS      Srv-nat64.escuela.upm.es.
Srv-nat64.escuela.upm.es.          IN      A       10.0.4.2
Srv-nat64.escuela.upm.es.          IN      AAAA    2001:720:041c:4004::2
Srv-nat64.escuela.upm.es.          IN      AAAA    2001:720:041c:4005::1

;
; Host DNS records under .escuela.upm.es
;
Srv-nat64.escuela.upm.es.      IN      A       10.0.4.2
        IN      AAAA    2001:720:041c:4004::2
        IN      AAAA    2001:720:041c:4005::1
Gateway.escuela.upm.es.        IN      A       10.0.3.1
        IN      AAAA    2001:720:041c:4001::1
        IN      AAAA    2001:720:041c:4002::1
        IN      AAAA    2001:720:041c:4003::1
```

```
        IN      AAAA    2001:720:041c:4004::1
Srv-web.escuela.upm.es.IN       A       10.0.3.2
        IN      AAAA    2001:720:041c:4003::2
Srv-dhcp.escuela.upm.es.        IN      A       10.0.3.3
        IN      AAAA    2001:720:041c:4003::3
Srv-dns.escuela.upm.es. IN      A       10.0.3.4
        IN      AAAA    2001:720:041c:4003::4

Xp-1.escuela.upm.es.    IN      A       10.0.1.2
        IN      AAAA    2001:720:041c:4001::2

Xp-2.escuela.upm.es     IN      A       10.0.2.2
        IN      AAAA    2001:720:041c:4002::2
Cli-ubuntu.escuela.upm.es.      IN      A       10.0.2.3

Ubuntu-1.escuela.upm.es. IN     AAAA    2001:720:041c:4005::2
```

Pruebas

Para probar el funcionamiento del servicio, desde uno de los clientes que toman como servidor DNS a "**Srv-dns**", por ejemplo "**Cli-ubuntu**", comprobamos que se resuelven consultas sobre los dominios que se administran en el servidor. Para ello podemos utilizar el comando dig de la siguiente forma:

```
# dig @10.0.3.4 srv-dhcp.escuela.upm.es
; <<>> DiG 9.7.0-P1 <<>> @10.0.3.4 srv-dhcp.escuela.upm.es
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 2407
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 4

;; QUESTION SECTION:
;srv-dhcp.escuela.upm.es.       IN      A

;; ANSWER SECTION:
srv-dhcp.escuela.upm.es. 300    IN      A       10.0.3.3

;; AUTHORITY SECTION:
escuela.upm.es.         300     IN      NS      Srv-dns.escuela.upm.es.
```

```
;; ADDITIONAL SECTION:
Srv-dns.escuela.upm.es. 300    IN     A      10.0.3.5
Srv-dns.escuela.upm.es. 300    IN     A      10.0.3.4
Srv-dns.escuela.upm.es. 300    IN     AAAA   2001:720:41c:4003::4
Srv-dns.escuela.upm.es. 300    IN     AAAA   2001:720:41c:4003::5

;; Query time: 1 msec
;; SERVER: 10.0.3.4#53(10.0.3.4)
;; WHEN: Wed Jul  6 11:46:19 2011
;; MSG SIZE  rcvd: 167
```

 # dig @10.0.3.4 -x 10.0.2.2

```
; <<>> DiG 9.7.0-P1 <<>> @10.0.3.4 -x 10.0.2.2
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27609
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 4

;; QUESTION SECTION:
;2.2.0.10.in-addr.arpa.         IN     PTR

;; ANSWER SECTION:
2.2.0.10.in-addr.arpa.  3600   IN     PTR    Xp-2.escuela.upm.es.

;; AUTHORITY SECTION:
2.0.10.in-addr.arpa.    3600   IN     NS     Srv-dns.escuela.upm.es.

;; ADDITIONAL SECTION:
Srv-dns.escuela.upm.es. 300    IN     A      10.0.3.5
Srv-dns.escuela.upm.es. 300    IN     A      10.0.3.4
Srv-dns.escuela.upm.es. 300    IN     AAAA   2001:720:41c:4003::4
Srv-dns.escuela.upm.es. 300    IN     AAAA   2001:720:41c:4003::5

;; Query time: 1 msec
;; SERVER: 10.0.3.4#53(10.0.3.4)
;; WHEN: Wed Jul  6 11:47:32 2011
;; MSG SIZE  rcvd: 182
```

Si en vez de hacer consultas sobre los dominios que administra "**Srv-dns**", hacemos consultas al servidor sobre otros dominios, el resultado es el siguiente:

```
# dig @10.0.3.4 www.google.com
; <<>> DiG 9.7.0-P1 <<>> @10.0.3.4 www.google.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 14785
;; flags: qr rd ra; QUERY: 1, ANSWER: 7, AUTHORITY: 4, ADDITIONAL: 0

;; QUESTION SECTION:
;www.google.com.                 IN      A

;; ANSWER SECTION:
www.google.com.         604692  IN      CNAME   www.l.google.com.
www.l.google.com.       192     IN      A       74.125.39.105
www.l.google.com.       192     IN      A       74.125.39.106
www.l.google.com.       192     IN      A       74.125.39.147
www.l.google.com.       192     IN      A       74.125.39.99
www.l.google.com.       192     IN      A       74.125.39.103
www.l.google.com.       192     IN      A       74.125.39.104

;; AUTHORITY SECTION:
google.com.             172692  IN      NS      ns1.google.com.
google.com.             172692  IN      NS      ns3.google.com.
google.com.             172692  IN      NS      ns2.google.com.
google.com.             172692  IN      NS      ns4.google.com.

