Práctica 3

IPv6: Protocolo IP de Nueva Generación

3.1. Introducción

El protocolo IP que hemos venido utilizando hasta ahora es el protocolo IPv4, definido en el RFC 791, de 1981. Cuando se propuso, se pensó que un direccionamiento basado en 32 bits, lo que proporciona un espacio de direcciones de $2^{32} = 4294967296$, sería suficiente. Además, dicho espacio de direcciones se dividió en **clases**, según el tamaño de las redes objeto, como se muestra en la figura 3.1.

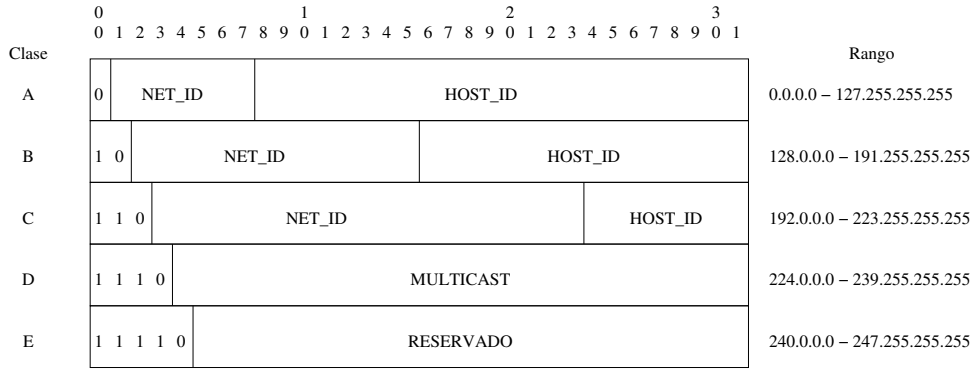


FIGURA 3.1: Asignación original de clases en IPv4

En los últimos años, con la expansión de Internet, el protocolo IPv4 ha debido afrontar dos problemas importantes: la escasez de direcciones libres, y la explosión de las tablas de rutas en los encaminadores troncales. Se han propuesto dos soluciones a dichos problemas: por un lado, para sortear la escasez de direcciones disponibles, se ha venido utilizando **NAT** (*Network Address Translation*, traducción de direcciones de Internet); por otro lado, el crecimiento desmesurado de las tablas de rutas en los encamina-

dores troncales se ha visto suavizada por el empleo del direccionamiento **CIDR** (*Classless Inter-Domain Routing*, encaminamiento sin clases).

3.2. Direccionamiento IPv6

Con 128 bits para especificar las direcciones IP, no se prevé que vaya a haber problemas de direccionamiento en el futuro. IPv6 viene a solucionar, entre otras cosas, el problema de escasez de direcciones de IPv4. El mecanismo NAT se vuelve innecesario y se recupera el modelo de conexión extremo-a-extremo de TCP. Además, disponer de mayor espacio de direcciones ayuda a un encaminamiento más eficaz, al permitir más niveles de jerarquía.

3.2.1. Tipos de direcciones

En IPv6 se han definido tres tipos de direcciones: **unicast**, **anycast** y **multicast**. Se debe destacar que no existen direcciones *broadcast*.

Unicast Una dirección *unicast* identifica a una única interfaz dentro de una red. Un datagrama dirigido a una dirección *unicast* se entrega a esa única interfaz.

Anycast Una dirección *anycast* identifica a un grupo de interfaces. Un datagrama dirigido a una dirección *anycast* se entrega a una única interfaz dentro de ese grupo. Idealmente, el datagrama se debe entregar a la interfaz más cercana al origen, según algún criterio predefinido.

Multicast Una dirección *multicast* identifica a un grupo de interfaces. Un datagrama dirigido a una dirección *multicast* debe ser entregado a todas las interfaces del grupo.

La distinción entre los diferentes tipos de direcciones se realiza gracias al *prefijo de formato*, como se verá en el apartado 3.2.4.

3.2.2. Notación

Las direcciones IPv6 son mucho más largas que las direcciones IPv4, por lo que se ha buscado una manera más compacta de escribirlas. Se ha elegido el formato hexadecimal, agrupando los dígitos en 8 grupos de 16 bits (4 dígitos hexadecimales) y separando cada grupo por el símbolo ":". Por ejemplo, una dirección *unicast* puede escribirse así:

```
2001:0db8:1f13:01bd:0000:0000:0000:0001
```
g_0 g_1 g_2 g_3 g_4 g_5 g_6 g_7

Para los dígitos hexadecimales pueden emplearse tanto letras mayúsculas como minúsculas. Por otro lado, los 0 al comienzo de un grupo pueden omitirse. Así, se podría escribir la dirección anterior de manera más compacta como:

```
2001: db8:1f13: 1bd:   0:   0:   0:   1
   g0    g1    g2    g3    g4    g5    g6   g7
```

donde hemos suprimido los 0 iniciales de los grupos g_1, y del g_3 al g_7.

