Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN ENXEÑARÍA INFORMÁTICA
MENCIÓN EN TECNOLOXÍAS DA INFORMACIÓN

Práctica 2: IPv6 en INET

Estudante 1: Óscar Olveira Miniño

Estudante 2: Alejandro Javier Herrero Arango

A Coruña, noviembre de 2024.

Índice general

Índice de figuras

Capítulo 1

IPv4

1.1 Ejercicio 1.1

1.1.1 Muestra una captura del tráfico de paquetes DHCP intercambiados entre el nodo host[0] y los servidores DHCP durante el proceso de obtención de su IP, obtenida en Wireshark (Nota: para que los tiempos mostrados en Wireshark coincidan con los tiempos de simulación, activa Visualización → Formato de visualización de fecha → Segundos desde 1970-01-01). Explica lo que ocurre y para qué sirve cada paquete. Para facilitar la captura, configura el startTime del cliente DHCP para que se inicie antes en host[0] que el resto de equipos.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000002	0.0.0.0	255.255.255.255	DHCP	305	DHCP Discover - Transaction ID 0xc7f0aac
2	0.000010	192.168.0.10	255.255.255.255	DHCP	332	DHCP Offer - Transaction ID 0xc7f0aac
3	0.000013	0.0.0.0	255.255.255.255	DHCP	317	DHCP Request - Transaction ID 0x17c4aa2f
4	0.000013	192.168.0.1	255.255.255.255	DHCP	332	DHCP Offer - Transaction ID 0xc7f0aac
5	0.000021	192.168.0.10	255.255.255.255	DHCP	332	DHCP ACK - Transaction ID 0x17c4aa2f
6	4.000005	0.0.0.0	255.255.255.255	DHCP	305	DHCP Discover - Transaction ID 0x3716a675
7	4.000007	0.0.0.0	255.255.255.255	DHCP	305	DHCP Discover - Transaction ID 0x5821ccc0
8	4.000010	192.168.0.10	255.255.255.255	DHCP	332	DHCP Offer - Transaction ID 0x3716a675
9	4.000013	192.168.0.1	255.255.255.255	DHCP	332	DHCP Offer - Transaction ID 0x3716a675
10	4.000016	192.168.0.10	255.255.255.255	DHCP	332	DHCP Offer - Transaction ID 0x5821ccc0
11	4.000019	192.168.0.1	255.255.255.255	DHCP	332	DHCP Offer - Transaction ID 0x5821ccc0
12	4.000021	0.0.0.0	255.255.255.255	DHCP	317	DHCP Request - Transaction ID 0x1a4eb343
13	4.000024	192.168.0.10	255.255.255.255	DHCP	332	DHCP ACK - Transaction ID 0x1a4eb343
14	4.000027	0.0.0.0	255.255.255.255	DHCP	317	DHCP Request - Transaction ID 0x5ba252fb
15	4.000030	192.168.0.10	255.255.255.255	DHCP	332	DHCP ACK - Transaction ID 0x5ba252fb
16	6.000000	0a:aa:00:00:00:02	Broadcast	ARP	64	Who has 192.168.0.10? Tell 192.168.0.11
17	6.000002	0a:aa:00:00:00:05	0a:aa:00:00:00:02	ARP	64	192.168.0.10 is at 0a:aa:00:00:00:05
18	6.000003	192.168.0.11	192.168.0.10	TCP	64	1025 → 80 [SYN] Seq=0 Win=7504 Len=0 MSS=536
19	6.000005	192.168.0.10	192.168.0.11	TCP	64	80 → 1025 [SYN, ACK] Seq=0 Ack=1 Win=7504 Len=0 MSS=536
20	6.000005	192.168.0.11	192.168.0.10	TCP	64	1025 → 80 [ACK] Seq=1 Ack=1 Win=7504 Len=0
21	6.000007	192.168.0.11	192.168.0.10	TCP	258	1025 → 80 [ACK] Seq=1 Ack=1 Win=7504 Len=200
22	6.000011	192.168.0.10	192.168.0.11	TCP	64	80 → 1025 [ACK] Seq=1 Ack=201 Win=7504 Len=0
23	6.000020	192.168.0.10	192.168.0.11	TCP	594	80 → 1025 [ACK] Seq=1 Ack=201 Win=7504 Len=536
24	6.000021	192.168.0.11	192.168.0.10	TCP	64	1025 → 80 [ACK] Seq=201 Ack=537 Win=7504 Len=0
25	6.000021	192.168.0.11	192.168.0.10	TCP	64	1025 → 80 [FIN, ACK] Seq=201 Ack=537 Win=7504 Len=0
26	6.000030	192.168.0.10	192.168.0.11	TCP	522	80 → 1025 [ACK] Seq=537 Ack=201 Win=7504 Len=464
27	6.000030	192.168.0.11	192.168.0.10	TCP	64	1025 → 80 [ACK] Seq=202 Ack=1001 Win=7504 Len=0
28	6.000030	192.168.0.10	192.168.0.11	TCP	64	80 → 1025 [ACK] Seq=1001 Ack=202 Win=7504 Len=0
29	6.000031	192.168.0.10	192.168.0.11	TCP	64	80 → 1025 [FIN, ACK] Seq=1001 Ack=202 Win=7504 Len=0
30	6.000032	192.168.0.11	192.168.0.10	TCP	64	1025 → 80 [ACK] Seq=202 Ack=1002 Win=7504 Len=0
31	10.000032	192.168.0.11	192.168.0.10	TCP	64	1026 → 80 [SYN] Seq=0 Win=7504 Len=0 MSS=536
32	10.000034	192.168.0.10	192.168.0.11	TCP	64	80 → 1026 [SYN, ACK] Seq=0 Ack=1 Win=7504 Len=0 MSS=536
33	10.000034	192.168.0.11	192.168.0.10	TCP	64	1026 → 80 [ACK] Seq=1 Ack=1 Win=7504 Len=0
34	10.000036	192.168.0.11	192.168.0.10	TCP	258	1026 → 80 [ACK] Seq=1 Ack=1 Win=7504 Len=200
35	10.000040	192.168.0.10	192.168.0.11	TCP	64	80 → 1026 [ACK] Seq=1 Ack=201 Win=7504 Len=0
36	10.000049	192.168.0.10	192.168.0.11	TCP	594	80 → 1026 [ACK] Seq=1 Ack=201 Win=7504 Len=536

Figura 1.1: Tráfico DHCP entre host[0] y los servidores

Como podemos observar, el host[0] empieza enviando un DHCPDISCOVER con el objetivo de descubrir un servidor DHCP disponible. El servidor local resulta ser el primero en responder la solicitud con un paquete DHCPOFFER con una dirección IP disponible. El cliente (host[0]), responde confirmando la elección del servidor local como su servidor DHCP y pidiendo la IP ofrecida mediante un DHCPREQUEST. El router también envía el paquete DHCPOFFER, pero lo hace más tarde, por lo que el cliente lo ignora y el router cancela la oferta (Ver 1.2). Finalmente, el servidor local contesta con un DHCPACK confirmando la recepción del DHCPREQUEST y

asignando definitivamente la IP elegida (Este proceso se repiten para los clientes host[1] y host[2]).

Posteriormente, el cliente host[0] intenta establecer una conexión con el servidor local, ya que este también ha sido configurado con una TcpGenericServerApp. Para ello, el cliente manda primero un broadcast ARP para averiguar cuál es la MAC de la máquina servidor con la IP que establece en la cabecera ARP. Después, el servidor contesta al broadcast ARP que mandó el cliente identificándose su MAC.

Finalmente, la conexión sigue adelante con los mensajes TCP correspondientes y repitiéndose cada 4 segundos (Esto ocurre ya que establecemos un idleInterval de 4 segundos).

Tipos de paquetes:

1. DHCPDISCOVER: Para descubrir servidores DHCP disponibles. El cliente (en este caso host[0]) manda un paquete con IP destino de broadcast.

2. DHCPOFFER: Respuesta del servidor a un paquete DHCPDISCOVER ofreciéndose como servidor DHCP. El servidor enviará un paquete donde la IP de origen será la del propio servidor (192.168.0.10) al destino 255.255.255.255, indicando además una IP ofrecida.

3. DHCPREQUEST: El cliente confirma la elección de su servidor DHCP y le solicita la IP ofrecida. Esta comunicación sigue siendo broadcast para informar al resto de servidores DHCP, si los hubiere para que cancelen sus ofertas.

4. DHCPACK: El servidor confirma la recepción de la petición por el cliente y le establece la IP solicitada. Este paquete tiene como destino la dirección de broadcast porque el cliente no ha recibido aún la IP, aunque se sabe a quién va dirigido gracias a un campo que incluye la MAC del cliente destino.

1.2 Ejercicio 1.2

1.2.1 ¿Cuál de los servidores proporciona la IP a host[0]? ¿Sabe el otro servidor que host[0] no cogió la IP ofrecida por él? ¿Cómo? (Muestra el contenido de los paquetes relevantes en Wireshark.)

Por lo que se puede ver en la figura 1.1, en el intercambio inicial de paquetes DHCP, es el servidor local, con IP 192.168.0.10, quien envía el primer paquete DHCPOFFER. De esta manera, el servidor local, se adelanta al router, que envía su DHCPOFFER ya después de que el cliente responda con el DHCPREQUEST. Por tanto, es obvio deducir que el cliente está respondiendo y tomando la ip del servidor local. De todas formas, se pueden observar en detalle los paquetes 3 (DHCPREQUEST) y 5 (DHCPACK) para confirmar nuestra hipótesis. El paquete número 3 incluye los campos Requested IP Adress con la dirección IP solicitada por el cliente y DHCP Server Identifier con el servidor elegido para que le asigne esa IP. De igual forma, en el paquete número 5, aparece la IP asignada en el campo Your (client) IP Adress y el servidor que la asigna, de nuevo en la opción DHCP Server Identifier. El router, que es el otro servidor DHCP, sí descubre que host[0] no coge su dirección IP, abandonando su oferta y dejando libre esa dirección para más adelante. Esto es gracias a que el paquete DHCPREQUEST, en el que el cliente contesta con la IP solicitada y el servidor al que se solicita, se envía a todos los posibles destinatarios dentro de la red 192.168.0.0.