;; Query time: 2 msec
;; SERVER: 10.0.3.4#53(10.0.3.4)
;; WHEN: Wed Jul  6 11:44:07 2011
;; MSG SIZE  rcvd: 220

# dig @10.0.3.4 -x  77.238.178.122
; <<>> DiG 9.7.0-P1 <<>> @10.0.3.4 -x 77.238.178.122
; (1 server found)
;; global options: +cmd
;; Got answer:
```

```
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37097
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 5, ADDITIONAL: 2

;; QUESTION SECTION:
;122.178.238.77.in-addr.arpa.   IN      PTR

;; ANSWER SECTION:
122.178.238.77.in-addr.arpa. 1800 IN    PTR     w2.rc.vip.ird.yahoo.com.

;; AUTHORITY SECTION:
178.238.77.in-addr.arpa. 172800 IN      NS      ns1.yahoo.com.
178.238.77.in-addr.arpa. 172800 IN      NS      ns3.yahoo.com.
178.238.77.in-addr.arpa. 172800 IN      NS      ns4.yahoo.com.
178.238.77.in-addr.arpa. 172800 IN      NS      ns2.yahoo.com.
178.238.77.in-addr.arpa. 172800 IN      NS      ns5.yahoo.com.

;; ADDITIONAL SECTION:
ns1.yahoo.com.          172621  IN      A       68.180.131.16
ns2.yahoo.com.          172621  IN      A       68.142.255.16

;; Query time: 388 msec
;; SERVER: 10.0.3.4#53(10.0.3.4)
;; WHEN: Wed Jul  6 11:45:10 2011
;; MSG SIZE  rcvd: 204
```

5.3.2 WEB

Introducción

La navegación por Internet utiliza el protocolo HTTP para transferir hipertextos, imágenes, videos,… Generalmente para la navegación web se usa el puerto 80. Este servicio, probablemente el más extendido de todos, se basa en un modelo cliente-servidor por lo que se requieren ambos para establecer la comunicación. Hay varios programas que ofrecen este servicio pero el más extendido es Apache cuyo entorno natural de ejecución son las plataformas Linux.

En la simulación realizada en este Trabajo Fin de Master se pretende acceder a un servidor apache que está configurado para atender peticiones de conexión en IPv4 y en IPv6. Para obtener soporte IPv6 es necesario instalar versiones 2.x. Configuraremos además tanto el protocolo HTTP como el protocolo HTTPS.

Instalación

La versión del servidor apache utilizada en esta simulación es 2.2.16. El primer paso es instalarla en el servidor. La máquina virtual sobre la que se aloja es "**Srv-web**" y para instalar el servidor debemos ejecutar el siguiente comando:

```
# apt-get install apache2
```

Configuración

A partir de Apache versión 2.0.x el soporte IPv6 viene habilitado por defecto, así que después de instalarlo sólo hay que configurarlo para que escuche IPv6.

La directiva de apache que controla las IPs y los puertos por los que escucha el servidor web es **Listen** y se encuentra en el fichero /etc/apache2/ports.conf. Debemos editar el fichero y poner lo siguiente :

```
Listen 10.0.3.2:80
Listen [2001:720:41c:4003::2]:80
```

Para comprobar que escucha en ambas direcciones por el puerto 80 ejecutamos:

```
# netstat -punta | grep 80
tcp        0      0 10.0.3.2:80             0.0.0.0:*               ESCUCHAR    2973/apache2
tcp6       0      0 2001:720:41c:4003::2:80 :::*                    ESCUCHAR    2973/apache2
```

A continuación configuramos los Host Virtuales en el fichero /etc/apache2/sites-available/default. Para el Host Virtual de IPv6 es necesario utilizar corchetes para "encerrar" la dirección IPv6:

```
NameVirtualHost  10.0.3.2:80
<VirtualHost 10.0.3.2:80>
ServerAdmin gloria@etsit.upm.es
ServerName Srv-web.escuela.upm.es
DocumentRoot /var/www
<Directory />
        Options FollowSymLinks
AllowOverride None
</Directory>
<Directory /var/www/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
```

```
            allow from all
</Directory>
ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
<Directory "/usr/lib/cgi-bin">
AllowOverride None
            Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
            Order allow,deny
            Allow from all
</Directory>
ErrorLog ${APACHE_LOG_DIR}/error.log
      # Possible values include: debug, info, notice, warn, error, crit,
      # alert, emerg.
LogLevel warn
CustomLog ${APACHE_LOG_DIR}/access.log combined
Alias /doc/ "/usr/share/doc/"
<Directory "/usr/share/doc/">
      Options Indexes MultiViews FollowSymLinks
      AllowOverride None
      Order deny,allow
      Deny from all
     Allow from 127.0.0.0/255.0.0.0 ::1/128
</Directory>
</VirtualHost>
```

NameVirtualHost [2001:720:41c:4003::2]:80