Además, los grupos contiguos de todo 0 se pueden comprimir con el símbolo "::". En el ejemplo, los grupos g_4 al g_6 son todo 0, por lo que se puede escribir:

```
2001:db8:1f13:1bd::1
```

Sin embargo, para evitar ambigüedades, el símbolo "::" sólo puede aparecer una vez en una dirección. Se puede emplear también para representar grupos consecutivos de 0 al comienzo o al final de la dirección. Por ejemplo, la dirección

```
0:0:0:0:0:0:0:1
```

puede escribirse como

```
::1
```

y la dirección

```
2001:db8:1f13:1bd:0:0:0:0
```

se puede escribir como

```
2001:db8:1f13:1bd::
```

Para las direcciones de prefijo de red IPv6 escritas de esta manera se emplea la notación *dirección_IPv6/longitud_de_prefijo*. El campo de *longitud_de_prefijo* es un número decimal que indica cuántos bits de la dirección forman parte del prefijo de red. Por ejemplo, para el caso anterior, si la longitud del prefijo es de 64 bits, se escribiría:

```
2001:db8:1f13:1bd::1/64
```

Existen dos direcciones *unicast* especiales: la dirección de *loopback* y la dirección *indeterminada*. La dirección de *loopback* se define como:

```
0:0:0:0:0:0:0:1
```

o, de manera compacta,

```
::1
```

y la dirección indeterminada como:

```
0:0:0:0:0:0:0:0
```

es decir

```
::
```

3.2.3. Ámbitos

La definición de ámbitos es una novedad en IPv6. Un ámbito es el *espacio* donde la dirección es válida. En las direcciones *unicast* se definen tres tipos de ámbitos:[1] enlace local, sitio local y global.

Una dirección con ámbito de enlace local es sólo válida dentro de la subred donde está definida. Ningún encaminador dará salida a un datagrama que vaya dirigido a una dirección de enlace local. Estas direcciones se identifican por comenzar con el prefijo `fe80::/10`.[2].

Una dirección con ámbito de sitio local es sólo válida dentro de un *sitio*. Esta es una definición un tanto vaga, ya que la validez de la dirección no viene condicionada por las características físicas de la red, sino que debe configurarse de manera manual. Un *sitio* puede ser, por ejemplo, la red de un campus universitario o la de una empresa. Los encaminadores dentro del sitio sí deben encaminar estas direcciones, pero no así un encaminador que conecte el sitio con otros sitios o con Internet. Son el equivalente a las direcciones privadas de IPv4. Una dirección de ámbito de sitio local comienza por el prefijo `fec0::/10`.[3]

Una dirección de ámbito global es válida en toda Internet. Las direcciones *unicast* de ámbito global las llamaremos *direcciones unicast globales agregables*. Comienzan por el prefijo `2000::/3`.

En el caso de las direcciones *multicast*, se tiene una variedad mucho más amplia en la definición de ámbitos, pues el ámbito se especifica con un campo de 4 bits, lo que permite definir 16 posibilidades:

0	reservado	8	organización local
1	interfaz local	9	sin asignar
2	enlace local	A	sin asignar
3	reservado	B	sin asignar
4	administración local	C	sin asignar
5	sitio local	D	sin asignar
6	sin asignar	E	global
7	sin asignar	F	reservado

Los ámbitos que están marcados como *sin asignar* pueden ser utilizados de manera local, según los requisitos de administración de la red. Todas las direcciones *multicast* comienzan por el prefijo `ff00::/8`.

3.2.4. Formato de las direcciones

Las direcciones IPv6 están mucho más estructuradas que las de IPv4. Una dirección IPv6 siempre comienza por un campo denominado **prefijo de formato**[4]. En la sección 3.2.3 se han definido ya algunos: `fe80::/10` para las direcciones *unicast* de enlace local; `fec0::/10` para las direcciones *unicast* de sitio

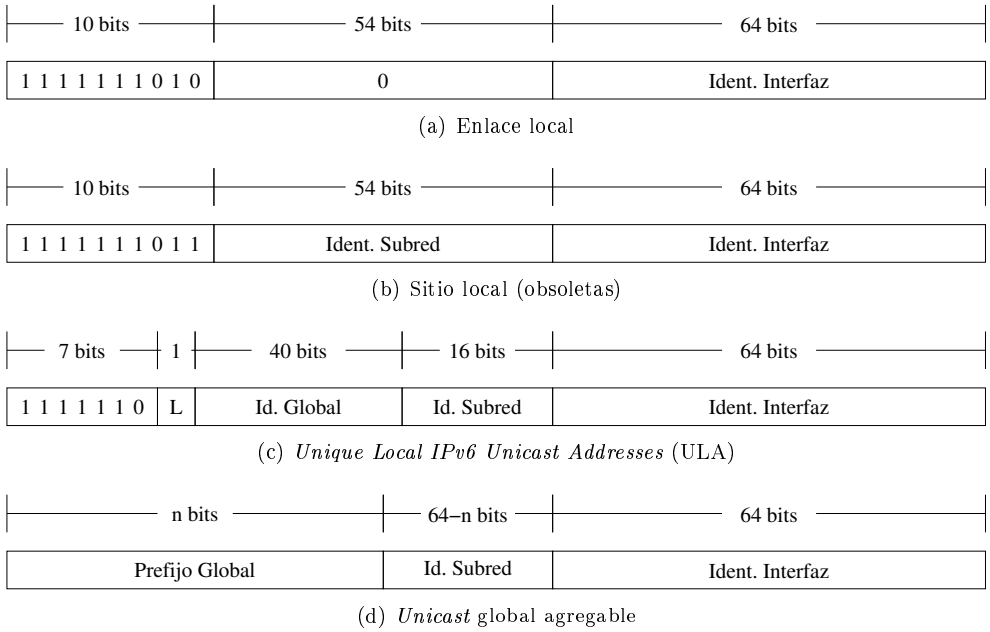
[1] Véase el RFC 3513.