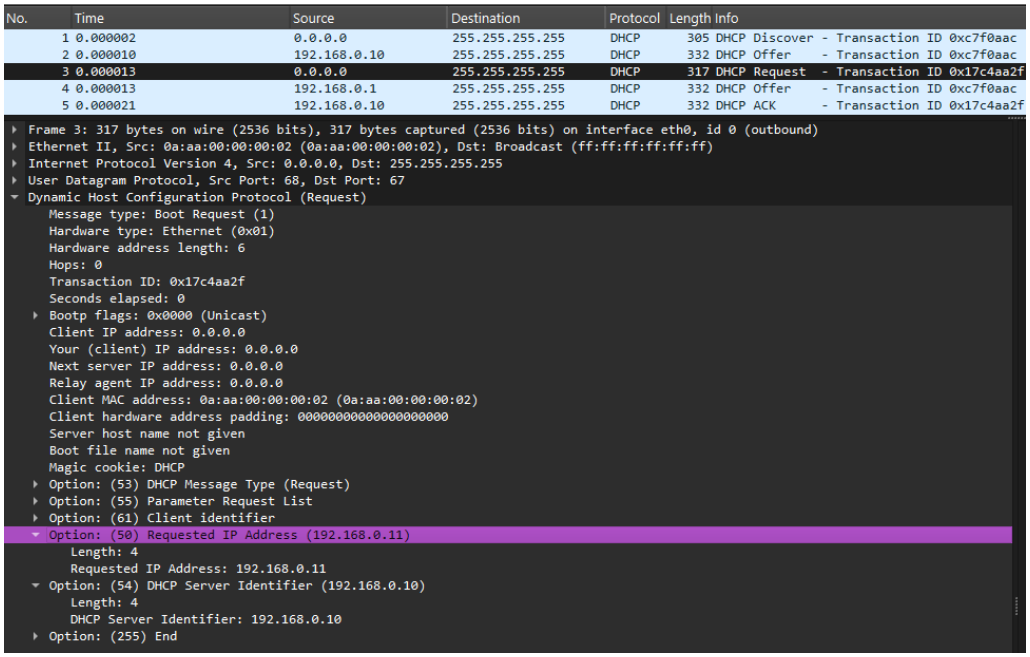


Figura 1.2: Detalle de paquete DHCPREQUEST de host[0]

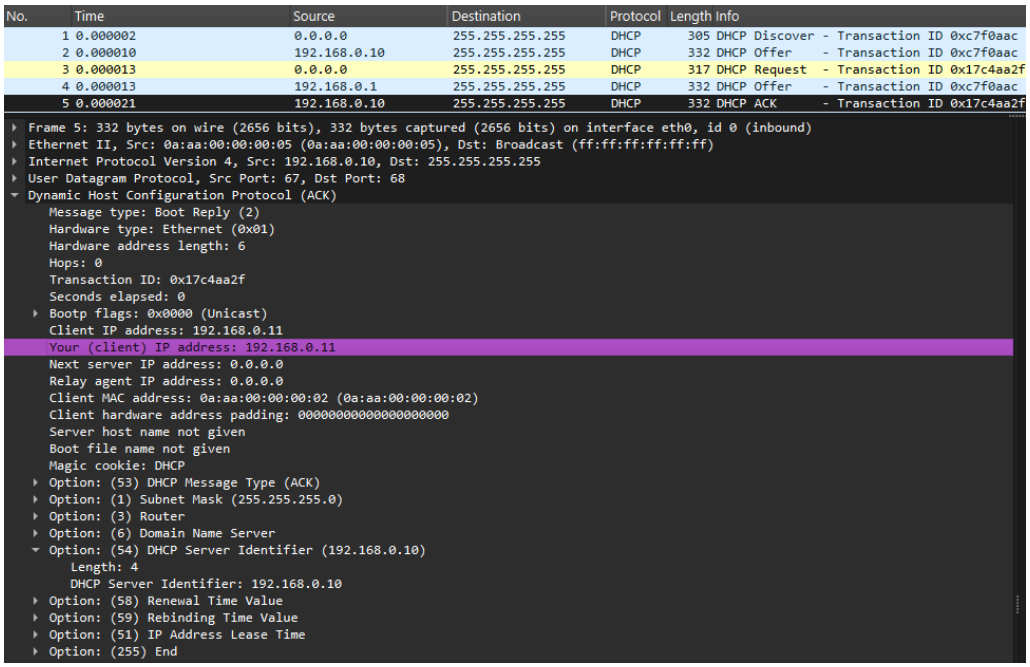


Figura 1.3: Detalle de paquete DHCPACK de server_local

1.3 Ejercicio 1.3

1.3.1 ¿De qué tipo son los primeros paquetes que se intercambian a partir de t = 6 segundos? ¿Cuál es su objetivo?

Los primeros paquetes que se intercambian a partir de 6s son paquetes ARP. El objetivo de estos paquetes es mapear una dirección IP con su correspondiente dirección MAC. Después, se guarda esta información en la caché ARP de los dispositivos involucrados. El host[0] pregunta por la MAC de la máquina con la IP 192.168.0.10, que es el servidor local que funciona como servidor TCP. Posteriormente, el servidor local responde enviando un paquete ARP de respuesta unicast al host con su dirección MAC. Sin este paso, no se podría establecer la conexión.

1.4 Ejercicio 1.4

1.4.1 ¿Cuáles son las direcciones origen y destino de estos paquetes (solicitud y respuesta)? ¿Tienen IP origen o destino? ¿Por qué?

En el paquete de solicitud, la dirección de origen es la MAC del cliente TCP host[0], mientras que la dirección de destino es ff:ff:ff:ff:ff:ff, es decir, la dirección de broadcast MAC, cualquier dispositivo en la LAN. En el paquete de respuesta, la dirección origen es la MAC de server_local y la dirección de destino la MAC de host[0]. Como ARP es un protocolo de capa de enlace, se envía a nivel ethernet y, por tanto, su origen y destino son, como se menciona anteriormente direcciones MAC y no IP. Sin embargo, las direcciones IP origen y destino sí están incluidas en el protocolo ARP, y por lo tanto en el paquete, dentro de los campos sender/target IP address, como se aprecia en la imagen.

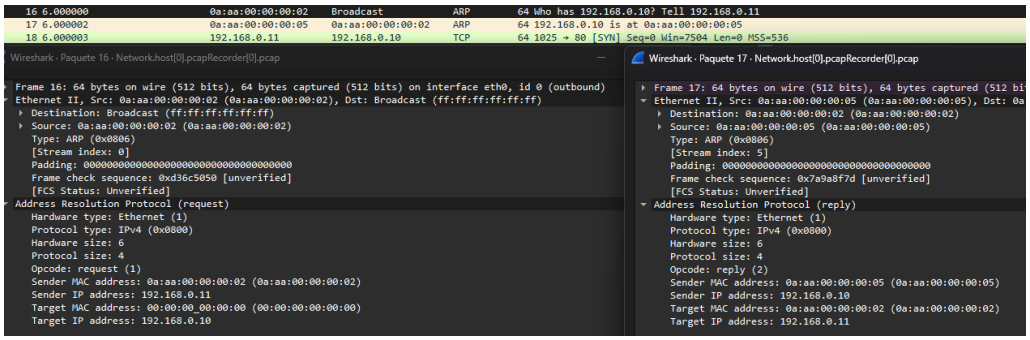


Figura 1.4: Detalle del intercambio de paquetes ARP entre host[0] y server_local

1.5 Ejercicio 1.5

1.5.1 Configura el servidor DHCP en serverlocal con numReservedAddresses = 10 y el cliente DHCP en host[2] para que arranque antes que los otros clientes DHCP. Esto hará que host[2] reciba la IP fija asignada a serverlocal (192.168.0.10). ¿Ocurre algún error cuando el host[2] recibe la IP de serverlocal? Configura el cliente TCP en host[0] para que se conecte a serverlocal y describe paso a paso qué ocurre durante el establecimiento de conexión TCP a partir de t = 6 segundos debido a esta duplicidad.

Configurando la red para que el host[2] reciba de primero la IP con un numReservedAddresses = 10, efectivamente recibe la IP estática del servidor local tal y como lo menciona en el enunciado y no ocurre ningún error aunque tengamos dos dispositivos con la misma IP. No obstante, aunque no ocurra ningún error, esto puede desencadenar problemas en la red ya que habría un conflicto de IPs como por ejemplo problemas de rendimiento en la red (retrasos en el procesamiento de paquetería o aumento de tráfico ARP al no poder resolver la MAC del dispositivo), pérdida de conexión, inconsistencias en las tablas ARP, etc.

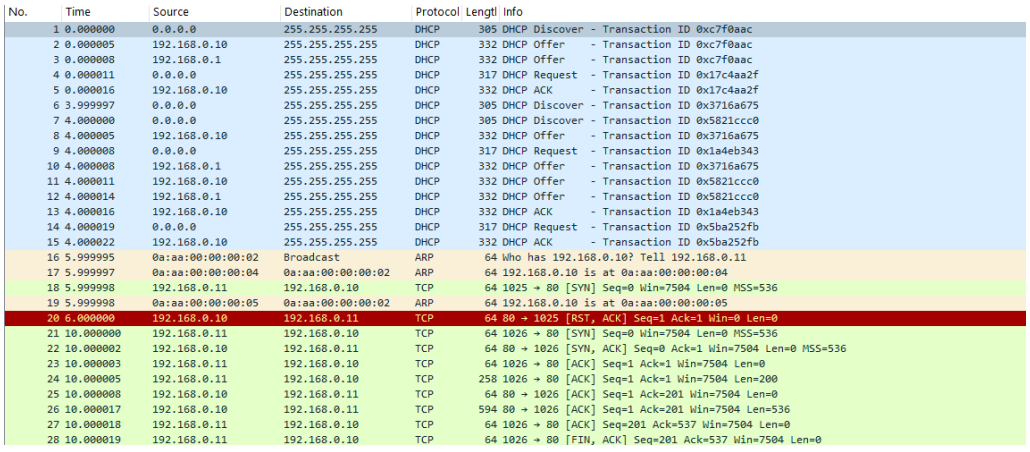


Figura 1.5: Tráfico DHCP entre host[0] y los servidores cuando host[2] recibe la ip servidor local

Configurando el cliente TCP en el host[0], como se puede ver en la imagen 1.5, el host[0] manda un paquete ARP modo broadcast para averiguar la MAC del dispositivo con la ip 192.168.0.10. Le contesta primero el servidor local, por lo que empieza la conexión en capa 4; pero, poco después, el host[2] contesta también al broadcast mandado por el host[0], ya que él tiene la misma IP asignada que el servidor local, por lo que se interrumpe la conexión del host[0] con el servidor local mandando un RST,ACK. Pasados 4 segundos (ya que establecemos

un idleInterval de 4 segundos), se vuelve establecer la conexión entre host[0] y el server local haciéndose esta de manera exitosa ya que en la tabla ARP del host[0] se almacenó como dispositivo correspondiente a la IP 192.168.0.10, la MAC de servidor local ya que fue el primer dispositivo en responde al mandar paquetes ARP en modo broadcast .

Capítulo 2

IPv6-SLAAC

2.1 Ejercicio 2.1

2.1.1 Muestra una captura del escenario en el momento inicial de la simulación en la que se vean todas las direcciones MAC e IP. Utilizando la dirección IP de host[0] como ejemplo, explica cómo se construye, destacando los campos y bits relevantes. Utiliza para explicarlo la notación IPv6 no abreviada (16 bytes: xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx)

