# Trabajo de Grado

## “Aplicaciones multicast sobre IPv6”


Autor
Matías Robles

Director
Ing. Luis Marrone

Co -Director
Lic. Miguel Luengo

Facultad de Informática
Universidad Nacional de La Plata

2003

*...A mis padres*

Agradezco a:

Ing. Luis Marrone y Lic. Miguel Luengo por el apoyo recibido en el desarrollo de esta Tesis

Facultad de Informática de la U.N.L.P. por la formación desinteresada.

... y a mis seres queridos por la paciencia en todos estos años.

# Indice

# Figuras

# Prefacio

Desde hace tiempo se viene hablando del colapso de Internet. En la actualidad vemos una red con una calidad de servicio deteriorada, con frecuentes congestionamientos y con grandes problemas para implementar aplicaciones innovadoras que requieren un conjunto de nuevas herramientas que están siendo desarrolladas. Entre éstas están: videoconferencia, educación a distancia, servicios interactivos (por ej. TV interactiva), etc.

Los problemas planteados anteriormente no son los únicos a los que debe enfrentarse la Internet actual. A ellos, por ejemplo, habría que sumarle la falta de direcciones disponibles para asignar. Por éste y otros motivos, se ha pensado en un nuevo protocolo IP: IPv6.

Si se observan los tipos de aplicaciones en las que se está trabajando, es evidente, que éstas deberán hacer uso del multicast para lograr un mejor desempeño.

El trabajo de investigación que se presenta a continuación tiene como objetivo principal implementar una red IPv6 y mostrar el funcionamiento de distintas aplicaciones multicast sobre ella.

En el Capítulo 1 se describen las características básicas de IPv6 y de los demás protocolos usados por éste.

Como la red IPv6 utilizada para realizar este trabajo es la primera en la Facultad de Informática de La Plata, en el Capítulo 2 se realiza una explicación detallada del funcionamiento de la misma. Además de indicar los pasos necesarios para instalar y configurar una red de este tipo, mediante el uso de herramientas de diagnóstico (TCPDump, Snoop, etc.) se estudia el protocolo en acción para entender mejor como trabaja.

En el Capítulo 3, se explica, cuales son las características de la tecnología multicast en IPv6 y se analiza el formato de los mensajes del protocolo Multicast Listener Discovery para IPv6.

En el último capítulo, el Capítulo 4, se analiza el funcionamiento de las distintas aplicaciones multicast ejecutándolas en distintos escenarios de prueba. Además se muestran y analizan los mensajes intercambiados entre ellas.

En un apartado se detallan las conclusiones, se realiza un análisis comparativo entre IPv4 e IPv6, y se expone porque IPv6 es un muy buen complemento para Internet2.

Además, éste trabajo contiene tres apéndices:

Apéndice A, se explica como se instala, configura y prueba la nueva versión de IP en distintos sistemas operativos.

Apéndice B: se detalla como se crea una dirección EUI-64 (de 64 bits de longitud) a partir de una dirección 48 bits.

Apéndice C: explica, brevemente, los protocolos de ruteo multicast.

En el último apartado se encuentra la bibliografía consultada, con una breve explicación de lo que se puede consultar en cada uno de los libros o artículos.

# Capítulo 1
# IPv6

## 1. - Introducción

Desde que fue publicado en 1981, el protocolo IPv4 no ha sufrido grandes modificaciones. En más de 20 años en uso, ha demostrado ser flexible, robusto y poderoso. Sin embargo, ha comenzado a mostrar ciertas limitaciones para adecuarse al funcionamiento de las redes actuales y sus nuevas demandas. El crecimiento exponencial que ha tenido Internet en esta última década y, el advenimiento de nuevas tecnologías, han provocado la aparición de ciertas condiciones que el diseño original no anticipó (obviamente, no se equivocaron los diseñadores de IPv4, si no que la tecnología ha avanzado mucha más rápido de lo esperado), y que algunas de ellas se enumeran a continuación:

- Escasez de direcciones IPv4 libres para otorgar
- Imposibilidad de los routers del backbone de Internet de mantener largas tablas de ruteo
- Inexistencia de una manera simple de configuración de direcciones
- Carencia de un buen método para el envío de tráfico en tiempo real (conocido como Quality of Service – QOS)
- Falta de un mecanismo de seguridad en la capa de red.

Pensando en solucionar estos problemas y tratando de anticipar los nuevos avances tecnológicos es que se ha diseñado un nuevo protocolo: IPv6. Las siguientes son sus características más sobresalientes:

- Formato de cabecera simplificado
- Espacio de direcciones más grande (128 bits)
- Direccionamiento e infraestructura de ruteo eficiente y jerárquica
- Configuración de direcciones con y sin estado
- Seguridad intrínseca en el núcleo del protocolo
- Mejor soporte para la Calidad de Servicio
- Nuevo protocolo para la interacción entre nodos vecinos
- Extensibilidad
- Multicast (envío de un mismo paquete a un grupo de receptores)
- Anycast (envío de un paquete a un receptor dentro de un grupo)
- Posibilidad de enviar paquetes con más de 65.535 bytes (jumbogramas)
- Renumeración y multi-homing, que facilita el cambio de proveedor de servicios de Internet
- Características de movilidad

## 2. - Cabecera IPv6

El formato de la cabecera IPv6 es el siguiente:

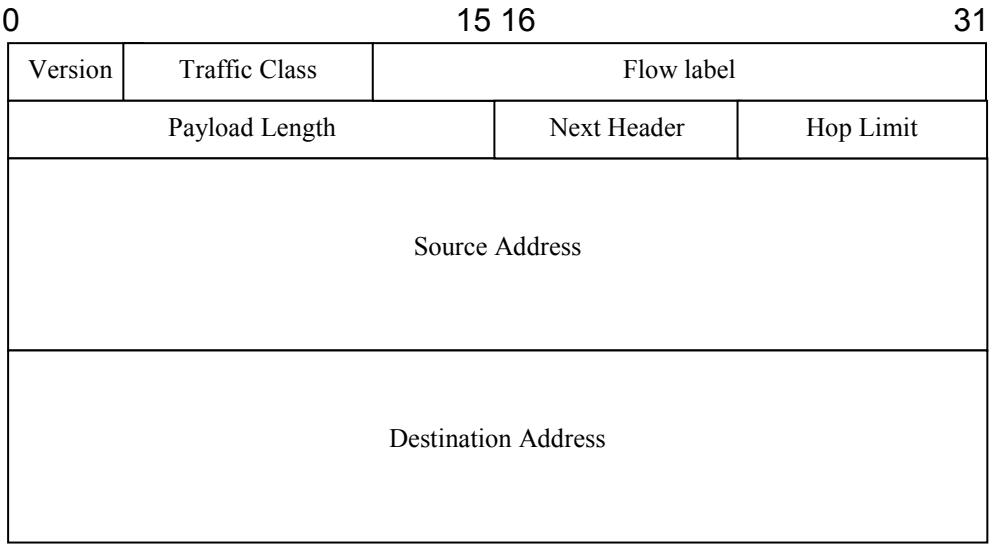


Figura 1.1 - Formato cabecera IPv6

La siguiente lista describe la función de cada campo en la cabecera:

- Version(4 bits): indica la versión del protocolo
- Traffic Class(8 bits): indica la clase o prioridad del paquete IPv6. Reemplaza al campo Type of Service de IPv4
- Flow Label(20 bits): indica que el paquete pertenece a una misma secuencia de paquetes entre un origen y un destino, requiriendo un manejo especial por parte de los routers intermedios.
- Payload Length(16 bits): indica la longitud de los datos después de la cabecera IPv6.
- Next Header(8 bits): indica cual es la cabecera de extensión siguiente (si existe) o el protocolo de capa superior (TCP o UDP). Reemplaza al campo Protocol Type de IPv4.
- Hop Limit(8 bits): indica la cantidad de routers por los que un paquete IPv6 puede pasar antes de ser descartado. Reemplaza al campo Time-to-Live (TTL) de IPv4.
- Source Address(128 bits): indica la dirección origen del emisor del paquete.
- Destination Address(128 bits): indica la dirección del nodo destino del paquete. (Nota: este campo puede no contener la dirección IPv6 del último destino si la cabecera de extensión Routing Header está presente)

El tamaño de las direcciones es de 128 bits (4 veces más grande que en IPv4), pero la longitud de la cabecera IPv6 es de 40 bytes, el doble del tamaño de la cabecera IPv4 (sin las opciones). Esto se debe a que se eliminaron varios campos, de los cuales algunos se pasaron como cabeceras de extensión.

En IPv4, existe el campo Options que es opcional y que puede tener longitud variable. Por esto la cabecera de IPv4 contiene el campo Header Length para saber su longitud exacta. Este campo no es necesario en IPv6 porque el tamaño de la cabecera es fijo.

Un paquete IPv6 puede llevar cero, una ó más cabeceras de extensión. Éstas se encuentran a continuación de la cabecera IPv6 y son las siguientes:

- Hop-by-Hop Option Header
- Routing Header
- Fragment Header
- Destination Header
- Authentication Header
- Encrypted Security Payload Header

Cada cabecera es identificada por el campo Next Header de la cabecera anterior. A excepción de la cabecera Hop-by-Hop Option, que debe ser examinada y procesada por cada nodo a lo largo del camino del paquete, las demás, solamente, son procesadas por el nodo destino, respetando estrictamente, el orden en el que aparecen en el paquete.

La figura siguiente muestra como se forman los paquetes cuando no tienen ninguna cabecera de extensión, cuando tiene una y cuando tienen más de una. Además se indica el valor del campo Next-Header.

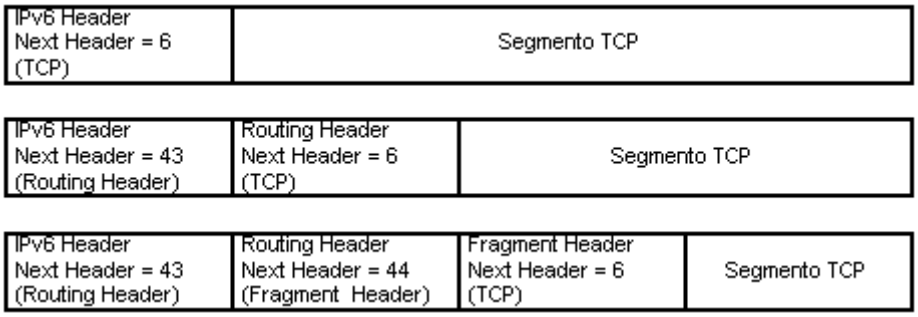


Figura 1.2 - Cabeceras de extensión

## 3.- Direcciones

Las direcciones son representadas como una serie de campos de 16 bits, hexadecimales, separados por dos puntos (:) en el formato x:x:x:x:x:x:x:x. Un doble dos puntos (::), se permite uno solo por dirección, puede ser usado para comprimir sucesivos campos hexadecimales de ceros.

En IPv6 no existen las direcciones broadcast, su función es sustituida por las direcciones multicast. Existe una dirección de loopback (::1) similar a la de IPv4, y se agrega un nuevo tipo de dirección: la dirección no especificada(::), que indica la ausencia de una dirección IPv6. Esta dirección no debe ser asignada a ninguna interface ni debe ser usada como dirección destino en un paquete IPv6.

Las direcciones se asignan a las interfaces, pueden tener más de una dirección asignada, y no a los nodos.

En IPv6, el concepto de máscara de red, es reemplazado por el concepto de prefijo. Éste es un número decimal que indica cuantos bits contiguos de más alto orden son usados para identificar la porción de red de la dirección. Una dirección IPv6 está compuesta de la siguiente manera:

*dirección IPv6 / prefijo*

### 3.1 Tipos de Direcciones

Existen 3 tipos de direcciones:

a) Unicast

Una dirección unicast es una dirección para una sola interface. Un paquete enviado a una dirección unicast es entregado, solamente, a la interface indicada por esa dirección.

Las direcciones unicast IPv6 son similares a las direcciones IPv4 con CIDR (Classless Inter-Domain Routing). Los siguientes son algunos tipos de direcciones unicast:

*Aggregatable Global Address*

Estas direcciones son utilizadas para el tráfico IPv6 a través de la Internet IPv6. Son similares a las direcciones unicast públicas utilizadas en IPv4 para comunicarse en Internet. Representan la parte más importante de la arquitectura de direcciones de IPv6. La figura siguiente muestra su estructura:

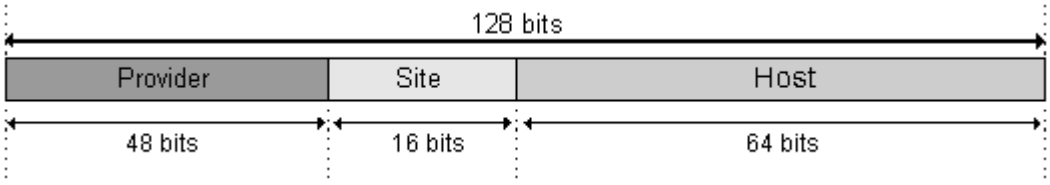


Figura 1.3 - Estructura dirección Unicast Aggregatable Global

- Provider: representa el prefijo de 48 bits cedido a una organización por algún proveedor autorizado.

- Site: con un /48 cedido a una organización, ésta puede manejar hasta 65.535 subredes diferentes. El sitio usa estos bits para subnetting.

- Host: esta parte, que representa los 64 bits de más bajo orden de la dirección, se llama *Interface ID*. Identifica una interface en un enlace. Debe ser único en ese enlace.

*Site Local Address*

Identificadas por el FP(Format Prefix) = 1111 1110 11. Estas direcciones son utilizadas dentro de una Intranet y son equivalentes al espacio de direcciones privadas de IPv4 (10.0.0.0/8, 172.16.0.0/12, y 192.168.0.0/16).

A diferencia de las direcciones de link-local, este tipo de direcciones no se configuran automáticamente y deben ser obtenidas a través de la autoconfiguración con o sin estado.

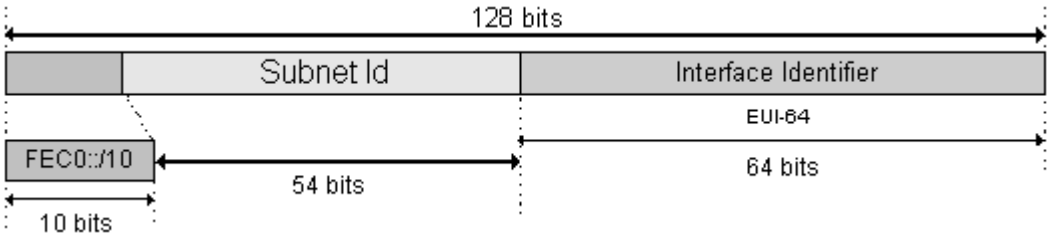


Figura 1.4 - Estructura dirección Site-Local

El prefijo para una dirección de site-local es FEC0::/10.

Un router nunca debe retransmitir tráfico de site-local fuera de un sitio, pero si lo puede hacer entre subredes dentro del mismo.

*Link Local Address*

Estas direcciones, identificadas por el FP = 1111 1110 10, son usadas para la autoconfiguración de direcciones, en funciones del Neighbor Discovery y cuando no existe un router en el link.

Cuando en una interface se habilita IPv6, la dirección de link-local es la primera dirección que se autoconfigura. Un nodo IPv6, no puede, no tener una dirección de este tipo asignada.

El alcance de estas direcciones es el link. Un router nunca debería reenviar un paquete con una dirección de link-local, como dirección origen o destino, más allá del mismo.

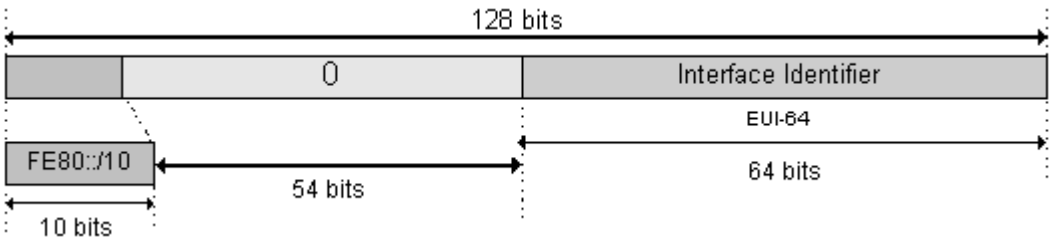


Figura 1.5 - Estructura dirección Link-Local

El prefijo para una dirección de link-local es FE80::/10.

Resumiendo, en IPv6, las direcciones tienen distinto alcance que es representado en el siguiente gráfico:

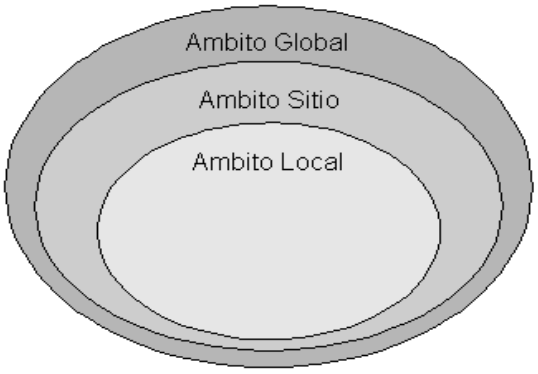


Figura 1.6 - Ambito direcciones unicast

b) Anycast Address

Identifica a un conjunto de interfaces, de tal manera, que al enviar un paquete a una dirección anycast es entregado a un solo miembro de ese grupo.

Estas direcciones son tomadas del espacio de direcciones unicast, es decir, que son sintácticamente indistinguibles una de las otras. Cuando se asignan a una interface se debe indicar explícitamente que la dirección es de tipo anycast.

c) Multicast Address

Identifica a un conjunto de interface, de tal modo, que un paquete enviado a una dirección multicast es entregado a todas las interfaces del grupo. Se identifican por el FP (Format Prefix) = 1111 1111, por lo cual, comienzan con el prefijo FF00::/8.

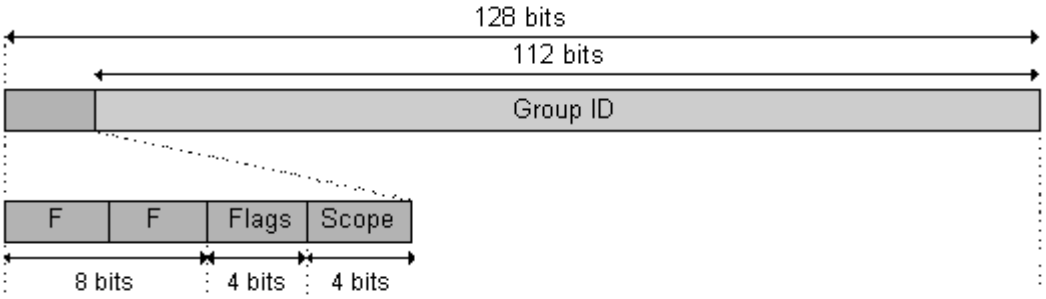


Figura 1.7 - Estructura dirección multicast

El campo Flags indica si una dirección multicast es permanente (0) o si es temporal (1).

El campo Scope limita el alcance de un grupo multicast.

Entre las direcciones multicast asignadas (permanentes) se encuentran:

- FF02::1/8 (dirección multicast de todos los nodos del link-local)
- FF02::2/8 (dirección multicast de todos los routers del link-local).

Existe otro tipo de dirección multicast, utilizada por el Neighbor Discovery, que es la *dirección multicast de nodo solicitado*. Esta dirección permite, a los nodos, un eficiente método de consulta durante el proceso de resolución de direcciones.

El prefijo de este tipo de direcciones es FF02:0:0:0:0:1:FFxx:xxxx/104, donde los últimos 24 bits son los últimos 24 bits de la dirección unicast o anycast que se está intentando resolver. Son utilizadas en los mensajes Neighbor Solicitation del Neighbor Discovery. El siguiente gráfico muestra la estructura de este tipo de direcciones:

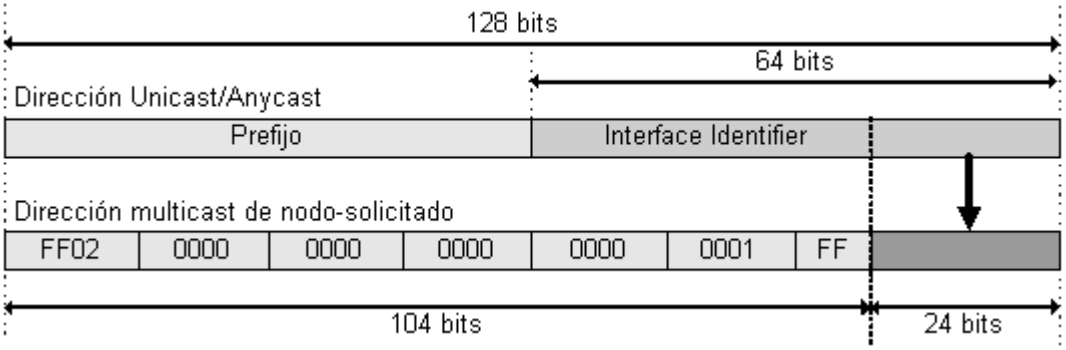


Figura 1.8 - Estructura dirección multicast de nodo solicitado

Todos los nodos (hosts y routers) se deben unir a las siguientes direcciones multicast:

- FF02::1/8
- FF02:0:0:0:0:1:FFxx:xxxx

Además, los routers se deben aceptar los mensajes enviados a la dirección FF02::2/8.

Una interface su puede unir a más de una dirección multicast.

### 3.2 Tiempo de vida de las direcciones

Las direcciones en IPv6 son asignadas a las interfaces por un período de tiempo determinado (puede ser infinito). Una dirección puede estar en uno de los siguientes estados:

- Tentativo: la dirección está en el proceso de verificación de su unicidad. Un nodo no debe aceptar paquetes que tienen como dirección destino una dirección en este estado.
- Válido: una dirección a la que se le ha probado su unicidad. Este estado cubre los estados preferido y desaconsejado.

- Preferido: indica el período de tiempo en el que una dirección puede ser usada en forma segura para enviar y recibir tráfico.
- Desaconsejado: el tiempo de vida preferido ha expirado pero la dirección todavía es válida. No se aconseja para establecer nuevas comunicaciones, pero las existentes pueden continuar usándola.
- Inválido: el tiempo de vida válido expiró y no se puede enviar ni recibir tráfico utilizando esa dirección.

La figura muestra como van cambiando los estados de una dirección a través del tiempo:

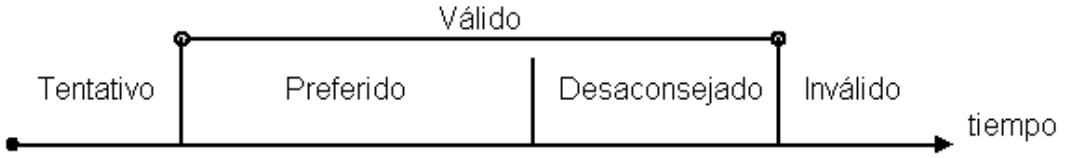


Figura 1.9 - Tiempo de vida de las direcciones

## 4. - ICMPv6

ICMP (Internet Control Message Protocolo) en IPv6 cumple las mismas funciones que ICMP en IPv4, es decir, que genera mensajes de error (como Destination Unreachable) y mensaje de información (como Echo Request).

Al igual que en IPv4, ICMPv6 debe ser parte integral de IPv6. Esto implica que se debe incorporar a cualquier implementación del protocolo.

Los mensajes de ICMPv6 se agrupan en dos tipos o clases: mensajes de error y mensajes informativos.

En IPv6, ICMPv6 tiene funciones adicionales. Por ejemplo, el proceso Neighbor Discovery, el PMTU y el protocolo Multicast Listener Discovery hacen uso de él para realizar su trabajo.

Un valor decimal de 58 en el campo Next Header de la cabecera IPv6 identifica a un paquete ICMPv6. Un paquete ICMPv6 se encuentra después de todas las cabeceras de extensión de IPv6.

## 5. - Neighbor Discovery

El Neighbor Discovery (ND) es un protocolo que corresponde a una combinación de protocolos de IPv4: el ARP (Address Resolution Protocol), el ICMP Router Discovery y el ICMP Redirect.

El ND posibilita que los nodos, en un mismo link, anuncien su existencia a sus vecinos, y aprendan acerca ellos. Resuelve un conjunto de problemas relacionados a la interacción entre nodos unidos al mismo link. En general, realiza tres funciones principales:

- Provee un mecanismo de resolución de direcciones (que reemplaza al ARP de IPv4)
- Permite a los hosts descubrir cuales son los routers vecinos que están presentes en el link y, provee mecanismos para permitirles obtener cierta información de configuración de ellos.
- Mediante el Neighbor Unreachability Detection (NUD), un host puede determinar cuando un vecino se vuelve inaccesible.

El ND está implementado dentro del ICMPv6, utiliza este protocolo para su funcionamiento, y define cinco diferentes tipos de paquetes:

- *Router Solicitation*: enviados por los hosts para pedirle a los routers que generen un anuncio de router inmediatamente.
- *Router Advertisement:* los routers anuncian su presencia, junto con varios parámetros del link y de Internet, para que los nodos se autoconfiguren.
- *Neighbor Solicitation:* enviado por un nodo para determinar la dirección de link-layer de un vecino, al verificar que una dirección es única en el link o al ejecutar el Neighbor Unreachability Detection.
- *Neighbor Advertisement:* es en respuesta a una solicitud de vecino. Un nodo, también, puede enviar un anuncio no solicitado (por ejemplo, para anunciar el cambio de su dirección de link-layer).
- *Redirect:* usado por los routers para informar a los hosts de un mejor primer salto a un destino o que éste es un vecino.

### 5.1 - Resolución de Direcciones

El proceso de resolución de direcciones consiste en el intercambio de mensajes Neighbor Solicitation y Neighbor Advertisement para resolver la dirección de link-layer de un nodo vecino.

Un nodo envía un Neighbor Solicitation a la dirección multicast de nodo-solicitado, que deduce de la dirección IPv6 asignada al nodo que se está intentando conocer su dirección de link-layer. El mensaje incluye, como opciones, la dirección de link-layer del nodo emisor (para que el destino sepa a donde contestar y no tenga que ejecutar este proceso nuevamente), y la dirección IPv6 del nodo consultado. Al enviarse la solicitud a la dirección multicast de nodo solicitado, únicamente los nodos que hayan mapeado a esta dirección serán interrumpidos para procesar el mensaje. En IPv4, el protocolo ARP hace este trabajo, pero al utilizar direcciones broadcast todos los nodos en el enlace son interrumpidos.

El nodo, que tiene asignada la dirección IPv6 consultada, le constesta con un mensaje Neighbor Advertisement, en el cual incluye su dirección de link-layer como una opción.

Ambos nodos deben actualizar sus caches con la información recibida en los mensajes intercambiados.

No debe ejecutar este proceso para resolver direcciones multicast.

### 5.2 - Neighbor Unreachability Detection (NUD)

Un nodo vecino es accesible si existe una confirmación, reciente, de que los paquetes IPv6 que se le enviaron, los ha recibido y procesado. No garantiza que se tenga acceso al nodo destino porque el vecino puede ser un router, que podría no ser el destinatario final.

Un modo de confirmar la accesibilidad, es mediante el envío de un mensaje Neighbor Solicitation a la dirección unicast del vecino y la posterior recepción de un mensaje Neighbor Advertisement *solicitado* (con el flag Solicited seteado a 1). Si el anuncio es no solicitado, no puede tomarse como una prueba de que el vecino es alcanzable.

El otro método es mediante la información intercambiada entre los protocolos de capas superiores, que indica, que la comunicación (que usa la dirección que se quiere saber si es accesible como primer salto), está progresando. Por ejemplo, en TCP se pueden utilizar los acuses de recibo ya que si hay acceso al destino final también lo hay hasta el primer salto (TCP le debe indicar esto al módulo IP).

### 5.3 - Router Discovery

Router Discovery es el proceso, a través del cual, los nodos intentan descubrir que routers se encuentran en el link. Es similar al ICMP Router Discovery de IPv4.

Los hosts envían mensajes Router Solicitation para encontrar un router en el link, y los routers envían Router Advertisement para anunciar su presencia, junto con otra información de configuración.

Los hosts utilizan los Router Advertisement recibidos para construir una lista de *routers default*, a los cuales le puede enviar tráfico. Estos anuncios también contienen información utilizada por los hosts para configurar direcciones, para determinar el MTU (Maximum Transmission Unit) del link, etc.

### 5.4 – Duplicate Address Detection (DAD)

Para asegurar que todas las direcciones sean únicas en un link, los nodos ejecutan el proceso Duplicate Address Detection antes de asignar una dirección a una interface (permanece en estado tentativo mientras dura el proceso). Este procedimiento es ejecutado sobre todas las direcciones unicast, independientemente de si son obtenidas, por autoconfiguración con estado o sin estado.

El DAD envía un mensaje Neighbor Solicitation a la dirección no especificada (::) porque la dirección que está siendo consultada no puede ser utilizada hasta que se compruebe que no está duplicada.

Si no está siendo utilizada por un nodo vecino, no habrá ninguna respuesta a esa solicitud. En caso contrario, el nodo que la tiene asignada le responderá con un Neighbor Advertisement a la dirección multicast de todos los nodos. En esta situación, el nodo no podrá utilizar la dirección.

### 5.5 - Estructura de un host

Uno de los principios en los cuales el diseño de IPv6 está basado es que los hosts deben trabajar correctamente aún si tienen una visión muy limitada de la red. Durante el inicio, un host se debe autoconfigurar, y luego debe aprender un mínimo de información acerca de los destinos con los cuales intercambiará datos. Esta información está almacenada en memoria en un conjunto de pequeñas estructuras llamadas *caches,* y es válida por un período de tiempo limitado. Estas estructuras son arreglos de registro, y cada registro es referido como una entrada.

Se definen cuatro tipos de caches diferentes:

- Neighbor Cache: contiene una entrada por cada vecino a los cuales el nodo le ha enviado tráfico recientemente.
- Destination Cache: contiene una entrada por cada destino a los cuales el nodo le ha enviado tráfico recientemente. La diferencia con la Neighbor Cache es que contiene entradas tanto para destinos on-link como off-link.
- Prefix List: contiene una entrada por cada prefijo on-link, y es utilizado para determinar si una dirección es on-link o no.
- Default Router List: contiene una entrada por cada router que puede ser utilizado como router default.

Los estados de las entradas en la Neighbor Cache pueden ser uno de los siguientes:

- Incomplete: la entrada ha sido creada, pero la dirección de link-layer no ha sido determinada todavía porque la resolución de direcciones está en progreso
- Reachable: se sabe que la entrada ha sido accedida recientemente.
- Stale: no se sabe si la entrada ha sido accedida recientemente, pero hasta que no se le envíe nuevo tráfico al vecino, ningún intento para comprobar su accesibilidad debería ser realizado.
- Delay: no se sabe si la entrada ha sido accedida recientemente, y se ha enviado tráfico al vecino. En este estado, los mensajes Neighbor Solicitation (probe) son retenidos para permitirle a los protocolos de capa superior confirmar la accesibilidad del vecino.
- Probe: la accesibilidad al vecino es muy incierta, y un mensaje probe ha sido enviado.

## 6. - Autoconfiguración de Direcciones Sin Estado

Un host toma varios pasos para decidir como autoconfigurar sus direcciones en IPv6. El proceso de autoconfiguración incluye crear una

dirección de link-local y verificar su unicidad en el link, determinar que información debe ser autoconfigurada (direcciones, otra información o ambas) y, en el caso de las direcciones, si deben ser obtenidas a través del mecanismo con estado, el mecanismo sin estado o ambos.

La autoconfiguración de direcciones sin estado no requiere de una configuración manual de los hosts. Esto permite a los hosts generar sus propias direcciones, usando una combinación de información disponible localmente e información anunciada por los routers. Estos anuncian prefijos que indican la/s subredes asociadas con un link, mientras que los hosts generan identificadores de interfaces que identifican únicamente a una interface en un link (o en un alcance mayor). Una dirección es formada por una combinación de los dos. En ausencia del router, un host solamente puede formar una dirección de link-local en forma automática.