[2] Este será el llamado *prefijo de formato*, que estudiaremos en la sección 3.2.4.

[3] Desde el documento RFC 3879 del año 2003, se desaconseja el uso de este tipo de direcciones. En su lugar se sugiere emplear las direcciones *unicast* locales únicas, ULA (de *Unique Local IPv6 Unicast Addresses*) definidas en el documento RFC 4193. Las ULA comienzan por el prefijo `fc00::/7`.

[4] La estructura de las direcciones IPv6 está definida en el `RFC 4291`.

local; `fc00::/7` para las direcciones ULA; `2000::/3` para las direcciones *unicast* globales agregables[5], y `ff00::/8` para las direcciones *multicast*. Las direcciones *anycast* no tienen un prefijo de formato específico, pues su espacio de direcciones coincide con el de las direcciones *unicast* globales agregables. Se puede ver el formato general de las direcciones *unicast* en la figura 3.2.



(a) Enlace local

(b) Sitio local (obsoletas)

(c) *Unique Local IPv6 Unicast Addresses* (ULA)

(d) *Unicast* global agregable

FIGURA 3.2: Formato de direcciones *unicast* IPv6.

En un principio se definieron las direcciones de sitio local, con prefijo fec0::/10 para que una organización pudiera definir su propio espacio de direcciones privado, sin conexión con el exterior. Sin embargo, pronto se vio que este mecanismo era muy problemático cuando se producían fusiones entre distintas entidades, pues la probabilidad de que colisionaran ambos espacios privados de direcciones era muy grande. Para solucionarlo, se han definido otras direcciones para uso privado, pero con un formato distinto: las ULA. En la figura 3.2(c) se muestra el formato de estas direcciones. Los 7 primeros bits son el prefijo de formato, `fc00::/7`. El siguiente bit se denomina bit `L`, que vale 1 cuanto la gestión es local. En el futuro se podrían definir direcciones con este bit a 0, pero actualmente debe valer siempre 1. Por tanto, cualquier dirección ULA comienza por `fd00::/8`. A continuación se añade un identificador de sitio de 40 bits. Este identificador se genera de manera pseudoaleatoria por un método descrito en el RFC 4193. La probabilidad de que dos sitios generen el mismo identificador es muy baja. Así, en el caso mencionado antes, de fusión entre dos entidades distintas, la probabilidad de colisión en el espacio de direcciones es muy baja.

Una dirección *unicast* global agregable es la que se utiliza para conseguir la conectividad global en Internet. De los 128 bits de la dirección, los 64 bits más significativos identifican a la red, mientras que

[5]Este prefijo de formato, además de una estructura jerárquica de las direcciones *unicast* globales agregables se definió en el `RFC 2374`. Sin embargo, en el `RFC 3587` se simplifica el formato de dichas direcciones al aquí presentado y se elimina el prefijo de formato aunque, actualmente, todos los prefijos delegados por IANA comienzan por `2000::/3`.

los 64 menos significativos identifican a una interfaz concreta dentro de esa red. Los bits que representan a la red tienen una estructura jerárquica más compleja que en IPv4. Los bits que identifican a la interfaz pueden configurarse a mano o bien de manera automática a partir de la dirección MAC de dicho interfaz, como se verá más adelante en la sección 3.6. En la figura 3.2(d) se muestra el formato de las direcciones *unicast* global agregable.

La estructura de las direcciones de multidifusión se muestra en la figura 3.3.

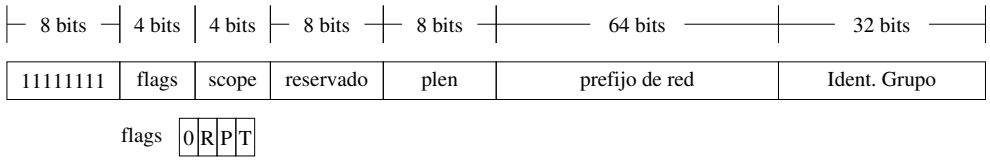


FIGURA 3.3: Formato de las direcciones de multidifusión.

El indicador `T` sirve para especificar si la dirección es permanente (`T=0`) o temporal (`T=1`). Las direcciones permanentes has sido especificadas por IANA (*Internet Assigned Number Authority*). No entraremos en más detalles sobre las direcciones de multidifusión, pero se verá el uso de algunas de ellas más adelante. Algunas direcciones de multidifusión habituales son las que se muestran en la tabla 3.1.

Todos los nodos	
`ff01::1`	Interfaz local
`ff02::1`	Enlace local
Todos los encaminadores	
`ff01::2`	Interfaz local
`ff02::2`	Enlace local
`ff05::2`	Sitio local
Todos los encaminadores RIP	
`ff02::9`	Enlace local

TABLA 3.1: Algunas direcciones de multidifusión

3.3. Datagramas IPv6

El formato de los datagramas IPv6 se puede ver en la figura 3.4. Aunque la longitud total de la cabecera de un datagrama IPv6 es mayor que en IPv4, su estructura se ha simplificado[6]. Muchos campos de la cabecera IPv4 han desaparecido, en particular todos los referentes a la fragmentación y reensamblado. La cabecera en IPv6 tiene una longitud fija, lo que facilita su tratamiento por parte de los encaminadores intermedios. Si es necesario añadir información adicional, se hace mediante cabeceras de extensión.