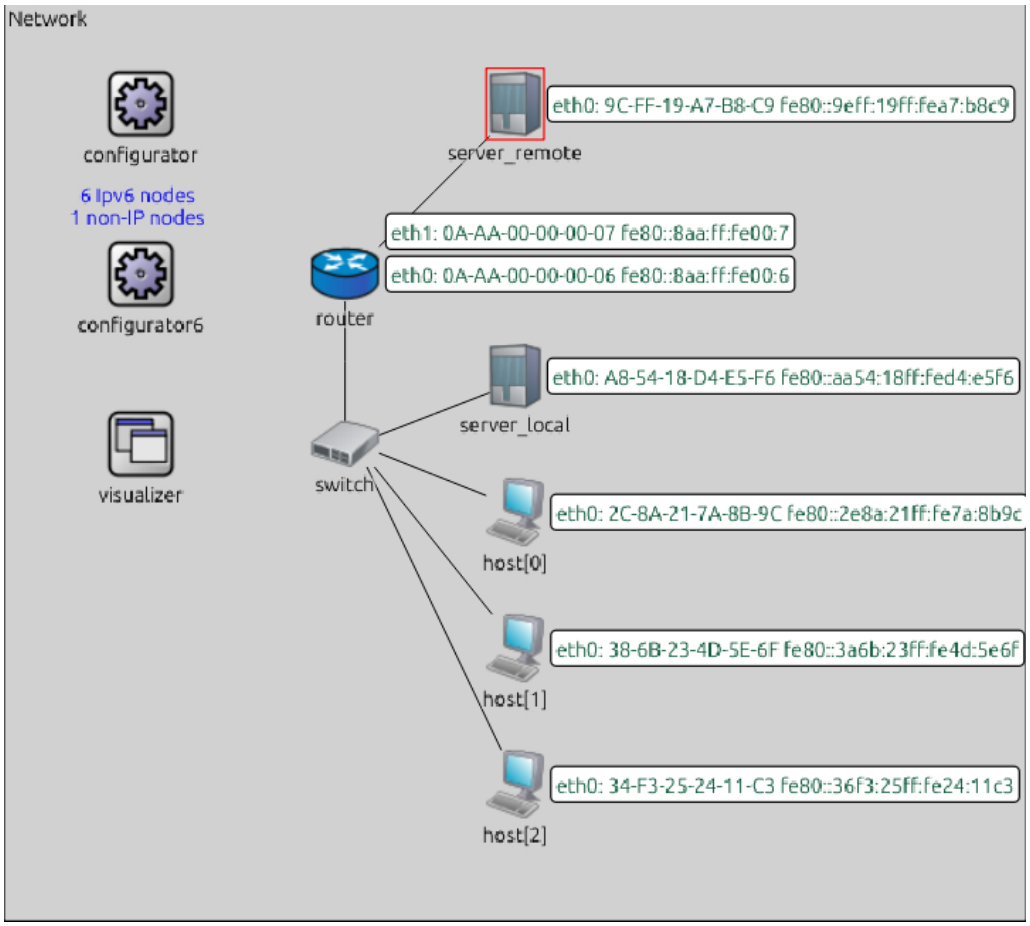


Figura 2.1: Escenario inicial dispositivos con IPv6

En el principio de la simulación, las direcciones que se muestran son direcciones unicast local de enlace. Estas son direcciones se generan automáticamente una vez que un dispositivo se conecta a una red. La estructura de este tipo de direcciones tienen como prefijo FE80::, no se enrutan y son únicas en ese enlace específico.

La segunda parte de la dirección se forma a partir de la dirección MAC del dispositivo. Para esto, fijamos el séptimo bit a 1 e insertamos 0XFFFE entre las dos mitades de la dirección MAC del dispositivo.

Como ejemplo, vamos a ver la dirección unicast local de enlace que genera el dispositivo host[0]. Como vemos en la imagen 2.1, su dirección está formada por el prefijo FE80:: y, después, 2E8A:21FF:FE7A:8B9C. Esa segunda parte, como se explicó antes, se forma a partir de su dirección MAC (2C-8A-21-7A-8B-9C). Como se puede observar, en los primeros 2 bytes de la dirección unicast local de enlace (2E8A), si lo comparamos con su dirección MAC, el segundo carácter se convierte en una E, ya que tenemos que añadir un uno en el séptimo bit (La letra C hexadecimal, que en binario es 1100, como su tercer bit corresponde al séptimo bit de la dirección unicast, pasa a ser 1 por lo que se convierte en E -> 1110). Llegados a este punto, tras el primer cambio, tenemos la siguiente estructura -> FE80::2E80:21.

Como se explicó anteriormente, una vez hecho el primer cambio, ahora añadimos 0XFFFE y posteriormente los últimos 3 bytes de la dirección MAC, por lo que queda como dirección final unicast local de enlace FE80::2E8A:21FF:FE7A:8B9C.

2.2 Ejercicio 2.2

2.2.1 Asigna al host[2] la misma dirección MAC que al host[0] y arranca la simulación. ¿Qué error ocurre antes de que haya transcurrido el primer segundo de simulación? Muestra una captura del error que aparece. ¿Qué paquete (tipo, origen y destino) provoca el error? ¿Por qué?

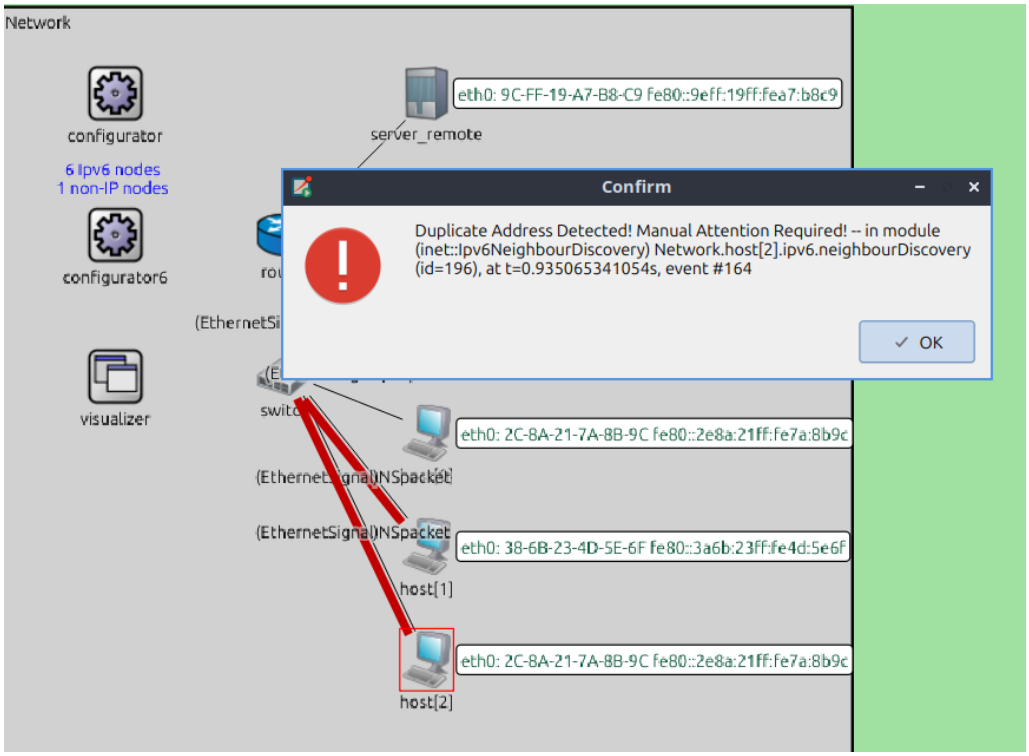


Figura 2.2: Fallo en la red con MAC host[2] igual a la de host[0]

Tal y como se muestra en la imagen 2.2, hay un fallo de direcciones duplicadas cuando el host[2] intenta

asignarse una dirección IPv6.

El paquete en concreto que provoca el error es de tipo ICMPv6 con mensaje NS (Neighbor Solicitation), donde el origen de momento es el host[2] (En ese momento sin ninguna dirección) y como destino una dirección multicast. Este error ocurre porque el host[2], cuando crea su dirección, le comunica a sus vecinos de la red (DAD) cómo es su dirección.Entonces, se encuentra que el host[0] tiene la misma dirección IPv6 que él se asignó, por lo que ocurre un conflicto de IPs.

2.3 Ejercicio 2.3

2.3.1 Cambia la MAC del host[2] de manera que coincida con la de host[0] en los últimos 3 bytes y difiera en los 3 primeros bytes (mantén esta MAC para el resto de las cuestiones). Asigna a serverremote la misma MAC que a host[0]. ¿Vuelve a ocurrir el error de dirección duplicada con serverremote y host[0]? ¿Por qué?

En este caso, al poner la misma MAC al host[0] y al server remote no produce ningún error porque son dispositivos que están en diferentes redes. El conflicto que sucedía en el ejercicio 2.2 sucede porque los dos dispositivos estaban en la misma red, por lo que salta el error de direcciones duplicadas. Esto ocurre igual que en IPv4 con las IPs privadas: En la misma red no puede haber dos IPs iguales, pero si hay dos redes diferentes, puede coincidir una IP privada de un dispositivo que está en la red1 con la IP de otro dispositivo que esté en la red2.

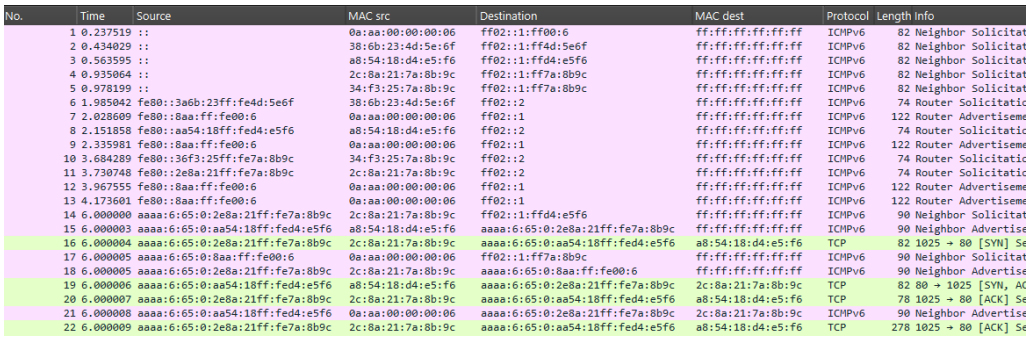
2.4 Ejercicio 2.4

2.4.1 ¿Cuánto tiempo transcurre desde el principio de la simulación hasta que el host[0] su IP link-local definitiva (i.e., fin de DAD)? Muestra la tabla de interfaces del nodo host[0] en la que se vea su estado antes y después del DAD timeout y explica qué cambia. (Nota: Qtenv muestra toda la información de cada interfaz en una línea; para verla correctamente copia el contenido con botón derecho → Copy Value y pégalo en la memoria como texto, en lugar de usar capturas de pantalla.)

Al comienzo de la simulación, host[0] comienza con su tabla de interfaces compuesta por la dirección de loopback en lo0 y su dirección IPv6 Link-Local en la interfaz eth0, producto de añadir al prefijo FF80:: el resultado de aplicar el proceso EUI-64 sobre su MAC. Esta última, como se aprecia en 2.5, está en estado *tentative* hasta que el proceso de DAD inicial termine y se confirme que puede tomar esa dirección.

El timeout del DAD, que comenzó con el envío del paquete de Neighbor Solicitation en el t = 0.93 (Ver 2.3), finaliza en t=2.86, como se puede ver en 2.4.

De esta forma, la tabla de interfaces después de ese momento, 2.6, cambia ligeramente, desapareciendo el *tentative* de la dirección. Esto se debe a que ningún vecino a contestado a su mensaje de NS en la red local, por lo que averigua que su dirección IP está libre y puede tomarla de forma definitiva.