El siguiente gráfico muestra el proceso de autoconfiguración de direcciones sin estado:

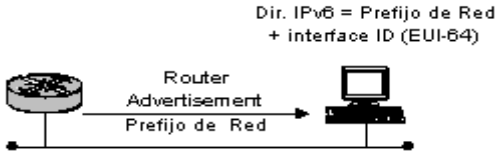


Figura 1.10 - Configuración de direcciones sin estado

Nota: existe otra forma de autoconfiguración, autoconfiguración con estado: DHCPv6.

## 7. - IPv6 en las capas superiores

Todos los protocolos de capa superior que incluyan las direcciones de la cabecera IP para calcular su checksum deberán ser modificados para usar direcciones IPv6 de 128 bits.

En UDP, a diferencia de IPv4 donde es opcional, el cálculo del checksum es obligatorio ya que la cabecera IPv6 no contiene este campo.

## 8. - Descubrir el Path MTU (Maximum Transmision Unit)

Como los routers no manejan la fragmentación, ésta es realizada por el nodo origen de un paquete. El proceso de descubrir el path MTU permite conocer el máximo MTU existente entre el nodo origen, quien ejecuta éste procedimiento antes de enviar el primer paquete, y el nodo destino. Conociendo este valor, los emisores pueden fragmentar los paquetes para enviarlos por ese path.

## 9. - Mecanismos de transición

La migración de IPv4 a IPv6 en un solo día es imposible, debido al inmenso tamaño de Internet y al número de usuarios de IPv4. Por esto, no existe un día especial, en el cual, IPv4 se "*apagará"* y se "*encenderá"* IPv6, si no, que será un proceso largo y paulatino, nodo por nodo. No existe un

coordinador global ni un orden específico de actualización (por ejemplo, no es necesario actualizar los routers de borde de un sitio antes que los nodos internos).

Ambos protocolos convivirán durante mucho tiempo. La integración y la coexistencia con IPv4 es un requisito para permitir la transición gradual hacia IPv6. Existe, un conjunto de mecanismos, que pueden implementar los hosts y routers IPv6 con el fin de ser compatibles con los nodos IPv4.

Los siguientes puntos describen algunas de las soluciones disponibles para posibilitar la convivencia de ambos protocolos. Cada uno de ellas tiene un conjunto de atributos que son específicos para resolver un problema determinado.

### 9.1 - Dual-Stack

La solución de *dual-stack* consiste en proveer a los hosts y los routers un soporte completo para los protocolos IPv4 e IPv6. Cada nodo es configurado con ambos protocolos, con lo cual, pueden interactuar con nodos IPv4 usando mensajes IPv4, y con nodos IPv6 usando paquetes IPv6.

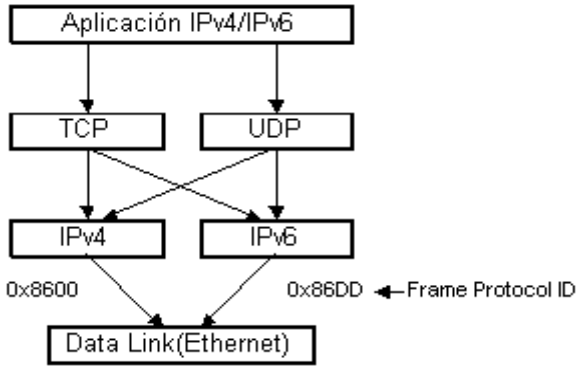


Figura 1.11 - Nodo Dual-Stack

Las aplicaciones deberán ser actualizadas para hacer uso del protocolo IPv6.

Como desventaja, esta solución no resuelve el problema de la falta de direcciones IPv4 públicas cuando es utilizada sola. Cada máquina, en la red, requiere direcciones públicas IPv4 e IPv6.

### 9.2 - Túneles

Mientras la infraestructura de ruteo para IPv6 está siendo desarrollada, ésta continuará basada en IPv4. La técnica del túnel permite que redes IPv6 aisladas se puedan comunicar sin necesidad de actualizar la estructura de ruteo IPv4 entre ellas. Los nodos extremos del túnel deben ser *dual-stack.*

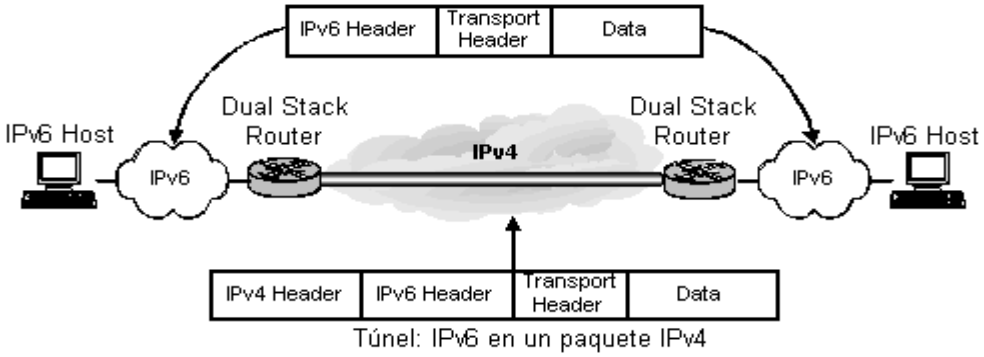


Figura 1.12 -Túnel IPv4 sobre IPv6

La figura 1.12 muestra como se componen los paquetes IPv6, encapsulados por un router dual-stack, para enviarlos por una red IPv4. El router origen del túnel le agrega una cabecera IPv4, en la cual, las direcciones origen y destino son las correspondientes a las de inicio y fin del túnel. El router, donde finaliza el túnel, es el encargado de desencapsular el paquete IPv6 (eliminando la cabecera IPv4) y retransmitirlo al destino final.

Los túneles pueden ser configurados manualmente o automáticamente. Estos difieren, principalmente, en el método por el cual el nodo encapsulador determina la dirección final del túnel.

### 9.2.1 - Túneles Manuales

En este tipo de túneles, una dirección IPv6 configurada manualmente se ingresa en la interface del túnel y, también, manualmente, se asignan las direcciones IPv4 al punto de inicio y el punto destino del túnel.

La dirección final del túnel corresponde a un router, no a la del destinatario final del paquete. No existe ninguna relación entre ellas, por lo que la dirección debe ser ingresada manualmente. El nodo encapsulador no puede determinar la dirección IPv4 del final del túnel a partir de las direcciones IPv6 del paquete original.

Nota: en el punto 2.2, del Capítulo 2, se explica como se configura un túnel manual.

### 9.2.2 - Túneles Automáticos

Los túneles automáticos permiten a los nodos enviar tráfico IPv6 por una red IPv4 sin tener que preconfigurar, manualmente, un túnel.

La dirección final del túnel corresponde al nodo destinatario del paquete: son iguales. Al utilizar direcciones IPv6 compatibles con IPv4, el nodo encapsulador puede determinar la dirección IPv4, del final del túnel, automáticamente de la dirección IPv6, y por lo tanto no se necesita configuración manual de los extremos del túnel.

El siguiente gráfico muestra como se forman estas direcciones, donde los últimos 32 bits corresponden a la dirección IPv4 del nodo destino.

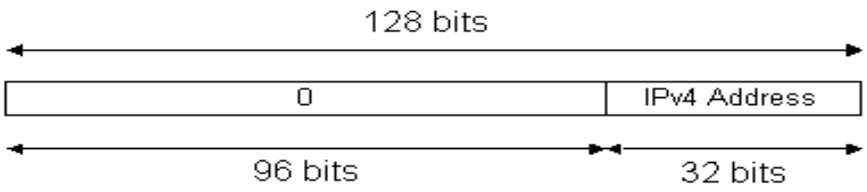


Figura 1.13 - Estructura dirección IPv4-compatible IPv6

Una dirección IPv4-compatible es globalmente única siempre que la dirección IPv4 no pertenezca al espacio de direcciones privadas de IPv4.

En el gráfico a continuación se muestra como un host que soporta este tipo de direcciones, al enviar un mensaje a la dirección ::A319:021E, el nodo encapsulador determina la dirección IPv4, 163.25.2.30, automáticamente de la dirección IPv6.

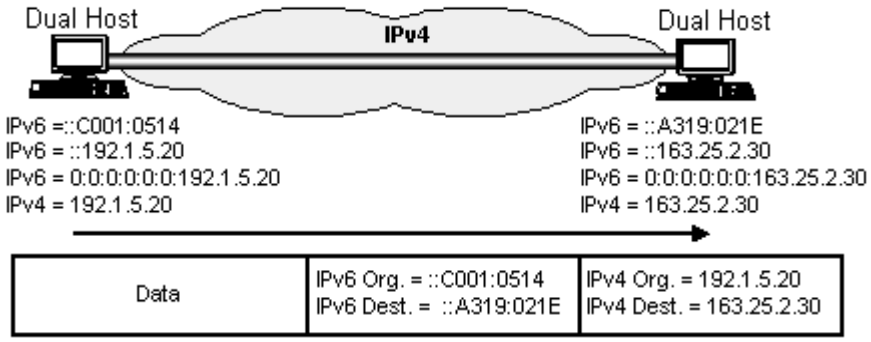


Figura 1.14 - Túneles Automáticos

### 9.2.3 - Túneles 6to4

Este método es un tipo de túnel automático *router-to-router*. Comienzan con el prefijo 2002::/16, y los siguientes 32 bits son ocupados por la dirección IPv4 del router. Esto permite construir un prefijo /48, con lo cual la organización dispone de los siguientes 16 bits para administrar.

Para poder comunicarse por Internet, la dirección IPv4 debe ser única globalmente.

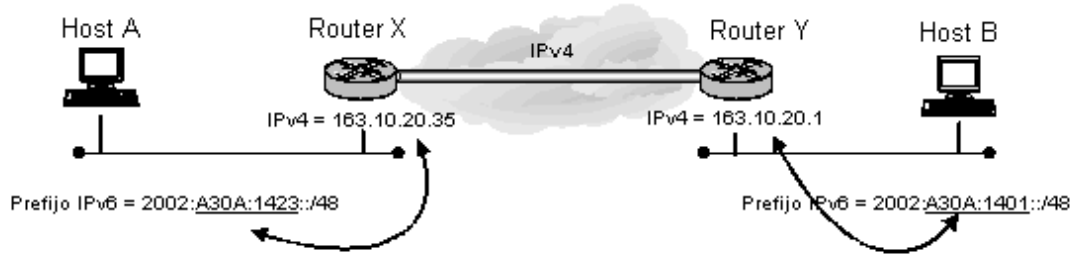


Figura 1.15 - Túnel 6to4

Un router utiliza la dirección IPv4, embebida en la dirección IPv6, para encontrar el otro extremo del túnel.

Para establecer este tipo de túneles, cada dominio IPv6 requiere un router dual-stack, que automáticamente construya el túnel. Se recomienda que cada router tenga una sola dirección 6to4 asignada a su interface externa. Dentro del sitio se puede utilizar un protocolo de ruteo IPv6, fuera de este se continúan utilizando los protocolos de ruteo IPv4.

Difieren de los túneles automáticos explicados en el punto anterior, en que este método permite formar prefijos de red y direcciones para un único host (aunque se recomienda que sea utilizados para configurar prefijos), mientras que el anterior, únicamente permite formar direcciones asignables a un nodo en particular.

# *Capítulo 2*
# *Implementación Red IPv6*

## 1. - Introducción

En el transcurso del año 2002 se decidió comenzar con la implementación de una red que fuese capaz de soportar la nueva versión de IP, IPv6. Además de que funcionase en el laboratorio, se consideró importante poder lograr conexión con la red de prueba de IPv6, conocida como el 6Bone.

La red inicial estaba armada de la siguiente manera:

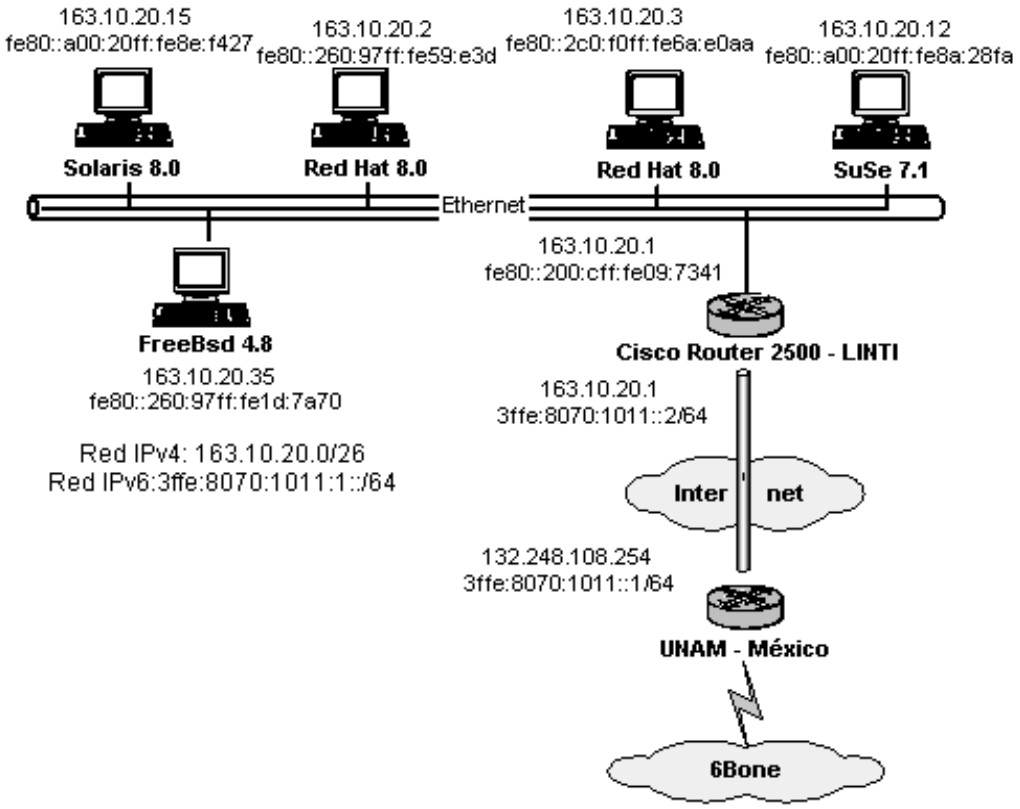


Figura 2.1 - Diseño red IPv6 LINTI (con un solo router)

Todos los sistemas operativos instalados soportan IPv6. El IOS del router Cisco es la versión 12.8, que también soporta el protocolo (en un principio se utilizó la versión 11.2, que era experimental). Con esto, la red IPv6 ya estaba funcionado (gracias a la capacidad de IPv6 de autoconfigurarse) a nivel LAN, pero faltaban configurar ciertos parámetros en el router para poder salir al 6bone.

Para poder realizar la conexión al 6Bone se debía conseguir un prefijo de red global. Este prefijo se solicitó a la UNAM (Universidad Autónoma de

México). Ellos cuentan con un prefijo /28 (3ffe:8070:1011::/28), y nos cedieron un /48: 3ffe:8070:1011::/48.

Como la comunicación con el router de la UNAM es a través de una infraestructura IPv4, se configuró un túnel entre el router del LINTI y el de la UNAM. El siguiente gráfico muestra la conexión:


Figura 2.2 - Conexión al 6bone a través de la UNAM

Ingresando a la página de la UNAM (www.ipv6.unam.mx), se puede ver la conexión de la UNLP al 6bone a través de esta universidad. El siguiente gráfico se encuentra en dicha página:

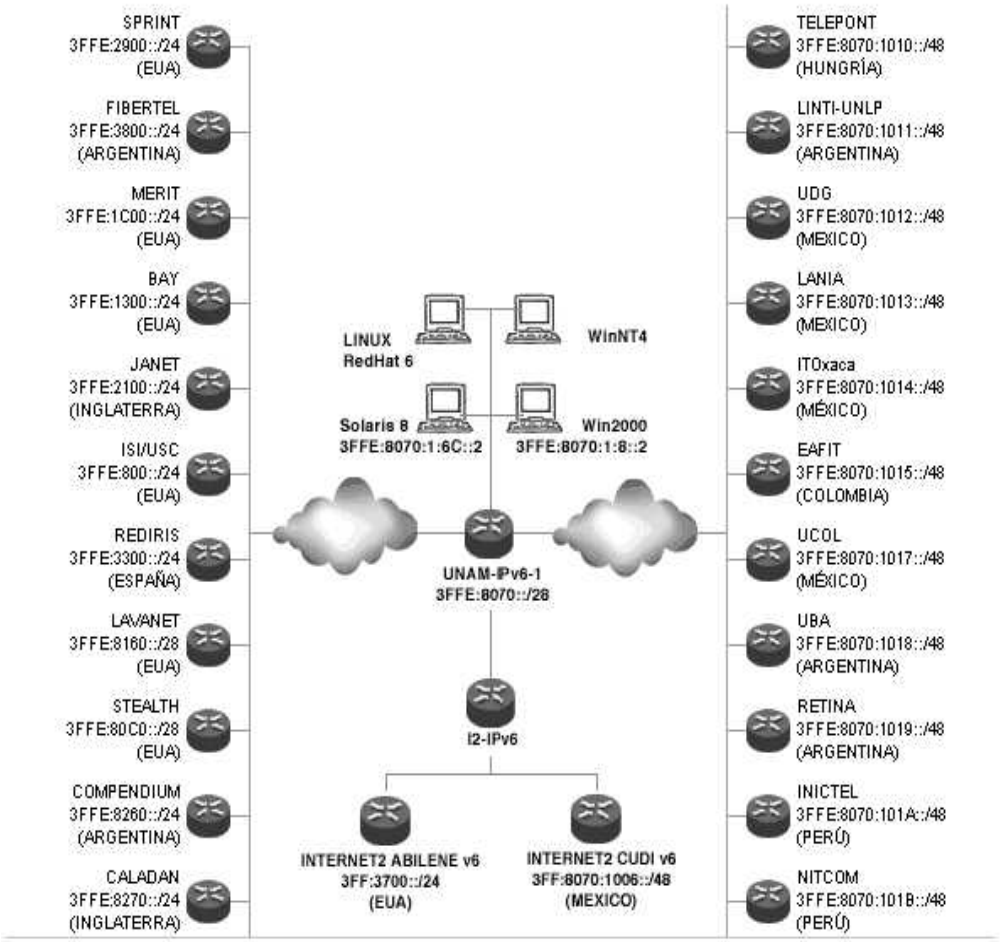


Figura 2.3 - Gráfico página UNAM

En la izquierda del gráfico se encuentran los TLA (Top-Level Aggregators), que representan el primer nivel de Internet Service Providers(ISP).

En la derecha del gráfico se observan algunos de los NLA (Next Level Aggregator), que corresponden al siguiente nivel en la jerarquía, cedidos por la UNAM para permitir el acceso al 6bone, entre los que se encuentra el LINTI.

Al cedernos un prefijo /48, la UNAM nos permitió disponer de un total de $2^{16}$ subredes, es decir, un total de 64.564 subredes distintas. De esto se desprende el siguiente rango de subredes posibles:

3ffe:8070:1011:0::/64
3ffe:8070:1011:1::/64
.....
.....
3ffe:8070:1011:65.564::/64

Si observamos la conformación de una dirección IPv6, los 16 bits del prefijo de red sobre los que tenemos autoridad son los que corresponden al Site Id.

Para nuestra LAN, se decidió usar la subred 1, con lo cual uno de los prefijos que publica el router quedó conformado así:

3ffe:8070:1011:1::/64

## 2. - Configuración del router (Cisco 2500)

No es necesario que el router publique un prefijo de red. Tampoco que exista un router. Una red IPv6 funciona correctamente aun cuando no existe un router. Los nodos con IPv6 habilitado generan una dirección de link-local, comienzan con el prefijo FF80::/10, que pueden utilizar para comunicarse entre ellos, pero no para comunicarse con un nodo fuera de esa red. Para poder salir de la red local, es necesario obtener una dirección de site-local o global. Estas direcciones pueden ser ingresadas manualmente en cada nodo, o configuradas automáticamente, utilizando un prefijo publicado por el router, para los primeros 64 bits de la dirección IPv6, e información obtenida localmente, en cada nodo, para los últimos 64 bits. Esto se conoce como autoconfiguración de direcciones sin estado. A continuación se muestran los pasos necesarios para configurar un router IPv6.

A diferencia de IPv4, que está habilitado por defecto, el reenvío de paquetes IPv6 está deshabilitado, hay que habilitarlo explícitamente. Primero se lo debe hacer globalmente, y luego se deben asignar las direcciones IPv6, a cada interface, individualmente.

Para habilitar el reenvío de tráfico IPv6 se utiliza el siguiente comando en modo configuración:

Router(config)**# ipv6 unicast-routing**

Una vez habilitado el reenvío globalmente, una dirección IPv6 debe ser configurada en una interface para que ésta sea capaz de reenviar tráfico IPv6. Al configurar una dirección de *site-local* o global en una interface, automáticamente se configura la dirección de *link-local.* Además, la interface configurada, automáticamente, se une a las siguientes direcciones multicast requeridas para el correcto funcionamiento de IPv6:

- Dirección multicast de nodo solicitado para cada dirección unicast o anycast asignada a la interface (FF02:0:0:0:01:FF00::/104)
- Dirección multicast de todos los nodos (FF02::1. Su alcance es el link-local)
- Dirección multicast de todos los routers (FF02::2. Su alcance es el link-local)

El comando para configurar una dirección IPv6 en una interface y habilitar el procesamiento IPv6 es:

Router(config-if)**#ipv6 address** ipv6 address-prefix [eui-64]

Si la opción eui-64 está especificada, el router debe usar el identificador de interface, en los 64 bits menos importantes para formar su dirección IPv6 (en al apéndice B se explica como se forma este identificador). En este caso solo se indica el prefijo de red. El comando sería el siguiente:

Router(config-if)**#ipv6 address** 3ffe:8070:1011:2:: eui-64

Si la opción no está especificada, se debe indicar una dirección IPv6 completa:

Router(config-if)**#ipv6 address** 3ffe:8070:1011:2::1/64

Otra opción disponible es especificar la opción **link-local**. Mediante ésta, se ingresa una nueva dirección de link-local para la interface que es usada en lugar de la que es configurada automáticamente cuando se habilita IPv6 en la interface. Obviamente, el prefijo de esta dirección debe ser FE80::/10.

Para habilitar el procesamiento en una interface, que no ha sido configurada con una dirección IPv6 explícita, se puede usar el comando, en el modo de configuración de la interface, **ipv6 enable.** Automáticamente se crea una dirección IPv6 de link-local.

Una vez que se habilitó el procesamiento en la interface, ésta comienza a enviar mensajes de Router Advertisement y es capaz de contestar Router Solicitations.

## 2.1 - Configuración del Neighbor Discovery

Por defecto, los prefijos configurados en una interface utilizando el comando **ipv6 address** son incluidos en los anuncios del router. Estos prefijos son anunciados con los flags *on-link* y *autoconfig* seteados. El flag *on-link* indica que el prefijo es asignado al link. Los nodos que envían tráfico a las direcciones que contienen el prefijo especificado consideran, al destino, como un vecino. El flag *autoconfig* le indica, a los hosts de link, que el prefijo especificado puede ser utilizado para la autoconfiguración de IPv6.

Si se configura un prefijo manualmente para ser anunciado, el router no publica ninguno de los prefijos autoconfigurados con el comando **ipv6 address**. El comando para configurar un prefijo es el siguiente:

```
Router(config-if)#ipv6 nd prefix-advertisement ipv6-prefix/prefix length
                    valid-lifetime preferred-lifetime [onlink] [autoconfig]
```

Por ejemplo, podemos configurar el siguiente prefijo, con los valores que se indican, seteados:

```
Router(config-if)#ipv6 nd prefix-advertisement 3ffe:8070:1011:1::/64
                                  1000 900 onlink autoconfig
```

Si el tiempo de vida preferido es mayor al tiempo de vida válido, el router generará el siguiente error:

```
IPv6(config-if)#ipv6 nd prefix 3ffe:8070:1011:1::/64 1000 ?
  <0-4294967295>  Preferred Lifetime (secs)
  infinite        Infinite Preferred Lifetime

IPv6(config-if)#ipv6 nd prefix 3ffe:8070:1011:1::/64 1000 2000
% Preferred Lifetime must not exceed Valid Lifetime
```

Además, de los prefijos, existen otras propiedades del Neighbor Discovery que pueden ser configuradas para facilitar la autoconfiguración de los hosts vecinos.

El router puede indicarle a los hosts si la configuración de direcciones es con estado o sin estado (o ambas). Si el flag *managed-config-flag* está seteado, los host deben utilizar la autoconfiguración de direcciones con estado (DHCPv6). Si no, la autoconfiguración es sin estado. El comando para setear este flag es:

```
Router(config-if)#ipv6 nd managed-config-flag
```

Los hosts puede utilizar, simultáneamente, la autoconfiguración con y sin estado. Seteando el flag other-config-flag, se le indica a los hosts que pueden utilizar la autoconfiguración con estado para obtener otra información (distinta de las direcciones). El comando es el siguiente:

```
Router(config-if)#ipv6 nd other-config-flag
```

El valor del tiempo de vida del router indica el tiempo, que los hosts, consideran a este router como *router default*. Si está seteado a 0, indica que el router no debe ser considerado como tal. Este valor es configurado con el siguiente comando:

Router(config-if)**#ipv6 nd ra-lifetime** seconds

A continuación se setea el tiempo de vida del router a 900 segundos:

Router(config-if)**#ipv6 nd ra-lifetime** 900

Una propiedad importante es el intervalo de tiempo entre dos transmisiones de mensajes Router Advertisement consecutivos. Un valor muy pequeño implicaría que el router esté, continuamente, enviando este tipo de mensajes, ocupando innecesariamente ancho de banda del enlace. Este intervalo no es fijo, si no, que el router elige un valor random dentro del 20% del valor seteado.

El comando para configurar este valor es:

Router(config-if)**#ipv6 nd ra-interval** seconds

Por ejemplo, lo establecemos a 5 segundos:

Router(config-if)**#ipv6 nd ra-interval** 5

El valor configurado en este parámetro debe ser menor o igual al tiempo de vida del router. Si es mayor, el router genera un error. Esto es para evitar que, en los host, se venza el tiempo de vida de los routers antes que arribe un nuevo anuncio. De esta manera, un nodo no se queda sin un router default. El siguiente ejemplo muestra el error generado por el router:

```
IPv6(config-if)#ipv6 nd ra-lifetime 10
IPv6(config-if)#ipv6 nd ra-interval 20
% Router Lifetime must be greater than or equal to the Router Advertisement
interval
IPv6(config-if)#
```

Después que un nodo obtiene una confirmación de accesibilidad de un vecino, no lo considera accesible para siempre, si no que lo hace por un período de tiempo determinado. Si ese tiempo es pequeño, el nodo puede detectar la falta de acceso al vecino más rápidamente, pero consume más ancho de banda en la red y recursos de procesamiento al tener que ejecutar el proceso de Detección de Accesibilidad a un Vecino más seguido. El comando para establecer este tiempo es el siguiente:

Router(config-if)**#ipv6 nd reachable-time** milisegundos

Por ejemplo, lo seteamos a 6000 milisegundos:

Router(config-if)**#ipv6 nd reachable-time** 6000

Se puede indicar la cantidad de mensajes Neighbor Solicitation, consecutivos, que el router debe enviar cuando está ejecutando el proceso de Detección de Direcciones Duplicadas. Se recomienda que sean dos como mínimo, para solucionar el problema de la pérdida de paquetes por la red.

Router(config-if)**#ipv6 nd dad attemps** attemps

Por ejemplo, lo seteamos a 2:

Router(config-if)**#ipv6 nd dad attemps** 2

Además de establecer el intervalo de tiempo entre dos Router Advertisement consecutivos, también podemos hacer lo mismo con los mensajes Neighbor Solicitations consecutivos. Este valor se incluye en los anuncios del router, y el comando para configurarlo es el siguiente:

Router(config-if)**#ipv6 nd ns-interval** milisegundos

Por ejemplo, podemos setearlo al siguiente valor:

Router(config-if)**#ipv6 nd ns-interval** 3000

Con el siguiente comando podemos setear la cantidad máxima de routers que podrán atravesar los paquetes generados por todos los nodos del link:

Router(config-if)**#ipv6 hop-limit** value

Por ejemplo, podemos establecerlo al siguiente valor:

Router(config-if)**#ipv6 hop-limit** 64

En todos los medios de transmisión existe un tamaño máximo de transmisión (MTU) por defecto. Por ejemplo, en Ethernet el MTU es de 1500 bytes. En IPv6, el MTU no puede ser menor a 1280 bytes. Mediante el siguiente comando se pude modificar el MTU por default:

Router(config-if)**#ipv6 mtu** bytes

Por ejemplo, podemos setearlo al siguiente valor:

Router(config-if)**#ipv6 mtu** 1400

En la primera versión del IOS (11.2) que se probó, la opción MTU solo se incluía en un anuncio de router si su valor era distinto al valor default, o si, se seteaba nuevamente el valor default. En la nueva versión, esta opción siempre se incluye. Este último funcionamiento es más adecuado con la especificación de la RFC 2450 (Neighbor Discovery). Ésta dice que si dos o más routers anuncian los mismos datos, el valor del último anuncio es el que vale. Por lo cual, si el router no anuncia el MTU (que si o si tiene configurado), los hosts no pueden actualizar el MTU.

El siguiente gráfico representa el envío de mensajes Router Advertisement en un link:

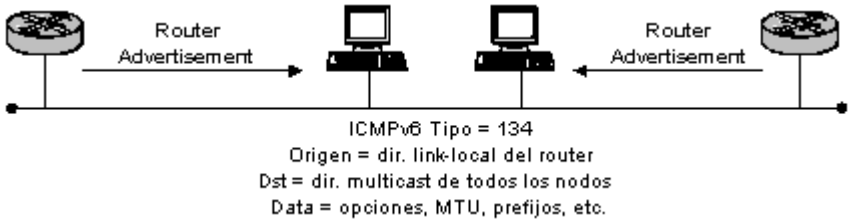


Figura 2.4 - Router Advertisement

A continuación se muestra como se conforman dichos paquetes:

```
ETHER:  ----- Ether Header -----
ETHER:
ETHER:  Packet 55 arrived at 17:05:17.73
ETHER:  Packet size = 142 bytes
ETHER:  Destination = 33:33:0:0:0:1, (multicast)
ETHER:  Source      = 0:0:c:9:73:41, Cisco
ETHER:  Ethertype = 86DD (IPv6)
ETHER:
IPv6:   ----- IPv6 Header -----
IPv6:
IPv6:   Version = 6
IPv6:   Traffic Class = 112
IPv6:   Flow label = 0x0
IPv6:   Payload length = 88
IPv6:   Next Header = 58 (ICMPv6)
IPv6:   Hop Limit = 255
IPv6:   Source address = fe80::200:cff:fe09:7341
IPv6:   Destination address = ff02::1
IPV&:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
ICMPv6:  Type = 134 (Router advertisement)
ICMPv6:  Code = 0
ICMPv6:  Checksum = 5eab
ICMPv6:  Max hops= 64, Router lifetime= 900
ICMPv6:  Managed addr conf flag:NOT SET,Other conf flag:NOT SET
ICMPv6:  Reachable time: 6000, Reachable retrans time 3000
ICMPv6:
ICMPv6:  +++ ICMPv6 Source LL Addr option +++
ICMPv6:  Link Layer address: 0:0:c:9:73:41
ICMPv6:
ICMPv6:  +++ ICMPv6 MTU option +++
ICMPv6:  MTU = 1400
ICMPv6:
ICMPv6:  +++ ICMPv6 Prefix option +++
ICMPv6:  Prefix length = 64
ICMPv6:  Onlink flag: SET, Autonomous addr conf flag: SET
ICMPv6:  Valid Lifetime 900, Preferred Lifetime 1000
ICMPv6:  Prefix 3ffe:8260:10c:8000::
ICMPv6:
ICMPv6:  +++ ICMPv6 Prefix option +++
ICMPv6:  Prefix length = 64
ICMPv6:  Onlink flag: SET, Autonomous addr conf flag: SET
ICMPv6:  Valid Lifetime 900, Preferred Lifetime 1000
ICMPv6:  Prefix 3ffe:8070:1011:1::
```

Este paquete tiene los siguientes parámetros para ser analizados:

- ETHER: Destination = 33:33:0:0:0:1, (multicast)
  El paquete es enviado a la dirección multicast de Ethernet. Los primeros 2 bytes tienen el valor hexadecimal 3333, y los últimos 4 bytes corresponden a los últimos 4 bytes de la dirección IPv6.