Versión: Este campo de 4 bits indica la versión de IP del datagrama, y debe valer 6.

[6] El formato del datagrama IPv6 está definido en el RFC 2460.

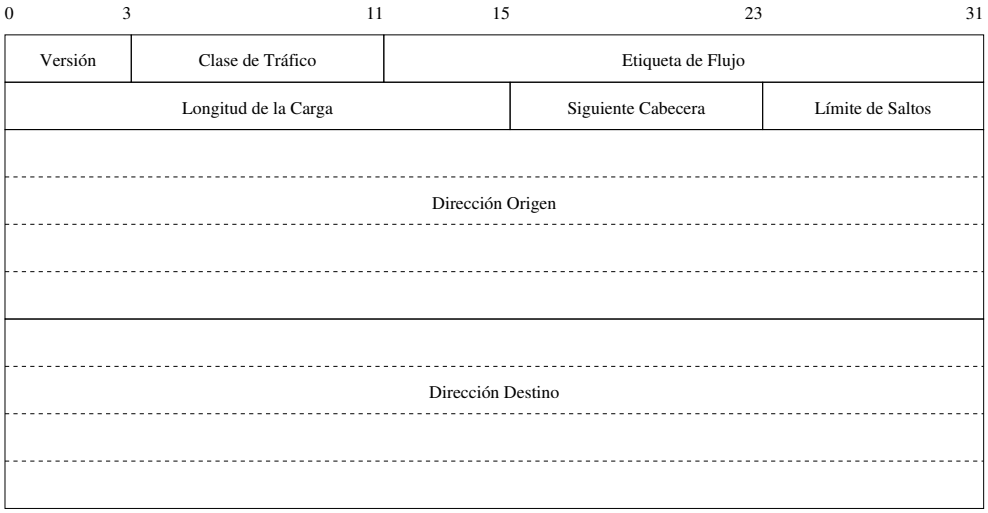


FIGURA 3.4: Formato de datagrama IPv6.

Tipo de Tráfico: Es un campo de 8 bits que puede servir para identificar distintos tipos de tráfico y prioridades, como por ejemplo tráfico en tiempo real.

Etiqueta de Flujo: Campo de 20 bits que puede identificar datagramas pertenecientes al mismo flujo de datos.

Longitud de la Carga: Es la longitud de los datos que siguen a la cabecera IPv6. Puesto que la longitud de la cabecera es fija, no se incluye en el cómputo de este campo.

Siguiente Cabecera: Es un campo de 8 bits que contiene el número que identifica el protocolo o cabecera de extensión que sigue a la cabecera IPv6.

Límite de Saltos: Es equivalente al campo `TTL` de la cabecera IPv4, con la diferencia de que en IPv6 se descuenta 1 en cada encaminador intermedio, independientemente del tiempo que el datagrama pase en dicho encaminador.

Dirección Origen: 128 bits que identifican al nodo origen del datagrama.

Dirección Destino: Los 128 bits que identifican al destinatario del datagrama.

3.4. Configuración estática

Para configurar las interfaces con direcciones IPv6 se puede emplear la orden `ip`, como hacíamos con IPv4. Ahora es imprescindible especificar la longitud del prefijo, pues no hay clases predefinidas:

```
# ip addr add fd12:761f:e831:28ea::1/64 dev eth0
```

Las rutas en IPv6 se pueden comprobar con las órdenes `ip` y `route`:

```
# ip -6 route show
2001:db8:1f13:1bd::/64 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440 hoplimit 0
fd12:761f:e831:28ea:1::/64 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440 hoplimit 0
fe80::/64 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440 hoplimit 0
default via 2001:db8:1f13:1bd::1 dev eth0  metric 1  mtu 1500 advmss 1440 hoplimit 0

# route -6
Kernel IPv6 routing table
Destination                    Next Hop                   Flag Met Ref Use If
2001:db8:1f13:1bd::/64         ::                         U    256 0     1 eth0
fd12:761f:e831:28ea::/64       ::                         U    256 0     0 eth0
fe80::/64                      ::                         U    256 0     0 eth0
::/0                           2001:db8:1f13:1bd::1       UG   1   3    25 eth0
::/0                           ::                         !n   -1  1    27 lo
::1/128                        ::                         Un   0   1    76 lo
2001:db8:1f13:1bd::501a/128    ::                         Un   0   1  1040 lo
fd12:761f:e831:28ea::1/128     ::                         Un   0   1     0 lo
fe80::21d:72ff:fe79:a8b4/128   ::                         Un   0   1     8 lo
ff00::/8                       ::                         U    256 0     0 eth0
::/0                           ::                         !n   -1  1    27 lo
```

Ejercicio: Iniciar dos máquinas virtuales, `uml1` y `uml2`. Configurar sus interfaces `eth0` con las direcciones `2001:db8:1::1/64` y `2001:db8:1::2/64` respectivamente. Comprobar con la orden `ping6` que son alcanzables mutuamente. Utilizar las órdenes `ip` y `route` para ver la tabla de rutas del *kernel*.

3.4.1. Configuración permanente

También se puede utilizar el archivo `/etc/network/interfaces` para guardar la configuración permanente de las interfaces, igual que hacíamos con IPv4:

```
 auto eth0
 iface eth0 inet static
        address 192.168.1.26
        netmask 255.255.255.0
        broadcast 192.168.1.255
        gateway 192.168.1.1
iface eth0 inet6 static
        address 2001:db8:1f13:1bd::501a
```

```
netmask 64
gateway 2001:db8:1f13:1bd::1
```

En este ejemplo se está configurando la interfaz `eth0` tanto para IPv4 como para IPv6. Además se está añadiendo una ruta por defecto, en la línea `gateway`.

