1. Aplicaciones Multicast IPv6

Versión 2.1, mayo 2022

Alumno (apellidos, nombre (DNI) : Juan José López Gómez

Alumno (apellidos, nombre (DNI) : Sergio Sánchez García

Fecha: 21-05-2022

Duración estimada de la práctica: 2 sesiones de 2h.

1.1. Entorno de trabajo

- Software de emulación de redes: GNS3 (Analizador de red: wireshark)
- Cisco IOS
- Lunix virtualizado (DebianAlumno)
- Framework de desarrollo en C para Linux que el alumno desee

1.2. Objetivos

- Utilizando el modelo cliente-servidor realizar una aplicación multicast basada en sockets IPv6
- Analizar el tráfico que la misma genera

1.3. Escenario de trabajo

En la fase de desarrollo se puede trabajar en el framework de desarrollo en C en Linux que el alumno desee. Para la realización de las pruebas de funcionamiento y análisis de tráfico se trabajará sobre un escenario prediseñado en GNS3. Descomprimir el fichero en el directorio GNS3/projects de la unidad Z. Se generará una carpeta con los archivos del escenario. Abrirlo con GNS3 y se mostrará lo siquiente.

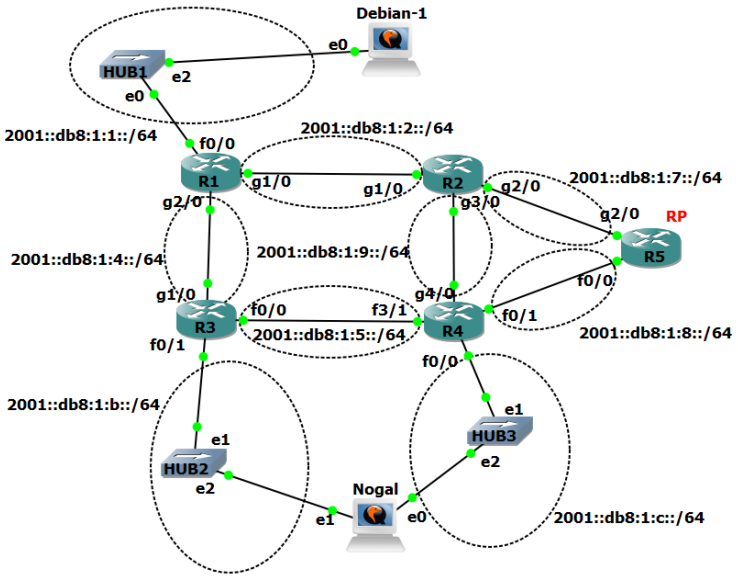


Figura 1: Escenario *multicast* IPv6.

1.4. Desarrollo de la aplicación

Utilizando el modelo cliente-servidor realizar una aplicación en red basada en sockets multicast para IPv6 con las siguientes especificaciones:

Programa Difusor o Fuente

Difundirá un mensaje que leerá por parámetros hasta que finalice ordenadamente capturando CTRL+C.

Se ajustará a la siguiente sintáxis:

```
difusor mensaje_a_difundir IPv6multicast interfaz puerto saltos intervalo
```

Si no se especifica ninguno de estos parámetros se difundirá un mensaje por defecto a una IPv6 multicast por defecto, por un interfaz por defecto, un puerto por defecto, un número de saltos y el intervalo de emisión de los mensajes.

Ejemplo:

```
difusor "Hola que tal" ff15::33 eth0 4343 10 15
```

Programa Suscriptor o Receptor

Se unirá al grupo multicast y leerá el mensaje que difunda la fuente mostrándo por pantalla junto con la IPv6 de la fuente. El suscriptor finalizará ordenadamente cuando se pulse Ctrl+C.

Se ajustará a la siguiente sintáxis:

```
suscriptor IPv6multicast interfaz puerto
```

Si no se especifica ninguno de estos parámetros la suscripción se realizará a una IPv6 multicast por defecto, por un interfaz por defecto y un puerto por defecto.

Ejemplo:

```
suscriptor ff15::33 eth0 4343
```

Las primeras líneas de todos los ficheros fuente contendrán unas líneas de comentarios donde se especifique el nombre del fichero fuente y el nombre completo, DNI y usuario con el que se ha subido la práctica. Ejemplo:

```
/*
** Fichero: multicast.c
** Autores:
** Antonio Pérez Puerto DNI 70876543M
** Araceli Sánchez Álvarez DNI 06876701A
** Usuario: i6876701
*/
```

Responde a las siguientes preguntas:

a) ¿A qué dirección (IP+puerto) hace bind el receptor o suscriptor? ¿Por qué?