- ETHER: Ethertype = 86DD (IPv6)
  Con este nuevo valor, en el campo EtherType de la cabecera Ethernet, se indica que se encapsula un paquete IPv6.

- IPv6: Version = 6
  En la cabecera del paquete IP se indica que la versión es 6

- IPv6: Next Header = 58 (ICMPv6)
  Indica que el paquete a continuación es ICMPv6

- IPv6: Destination address = ff02::1
  El Router Advertisement es no solicitado, esto implica que se envía a la dirección multicast de todos los nodos.

- ICMPv6: Type = 134 (Router advertisement)
  Indica que el mensaje es un anuncio de router.

- ICMPv6: MTU = 1400
  Indica la Máxima Unidad de Transferencia del link.

Además, el anuncio lleva toda la información, que seteamos, como opciones. Por ejemplo, este anuncio informa dos prefijos de red con sus correspondientes parámetros seteados.

## 2.2 - Túneles

La comunicación con la UNAM, para poder acceder al 6bone, se realizó a través de la infraestructura de ruteo IPv4. Por esto, hubo que utilizar uno de los mecanismos de transición de IPv4 a IPv6: túneles configurados manualmente.

La UNAM nos dió toda la información necesaria para establecer el túnel: la dirección IPv4 y las direcciones IPv6 que se utilizarán en los extremos. Los pasos para establecer el túnel con la UNAM son los siguientes:

En el modo configuración global del router se debe especificar la interface del túnel y asignarle un número:

Router(config)# **interface tunel** tunel-number

El siguiente paso es setear una dirección IPv6 y habilitar el procesamiento de IPv6 en la interface:

Router(config-if)# **ipv6 address** ipv6-address/prefix-length [eui-64]

La dirección IPv6 utilizada es la dirección asignada por ellos:

Router(config-if)# **ipv6 address** 3ffe:8070:1011::2/64

A continuación se especifica una dirección IPv4 y su máscara de red:

Router(config-if)# **ip address** ip-address mask

Como no se puso una dirección IPv4, entonces se utilizó el siguiente comando:

Router(config-if)# **no ip address**

Los pasos siguientes son agregar las direcciones origen y destino del túnel:

Router(config-if)# **tunnel source** {ip-address|interface-type interface-numbre}

Router(config-if)# **tunnel source** 163.10.20.1

Con el siguiente comando indicamos la dirección final del túnel:

Router(config-if)# **tunnel destination** {ip-address}

Ponemos la dirección de la UNAM, que nos fue indicada por ellos, como dirección de destino:

Router(config-if)# **tunnel destination** 132.248.108.254

Por último se debe especificar que el túnel es manual:

Router(config-if)# **tunnel mode ipv6ip**

El túnel con la UNAM quedó configurado de la siguiente manera:

```
interface Tunnel1
description Tunel con la UNAM
no ip address
ipv6 address 3FFE:8070:1011::2/64
tunnel source 163.10.20.1
tunnel destination 132.248.108.254
tunnel mode ipv6ip
```

El siguiente gráfico muestra como quedó el túnel establecido entre el LINTI y la UNAM a través de Internet, y las direcciones seteadas en cada uno de los routers:

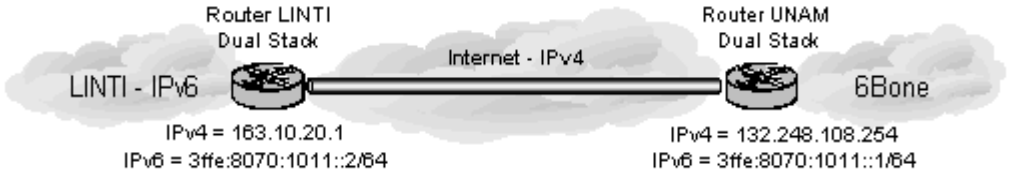


Figura 2.5 - Túnel LINTI - UNAM

## 2.3 - Configuración de ruteo

El último paso para poder conectarnos al 6bone era habilitar el ruteo del tráfico IPv6 hacia el router de la UNAM. Para esto, se configuró el protocolo BGP4+, que es similar al BGP4, pero permite realizar su trabajo basado en direcciones IPv6.

Al no existir clases en IPv6, el ruteo es como en IPv4 con CIDR, y se utiliza el prefijo de direcciones, en lugar de la máscara de red, para saber cual es la parte de la red en una dirección.

El siguiente ejemplo muestra como quedó configurado el BGP4+:

```
router bgp 5692
no synchronization
bgp router-id 163.10.20.1
bgp cluster-id 2735392562
bgp log-neighbor-changes
neighbor 3FFE:8070:1011::1 remote-as 278
neighbor 3FFE:8070:1011::1 description BGP Contra la UNAM
no neighbor 3FFE:8070:1011::1 activate
no auto-summary
!
address-family ipv6
neighbor 3FFE:8070:1011::1 activate
neighbor 3FFE:8070:1011::1 soft-reconfiguration inbound
network 3FFE:8070:1011::/48
aggregate-address 3FFE:8070:1011::/48 as-set summary-only
no synchronization
redistribute connected
exit-address-family
!
```

Los puntos más interesantes para analizar son los siguientes:

- neighbor 3FFE:8070:1011::1 remote-as 278
  Indica al proceso BGP local la dirección IPv6 y el número de AS (Autonomous System) del router remoto de la conexión, que pertenece a la UNAM.

- address-family ipv6
  Indica que el router está en modo de configuración para la familia de direcciones de IPv6.

- neighbor 3FFE:8070:1011::1 activate
  Activamos al vecino para intercambiar la información de ruteo.

- network 3FFE:8070:1011::/48
  Indicamos la/s redes que queremos que BGP anuncie. La red anunciada es la que nos cedió la UNAM.

- aggregate-address 3FFE:8070:1011::/48 as-set summary-only
  El router anuncia el prefijo indicado únicamente, y no rutas más específicas.

## 2.4 - Configuración final

Mediante el siguiente comando podemos ver como quedaron configurados la interface y el túnel:

```
IPv6#show ipv6 interface
Ethernet0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::200:CFF:FE09:7341
  Global unicast address(es):
   3FFE:8070:1011:1::1, subnet is 3FFE:8070:1011:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF09:7341
    FF02::1:FF00:1
  MTU is 1400 bytes
  ICMP error messages limited to one every 1 milliseconds
  ND DAD is enabled, number of DAD attempts: 2
  ND reachable time is 6000 milliseconds
  ND advertised reachable time is 6000 milliseconds
  ND advertised retransmit interval is 3000 milliseconds
  ND router advertisements are sent every 30 seconds
  ND router advertisements live for 900 seconds
  Hosts use stateless autoconfig for addresses.
Tunnel1 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::A30A:1401
  Description: Tunel con la UNAM
  Global unicast address(es):
    3FFE:8070:1011::2, subnet is 3FFE:8070:1011::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF00:2
    FF02::1:FF0A:1401
  MTU is 1480 bytes
  ICMP error messages limited to one every 1 milliseconds
  ND reachable time is 30000 milliseconds
  Hosts use stateless autoconfig for addresses.
```

Podemos observar que el router, automáticamente, se unió a la dirección multicast de todos los nodos (ff02::1), a la dirección multicast de todos los routers (ff02::2) y a todas las direcciones multicast de nodo solicitado que le corresponden.

### 3. - Duplicate Address Detection (DAD)

Cuando se configura, automáticamente o no, una dirección IPv6 unicast en una interface, se debe verificar su unicidad dentro del link antes de asignarla. La nueva dirección permanece en estado tentativo hasta que finaliza la comprobación. En el gráfico 2.6, se muestra el intercambio de mensajes necesarios para realizar este proceso:

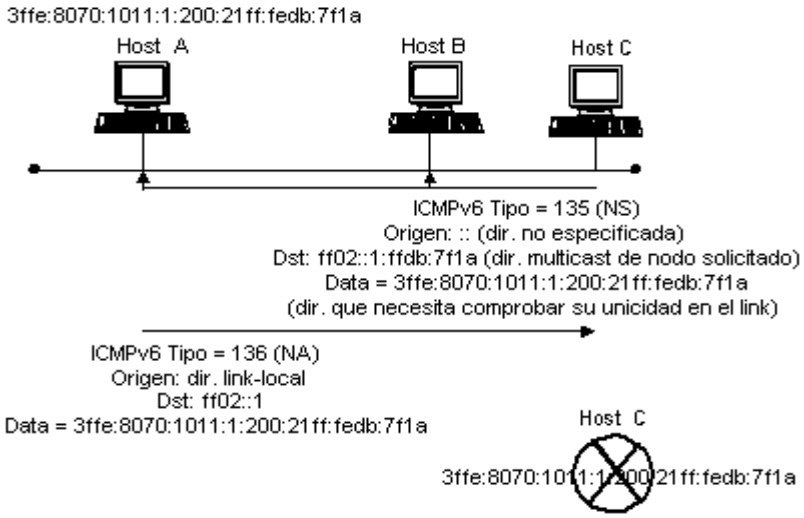


Figura 2.6 - Proceso asignación direcciones

El Neighbor Solicitation enviado por el nodo ejecutando el DAD es el siguiente:

```
ETHER:  ----- Ether Header -----
ETHER:
ETHER:  Packet 2 arrived at 16:54:42.00
ETHER:  Packet size = 78 bytes
ETHER:  Destination = 33:33:ff:db:7f:1a, (multicast)
ETHER:  Source      = 8:0:20:8e:f4:27, Sun
ETHER:  Ethertype = 86DD (IPv6)
ETHER:
IPv6:   ----- IPv6 Header -----
IPv6:
IPv6:   Version = 6
IPv6:   Traffic Class = 0
IPv6:   Flow label = 0x0
IPv6:   Payload length = 24
IPv6:   Next Header = 58 (ICMPv6)
IPv6:   Hop Limit = 255
IPv6:   Source address = ::, UNSPECIFIED
IPv6:   Destination address = ff02::1:ffdb:7f1a
IPv6:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
ICMPv6:  Type = 135 (Neighbor solicitation)
ICMPv6:  Code = 0
ICMPv6:  Checksum = 883c
ICMPv6:  Target node =
               3ffe:8070:1011:1:200:21ff:fedb:7f1a
```

Entre los campos para analizar están los siguientes:

- IPv6: Source address = ::, UNSPECIFIED
  Como se explicó anteriormente, la dirección :: es la dirección no especificada. Un nodo envía una solicitud de vecino con esta dirección como dirección origen, cuando está ejecutando el proceso de Detección de Direcciones Duplicadas.

- IPv6: Destination address = ff02::1:ffdb:7f1a
  Al no existir más las direcciones de broadcast, el mensaje es enviado a la dirección multicast de nodo solicitado. Unicamente, los nodos con una dirección que contengan el valor *db:7f1a* en los últimos 24 bits, aceptarán esta solicitud. En caso contrario, los nodos deberán descartar la solicitud silenciosamente.

- ICMPv6: Type = 135 (Neighbor solicitation)
  Indica que el mensaje es una Solicitud de Vecino.

- ICMPv6: Target node = 3ffe:8070:1011:1:200:21ff:fedb:7f1a
  El contenido de este campo es la dirección IPv6 para la cual el nodo está ejecutando el proceso de Detección de Direcciones Duplicadas.

Si en el link existe un nodo que ya tiene asignada dicha dirección, éste le contesta con un Neighbor Advertisement de la siguiente manera:

```
ETHER:  ----- Ether Header -----
ETHER:
ETHER:  Packet 3 arrived at 16:54:42.00
ETHER:  Packet size = 86 bytes
ETHER:  Destination = 33:33:0:0:0:1, (multicast)
ETHER:  Source      = 0:0:21:db:7f:1a, SC&C
ETHER:  Ethertype = 86DD (IPv6)
ETHER:
IPv6:   ----- IPv6 Header -----
IPv6:
IPv6:   Version = 6
IPv6:   Traffic Class = 0
IPv6:   Flow label = 0x0
IPv6:   Payload length = 32
IPv6:   Next Header = 58 (ICMPv6)
IPv6:   Hop Limit = 255
IPv6:   Source address = fe80::200:21ff:fedb:7f1a
IPv6:   Destination address = ff02::1
IPv6:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
ICMPv6:  Type = 136 (Neighbor advertisement)
ICMPv6:  Code = 0
ICMPv6:  Checksum = a2bd
ICMPv6:  Target node=3ffe:8070:1011:1:200:21ff:fedb:7f1a
ICMPv6:  Router flag:NOT SET, Solicited flag:NOT SET,
                      Override flag:SET
ICMPv6:
```

```
ICMPv6:  +++ ICMPv6 Target LL Addr option +++
ICMPv6:  Link Layer address: 0:0:21:db:7f:1a
```

A continuación se explican los campos remarcados:

- IPv6: Destination address = ff02::1
  El nodo, que tiene la dirección asignada, le contesta a la dirección multicast de todos los nodos. No tiene una dirección unicast a la cual contestarle. La solicitud fue enviada con la dirección no especificada como dirección origen.

- ICMPv6: Type = 135 (Neighbor advertisement)
  Indica que el mensaje es un Anuncio de Vecino.

- ICMPv6: Router flag:NOT SET, Solicited flag:NOT SET,
  Override flag:SET
  Estos flags se explican en el punto 5 de este capítulo.

- Target node = 3ffe:8070:1011:1:200:21ff:fedb:7f1a
  El campo Target Nodo del mensaje contiene la dirección que se está intentando saber si está duplicada.

- ICMPv6: Link Layer address: 0:0:21:db:7f:1a
  Como una opción, el nodo le envía su dirección de link-layer.

### 4. - Router Discovery

Cuando se comprobó la unicidad de la dirección IPv6 en el link, el siguiente paso es determinar si existe, un router, en el link, y así, el nodo puede obtener información para la autoconfiguración. A pesar que el router envía anuncios periódicamente, puede suceder (por ejemplo, cuando se habilita una interface) que el host no quiera esperar hasta que el router envíe el próximo anuncio programado. Si el intervalo entre dos anuncios consecutivos es muy grande, el host tendría que esperar demasiado tiempo para poder autoconfigurarse. Para acortar el tiempo de espera, el host puede enviar un mensaje Router Solicitation, al cual, el router le contestará con Router Advertisement.

La figura representa el intercambio de paquetes entre un router y el host que envía la solicitud:

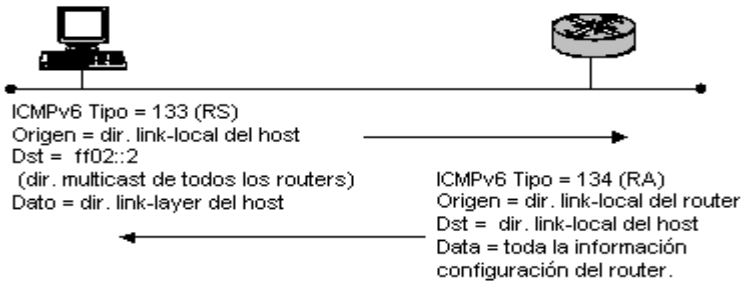


Figura 2.7 - Buscando un router

Los paquetes intercambiados en el proceso anterior se analizan a continuación:

```
ETHER:  ----- Ether Header -----
ETHER:
ETHER:  Packet 62 arrived at 17:06:0.21
ETHER:  Packet size = 70 bytes
ETHER:  Destination = 33:33:0:0:0:2, (multicast)
ETHER:  Source      = 8:0:20:8a:28:fa, Sun
ETHER:  Ethertype = 86DD (IPv6)
ETHER:
IPv6:   ----- IPv6 Header -----
IPv6:
IPv6:   Version = 6
IPv6:   Traffic Class = 0
IPv6:   Flow label = 0x0
IPv6:   Payload length = 16
IPv6:   Next Header = 58 (ICMPv6)
IPv6:   Hop Limit = 255
IPv6:   Source address = fe80::800:208a:28fa
IPv6:   Destination address = ff02::2
IPv6:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
ICMPv6:  Type = 133 (Router solicitation)
ICMPv6:  Code = 0
ICMPv6:  Checksum = d925
ICMPv6:  +++ ICMPv6 Source LL Addr option +++
ICMPv6:  Link Layer address: 8:0:20:8a:28:fa
```

Los puntos principales para analizar son los siguientes:

- IPv6: Destination address = ff02::2
  El mensaje es enviado a la dirección multicast de todos los routers.

- ICMPv6: Type = 133 (Router solicitation)
  Indica que el mensaje es una solicitud de router.

- ICMPv6: Link Layer address: 8:0:20:8a:28:fa
  El host envía su dirección de link-layer, como una opción, para que el router sepa a que dirección enviar la contestación, y no tenga necesidad de preguntar a la red cuál es la dirección de link-layer del nodo al que le tiene que responder.

Como el envío del anuncio es solicitado por un nodo en particular, el mensaje se envía a ese nodo únicamente (no a la dirección multicast de todos los nodos como en el caso anterior). Por esto, la Destination Address del Router Advertisement es igual a la Source Address del mensaje de solicitud. El Router Advertisement con el que contesta el router es el siguiente:

```
ETHER:  ----- Ether Header -----
ETHER:
ETHER:  Packet 63 arrived at 17:06:0:40
```

```
ETHER:  Packet size = 142 bytes
ETHER:  Destination = 8:0:20:8a:28:fa, Sun
ETHER:  Source      = 0:0:c:9:73:41, Cisco
ETHER:  Ethertype = 86DD (IPv6)
ETHER:
IPv6:   ----- IPv6 Header -----
IPv6:
IPv6:   Version = 6
IPv6:   Traffic Class = 112
IPv6:   Flow label = 0x0
IPv6:   Payload length = 88
IPv6:   Next Header = 58 (ICMPv6)
IPv6:   Hop Limit = 255
IPv6:   Source address = fe80::200:cff:fe09:7341
IPv6:   Destination address = fe80::800:208a:28fa
IPv6:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
ICMPv6:  Type = 134 (Router advertisement)
ICMPv6:  Code = 0
ICMPv6:  Checksum = 5eab
```

.....

Los siguientes campos son los más interesantes para analizar:

- ETHER: Destination = 8:0:20:8a:28:fa, Sun
  El router obtiene la dirección física del host destino, de la opción *link-layer address* del mensaje de solicitud de router.

- IPv6: Destination address = fe80::800:208a:28fa
  Como el anuncio del router es solicitado, el mensaje se envía a una dirección unicast, y no a la dirección multicast de todos los nodos.

- ICMPv6: Type = 134 (Router advertisement)
  Indica que el mensaje es un anuncio de router.

Los campos restantes son iguales a los que el router había enviado en el anuncio anterior.

## 5. - Resolución de Direcciones

Cuando un nodo se quiere comunicar con un vecino pero no conoce su dirección de link-layer, debe ejecutar el proceso de resolución de direcciones. El gráfico 2.8 muestra el intercambio de mensajes, donde el nodo A intenta saber la dirección de link-layer del nodo B.

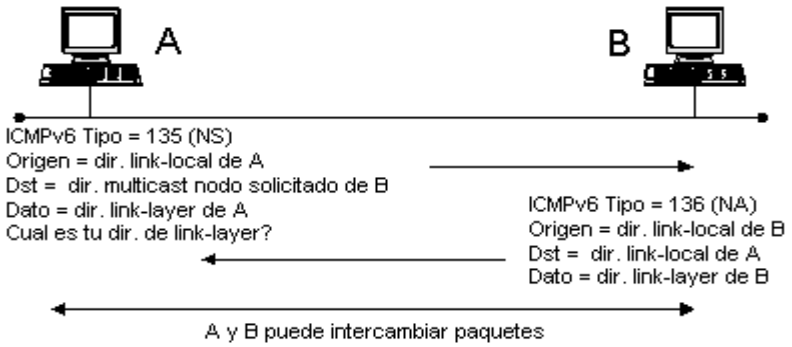


Figura 2.8 - Resolución de dirección de link-layer

A continuación se muestran los mensajes intercambiados entre los nodos:

```
ETHER: ----- Ether Header -----
ETHER:
ETHER:  Packet 1 arrived at 17:10:23.41
ETHER:  Packet size = 86 bytes
ETHER:  Destination = 33:33:ff:0:0:1, (multicast)
ETHER:  Source      = 8:0:20:8e:f4:27, Sun
ETHER:  Ethertype = 86DD (IPv6)
ETHER:
IPv6:   ----- IPv6 Header -----
IPv6:
IPv6:  Version = 6
IPv6:  Traffic Class = 0
IPv6:  Flow label = 0x0
IPv6:  Payload length = 32
IPv6:  Next Header = 58 (ICMPv6)
IPv6:  Hop Limit = 255
IPv6:  Source address =
                3ffe:8070:1011:1:a00:20ff:fe8e:f427
IPv6:  Destination address = ff02::1:ff00:1
IPv6:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
ICMPv6:  Type = 135 (Neighbor solicitation)
ICMPv6:  Code = 0
ICMPv6:  Checksum = 9e2f
ICMPv6:  Target node = 3ffe:8070:1011:1::1
ICMPv6:
ICMPv6:  +++ ICMPv6 Source LL Addr option +++
ICMPv6:  Link Layer address: 8:0:20:8e:f4:27
```

A continuación se explican los parámetros más importantes:

- ETHER: Destination = 33:33:ff:0:0:1, (multicast)
  El mensaje se envía a la dirección multicast de Ethernet derivada de la IPv6 Destination Address

- IPv6: Destination address = ff02::1:ff00:1
  El nodo envía la solicitud a la dirección multicast de nodo solicitado, que corresponde a la dirección IPv6.

- ICMPv6: Target node = 3ffe:8070:1011:1::1
  Se envía la dirección IPv6 que disparó este proceso. Como la IPv6 Destination Address es una dirección multicast de nodo solicitado, pueden existir varios receptores, entonces es necesario indicar la dirección unicast del nodo que se intenta conocer su dirección física.

- ICMPv6: Link Layer address: 8:0:20:8e:f4:27
  El nodo origen envía su dirección de link-layer para que el receptor sepa a quien constestar.

El nodo receptor contesta con el siguiente Neighbor Advertisement:

```
.....
IPv6:  Version = 6
IPv6:  Traffic Class = 112
IPv6:  Flow label = 0x0
IPv6:  Payload length = 32
IPv6:  Next Header = 58 (ICMPv6)
IPv6:  Hop Limit = 255
IPv6:  Source address = 3ffe:8070:1011:1::1
IPv6:  Destination address =
         3ffe:8070:1011:1:a00:20ff:fe8e:f427
IPv6:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
ICMPv6:  Type = 136 (Neighbor advertisement)
ICMPv6:  Code = 0
ICMPv6:  Checksum = 871e
ICMPv6:  Target node = 3ffe:8070:1011:1::1
ICMPv6:  Router flag: SET,Solicited flag: SET,
                Override flag: SET
ICMPv6:
ICMPv6:  +++ ICMPv6 Target LL Addr option +++
ICMPv6:  Link Layer address: 0:0:c:9:73:41
```

Las líneas remarcadas se explican en los siguientes puntos:

- IPv6: Destination address = 3ffe:8070:1011:1:a00:20ff:fe8e:f427
  El nodo utiliza como dirección IPv6 destino la dirección IPv6 origen del mensaje Neighbor Solicitation recibido.

- ICMPv6: Router flag: SET, Solicited flag: SET, Override flag: SET

  Router flag: SET
  Al estar seteado indica que el emisor del anuncio es un router.

  Solicited flag: SET
  Al estar seteado indica que el anuncio es en respuesta a un Neighbor Solicitation.

Override flag: SET
Al estar seteado indica que el anuncio debería sobreescribir la entrada en la cache y actualizar la dirección de link-layer

- ICMPv6: Link Layer address: 0:0:c:9:73:41
  El nodo le envía, a quien emitió la solicitud, su dirección de link-layer como una opción.

### 6. - Neighbor Unreachability Detection (NUD)

Los nodos almacenan en la *cache* si tienen accesibilidad, o no, a un nodo vecino. Recordemos que existen cinco estados posibles:

INCOMPLETE
REACH
STALE
DELAY
PROBE

Mediante el comando **show ipv6 neighbors,** en un router Cisco, podemos ver los nodos vecinos a los que recientemente hemos tenido acceso y, en cual, de los posibles estados, se encuentran. Por ejemplo, si lo ejecutamos obtenemos lo siguiente:

```
IPv6#show ipv6 neighbors
IPv6 Address                              Age  Link-layer Addr State Interface
3FFE:8260:10C:8000:A00:20FF:FE8E:F427  1405 0800.208e.f427  STALE Ethernet0
3FFE:8070:1011:1:A00:20FF:FE8E:F427     11  0800.208e.f427  STALE Ethernet0
FE80::A00:20FF:FE8E:F427                 7  0800.208e.f427  STALE Ethernet0
FE80::800:208A:28FA                   1361 0800.208a.28fa  STALE Ethernet0
```

El router ha accedido a estas cuatro direcciones IPv6 por la interface Ethernet0 recientemente, pero como el estado es STALE no nos indica si alguna de esas direcciones es, todavía, accesible.

Si enviamos un ping a una de esas direcciones, podemos ver como cambia su estado:

```
IPv6#ping
Protocol [ip]: ipv6
Target IPv6 address: 3FFE:8070:1011:1:A00:20FF:FE8E:F427
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands? [no]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to
3FFE:8070:1011:1:A00:20FF:FE8E:F427, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/4 ms
```

Si ejecutamos el comando show ipv6 neighbors nuevamente, se puede ver como se modificó el State de la dirección a la que se envió el ping:

```
IPv6#show ipv6 neighbors
IPv6 Address                              Age Link-layer Addr State Interface
3FFE:8260:10C:8000:A00:20FF:FE8E:F427    1406 0800.208e.f427  STALE Ethernet0
3FFE:8070:1011:1:A00:20FF:FE8E:F427         0 0800.208e.f427  REACH Ethernet0
FE80::A00:20FF:FE8E:F427                    8 0800.208e.f427  STALE Ethernet0
FE80::800:208A:28FA                      1362 0800.208a.28fa  STALE Ethernet0
```

La dirección 3FFE:8070:1011:1:A00:20FF:FE8E:F427 es accesible, el estado REACH así lo indica.

Los nodos intercambian mensajes Neighbor Solicitation y Neighbor Advertisement para realizar estas comprobaciones. El nodo que desea comunicarse con un vecino le envía un Neighbor Solicitation, quien le contesta con un Neighbor Advertisement.

El mensaje Neighbor Solicitation es, similar, al enviado en el proceso de resolución de direcciones. La única diferencia es que se envía a una dirección unicast (la del nodo que se quiere saber si es accesible) y no a la dirección multicast de nodo solicitado.

El nodo destino le constesta con un Neighbor Advertisement similar al anuncio mostrado en el punto anterior. No es necesario que envíe su dirección de link-layer como una opción, porque el nodo que envió la solicitud la conoce, si no, no hubiera podido enviar dicha solicitud (tendría que haber ejecutado el proceso de resolución de direcciones antes).

## 7. - Testeando la red

Al igual que en IPv4, la opción para comprobar si el protocolo IPv6 está funcionado correctamente, es mediante el comando *ping*. El emisor envía un *echo request* al destino, y éste le responde con un *echo replay*.

En el siguiente ejemplo, se ejecuta el comando ping desde una máquina Sun con SO Solaris hacia el router.

```
ping -s 3ffe:8070:1011::1

bash-2.03# snoop -v ip6
Using device /dev/hme (promiscuous mode)

.....
IPv6:   ----- IPv6 Header -----
IPv6:
IPv6:   Version = 6
IPv6:   Traffic Class = 0
IPv6:   Flow label = 0x0
IPv6:   Payload length = 64
IPv6:   Next Header = 58 (ICMPv6)
IPv6:   Hop Limit = 60
IPv6:   Source address = 3ffe:8070:1011:1:a00:20ff:fe8e:f427
IPv6:   Destination address = 3ffe:8070:1011::1
IPv6:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
ICMPv6:  Type = 128 (Echo request)
ICMPv6:  Code = 0 (ID: 1711 Sequence number: 0)
```

```
ICMPv6:  Checksum = 5284
```

En este paquete se observa un mensaje Echo Request, al que el router le contesta con el siguiente Echo Replay:

```
.....
IPv6:   ----- IPv6 Header -----
IPv6:
IPv6:   Version = 6
IPv6:   Traffic Class = 0
IPv6:   Flow label = 0x0
IPv6:   Payload length = 64
IPv6:   Next Header = 58 (ICMPv6)
IPv6:   Hop Limit = 63
IPv6:   Source address = 3ffe:8070:1011::1
IPv6:   Destination address =
                  3ffe:8070:1011:1:a00:20ff:fe8e:f427
IPv6:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
ICMPv6:  Type = 129 (Echo reply)
ICMPv6:  Code = 0 (ID: 1711 Sequence number: 0)
ICMPv6:  Checksum = 5184
```

## 8. - Redirect

Un mensaje Redirect es enviado por un router para avisarle a un nodo que existe un mejor primer salto para llegar al destino. Es decir, que el router le indica al host que el destino es un vecino o que debe usar otro gateway, más directo, para enviar los próximos paquetes a ese destino. El gráfico muestra como el Router B, al recibir un paquete para el host Y proveniente del host X, la indica a éste que existe un mejor camino para llegar a Y a través del router A.

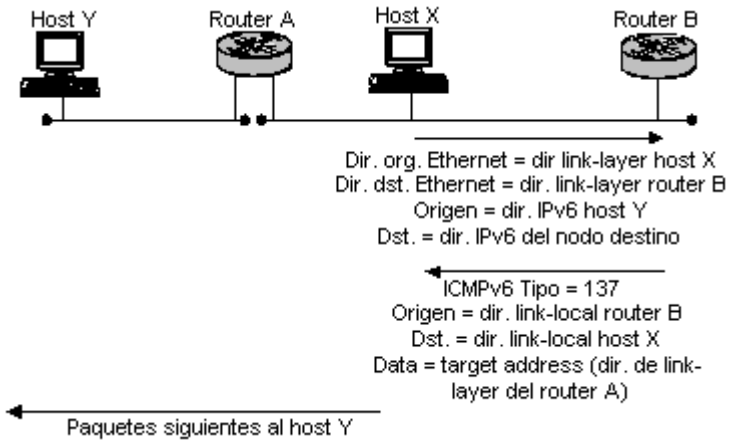


Figura 2.9 - Redirect

Para poder mostrar su funcionamiento agregamos una ruta en un host, que indica que para ir a un nodo vecino hay que enviar el mensaje a través del router. La ruta que se agregó es la siguiente (se hizo en Solaris 8.0):

```
bash-2.03# route add -inet6 3ffe:8070:1011:1::3/64 fe80::200:cff:fe09:7341
add net 3ffe:8070:1011:1::3/64
```

La tabla de ruteo del host quedó de la siguiente manera:

```
bash-2.03# netstat -rn
Routing Table: IPv6
Destination/Mask       Gateway                       Flags  Ref   Use  If
--------------------   --------------------------- ----- ----- ---- ------
....
3ffe:8070:1011:1::/64  fe80::200:cff:fe09:7341          UG    1    0
3ffe:8070:1011:1::/64  3ffe:8070:1011:1:a00:20ff:fe8e:f427   U     1    0   hme0:2
3ffe:8070:1011:1::/64  3ffe:8070:1011:1::5            U     1    0   hme0:1
default                fe80::260:97ff:fe1d:7a70       UG    1    0   hme0
default                fe80::200:cff:fe09:7341        UG    1    0   hme0
...
```

La línea remarcada dice que para ir a la red 3ffe:8070:1011:1::/64 hay que enviar los mensajes a través del router, como si fuese una dirección off-link.