No.	Time	Source	MAC src	Destination	MAC dest	Protocol	Length	Info
1	0.237519	::	0a:aa:00:00:00:06	ff02::1:ff00:6	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor Solicitat
2	0.434029	::	38:6b:23:4d:5e:6f	ff02::1:ff4d:5e6f	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor Solicitat
3	0.563595	::	a8:54:18:d4:e5:f6	ff02::1:ffd4:e5f6	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor Solicitat
4	0.935064	::	2c:8a:21:7a:8b:9c	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor Solicitat
5	0.978199	::	34:f3:25:7a:8b:9c	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor Solicitat
6	1.985042	fe80::3a6b:23ff:fe4d:5e6f	38:6b:23:4d:5e:6f	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Solicitatio
7	2.028609	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Advertiseme
8	2.151858	fe80::aa54:18ff:fed4:e5f6	a8:54:18:d4:e5:f6	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Solicitatio
9	2.335981	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Advertiseme
10	3.684289	fe80::36f3:25ff:fe7a:8b9c	34:f3:25:7a:8b:9c	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Solicitatio
11	3.730748	fe80::2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Solicitatio
12	3.967555	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Advertiseme
13	4.173601	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Advertiseme
14	6.000000	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	ff02::1:ffd4:e5f6	ff:ff:ff:ff:ff:ff	ICMPv6	90	Neighbor Solicitat
15	6.000003	aaaa:6:65:0:aa54:18ff:fed4:e5f6	a8:54:18:d4:e5:f6	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	90	Neighbor Advertise
16	6.000004	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	aaaa:6:65:0:aa54:18ff:fed4:e5f6	a8:54:18:d4:e5:f6	TCP	82	1025 → 80 [SYN] Se
17	6.000005	aaaa:6:65:0:8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	90	Neighbor Solicitat
18	6.000005	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	aaaa:6:65:0:8aa:ff:fe00:6	ff:ff:ff:ff:ff:ff	ICMPv6	90	Neighbor Advertise
19	6.000006	aaaa:6:65:0:aa54:18ff:fed4:e5f6	a8:54:18:d4:e5:f6	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	TCP	82	80 → 1025 [SYN, AC
20	6.000007	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	aaaa:6:65:0:aa54:18ff:fed4:e5f6	a8:54:18:d4:e5:f6	TCP	78	1025 → 80 [ACK] Se
21	6.000008	aaaa:6:65:0:aa54:18ff:fed4:e5f6	0a:aa:00:00:00:06	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	ICMPv6	90	Neighbor Advertise
22	6.000009	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	aaaa:6:65:0:aa54:18ff:fed4:e5f6	a8:54:18:d4:e5:f6	TCP	278	1025 → 80 [ACK] Se

Figura 2.3: Captura de paquetes entrantes y salientes de host[0]

```
** Event #391  t=2.860660434282  Network.host[0].ipv6.neighbourDiscovery (Ipv6NeighbourDiscovery, id=120)  on selfmsg dadTimeout
INFO: DAD Timeout message received
DETAIL: numOfDADMessagesSent is: 1
DETAIL: dupAddrDetectTrans is: 1
DETAIL: delete dadEntry and msg
INFO: creating router discovery message timer
```

Figura 2.4: Mensaje de fin de DAD para host[0]

```
1   lo0 ID:100 MTU:4470 UP LOOPBACK CARRIER macAddr:n/a Ipv6:{
2       Addrs:::1(loopback)  expiryTime: inf prefExpiryTime: inf
3       Node: dupAddrDetectTrans=1 reachableTime=36.455680949148
4   }
5   eth0 ID:101 MTU:1500 UP BROADCAST CARRIER MULTICAST macAddr:2C-8A-21-7A-8B-9C
    Ipv6:{
6       Addrs:fe80::2e8a:21ff:fe7a:8b9c(link tent)  expiryTime: inf prefExpiryTime:
    inf
7       mcastgrps:ff02::1   Node: dupAddrDetectTrans=1 reachableTime=40.327972322702
8   }
9
```

Figura 2.5: Tabla de interfaces de host[0] en t=0

```
1   lo0 ID:100 MTU:4470 UP LOOPBACK CARRIER macAddr:n/a Ipv6:{
2     Addrs:::1(loopback)  expiryTime: inf prefExpiryTime: inf
3     Node: dupAddrDetectTrans=1 reachableTime=36.455680949148
4   }
5
6   eth0 ID:101 MTU:1500 UP BROADCAST CARRIER MULTICAST macAddr:2C-8A-21-7A-8B-9C
    Ipv6:{
7   Addrs:fe80::2e8a:21ff:fe7a:8b9c(link)  expiryTime: inf prefExpiryTime: inf
8   mcastgrps:ff02::1   Node: dupAddrDetectTrans=1 reachableTime=40.327972322702
9   }
10
```

Figura 2.6: Tabla de interfaces de host[0] en t=3

2.5 Ejercicio 2.5

2.5.1 ¿En qué instante de la simulación obtienen los equipos sus direcciones IP globales? ¿Cómo obtienen esta última? Muestra la tabla de interfaces de nodo host[0] en la que se vea su estado antes y después de obtener la dirección global y explica qué cambia.

Respecto a la captura de paquetes de la simulación realizada anteriormente (Ver 2.3), los equipos obtienen sus direcciones IP globales tras recibir un paquete de Router Advertisement. Como el primero en enviar un Router Solicitation es host[1] (Paquete número 6), es también el primero (Y único) que puede procesar el RA siguiente, obteniendo su dirección global en t=2.02. Después, ocurre lo mismo con server_local, que obtiene su dirección en t=2.33, y con server_remote, que la obtiene en t=2.72 (No se ve en 2.3 porque server_remote está en otra red). Finalmente, y como se observa parcialmente en 2.7, host[0] y host[2] obtienen la dirección global en t=2.72. Ambos la obtienen casi a la vez porque ambos emiten el paquete de Router Solicitation casi a la vez, de forma que, aunque el router devuelve dos paquetes RA, ambos obtienen la dirección global ya al recibir el primero.
Como se ha explicado en parte anteriormente, el proceso de obtener la dirección global comienza con un mensaje Router Solicitation del equipo al router correspondiente, enviado a la dirección multicast del grupo de los routers en el enlace local (FF02:02). En respuesta, el router responde un Router Advertisement en el que se anuncia junto con el prefijo global de la red correspondiente, que es con el que los nodos formarán su dirección global, a la dirección multicast de los nodos de la LAN (FF02::01). Los equipos que ya hayan enviado un paquete de RS pueden procesarlo, mientras que los que no lo hayan enviado o ya tengan una dirección global establecida, no.
Ahora, observando la tabla de interfaces de host[0] en 2.9, observamos que una nueva dirección ha aparecido en la interfaz eth0, con la especificación *global*, que se ha formado uniendo el prefijo global del red proporcionado por el router y la dirección MAC del dispositivo tras aplicar el proceso EUI-64, y a la que se le establece un tiempo de caducidad tras el que el equipo deberá volver a enviar un RS al router continuar teniendo una dirección global. Esta dirección es la que permite al dispositivo identificarse de forma única en Internet.

```
** Event #501  t=3.967555297358  Network.host[0].ipv6.neighbourDiscovery (Ipv6NeighbourDiscovery, id=120)  on RApacket (inet::Packet
DETAIL: rdEntry is not nullptr, RD cancelled!
INFO: Interface is a host, processing RA.
INFO: Processing RA for Router Updates
INFO: Neighbour Cache Entry does not contain RA's source address
INFO: RA's router lifetime is non-zero, creating an entry in the Host's default router list with lifetime=1800
INFO: /// Removing default route for interface=101
INFO: RA's Cur Hop Limit is non-zero. Setting host's Cur Hop Limit to received value.
INFO: RA's reachable time is non-zero  and RA's and Host's reachable time differ,
INFO: setting host's base reachable time to received value.
INFO:
INFO: RA's retrans timer is non-zero, copying retrans timer variable.
INFO: Fetching Prefix Information: option 3 of 3
INFO: Processing Prefix Info for address auto-configuration.
INFO: Prefix not assigned to interface. Possible new router detected. Auto-configuring new address.
INFO: Assigning new address to: eth0
INFO (Ipv6RoutingTable)Network.host[0].ipv6.routingTable: ** Signal at T=3.967555297358 to Network.host[0].ipv6.routingTable: interf
INFO (Ipv6RoutingTable)Network.host[0].ipv6.routingTable:  Addrs:aaaa:6:65:0:2e8a:21ff:fe7a:8b9c(global)  expiryTime: 2592003.967555
INFO (Ipv6RoutingTable)Network.host[0].ipv6.routingTable:              , fe80::2e8a:21ff:fe7a:8b9c(link)  expiryTime: inf prefExpiry
INFO (Ipv6RoutingTable)Network.host[0].ipv6.routingTable:  mcastgrps:ff02::1  Node: dupAddrDetectTrans=1 reachableTime=4676.97082981
INFO (Ipv6RoutingTable)Network.host[0].ipv6.routingTable:    }
INFO (Ipv6RoutingTable)Network.host[0].ipv6.routingTable:  changed field: 0
INFO (Ipv6RoutingTable)Network.host[0].ipv6.routingTable:
```

Figura 2.7: Evento de obtención de dirección global para host[0]

```
1        lo0 ID:100 MTU:4470 UP LOOPBACK CARRIER macAddr:n/a Ipv6:{
2            Addrs:::1(loopback)  expiryTime: inf prefExpiryTime: inf
3            Node: dupAddrDetectTrans=1 reachableTime=36.455680949148
4        }
5        eth0 ID:101 MTU:1500 UP BROADCAST CARRIER MULTICAST
     macAddr:2C-8A-21-7A-8B-9C Ipv6:{
6            Addrs:fe80::2e8a:21ff:fe7a:8b9c(link)  expiryTime: inf prefExpiryTime:
     inf
7            mcastgrps:ff02::1   Node: dupAddrDetectTrans=1
     reachableTime=40.327972322702
8        }
9
```

Figura 2.8: Tabla de interfaces de host[0] en t=3

```
1        lo0 ID:100 MTU:4470 UP LOOPBACK CARRIER macAddr:n/a Ipv6:{
2            Addrs:::1(loopback)  expiryTime: inf prefExpiryTime: inf
3                Node: dupAddrDetectTrans=1 reachableTime=36.455680949148
4        }
5        eth0 ID:101 MTU:1500 UP BROADCAST CARRIER MULTICAST
     macAddr:2C-8A-21-7A-8B-9C Ipv6:{
6            Addrs:aaaa:6:65:0:2e8a:21ff:fe7a:8b9c(global)  expiryTime:
     2592003.967555297358 prefExpiryTime: 604803.967555297358,
7            fe80::2e8a:21ff:fe7a:8b9c(link)  expiryTime: inf prefExpiryTime: inf
8            mcastgrps:ff02::1   Node: dupAddrDetectTrans=1
     reachableTime=4609.905032347888
9        }
10
```

Figura 2.9: Tabla de interfaces de host[0] en t=4

2.6 Ejercicio 2.6

2.6.1 Explica cómo se construye la IP global usando el nodo host[0] como ejemplo, de nuevo usando la notación IPv6 no abreviada

Una vez que el equipo manda un mensaje ICMPv6 con mensaje NS (Neighbor Solicitation) y no ocurre ningún fallo (No hay conflicto de IPs), procede a mandar un paquete ICMPv6 con mensaje RS (Router Solicitation), para que el router le asigne una IPv6 global (Este tipo de IPs ya son enrutables y únicas a nivel global). El router contesta con un paquete ICMPv6 con mensaje RA (Router Advertisement) indicando el prefijo global de red (Los primeros 64 bits de la dirección de la red al que está conectada host[0], en este caso AAAA:0006:0065:0000). Una vez que el host[0] obtiene el prefijo de la red, construye su IPv6 global substituyendo el prefijo de red del enlace local por la de la red, quedando su dirección como: AAAA:0006:0065:0000:2E8A:21FF:FE7A:8B9C. Esto se puede ver en la imagen 2.10.