3.5. Descubrimiento de vecinos

A menudo, la capa de red necesita conocer algunos detalles de la capa de enlace para poder funcionar correctamente. Esto es lo que sucede, por ejemplo, con IP funcionando sobre Ethernet. Para enviar un datagrama IP encapsulado en una trama Ethernet, necesitamos conocer la dirección MAC del siguiente salto, ya sea éste el destinatario final del datagrama, o el encaminador correspondiente. En IPv4, el protocolo ARP se encarga de averiguar la correspondencia entre direcciones IP y direcciones MAC. Una pregunta ARP viaja en una trama Ethernet dirigida a la dirección de difusión, por lo que se transmite por todo el dominio de difusión de la red y es capturada por todas las interfaces de red conectados. Como puede suponerse, esto provoca una degradación del rendimiento en redes grandes. Se estudiará que el mecanismo adoptado por IPv6 ayuda a paliar éste y otros problemas.

Además, en IPv6 se ha previsto un mecanismo de autoconfiguración de las direcciones que hace innecesario el uso de servidores DHCP específicos y facilitan la movilidad, aunque también pueden usarse servidores DHCPv6.

Se ha sustituido el protocolo ARP por el de **Descubrimiento de Vecinos** (**ND**, de *Neighbor Discovery*) que hace uso de ICMPv6. Explicado de manera somera, cuando un nodo `A` necesita averiguar la dirección MAC de otro nodo `B`, enviará un mensaje de Solicitud de Vecino (*Neighbor Solicitation*) a la dirección de multidifusión de nodo solicitado, que se construye de la siguiente manera: el prefijo de dirección de multidifusión de nodo solicitado es `ff02:0:0:0:0:1:ff00::/104`. Si el nodo `A` pregunta por la dirección MAC de la dirección de `B`, pongamos la `2001:db8:9260:50aa:021d:72ff:fe8a:a8d4`, toma los 24 bits menos significativos, en este caso `008a:a8d4` y los añade al prefijo, resultando la dirección de multidifusión `ff02::1:ff8a:a8d4`. El datagrama IPv6 resultante se encapsula en una trama Ethernet que va dirigida a la dirección de multidifusión `33:33:ff:8a:a8:d4`, obtenida a partir del prefijo `33:33` y añadiendo los 32 últimos bits de la dirección de multidifusión IPv6. La ventaja frente ARP es evidente: ahora, la solicitud de nodo no va dirigida a la dirección de difusión, sino a una dirección de multidifusión, por lo que sólo la interfaz, o interfaces, que estén asociadas a esa dirección procesarán la trama. Idealmente, sólo la interfaz del nodo por el que estamos preguntando procesará la trama. El formato de los mensajes ICMPv6 de solicitud y anuncio de vecino se puede ver en la figura 3.5.

Ejercicio: Con dos máquinas virtuales configuradas como en el apartado 3.4 utilizar la orden `ping6` mientras se analiza la red con `wireshark` para comprobar el intercambio de mensajes ICMPv6 de descubrimiento de vecinos.

Ejercicio: Utilizar la orden `ip neig` para comprobar y manipular la tabla de vecinos.

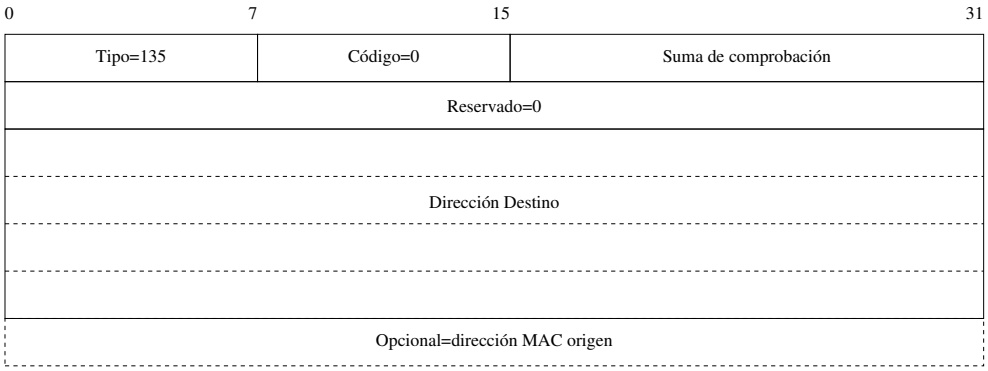


FIGURA 3.5: Formato de mensajes ICMPv6 de descubrimiento de vecino. (a) Solicitud de vecino. (b) Anuncio de vecino.

3.6. Autoconfiguración

El protocolo de descubrimiento de vecinos también se emplea para la autoconfiguración de las interfaces de red. Cuando se activa una interfaz, se envían por la red mensajes de descubrimiento de encaminador. Si hay algún encaminador IPv6 en la red, responderá con un anuncio que contiene, entre otra información, el prefijo de red. La máquina que se acaba de conectar puede, entonces, obtener una direccion IPv6 válida para conectarse a la red de manera automática.