Al ejecutar un ping a la dirección del nodo vecino, obtenemos la siguiente secuencia de mensajes:

```
bash-2.03# ping -a -s 3ffe:8070:1011:1::3
PING 3ffe:8070:1011:1::3: 56 data bytes
64 bytes from 3ffe:8070:1011:1::3: icmp_seq=0. time=6. ms
ICMPv6 redirect from gateway fe80::200:cff:fe09:7341
to 3ffe:8070:1011:1::3 for 3ffe:8070:1011:1::3
64 bytes from 3ffe:8070:1011:1::3: icmp_seq=1. time=1. ms
64 bytes from 3ffe:8070:1011:1::3: icmp_seq=2. time=1. ms
64 bytes from 3ffe:8070:1011:1::3: icmp_seq=3. time=1. ms
64 bytes from 3ffe:8070:1011:1::3: icmp_seq=4. time=1. ms
64 bytes from 3ffe:8070:1011:1::3: icmp_seq=5. time=1. ms
^C
----3ffe:8070:1011:1::3 PING Statistics----
6 packets transmitted, 6 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 1/1/6
bash-2.03#
```

El mensaje remarcado muestra que el router ha enviado un mensaje Redirect al host que hizo el ping. Este mensaje dice que para ir al nodo con la dirección 3ffe:8070:1011:1::3, los siguientes mensajes se deben enviar al nodo con la misma dirección. Como las direcciones de los dos nodos son iguales, el nodo destino es un vecino. Si no fuesen iguales, el mensaje indica el router a donde se deben enviar los siguientes paquetes, y el nodo es *off-link*, pertenece a otra subred. A continuación se muestran en detalle los campos de un mensaje Redirect:

```
.......
IPv6:   ----- IPv6 Header -----
IPv6:
IPv6:   Version = 6
IPv6:   Traffic Class = 224
IPv6:   Flow label = 0x0
IPv6:   Payload length = 160
IPv6:   Next Header = 58 (ICMPv6)
IPv6:   Hop Limit = 255
IPv6:   Source address = fe80::200:cff:fe09:7341
IPv6:   Destination address = 3ffe:8070:1011:1::5
IPv6:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
```

```
ICMPv6:  Type = 137 (Redirect)
ICMPv6:  Code = 0
ICMPv6:  Checksum = d3cd
ICMPv6:  Target address= 3ffe:8070:1011:1::3
ICMPv6:  Destination address= 3ffe:8070:1011:1::3
ICMPv6:
ICMPv6:  +++ ICMPv6 Target LL Addr option +++
ICMPv6:  Link Layer address: 0:60:97:1d:7a:70
```

Los datos más relevantes son los siguientes:

- IPv6: Source address = fe80::200:cff:fe09:7341
  La dirección de origen del paquete es la dirección de link-local del router, que es quien envía el mensaje Redirect

- IPv6: Destination address = 3ffe:8070:1011:1::5
  La dirección IPv6 del nodo que envió el paquete, que hizo que el router enviara un mensaje Redirect, es la dirección origen del mensaje disparador del evento.

- ICMPv6: Type = 137 (Redirect)
  El valor 137 en el campo Type de la cabecera de ICMPv6 indica que el mensaje es un Redirect.

- ICMPv6: Target address = 3ffe:8070:1011:1::3
  Indica la dirección del nodo al cual se había enviado el paquete que generó el mensaje Redirect, es la dirección destino del mensaje disparador del evento.

- ICMPv6: Destination address = 3ffe:8070:1011:1::3
  Dirección del nodo a la que se deben enviar los siguientes mensajes.

- ICMPv6: Link Layer address: 0:60:97:1d:7a:70
  Como el router conoce la dirección de link-layer del nodo al que se envía el mensaje original, se la envía al host origen.

Si los valores en los campos Target Address y Destination Address del paquete ICMPv6 son iguales, el host debe asumir que el nodo al que le está enviando el mensaje es un vecino. Si son diferentes, la Destination Address indica el router al cual se le deben enviar los próximos paquetes.

### 9. - Domain Name System v6 (DNSv6)

Si el uso del DNS nos permite acceder a los hosts por un nombre y, así, evitar tener que recordar una dirección IPv4, formada simplemente por cuatro números decimales, su utilización es sumamente aconsejable en IPv6 si tenemos en cuenta el tamaño de las direcciones.

El DNS para IPv6 se configura en el mismo archivo que para IPv4. Para diferenciar las direcciones, se creó un nuevo registro AAAA (esto es por el tamaño de la direcciones en IPv6 es cuatro veces más grande que las de IPv4). Actualmente se aconseja la utilización del registro A6 en vez de AAAA.

El archivo de DNS que se configuró es el siguiente:

```
$TTL      86400
@  1D IN SOA homero.redes.info.unlp.edu.ar.
   root.redes.info.unlp.edu.ar. (
                            51              ; serial (d. adams)
                            3H              ; refresh
                            15M             ; retry
                            1W              ; expiry
                            1D )            ; minimum

                    1D IN NS        @
homero              1D IN A         163.10.20.13

;DISPOSITIVOS
;============
router              1D IN A         163.10.20.1
paturuzu            1D IN A         163.10.20.3
upa                 1D IN A         163.10.20.2
homero              1D IN A         163.10.20.13
freebsd6            1D IN A         163.10.20.35

;DIRECCIONES IPv6
;================
freebsd6  1D IN AAAA      3ffe:8070:1011:1:200:21ff:fedb:7f1a
upa       1D IN AAAA      3ffe:8070:1011:1:a00:20ff:fe8a:28fa
paturuzu  1D IN AAAA      3ffe:8070:1011:1:2c0:f0ff:fe6a:e0f8
router6   1D IN AAAA      3ffe:8070:1011:1::1
```

El siguiente gráfico muestra que sucede cuando queremos acceder a la página www.kame.net. El servidor DNS responde con los dos tipos de direcciones, pero el host decide acceder usando el protocolo IPv6.

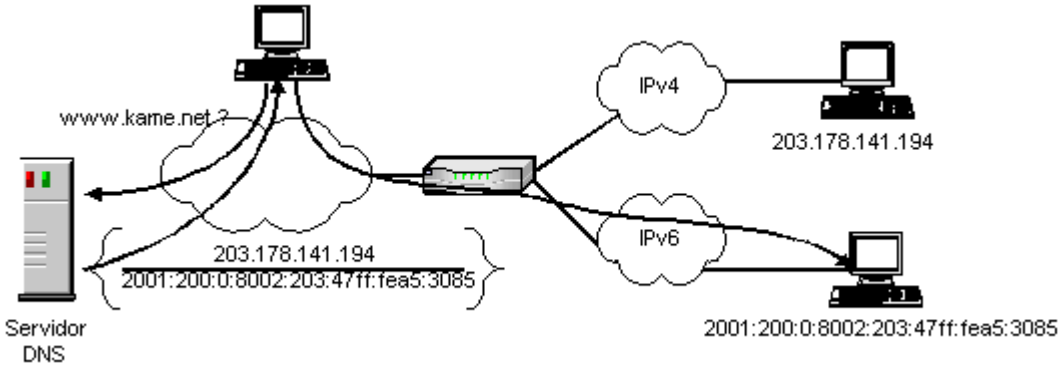


Figura 2.10 - Funcionamiento DNS

A continuación se muestran los paquetes intercambiados en el ejemplo anterior.

El primer paquete es un query al servidor de DNS, que es el host homero, que tiene el IPv4 163.10.20.13.

```
.....
Internet Protocol, Src Addr: paturuzu (163.10.20.2), Dst Addr:
homero.redes.info.unlp.edu.ar (163.10.20.13)
    Version: 4
    ......
```

```
    Protocol: UDP (0x11)
    Source: paturuzu (163.10.20.2)
    Destination: homero.redes.info.unlp.edu.ar (163.10.20.13)
User Datagram Protocol, Src Port: 32787 (32787), Dst Port: domain
(53)
    Source port: 32787 (32787)
    Destination port: domain (53)
    ...
Domain Name System (query)
    Transaction ID: 0x6513
    Flags: 0x0100 (Standard query)
    .......
Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
    Queries
        www.kame.net: type AAAA, class inet
            Name: www.kame.net
            Type: IPv6 address
            Class: inet
```

Aunque el protocolo que se utiliza para realizar la consulta de DNS es IPv4, también se solicita la dirección IPv6 de la página (el nodo que realiza la consulta es dual-stack). No es necesario que el servidor de DNS tenga habilitado IPv6.

Entre los campos más interesantes están lo siguientes:

- Destination port = 53 (DNS)
  El puerto de DNS para IPv6 es 53, igual que para IPv4.

- Type: IPv6 address
  El campo type=AAAA indica que se solicita una dirección IPv6.

Los demás campos son similares a los que se utilizan en IPv4.

El servidor de DNS contesta indicando cuales son las direcciones IPv6 de la página solicitada:

```
Internet Protocol, Src Addr: homero.redes.info.unlp.edu.ar
(163.10.20.13), Dst Addr: paturuzu (163.10.20.2)
    Version: 4
    .....
    Protocol: UDP (0x11)
    Source: homero.redes.info.unlp.edu.ar (163.10.20.13)
    Destination: paturuzu (163.10.20.2)
User Datagram Protocol, Src Port: domain (53), Dst Port: 32787
(32787)
    Source port: domain (53)
    Destination port: 32787 (32787)
    .......
Domain Name System (response)
    Transaction ID: 0x6513
    Flags: 0x8180 (Standard query response, No error)
    .....
    Answers
```

```
    .....
        Primary name: orange.kame.net
        orange.kame.net: type AAAA, class inet,
             addr 2001:200:0:8002:203:47ff:fea5:3085
    .....
```

La contestación indica que se puede acceder a la página a través de la dirección IPv6: 2001:200:0:8002:203:47ff:fea5:3085.

El siguiente mensaje también es un query al DNS, pero a diferencia de la consulta anterior, se piden las direcciones IPv4:

```
Internet Protocol, Src Addr: paturuzu (163.10.20.2), Dst Addr:
  homero.redes.info.unlp.edu.ar (163.10.20.13)
    Version: 4
    Protocol: UDP (0x11)
    Source: paturuzu (163.10.20.2)
    Destination: homero.redes.info.unlp.edu.ar (163.10.20.13)
User Datagram Protocol, Src Port: 32787 (32787), Dst Port: domain
(53)
    Source port: 32787 (32787)
    Destination port: domain (53)
    .......
Domain Name System (query)
    Transaction ID: 0x6514
    Flags: 0x0100 (Standard query)
    .....
    Queries
        www.kame.net: type A, class inet
            Name: www.kame.net
            Type: Host address
            Class: inet
```

Como vemos, el campo type es igual a A, es decir que solicita registros de tipo A (los pertenecientes a las direcciones IPv4).

El siguiente paquete es la contestación del servidor de DNS a la consulta anterior. También existen direcciones IPv4 para acceder a la página:

```
Internet Protocol, Src Addr: homero.redes.info.unlp.edu.ar
  (163.10.20.13), Dst Addr: paturuzu (163.10.20.2)
    Version: 4
    ........
    Protocol: UDP (0x11)
    Source: homero.redes.info.unlp.edu.ar (163.10.20.13)
    Destination: paturuzu (163.10.20.2)
User Datagram Protocol, Src Port: domain (53), Dst Port: 32787
(32787)
    Source port: domain (53)
    Destination port: 32787 (32787)
    .....
Domain Name System (response)
    Transaction ID: 0x6514
    Flags: 0x8180 (Standard query response, No error)
    ......
    Answers
     Primary name: orange.kame.net
     orange.kame.net: type A, class inet, addr 203.178.141.194
     ......
```

El siguiente paquete es un mensaje TCP al servidor Web de la página (orange.kame.net) para iniciar la conexión. El flag SYN en la cabecera TCP así lo indica. Como se puede observar, el sistema operativo eligió el protocolo IPv6 para realizar dicha conexión, aunque tenía ambos protocolos para realizarla.

```
Internet Protocol Version 6
    Version: 6
    .......
    Next header: TCP (0x06)
    Source address: 3ffe:8070:1011:1:260:97ff:fe59:e3d
            (3ffe:8070:1011:1:260:97ff:fe59:e3d)
    Destination address: orange.kame.net
            (2001:200:0:8002:203:47ff:fea5:3085)
Transmission Control Protocol, Src Port: 32929 (32929),
      Dst Port: http (80), Seq: 4117523274, Ack: 0, Len: 0
    Flags: 0x0002 (SYN)
    .....
```

El gráfico a continuación muestra la página www.kame.net. En la parte inferior izquierda, se puede observar que el protocolo utilizado para realizar la conexión es IPv6. La dirección que se muestra es la dirección IPv6 del nodo paturuzu, que es de donde se solicitó la página.

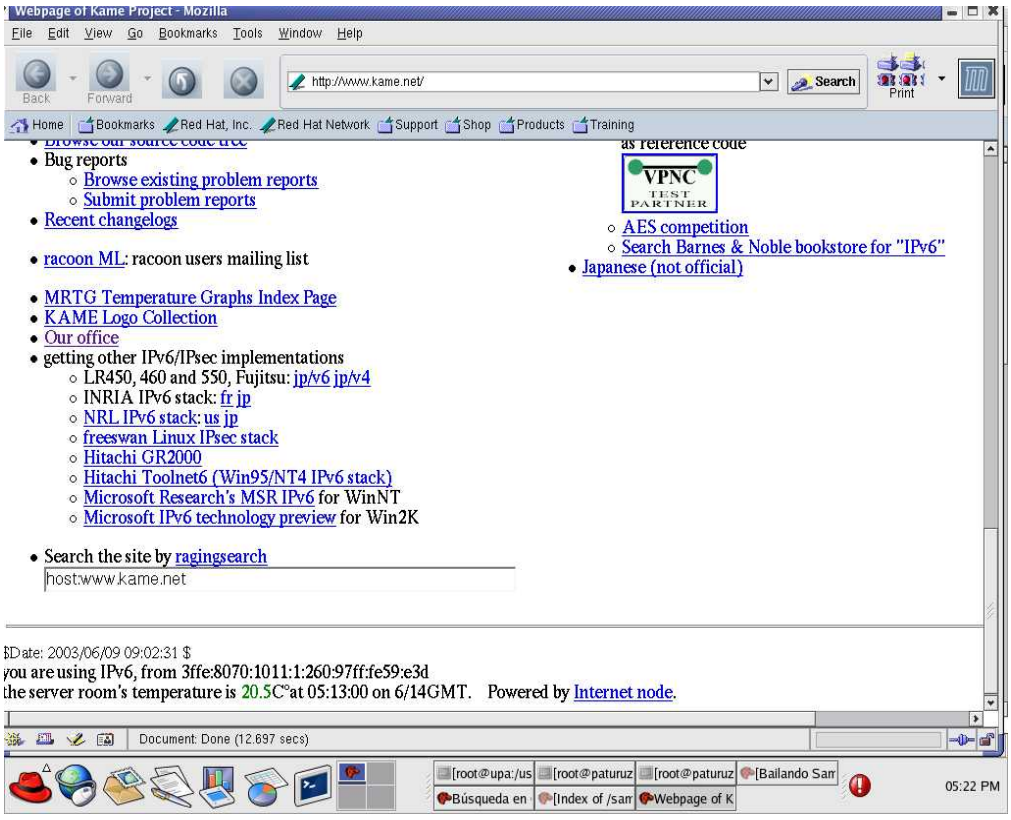


Figura 2.11 - Página www.kame.net

## 10.- Neighbor Discovery en el Router

El siguiente es un ejemplo del funcionamiento del protocolo Neighbor Discovery en un router Cisco. Para hacer esto, se debe poner el router en modo debug, lo que permite ver todos los mensajes de dicho protocolo:

```
IPv6#debug ipv6 nd
ICMP Neighbor Discovery events debugging is on
IPv6#terminal monitor
```

A continuación se muestran, y explican, los mensajes enviados y recibidos por el router, y como éste va modificando su estado:

```
ICMPv6-ND: Sending RA to FF02::1 on Ethernet0
ICMPv6-ND:     prefix = 3FFE:8070:1011:1::/64 onlink autoconfig
ICMPv6-ND: Sending RA to FF02::1 on Ethernet0
ICMPv6-ND:     prefix = 3FFE:8070:1011:1::/64 onlink autoconfig
```

El router solamente ha enviado mensajes Router Advertisement no solicitados (esto es así porque la dirección destino es la dirección multicast de todos los nodos).

```
ICMPv6-ND: Received RS on Ethernet0 from FE80::800:208A:28FA
```

En este instante, se reinicia un host quien envía un Router Solicitation para ver si existe un router en el link.

```
ICMPv6-ND: Sending RA to FE80::800:208A:28FA on Ethernet0
ICMPv6-ND:     prefix = 3FFE:8070:1011:1::/64 onlink autoconfig
```

Como el Router Advertisement es en repuesta a una solicitud de un host, la dirección destino del anuncio es igual a la dirección origen de la solicitud.

```
ICMPv6-ND: Received RS on Ethernet0 from FE80::A00:20FF:FE8A:28FA
ICMPv6-ND: INCMP created: FE80::A00:20FF:FE8A:28FA
ICMPv6-ND: INCMP -> STALE: FE80::A00:20FF:FE8A:28FA
```

El router recibe un Router Solicitation. Al no contener la dirección de link-layer del emisor de la solicitud en la Neighbor Cache, crea una entrada con el estado INCOMPLETE. Como no es creada por la recepción de un Neighbor Advertisement, el estado de la entrada cambia directamente a STALE.

```
ICMPv6-ND: Received NS for FE80::200:CFF:FE09:7341 on Ethernet0 from
              FE80::A00:20FF:FE8E:F427
ICMPv6-ND: Sending NA for FE80::A00:20FF:FE8E:F427 on Ethernet0
```

El host FE80::A00:20FF:FE8E:F427 le envía un Neighbor Solicitation al router, quien le contesta con un Neighbor Advertisement seteando la dirección destino del anuncio igual a la dirección origen de la solicitud.

```
ICMPv6-ND: STALE -> DELAY: FE80::A00:20FF:FE8E:F427
```

Más de ReachableTime milisegundos han pasado desde que la última confirmación positiva fue recibida indicando que el *forward-path* estaba

funcionando correctamente. Mientras está en el estado STALE, el router no realiza ninguna acción hasta que un paquete es enviado. Cuando se envía un paquete a ese host se cambia el estado a DELAY.

```
ICMPv6-ND: DELAY -> PROBE: FE80::A00:20FF:FE8E:F427
ICMPv6-ND: Sending NS for FE80::A00:20FF:FE8E:F427 on Ethernet0
```

Un paquete ha sido enviado dentro de los últimos *DELAY_FIRST_PROBE_TIME* segundos. Como no se recibió ninguna confirmación dentro de ese tiempo, el router envía un Neigbhor Solicitation y cambia el estado a PROBE,

```
ICMPv6-ND: Received NA for FE80::200:CFF:FE09:7341 on Ethernet0 from
              FE80::A00:20FF:FE8E:F427
ICMPv6-ND: PROBE -> REACH: FE80::A00:20FF:FE8E:F427
```

Como el destino contestó con un Neighbor Advertisement, el router tiene confirmación de accesibilidad a ese destino. El estado cambia a REACHABLE.

```
ICMPv6-ND: Sending NS for FE80::800:208A:28FA on Ethernet0
ICMPv6-ND: DELAY -> PROBE: FE80::A00:20FF:FE8A:28FA
ICMPv6-ND: Sending NS for FE80::A00:20FF:FE8A:28FA on Ethernet0
ICMPv6-ND: Sending NS for FE80::A00:20FF:FE8A:28FA on Ethernet0
ICMPv6-ND: PROBE deleted: FE80::A00:20FF:FE8A:28FA
```

El estado para el host FE80::800:208A:28FA es DELAY. Cuando el router le envía un Neighbor Solicitation, el estado, en la entrada, automáticamente, cambia a PROBE. Como el destino no contesta, el router le envía dos solicitudes más. Al no contestar, el router asume que el vecino ya no está más accesible y la entrada es eliminada.

## 11. - Tabla Ruteo (Solaris)

A continuación se muestran las tablas de ruteo de un nodo *dual-stack*, mediante el comando *netstat*. Este se ejecuta en el host Solaris:

```
bash-2.03# netstat -rn

Routing Table: IPv4
  Destination           Gateway           Flags  Ref   Use   Interface
-------------------- -------------------- ----- ----- ------ ---------
163.10.20.0          163.10.20.13          U        1    353  hme0
224.0.0.0            163.10.20.13          U        1      0  hme0
default              163.10.20.1           UG       1   3121
127.0.0.1            127.0.0.1             UH      29   2268  lo0

Routing Table: IPv6
Destination/Mask    Gateway                       Flags Ref Use If
------------------  ----------------------------- ----- --- --- ----
3ffe:8070:1011:1::/64 3ffe:8070:1011:1:a00:20ff:fe8e:f427  U     1   8 hme0:1
fe80::/10           fe80::a00:20ff:fe8e:f427       U     1   2   hme0
ff00::/8            fe80::a00:20ff:fe8e:f427       U     1   0   hme0
default             fe80::200:cff:fe09:7341        UG    1   13  hme0
::1                 ::1                            UH    1   0   lo0
```

Como podemos observar, las tablas de ruteo en IPv4 e IPv6 son similares, la mayor diferencia se encuentra en el formato de las direcciones.

## 12. - Fragmentación de paquetes

Como se explicó en el Capítulo 1, la información referente a la fragmentación de paquetes se ubica en una cabecera de extensión y no en la cabecera de IPv6. El siguiente es un ejemplo de una fragmentación de paquetes.

Para lograr esto enviamos un ping con un tamaño de paquete de 1800 bytes cuando el MTU del link es de 1500 bytes.

```
.....
IPv6:   ----- IPv6 Header -----
IPv6:
IPv6:   Version = 6
IPv6:   Traffic Class = 0
IPv6:   Flow label = 0x0
IPv6:   Payload length = 1456
IPv6:   Next Header = 44 (IPv6-Frag)
IPv6:   Hop Limit = 64
IPv6:   Source address = fe80::200:cff:fe09:7341
IPv6:   Destination address = fe80::a00:20ff:fe8e:f427
IPv6:
IPv6-Frag:  ----- IPv6 Fragment Header -----
IPv6-Frag:
IPv6-Frag:  Next Header = 58 (ICMPv6)
IPv6-Frag:  Fragment Offset = 0
IPv6-Frag:  More Fragments Flag = true
IPv6-Frag:  Identification = 1
IPv6-Frag:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
ICMPv6:  Type = 58 (Unknown)
ICMPv6:  Code = 0
ICMPv6:  Checksum = 1
```

Los datos más relevantes son los siguientes:

- IPv6: Next Header = 44 (IPv6-Frag)
  El campo Next Header de la cabecera IPv6 indica que la siguiente es una cabecera de extensión, la de fragmentación más precisamente.

- IPv6-Frag: Fragment Offset = 0
  Indica la posición del fragmento dentro del paquete original. Como es 0, éste es el primer fragmento del paquete.

- More Fragments Flag = true
  Indica que hay más fragmentos.

- IPv6-Frag: Identification = 1
  Sirve para identificar todos los fragmentos de un mismo paquete.

Éste es el segundo fragmento del paquete:

```
........
IPv6:   ----- IPv6 Header -----
IPv6:
IPv6:   Version = 6
IPv6:   Traffic Class = 0
IPv6:   Flow label = 0x0
IPv6:   Payload length = 320
IPv6:   Next Header = 44 (IPv6-Frag)
IPv6:   Hop Limit = 64
IPv6:   Source address = fe80::200:cff:fe09:7341
IPv6:   Destination address = fe80::a00:20ff:fe8e:f427
IPv6:
IPv6-Frag:  ----- IPv6 Fragment Header -----
IPv6-Frag:
IPv6-Frag:  Next Header = 58 (ICMPv6)
IPv6-Frag:  Fragment Offset = 1448
IPv6-Frag:  More Fragments Flag = false
IPv6-Frag:  Identification = 1
IPv6-Frag:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
ICMPv6:  Type = 58 (Unknown)
ICMPv6:  Code = 0
ICMPv6:  Checksum = 5a8
```

Entre los parámetros más importantes se encuentran:

- IPv6: Next Header = 44 (IPv6-Frag)
  Indica que la siguiente cabecera de extensión es un Fragment Header.

- IPv6-Frag: Fragment Offset = 1448
  Indica la posición del fragmento en el paquete original.

- IPv6-Frag: More Fragments Flag = false
  Indica que no existen más fragmentos. Éste es el último fragmento del paquete, pero no es el último fragmento que puede recibir el nodo para un paquete determinado.

- IPv6-Frag: Identification = 1
  El campo Identification es igual al del primer fragmento.

Si se observa en la cabecera IPv6 del primer fragmento, el valor del campo Payload Length es 1456 bytes, sin embargo, en la cabecera Fragment Header del segundo paquete el Fragment Offset está seteado a 1448, es decir que los datos del segundo paquete se agregarían 8 byte antes del final de los datos del primer paquete. Estos valores son correctos. Se debe tener en cuenta que el campo Payload Length indica la suma de la cabecera ICMPv6 (que es 8 bytes) más la cantidad de datos del fragmento.

## 13. - Routing Header

La finalidad de esta cabecera es, indicarle a un mensaje, la lista de routers intermedios por los que éste debe pasar hasta llegar a su destino final. Es semejante a las opción *Loose Source* de IPv4.

El siguiente es un ejemplo para mostrar como se forma un cabecera de extensión *Routing Header*. Para lograrlo enviamos un ping a la dirección 3ffe:38e1::2:2 con la opción -g. Esta opción obliga a que el mensaje pase por estos nodos antes de llegar al destino.

```
bash-2.03#  ping  -g  3FFE:8070:1011:1::1  -g  3FFE:8070:1011::2
3FFE:38E1::2:2

.....
IPv6:   ----- IPv6 Header -----
IPv6:
IPv6:   Version = 6
IPv6:   Traffic Class = 0
IPv6:   Flow label = 0x0
IPv6:   Payload length = 104
IPv6:   Next Header = 43 (IPv6-Route)
IPv6:   Hop Limit = 60
IPv6:   Source address = 3ffe:8070:1011:1:a00:20ff:fe8e:f427
IPv6:   Destination address = 3ffe:8070:1011:1::1
IPv6:
IPv6-Route:  ----- IPv6 Routing Header -----
IPv6-Route:
IPv6-Route:  Next header = 58 (ICMPv6)
IPv6-Route:  Header length = 40
IPv6-Route:  Routing type = 0
IPv6-Route:  Segments left = 2
IPv6-Route:  address[0]=3ffe:8070:1011::2
IPv6-Route:  address[1]=3ffe:38e1::2:2
IPv6-Route:
ICMPv6:  ----- ICMPv6 Header -----
ICMPv6:
ICMPv6:  Type = 58 (Unknown)
ICMPv6:  Code = 4
ICMPv6:  Checksum = 2
```

Los campos más importantes son:

- IPv6: Next Header = 43 (IPv6-Route)
  Indica que la siguiente es la cabecera de extensión Routing Header.

- IPv6: Destination address = 3ffe:8070:1011:1::1
  La dirección destino del paquete es la dirección IPv6 del primer nodo del path indicado en el ping.

- IPv6-Route: Routing type = 0
  Fuerza el ruteo del paquete a través de una lista de routers intermedios.

- IPv6-Route: Segments left = 2
  Este campo es utilizado por los nodos destinos (el que se indica en la dirección destino de la cabecera IPv6) para saber cual es la dirección del siguiente nodo por el que debe pasar el paquete.

- IPv6-Route: address[0]=3ffe:8070:1011::2
  IPv6-Route: address[1]=3ffe:38e1::2:2
  Lista de direcciones de routers por donde debe pasar el paquete. La última dirección (3ffe:38e1::2:2) indica el destino final del paquete. Si el mensaje incluye la cabecera de extensión Destination Header, a continuación de la cabecera Hop by Hop Header, debe ser procesada por todos los routers por donde pasa el paquete, y cuya dirección esté incluida el cabecera Routing. Esto se debe, a que la dirección destino de la cabecera IPv6 se va tomando, sucesivamente, de esta lista hasta llegar a la última dirección que es el destino final del paquete.

## 14. - Con dos routers

Para el siguiente paso en la prueba del protocolo, se configuró e instaló un nuevo router. Después de analizar las distintas implementaciones se decidió instalar la versión 4.8 de FreeBSD. Además, de ser una de las más avanzadas, tiene soporte para el ruteo multicast, fundamental en la última parte del trabajo. El siguiente gráfico muestra como quedó la red con la nueva configuración:

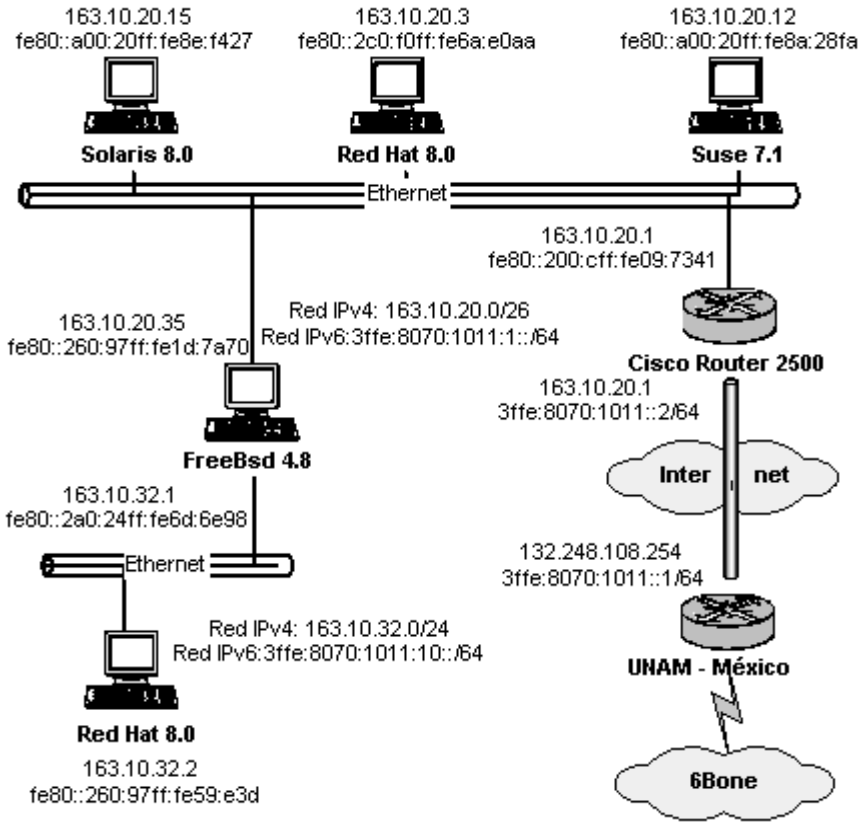


Figura 2.12 - Diseño red IPv6 LINTI (con dos routers)

El archivo para configurar el PC Router con FreeBSD es el rc.conf, que se encuentra en el directorio /etc. A continuación se muestra como queda:

```
#Se habilita IPv6 en el router
ipv6_enable="YES"

#Se indican las interfaces en las que se quiere habilitar IPv6. Si se pone la
#palabra auto se habilita en todas las interfaces
ipv6_network_interface="xl0 ep0"

#Se configuran las direcciones en las dos interfaces.
ipv6_ifconfig_xl0="3ffe:8070:1011:1::10 prefixlen 64"
ipv6_ifconfig_ep0="3ffe:8070:1011:10::10 prefixlen 64"
ipv6_defaultrouter="fe80::1%xl0"

#Se habilita el host para que funcione como un gateway
ipv6_gateway_enable="YES"

#Se habilita el ruteo para IPv6. Se ejecuta el protocolo de ruteo RIPng
ipv6_router_enable="YES"
ipv6_router="/usr/sbin/route6d"

#Se habilita el router para que envíe anuncios de router en todas las
#interfaces con soporte IPv6.
rtadvd_enable="YES"

#Si se quiere que el router envíe anuncios en determinadas interfaces, se
#agrega la siguiente línea:
rtadvd_interfaces="ep0"
```

Con el comando ifconfig se puede ver como quedan configuradas las interfaces en el router:

```
freebsd# ifconfig
xl0: flags=8a43<UP,BROADCAST,RUNNING,ALLMULTI,SIMPLEX,MULTICAST> mtu
1500
        inet 163.10.20.35 netmask 0xfffffc0 broadcast 163.10.20.63
        inet6 fe80::260:97ff:fe1d:7a70%xl0 prefixlen 64 scopeid 0x1
        inet6 3ffe:8070:1011:1::10 prefixlen 64
        ether 00:60:97:1d:7a:70
        media: Ethernet 10baseT/UTP (10baseT/UTP <half-duplex>)
lp0: flags=8851<UP,POINTOPOINT,RUNNING,SIMPLEX,MULTICAST> mtu 1500
ep0: flags=8a43<UP,BROADCAST,RUNNING,ALLMULTI,SIMPLEX,MULTICAST> mtu
1500
        inet 163.10.32.1 netmask 0xffffff00 broadcast 163.10.32.255
        inet6 fe80::2a0:24ff:fe6d:6e98%ep0 prefixlen 64 scopeid 0x3
        inet6 3ffe:8070:1011:10::1 prefixlen 64
        ether 00:a0:24:6d:6e:98
        media: Ethernet 10baseT/UTP
....
```

A partir de este momento, la PC está funcionando como un router. Los anuncios enviados por el router se configuran (por ejemplo, el tiempo de vida del router o el tiempo de vida válido de una dirección) con valores por defecto.