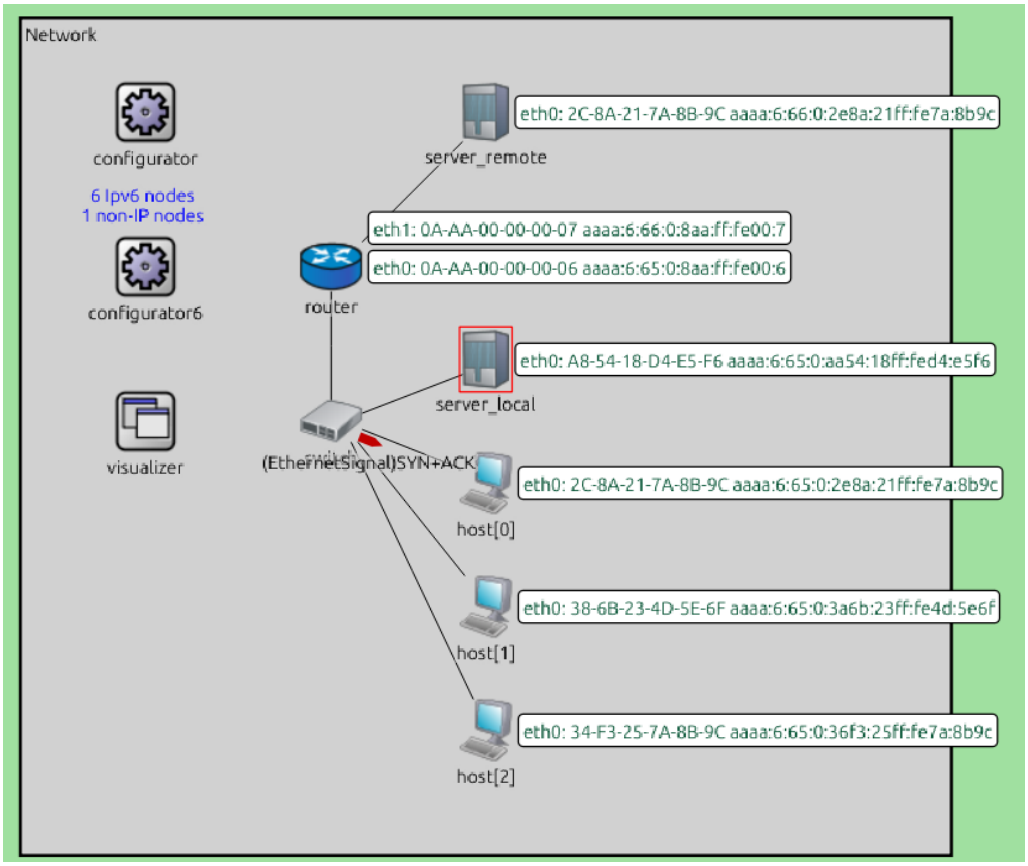


Figura 2.10: Foto red con direcciones IPv6 globales asignadas

2.7 Ejercicio 2.7

2.7.1 Configura host[0] para que se conecte al servidor server remote usando su dirección fe80::x:x:x:x: (asegúrate de que la dirección MAC de server remote es única). ¿Qué ocurre? Repite lo mismo para server local. ¿Qué ocurre?

Para realizar este ejercicio, se aplicaron los cambios descritos en la figura 2.11 sobre la TcpBasicClientApplication de host[0], de forma que se conectara a los servidores directamente a través de su dirección de enlace local. En primer lugar, probamos la conexión a server_remote. Como se observa en la figura 2.12, host[0] no recibe el ACK correspondiente a ninguno de sus paquetes de intento de conexión. La primera vez que intenta la conexión, no recibe respuesta, por lo que lo intenta de nuevo una y otra vez enviando TCP Retransmisions. Esto es debido a que server_remote se encuentra en una LAN distinta a host[0], que intenta alcanzar su dirección de enlace local. Los routers en IPv6 no enrutan paquetes como lo hacen en IPv4 entre redes, por lo que la única forma que tendría host[0] de completar la conexión sería usando la dirección unicast global del servidor.
Por otra parte, descubrimos que al realizar la conexión con server_local, ocurre lo mismo (2.13). Esto es debido a un comportamiento propio de INET, que limpia de la tabla de enrutamiento la dirección local de enlace (2.14) que permite conectarse a otros dispositivos en el mismo segmento de red después de recibir el prefijo de red del router (2.15). En la realidad, la conexión al servidor local sí debería realizarse con éxito, ya que está en la misma LAN que host[0] y debería poder descubrirlo en el proceso de Neighbor Discovery.

```
*.host[0].numApps = 2
*.host[0].app[0].typename = "TcpBasicClientApp"
*.host[0].app[0].connectAddress = "fe80::9eff:19ff:fea7:b8c9"   #server_remote
#*.host[0].app[0].connectAddress = "fe80::aa54:18ff:fed4:e5f6"   server_local
*.host[0].app[0].connectPort = 80
*.host[0].app[0].startTime = 6s
*.host[0].app[0].idleInterval = 4s
*.host[0].app[0].replyLength = 1000B
*.host[0].app[0].thinkTime = 0s
```

Figura 2.11: Configuración establecida para este ejercicio

	Time	Source	MAC src	Destination	MAC dest	Protocol	Length Info
1	0.237519	::	0a:aa:00:00:00:06	ff02::1:ff00:6	ff:ff:ff:ff:ff:ff	ICMPv6	82 Neighbor Solicitation
2	0.434029	::	38:6b:23:4d:5e:6f	ff02::1:ff4d:5e6f	ff:ff:ff:ff:ff:ff	ICMPv6	82 Neighbor Solicitation
3	0.563595	::	a8:54:18:d4:e5:f6	ff02::1:ffd4:e5f6	ff:ff:ff:ff:ff:ff	ICMPv6	82 Neighbor Solicitation
4	0.935064	::	2c:8a:21:7a:8b:9c	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	82 Neighbor Solicitation
5	0.978199	::	34:f3:25:7a:8b:9c	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	82 Neighbor Solicitation
6	1.985042	fe80::3a6b:23ff:fe4d:5e6f	38:6b:23:4d:5e:6f	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74 Router Solicitation f
7	2.028609	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122 Router Advertisement
8	2.151858	fe80::aa54:18ff:fed4:e5f6	a8:54:18:d4:e5:f6	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74 Router Solicitation f
9	2.335981	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122 Router Advertisement
10	3.684289	fe80::36f3:25ff:fe7a:8b9c	34:f3:25:7a:8b:9c	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74 Router Solicitation f
11	3.730748	fe80::2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74 Router Solicitation f
12	3.967555	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122 Router Advertisement
13	4.173601	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122 Router Advertisement
14	6.000000	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	fe80::9eff:19ff:fea7:b8c9	0a:aa:00:00:00:06	TCP	82 1025 → 80 [SYN] Seq=0
15	9.000000	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	fe80::9eff:19ff:fea7:b8c9	0a:aa:00:00:00:06	TCP	82 [TCP Retransmission]
16	11.0000…	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	fe80::8aa:ff:fe00:6	ff:ff:ff:ff:ff:ff	ICMPv6	82 Neighbor Solicitation
17	11.0000…	aaaa:6:65:0:8aa:ff:fe00:6	0a:aa:00:00:00:06	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	90 Neighbor Advertisement
18	15.0000…	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	fe80::9eff:19ff:fea7:b8c9	0a:aa:00:00:00:06	TCP	82 [TCP Retransmission]
19	27.0000…	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	fe80::9eff:19ff:fea7:b8c9	0a:aa:00:00:00:06	TCP	82 [TCP Retransmission]
20	51.0000…	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	fe80::9eff:19ff:fea7:b8c9	0a:aa:00:00:00:06	TCP	82 [TCP Retransmission]

Figura 2.12: Captura de paquetes en la conexión host[0] a server_remote

	Time	Source	MAC src	Destination	MAC dest	Protocol	Length Info
1	0.237519	::	0a:aa:00:00:00:06	ff02::1:ff00:6	ff:ff:ff:ff:ff:ff	ICMPv6	82 Neighbor Solicitation
2	0.434029	::	38:6b:23:4d:5e:6f	ff02::1:ff4d:5e6f	ff:ff:ff:ff:ff:ff	ICMPv6	82 Neighbor Solicitation
3	0.563595	::	a8:54:18:d4:e5:f6	ff02::1:ffd4:e5f6	ff:ff:ff:ff:ff:ff	ICMPv6	82 Neighbor Solicitation
4	0.935064	::	2c:8a:21:7a:8b:9c	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	82 Neighbor Solicitation
5	0.978199	::	34:f3:25:7a:8b:9c	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	82 Neighbor Solicitation
6	1.985042	fe80::3a6b:23ff:fe4d:5e6f	38:6b:23:4d:5e:6f	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74 Router Solicitation f
7	2.028609	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122 Router Advertisement
8	2.151858	fe80::aa54:18ff:fed4:e5f6	a8:54:18:d4:e5:f6	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74 Router Solicitation f
9	2.335981	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122 Router Advertisement
10	3.684289	fe80::36f3:25ff:fe7a:8b9c	34:f3:25:7a:8b:9c	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74 Router Solicitation f
11	3.730748	fe80::2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74 Router Solicitation f
12	3.967555	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122 Router Advertisement
13	4.173601	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122 Router Advertisement
14	6.000000	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	fe80::aa54:18ff:fed4:e5f6	0a:aa:00:00:00:06	TCP	82 1025 → 80 [SYN] Seq=0
15	9.000000	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	fe80::aa54:18ff:fed4:e5f6	0a:aa:00:00:00:06	TCP	82 [TCP Retransmission]
16	11.0000…	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	fe80::8aa:ff:fe00:6	ff:ff:ff:ff:ff:ff	ICMPv6	82 Neighbor Solicitation
17	11.0000…	aaaa:6:65:0:8aa:ff:fe00:6	0a:aa:00:00:00:06	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	90 Neighbor Advertisement
18	15.0000…	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	fe80::aa54:18ff:fed4:e5f6	0a:aa:00:00:00:06	TCP	82 [TCP Retransmission]
19	27.0000…	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	fe80::aa54:18ff:fed4:e5f6	0a:aa:00:00:00:06	TCP	82 [TCP Retransmission]

Figura 2.13: Captura de paquetes en la conexión host[0] a server_local

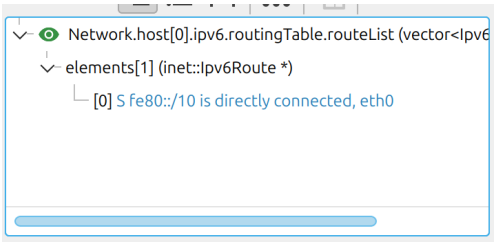


Figura 2.14: Routing table de host[0] al inicio de la simulación

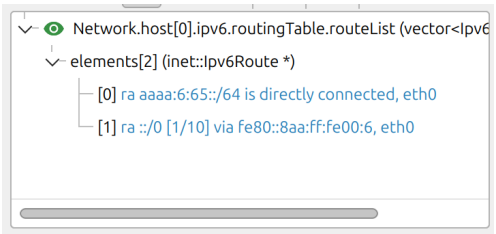


Figura 2.15: Routing table de host[0] tras obtener la dirección unicast global

Capítulo 3

IPv6-Multicast

3.1 Ejercicio 3.1

3.1.1 En los primeros 2 segundos de simulación se envían varios paquetes NS. Muestra una captura del tráfico de paquetes en Wireshark en la que se vean las direcciones IP y MAC origen y destino de los NS enviados desde cada uno de los nodos host[*]. (Nota: para mostrar las direcciones MAC añade dos nuevas columnas: Hw src addr (unresolved) y Hw dest addr (unresolved). Colócalas a la derecha de las direcciones IP.) ¿De qué tipo son las direcciones IP? ¿Cómo se construyen?