3.6.1. Cálculo del identificador de interfaz

Una vez obtenido el prefijo de la red, sólo resta generar un identificador de interfaz que no colisione con ninguno de los ya existentes dentro de la red. Aunque esta generación se puede hacer de muchas maneras, se ha definido una forma para obtenerlo a partir de la dirección MAC[7]. Puesto que la dirección MAC tiene una longitud de 48 bits, y el identificador de interfaz de 64 bits, la dirección MAC se divide en

[7]Este método fue adoptado en el `RFC 2373`, junto con otros sistemas para los casos en que la interfaz no disponga de dirección MAC o tenga otro tipo de identificadores.

dos campos de 24 bits cada uno y se inserta entre ambos el patrón `fffe`. Además, el segundo bit menos significativo del byte más significativo de la dirección MAC se complementa de valor. Por ejemplo:

Dirección MAC: `00:01:02:03:04:05`

Prefijo de red: `2001:db8:1::/64`

A partir de la dirección MAC: `02 01 02 ff fe 03 04 05`

Dirección obtenida: `2001:db8:1::201:2ff:fe03:405/64`

Esta dirección es *tentativa*. No se puede utilizar hasta que estemos seguros de que no entra en conflicto con ninguna dirección existente previamente en la red. Por eso se envía una *solicitud de vecino* sobre esta dirección. Si obtenemos respuesta, es que la dirección está en uso y no se puede utilizar. Si nadie responde a la solicitud de vecino, significa que esta dirección no está siendo utilizada por otra máquina y, por tanto, se puede asignar a la interfaz[8].

En caso de que la autoconfiguración falle, por detección de dirección duplicada, es necesario realizar una configuración manual de la interfaz.

El mecanismo de autoconfiguración se puede desactivar o activar mediante las variables del núcleo `net.ipv6.conf.*.autoconf` y `net.ipv6.conf.*.accept_ra`.

3.6.2. Anuncio de prefijos mediante `quagga`

Para que la autoconfiguración sea posible necesitamos tener en la red un encaminador que anuncie el prefijo, o prefijos, que se utilizan en la red. Cuando este encaminador recibe un mensaje de descubrimiento de encaminador, responde con la información correspondiente[9].

Podemos emplear el demonio `zebra` para configurar un encaminador que anuncie prefijos IPv6. Para ello, en el archivo `/etc/quagga/zebra.conf` debemos incluir la información necesaria. Por ejemplo:

```
interface eth0
  no ipv6 nd suppress-ra
  ipv6 nd prefix 2001:db8:1::/64 86400 3600
  ipv6 nd ra-interval 600
  ipv6 nd ra-lifetime 1800
```

Ejercicio: Iniciar tres máquinas virtuales, `uml1`, `uml2` y `uml3`. Configurar `uml1` para que anuncie el prefijo de red `2001:db8:1::/64` por su interfaz `eth0`. Asegurarse de que las interfaces `eth0` de las máquinas `uml2` y `uml3` no tienen asignada ninguna dirección estática (al ejecutar la orden `ip -6 addr show dev eth0` sólo debe aparecer una dirección de enlace local con prefijo `fe80::/64`). Levantar la interfaz `eth0` de estas máquinas con la orden `ifconfig eth0 up`. Comprobar con `wireshark` los mensajes intercambiados, y con las órdenes `ip -6 addr show` e `ip -6 route show` las direcciones y rutas configuradas.

[8] Este mecanismo se denomina **prueba de dirección duplicada** (*duplicate address detection, dad*) y se emplea siempre que se asigna una dirección IPv6 de tipo *unicast* a una interfaz, ya sea por autoconfiguración o de manera manual.

[9] No entraremos aquí en el formato de los mensajes de anuncio de encaminador (*router advertisement*). Estos pueden, por ejemplo, incluir información relativa a los servidores DNS presentes en la red (`RFC 6106`), o la necesidad o no de emplear un servidor DHCPv6 para la obtención de información adicional (`RFC 3736`).

Ejercicio: Añadir a `uml1` el anuncio del prefijo `2001:db8:101:1::/64` además del configurado anteriormente. Comprobar que las máquinas `uml2` y `uml3` adquieren las direcciones correspondientes. Con la orden `ping6`, verificar que cada una de las máquinas es accesible desde todas las demás.

3.6.3. Extensiones de privacidad

Que una parte de la dirección IP dependa de la dirección MAC y sea fija plantea un problema de privacidad. Por plantear un caso, una empresa conoce las direcciones MAC de los equipos portátiles que da a sus empleados. Desde el servidor WEB de la empresa, por ejemplo, se podría rastrear entonces desde dónde se ha conectado cada uno de ellos sin más que examinar en los archivos de traza del servidor las direcciones IP de las conexiones, puesto que los 64 bits menos significativos están asociados a cada empleado concreto, y los 64 bits más significativos identifican a la red desde la cual se conecta.

Para evitar el rastreo de los equipos se han propuesto varios mecanismos. En el `RFC 3041` se propuso generar, además de la dirección IP obtenida mediante el método mencionado anteriormente, otra en la que el identificador de interfaz se ha obtenido a partir de cierto método aleatorio. Esta dirección adicional será la preferida para las conexiones salientes. Además, las direcciones generadas cambian periódicamente. El trabajo del documento `RFC 3041` ha sido actualizado en el `RFC 4941`.