Utilizando el archivo rtadvd.conf (si no existe se lo debe crear), en el directorio /etc, se pueden modificar estos datos:

```
freebsd# more rtadvd.conf
default:\
    :raflags#0:rltime#300:rtime#200:retrans#100:\
    :pinfoflags#192:vltime#1300:pltime#1200:\

xl0:\
   :addrs#1:addr="3ffe:8070:1011:15::":prefixlen#64:chlim#128:\
   :mtu#1300:maxinterval#30:tc=default:

ep0:\
   :addrs#1:addr="3ffe:8070:1011:10::":prefixlen#64:chlim#64:\
   :mtu#1400:maxinterval#30:tc=default:
```

Se pueden configurar todos los parámetros que contiene un mensaje de anuncio de router. Si no se especifica algún valor, se utiliza el valor por defecto. La entrada *default* no es necesaria. Se puede utilizar para configurar valores comunes a ambas interfaces. Tener en cuenta que si un valor se declara en la interface y en la entrada *default*, el valor válido es el que se setea en esta última. Si no se configura este archivo, el router publica los prefijos que deduce de las direcciones IPv6 asignadas a las interfaces.

A continuación se muestra como queda autoconfigurado el host con Red Hat 8.0 que se encuentra entre los dos routers y recibe anuncios de ambos:

```
[root@upa root]# ip -6 addr show
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    inet6 ::1/128 scope host
2: eth0:<BROADCAST,MULTICAST,PROMISC,UP>mtu 1500 qdisc pfifo_fast
qlen 100
   inet6 fe80::2c0:f0ff:fe6a:e0aa/10 scope link
   inet6 3ffe:8070:1011:15:2c0:f0ff:fe6a:e0aa/64 scope global dynamic
      valid_lft 1300sec preferred_lft 1200sec
   inet6 3ffe:8070:1011:1:2c0:f0ff:fe6a:e0aa/64 scope global dynamic
      valid_lft 6825sec preferred_lft 5825sec
```

Para la autoconfiguración de la primera de las dos direcciones globales el host utiliza el anuncio del router FreeBSD y para la segunda el del Cisco.

Con el siguiente comando, ejecutado en Red Hat 8.0, se puede observar que en el link existen dos routers. Los dos anuncios se recibieron por la misma interface, y se encuentran en estado *stale*

```
[root@upa root]# ip -6 neigh show
fe80::260:97ff:fe1d:7a70 dev eth0 lladdr 00:60:97:1d:7a:70 router nud stale
fe80::200:cff:fe09:7341 dev eth0 lladdr 00:00:0c:09:73:41 router nud stale
```

La siguiente prueba fue crear un túnel estático entre los dos routers. En el FreeBSD, se crean especificando las siguientes líneas en el archivo rc.conf:

#Nombre de la interface lógica que se está creando
```
gif_interfaces="gif0"
```

#Se indica los puntos de origen y destino del túnel
```
gifconfig_gif0="163.10.20.35 163.10.20.1"
```

#Se indica la dirección IPv6 de la interface
```
ipv6_ifconfig_gif0="3ffe:8070:1011:2::1 prefixlen 64"
```

En el router Cisco la configuración se hace como se indicó más arriba, y queda de la siguiente manera:

```
IPv6#show tunnel2
interface Tunnel2
description Tunel interno con FreeBSD 4.8
no ip address
ipv6 address 3FFE:8070:1011:2::2/64
tunnel source 163.10.20.1
tunnel destination 163.10.32.35
tunnel mode ipv6ip
```

Para probar el túnel se ejecuta un ping en el FreeBSD:

```
freebsd#ping6 -I gif0 3ffe:8070:1011:2::2
```

El paquete del echo request es el siguiente:

```
.....
Ethernet II, Src: 00:60:97:1d:7a:70, Dst: 00:00:0c:09:73:41
    Destination: 00:00:0c:09:73:41 (00:00:0c:09:73:41)
    Source: 00:60:97:1d:7a:70 (00:60:97:1d:7a:70)
    Type: IP (0x0800)
Internet Protocol, Src Addr:163.10.20.35, Dst Addr:163.10.20.1
    Version: 4
    Header length: 20 bytes
    ........
Protocol: IPv6 (0x29)
    Header checksum: 0x2ddd (correct)
    Source: 163.10.20.35 (163.10.20.35)
    Destination: 163.10.20.1 (163.10.20.1)
Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 16
    Next header: ICMPv6 (0x3a)
    Hop limit: 64
    Source address: 3ffe:8070:1011:2::1
    Destination address: 3ffe:8070:1011:2::2
Internet Control Message Protocol v6
    Type: 128 (Echo request)
    Code: 0
    Checksum: 0x8d82 (correct)
    ID: 0x0115
    Sequence: 0x0000
    Data (8 bytes)
```

A continuación se explican algunos de los parámetros del mensaje:

- Version: 4
  El protocolo exterior es IPv4, a pesar que el ping se hizo a una dirección IPv6.

- Protocol: IPv6 (0x29)
  Indica que la cabecera que está a continuación pertenece al protocolo IPv6.

- Source: 163.10.20.35 (163.10.20.35)
  Ésta es la dirección origen del túnel por donde se envía el Echo Request. Indica cual es el nodo que encapsuló el paquete IPv6 en un paquete IPv4.

- Destination: 163.10.20.1 (163.10.20.1)
  Ésta es la dirección final del túnel e indica quien es el encargado de desencapsular el paquete IPv6.

- Source address: 3ffe:8070:1011:2::1 (3ffe:8070:1011:2::1)
  Indica la dirección origen del Echo Request.

- Destination address: 3ffe:8070:1011:2::2 (3ffe:8070:1011:2::2)
  Indica la dirección destino del mensaje.

Al tener los dos router configurados, se probó como responderían los nodos si los routers publicaban la misma información pero con distintos valores. La RFC 2461, que define el Neigbhor Discovery, es clara al respecto: el valor del último anuncio recibido es el válido.

Se modificó el valor del MTU, en ambos routers, a valores diferentes. A 1350 bytes en el FreeBSD y a 1400 bytes en el Cisco 2500.

En los Red Hat 8.0 se presentó el siguiente problema:
al ejecutar el comando ifconfig (o ip -6 addr show), el valor del MTU en la interface eth0 (que es la que recibe los anuncios del router) permanece fijo en 1500 bytes. Pero, si se miran las variables del kernel, con el comando sysctl, se observa que el valor se ha modificado de acuerdo a los anuncios recibidos.

Además, por ejemplo, si se ejecuta un ping -s 1405 *dir. IPv6*, el paquete es fragmentado. Esto mismo sucede en el FreeBsd si está funcionando como host.

Sin dudas, lo que está funcionando incorrectamente son los comando ifconfig (e ip -6 en RedHat) en ambos sistemas.

### 15. - Configuración túnel 6to4

El siguiente es otro método de transición de IPv4 a IPv6, el cual está descripto en el punto 9.2.3 del Capítulo 1. El gráfico 2.13 muestra como se armó la red para configurar un túnel de este tipo.

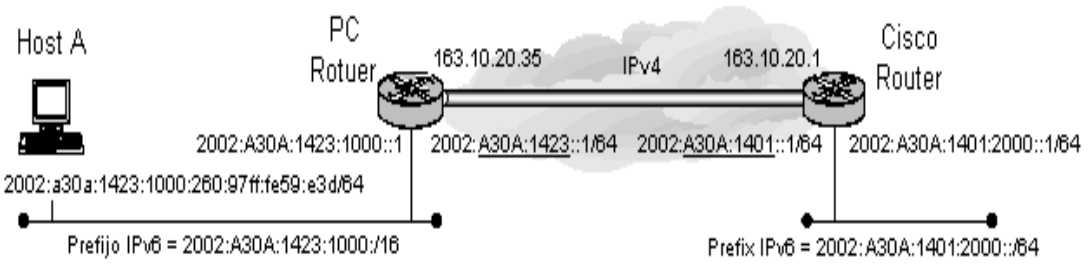


Figura 2.13 - Túnel 6to4 LINTI

Para poder realizar la prueba, se realizaron las siguientes modificaciones en los archivos correspondientes.

En el PC Router se agregaron las siguientes líneas al archivo rc.conf:

#Se configura la siguiente dirección en la interface ep0.
```
ipv6_ifconfig_ep0="2002:a30a:1423:1000::1 prefixlen 64"
```

#Indicamos el IPv4 de la interface donde comienza el túnel. Con este IP, el #nodo arma el prefijo /48 (2002 más la dirección IPv4 en hexadecimal)
```
stf_interface_ipv4addr="163.10.20.35"
```

#Indicamos que la ruta default para esa dirección es el túnel recién creado.
```
ipv6_defaultrouter="2002:a30a:1423::"
```

El siguiente archivo que se modificó es el rtadvd.conf para anunciar en la interface ep0 el prefijo 2002:a30a:1423:1000::/64. La interface xl0 no fue modificada.

```
ep0:\
   :addrs#1:addr="2002:a30a:1423:1000::":prefixlen#64:chlim#64:\
   :maxinterval#300:vltime#1200:pltime#1000:rtime#1400:retrans#1300:\
   :mtu#1400:pinfoflags=192:raflags#0:rltime#1500:
```

La configuración en el router Cisco también fue modificada. El túnel quedó configurado de la siguiente manera.

```
interface Tunnel64
 description Tunel 6to4 de prueba
 no ip address
 no ip redirects
 ipv6 address 2002:A30A:1401::1/128
 tunnel source Ethernet0
 tunnel mode ipv6ip 6to4
```

La interface origen del túnel debe estar configurada con una dirección IPv4. Con el comando *tunnel mode ipv6ip 6to4* se le indica al router que el túnel es del tipo 6to4.

Además, se debe agregar una ruta estática para el prefijo 2002::/16 al túnel especificado. Todos los paquetes que comienzan con el prefijo 2002 se deben enviar a través del túnel Tunnel64. El siguiente comando hace esto

```
ipv6 route 2002::/16 Tunnel64
```

Para probar el túnel, en el Host A se realiza un ping al router Cisco. Para llegar a éste, el mensaje debe pasar por el PC Router. La red que existe entre los dos routers soporta IPv4 únicamente, por lo cual, se estableció un túnel entre ambos. El siguiente paquete es un Echo Request enviado a la dirección IPv6: 2002:a30a:1401:2000::1. Cuando el paquete pasa por el PC Router, éste lo reenvía a su destino por el túnel stf0. A pesar de no estar configurada manualmente, el router obtiene la dirección IPv4 final del túnel de la dirección IPv6 destino del paquete. Como vemos, el router agrega una cabecera IPv4, donde la dirección origen es la dirección IPv4 configurada en el archivo rc.conf, y la dirección destino corresponde a los 32 bits a continuación del prefijo 2002::/16.

```
Frame 9 (138 bytes on wire, 138 bytes captured)
    Arrival Time: Aug 20, 2003 13:50:03.005128000
    Time delta from previous packet: 3.007420000 seconds
    Time relative to first packet: 14.153024000 seconds
    Frame Number: 9
    Packet Length: 138 bytes
    Capture Length: 138 bytes
Ethernet II, Src: 00:60:97:1d:7a:70, Dst: 00:00:0c:09:73:41
    Destination: 00:00:0c:09:73:41 (00:00:0c:09:73:41)
    Source: 00:60:97:1d:7a:70 (00:60:97:1d:7a:70)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 163.10.20.35, Dst Addr: 163.10.20.1
    Version: 4
    ....
    Protocol: IPv6 (0x29)
    Source: 163.10.20.35 (163.10.20.35)
    Destination: 163.10.20.1 (163.10.20.1)
Internet Protocol Version 6
    Version: 6
    Next header: ICMPv6 (0x3a)
    Source address: 2002:a30a:1423:1000:260:97ff:fe59:e3d
    Destination address: 2002:a30a:1401:2000::1
Internet Control Message Protocol v6
    Type: 128 (Echo request)
    ......
```

El router Cisco recibe el Echo Request enviado a su dirección IPv6 y lo contesta con un Echo Replay. En este caso, este router es quien encapsula el paquete, con lo cual se modifican las cabeceras de los paquetes.

```
Frame 10 (138 bytes on wire, 138 bytes captured)
    ........
Ethernet II, Src: 00:00:0c:09:73:41, Dst: 00:60:97:1d:7a:70
    Destination: 00:60:97:1d:7a:70 (00:60:97:1d:7a:70)
    Source: 00:00:0c:09:73:41 (00:00:0c:09:73:41)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 163.10.20.1, Dst Addr: 163.10.20.35
    Version: 4
    .......
    Protocol: IPv6 (0x29)
    Source: 163.10.20.1 (163.10.20.1)
    Destination: 163.10.20.35 (163.10.20.35)
Internet Protocol Version 6
    Version: 6
    Next header: ICMPv6 (0x3a)
    Source address: 2002:a30a:1401:2000::1
```

```
    Destination address: 2002:a30a:1423:1000:260:97ff:fe59:e3d
Internet Control Message Protocol v6
    Type: 129 (Echo reply)
```

El ejemplo anterior muestra el funcionamiento de un túnel 6to4. A pesar que en ningún momento se indicó cual es la dirección IPv4 final del túnel, el nodo encapsulador la obtiene de la dirección destino del paquete IPv6 que está siendo tunelizado.

### 16. - Deteniendo el router

Por último, se examinó que sucede cuando un nodo deja de funcionar como un router. Con los dos routers en funcionamiento, sé deshabilitó el protocolo IPv6 en el router Cisco.

```
IPv6(config)#no ipv6 unicast-routing
IPv6(config)#exit
```

Al observar como queda configurado IPv6 en la interface obtenemos lo siguiente:

```
IPv6#sho ipv6 int eth0
Ethernet0 is up, line protocol is up
 ....
 ND reachable time is 3000 milliseconds
 Default router is FE80::260:97FF:FE1D:7A70 on Ethernet0
```

La última línea del comando muestra que el router ya no funciona más como tal, y que tiene un nuevo router vecino como *default router*. La dirección que se muestra es la dirección de link-local del router FreeBsd.

Cuando un router deja de funcionar, debe indicárselo a los demás nodos del link. Enviando un mensaje Router Advertisement, con el tiempo de vida del router igual a 0, se logra esto. El siguiente mensaje es un anuncio de router de este tipo:

```
Frame 1477 (70 bytes on wire, 70 bytes captured)
    ...
    Source address: fe80::200:cff:fe09:
    Destination address: ff02::1
Internet Control Message Protocol v6
    Type: 134 (Router advertisement)
    Code: 0
    Checksum: 0xa85d (correct)
    Cur hop limit: 64
    Flags: 0x00
        0... .... = Not managed
        .0.. .... = Not other
        ..0. .... = Not Home Agent
        ...0 0... = Router preference: Medium
    Router lifetime: 0
    Reachable time: 2000
    Retrans time: 3000
```

El anuncio se envía a la dirección multicast de todos los nodos con el parámetro Router Lifetime = 0, y luego el router cesa de enviar sus anuncios.

Un anuncio de este tipo también es utilizado cuando el router no quiere que los hosts lo vean como un router default, pero si quiere publicar algunos parámetros de autoconfiguración para ellos.

En un router Cisco se envía un solo paquete para señalar este evento, en cambio, si esto se hace en un PC Router funcionando con FreeBsd, se envían tres mensajes. Este funcionamiento es mejor (y mas adecuado a lo que especifica la RFC 2461) porque trata de solucionar el problema de la pérdida de mensajes en la red.

# *Capítulo 3*
# *Multicast*

## 1. - Introducción

Multicasting es una tecnología que tendrá un rol clave en la próxima generación de Internet. Esto será así, porque las aplicaciones que están siendo desarrolladas, como videoconferencia, aprendizaje a distancia, distribución de software, etc., inherentemente, harán uso de ella.

En un escenario de unicast, cuando un emisor quiere enviar el mismo paquete a varios destinos diferentes, debe enviar tantos paquetes como receptores haya. El siguiente gráfico muestra un ejemplo de esto:

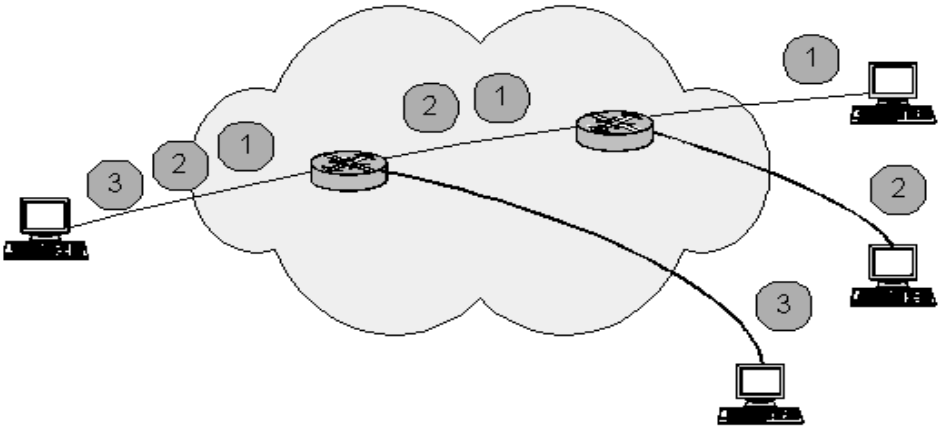


Figura 3.1 - Envío mensajes unicast

En cambio, en las transmisiones multicast, el emisor envía una sola copia del paquete y la red lo replica, en el último salto posible, para cada receptor. En cada subred existe una sola copia de un paquete a la vez.

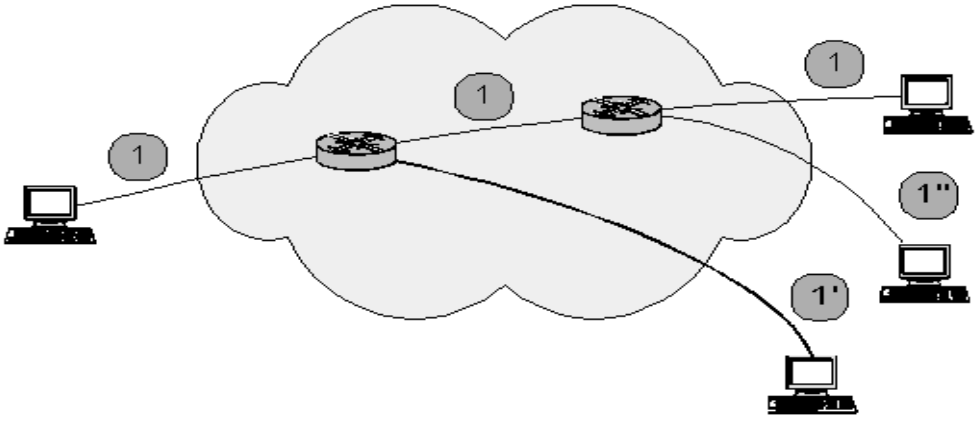


Figura 3.2 - Envío mensajes multicast

Esta última forma de enviar los paquetes permite:

- Que el ancho de banda de la red sea utilizado más eficientemente, dado que múltiples streams de datos son reemplazados por un único stream.
- Que los routers mejoren su performance, al tener que procesar y reenviar, menor cantidad de paquetes
- Que se desarrollen aplicaciones distribuidas que no serían posibles en ambientes unicast

El envío de los paquetes se realiza a un conjunto de hosts que previamente se han unido a un grupo multicast determinado, es decir, la dirección destino de los paquetes es una dirección multicast.

Los hosts pueden unirse y dejar un grupo multicast en cualquier momento, y no hay límite en la cantidad de grupos que el nodo puede ser miembro. A su vez, los grupos pueden tener una cantidad ilimitada de integrantes.

No es necesario que los emisores deban unirse al grupo al cual le están enviando tráfico multicast.

Esta tecnología se basa en UDP, con lo cual, las aplicaciones tendrán que esperar ciertos problemas, y que en lo posible, deberán solucionar:

1. Recepción de paquetes duplicados.
2. Paquetes fuera de secuencia.
3. Envío de mejor esfuerzo (pérdida de paquetes)
4. Problemas de congestión

Si los hosts pertenecientes a un grupo multicast se encuentran en redes diferentes, es necesario que los routers que se encuentran entre ellas, ejecuten un protocolo de ruteo multicast, tipo PIM6-SM, PIM6-DM, etc. Estos protocolos se encargan de armar árboles de distribución desde el origen hacia todos los receptores.

## 2. - Multicast Listener Discovery para IPv6 (MLDv6)

El propósito del MLD es habilitar, a cada router, para descubrir la presencia de receptores multicast (esto es, nodos deseando recibir tráfico multicast) en los enlaces directamente ligados a él, y cuales son, específicamente, las direcciones multicast de interés para esos nodos vecinos. Esta información es entregada al protocolo de ruteo multicast que esté siendo utilizado en el router.

MLD es un protocolo asimétrico porque su comportamiento en los routers es diferente al que tiene en los nodos receptores.

Si un router tiene más de una interface al mismo link necesita ejecutar, la parte del router del MLD, solamente, en una de sus interfaces. Los

receptores, por otra parte, deben ejecutar, la parte del MLD específica para los hosts, en todas las interfaces para los cuales una aplicación o protocolo de capa superior ha solicitado recepción de paquetes multicast.

El siguiente gráfico muestra como se compone un mensaje IPv6 que transporta datos de MLDv6:

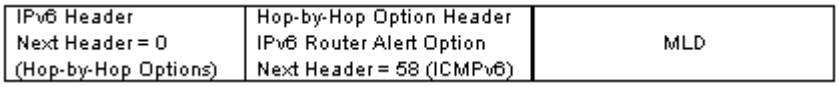


Figura 3.3 - Formato mensaje MLDv6

Los nodos insertan la opción IPv6 Router Alert para indicarle a los routers que deben procesar el mensaje MLD enviado a una dirección multicast, y a la cual el router no está escuchando. De esta manera, se evita que los routers tengan que analizar todos los paquetes que pasan por él para ver si contiene información que les es relevante.

## 2.1. - Formato de los mensajes

MLD es un subprotocolo de ICMPv6, esto es, los tipos de mensajes son un subconjunto de éste. Todos los mensajes son enviados con la dirección de link-local como dirección origen de IPv6, el campo Hop-Limit igual a 1, y una opción de Router Alert en una cabecera Hop-by-Hop. Los mensajes MLD tienen la siguiente estructura:

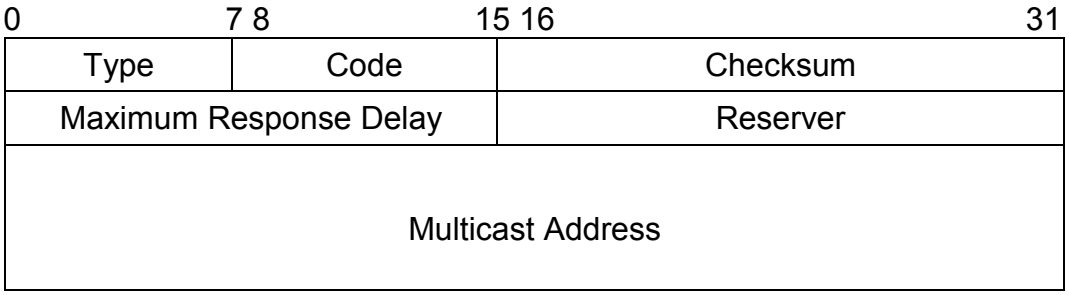


Figura 3.4 - Estructura paquete MLDv6

a) Type

Hay tres tipos de mensajes MLD:

- Multicast Listener Query (Type = 130)

  Hay dos subtipos de mensajes Multicast Listener Query:

  General Query: usado para saber cuales direcciones multicast tienen receptores en un link.

  Multicast Address Specific Query: usado para saber si una determinada dirección tiene receptores en un link.

- Multicast Listener Report (Type = 131)
- Multicast Listener Done (Type = 132)

Este campo permite distinguir los mensajes MLD de los demás protocolos que hacen uso de ICMPv6, porque en el campo Next Header de la cabecera anterior, el valor que se setea es 58, que indica que a continuación hay un paquete ICMPv6.

b) Code

Seteado a 0 por el emisor, ignorado por el receptor.

c) Checksum

El estándar checksum de ICMPv6.

d) Maximum Response Delay

Este campo tiene significado solamente en los mensajes Query y especifica el máximo retardo (en milisegundos) permitido a los hosts antes de enviar un mensaje de respuesta Report. En todos los demás mensajes es seteado a 0 por los emisores e ignorado por los receptores. Este tiempo permite que los routers ajusten la *"leave latency"* (el tiempo entre el momento que el último nodo en un link cesa de escuchar a una dirección de multicast particular y el momento en que el protocolo de ruteo es notificado que no existen más receptores para esa dirección en ese link).

e) Reserved

Seteado a 0 por el emisor, ignorado por el receptor.

f) Multicast Address

Este campo es seteado a 0 si el mensaje es un General Query, o a una dirección multicast IPv6 específica si el mensaje es un Multicast Address Specific Query. En un mensaje Report o Done contiene una dirección multicast específica para la cual el emisor está escuchando o dejando de hacerlo, respectivamente.

## 2.2 - Descripción del protocolo

Los routers usan el MLD para aprender que direcciones multicast tienen receptores en cada uno de los links conectados. Por cada link conectado, el router mantiene una lista de las direcciones multicast que tienen receptores y un timer asociado a cada una de ellas. El router solo necesita saber si existen receptores para un determinado grupo, no necesita conocer la identidad de los receptores (por ejemplo, la dirección unicast) ni tampoco cuantos receptores existen en el link.

Para cada link conectado, el router selecciona una de sus direcciones unicast de link-local en ese link para utilizarla, como dirección origen, en todo los paquetes MLD que transmite en ese link.

Cada interface, en la cual el router está ejecutando el protocolo MLD, debe ser configurada para que pueda reconocer todas las direcciones multicast de capa de enlace que son generadas por IPv6 multicast. Por ejemplo, si el router está conectado a una red Ethernet, la interface debe reconocer todos los frames en los que la dirección destino comienzan con el valor hexadecimal 3333.

Para cada uno de sus links conectados, un router puede asumir uno de los dos siguiente roles:

- Querier, o
- Non-Querier

Normalmente hay un solo Querier por link. Todos los routers comienzan como Querier en cada uno de sus redes. Si el router escucha un mensaje Query, cuya dirección IPv6 origen es menor a la que él seleccionó para ese link, debe cambiarse a Non-Querier (en ese link únicamente). Si pasa un tiempo determinado sin recibir ningún mensaje Query, el router asume el rol de Querier (en ese link únicamente).

Un router Querier, periódicamente, envía un General Query al link para solicitar un reporte de todas las direcciones multicast de interés en ese link.

Los General Queries son enviados con alcance del link, a la dirección multicast de todos los nodos (FF02::1), con el campo Multicas Address igual a 0 y un Maximum Response Delay seteado.

A continuación se muestra un paquete Multicast Listener Query:

```
Frame 45 (86 on wire, 86 captured)
    Arrival Time: May  5, 2003 16:48:46.523473000
    Time delta from previous packet: 14.810923000 seconds
    Time relative to first packet: 68.555895000 seconds
    Frame Number: 45
    Packet Length: 86 bytes
    Capture Length: 86 bytes
Ethernet II
    Destination: 33:33:00:00:00:01 (33:33:00:00:00:01)
    Source: 00:60:97:1d:7a:70 (3Com_1d:7a:70)
    Type: IPv6 (0x86dd)
Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 32
    Next header: IPv6 hop-by-hop option (0x00)
    Hop limit: 1
    Source address: fe80::260:97ff:fe1d:7a70
    Destination address: ff02::1 (ff02::1)
Hop-by-hop Option Header
```

```
    Next header: ICMPv6 (0x3a)
    Length: 0 (8 bytes)
    PadN: 2 bytes
    Router alert: MLD (4 bytes)
Internet Control Message Protocol v6
    Type: 130 (Multicast listener query)
    Code: 0
    Checksum: 0x462a (correct)
    Maximum response delay: 10000
    Multicast Address: ::
```

La siguiente es una explicación de los campos más importantes:

- Destination: 33:33:00:00:00:01 (33:33:00:00:00:01)
  Como se explicó en el Capítulo 2, las direcciones de link-layer en Ethernet, que comienzan con 33:33, corresponden a direcciones multicast de IPv6.

- Hop limit: 1
  Si existe otro router en el link, éste tiene que descartar este paquete y no debe reenviarlo fuera del link.

- Source address: fe80::260:97ff:fe1d:7a70
  El router envía sus queries con la dirección link-local, de la interface por donde se envía el mensaje, como dirección origen

- Destination address: ff02::1 (ff02::1)
  El router envía sus queries a todos los nodos en el link utilizando la dirección multicast de todos los nodos.

- Hop-by-hop Option Header
  Router alert: MLD (4 bytes)
  Explicada en la Figura 3.3

- Type: 130 (Multicast listener query)
  Indica que el tipo de paquete ICMPv6 es un Query General.

- Maximum response delay: 10000
  Especifica que los nodos tienen 10000 milisegundos, como máximo, para contestar a este Query General.

- Multicast Address: ::
  Como el mensaje es un Query General, no se especifica ninguna dirección.

Un router puede enviar un Query a una dirección específica. Solamente los nodos que estén asociados a la dirección que el router está consultando responderán al query.

A continuación se muestra un paquete Multicast Address Specific Query:

```
Frame 103 (86 on wire, 86 captured)
    Arrival Time: May  5, 2003 16:49:22.712856000
    Time delta from previous packet: 0.000836000 seconds
    Time relative to first packet: 104.745278000 seconds
    Frame Number: 103
    Packet Length: 86 bytes
    Capture Length: 86 bytes
Ethernet II
    Destination: 33:33:00:00:01:25 (33:33:00:00:01:25)
    Source: 00:60:97:1d:7a:70 (3Com_1d:7a:70)
    Type: IPv6 (0x86dd)
Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 32
    Next header: IPv6 hop-by-hop option (0x00)
    Hop limit: 1
    Source address: fe80::260:97ff:fe1d:7a70
    Destination address: ff1e::125 (ff1e::125)
Hop-by-hop Option Header
    Next header: ICMPv6 (0x3a)
    Length: 0 (8 bytes)
    PadN: 2 bytes
    Router alert: MLD (4 bytes)
Internet Control Message Protocol v6
    Type: 130 (Multicast listener query)
    Code: 0
    Checksum: 0x67ce (correct)
    Maximum response delay: 1000
    Multicast Address: ff1e::125
```

Examinando los campos del paquete se puede observar que son muy pocas las diferencias que existen con un Query General. Los siguiente puntos explican los campos más relevantes:

- Destination address: ff1e::125 (ff1e::125)
  El paquete se envía a la dirección multicast que se está consultando, y no, a la dirección multicast de todos los nodos.

- Type: 130 (Multicast listener query)
  El valor de este campo es igual al que llevan los mensajes Query General. Como el campo Multicast Address no está vacío, podemos saber que el mensaje es una consulta específica a una dirección.

- Multicast Address: ff1e::125
  Indica cual es la dirección de multicast que se está consultando.

Cuando un nodo comienza a escuchar a una dirección multicast, en una interface, debería transmitir inmediatamente un Report no solicitado. También, debe enviar un Report cuando el timer del nodo, seteado igual al valor recibido en el campo Maximun Response Delay de un mensaje Query, para una dirección multicast en particular y, en una interface particular, expira.

Un paquete Multicast Listener Report tiene el siguiente formato:

```
Frame 19 (86 on wire, 86 captured)
    Arrival Time: May  5, 2003 16:48:12.943647000
    Time delta from previous packet: 0.007575000 seconds
    Time relative to first packet: 34.976069000 seconds
    Frame Number: 19
    Packet Length: 86 bytes
    Capture Length: 86 bytes
Ethernet II
    Destination: 33:33:ff:00:00:00 (33:33:ff:00:01:30)
    Source: 00:60:97:1d:7a:70 (3Com_1d:7a:70)
    Type: IPv6 (0x86dd)
Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 32
    Next header: IPv6 hop-by-hop option (0x00)
    Hop limit: 1
    Source address: fe80::260:97ff:fe1d:7a70
    Destination address: ff1e::130 (ff1e::130)
Hop-by-hop Option Header
    Next header: ICMPv6 (0x3a)
    Length: 0 (8 bytes)
    PadN: 2 bytes
    Router alert: MLD (4 bytes)
Internet Control Message Protocol v6
    Type: 131 (Multicast listener report)
    Code: 0
    Checksum: 0x6f34 (correct)
    Maximum response delay: 0
    Multicast Address: ff1e::130
```

Los siguientes parámetros son lo más interesantes para describir:

- Destinación address: ff1e::130 (ff1e: 130)
  La dirección que está siendo reportada es indicada en este campo.