	Time	Source	MAC src	Destination	MAC dest	Protocol	Length	Info
1	0.000000	::	0a:aa:00:00:00:06	ff02::1:ff00:6	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor Soli
2	0.196510	::	38:6b:23:4d:5e:6f	ff02::1:ff4d:5e6f	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor Soli
3	0.326076	::	a8:54:18:d4:e5:f6	ff02::1:ffd4:e5f6	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor Soli
4	0.697545	::	2c:8a:21:7a:8b:9c	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor Soli
5	0.740680	::	34:f3:25:7a:8b:9c	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor Soli
6	1.747523	fe80::3a6b:23ff:fe4…	38:6b:23:4d:5e:6f	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Solici
7	1.791090	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Advert
8	1.914339	fe80::aa54:18ff:fed…	a8:54:18:d4:e5:f6	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Solici
9	2.098462	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Advert
10	3.446770	fe80::36f3:25ff:fe7…	34:f3:25:7a:8b:9c	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Solici
11	3.493229	fe80::2e8a:21ff:fe7…	2c:8a:21:7a:8b:9c	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Solici
12	3.730036	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Advert

Figura 3.1: Captura MAC origen y destino host[0]

Como las direcciones IPs de destino son multicast, en todos los hosts se ve la misma paquetería, por lo que solo subimos la captura de tráfico en el host[0].

Las direcciones, como hemos mencionado anteriormente, son multicast y se construyen con el prefijo multicast link-local (FF02::1, donde FF02 quiere decir que es tipo multicast, siendo el 2 un scope de local de enlace y el último 1 quiere decir todos los dispositivos. Si fuera un 2 serían todos los routers). Por último, tendríamos como sufijo, los 3 bytes menos significativos de la MAC del dispositivo de ese host precedido de FF.

Por ejemplo, en el primer caso tenemos como destino FF02::1:FF00:0006 donde F002::1 es, como explicamos antes, el prefijo multicast link-local y como sufijo FF00:0006, donde 000006 son los 3 últimos bytes de la MAC de la interfaz del router en ese enlace (0A:AA:00:00:00:06).

3.2 Ejercicio 3.2

3.2.1 ¿Coincide alguna de las direcciones IP destino de los diferentes paquetes NS? ¿Por qué? ¿Qué consecuencia tiene esto?

Coinciden las direcciones IP del host[0] y host[2], ya que los dos tienen los 3 últimos bytes de la MAC iguales por lo que la dirección multicast se construye de la misma forma.

Con respecto a las consecuencias que se puede tener al haber dos direcciones multicast iguales, no hay ninguna, ya que los mensajes ICMPv6 de tipo NS (Neighbor Solicitation) guardan en su cabecera la dirección unicast de destino, por lo que no se daría ningún conflicto (target address). Aunque no pase nada, lo normal es que las direcciones multicast de nodo solicitado (FF02::1) sean únicas.

3.3 Ejercicio 3.3

3.3.1 ¿Por qué se envían los NS a esas direcciones IP?

Se envían ICMPv6 de tipo NS a esas direcciones ya que haciéndolo con direcciones multicast, se puede mandar un único paquete a uno o varios destinos, de esta forma optimizamos la comunicación y reducimos el tráfico de la red para que no se congestione por lo que aumenta la eficiencia.

3.4 Ejercicio 3.4

3.4.1 ¿Qué direcciones MAC destino tienen los paquetes NS anteriores? ¿Cuáles deberían tener según lo visto en clases de teoría? (Utiliza el paquete NS que sale desde host[1] como ejemplo y escribe los 6 bytes que debería tener la dirección MAC en formato hexadecimal.)

No.	Time	Source	MAC src	Destination	MAC dest	Protocol	Length	Info
1	0.000000	::	0a:aa:00:00:00:06	ff02::1:ff00:6	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor So
2	0.196509	::	38:6b:23:4d:5e:6f	ff02::1:ff4d:5e6f	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor So
3	0.326076	::	a8:54:18:d4:e5:f6	ff02::1:ffd4:e5f6	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor So
4	0.697546	::	2c:8a:21:7a:8b:9c	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor So
5	0.740680	::	34:f3:25:7a:8b:9c	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor So
6	1.747522	fe80::3a6b:23ff:fe4…	38:6b:23:4d:5e:6f	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Soli
7	1.791090	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Adve
8	1.914339	fe80::aa54:18ff:fed…	a8:54:18:d4:e5:f6	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Soli
9	2.098462	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Adve
10	3.446770	fe80::36f3:25ff:fe7…	34:f3:25:7a:8b:9c	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Soli

Figura 3.2: Captura MAC origen y destino paquetes NS en host[1]

Como se puede observar en la imagen 3.2, todos los paquetes tienen como MAC destino FF:FF:FF:FF:FF:FF. Según se ha visto en la clase de teoría, las MAC tienen la estructura 33:33:FF como prefijo de la MAC y, después, los 3 últimos bytes son los 3 bytes menos significativos de la dirección multicast. En este caso, el destino es la dirección MAC broadcast, ya que Omnett no sabe como construir este tipo de direcciones MAC como se explica en la teoría, por eso todos los hosts reciben la misma paquetería. De la otra forma, cada uno recibiría su propia paquetería.

3.5 Ejercicio 3.5

3.5.1 ¿Qué consecuencia tiene esta diferencia con respecto a lo visto en clase de teoría?

Resulta que las consecuencias de usar dirección MAC broadcast es que todo el mundo recibe paquetes que en realidad no tendrían que recibir por lo que se sobrecarga la red. Usando la dirección MAC que se especifica en la teoría, solamente se mandaría el paquete a ese host.

3.6 Ejercicio 3.6

3.6.1 Muestra las direcciones IP y MAC destino de los mensajes RS y RA que aparecen en torno al segundo 2 de simulación. ¿De qué tipo son las direcciones IP?

No.	Time	Source	MAC src	Destination	MAC dest	Protocol	Length	Info
4	0.935064	::	2c:8a:21:7a:8b:9c	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor Solicitation
5	0.978199	::	34:f3:25:7a:8b:9c	ff02::1:ff7a:8b9c	ff:ff:ff:ff:ff:ff	ICMPv6	82	Neighbor Solicitation
6	1.985042	fe80::3a6b:23ff:fe4d:5e6f	38:6b:23:4d:5e:6f	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Solicitation f
7	2.028609	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Advertisement
8	2.151858	fe80::aa54:18ff:fed4:e5f6	a8:54:18:d4:e5:f6	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Solicitation f
9	2.335981	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Advertisement
10	3.684289	fe80::36f3:25ff:fe7a:8b9c	34:f3:25:7a:8b:9c	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Solicitation f
11	3.730748	fe80::2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	ff02::2	ff:ff:ff:ff:ff:ff	ICMPv6	74	Router Solicitation f
12	3.967555	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Advertisement
13	4.173601	fe80::8aa:ff:fe00:6	0a:aa:00:00:00:06	ff02::1	ff:ff:ff:ff:ff:ff	ICMPv6	122	Router Advertisement
14	6.000000	aaaa:6:65:0:2e8a:21ff:fe7a:8b9c	2c:8a:21:7a:8b:9c	aaaa:6:66:0:9eff:19ff:fea7:b8c9	0a:aa:00:00:00:06	TCP	82	1025 → 80 [SYN] Seq=0

Figura 3.3: Captura paquetes RS y RA host[0]

Como se puede observar en la imagen 3.3, en el caso de los paquetes ICPMv6 con mensaje RS tienen como dirección destino la dirección multicast FF02::2 y como MAC de destino, la dirección FF:FF:FF:FF:FF:FF. En este caso, las direcciones multicast son de este tipo ya que son direcciones a las que van dirigidos los paquetes a todos los routers de la misma red de en enlace para así optimizar el proceso de descubrimiento de routers y no malgastar ancho de banda.

En el caso de los paquetes ICMPv6 con mensaje RA, estos tienen como dirección destino la dirección multicast FF02::1 y como MAC de destino la dirección FF:FF:FF:FF:FF:FF. En este caso, la dirección multicast tiene esta estructura ya que va dirigido a todo dispositivo que está en el mismo enlace de red.

3.7 Ejercicio 3.7

3.7.1 ¿Por qué el RA de respuesta a un RS no usa IP destino unicast?

Este enfoque permite enviar la información de configuración a todos los dispositivos de la red sin necesidad de identificar la dirección específica de cada uno. Si se utilizara una dirección unicast, sería necesario conocer la dirección única de cada dispositivo, lo que resultaría poco práctico en redes grandes o cuando los dispositivos conectados pueden cambiar con frecuencia. Así, de esta forma, el tráfico no se sobrecarga y resultaría todo mas eficiente.

3.8 Ejercicio 3.8

3.8.1 ¿Qué direcciones MAC destino deberían tener los RS según lo visto en clases de teoría? ¿Y los RA?

La MAC destino del paquete RS debería ser 33:33:00:00:00:02, asegurando que solo los routers reciban las solicitudes de descubrimiento de nodos, y los paquetes RA deberían tener como MAC destino 33:33:00:00:00:01, por lo que así sería captado por las interfaces Ethernet de todos los dispositivos IPv6 de esa misma red.

3.9 Ejercicio 3.9

3.9.1 Aproximadamente en t = 6 s el router envía dos NS. ¿De qué tipo son las direcciones IP destino de estos paquetes? Explica cómo se construyen (notación IPv6 no abreviada).

En IPv6, las direcciones de destino de los mensajes de Neighbor Solicitation son direcciones multicast de nodo solicitado. Estas direcciones se utilizan en el protocolo Neighbor Discovery para resolver la dirección MAC correspondiente a una dirección IPv6 de destino específica en la red local o para anunciar la presencia del emisor del paquete en el proceso de Duplicate Address Detection (El caso estudiado en este ejercicio es el especificado en primer lugar). Cada dirección unicast IPv6 tiene asociada una dirección multicast de nodo solicitado, las cuales, a su vez, se corresponden siempre con una dirección multicast Ethernet.
Las direcciones multicast de nodo solicitado se forman añadiendo, tras el prefijo
FF02::1:FF00:0000/104 (ff02:0000:0000:0000:0000:0001:ffXX:XXXX), los 24 bits menos significativos de la dirección unicast correspondiente.
En nuestro caso, como se observa en el paquete 15 de la figura 3.4, la dirección de destino es, siguiendo la notación IPv6 no abreviada, ff02:0000:0000:0000:0000:0001:ff7a:8b9c. De esta forma, el paquete será procesado solo por aquellos equipos cuya dirección unicast termine por 7a:8b9c, aunque sea enviado a todos los nodos en la red del destinatario.

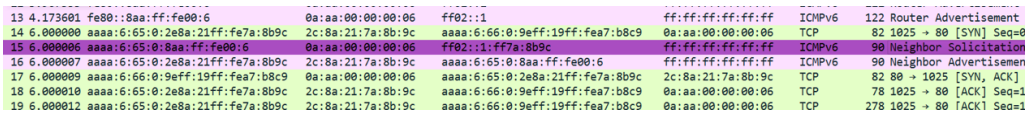


Figura 3.4: Intercambio de paquetes en t=6 entre el router y host[0]

3.10 Ejercicio 3.10

3.10.1 ¿En qué equipos llega el segundo paquete NS enviado por el router al módulo ipv6? ¿Y al submódulo ipv6.neighborDiscovery? ¿Por qué? (Nota: para ver el recorrido del paquete en el módulo ipv6, haz doble click en el nodo deseado y luego en el módulo ipv6. Puedes mostrar varios nodos a la vez en diferentes ventanas con botón derecho → Open Graphical View for 'ipv6' una vez dentro de ese nivel.)

En primer lugar, cabe destacar que el segundo paquete NS al que nos referiremos en este ejercicio es el que aparece en la figura 3.4, es decir, el enviado del router al switch de la LAN a la que está host[0] conectado. Esto se sabe, porque, como se aprecia en la figura 3.5, el primer paquete (Evento 569) es enviado hacia el servidor remoto;

mientras que el evento 605 refleja el envío de ese segundo paquete hacia la dirección multicast de nodo solicitado de host[0].