```
<VirtualHost [2001:720:41c:4003::2]:80>
ServerAdmin gloria@etsit.upm.es
ServerName [Srv-web.escuela.upm.es]
DocumentRoot /var/www/ipv6
ErrorLog "/var/log/apache2/ipv6.error.log"
CustomLog "/var/log/apache2/ipv6.access.log" common
<Directory /var/www/ipv6>
            Options Indexes FollowSymLinks MultiViews
            AllowOverride None
            Order allow,deny
            allow from all
</Directory>
</VirtualHost>
```

Este archivo muestra una configuración muy simple. En las líneas NameVirtualHost se indica cuáles son las IP's que se corresponden con los "VirtualHost" que se van a utilizar. Posteriormente, en cada etiqueta "VirtualHost" hay que completar el atributo "DocumentRoot" con el directorio en el que se encuentran los archivos que deseamos mostrar al usuario que se conecta por IPv4 o por IPv6.

También queremos activar el modo ssl, así que en primer lugar debemos instalarlo y habilitarlo. Para instalarlo, debemos ejecutar los siguientes comandos::

```
# a2enmod ssl
Enabling module ssl.
See /usr/share/doc/apache2.2-common/README.Debian.gz on how to configure SSL and create self-signed certificates.
Run '/etc/init.d/apache2 restart' to activate new configuration!
# a2ensite default-ssl
```

Después de la instalación es necesario reiniciar el servicio:

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

A continuación generamos el certificado ssl y para ello, ejecutamos los siguientes comandos:

```
# opensslreq $@ -new -x509 -days 365 -nodes -out /etc/apache2/apache.pem -keyout /etc/apache2/apache.key
# chmod 600 /etc/apache2/apache.pem
```

Una vez instalado, debemos editar el fichero/etc/apache2/sites-available/default-ssl para configurar los Host Virtuales en modo seguro:

```
<IfModule mod_ssl.c>
<VirtualHost 10.0.3.2:443>
ServerAdmin gloria@etsit.upm.es
DocumentRoot /var/www
<Directory />
        Options FollowSymLinks
AllowOverride None
</Directory>
<Directory /var/www/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
         allow from all
```

```
</Directory>
ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
<Directory "/usr/lib/cgi-bin">
AllowOverride None
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
</Directory>
ErrorLog ${APACHE_LOG_DIR}/error.log
    # Possible values include: debug, info, notice, warn, error, crit,
    # alert, emerg.
LogLevel warn
CustomLog ${APACHE_LOG_DIR}/ssl_access.log combined
    Alias /doc/ "/usr/share/doc/"
<Directory "/usr/share/doc/">
        Options Indexes MultiViews FollowSymLinks
         AllowOverride None
        Order deny,allow
        Deny from all
        Allow from 127.0.0.0/255.0.0.0 ::1/128
</Directory>
    #   SSL Engine Switch:
    #   Enable/Disable SSL for this virtual host.
SSLEngine on
SSLCertificateFile /etc/apache2/apache.pem
SSLCertificateKeyFile /etc/apache2/apache.key
<FilesMatch "\.(cgi|shtml|phtml|php)$">
   SSLOptions +StdEnvVars
</FilesMatch>
<Directory /usr/lib/cgi-bin>
SSLOptions +StdEnvVars
</Directory>
BrowserMatch "MSIE [2-6]" \
nokeepalivessl-unclean-shutdown \
downgrade-1.0 force-response-1.0
    # MSIE 7 and newer should be able to use keepalive
BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown
</VirtualHost>
```

```
<VirtualHost [2001:720:41c:4003::2]:443>
ServerAdmin gloria@etsit.upm.es
ServerName [Srv-web.escuela.upm.e6]
DocumentRoot /var/www/ipv6/
<Directory /var/www/ipv6>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
</Directory>
SSLEngine on
SSLCertificateFile /etc/apache2/apache.pem
SSLCertificateKeyFile /etc/apache2/apache.key
</VirtualHost>
</IfModule>
```

Pruebas

Una vez que el servidor apache ha sido configurado e iniciado de forma satisfactoria, tan solo hay que ejecutar los siguientes comandos para leer una página Web tanto en IPv4 como en IPv6:

- **IPv4**

  Para comprobar el funcionamiento del servicio en IPv4 sólo tenemos que ejecutar el siguiente comando:

```
# w3m 10.0.3.2
```

  El resultado se puede ver en la siguiente figura:

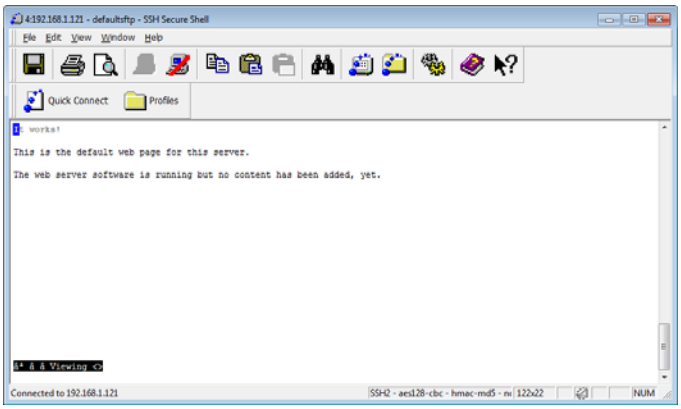

Figura 18. Petición de página Web en IPv4

- **IPv6**

Para comprobar el funcionamiento del servicio en IPv6 sólo tenemos que ejecutar el siguiente comando:

```
# w3m [2001:720:41c:4003::2]
```

El resultado se puede ver en la siguiente figura:

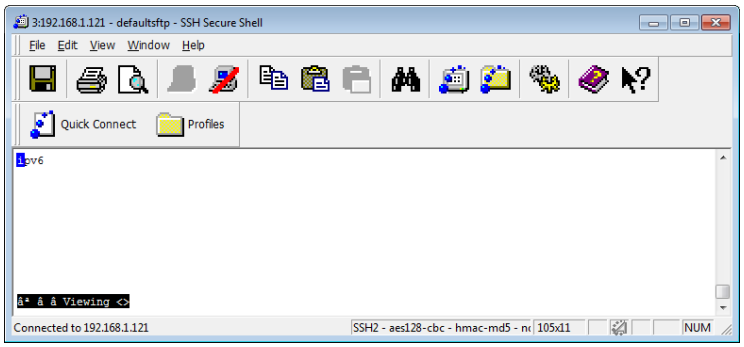

Figura 19. Petición de página Web en IPv6

5.3.3 FTP

FTP (File Transfer Protocol) o Protocolo de Transferencia de Archivos es uno de los protocolos estándar más utilizados en Internet siendo el más idóneo para la transferencia de grandes bloques de datos a través de redes que soporten TCP/IP. El servicio utiliza los puertos 20 y 21, exclusivamente sobre TCP. El puerto 20 es utilizado para el flujo de datos entre cliente y servidor. El puerto 21 es utilizando para el envío de órdenes del cliente hacia el servidor. Prácticamente todos los sistemas operativos y plataformas incluyen soporte para FTP, lo que permite que cualquier ordenador conectado a una red basada sobre TCP/IP pueda hacer uso de este servicio a través de un cliente FTP.

El servidor de FTP elegido es VSFTPD (Very Secure FTP Daemon). Se distingue principalmente porque sus valores por defecto son muy seguros y por su sencillez en la configuración, comparado con otras alternativas como Wu-ftpd.

Instalación

La versión del servidor VSFTPD utilizada en esta simulación es la 2.3.0. El primer paso es instalarla en el servidor. La máquina virtual sobre la que se aloja es "**Srv-web**". Para instalar el servidor sólo tenemos que ejecutar la siguiente instrucción:

```
# apt-get install vsftpd
```

Configuración

Una vez instalado, debemos configurar el servicio para que VSFTPD funcione tanto con el protocolo IPv4 como con el protocolo IPv6. Para ello, debemos editar el fichero de configuración /etc/vsftpd.conf, comentar la siguiente directiva:

```
#listen=YES
```

Y después añadir la siguiente directiva:

```
listen_ipv6=YES
```

Una vez configurado, debemos iniciar el servicio y para ello ejecutamos la siguiente instrucción:

```
# /etc/init.d/vsftpd start
```

Una vez iniciado, podemos comprobar que el servicio está arrancado ejecutando la siguiente instrucción:

```
# netstat -punta | grep ftp| grep 21
tcp6       0      0 :::21                   :::*                    ESCUCHAR    9007/vsftpd
```

Pruebas

Para comprobar que el servicio funciona correctamente podemos comprobar en primer lugar que el servidor ftp está disponible para IPv4. Para ello, nos conectamos desde otro equipo al servicio ftp ejecutando el siguiente comando y utilizando un usuario del sistema:

```
# ftp 10.0.3.2
Connected to 10.0.3.2.
220 (vsFTPd 2.3.0)
Name (10.0.3.2:root): vnx
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>pwd
257 "/home/vnx"
ftp>ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
```

```
drwxr-xr-x  2 1000  1000  4096 Mar 12 22:54 Desktop
226 Directory send OK.
ftp> quit
221 Goodbye.
```

A continuación podemos comprobar que funciona con la dirección IPv6 y hacemos lo mismo que antes, pero nos conectamos a la dirección IPv6 del servidor:

```
# ftp 2001:720:41c:4003::2
Connected to 2001:720:41c:4003::2.
220 (vsFTPd 2.3.0)
Name (2001:720:41c:4003::2:root): vnx
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>pwd
257 "/home/vnx"
ftp>
```

5.3.4 NAT64

Introducción

El objetivo de las técnicas de traducción de protocolos es proveer de rutas transparentes a los nodos en las redes de IPv6 para comunicarlos con los nodos de redes IPv4 y viceversa.

El mecanismo de traducción que utilizaremos en este Trabajo Fin de Máster será NAT64. Se trata de un mecanismo para permitir a los clientes IPv6 comunicarse con servidores IPv4.

En nuestro escenario instalaremos un servidor NAT64 que además de la traducción, realizará la función de DNS.

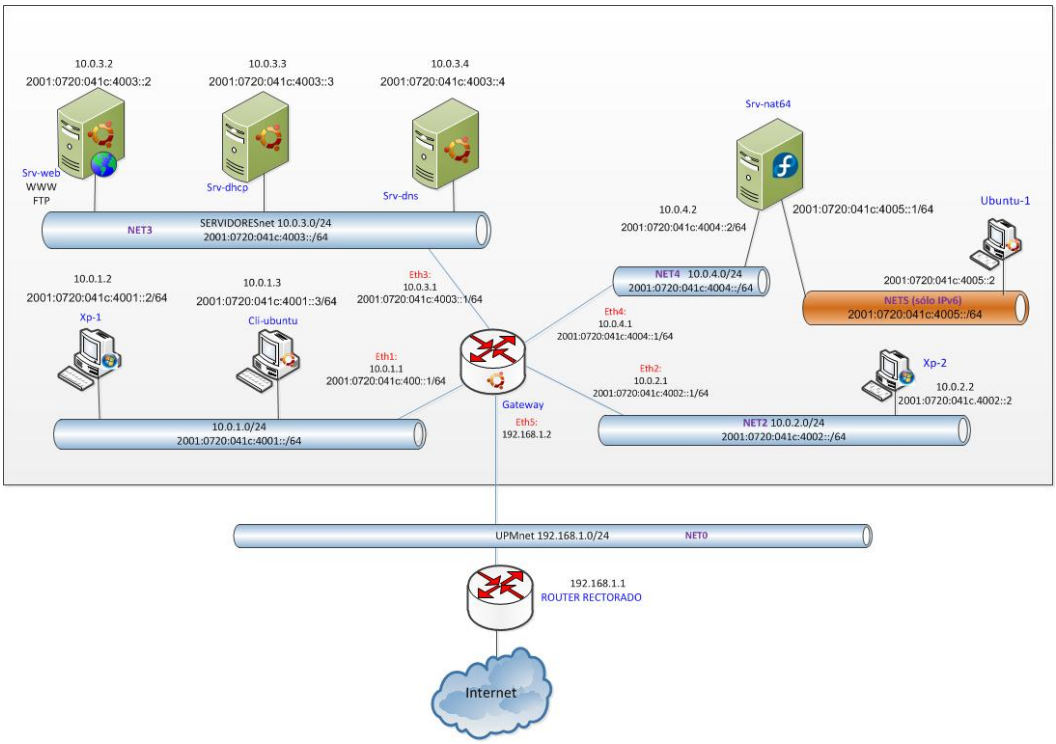

Figura 20. Escenario de simulación

A la vista de este escenario no debemos olvidar añadir en "**Gateway**" las siguientes rutas:

```
# route -A inet6 add 2001:720:41c:4005::/64 gw 2001:720:41c:4004::2 dev eth4
# route -A inet6 add 64:FF9B::/96 gw 2001:720:41c:4004::2 dev eth4
```

Siendo 64:FF9B::/96 el prefijo que usará NAT64 para realizar la traducción **.**

Instalación

Para que funcione la translación IP necesitamos configurar además del servicio NAT64, el servicio DNS64. Para ambos servicios vamos a utilizar una implementación Open-Source Ecdysis [15] desarrollada por la empresa Viagenie.

Veamos los pasos a seguir para instalar cada uno de los servicios:

- **DNS64**

  En primer lugar, nos descargamos la última versión de la implementación Ecdysis-bind que es **ecdysis-bind-9.7.2-P2D20101117.tar.gz.**

  Una vez descargado el paquete, tenemos que descomprimirlo ejecutando el siguiente comando:

```
# tar xvzf ecdysis-bind-9.7.2-P2D20101117.tar.gz
```

Una vez descomprimido debemos instalarlo, y para ello, ejecutamos los siguientes comandos:

```
# cd ecdysis-bind-9.7.2-P2D20101117
#./configure
# make
# make install
```

- **NAT64**

En primer lugar, nos descargamos la última versión de la implementación Ecdysis-nf-nat64 que es **ecdysis-nf-nat64-20101117.tar.gz**.

Una vez descargado el paquete, tenemos que descomprimirlo ejecutando el siguiente comando:

```
# tar xvzf ecdysis-nf-nat64-20101117.tar.gz
```

Una vez descomprimido debemos instalarlo, y para ello, ejecutamos los siguientes comandos:

```
# cd ecdysis-nf-nat64-20101117
# make
# make install
```

Configuración

Una vez instalados los paquetes correspondientes, debemos configurar tanto DNS64 como NAT64.

- **DNS64**

Para configurar el servicio DNS64 tenemos que editar el fichero de configuración del BIND, el fichero /etc/named.conf y dejarlo como sigue:

```
// named.conf
//
options {
listen-on port 53 { 10.0.4.2; };
listen-on-v6 port 53 { 2001:720:41c:4004::2; };
directory       "/var/named";
dump-file       "/var/named/data/cache_dump.db";
statistics-file "/var/named/data/named_stats.txt";
memstatistics-file "/var/named/data/named_mem_stats.txt";
```

```
dns64-prefix 64:FF9B::/96;
recursion yes;
allow-query {10.0/16; 2001:720::/48;};
dnssec-enable yes;
dnssec-validation yes;
dnssec-lookaside auto;