- Tape: 131 (Multicast listener report)
  Indica que el tipo de paquete ICMPv6 es un Multicast Listener Report.

- Maximum response delay: 0
  Este campo solamente es seteado en los mensajes Query, en los demás su valor es igual a cero.

- Multicast Address: ff1e::130
  Se indica, también, la dirección multicast que está siendo reportada.

Si un nodo recibe un Report por una interface, para una dirección multicast mientras tiene un timer corriendo para esa misma dirección en esa interface, debe detener el timer y no enviar un Report para esa dirección con lo cual evita la duplicación de reports en el link.

Cuando un nodo deja de escuchar a una dirección multicast en una interface, debería enviar un solo mensaje Done a la dirección multicast de todos los routers (FF02::2), transportando en el campo Multicast Address la dirección multicast que está dejando de escuchar.

```
Frame 102 (86 on wire, 86 captured)
    Arrival Time: May  5, 2003 16:49:22.712020000
    Time delta from previous packet: 0.001134000 seconds
    Time relative to first packet: 104.744442000 seconds
    Frame Number: 102
    Packet Length: 86 bytes
    Capture Length: 86 bytes
Ethernet II
    Destination: 33:33:00:00:00:02 (33:33:00:00:00:02)
    Source: 00:60:97:59:0e:3d (paturuzu)
    Type: IPv6 (0x86dd)
Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 32
    Next header: IPv6 hop-by-hop option (0x00)
    Hop limit: 1
    Source address: fe80::260:97ff:fe59:e3d
    Destination address: ff02::2 (ff02::2)
Hop-by-hop Option Header
    Next header: ICMPv6 (0x3a)
    Length: 0 (8 bytes)
    Router alert: MLD (4 bytes)
    PadN: 2 bytes
Internet Control Message Protocol v6
    Type: 132 (Multicast listener done)
    Code: 0
    Checksum: 0xd6ec (correct)
    Maximum response delay: 0
    Multicast Address: ff1e::130
```

- Destination address: ff02::2 (ff02::2)
  Los nodos envían los mensajes Done a la dirección multicast de todos los routers. Es a ellos, a quienes un host debe avisar que está dejando de escuchar en una dirección determinada.

- Type: 132 (Multicast listener done)
  Indica que el paquete ICMPv6 es un Multicast Listener Done.

- Multicast Address: ff1e::130
  Indica la dirección multicast que un host ha cesado de escuchar.

Cuando un router en estado Querier recibe un mensaje Done desde un link, si la Multicast Address indicada en el mensaje está presente en la lista de direcciones que tienen receptores en el link, debe enviar un mensaje Multicast Address Specific Query, a esa dirección multicast, con un valor en el campo Maximun Response Delay. Si vencido este tiempo, no se recibe ninguna contestación, asume que no hay más receptores para esa dirección en el link, por lo cual, es eliminada y, su desaparición, dada a conocer al protocolo de

ruteo multicast. Los routers que están en estado Non-Querier deben ignorar los mensajes Done.

## 3. - Diferencia entre IPv4 e IPv6 Multicast

La principal diferencia que existe entre IPv6 Multicast y su antecesor, IPv4, es que en la nueva versión de IP el multicasting es obligatorio, es decir que todo nodo que implemente IPv6 debe saber procesar paquetes multicast, en cambio, en IPv4 era opcional.

En IPv6, el Multicast Listener Discovery es equivalente al Internet Group Management Protocol, versión 2, (IGMPv2) de IPv4. Pero a diferencia de éste, el MLD no define una estructura de mensajes propia, si no, que utiliza mensajes de ICMPv6.

En la figura siguiente se puede observar que en IPv6 el protocolo MLD forma parte de ICMPv6, y no define su propio formato de mensajes como en IPv4.

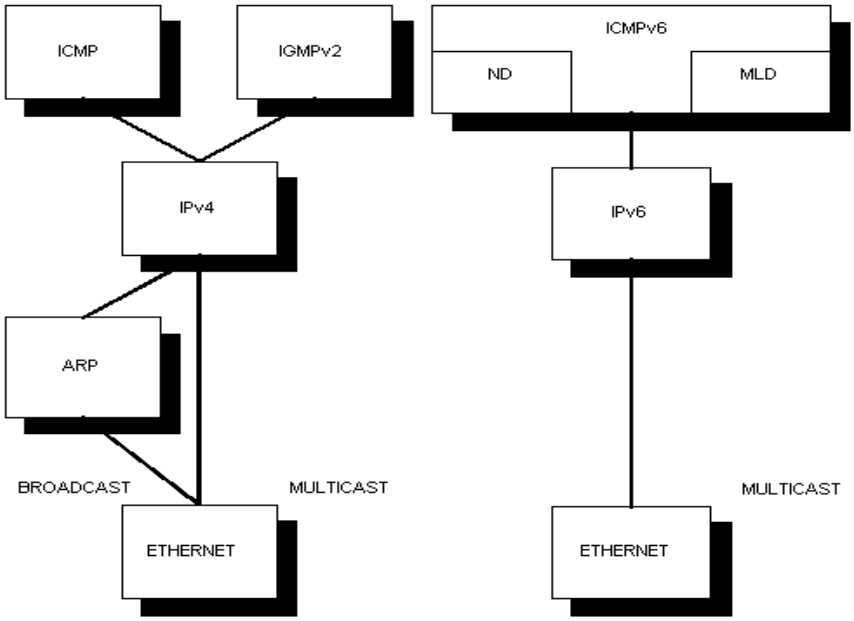


Figura 3.5 - Diferencias entre IGMPv2 y MLD

Los mensajes MLD, como los de IGMP, tiene alcance en el link. Esto lo determina el campo Hop Limit en la cabecera IPv6 , y el campo Time to Live en la cabecera IPv4, que se setea a 1, con lo cual, los routers deben descartar estos paquetes y no retransmitirlos.

En cambio, para enviar información a un grupo multicast, ambos protocolos utilizan el UDP. En IPv6, el alcance de un paquete UDP enviado a una dirección multicast está limitado, además de por el campo Hop Limit de la cabecera IPv6, por un campo en la misma dirección, mientra que en IPv4 el alcance está especificado en el campo Time To Live de la cabecera IP.

# Capítulo 4
# Multicast sobre IPv6

## 1. - Introducción

Para poder realizar las pruebas de multicast sobre IPv6 se utilizaron herramientas multimedia con soporte multicast. Estas herramientas, que son *open-source*, se bajaron de Internet de la siguiente página:

http://www-mice.cs.ucl.ac.uk/multimedia/software/

Entre todas las herramientas disponibles, con soporte IPv6, se optó por las siguientes tres:

1. RAT (Robust Audio Tool)
2. VIC (Video Conference Tool)
3. NTE (Network Text Editor)

A continuación damos una explicación de lo que realizan cada una ellas:

### 1.1 - Robust Audio Tool (RAT)

RAT es una aplicación de audio.

Esta herramienta permite a los usuarios participar en conferencias de audio sobre internet. La comunicación puede ser entre dos participantes conectados directamente, o entre varios participantes, todos conectados a una dirección multicast común.

RAT puede ejecutarse en un rango de plataformas: FreeBSD, HP-UX, IRIX, Linux, NetBSD, Solaris, SunOS, y Windows 95/NT.

Para ejecutar la aplicación se utiliza el siguiente comando:

rat 'dirección multicast IPv6'/port.

Por ejemplo, el siguiente comando ejecuta la aplicación para que se conecte a la dirección multicast ff1e::130 y en el puerto 6500

rat ff1e::130/6500

El siguiente gráfico muestra como es la interface del programa:

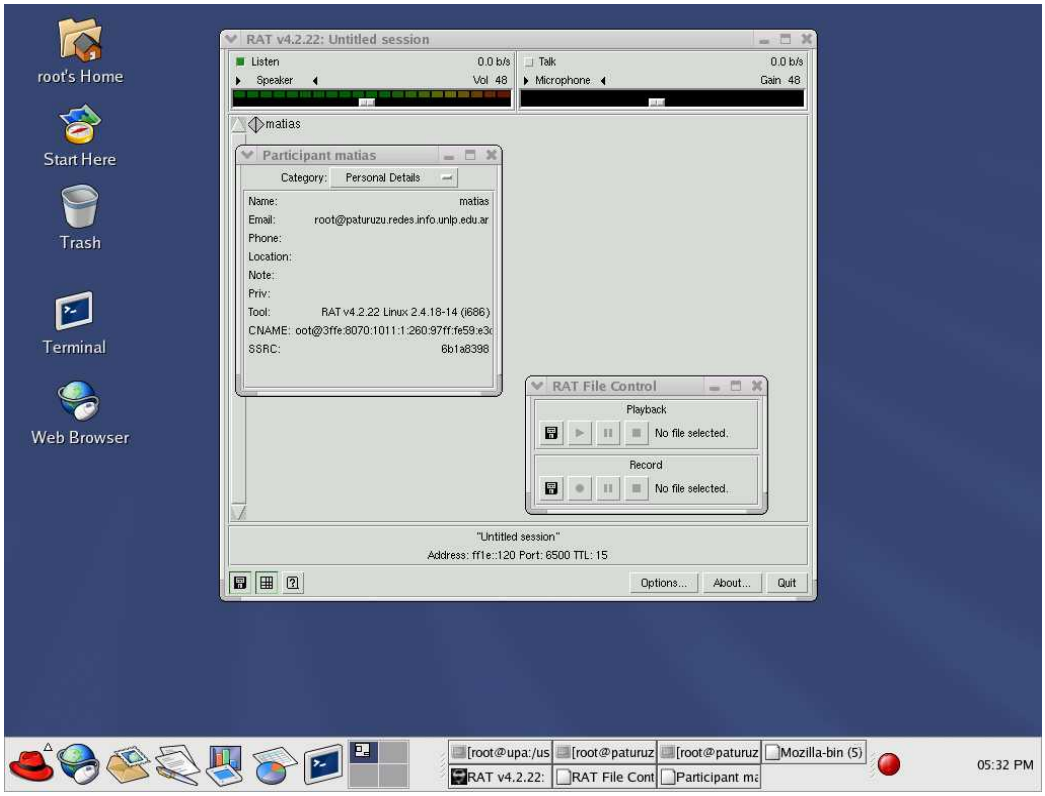


Figura 4.1 - Interface Robust Audio Tool

## 1.2 - Video Conference Audio (VIC)

VIC es una herramienta de video multicast (o unicast).

Se ejecuta de la misma forma que RAT (escribiendo vic en vez de rat) y también soporta las mismas plataformas.

Para ejecutarlo:

vic 'dirección multicast IPv6'/port.

El siguiente gráfico muestra como es la interface de la herramienta:

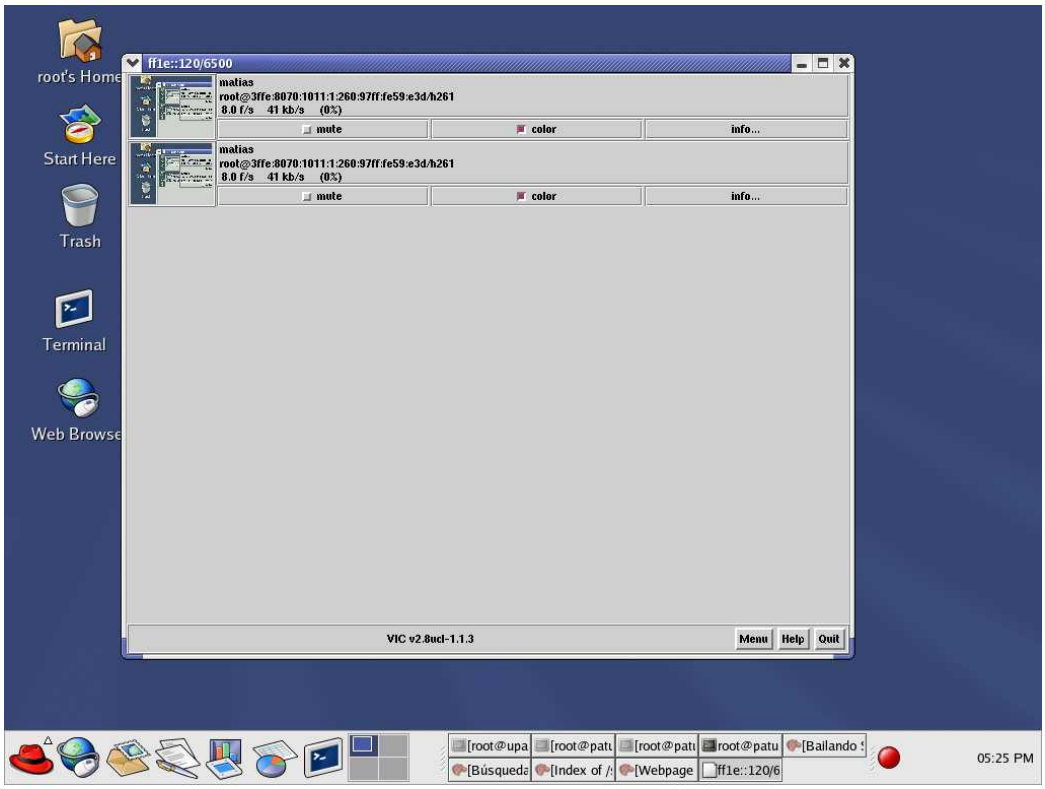


Figura 4.2 - Interface Video Conference Tool

### 1.3 - Network Tet Editor (NTE)

NTE es un editor de texto compartido. Permite, que varias personas editen el mismo documento simultáneamente, y que lo modifiquen. Salvo que un usuario bloquee una porción del texto, éste puede ser modificado y borrado por cualquiera de los miembros.

Si dos usuarios editan la misma línea al mismo tiempo se producirá un conflicto, y ocurrirá que solamente uno de los cambios será preservado.

NTE se puede ejecutar en las mismas plataformas que las dos herramientas anteriores, y se ejecuta de igual manera:

nte 'dirección multicast IPv6'/port

El siguiente gráfico muestra como es la interface del programa:

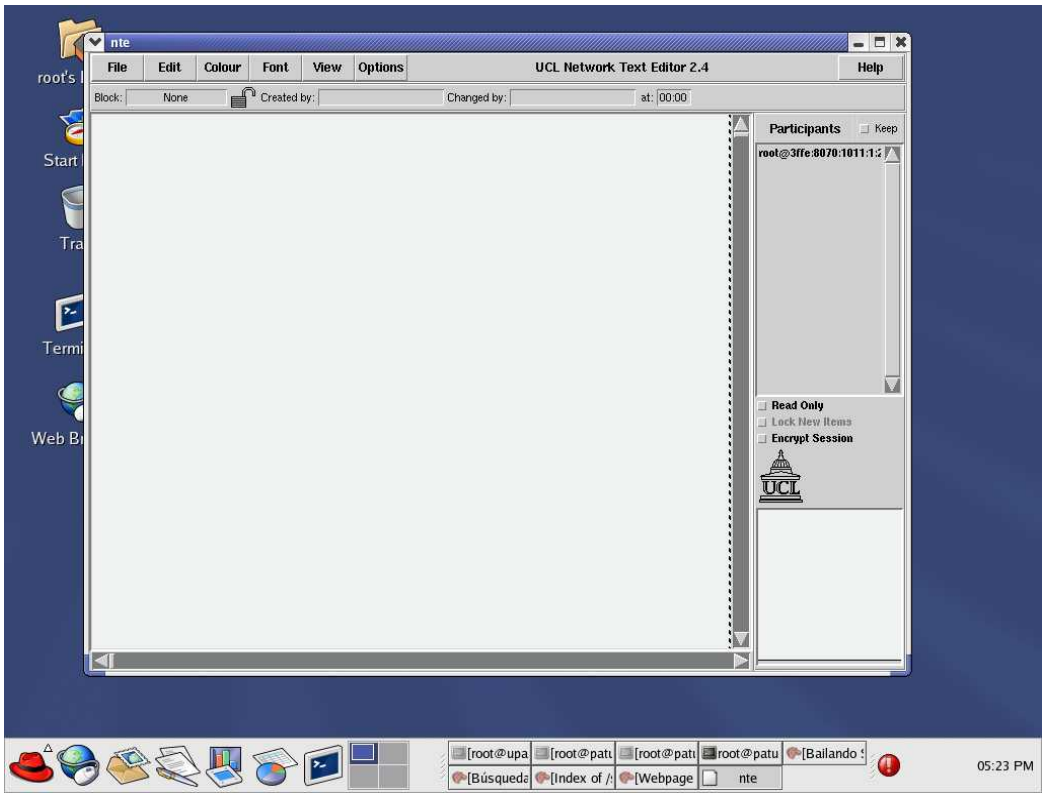


Figura 4.3 - Interface Network Text Editor

## 2. - Escenarios de prueba

Las pruebas que se realizaron fueron sobre los siguientes escenarios:

1. Las herramientas corren en distintos hosts y en la misma subred, en la cual no es necesaria la presencia de un router. El siguiente dibujo muestra la disposición.

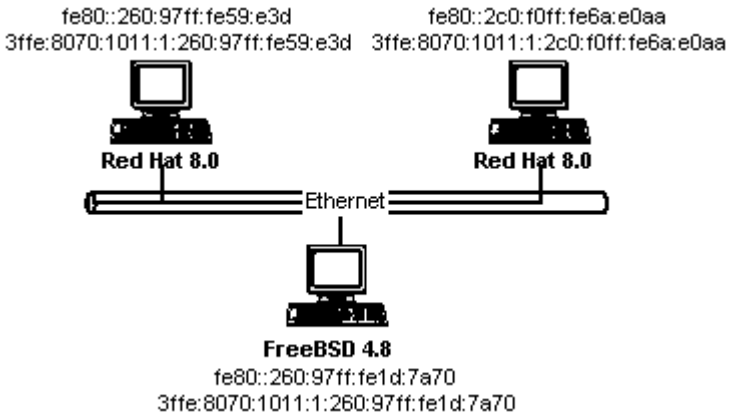


Figura 4.4 - Diseño primer escenario de prueba
de las aplicaciones multicast

2. Las herramientas corren en distintos hosts y en subredes diferentes. El siguiente dibujo muestra la nueva red:

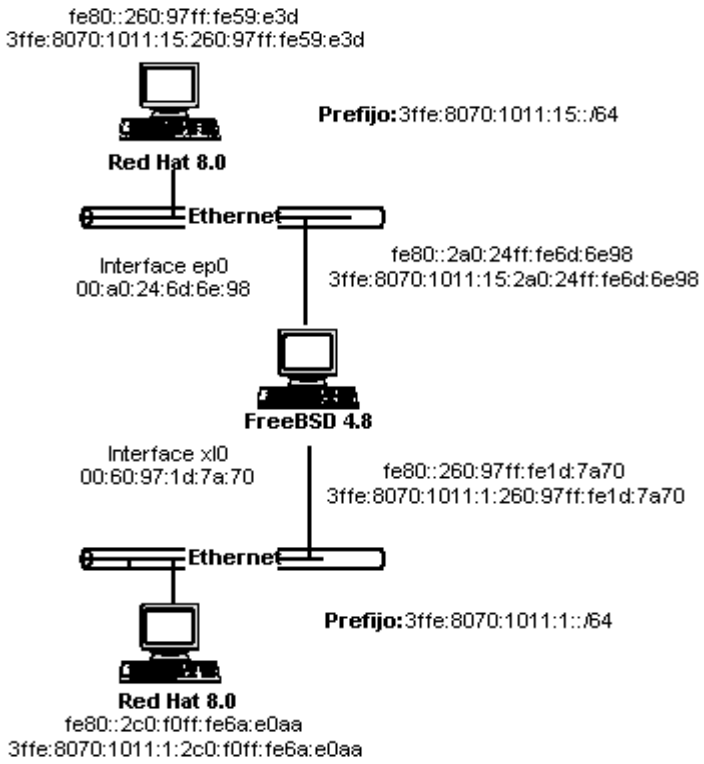


Figura 4.5 - Diseño segundo escenario de prueba
de las aplicaciones multicast

Existen algunas limitaciones para la selección del grupo multicast. La dirección multicast, como se explicó en el capítulo anterior, utiliza los últimos cuatros bits del segundo byte para indicar el alcance, o tiempo de vida, de los mensajes. Los alcances definidos son los siguientes:

- 1 = nodo local
- 2 = link-local
- 5 = site-local
- 8 = organization
- e = global

Observando estos valores, se deduce, que para el primer escenario, el valor de dicho byte debe ser, como mínimo, 2; y, para el segundo escenario, debe ser como mínimo, 5. Esto último se debe a que los paquetes son ruteados entre dos subredes y no se puede utilizar un alcance con un valor menor a 5: el router no reenvía paquetes con alcance de link-local o menor.

### 2.1 - Una sola subred

Como se puede observar en el gráfico, los nodos sobre los cuales se ejecutan las herramientas se encuentran en la misma subred.

Esta situación es muy simple. No es necesario que el router esté ejecutando el protocolo Multicast Listener Discovery, ni ningún protocolo de ruteo, o que haya un router presente en la red.

Las tres aplicaciones son idénticas en su funcionamiento. La única diferencia que existe es que VIC utiliza un número de puerto distinto, para intercambiar la información, al que se especifica cuando se ejecuta la aplicación.

A continuación se muestran las pruebas realizadas en una red como la que se observa en el primer escenario. En el punto 2.1.1, se muestra el funcionamiento de las aplicaciones sin el protocolo MLD habilitado en la red, y, en el punto 2.1.2, se realizan la misma prueba pero con dicho protocolo habilitado.

### 2.1.1 - Sin MLD

El siguiente intercambio de paquetes se produce entre tres nodos ejecutando la herramienta NTE. En este ejemplo no existe un router en la red.

La aplicación se unió al grupo multicast ff1e::120 y al puerto 6500. El comando es el siguiente:

```
# nte ff1e::120/6500.
```

El grupo multicast y el port se eligieron al azar, respetando el alcance del tipo de dirección.

| No. | Source | Destination | Protocol | Info |
|---|---|---|---|---|
| 1 | fe80::260:97ff:fe1d:7a70 | ff1e::120 | ICMPv6 | MLR |
| 3 | 3ffe:8070:1011:1::10 | ff1e::120 | UDP | SP:6500 DP:6500 |
| 9 | fe80::260:97ff:fe1d:7a70 | ff1e::120 | ICMPv6 | MLR |
| 15 | fe80::2c0:f0ff:fe6a:e0aa | ff1e::120 | ICMPv6 | MLR |
| 16 | 3ffe:8070:1011:1:2c0:f0ff:fe6a:e0aa | ff1e::120 | UDP | SP:6500 DP:6500 |
| 67 | fe80::2c0:f0ff:fe6a:e0aa | ff1e::120 | ICMPv6 | MLR |
| 164 | fe80::2c0:f0ff:fe6a:e0aa | ff02::2 | ICMPv6 | MLD |
| 203 | fe80::2c0:f0ff:fe6a:e0aa | ff1e::120 | ICMPv6 | MLR |
| 263 | fe80::2c0:f0ff:fe6a:e0aa | ff1e::120 | ICMPv6 | MLR |
| 399 | fe80::260:97ff:fe1d:7a70 | ff1e::120 | ICMPv6 | MLR |
| 424 | fe80::260:97ff:fe1d:7a70 | ff1e::120 | ICMPv6 | MLR |
| 563 | fe80::260:97ff:fe1d:7a70 | ff02::2 | ICMPv6 | MLR |

MLR: Multicast Listener Report, MLD: Multicast Listener Done
SP: Source Port, DP: Destination Port

En el mensaje Nro. 1, el nodo con la dirección fe80::260:97ff:fe1d:7a70, se une a la dirección multicast ff1e::120 enviando un mensaje Multicast

Listener Report. Como dirección origen utiliza la dirección de link-local asociada a la interface. Inmediatamente comienza a transmitir información enviando paquetes UDP a la dirección multicast y al puerto indicado. En estos paquetes se utiliza una dirección IPv6 global como dirección origen.

Para los casos en que los paquetes multicast no son reenviados por el router, es decir, que no salen del link, la dirección origen de los mensajes UDP, que tienen direcciones multicast como dirección destino, puede ser una dirección de link-local, o con un alcance mayor. Pero, si el paquete es retransmitido por el router, la dirección origen debe ser una dirección en la que su alcance sea, como mínimo, igual al alcance del grupo multicast al que se está enviando el mensaje. Por ejemplo, si el grupo multicast se expande más allá de la organización, la dirección origen debe ser una dirección global para que el router reenvíe el paquete fuera del sitio.

Como indica la RFC 2710, el nodo envía dos mensajes de reporte (en el mensaje Nro. 9 envía el segundo) por si el primero se pierde. Esto no es obligatorio, depende de la implementación.

En el paquete 13, el nodo con la dirección de link-local fe80::2c0:f0ff:fe6a:e0aa se agrega al grupo multicast ff1e::120. Envía el segundo mensaje de reporte en el mensaje 65.

Todos los mensajes que no se muestran son paquetes UDP, en los cuales, las aplicaciones intercambian información.

En el mensaje 164, un nodo abandona el grupo multicast y, por ese motivo, envía un paquete Multicast Listener Done a la dirección multicast de todos los routers. También utiliza la dirección de link-local como dirección origen.

En los mensajes 203 y 424, el nodo que había abandonado el grupo, en el mensaje 164, vuelve a unirse.

En los mensajes 399 y 424 se observa que el nodo con la dirección de link-local fe80::260:97ff:fe1d:7a70 se une al grupo multicast, pero este host se había unido al mismo grupo en los mensajes 1 y 9, y nunca había anunciado su abandono (es decir, no envió un mensaje Done). Esto se debe a que, únicamente, el último host que se une al grupo envía un mensaje Done (como vemos que este nodo lo hace en mensaje 563), y todos los host que se unieron con anterioridad al grupo no envían nada al dejarlo, sin importar si lo hacen antes o después del último nodo que ingresó al grupo.

### 2.1.2 - Con MLD

Un segundo ejemplo sobre este escenario, es probar la aplicación pero con el protocolo MLD ejecutando en un router, el nodo con FreeBSD. A continuación se muestra como es el intercambio de mensajes entre dos aplicaciones multicast ejecutando en la misma subred:

| No. | Source | Destination | Protocol | Info |
|---|---|---|---|---|
| 15 | fe80::260:97ff:fe1d:7a70 | ff02::1 | ICMPv6 | MLQ |
| 24 | fe80::260:97ff:fe1d:7a70 | ff02::d | ICMPv6 | MLR |
| 28 | fe80::260:97ff:fe1d:7a70 | ff02::9 | ICMPv6 | MLR |
| 34 | fe80::2c0:f0ff:fe6a:e0aa | ff1e::120 | ICMPv6 | MLR |
| 39 | fe80::2c0:f0ff:fe6a:e0aa | ff1e::120 | ICMPv6 | MLR |
| 102 | fe80::260:97ff:fe59:e3d | ff1e::120 | ICMPv6 | MLR |
| 104 | fe80::260:97ff:fe59:e3d | ff1e::120 | ICMPv6 | MLR |
| 196 | fe80::260:97ff:fe59:e3d | ff02::2 | ICMPv6 | MLD |
| 197 | fe80::260:97ff:fe1d:7a70 | ff1e::120 | ICMPv6 | MLQ |
| 198 | fe80::2c0:f0ff:fe6a:e0aa | ff1e::120 | ICMPv6 | MLR |
| 222 | fe80::2c0:f0ff:fe6a:e0aa | ff02::2 | ICMPv6 | MLD |
| 223 | fe80::260:97ff:fe1d:7a70 | ff1e::120 | ICMPv6 | MLQ |
| 224 | fe80::260:97ff:fe1d:7a70 | ff1e::120 | ICMPv6 | MLQ |

MLD:Multicas Listener Done; MLQ:Multicas Listener QueryDone;
MLD:Multicas Listener Report
SP: Source Port DP: Destination Port

Con la activación del Multicast Listener Discovery en el router, más precisamente en el PC Router, el intercambio de paquetes muestra nuevos tipos de mensajes.

El mensaje Nro. 15 muestra un paquete Multicast Listener Query. Como está dirigido a la dirección multicast de todos los nodos es un query general. Los mensajes 16 y 17 son mensajes de reporte en respuesta a esa consulta. Los nodos reportan todas las direcciones multicast que están escuchando (si dos o más nodos escuchan a la misma dirección, no es necesario que todos respondan, es suficiente que uno solo lo haga).

En los mensajes 34 y 39, el nodo con la dirección de link-local fe80::2c0:f0ff:fe6a:e0aa se une al grupo multicast ff1e::120 (ídem en los mensajes 102 y 104) enviando dos mensajes de reporte. Estos reportes son iguales a los enviados en contestación a un query. La manera de diferenciarlos es la proximidad a un mensaje de consulta.

En el mensaje 196, el nodo con la dirección fe80::260:97ff:fe59:e3d envía un mensaje Multicast Listener Done para indicar que deja el grupo multicast ff1e::120.

Como el router no sabe cuántos nodos están unidos a un grupo determinado en un link, debe consultar a los demás nodos para saber si existen más receptores para ese grupo. Con este fin, envía el mensaje 197, que es un query específico a la dirección de grupo ff1e::120. Unicamente los hosts que estén unidos a ésta dirección, contestarán (al igual que en el query general, es suficiente que uno solo conteste).

El mensaje 198, es un reporte del host con la dirección de link-local fe80::2c0:f0ff:fe6a:e0aa indicando que él todavía pertenece a ese grupo. Si nadie contesta, el router determina que no existe nadie más, en el link, interesado en esa dirección. Esto último sucede en siguientes tres mensajes que se muestran (mensajes 222, 223 y 224). Este mismo nodo envía un mensaje Done indicando que deja el grupo. El router, al recibirlo, envía dos mensajes Query Specific para ver si existe algún receptor más, en el link, para ese grupo multicast (el ff1e::120). Como nadie constesta, el router asume que no existe ningún nodo unido a este grupo.

### 2.2 - Dos subredes

Los hosts, sobre los que corren las herramientas, están ubicados en dos subredes distintas, conectadas a través de un router.

El primer paso para poder aplicar este modelo fue establecer un router que soportase el ruteo multicast.

Al momento de realizar este trabajo, los IOS de Cisco no soportaban, ni el protocolo MLD ni el ruteo multicast. Solamente existían versiones experimentales de la tercer fase de desarrollo de IPv6. Esto se debe, a que Cisco decidió realizar la implementación de IPv6 en tres fases, y recién, en la última fase, estarían disponibles ambos protocolos.

Entre los sistemas operativos que soportan ambos protocolos se encuentran FreeBSD y Windows 2000. Se decidió utilizar la versión 4.8 de FreeBSD (como se mostró en el Capítulo 2).

Este sistema operativo soporta el protocolo de ruteo multicast Protocol Independent Multicast para IPv6 en sus dos versiones:

- PIM6-SM (PIM6-Sparse Mode)
- PIM6-DM (PIM6-Dense Mode)

Las pruebas se realizaron sobre ambos protocolos. A continuación se muestra como se los configura para que el router pueda comenzar a reenviar tráfico multicast.

Por defecto, el ruteo multicast está deshabilitado en FreeBSD. Se lo puede habilitar modificando el archivo /etc/rc.conf.

Por problemas de licencias, ninguno de los dos protocolos se instalan en la versión FreeBSD-Realase, pero están disponibles en *packages* o *ports* para ser instalados.

Nota: en el Apéndice C se explica el funcionamiento de los protocolos de ruteo multicast.

PIM6-DM

Para instalarlo se debe ir al directorio /usr/ports/net/pim6dd y ejecutar el siguiente comando:

*make install*

Para ejecutar el protocolo automáticamente cuando se inicia el router se deben agregar, en el archivo /etc/rc.conf, las siguiente líneas:

```
#Habilitamos el ruteo multicast
mroute6d_enable="YES"
#Indicamos el nombre, y donde se encuentra, el proceso
mroute6d_program="/usr/local/sbin/pim6dd"
```

El proceso se autoconfigura para comenzar a reenviar tráfico multicast en todas las interfaces con capacidad de multicast.