El paquete NS es primero enviado al switch, que, a su vez, lo reenvía a todos los hosts y a server_local. Esto se debe a que INET no implementa correctamente direcciones MAC multicast, por lo que la dirección de nodo solicitado de destino equivale a la dirección broadcast de capa 2, ff:ff:ff:ff:ff:ff (Ver 3.4). Podemos ver en detalle en las figuras 3.6 y 3.7, que el paquete entra al módulo IPv6, tanto de host[0] como de host[1], que realmente no es el objetivo de esta transmisión (De igual forma ocurre en server_local y host[2]).

Por otra parte, se puede ver en la comparación representada en la figura 3.8 entre host[0] y host[1], el paquete solo pasa al módulo neighborDiscovery en el caso de host[0], ya que es previamente procesado en el módulo IPv6, donde sí puede leerse su dirección destino de nodo solicitado. Sin embargo, debido a que host[0] y host[2] tienen los últimos 3 bytes de su MAC, y por lo tanto, de su dirección multicast de nodo solicitado, idénticos, el paquete es recibido también en el módulo IPv6 de host[2] (Ver 3.9).

En resumen, el paquete NS alcanza el módulo IPv6 de host[0], host[1], host[2] y server_local y el módulo neighborDiscovery de host[0] y host[2], únicamente.

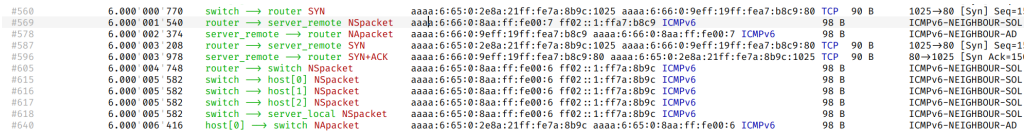


Figura 3.5: Paquetes enviados por el router en t=6 (Captura de log)

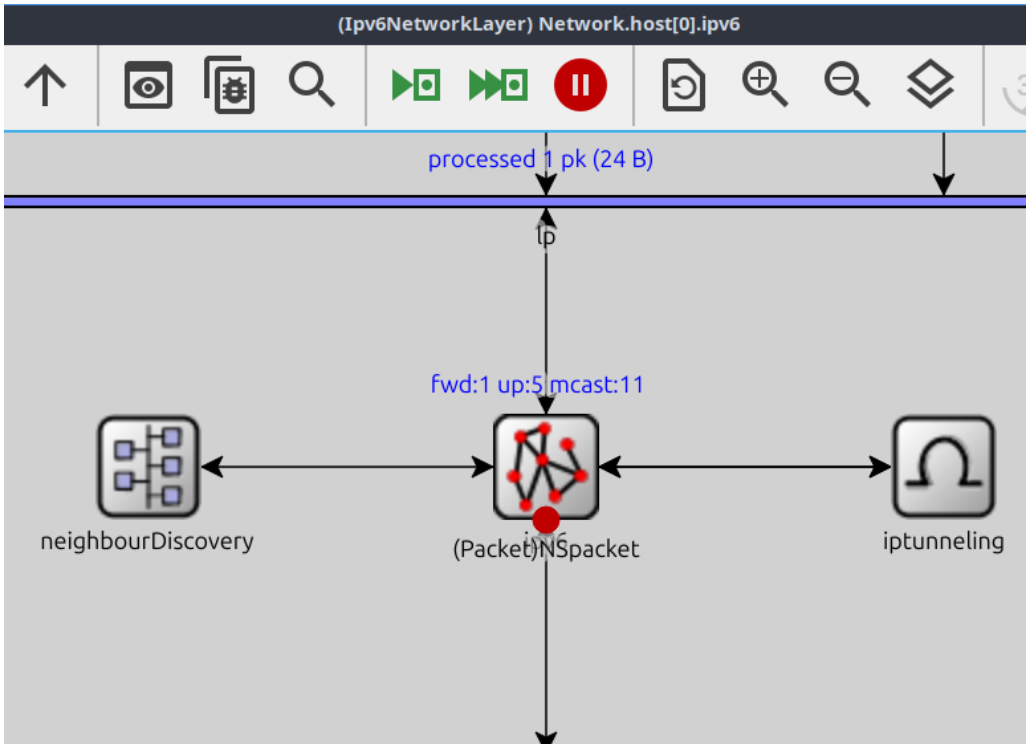


Figura 3.6: Paquete de NS entrando al módulo IPv6 de host[0]

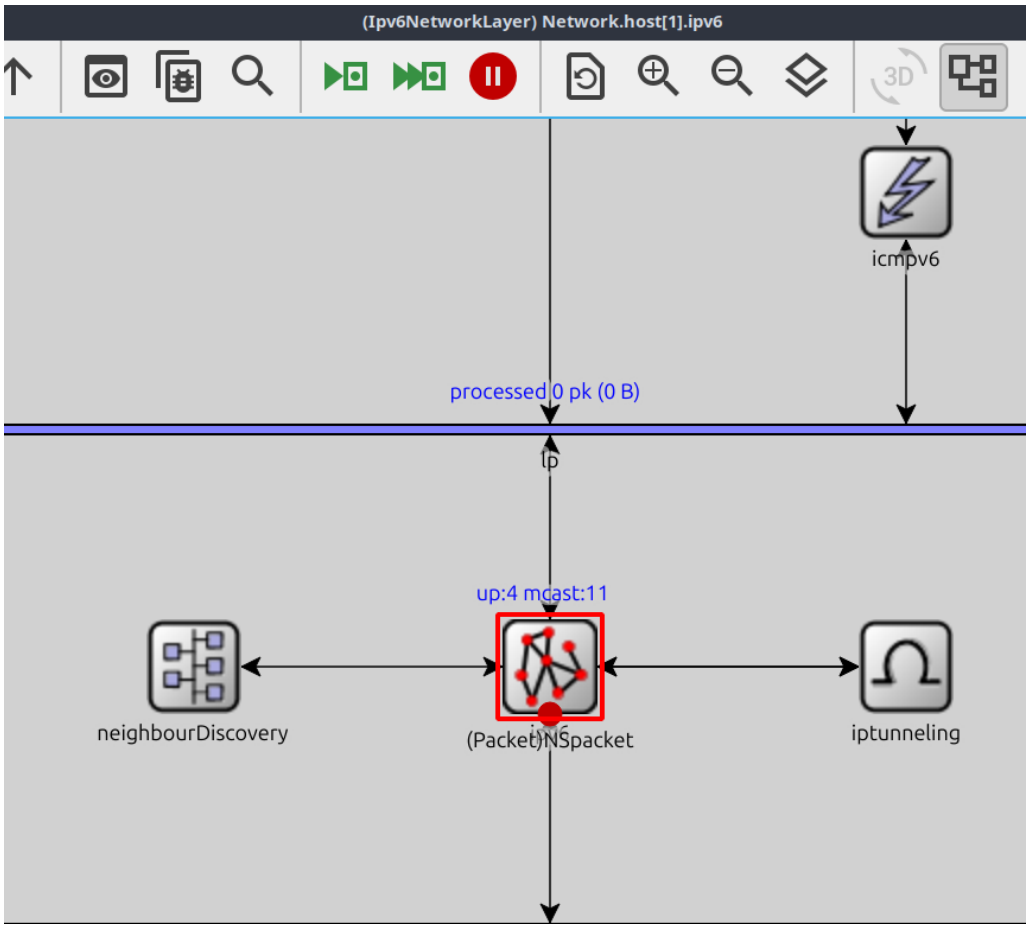


Figura 3.7: Paquete de NS entrando al módulo IPv6 de host[1]

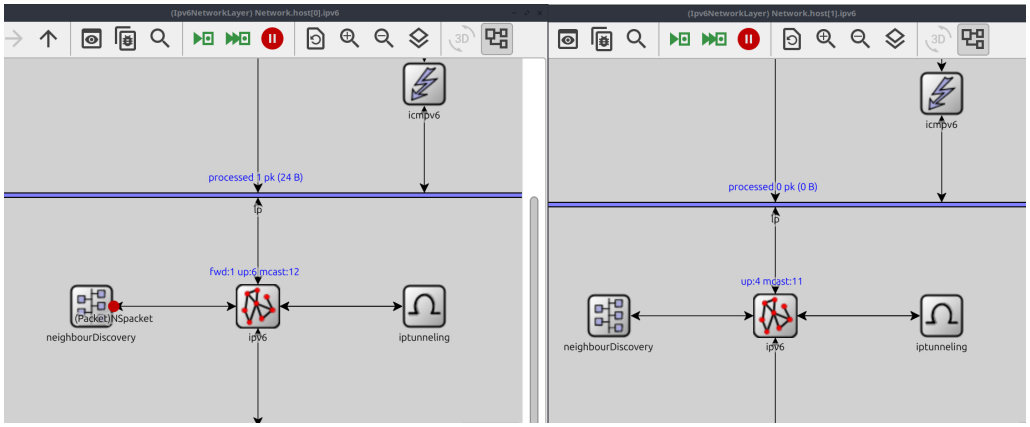


Figura 3.8: Comparación entre host[0] y host[1] de paquetes entrantes al módulo neighborDiscovery

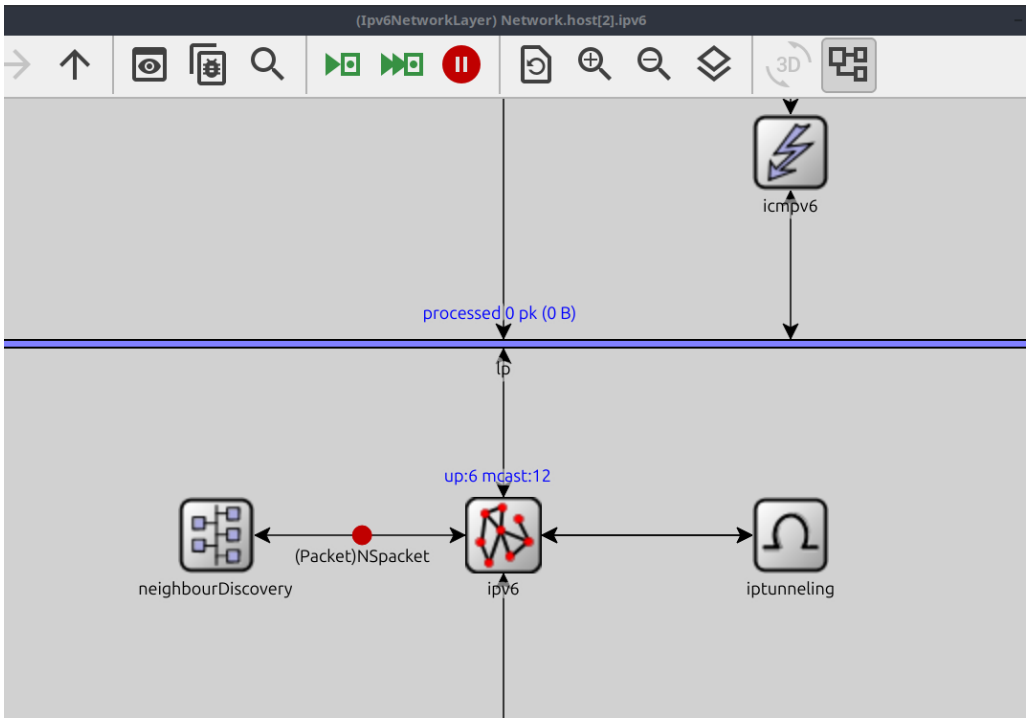


Figura 3.9: Paquete de NS entrando al módulo neighborDiscovery en host[2]

3.11 Ejercicio 3.11

3.11.1 ¿En qué equipos llegaría el mensaje al módulo ipv6 si INET implementase direcciones MAC multicast (33- 33-xx-xx-xx-xx)? ¿Por qué?