El suscriptor hace bind a la dirección ff15::33 qué es la dirección multicast y al puerto 4343 desde los que se difunde el mensaje porque es por donde recibirá el mensaje que ha sido enviado desde la fuente.

b) ¿A qué dirección (IP+puerto) hace bind la fuente o emisor? ¿Por qué?

La fuente no hace bind a ninguna dirección porque este sólo tiene que difundir el mensaje deseado pero no debe recibir ningún tipo de dato ya que el protocolo utilizado para todas las aplicaciones multicast es UDP, por lo que no espera recibir respuesta.

c) ¿A qué dirección (IP+puerto) envía el emisor? ¿En que función del API de sockets se especifica? ¿Dónde se especifica el número de saltos? ¿Para qué sirve?

El emisor envía a la dirección ff15::33 al puerto 4343.

La función del API de sockets en que se especifica es la función sendto() .

El número de saltos se especifica en la función setsockopt, específicamente en los parámetros 3 (opción para limitar el número máximo de los paquetes multicast) y 4 (valor para la opción del parámetro 3), sirve para limitar el número de saltos que el paquete enviado puede dar por routers antes de que sea descartado.

d) ¿Quién debe especificar el deseo de recibir datagramas de una determinada difusión multicast, la fuente o el receptor? ¿En qué función del API de sockets se especifica?

Quién debe especificar el deseo de recibir datagramas de una determinada difusión multicast es el receptor. Esto se especifica en la función setsockopt, específicamente en los parámetros 3 (opción para suscribirnos a una dirección multicast) y 4 (valor para la opción del parámetro 3).

e) ¿Es cierto o falso que la fuente ha de suscribirse al grupo multicast?

La fuente no se debe suscribir al grupo multicast sino que sólo debe difundir el mensaje pero no espera recibir nada por lo que no es necesaria su suscripción.

f) Cuando un equipo tiene más de una interfaz de red si desea ser emisor de un grupo multicast, ¿en qué función del API de sockets debe especificar la interfaz por la que desea enviar? ¿Y si desea ser el receptor?

Si un equipo con varias interfaces de red desea ser emisor de un grupo multicast debe especificar la interfaz por la que desea envíar en la función setsockopt indicando en el tercer parámetro la opción IP_MULTICAST_IF, en el siguiente parámetro se indica el valor de la interfaz por la que difundir.

Si desea ser el receptor sería igual pero obviamente la llamada la tendría que hacer el receptor.

g) ¿Qué sucede internamente en un equipo que se suscribe a una dirección multicast? ¿Y cuando lo abandona? Especifica la IP multicast que has utilizado en tu práctica y la MAC correspondiente.

Internamente, cuando un equipo se suscribe a una dirección multicast este genera y envía un LISTENER REPORT avisando de que se va a suscribir a una dirección multicast, ocurre lo mismo cuando lo abandona.

La IP multicast utilizada en la práctica es la ff15:33 ¿Se manda algún mensaje por la red cuando un equipo se suscribe a un grupo multicast? ¿Y cuando lo abandona? ¿Cuál es su propósito?

Cuando un equipo se suscribe a un grupo multicast manda un mensaje MEMBERSHIP REPORT para indicar que se ha suscrito al grupo, cuando lo abandona envía un LEAVE GROUP para indicar que deja de pertenecer al grupo multicast. Ambos mensajes son de propósito informativo para que el resto de dispositivos del grupo actúe conforme al cambio indicado.

h) ¿Cómo sabe el receptor desde donde le llega la difusión? ¿Con qué función del API de sockets la puede obtener?

El receptor sabe desde donde llega la difusión ya que esta se indica en formato binario en la estructura del tipo sockaddr cuando se recibe el mensaje. Esta dirección se puede obtener con la función intet_ntop() que permite transformar la dirección IPv6 binaria a texto de formato estándar.