Este protocolo no necesita ningún tipo de configuración adicional, es decir, que una vez que está ejecutándose, inmediatamente comienza a rutear el tráfico multicast entre ambas subredes.

PIM6-SM

Al igual que PIM6-DM, se debe instalar manualmente. Pero a diferencia de éste, el directorio, en el cual, se debe ejecutar el comando *make install* es /usr/ports/net/pim6sd.

También se deben agregar las dos líneas en el archivo /etc/rc.conf, como en el caso anterior, pero es necesario modificar el nombre del archivo (se reemplaza *pim6dd* por *pim6sd*):

```
mroute6d_enable="YES"
mroute6d_program="/usr/local/sbin/pim6sd"
```

PIM6-SM no comienza a rutear inmediatamente como PIM6-DM. Para que esto suceda, se deben realizar algunas configuraciones adicionales.

Por la forma de trabajar, PIM6-SM necesita que en el dominio multicast exista un router que funcione como Rendezvous Point.

El proceso pim6sd utiliza el archivo /usr/local/etc/pim6sd.conf para autoconfigurarse. Esto se puede modificar con la opción –c, y así hacer que el proceso se configure desde un archivo en otro directorio. Por ejemplo:

pim6sd -c /etc/pim6sd.conf

Ahora, pim6sd tomará los parámetros de configuración del archivo pim6sd.conf del directorio /etc.

Este protocolo es más complejo que pim6dd, pero también es más completo, al permitir realizar configuraciones adicionales.

Entre estas configuraciones se encuentra la posibilidad de establecer los parámetros para el funcionamiento del Multicast Listener Discovery sobre cada una de las interfaces donde el protocolo está habilitado.

El siguiente archivo muestra como quedó configurado el archivo pim6sd.conf:

```
#Indicamos que los query del MLD, en la interface xl0, se enviarán
#cada 90 segundos, que los hosts tendrán 2000 milisegundos para
#contestar y que la versión del MLD utilizada es 1.
phyint xl0 query_interval 90 query_rsp_interval 2000 mld_version 1;

#Igual que para la interface xl0, pero con distintos valores.
phyint ep0 query_interval 10 query_rsp_interval 5000 mld_version 1;

#Indicamos que el router funciona como Rendezvous Point.
cand_rp;

#Indicamos cual es el grupo multicast que se ruteará.
group_prefix ff1e::/16;
```

Para que el protocolo funcione, solamente, es necesario configurar el router como Rendezvous Point e, indicarle, cuál es el grupo multicast que deberá rutear.

Con esta configuración, los valores en un mensaje Query del protocolo MLD son los siguientes:

```
......
Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 32
    Next header: IPv6 hop-by-hop option (0x00)
    Hop limit: 64
    Source address: fe80::260:97ff:fe1d:7a70
    Destination address: ff1e::130
Hop-by-hop Option Header
    Next header: ICMPv6 (0x3a)
    Length: 0 (8 bytes)
    PadN: 2 bytes
    Router alert: MLD (4 bytes)
Internet Control Message Protocol v6
    Type: 130 (Multicast listener query)
    Code: 0
    Checksum: 0xcc4c (correct)
    Maximum response delay: 2000
    Multicast Address: ::
```

Como vemos el valor del campo Maximum response delay es 2000, que es el valor que se estableció en el archivo pim6sd.conf.

Con los protocolos de ruteo multicast funcionando se realizó la siguiente prueba: utilizando dos hosts, cada uno ubicado en una subred diferente, se ejecutó la herramienta NTE en ambos.

El comando para realizar la ejecución de la aplicación es el mismo que para el primer escenario.

El funcionamiento del protocolo MLD es similar al caso explicado en el punto anterior, pero se ejecuta en ambas interfaces. Además, es el encargado de indicarle al protocolo de ruteo multicast de los grupos que tienen receptores en cada una de las interfaces, para que sepa a donde reenviar los mensajes, así como de los grupos que no tienen más receptores en alguna de los links para que deje de reenviar por las interfaces correspondientes.

La diferencia con el primer escenarios radica, en que los paquetes UDP enviados a la dirección multicast en una subred, son ruteados por el PC Router para que puedan ser recepcionados por la aplicación ejecutando en la otra subred.

Los mensajes de MLD no son ruteados por el router, solamente tienen alcance local, no salen del link.

A continuación, se muestra un paquete UDP que es reenviado por el router utilizando la interface ep0.

```
Frame 231 (409 bytes on wire, 409 bytes captured)
    Arrival Time: Jul 24, 2003 09:10:17.252013000
    Time delta from previous packet: 0.033529000 seconds
    Time relative to first packet: 91.674327000 seconds
    Frame Number: 231
    Packet Length: 409 bytes
    Capture Length: 409 bytes
Ethernet II, Src: 00:a0:24:6d:6e:98, Dst: 33:33:00:00:01:20
    Destination: 33:33:00:00:01:20 (33:33:00:00:01:20)
    Source: 00:a0:24:6d:6e:98 (00:a0:24:6d:6e:98)
    Type: IPv6 (0x86dd)
Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 355
    Next header: UDP (0x11)
    Hop limit: 15
    Source address: 3ffe:8070:1011:1:2c0:f0ff:fe6a:e0aa
    Destination address: ff1e::120 (ff1e::120)
User Datagram Protocol, Src Port: 6500, Dst Port: 6500
    Source port: 6500 (6500)
    Destination port: 6500 (6500)
    Length: 355
    Checksum: 0x3f05 (correct)
Data (347 bytes)
```

Este paquete se envía por la interface ep0 del PC Router, a la dirección multicast de Ethernet. Es ruteado por el protocolo de ruteo multicast porque el prefijo de red de la dirección origen es 3ffe:8070:1011:1::/64, y la subred que se publica por la interface ep0 es 3ffe:8070:1011:15::/64.

Con el comando netstat -g podemos ver distintos datos estadísticos del ruteo multicast:

```
freebsd# netstat -g
No IPv4 multicast routing compiled into this system.

IPv6 Multicast Interface Table
 Mif   Rate    PhyIF    Pkts-In    Pkts-Out
   0      0      xl0      227         142      (1)
   1      0      ep0      142         227
   2      0    faith0       0           0
   4      0     reg0        0           0

IPv6 Multicast Forwarding Cache
Origin                                  Group     Packets Waits In-Mif Out-Mifs
3ffe:8070:1011:15:260:97ff:fe59:e3d  ff1e::120    45     0     1       0
3ffe:8070:1011:1::10                 ff1e::120    118    0     0       1
```

Las únicas interfaces que se muestran son las que soportan multicast, es por esto que no se muestra la dirección de loopback

Las direcciones unicast corresponden a las Source Address de los paquetes UDP. Esto permite que el protocolo de ruteo multicast pueda realizar su trabajo, al aprender las redes que tienen nodos escuchando un determinado grupo y, por cuales interface/s tiene que reenviar los paquetes multicast. Por ejemplo, los paquetes que tienen dirección origen 3ffe:8070:1011:1::10, entran por la interface multicast 0 (que corresponde a la interface física xl0) y salen por la interface 1 (que corresponde a la interface física ep0). Puede existir más de una interface de salida.

El campo Group indica todos los grupos multicast que tienen al menos un nodo asociado. Una dirección origen puede aparecer ligada a más de un grupo multicast.

Para cada dirección origen, y su grupo asociado, se muestran la cantidad de paquetes que entraron en una interface, y por cual interface salieron. La cantidad de paquetes que entraron por una interface, y salieron por otra, no coincide con los valores mostrados en (1). Esto se debe a que en esta última se refleja la suma de los paquetes enviados por todos los nodos desde que se habilitó el ruteo multicast en el router.

# Conclusiones

Después de haber estudiado el protocolo IPv6 se puede realizar una comparación con su antecesor, IPv4, y analizar como se inserta en el mundo actual, específicamente, su acoplamiento a Internet2.

En primer lugar, la pregunta que nos deberíamos hacer es si el cambio de protocolo es realmente necesario. Teniendo en cuenta el tamaño del emprendimiento y el tiempo que éste llevaría, esta pregunta no debería ser tratada de manera displicente.

A pesar de la existencia de opiniones opuestas, que dicen que IPv4 puede seguir funcionando con la aplicación de nuevas modificaciones y protocolos, creo que son solo 'parches' y que tarde o temprano el uso del mismo será, como mínimo, demasiado complejo. Es decir, que esto sería solamente retrasar lo inevitable.

Observando la expansión de Internet y como se ha saturado en tan pocos años de uso masivo (tener en cuenta que se comenzó a popularizar mundialmente en la década del 90), con la llegada de nuevas tecnologías (por ejemplo, IP Móvil) y la necesidad de las empresas de proveer nuevos servicios usando la red, es evidente que el protocolo estará cada vez más al límite.

Sin dudas, la complejidad del cambio es el primer y principal punto negativo a tener en cuenta. No solamente se cambia el protocolo IP, también lo deben hacer todos los demás protocolos que forman IP para realizar sus tareas. Entre ellos, podemos citar TCP, UDP, ICMP, IGMP, ARP, etc. que serán modificados o reemplazados. Además, las aplicaciones comerciales (como los Sistemas Operativos) y particulares deberán ser actualizadas.

No solamente los protocolos y aplicaciones deberán ponerse al día, también lo tendrán que hacer, en mayor o menor medida, los administradores de redes, desarrolladores, etc. Esta cuestión no debe ser pasada por alto ya que posiblemente, vencer el miedo al cambio de las personas, sea el escollo más difícil que se deba superar.

Todas las personas que están involucradas en esta tarea aseguran que la transición hacia el nuevo protocolo será de varios años, y algunos van más allá, al afirmar que IPv4, quizás, nunca se deje de usar.

Son varios los puntos a favor del cambio que se pueden citar, entre ellos, la falta de direcciones de red para otorgar. Existen países que solamente tiene una dirección de clase C asignada, y hace poco tiempo a China solamente se le concedió una dirección de clase B para conectar todas sus escuelas (son alrededor de 65.000) a Internet.

Con IPv6, este problema se soluciona de manera holgada ya que la cantidad de direcciones disponibles es mucho mayor, pero además quienes son los encargados de asignarlas lo están haciendo de manera ordenada. Algo que, evidentemente, no se tuvo en cuenta en las primeras épocas de IPv4 (no se debe buscar culpables por esto, simplemente, la realidad superó, holgadamente, al más optimista de los pronósticos).

Tan grande es el rango de direcciones disponibles, que se supone, que no será la falta de éstas un motivo para cambiar el protocolo la próxima vez que se tenga que hacer, si no, la aparición de nuevas tecnologías que IPv6 no pueda soportar.

La cabecera IPv6 es de tamaño fijo y tiene 8 campos, contra los al menos 12 que tiene IPv4, lo cual, sumado a su tamaño fijo, permite que los routers sean más eficientes en el ruteo porque tienen menor cantidad de campos para examinar.

Aunque el tamaño de las direcciones en IPv6 es cuatro veces más grande que IPv4, la cabecera es 40 bytes, el doble de la cabecera de IPv4 (sin opciones). Una cabecera IPv6 ocupa alrededor del 2% de espacio en un frame Ethernet de 1500 bytes, mientras que IPv4 ocupa solamente el 1%. Esto puede ser tomado como una desventaja de IPv6.

La decisión de poder agregar cabeceras de extensión le brinda al protocolo una extensibilidad que no posee su antecesor. Esto le permitirá adecuarse a la llegada de nuevas tecnologías.

La composición de los formatos de direcciones ha sido pensada, no solamente para evitar su desaprovechamiento, si no, que también, para que los routers no tengan que manejar tablas de ruteo de tamaño gigantesco.

La autoconfiguración de direcciones permite tener, con muy poco conocimiento de los usuarios, una red operativa. Además, los administradores de red tendrán que hacer muy pocas modificaciones para, por ejemplo, cambiar la dirección de una red. Se podría decir que es *plug and play*.

El reemplazo de las direcciones de broadcast por las de multicast permite una mejor performance de los nodos porque solamente serán interrumpidos por paquetes enviados a sus direcciones unicast o a las direcciones multicast a las que el nodo se haya unido.

Al no autorizarlos a fragmentar los paquetes y, al haber eliminado el campo Checksum en la cabecera de IPv6 (se utiliza el checksum de la capa de transporte, y en UDP pasa a ser obligatorio), los routers tendrán mayor velocidad para reenvíar los paquetes.

Debemos recordar IPv4 e IPv6 no interoperan, es decir, que un nodo IPv4 no se puede comunicar directamente con uno IPv6, y que una aplicación IPv4 no trabaja con IPv6. Por esto existen varios métodos para realizar la

transición de IPv4 hacia IPv6. Algunos de ellos fueron probados en este desarrollo. Son muy simples de configurar y poner en funcionamiento.

En IPv6, el ruteo es similar a IPv4 con CIDR (Classless Internet Domain Routing), pero con la flexibilidad que permiten direcciones de 128 bits. De esto se desprende que las modificaciones en los protocolos de ruteo dinámicos (RIP, BGP, OSPF, etc.) no son muy profundas: cambios relacionados al formato de las direcciones y, en algunos casos, adaptarlos para que soporten IPv4 e IPv6 simultáneamente.

Con respecto al multicast, el funcionamiento de las aplicaciones sobre el protocolo IPv6 es similar que tienen en IPv4 (pruebas similares se desarrollaron usando ambos protocolos). Es lógico que esto sea así, porque el Multicast Listener Discovery es una adaptación del protocolo Internet Group Managment Protocol (IGMP), versión 2.

Las tres herramientas probadas presentaron un funcionamiento y rendimiento similar. La única diferencia la presentó la aplicación VIC, que usa un puerto distinto al especificado para el intercambio de paquete. Por ejemplo, si se indicaba el puerto 6500, como en todas las pruebas, la aplicación utilizaba el puerto 6501.

Las aplicaciones consumen un ancho de banda considerable. Por ejemplo, en el NTE se envían todos los movimientos del mouse para que se reflejen en los demás nodos que están ejecutando la herramienta. La ventaja, con respecto a unicast, radica en que, sin importar la cantidad de nodos ejecutando la aplicación en un link, un mismo paquete se envía una sola vez en esa subred. Con unicast, se tendría que enviar un paquete a cada nodo, esto incrementa en N la cantidad de paquetes enviados.

El router procesa menor cantidad de paquetes, con lo cual, tiene un mejor rendimiento, pero necesita un protocolo de ruteo especial porque los protocolos unicast, como BGP, RIP, etc. no son capaces de trabajar con direcciones multicast.

En IPv4, el tiempo de vida de los paquetes UDP enviados a una dirección multicast dependen del valor del campo Time-to-Live de la cabecera IP. En IPv6, el alcance de los paquetes está limitado por dos valores: el campo Hop-Limit de la cabecera IP y el valor del campo Scope de la dirección multicast. En menor alcance es el que prevalecerá.

Para determinar la dirección unicast de los nodos que están asociados al grupo multicast, las aplicaciones utilizan la dirección origen de los paquetes UDP, para poder mostrarlos en la ventana.

El protocolo de ruteo PIM-DM (Dense Mode) es más simple de configurar que el protocolo PIM-SM, pero ofrece menor control que este último.

La tecnología multicast tendrá, en el futuro, un papel muy importante, imprescindible. Si bien, su funcionamiento puede ser reemplazado por el envío de múltiples mensajes unicast, es impracticable que esto pueda ser realizado en una red como Internet. La cantidad de usuarios asociados a un grupo multicast y lo disperso que estos pueden encontrarse generarán una catarata de mensajes imposibles de tratar por los routers.

A diferencia de IPv4, donde era opcional, en IPv6 todos los nodos deberán soportar el multicast. Existe una razón principal para esto: el reemplazo de los mensajes broadcast por los de multicast, con todos las ventajas que esto conlleva; pero en forma secundaria, provee a los nodos la capacidad de poder utilizar todas las nuevas tecnologías que se están desarrollando y harán uso de esta técnica.

Intenet2: que papel tendrá IPv6

IPv6 no solamente se ha diseñado para solucionar los problemas que se encuentran en IPv4, también está pensado para poder adecuarse a nuevas ideas (el uso de las cabeceras de extensión podría permitir esto) que aparezcan en el futuro.

Teniendo en cuenta que Internet2 es un banco de pruebas para esos nuevos proyectos no es ilógico pensar en IPv6 como el protocolo más adecuado. Al tener ambos, Internet2 e IPv6, un desarrollo en paralelo, la integración puede ser más coherente que si se utilizase IPv4, al que seguramente, habría que seguir agregándole parches.

Si se observan los tipos de aplicaciones en las que se está trabajando en Internet2, es evidente que éstas deberán hacer uso del multicast para lograr un mejor funcionamiento.

Sin dudas, la combinación de Internet2 e IPv6 puede permitir una rápida y mejor integración entre las aplicaciones (una videoconferencia utilizando multicasting) con las herramientas (IPv6, que al tener el multicasting obligatorio no es necesario saber si todos los routers entre el origen y los destinos soportan multicast como sucede con IPv4).

Por último, creo que este trabajo deja abierta la puerta para una futura conexión al m6bone (la red mundial IPv6 con soporte multicast), lo que permitiría realizar pruebas multicast con universidades e institutos de todo el mundo. A su vez, podría ser tomado como punto de partida para el desarrollo de aplicaciones que hagan uso de la tecnología multicast.

# Apéndice A
# IPv6 en distintos SO

## A.- Windows NT 4.0 / Windows 2000 / Windows XP

Entre los comandos más comunes se encuentran:

- net stop tcpip6: detiene la pila IPv6 y la saca de memoria.

```
C:\>net stop tcpip6
El servicio de MSR IPv6 Protocol fue detenido con éxito.
```

- net start tcpip6: inicia la pila IPv6.

```
C:\>net start tcpip6
El servicio de MSR IPv6 Protocol se ha iniciado con éxito.
```

Una vez iniciada la ejecución de IPv6, toda la configuración se realiza con el comando ipv6, a excepción de IPSec. Este comando tiene varias opciones, entre las que se encuentran:

- ipv6 if #interface

Permite ver la configuración de una interface determinada. Si no se indica el número de interface, muestra todas las interfaces del nodo.

```
C:\>ipv6 if 4
Interface 4 (site 1):
  cable reconnected
  uses Neighbor Discovery
  link-level address: 00-01-02-76-01-e5
    preferred address fe80::201:2ff:fe76:1e5, infinite/infinite
    multicast address ff02::1, 1 refs, not reportable
    multicast address ff02::1:ff76:1e5, 1 refs, last reporter
  link MTU 1500 (true link MTU 1500)
  current hop limit 128
  reachable time 35500ms (base 30000ms)
  retransmission interval 1000ms
  DAD transmits 1
```

- ipv6 adu #interface/address [lifetime VL[/PL]] [anycast] [unicast]

Este comando permite asignarle una dirección a una interface.

```
C:\>ipv6 adu 4/3ffe:8070:1011:1:201:2ff:fe76:1e5
C:\>ipv6 if 4
Interface 4 (site 1):
  cable reconnected
  uses Neighbor Discovery
  link-level address: 00-01-02-76-01-e5
   preferred address 3ffe:8070:1011:1:201:2ff:fe76:1e5,infinite/infinite
   preferred address fe80::201:2ff:fe76:1e5, infinite/infinite
   multicast address ff02::1, 1 refs, not reportable
   multicast address ff02::1:ff76:1e5, 2 refs, last reporter
  link MTU 1500 (true link MTU 1500)
```

```
  current hop limit 128
  reachable time 30000ms (base 30000ms)
  retransmission interval 1000ms
  DAD transmits 1
```

Como no se indica el tiempo de vida preferido, es infinito. Para indicar un valor de tiempo de vida es de la siguiente manera:

```
C:\>ipv6 adu 4/3ffe:8070:1011:1:201:2ff:fe76:1e5 lifetime 3000/2000
C:\>ipv6 if 4
Interface 4 (site 1):
  cable reconnected
  uses Neighbor Discovery
  link-level address: 00-01-02-76-01-e5
   preferred address 3ffe:8070:1011:1:201:2ff:fe76:1e5, 2996s/1996s
   preferred address fe80::201:2ff:fe76:1e5, infinite/infinite
   multicast address ff02::1, 1 refs, not reportable
   multicast address ff02::1:ff76:1e5, 2 refs, last reporter
  link MTU 1500 (true link MTU 1500)
  current hop limit 128
  reachable time 30000ms (base 30000ms)
  retransmission interval 1000ms
  DAD transmits 1
```

Para eliminar una dirección, se utiliza el mismo comando pero con el tiempo de vida igual a 0.

```
C:\>ipv6 adu 4/3ffe:8070:1011:1:201:2ff:fe76:1e5 lifetime 0
C:\>ipv6 if 4
Interface 4 (site 1):
  cable reconnected
  uses Neighbor Discovery
  link-level address: 00-01-02-76-01-e5
    preferred address fe80::201:2ff:fe76:1e5, infinite/infinite
    multicast address ff02::1, 1 refs, not reportable
    multicast address ff02::1:ff76:1e5, 1 refs, last reporter
  link MTU 1500 (true link MTU 1500)
  current hop limit 128
  reachable time 30000ms (base 30000ms)
  retransmission interval 1000ms
  DAD transmits 1
```

- Direcciones Duplicadas

En el siguiente ejemplo vemos que sucede en Windows cuando se quiere agregar una dirección que ya está asignada a otro host.

```
C:\>ipv6 adu 4/fe80::24f:49ff:fe03:5fba
C:\>ipv6 if 4
Interface 4 (site 1):
  cable reconnected
  uses Neighbor Discovery
  link-level address: 00-01-02-76-01-e5
    duplicate address fe80::24f:49ff:fe03:5fba, infinite/infinite
    preferred address fe80::201:2ff:fe76:1e5, infinite/infinite
    multicast address ff02::1, 1 refs, not reportable
    multicast address ff02::1:ff76:1e5, 1 refs, last reporter
    multicast address ff02::1:ff03:5fba, 1 refs, last reporter
  link MTU 1500 (true link MTU 1500)
  current hop limit 128
  reachable time 22500ms (base 30000ms)
  retransmission interval 1000ms
  DAD transmits 1
```

Como se puede ver, Windows no retorna ningún error cuando se agrega una dirección duplicada. Al observar el estado de la interface vemos, que la dirección, tiene la palabra *duplicate* indicando la anomalía.

A continuación se muestra como se instala el protocolo en las distintas versiones de Windows

- Windows NT 4.0

La implementación de IPv6 para Windows NT 4.0 se puede bajar del sitio de Microsoft, http://research.microsoft.com/msripv6.

Una vez que se baja el archivo binario, msripv6-bin-1.4.exe, se lo debe ejecutar. Este archivo crea un directorio en el C:\IPv6Kit. Para terminar de instalar IPv6, hay que ir a las propiedades del Entorno de Red, seleccionar en la parte de Protocolos, cliquear el botón Agregar y luego el botón Utilizar Disco. En la ventana que se abre, se debe ingresar al path del directorio IPv6Kit y aceptar. El protocolo se agrega en la parte de Protocolos como MSR IPv6 Protocol, y solamente puede ser configurado manualmente desde la línea de comandos.

Cuando se reinicia la máquina, IPv6 comienza a funcionar automáticamente. Si se quiere mantener la configuración agregada manualmente cada vez que se reinicia la máquina, las instrucciones se deben guardar en un scrip de comandos (archivo .cmd) que se ejecute cada vez que se inicia el sistema.

- Windows 2000

Para esta versión se debe bajar el archivo tpipv6-001205.exe del sitio de Microsoft, http://msdn.microsoft.com/downloads/sdks/platform/tpipv6.asp.

Para realizar la instalación de IPv6 se requiere, como mínimo, una versión de Windows 2000 con el Service Pack 1 instalado.

En la página http://msdn.microsoft.com/downloads/sdks/platform/tpipv6/faq.asp se pueden encontrar las instrucciones para realizar la instalación según el Service Pack instalado.

- Windows XP

En Windows XP no es necesario tener ningún Service Pack instalado ni bajar un archivo del sitio de Microsoft porque IPv6 viene integrado al sistema operativo. Las instrucciones para instalar IPv6 se pueden encontrar en http://www.microsoft.com/windowsxp/pro/techinfo/administration/ipv6/default.asp. No se necesita tener ningún Service Pack instalado, pero si se tiene el Service Pack 1, las formas de realizar la instalación son distintas.

Nota: en la nueva versión de Windows, Windows .Net Server 2003 no se utiliza más el comando ipv6.exe para configurar IPv6, si no que es con el comando netsh

## A.2- Solaris 8.0

Al igual que en IPv4, que utiliza el archivo hostname.*interface,* IPv6 utiliza el archivo hostname6.*interface,* donde interface indica la interface el la cual se quiere habilitar el protocolo. Estos archivos se crean en el directorio /etc. Hay que reiniciar la máquina para que las modificaciones tengan efecto.

Los archivos de inicialización de la interface no contienen datos al momento de su creación, y puede que siempre estén en este estado si las direcciones se obtienen dinámicamente. Si se quiere agregar una dirección manualmente y que no se borren la próxima vez que se reinicie la máquina se debe agregar la siguiente instrucción en este archivo (una instrucción por cada dirección que se quiera configurar en la interface):

addif *dirección IPv6/prefijo* up

Entre los comandos más utilizados para configurar IPv6 en Solaris tenemos los siguientes:

- ifconfig -a

Nos permite ver las interfaces y sus direcciones (y se existe algún túnel). Además muestra información asociada a la interface, como por ejemplo: si es una dirección IPv4 o IPv6, si la interface está habilitada (UP), el MTU, etc.

```
bash-2.03# ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
hme0:flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
        inet 163.10.20.13 netmask ffffffc0 broadcast 163.10.20.63
        ether 8:0:20:8e:f4:27
lo0: flags=2000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv6> mtu 8252 index 1
        inet6 ::1/128
hme0: flags=2000841<UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
        ether 8:0:20:8e:f4:27
        inet6 fe80::a00:20ff:fe8e:f427/10
hme0:1:flags=2080841<UP,RUNNING,MULTICAST,ADDRCONF,IPv6>mtu 1500 index 2
        inet6 3ffe:8070:1011:1:a00:20ff:fe8e:f427/64
```

ADDRCONF: indica que la dirección se obtuvo por autoconfiguración.

Nota: con ifconfig -a4 vemos solamente las direcciones de IPv4, idem con ifconfig -a6.

- ifconfig *interface* inet6 addif dirección IPv6/longitud prefijo *up*

Con este comando agregamos una dirección IPv6 a una interface. Para esto se debe indicar la interface a la que se quiere agregar la dirección, la dirección IPv6 y la longitud del prefijo. Con la opción *up*, se indica que la dirección está habilitada.

En el siguiente ejemplo vemos que al agregar una dirección a la interface hme0, el sistema crea una interface lógica hme0:2

```
bash-2.03# ifconfig hme0 inet6 addif 3ffe:8070:1011:1::23/64 up
Created new logical interface hme0:2
bash-2.03# ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4>mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
hme0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4>mtu 1500 index 2
        inet 163.10.20.13 netmask ffffffc0 broadcast 163.10.20.63
        ether 8:0:20:8e:f4:27
lo0: flags=2000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv6>mtu 8252 index 1
        inet6 ::1/128
hme0: flags=2000841<UP,RUNNING,MULTICAST,IPv6>mtu 1500 index 2
        ether 8:0:20:8e:f4:27
        inet6 fe80::a00:20ff:fe8e:f427/10
hme0:1: flag=2080841<UP,RUNNING,MULTICAST,ADDRCONF,IPv6>mtu 1500 index 2
        inet6 3ffe:8070:1011:1:a00:20ff:fe8e:f427/64
hme0:2: flags=2000841<UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
              inet6 3ffe:8070:1011:1::23/64
```

Si observamos la interface lógica hme0:2 podemos ver que no tiene la palabra clave ADDRCONF, que si existe en la interface lógica hme0:1. Esto es porque ésta dirección fue obtenida por el método de autoconfiguración de direcciones y la segunda no.

- ifconfig *interface lógica* inet6 down

Mediante éste comando deshabilitamos una dirección. Observando la interface lógica hme0:2, vemos que no tiene la palabra clave UP que indica que está habilitada.

```
bash-2.03# ifconfig hme0:2 inet6 down
bash-2.03# ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4>mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
hme0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4>mtu 1500 index 2
        inet 163.10.20.13 netmask ffffffc0 broadcast 163.10.20.63
        ether 8:0:20:8e:f4:27
lo0: flags=2000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv6>mtu 8252 index 1
        inet6 ::1/128
hme0: flags=2000841<UP,RUNNING,MULTICAST,IPv6>mtu 1500 index 2
        ether 8:0:20:8e:f4:27
        inet6 fe80::a00:20ff:fe8e:f427/10
hme0:1: flag=2080841<UP,RUNNING,MULTICAST,ADDRCONF,IPv6>mtu 1500 index 2
        inet6 3ffe:8070:1011:1:a00:20ff:fe8e:f427/64
hme0:2: flags=2000840<RUNNING,MULTICAST,IPv6> mtu 1500 index 2
        inet6 3ffe:8070:1011:1::23/64
```

- ifconfig *interface* inet6 removeif *dirección IPv6*

Para eliminar una dirección utilizamos el ifconfig con estas opciones. Vemos que la interface lógica hme0:2 no existe más.

```
bash-2.03# ifconfig hme0 inet6 removeif 3ffe:8070:1011:1::23
bash-2.03# ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
hme0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4>mtu 1500 index 2
        inet 163.10.20.13 netmask ffffffc0 broadcast 163.10.20.63
        ether 8:0:20:8e:f4:27
lo0: flags=2000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv6>mtu 8252 index 1
        inet6 ::1/128
hme0: flags=2000841<UP,RUNNING,MULTICAST,IPv6>mtu 1500 index 2
        ether 8:0:20:8e:f4:27
        inet6 fe80::a00:20ff:fe8e:f427/10
```

```
hme0:1: flag=2080841<UP,RUNNING,MULTICAST,ADDRCONF,IPv6>mtu 1500 index 2
        inet6 3ffe:8070:1011:1:a00:20ff:fe8e:f427/64
```

- ping

Una de las herramientas que podemos utilizar es el ping. Con la opción -s, enviamos varios echo request al destino hasta que lo detenemos con ^C.

```
bash-2.03# ping fe80::200:cff:fe09:7341
fe80::200:cff:fe09:7341 is alive
bash-2.03# ping -s fe80::200:cff:fe09:7341
PING fe80::200:cff:fe09:7341: 56 data bytes
64 bytes from fe80::200:cff:fe09:7341: icmp_seq=0. time=2. ms
64 bytes from fe80::200:cff:fe09:7341: icmp_seq=1. time=2. ms
64 bytes from fe80::200:cff:fe09:7341: icmp_seq=2. time=2. ms
64 bytes from fe80::200:cff:fe09:7341: icmp_seq=3. time=2. ms
64 bytes from fe80::200:cff:fe09:7341: icmp_seq=4. time=2. ms
^C
----fe80::200:cff:fe09:7341 PING Statistics----
5 packets transmitted, 5 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 2/2/2
```

- Direcciones duplicadas

El siguiente es un ejemplo que permite mostrar el comportamiento de Solaris cuando se quiere agregar una dirección duplicada. (La dirección que se quiere agregar es la dirección de link-local del router).