Si INET implementase este tipo de direcciones, el mensaje llegaría al modulo IPv6 (Prefijo 33-33) de los equipos que tengan asignada la dirección MAC 33-33-xx-xx-xx-xx, siendo los últimos 32 bits iguales a los últimos 32 de la dirección multicast de nodo solicitado del equipo. O lo que es lo mismo, el prefijo FF seguido de los últimos 24 bits de la dirección unicast correspondiente a la de nodo solicitado.
Como la configuración actual asigna a host[0] y host[2] direcciones MAC que coinciden en los últimos 3 bytes, el mensaje NS al que nos referimos, con destino 33-33-FF-7A-8B-9C, sería recibido por los módulos IPv6 de los equipos host[0] y host[2] únicamente y no a todos los equipos actualmente existentes en la LAN, como ocurre en la situación estudiada en el ejercicio anterior.

Capítulo 4

IPv6-Neighbor Discovery

4.1 Ejercicio 4.1

4.1.1 ¿Por qué es necesario el NS que se envía en t = 6 s? Muestra una captura del log del nodo que lo envía que muestre el motivo del envío. ¿Qué paquete cumple la misma función en IPv4?

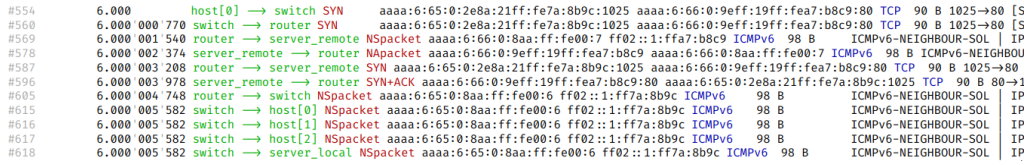


Figura 4.1: Ventana logs paquete NS en t=6s

Como se puede ver en la imagen 4.1, el nodo que envia el paquete Neighbor Solicitation en t=6s es el router. El host[0] empieza la conexión mandando un SYN al servidor remoto pero como no tiene información de su MAC, el router manda al servidor un NS para asi averiguar su MAC ya que solamente le llega una IP pero no sabe a que dispositivo le corresponde. Esto ocurre ya que como se mencinó antes, al no existir una entrada en la Neighbor Cache para el serverRemote, el dispositivo debe enviar un mensaje NS para solicitar esa información a través del protocolo Neighbor Discovery.

En el caso de IPv4, el mensaje que realiza una función similar al NS (Neighbor Solicitation) de IPv6 es el mensaje de solicitud ARP (ARP Request). Tanto el NS en IPv6 como la solicitud ARP en IPv4 se utilizan para descubrir la dirección física (de la capa de enlace) que corresponde a una dirección IP o de red específica.

4.2 Ejercicio 4.2

4.2.1 ¿Por qué no se usa una IP unicast para ese mensaje, si ya es conocida?

No se utiliza la dirección unicast aunque sea conocida ya que en IPv6 se emplea una dirección de multicast especial para la resolución de direcciones. Esto se hace para evitar enviar el mensaje a todos los dispositivos de la red. Esta dirección multicast es escuchada solo por el dispositivo que posee la dirección de destino, limitando el tráfico de resolución de direcciones a los dispositivos interesados y asi no sobrecargar la red, ganando asi eficiencia y recursos.

4.3 Ejercicio 4.3

4.3.1 Muestra capturas de la neighbor cache del nodo que envió el NS un segundo antes de enviarlo, justo después de enviarlo pero antes de recibir el paquete de respuesta y justo después de recibir la respuesta y explica las diferencias entre los 3 estados. (Nota: para ver la neighbor cache haz doble click sobre el nodo → ipv6 → neighborDiscovery, y a continuación expande owned objects en la ventana inferior izquierda. La neighbor cache es el atributo neighbourMap.)

1. Un segundo antes de enviar el NS

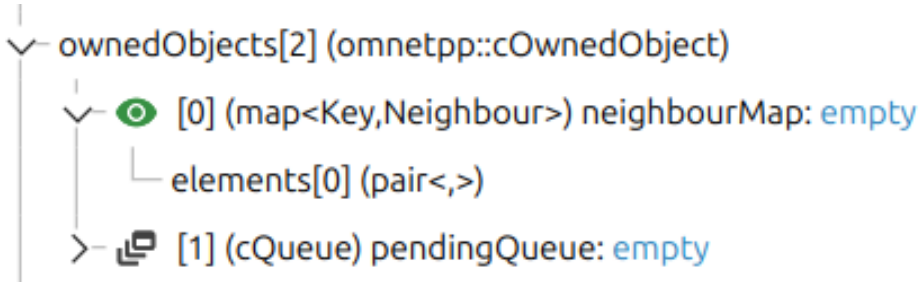


Figura 4.2: Neighbor cache del router antes de enviar NS

En este momento, la neighbor cache del router está vacía ya que en ningún momento tuvo que mandar un paquete IPv6 al servidor remoto, por lo que no necesitó nunca guardar su dirección MAC.

2. Justo después de enviar el paquete NS

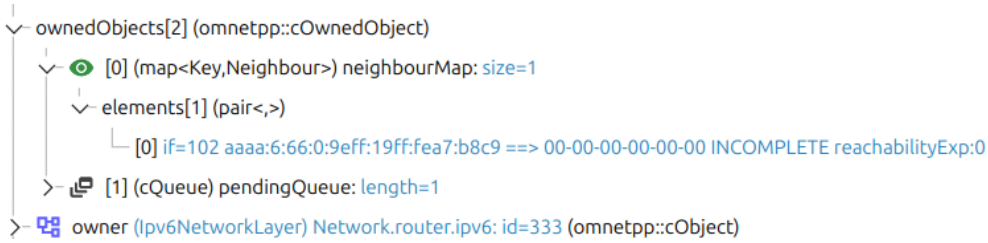


Figura 4.3: Neighbor cache del router después enviar NS

Como se puede ver en la imagen 4.3, el estado de la entrada de la neighbor cache del router tras haber enviado un mensaje NS ya que al querer enrutar el paquete SYN que envió el host[0], tiene que averiguar la MAC de la IP destino, es el estado INCOMPLETE ya que esta a la espera de la respuesta de la MAC del servidor. En la entrada tiene como destino la IP global del servidor remoto y la MAC asociada es la 00:00:00:00:00:00 ya que no la sabe

3. Un segundo después de enviar el paquete NS

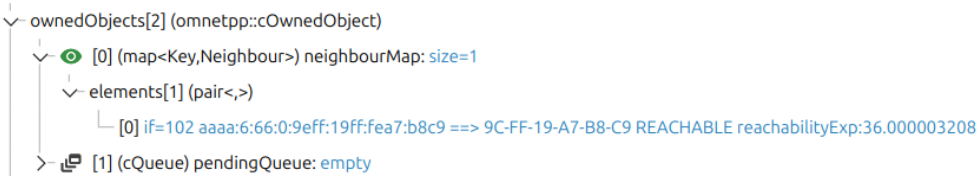


Figura 4.4: Neighbor cache del router al recibir respuesta del paquete NS

Como se puede ver en la imagen 4.4, el estado de la entrada de la cache del router pasa a ser REACHABLE ya que se recibió la respuesta indicando que es una dirección alcanzable. También se puede ver que en la entrada de la caché ya aparece la MAC del servidor mapeada con su dirección IPv6

4.4 Ejercicio 4.4

4.4.1 ¿Envía el host[1] algún otro mensaje NS después del instante t = 6 s? ¿Cuál es su objetivo? Muestra una captura del mensaje en Wireshark y una captura del log en la que se muestre el motivo del envío.

En la figura 4.5, se observa que host[0] vuelve a enviar un paquete de NS en t = 11 (También se repite en t = 47, y así sucesivamente). El motivo del envío, es la finalización del NUD (Neighbor Unreachability Detection), como refleja la figura 4.6. Este timeout define el tiempo que pasa en DELAY la entrada de la dirección unicast del router en la tabla de caché de neighborDiscovery de host[0] (Normalmente 5s).

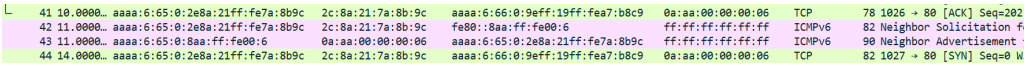


Figura 4.5: Intercambio de NS y NA entre host[0] y Router

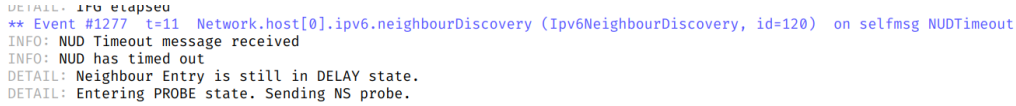


Figura 4.6: Motivo del envío del Ns

4.5 Ejercicio 4.5

4.5.1 ¿Es la IP destino de este NS del mismo tipo que en los NS enviados anteriormente? ¿Por qué?

La IP de destino en el NS en t=11 está en formato unicast link-local, diferenciándose este paquete de los enviados anteriormente en la simulación, que se enviaban a multicast de nodo solicitado. Esto, es debido a que ahora host[0] sí conoce la dirección del router, simplemente quiere saber si está activo y en la misma dirección que host[0] conoce. Las direcciones multicast se necesitan cuando se quiere enviar un mensaje a varios nodos, que no es el caso en esta situación.

4.6 Ejercicio 4.6

4.6.1 Muestra capturas de la neighbor cache del host[1] en los siguientes instantes de tiempo:

1. 7 segundos antes del envío del NS de la pregunta anterior.

Figura 4.7: Estado de la tabla en t=4 (La ruta al router comienza en estado STALE)

2. 3 segundos antes del envío del NS.

Figura 4.8: Estado de la tabla en t=10 (La ruta pasa a estado DELAY hasta que pasen 5 segundos o se reciba un paquete de esa dirección)

3. Justo después del envío del NS y antes de recibir el NA respuesta.

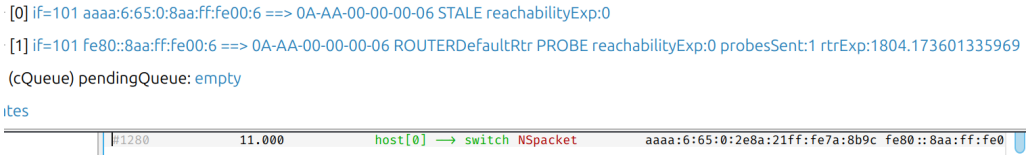


Figura 4.9: Estado de la tabla en t=11, tras el envío del NS (La ruta pasa a estado PROBE y se mantendrá ahí hasta que se lleve a cabo el intercambio de paquetes NS y NA)

4. Justo después de recibir el NA respuestas

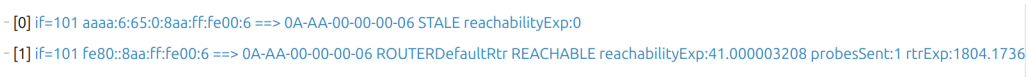


Figura 4.10: Estado de la tabla en t=11, tras el envío del NA (La ruta pasa a estado Reachable, porque el router contesta con el NA)