        /* Path to ISC DLV key */
bindkeys-file "/etc/named.iscdlv.key";

managed-keys-directory "/var/named/dynamic";
};
logging {
channel default_debug {
file "data/named.run";
severity dynamic;
        };
};
zone "." IN {
type hint;
file "named.ca";
};
include "/etc/named.rfc1912.zones";
include "/etc/named.root.key";
```

Además, tenemos que modificar el fichero /var/named/db.escuela.upm.es y dejarlo como sigue:

```
; Domain:    .escuela.upm.es
; Filename:  db.escuela.upm.es

$TTL 300
@               IN      SOA     Srv-nat64.escuela.upm.es.               root.Srv-nat64.escuela.upm.es. (
                                        1               ; serial
                                        3600            ; refresh (1hour)
                                        1800            ; retry (30 min)
                                        604800          ; expire (7 days)
                                        3600    ; minimum (1 hour)
                                        )
```

```
escuela.upm.es.          IN      NS      Srv-nat64.escuela.upm.es.
Srv-nat64.escuela.upm.es.                IN      A       10.0.4.2
Srv-nat64.escuela.upm.es.                IN      AAAA    2001:720:041c:4004::2
Srv-nat64.escuela.upm.es.                IN      AAAA    2001:720:041c:4005::1

; Host DNS records under .escuela.upm.es
;
Srv-nat64.escuela.upm.es.        IN      A       10.0.4.2
        IN      AAAA    2001:720:041c:4004::2
        IN      AAAA    2001:720:041c:4005::1
Gateway.escuela.upm.es.          IN      A       10.0.3.1
        IN      AAAA    2001:720:041c:4001::1
        IN      AAAA    2001:720:041c:4002::1
        IN      AAAA    2001:720:041c:4003::1
        IN      AAAA    2001:720:041c:4004::1
Srv-web.escuela.upm.es.IN       A       10.0.3.2
;       IN      AAAA    2001:720:041c:4003::2
Srv-dhcp.escuela.upm.es.         IN      A       10.0.3.3
        IN      AAAA    2001:720:041c:4003::3
Srv-dns.escuela.upm.es. IN      A       10.0.3.4
        IN      AAAA    2001:720:041c:4003::4
Xp-1.escuela.upm.es.     IN      A       10.0.1.2
        IN      AAAA    2001:720:041c:4001::2
Xp-2.escuela.upm.es      IN      A       10.0.2.2
        IN      AAAA    2001:720:041c:4002::2
Cli-ubuntu.escuela.upm.es.       IN      A       10.0.2.3
Ubuntu-1.escuela.upm.es. IN      AAAA    2001:720:041c:4005::2
```

Como vemos la máquina "**Srv-web.escuela.upm.es**" sólo tiene registro IPv4. Esto lo hacemos así para comprobar después que desde una máquina en una red sólo–IPv6 nos podemos conectar a un servidor web IPv4.

Una vez configurado el servicio DNS64, debemos reiniciar el servicio. Para ello, podemos ejecutar el siguiente comando:

```
# named –c named.conf
```

- **NAT64**

Para configurar el servicio NAT64 tenemos que modificar el fichero nat64-config.sh añadiendo los siguientes parámetros:

```
IPV4_ADDR="10.0.4.2"
PREFIX_ADDR="64:ff9b::"
PREFIX_LEN="96":
```

Una vez configurado, sólo nos queda iniciar el servicio. Para ello, ejecutamos el siguiente comando:

```
# ./nat64-config.sh
```

Pruebas

Una vez instalados y configurados los servicios de DNS64 y NAT64, vamos a realizar unas pruebas para probar que todo funciona correctamente.

- **DNS64**

  Para comprobar que DNS64 funciona correctamente vamos a realizar peticiones al servidor de las direcciones IPv6 asociadas a los registros AAAA. Hay que recordar que si hacemos una petición al servidor DNS64 de un registro AAAA de una máquina con sólo dirección IPv4 y por lo tanto, no posee un registro asociado del tipo AAAA, entonces, el servidor DNS64 nos devolverá una IPv6 del tipo 64:ff9b::xxxx.xxxx donde xxxx.xxxx es la dirección IPv4 de la máquina en formato hexadecimal.

  Para hacer la comprobación utilizamos el comando **dig**. Por ejemplo:

```
# dig @10.0.4.2 www.yahoo.es aaaa
; <<>>DiG 9.7.2-P2 <<>> @10.0.4.2 www.yahoo.es aaaa
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41334
;; flags: qrrdra; QUERY: 1, ANSWER: 5, AUTHORITY: 2, ADDITIONAL: 0

;; QUESTION SECTION:
;www.yahoo.es.                  IN      AAAA
;; ANSWER SECTION:
www.yahoo.es.           286     IN      CNAME   rc.yahoo.com.
rc.yahoo.com.           287     IN      CNAME   rc.g01.yahoodns.net.
rc.g01.yahoodns.net.    289     IN      CNAME   any-rc.a01.yahoodns.net.
any-rc.a01.yahoodns.net. 289    IN      AAAA    64:ff9b::57f8:7894
any-rc.a01.yahoodns.net. 289    IN      AAAA    64:ff9b::4dee:b27a
```

```
;; AUTHORITY SECTION:
a01.yahoodns.net.      172789  IN      NS      yf2.yahoo.com.
a01.yahoodns.net.      172789  IN      NS      yf1.yahoo.com.