1.5. Probando la aplicación y analizando el tráfico multicast generado

En este apartado se detallan las pruebas de funcionamiento y el análisis del tráfico generado en la difusión que han de realizarse. En el análisis del tráfico hay que prestar atención a los mensajes ICMPv6 que se generan en la suscripción, así como los mensajes de la aplicación viajando desde la fuente a los suscriptores. Para ello tendremos una fuente en Nogal difundiendo por su interfaz eth1 y un suscriptor en Debian-1

Mostrar con capturas de pantalla que la aplicación funciona correctamente de forma que se vea como se han invocado la fuente y los suscriptores y la salida generada por ellos.

De igual forma observar con las capturas de tráfico los mensajes de suscripción y abandono del grupo multicast y por qué ramas viajan los datagramas de nuestra aplicación. Incluye en el informe la explicación de estos mensajes y adjunta los ficheros de las capturas de tráfico realizadas (solo con las tramas que nos interesan filtrando adecuadamente desde wireshark). Consulta el estado de las tablas de rutas multicast (*show ipv6 mroute*) y la lista de suscriptores (*show ipv6 mld groups*) para apoyar tu explicación.

Inicialmente todas las tablas de los routers están sin información:

```
R1#show ipv6 mroute
No mroute entries found.

R1#show ipv6 mld groups
No groups found.
```

Capturas que demuestran el funcionamiento correcto de la aplicación:

(invoque de fuente)

```
root@Nogal:~# ./fuente
Parameters of execution:
Message: default message to be sent
IP: ff15::33
Interface: eth0
Port: 6969
Hops: 15
Interval: 1
```

(invoque de suscriptor)

```
root@Debian-1:~# ./suscriptor
Parameters of execution:
IP: ff15::33
Interface: eth0
Port: 6969
```

(salida generada por ellos)

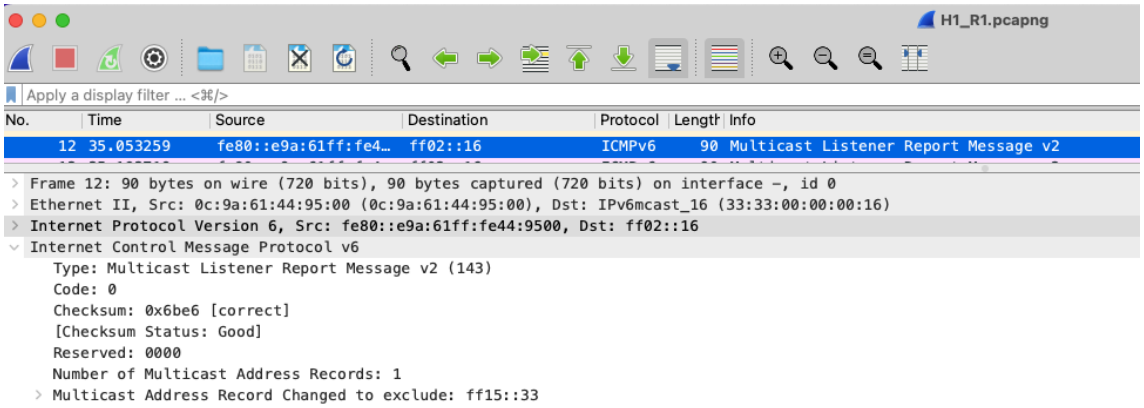
```
root@Nogal:~# ./fuente "pon buena nota" ff15::33 eth1 6969 15 1
Parameters of execution:
Message: pon buena nota
IP: ff15::33
Interface: eth1
Port: 6969
Hops: 15
Interval: 1
```

```
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
pon buena nota recibido desde 2001:db8:1:b:e9a:61ff:fe2b:f501
```

Tras ejecutar la fuente en Nogal este comienza a enviar mensajes UDP por la interfaz indicada, es decir, eth0. Estos mensaje siguen la ruta Nogal-HUB3-R4-R2-R5