En Linux se controla este mecanismo mediante la variable `net.ipv6.conf.`*iface*`.use_tempaddr`. Cuando esta variable vale 0 (su valor por defecto), entonces sólo se genera la dirección basada en el identificador de interfaz. Cuando vale 1, se genera una dirección aleatoria, que se empleará en caso de que falle la prueba de dirección duplicada de la autoconfiguración. Si la variable vale 2, se genera esta dirección aleatoria y además se utiliza como dirección origen en las conexiones salientes[10].

Otros parámetros que afectan a las extensiones de privacidad son el tiempo de vida válido (*valid_lft*) y el tiempo de vida preferido (*preferred_lft*). El primero indica el tiempo máximo durante el cual la dirección seguirá siendo válida, para las conexiones que ya hayan sido establecidas. El segundo, el tiempo que resta para que la dirección sea renovada. Ambos parámetros se pueden configurar con las correspondientes variables `sysctl`: `net.ipv6.conf.`*iface*`.temp_valid_lft` y `net.ipv6.conf.`*iface*`.temp_prefered_lft`, donde *iface* debe sustituirse por el nombre de la interfaz concreta.

En el `RFC 7217` se propone un método para generar identificadores de interfaz "opacos" y estables, es decir, que no es posible obtener la dirección MAC a partir de dicho identificador, que el mismo cambia al cambiar de red, pero que permanece estable siempre que nos conectemos a la misma red. Ahora es la opción por defecto en la mayoría de las distribuciones de GNU/Linux.

[10] Existe una amplia discusión sobre la conveniencia o no del empleo de las opciones de privacidad. En el `RFC 4941` se especifica que dichas extensiones deberían estar deshabilitadas por defecto. Muchos administradores de red apoyan esta política e, incluso, abogan por prohibirlas en sus redes completamente. Sin embargo, otros muchos afirman que las extensiones de privacidad deberían estar habilitadas de manera predeterminada y que es necesario buscar otros mecanismos, aparte de la mera dirección IP, para facilitar el análisis forense y la respuesta a incidentes.

3.7. Mecanismos de transición

Hasta que se implante definitivamente IPv6, convivirá con el antiguo protocolo IPv4. Así se dará el caso de tener "islas" IPv6 que deban conectarse a través de proveedores de servicios que todavía no soportan el nuevo protocolo y siguen usando IPv4. Durante el periodo de transición se han previsto diversos mecanismos para facilitar la adopción del nuevo protocolo. Uno de ellos es el de túneles configurados (*router-to-router tunneling*).

3.7.1. Túneles IPv6 sobre IPv4

Supongamos una situación como la descrita en la figura 3.6. Tenemos dos redes, $\texttt{LAN}_1$ y $\texttt{LAN}_2$, que utilizan el protocolo IPv6. Sin embargo, la red `WAN`, que es la que permitiría conectar ambas redes LAN, aún no ha realizado la transición y sólo soporta IPv4. El tráfico IPv6 entre ambas redes LAN no puede atravesar la WAN, pues los encaminadores que la forman no entienden el nuevo formato de los datagramas y los descartarían. Así pues, es necesario "ocultar" de alguna manera el tráfico IPv6 dentro de datagramas que puedan atravesar la WAN sin problemas.

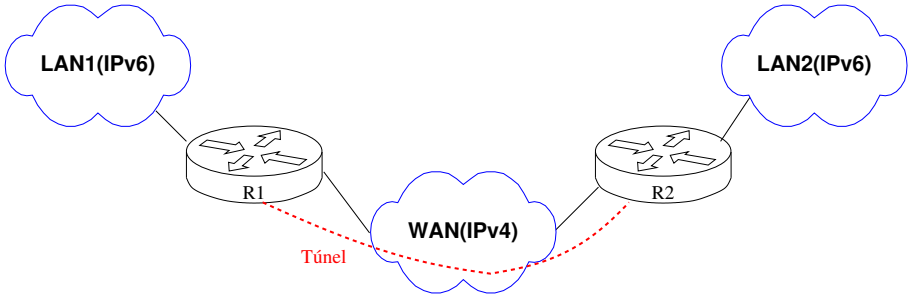


FIGURA 3.6: Túneles IPv6 sobre IPv4

Los encaminadores $\texttt{R}_1$ y $\texttt{R}_2$ están conectados a la red WAN que sólo soporta IPv4, y también a sus respectivas redes LAN, que funcionan con IPv6. Ambos encaminadores, por tanto, tienen implementación de doble pila, es decir, son capaces de trabajar con ambos protocolos. Este mecanismo de túneles funciona de la siguiente manera: cuando, por ejemplo, el encaminador $\texttt{R}_1$ recibe un datagrama IPv6 desde $\texttt{LAN}_1$ que va dirigido a una dirección IPv6 perteneciente a la red $\texttt{LAN}_2$, lo encapsula en el campo de carga de un datagrama IPv4. Este datagrama IPv4 viajará a través de la red `WAN`, con origen en $\texttt{R}_1$ y destino $\texttt{R}_2$. Cuando el datagrama llega a $\texttt{R}_2$, éste recupera el datagrama IPv6 original y lo redirige hacia su destino final dentro de $\texttt{LAN}_2$. Este modo de funcionamiento puede verse como un enlace virtual creado entre $\texttt{R}_1$ y $\texttt{R}_2$. El nuevo enlace, o túnel, se puede considerar como un enlace punto a punto entre las dos máquinas $\texttt{R}_1$ y $\texttt{R}_2$[11].