;; Query time: 76 msec
;; SERVER: 10.0.4.2#53(10.0.4.2)
;; WHEN: Tue Jun 28 20:32:21 2011
;; MSG SIZE  rcvd: 206
```

En el ejemplo anterior, al hacer una petición del registro AAAA de una máquina que no tiene este registro asociado, el servidor nos devuelve la dirección IPv6 del tipo 64:ff9b::xxxx.xxxx. Esto indica que el servidor DNS64 funciona correctamente para peticiones de registros AAAA de dominios que no administra.

Cuando hacemos peticiones al servidor de un registro AAAA de una máquina perteneciente al dominio que administra, por ejemplo "**Srv-web**", vemos que no devuelve la IPv6 del tipo 64:ff9b::xxxx.xxxx, tal y como se muestra a continuación:

```
# dig @10.0.4.2 Srv-web.escuela.upm.es aaaa
; <<>> DiG 9.7.2-P2 <<>> @10.0.4.2 Srv-web.escuela.upm.es aaaa
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 2922
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 0
;; QUESTION SECTION:
;Srv-web.escuela.upm.es.        IN      AAAA
;; AUTHORITY SECTION:
escuela.upm.es.         300     IN      SOA     nat64-15.escuela.upm.es. root.nat64-15.escuela.upm.es. 1 3600 1800 604800 3600
;; Query time: 1 msec
;; SERVER: 10.0.4.2#53(10.0.4.2)
;; WHEN: Mon Jun 20 22:07:07 2011
;; MSG SIZE  rcvd: 89
#
```

Desconocemos los motivos por los cuales se comporta de esta forma sólo para peticiones de registros de tipo AAAA de dominios que se administran desde el servidor DNS64.

- **NAT64**

Debido a los problemas detectados con DNS64 cuando hacemos peticiones de registros del tipo AAAA de máquinas asociadas a dominios que administra el servidor DNS64, para probar el servicio NAT64 tenemos que evitar que se haga una consulta al servidor DNS64 y, para ello, editamos el fichero /etc/hosts de la máquina cliente "**Ubuntu-1**" y añadimos la siguiente línea:

```
127.0.0.1       localhost
127.0.1.1       router
# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
64:ff9b::0a00:0302      Srv-web.escuela.upm.es
```

Una vez configurado el fichero /etc/hosts de la máquina ya podemos hacer una petición a un servidor con sólo IPv4 desde una máquina con sólo IPv6, teniendo en cuenta que la resolución de la consulta no la hace el servidor DNS64, sino el propio equipo a través del fichero /etc/hosts.

Si queremos, desde "**Ubuntu-1**", hacer una petición al servidor web alojado en el servidor "**Srv-web**", al que previamente hemos deshabilitado la dirección IPv6, sólo tenemos que ejecutar el siguiente comando:

```
# w3m Srv-web.escuela.upm.es
```

El resultado de la petición se muestra en la siguiente figura:

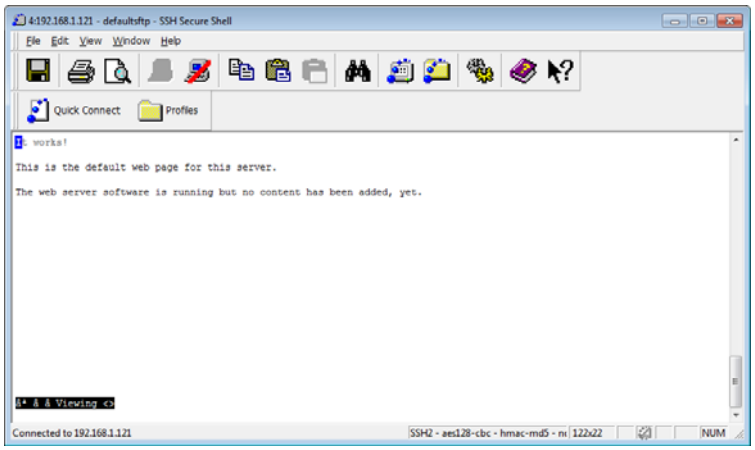

Figura 21. Petición de una página web a "Srv-web" desde "Ubuntu-1"

Otra forma de hacer la comprobación es hacer un **ping6** desde "**Ubuntu-1**" hasta el "**Srv-web**" y vemos que los paquetes llegan:

```
# ping6 Srv-web.escuela.upm.es
PING Srv-web.escuela.upm.es (Srv-web.escuela.upm.es) 56 data bytes
64 bytes from Srv-web.escuela.upm.es: icmp_seq=1 ttl=62 time=5.11 ms
64 bytes from Srv-web.escuela.upm.es: icmp_seq=2 ttl=62 time=1.19 ms
64 bytes from Srv-web.escuela.upm.es: icmp_seq=3 ttl=62 time=1.06 ms
^C
--- Srv-web.escuela.upm.es ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 1.060/2.456/5.117/1.882 ms
#
```

Mientras hacemos el **ping6** podemos ver los paquetes que llegan a "**Srv-web**" y veremos que son paquetes IPv4 cuya dirección origen es la 10.0.4.2, que es la dirección IPv4 de "**Srv-nat64**":