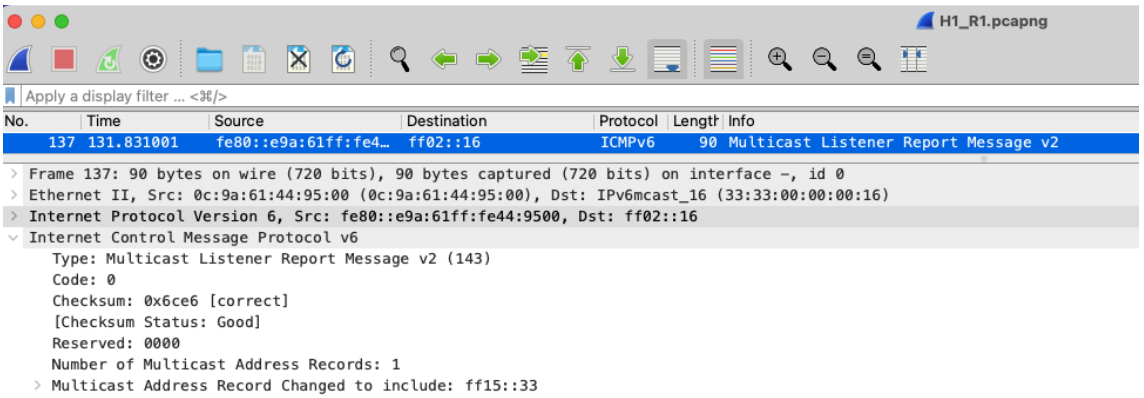
30	94.687365	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
31	95.581867	ca:05:09:6c:00:08	ca:05:09:6c:00:08	LOOP	60	Reply
32	95.626470	fe80::e9a:61ff:fe2…	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
33	95.688734	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
34	96.690456	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
35	97.691935	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
36	98.693026	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
37	99.694271	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
38	100.013008	fe80::c805:9ff:fe6…	ff02::5	OSPF	90	Hello Packet
39	100.696159	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
40	101.697812	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
41	102.699459	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
42	103.700737	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
43	104.702194	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
44	105.584652	ca:05:09:6c:00:08	ca:05:09:6c:00:08	LOOP	60	Reply
45	105.704883	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
46	106.708082	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
47	107.708655	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
48	108.711506	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
49	109.350939	fe80::c805:9ff:fe6…	ff02::d	PIMv2	136	Hello
50	109.713751	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
51	110.004147	fe80::c805:9ff:fe6…	ff02::5	OSPF	90	Hello Packet
52	110.714715	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
53	111.717612	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
54	112.723374	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
55	113.720951	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
56	114.722442	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
57	115.584566	ca:05:09:6c:00:08	ca:05:09:6c:00:08	LOOP	60	Reply
58	115.723656	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
59	116.726730	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
60	117.728613	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
61	118.730209	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
62	119.732128	2001:db8:1:c:e9a:6…	ff15::33	UDP	88	35814 → 6969 Len=26
63	120.000525	fe80::c805:9ff:fe6…	ff02::5	OSPF	90	Hello Packet

Posteriormente suscribimos al grupo multicast el equipo Debian-1, esto se indica con el mensaje Listener Report que indica el deseo de suscripción con el Multicast Address Record Changed to exclude:



De esta forma los mensajes enviados por Nogal seguirán la ruta HUB3-R4-R2-R1-HUB1 para llegar al único suscriptor del grupo multicast, esto se puede ver tanto en las capturas de wireshark como en los routers R4, R2 y R1 que realizan las entradas correspondientes en las tablas de rutas para el nuevo grupo multicast y para el suscriptor.

Finalmente Debian-1 indica que elimina su suscripción al grupo multicast mediante el mensaje Multicast Listener Report con Multicast Address Record Changed to include.



Utilería:

Fichero Makefile para compilar en los Debian:

```
CC = gcc
CFLAGS =
LIBS =

PROGS = suscriptor difusor

all: ${PROGS}

suscriptor: suscriptor.o
${CC} ${CFLAGS} -o $@ suscriptor.o ${LIBS}

difusor: difusor.o
${CC} ${CFLAGS} -o $@ difusor.o ${LIBS}

clean:
rm *.o ${PROGS}
```

Para copiar ficheros entre los equipos del escenario de GNS se puede utilizar scp de esta forma:

```
scp fichero root@nombre_equipo_destino:/directorio_destino
```

Ejemplo para copiar el fichero pp a nogal por su interfaz eth1 con R3:

```
scp pp root@NogalR3:/root
```

Para referirnos por nombre a los equipos Debian del escenario se han dado de alta los nombres y sus IPsV6 en el fichero /etc/hosts de estos equipos. Su contenido es:

```
2001:db8:1:b:e9a:61ff:fe2b:f501 NogalR3
2001:db8:1:c:e9a:61ff:fe2b:f500 NogalR4
2001:db8:1:1:e9a:61ff:fe44:9500 Debian-1
```