Al querer agregar una dirección duplicada, se ejecuta la Detección de Direcciones Duplicadas. Como intentamos agregar una dirección que ya está asignada, el sistema nos muestra un error:

```
bash-2.03# ifconfig hme0 inet6 addif fe80::200:cff:fe09:7341/64 up
Created new logical interface hme0:2
ifconfig: Duplicate address detected on link hme0 for address
fe80::200:cff:fe09:7341. Code 1
```

Si observamios la configuración de las direcciones vemos que se creó la interface lógica pero está deshabilitada.

```
bash-2.03# ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4>mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
hme0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4>mtu 1500 index 2
        inet 163.10.20.13 netmask ffffffc0 broadcast 163.10.20.63
        ether 8:0:20:8e:f4:27
lo0: flags=2000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv6> mtu 8252 index 1
        inet6 ::1/128
hme0: flags=2000841<UP,RUNNING,MULTICAST,IPv6> mtu 1500 index 2
        ether 8:0:20:8e:f4:27
        inet6 fe80::a00:20ff:fe8e:f427/10
hme0:1: flag=2080841<UP,RUNNING,MULTICAST,ADDRCONF,IPv6>mtu 1500 index 2
        inet6 3ffe:8260:10c:8000:a00:20ff:fe8e:f427/64
hme0:2: flag=2080841<UP,RUNNING,MULTICAST,ADDRCONF,IPv6>mtu 1500 index 2
        inet6 3ffe:8070:1011:1:a00:20ff:fe8e:f427/64
hme0:3: flags=2000840<RUNNING,MULTICAST,IPv6> mtu 1500 index 2
        inet6 fe80::200:cff:fe09:7341/64
```

Al querer habilitar la interface lógica, se ejecuta nuevamente la Detección de Direcciones Duplicada y muestra el mismo error.

```
bash-2.03# ifconfig hme0:2 inet6 up
ifconfig: Duplicate address detected on link hme0 for address
fe80::200:cff:fe09:7341. Code 1
```

## A.3.- Linux (Red Hat 8.0)

Las distribuciones modernas de Linux contienen el kernel con soporte para IPv6; la capacidad de IPv6 es compilada generalmente como un módulo, pero suele suceder que éste no se cargue automáticamente en el arranque del SO.

Para ver si está cargado debemos mirar si la siguiente entrada existe en el /proc-file-system:

/proc/net/if_net6

Si está entrada no existe, el módulo se puede cargar de dos manera:

1. - Manualmente mediante el siguiente comando:

# modprobe ipv6

Nota: se debe tener en cuenta que la descarga del módulo no es soportada y puede resultar, bajo ciertas circunstancias, en un kernel crash.

2. - Automáticamente en el inicio del sistema agregando la siguiente lineal en le archivo cargador de módulos del kernel (/etc/modules.conf o /etc/conf.modules):

alias net-pf-10 ipv6

Una vez cargado el módulo, el protocolo comienza a funcionar en forma inmediata, es decir, se autoconfiguran todas las direcciones y solicita un anuncio de router. Red Hat permite realizar todas sus operaciones (agregar, eliminar una dirección, ver el estado de las interfaces, etc.) mediante dos comandos diferentes:

- ifconfig
- ip

- Agregar una dirección

Con el comando ifconfig:

```
[root@paturuzu matias]# ifconfig eth0 inet6 add 3ffe:8070:1011:1::80/64
[root@paturuzu matias]# ip -6 addr show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    inet6 3ffe:8070:1011:1::80/64 scope global
    inet6 3ffe:8070:1011:1:260:97ff:fe59:e3d/64 scope global dynamic
       valid_lft 6991sec preferred_lft 5991sec
    inet6 fe80::260:97ff:fe59:e3d/10 scope link
```

Con el comando ip:

```
[root@paturuzu matias]# ip -6 addr add 3ffe:8070:1011:1::70/64 dev eth0
[root@paturuzu matias]# ip -6 addr show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    inet6 3ffe:8070:1011:1::80/64 scope global
    inet6 3ffe:8070:1011:1:260:97ff:fe59:e3d/64 scope global dynamic
       valid_lft 6991sec preferred_lft 5991sec
    inet6 fe80::260:97ff:fe59:e3d/10 scope link
    inet6 3ffe:8070:1011:1::70/64 scope global
```

- Eliminar una dirección

Con el comando ifconfig:

```
[root@paturuzu matias]# ifconfig eth0 inet6 del 3ffe:8070:1011:1::80/64
[root@paturuzu matias]# ip -6 addr show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    inet6 3ffe:8070:1011:1:260:97ff:fe59:e3d/64 scope global dynamic
       valid_lft 6997sec preferred_lft 5997sec
    inet6 fe80::260:97ff:fe59:e3d/10 scope link
    inet6 3ffe:8070:1011:1::70/64 scope global
```

Con el comando ip:

```
[root@paturuzu matias]# ip -6 addr del 3ffe:8070:1011:1::70/64 dev eth0
[root@paturuzu matias]# ip -6 addr show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    inet6 3ffe:8070:1011:1:260:97ff:fe59:e3d/64 scope global dynamic
       valid_lft 6991sec preferred_lft 5991sec
    inet6 fe80::260:97ff:fe59:e3d/10 scope link
```

- Direcciones Duplicadas

Al agregar una dirección duplicada (la dirección que se intenta agregar es la dirección IPv6 del router Cisco), ésta queda en estado tentativo:

```
[root@paturuzu matias]# ip -6 addr add 3ffe:8070:1011:1::1/64 dev eth0
[root@paturuzu matias]# ip -6 addr show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    inet6 3ffe:8070:1011:1:260:97ff:fe59:e3d/64 scope global dynamic
       valid_lft 6978sec preferred_lft 5978sec
    inet6 3ffe:8070:1011:1::1/64 scope global tentative
    inet6 fe80::260:97ff:fe59:e3d/10 scope link
```

- Ping

El comando ping de IPv4 es reemplazado por ping6.

```
[root@paturuzu /]# ping6 FE80::200:CFF:FE09:7341
connect: Invalid argument
[root@paturuzu /]# ping6 -I eth0 FE80::200:CFF:FE09:7341
PING FE80::200:CFF:FE09:7341(fe80::200:cff:fe09:7341) from
fe80::260:97ff:fe59:e3d eth0: 56 data bytes
64 bytes from fe80::200:cff:fe09:7341: icmp_seq=1 ttl=50 time=2.60 ms
64 bytes from fe80::200:cff:fe09:7341: icmp_seq=2 ttl=50 time=2.17 ms
64 bytes from fe80::200:cff:fe09:7341: icmp_seq=3 ttl=50 time=2.44 ms

--- FE80::200:CFF:FE09:7341 ping statistics ---
3 packets transmitted, 3 received, 0% loss, time 4008ms
rtt min/avg/max/mdev = 2.173/2.366/2.605/0.171 ms
```

## A.4.- FreeBSD 4.8

FreeBSD, como todos los sistemas BSD (NetBSD, OpenBSD), incluye la implementación del proyecto KAME (éste es un proyecto desarrollado por 6 empresas de Japón para proveer a estos SO con IPv6 e IPSec) en forma predeterminada desde la versión 4.0. Si se tiene una versión anterior, en la dirección www.kame.net se pueden encontrar implementaciones para esos SO.

La mayoría de la información de configuración se encuentra en el archivo /etc/rc.conf, en donde se debe modificar la siguiente línea para habilitar IPv6:

ipv6_enable="YES" --es NO por defecto

Se puede indicar en que interfaces se desea habilitar IPv6.

El comando ifconfig permite ver la configuración de las interfaces:

```
freebsd# ifconfig
xl0: flags=8a43<UP,BROADCAST,RUNNING,ALLMULTI,SIMPLEX,MULTICAST>mtu 1500
        inet 163.10.20.35 netmask 0xfffffc0 broadcast 163.10.20.63
        inet6 fe80::260:97ff:fe1d:7a70%xl0 prefixlen 64 scopeid 0x1
        inet6 3ffe:8070:1011:1::9 prefixlen 64
        ether 00:60:97:1d:7a:70
        media: Ethernet 10baseT/UTP (10baseT/UTP <half-duplex>)
lp0: flags=8851<UP,POINTOPOINT,RUNNING,SIMPLEX,MULTICAST> mtu 1500
faith0: flags=8243<UP,BROADCAST,RUNNING,ALLMULTI,MULTICAST> mtu 1500
        inet6 fe80::260:97ff:fe1d:7a70%faith0 prefixlen 64 scopeid 0x4
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x5
        inet 127.0.0.1 netmask 0xff000000
ppp0: flags=8010<POINTOPOINT,MULTICAST> mtu 1500
sl0: flags=c010<POINTOPOINT,LINK2,MULTICAST> mtu 552
```

- Agregar una dirección

El comando para agregar una dirección es ifconfig. Se debe indicar la interface a la cual se le agrega la dirección, que la dirección es del tipo IPv6 y se debe indicar el prefijo de la dirección.

```
freebsd# ifconfig xl0 inet6 add 3ffe:8070:1011:1::10/64
freebsd# ifconfig
xl0: flags=8a43<UP,BROADCAST,RUNNING,ALLMULTI,SIMPLEX,MULTICAST> mtu 1500
        inet 163.10.20.35 netmask 0xfffffc0 broadcast 163.10.20.63
        inet6 fe80::260:97ff:fe1d:7a70%xl0 prefixlen 64 scopeid 0x1
        inet6 3ffe:8070:1011:1::9 prefixlen 64
        inet6 3ffe:8070:1011:1::10 prefixlen 64
        ether 00:60:97:1d:7a:70
        media: Ethernet 10baseT/UTP (10baseT/UTP <half-duplex>)
lp0: flags=8851<UP,POINTOPOINT,RUNNING,SIMPLEX,MULTICAST> mtu 1500
.....
```

- Eliminar una dirección

El comando delete se utiliza para eliminar una dirección asociada a una interface. También se debe especificar la interface y el tipo de dirección, pero no se debe especificar el prefijo.

```
freebsd# ifconfig xl0 inet6 delete 3ffe:8070:1011:1::10
freebsd# ifconfig
xl0: flags=8a43<UP,BROADCAST,RUNNING,ALLMULTI,SIMPLEX,MULTICAST> mtu 1500
        inet 163.10.20.35 netmask 0xffffffc0 broadcast 163.10.20.63
        inet6 fe80::260:97ff:fe1d:7a70%xl0 prefixlen 64 scopeid 0x1
        inet6 3ffe:8070:1011:1::9 prefixlen 64
        ether 00:60:97:1d:7a:70
        media: Ethernet 10baseT/UTP (10baseT/UTP <half-duplex>)
lp0: flags=8851<UP,POINTOPOINT,RUNNING,SIMPLEX,MULTICAST> mtu 1500
```

- Agregar una dirección duplicada

Al agregar una dirección duplicada, el sistema no indica si la dirección es duplicada o no. Para observar esto, hay que ejecutar el comando ifconfig:

```
freebsd# ifconfig xl0 inet6 add 3ffe:8070:1011:1::1/64
freebsd# ifconfig
xl0:flags=8a43<UP,BROADCAST,RUNNING,ALLMULTI,SIMPLEX,MULTICAST>mtu 1500
        inet 163.10.20.35 netmask 0xffffffc0 broadcast 163.10.20.63
        inet6 fe80::260:97ff:fe1d:7a70%xl0 prefixlen 64 scopeid 0x1
        inet6 3ffe:8070:1011:1::9 prefixlen 64
        inet6 3ffe:8070:1011:1::1 prefixlen 64 duplicated
        ether 00:60:97:1d:7a:70
        media: Ethernet 10baseT/UTP (10baseT/UTP <half-duplex>)
lp0: flags=8851<UP,POINTOPOINT,RUNNING,SIMPLEX,MULTICAST> mtu 1500
.....
```

- Ping

El comando que se utiliza es el ping6. Si el ping se envía a una dirección de link-local se debe indicar la interface por donde se debe enviarlo. En caso que no se haga se generará un error.

```
freebsd# ping6 fe80::200:cff:fe09:7341
PING6(56=40+8+8 bytes) fe80::1%lo0 --> fe80::200:cff:fe09:7341
ping6: sendmsg: No route to host
ping6: wrote fe80::200:cff:fe09:7341 16 chars, ret=-1
ping6: sendmsg: No route to host
ping6: wrote fe80::200:cff:fe09:7341 16 chars, ret=-1
^C
--- fe80::200:cff:fe09:7341 ping6 statistics ---
2 packets transmitted, 0 packets received, 100% packet loss

freebsd# ping6 -I xl0 fe80::200:cff:fe09:7341
PING6(56=40+8+8 bytes) fe80::260:97ff:fe1d:7a70%xl0-> fe80::200:cff:fe09:7341
16 bytes from fe80::200:cff:fe09:7341%xl0,icmp_seq=0 hlim=50 time=2.805 ms
16 bytes from fe80::200:cff:fe09:7341%xl0,icmp_seq=1 hlim=50 time=2.55 ms
16 bytes from fe80::200:cff:fe09:7341%xl0,icmp_seq=2 hlim=50 time=2.347 ms
^C
--- fe80::200:cff:fe09:7341 ping6 statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/std-dev = 2.347/2.567/2.805/0.187 ms
freebsd#
```

# Apéndice B

## Método de obtención de un Identificador de Interface de una MAC Address

Para obtener el identificador de interface, de una interface Ethernet, nos basamos en las direcciones MAC de 48 bits. El proceso es simple y consiste en insertar los bytes FFFE (hexadecimal) entre los bytes 3 y 4 de la dirección MAC de 48 bits.

El identificador que se obtiene se conoce como EUI-64 (Extended Unique Identifier de 64 bits), de los cuales los primeros 24 bits indican la compañía, y son asignados por la IEEE, y los restantes 40 bits son asignados internamente por dichas compañías.

Para obtener el identificador de interfaz se debe complementar el bit U/L (Universal/Local). Este bit se utiliza para indicar si la dirección es local o universal, es decir única, y es el bit 7 del primer byte del identificador. El bit queda igual a 1. La finalidad de complementar el bit es, puramente, con fines administrativos. Si un administrador ingresa direcciones IPv6 manualmente, le es más simple ingresarlas, por ejemplo: fec0::1, fec0::2, etc. y no tener que ingresar un uno en el bit U/L.

El siguiente gráfico muestra como se forman las direcciones EUI-64:

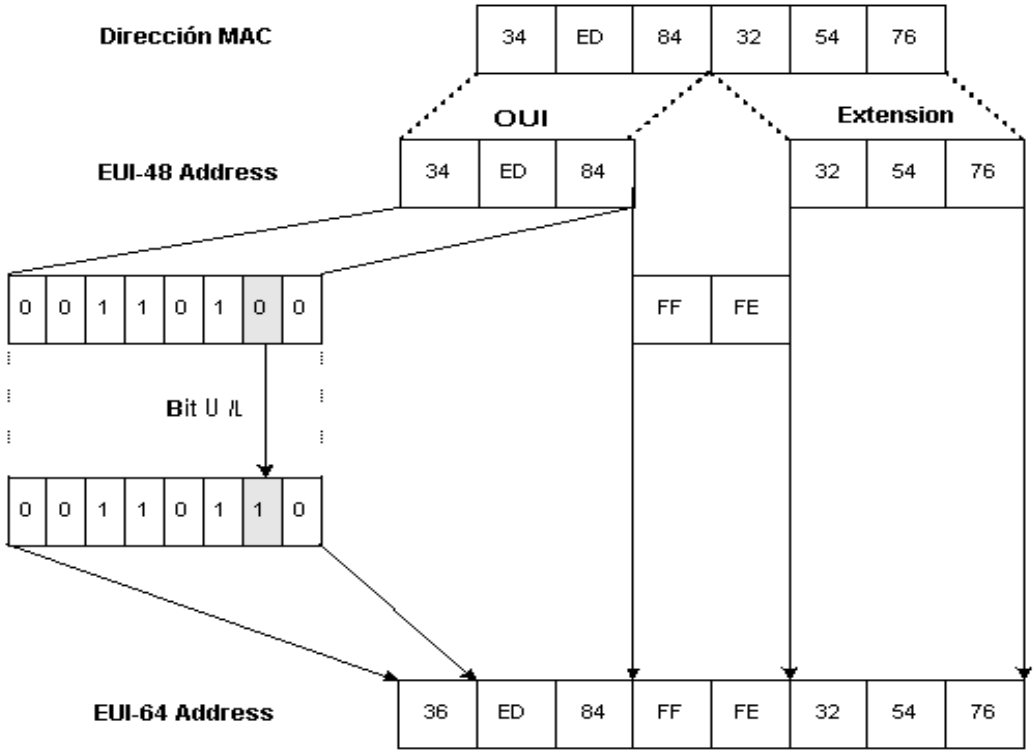


Figura A.1 - Armado dirección EUI-64

# Apéndice C
# Protolocos de Ruteo Multicast

## C.1 - Introducción

En multicast, los paquetes son enviados a un conjunto de receptores, que pueden estar ubicados en distintas subredes. La dirección destino del paquete no identifica un destinatario, o una red, conocida y determinada, si no, que puede referirse a varios receptores diferentes. Teniendo en cuenta estas caraterísticas, es obvio, que ninguno de los protocolos de ruteo unicast pueden cumplir eficientemente con este trabajo. De esto se deduce la necesidad de protocolos de ruteo, específicos, para rutear tráfico multicast.

Los protocolos de ruteo multicast están basados en los tres siguientes fundamentos:

- Multicast Distributions Trees
- Multicast Forwarding
- Tipos de protocolos multicast

## C.2 - Multicast Distributions Trees

Define el camino descendiente por el cual el tráfico fluye desde el origen hacia todos los receptores. Existen dos tipo de árboles de distribución multicast:

- Shortest Path Tree (or Source Distribution Tree)
- Shared Tree

Ambos tipos de árboles están libres de loops. Los mensajes son replicados, unicamente, donde el árbol tiene ramificaciones.

Los miembros de los grupos multicast pueden unirse, o dejar, un grupo en cualquier momento, con lo cual los árboles de distribución se actualizan dinámicamente. Cuando todos los receptores, en una rama en particular, dejan de solicitar el tráfico para un grupo multicast, los routers podan esa rama del árbol de distribución, y dejan de enviar tráfico en ese link. Si un nodo en esa rama se vuelve activo, y solicita tráfico multicast nuevamente, el router modificará, automáticamente, el árbol de distribución y comenzará a reenviarle los paquetes nuevamente.

### C.2.1 - Shortest Path Tree

La forma más simple de un árbol de distribución multicast es un árbol con su raíz en la fuente y ramificaciones, que forman un spanning-tree, que llega hasta todos los receptores. Como el árbol generado utiliza los caminos más cortos hacia los destinos, se lo llama Shortest Path Tree.

Los paquetes son reenviados según la dirección origen del paquete y la dirección de grupo multicast, a la que los paquetes son enviados. Esto se conoce por la notación (S, G):

S: dirección IP del origen
G: dirección multicast del grupo

Esta notación implica que existe un SPT, separado, para cada fuente individual enviando a cada grupo.

Estos árboles tienen la ventaja de crear un camino óptimo desde el origen hacia todos los receptores. Esto garantiza una mínima latencia en la red para reenviar tráfico multicast. La desventaja es que los routers deben mantener información del camino para cada fuente. En una red con muchos grupos multicast, la tabla de ruteo multicast se puede hacer demasiado grande.

### C.2.2 - Shared Tree

A diferencia de los source trees que tienen su raiz en la fuente, los shared trees usan una sola raiz común ubicada en algún lugar de la red. En estos casos, la fuente debe enviar su tráfico a la raiz del árbol, quien a su vez, se encarga de reenviarlo por los caminos descendientes del árbol hasta alcanzar todos los receptores.

El tráfico multicast es reenviado teniendo en cuenta unicamente el grupo multicast al cual el paquete está dirigido, sin importar la dirección origen. Por esta razón, la notación que se utiliza es (*, G), donde * significa cualquier fuente.

Como ventaja, requieren un mínimo de información de estado en cada router. La desventaja es, que en ciertas circunstancias, el camino entre el origen y el destino no siempre es óptimo. Se debe considerar, cuidadosamente, la ubicación del nodo que funcionará como raiz.

## C.3 - Multicast Forwarding

En el ruteo unicast, el tráfico es ruteado, a través de la red, a lo largo de un solo camino hasta el host destino. Al router no le interesa la dirección origen del paquete, solo le preocupa la dirección destino de éste y como reenviarlo por el camino correcto hacia él. El router escanea su tabla de ruteo unicast, y luego envía el paquete por la interface correcta en la dirección del destino.

En el ruteo multicast, un nodo envía tráfico a un grupo arbitrario de hosts que están representadas por una dirección de grupo multicast. El router debe determinar que dirección es upstream (hacia la fuente) y cuales son downstream (hacia los receptores) para evitar los loops. Si existen varios caminos distintos, descendientes, hacia los receptores, el router debe replicar los paquetes en cada uno de esos caminos.

Al ruteo unicast le interesa conocer hacia donde va el paquete, en cambio, al ruteo multicast le interesa saber de donde viene. Este último es conocido como Reverse Path Forwarding(RPF).

RPF es un concepto fundamental en el ruteo multicast que habilita a los routers a reenviar tráfico multicast por el árbol de distribución. RPF utiliza la tabla de ruteo unicast para determinar que vecinos son upstream y cuales downstream. Un router solo reenviará datagramas multicast si la interface por la que los recibe se encuentra, en el árbol de distribución, en el camino hacia la fuente. Un router tendrá una sola interface entrante designada por la que puede recibir tráfico multicast de un grupo determinado, pero lo puede recibir por más de una interface simultaneamente. Ejecutando el chequeo RFP puede evitar el reenvío duplicado.

El router cheque la dirección IP origen del paquete contra su tabla de ruteo unicast (RPF check), y se asegura que el paquete ingresó por la interface correcta. Si es así, el chequeo RPF tuvo éxito y el paquete será reenviado por cada una de las interfaces de la lista de interfaces salientes de la tabla de ruteo multicast; en caso contrario, el paquete será, silenciosamente, descartado. Esto garantiza que los árboles de distribución estén libres de loops.

## C.4 - Tipos de protocolos multicast

Los protocolos de ruteo multicast están agrupados en dos grandes categorías:

- Dense Mode
- Sparse Mode

### C.4.1 - Dense Mode

Este modelo asume que existen miembros en todos los puntos de la red, de aquí el concepto de una distribución *densa* de los receptores. Es un modelo de fuerza bruta, pero en los caso donde hay al menos un receptor en cada subred, puede ser un mecanismo eficiente.

Inicialmente, el tráfico multicast es enviado por toda la red (flooding). Los routers que no tienen ningún receptor en camino descendiente podan (prune) las ramas de árbol de distribución, para no reenvíar más tráfico innecesario. Esta poda se vence a los tres minutos, y la rama es nuevamente inundada con tráfico multicast.

Mediante el *flooding* y *prune* los routers obtienen su información de estado, recibiendo streams de datos. Estos streams contienen información sobre la dirección origen del grupo y del grupo, con la cual los routers pueden construir sus propias tablas de ruteo multicast.

El modo denso es más aplicable en redes con buen ancho de banda; y solo permite source tree (no trabaja con shared tree).

Los siguientes protocolos son de modo denso:

- DVMRP (Distance Vector Multicast Routing Protocol)
- MOSPF (Multicast Open Short Path First)
- PIM-DM (Protocol Independent Multicast – Dense Mode)

### C.4.2 - Sparse Mode

Este modelo, a diferencia del anterior, asume que ningún receptor está interesado en recibir el tráfico multicast, a menos que explicitamente pida por él.

Utiliza un shared tree para distribuir la información, es decir que existe un host, en el árbol, conocido como root o pivote (en PIM-SM se lo llama Rendezvous Point). Este host debe ser seleccionado y configurado manualmente.

Las fuentes se registran con la raiz, y le envían el tráfico multicast a éste, quien a su vez lo reenvía por el árbol de distribución.

Los receptores deben indicarle a nodo raiz su deseo de recibir el tráfico multicast.

Los siguientes protocolos son sparse-mode:

- PIM-SM (Protocolo Independent Multicast – Sparse Mode)
- CBT (Core Based Trees)

**Importante:**
La diferencia entre *dense mode* y *sparse-mode* no se debe a la cantidad de integrantes del grupo multicast, si no, a la forma en que éstos están distribuidos en la red

# Bibliografía

Tutorial de IPv6 – ConsultIntel- Palet, Jordi
Excelente tutorial para comenzar a entender porque se necesita IPv6 y sus características.

Microsoft Windows 2000 Server – Introduction to IPv6
Joseph Davies – February 2002
Una explicación muy completa IPv6 y de todos los protocolos que hacen a su funcionamiento (Neigbhor Discovery, Multicast Listener Discovery, etc.)

Understanding IPv6 by David Morton
www.itp-journals.com/nasample/c0655.pdf
Basado en la primeras RFCs que definieron el protocolo pero no está desactualizado. Es simple y no demasiado técnico

IPv6 for Cisco IOS Software File 1 Of 3: Overview
Excelente resumen sobre la funcionalidad de IPv6.

IPv6 for Cisco IOS Software File 1 Of 2: Configuring
Detalla como se configura IPv6 en un router Cisco y otros protocolos fundamentales para su funcionamiento.

IPv6 for Cisco IOS Software File 1 Of 3: IPv6 Command Reference
Describe todos los comandos disponibles para configurar IPv6.

Cisco - The ABCs of IPv6
Un muy buen documento explicando IPv6 de manera general.

Neighbor Discovery and Stateless Autoconfiguration in IPv6 By Thomas Narten
www.cs.ucsb.edu/~almeroth/classes/ F99.595N/autoconfig.pdf
Un paper excelente para aprender como trabajan estos dos protocolos. Recomiendo su lectura.

Configuring MSR IPv6
Explica como se configura IPv6 en Windows NT 4.0

Solaris System Administrator Guide, Volumen 3
Muy bueno para aprender configurar Solaris con IPv6 (además tiene una explicación bastante amplia del protocolo).

RFC 791 - Internet Protocol - Septiembre 1981
Esta RFC explica el protocolo IP versión 4.

RFC 1256 - ICMP Router Discovery Messages - Septiembre 1991
Especifica una extensión al ICMP que permite a los hosts, en una red multicast o broadcast, descubrir automáticamente las direcciones IP de los routers vecinos.

RFC 1191 - Path MTU Discovery - Noviembre 1990.
Describe una técnica para descubrir dinámicamente el MTU de un camino arbitrario sobre Internet.

RFC 1518 - Classless Inter-Domain Routing - Septiembre 1993
Especifica una arquitectura y plan para la alocación de direcciones IPv4 en Internet.

RFC 2080 - RIPng para IPv6 - Enero 1997
Especifica los cambios mínimos necesarios al protocolo RIP para que éste pueda trabajar con IPv6.

RFC 2113 - IP Router Alert Option - Febrero 1997
Describe una nueva opción en IPv4 que indica a los routers de tránsito que examinen más detenidamente el contenido de un paquete.

RFC 2185 - Routing Aspects Of IPv6 Transition - Septiembre 1997
Este documento da una introducción a los aspectos a tener en cuenta de ruteo en la transición de IPv4 a IPv6.

RFC 2236 - Internet Group Management Protocol, Version 2 - Noviembre 1997
Describe el proceso que permite a los nodos comunicar su membresía a un grupo multicast.

RFC 2362 - Protocol Independent Multicast-Sparse Mode(PIM-SM) - Junio 1998
Explica el protocolo PIM, para IPv6, que permite rutear eficientemente a grupos multicast distribuidos en redes WAN, o inter-dominios.

RFC 2373 - IP Version 6 Addressing Architecture - Julio 1998
Describe la arquitectura de las direcciones IPv6. Incluye: el modelo de direcciones, su representación textual, definición de direcciones unicast, anycast, etc.

RFC 2374 - An IPv6 Aggregatable Global Unicast Address Format - Julio 1998
Describe un formato de dirección IPv6, para usar en Internet, diseñado para facilitar un ruteo escalable.

RFC 2460 - Internet Protocol, Version 6 Specification - Diciembre 1998
Este documento especifica la versión 6 del protocolo Internet.

RFC 2461 - Neighbor Discovery for IP versión 6 - Diciembre 1998.
Esta RFC explica cómo interactuan los nodos vecinos en un link. Cubre varios protocolos de IPv4, y es una de los aspectos más importantes de IPv6. No se puede conocer el funcionamiento de IPv6 si no se la lee.

RFC 2462 - IPv6 Stateless Address Autoconfiguration - Diciembre 1998
Explica los pasos que un nodo debe seguir para autoconfigurar sus interfaces. Idem a la RFC anterior.

RFC 2463 - Internet Control Message Protocol (ICMPv6) for IP Version 6 Specification - Diciembre 1998.
Describe un conjunto de mensajes ICMP para usar con la versión 6 del protocolo IP.

RFC 2464 - Transmission of IPv6 Packets over Ethernet Networks - Diciembre 1998
Describe el formato de los frames para transmitir paquetes IPv6, y el método para formar las direcciones de link-local.

RFC 2473 - Generic Packet Tunneling in IPv6 Specification - Diciembre 1998
Este documento define el modelo y los mecanismos genéricos para la encapsulación de paquetes IPv6 o IPv4 en túneles IPv6.

RFC 2529 - Transmission of IPv6 over IPv4 Domains without Explicit Tunnels - Marzo 1999
Especifica un método que permite que nodos IPv6 aislados puedan funcionar utilizando un dominio IPv4, sin necesidad de establecer un túnel manual.

RFC 2710 - Multicast Listener Discovery(MLD) for IPv6 - Octubre 1999
Explica el método utilizado para los routers IPv6 para descubrir si un grupo multicast tiene participantes en un link.

RFC 2711 - IPv6 Router Alert Option - Octubre 1999
Idem a la RFC 2113, pero para IPv6.

RFC 2766 - Network Address Translation - Protocol Translation (NAT-PT) - Febrero 2000
Describe un método de transición de Ipv4 a IPv6, en el cual un nodo IPv6 se puede comunicar con un nodo IPv4 (y viceversa).

RFC 2772 - 6Bone Backbone Routing Guidelines - Febrero 2000
Este documento provee una guía a los operadores de los equipos de ruteo del 6bone para usarlo como referencia en el desarrollo de un sistema de ruteo estable y escalable.

RFC 2874 - DNS Extensions to Support IPv6 Address Aggregation and Renumbering - Julio 2000
Describe las modificaciones necesarias para que el DNS soporte el protocolo IPv6

RFC 2858 - Multiprotocol Extensions for BGP-4 - Junio 2000
Define extensiones el BGP4 para que pueda transportar información de ruteo para varios protocolos de red, entre ellos, IPv6.

RFC 2893 - Transition Mechanisms for IPv6 Hosts and Routers - Agosto 2002
Este documento describe los mecanismos de compatibilidad que puede utilizarse para que trabajen conjuntamente los IPv4 e IPv6.

RFC 3056 - Connection of IPv6 Domains via IPv4 Clouds - Abril 2003
Especifica un mecanismo interno para que sitios IPv6 se puedan comunicar entre ellos a través de una infraestructura de ruteo IPv4 sin establecer túneles explícitos.

RFC 3177 - IAB/IESG Recommendations on IPv6 Address Allocations to Sites - Septiembre 2001
Este documento es una recomendación, a las organizaciones correspondientes, de cómo se debe realizar la asignación de bloques de direcciones IPv6 a los sitios.

RFC 3513 - Internet Protocol Version 6 (IPv6) Addressing Architecture - Abril 2003
Esta RFC es una actualización de la RFC 2373.

Cisco - Implementing RIP for IPv6
Detalla los pasos necesarios para instalar y configurar el protocolo de ruteo RIP para IPv6 en un router Cisco.

Cisco - Chapter 2 - IPv6 Addressing
Explica en forma detallada como están compuestas las direcciones en IPv6. Es un archivo pdf que se puede conseguir en la siguiente dirección:
http://searchnetworking.techtarget.com/searchNetworking/Downloads/Cisco_IPv6.pdf

GNU Zebra - Septiembre 2002
www.zebra.org
Este documento describe, detalladamente, el funcionamiento de esta excelente herramienta que permite simular un router Cisco sobre Linux, FreeBsd, etc. Soporta IPv6 y los protocolos de ruteo RIPv6, BGP4+ y OSPFv6.

Linux IPv6 How-to - Peter Bieringer
http://www.bieringer.de/linux/IPv6/
Excelente documento para configurar IPv6 sobre Linux.

FreeBSD HandBook - FreeBSD Document Project
www.freebsd.org
Es muy completo en la parte de instalación y configuración del SO pero detalla muy poco sobre IPv6.

IPv6: Connecting to 6bone using manually configured tunnels
www.cisco.com
Documento muy simple y claro que explica como establecer un túnel manual entre dos routers Cisco

Internet Protocol (IP) Multicast – Technology Overview – Cisco Sytems
www.cisco.com
Explicación del funcionamiento del multicast y de los protocolos de ruteo multicast en IPv4.

Cisco - IP Multicast Training Material
http://www.vayner.net/Docs/Cisco/multicast/
Excelente sitio en Internet donde se explica todo lo referente a multicast para IPv4 (IGMP, PIM-SM, PIM-DM, etc.).

IPv6 Multicast Protocolos - Konstanstin Kabassanov - Octubre 2002
http://www.cting.upm.es/ponencias-jing/2002/konstantin/konstantin.PDF
Presentación muy claro y conciso que describe el protocolo MLD y los protocolos de ruteo.