```
# tcpdump -i eth1 | grep 10.0.4.2
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
22:27:32.005634 IP 10.0.4.2 > 10.0.3.2: ICMP echo request, id 50183, seq 1, length 64
22:27:32.005650 IP 10.0.3.2 > 10.0.4.2: ICMP echo reply, id 50183, seq 1, length 64
22:27:33.006634 IP 10.0.4.2 > 10.0.3.2: ICMP echo request, id 50183, seq 2, length 64
22:27:33.006649 IP 10.0.3.2 > 10.0.4.2: ICMP echo reply, id 50183, seq 2, length 64
22:27:32.005634 IP 10.0.4.2 > 10.0.3.2: ICMP echo request, id 50183, seq 1, length 64
22:27:32.005650 IP 10.0.3.2 > 10.0.4.2: ICMP echo reply, id 50183, seq 1, length 64
22:27:33.006634 IP 10.0.4.2 > 10.0.3.2: ICMP echo request, id 50183, seq 2, length 64
22:27:33.006649 IP 10.0.3.2 > 10.0.4.2: ICMP echo reply, id 50183, seq 2, length 64
```

6 Conclusiones

La necesidad de migrar hacia el nuevo protocolo de Internet IPv6 es cada vez más inminente, especialmente en entornos educativos tecnológicos, si éstos no quieren frenar su avance en investigación y docencia. Las Escuelas de la UPM deben afrontar este reto y los Servicios Informáticos de los diferentes Centros deben estar preparados para llevarlo a cabo.

Gracias a las herramientas de virtualización es posible realizar una simulación del posible escenario con el que el personal de los diferentes centros de cálculo de la Universidad se va a encontrar a corto o medio plazo. Estas técnicas de virtualización, utilizadas de forma conveniente, pueden suponer un importante ahorro de costes en la implementación del sistema, pues permiten identificar y resolver los problemas de la transición de IPv4 a IPv6 en un entorno prácticamente idéntico al real. En este escenario los técnicos pueden trabajar con IPv6, diseñar sus estrategias de transición y hasta configurar los servicios con ambos protocolos, para que en el momento del cambio real al nuevo protocolo, todo haya sido analizado y estudiado y de esta manera los errores que se puedan producir sean mínimos.

VNX/VNUML es una herramienta de virtualización de propósito general flexible y que simplifica el trabajo del usuario ofreciéndole una gran potencia pero sin tener que enfrentarse a los detalles complejos de creación de máquinas y redes virtuales. De esta forma, el usuario puede concentrarse en la definición de la simulación, que es lo realmente importante desde su punto de vista.

El trabajo realizado en este Trabajo Fin de Máster junto con el realizado en el Trabajo Fin de Máster de Rafael García García, refleja los principales pasos a seguir en cualquier Escuela de la UPM para conseguir que los servicios básicos demandados por la Comunidad Universitaria, funcionen tanto en IPv4 como en IPv6 durante el periodo de transición al nuevo protocolo, utilizando en ellos una configuración de doble pila de protocolos. También se refleja en estos dos Trabajos un posible mecanismo para comunicar las redes sólo-IPv6, que se puedan crear en las Escuelas con los sistemas IPv4 que probablemente todavía perdurarán durante un tiempo.

Tras el trabajo realizado, los autores de estos dos Trabajos Fin de Máster podemos concluir que el uso de VNX/VNUML para simular la transición de IPv4 a IPv6 resulta muy adecuado ya que es una herramienta que permite, sin necesidad de una gran inversión económica, poder conocer cómo funciona IPv6, configurar los servicios que

se deben ofrecer, resolver los problemas con los que nos vamos a encontrar cuando se lleve a cabo el despliegue y tener definido un escenario prácticamente idéntico al que vamos a encontrarnos en el momento de la transición.

7 Referencias

[1]L. SEAWRIGHT Y R. MACKINNON, "VM/370: A STUDY OF MULTIPLICITY AND USEFULNESS", IBM SYSTEMS JOURNAL, 18(1), 1979.

[2] WWW.XEN.ORG.

[3] www.vmware.com

[4] J. DIKE, "USER MODE LINUX", PROC. 5TH ANNUAL LINUX SHOWCASE & CONF., OAKLAND CA,2001.

[5] HTTP://WWW.GNS3.NET/

[6] HTTP://WWW.NETKIT.ORG/

[7] HTTP://WWW.MARIONNET.ORG/

[8] "VNX HOME PAGE", HTTP://WWW.DIT.UPM.ES/VNX

[9] T. BRAY, J. PAOLI, C. M. SPERBERG-MCQUEEN, E. MALER, "EXTENSIBLE MARKUP LANGUAGE(XML) 1.0 (SECOND EDITION)", W3C RECOMMENDATION, OCTUBRE 2000.

[10] "THE PERL DIRECTORY", http://www.perl.org.

[11] "THE LINUX KERNEL ARCHIVES", http://www.kernel.org/

[12] "802.1Q: VIRTUAL LANS", IEEE 802.1Q WORKING GROUP, IEEE, 2001.

[13] TECNOLOGÍA XML EN LA HERRAMIENTA DESIMULACIÓN DE REDES VNUML, FERMÍN GALÁN, 2004

[14] UPM UNIVERSITY MIGRATION TO IPV6, T. DE MIGUEL Y D. FERNÁNDEZ, MAR 2003.

[15] HTTP://ECDYSIS.VIAGENIE.CA