El núcleo 2.6 de Linux tiene soporte para túneles de este tipo. Para crear un túnel, se utiliza la orden `ip`:

[11] Existen otros tipos de túneles, como los túneles automáticos, túneles *host-to-host*, 6over4, etc., pero sólo estudiaremos aquí los túneles manuales entre encaminadores.

```
# ip tunnel add <nombre_tunel> mode sit remote <ip_remota> ttl <ttl> [local <ip_local>]
```

El parámetro `<nombre_tunel>` es el nombre que queremos asignar al nuevo dispositivo. Este nombre aparecerá como una interfaz más dentro de la lista de interfaces, y podremos asignarle una dirección, añadir rutas a través de ella, activarla, desactivarla, modificar la `mtu`, etc.

El parámetro `sit` significa *Simple Internet Transition*. Es el modo de decir a la orden `ip` que queremos crear un túnel IPv6 sobre IPv4.

La dirección `<ip_remota>` es la dirección IPv4 del otro extremo del túnel, es decir, hacia dónde se deben encaminar los datagramas IPv4 que transportan IPv6 a través de este enlace. En el caso de la figura 3.6, R_1 pondrá como dirección remota la de R_2, y viceversa.

El parámetro `<ttl>` es el tiempo de vida predeterminado (*Time To Live*) de los datagramas que se envían por este enlace. Si no se especifica, será el predeterminado del núcleo (64 en Linux).

En algunos casos puede ser necesario especificar la dirección IPv4 que debe utilizarse como origen. Esto se hace mediante el parámetro opcional `local <ip_local>`.

Ejercicio: Crear la configuración de la figura 3.7. Las máquinas `uml2` y `uml3` son encaminadores y las responsables de establecer el túnel. Las máquinas `uml1` y `uml4` sólo tienen configuradas direcciones IPv6. Su encaminador predeterminado es `uml2` o `uml3`.

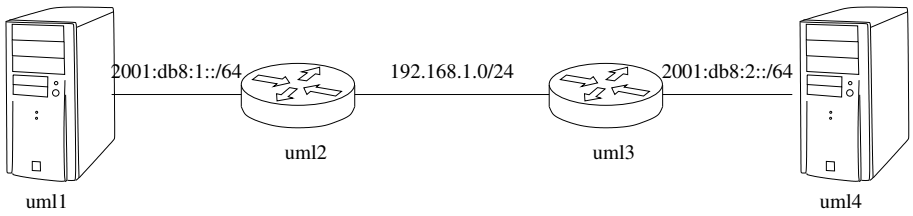


FIGURA 3.7: Práctica: túneles IPv6 sobre IPv4

En las máquinas `uml2` y `uml3` se deben añadir las rutas correspondientes a través de la interfaz del túnel:

```
uml2:~# ip route add 2001:db8:2::/64 dev <nombre_tunel>
uml3:~# ip route add 2001:db8:1::/64 dev <nombre_tunel>
```

y se activa el reenvío para IPv6 en ambas:

```
# sysctl -w net.ipv6.conf.all.forwarding=1
```

Enviar datagramas desde `uml1` hasta `uml4`. Capturar el tráfico con `wireshark` y examinar el encapsulado de IPv6 sobre IPv4 cuando pasa por el túnel, como se muestra en la figura 3.8. Puede observarse que el campo de `Protocolo` del datagrama IPv4 contiene el valor 41 (IPv6).

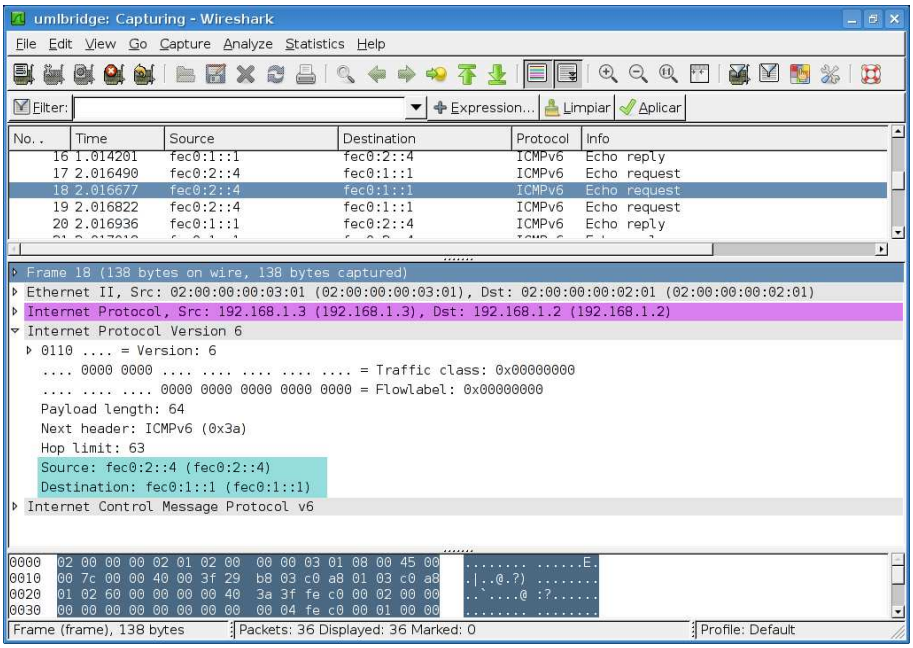


FIGURA 3.8: Encapsulado IPv6 sobre IPv4