

## Práctica 9

### 1. ¿Qué es IPv6? ¿Por qué es necesaria su implementación?

Es la versión mas nueva del protocolo IP. Proporciona mayor espacio de direcciones (128 bits), un formato de cabecera simplificado y menor overhead de procesamiento. Es necesaria su implementación para solucionar las limitaciones (en cuanto a direcciones IP) que traía IPv4. Garantiza que haya suficientes direcciones IP únicas para todos los dispositivos. Además, IPv6 proporciona mejoras en seguridad y eficiencia en comparación con IPv4.

### 2. ¿Por qué no es necesario el campo Header Length en IPv6?

Porque en IPv4 había un campo opcional cuyo tamaño podía variar, haciendo que varíe la longitud del encabezado. En IPv6 todos los campos tienen un tamaño fijo, haciendo que el encabezado tenga una longitud constante.

### 3. ¿En qué se diferencia el checksum de IPv4 e IPv6? Y en cuánto a los campos checksum de TCP y UDP, ¿sufren alguna modificación en cuanto a su obligatoriedad de cálculo?

El checksum se utiliza para verificar que no haya corrupciones en el paquete IP. En IPv6, el campo de checksum es eliminado para simplificar el procesamiento de los paquetes en los routers y dispositivos, dejando esta verificación a las capas superiores (TCP y UDP). En cuanto a estas, su obligatoriedad no ha cambiado.

### 4. ¿Qué sucede con el campo Opciones en IPv6? ¿Existe, en IPv6, algún forma de enviar información opcional?

Se reemplazaron por las extensiones de encabezado. Estas:

- Permiten la extensibilidad del protocolo.
- Se encuentran a continuación del header.
- En general, son procesadas por los extremos.

### 5. Si quisiese que IPv6 soporte una nueva funcionalidad, ¿cómo lo haría?

Esta se implementaría como una extensión de encabezado.

### 6. ¿Es necesario el protocolo ICMP en IPv6? ¿Cumple las mismas funciones que en IPv4?

Mientras que ICMP podría ser prescindible en IPv4, en IPv6 es obligatorio. En IPv6 cumple funciones similares a las que desempeña en IPv4, aunque con algunas diferencias:

1. Descubrimiento de vecinos: ICMPv6 ND reemplaza ARP en IPv4, mapea direcciones IPv6 a direcciones de enlace para envío de paquetes.

2. Gestión de errores: ICMPv6 informa sobre errores de entrega, como paquetes demasiado grandes, tiempo de vida agotado y destino inalcanzable.
  3. Redirección de rutas: ICMPv6 informa a los hosts sobre rutas más eficientes en la red.
  4. Pruebas de conectividad: ICMPv6 incluye mensajes Echo Request y Echo Reply para pruebas de conectividad, similar a ping en IPv4.
  5. MLD (Multicast Listener Discovery): ICMPv6 se utiliza para el descubrimiento de escuchadores de multidifusión en redes IPv6.
7. **Transforme las siguientes direcciones MACs en Identificadores de Interfaces de 64 bits.**

- **00:1b:77:b1:49:a1**
- **e8:1c:23:a3:21:f4**

Invertir el 7mo bit de los primeros 8

Agregar el ff:fe en la mitad de la dirección MAC, ahí quedan los 64 bits

**00:1b:77:b1:49:a1**

00 → 00000000 → 00000010 → 02

021b:77ff:feb1:49a1

**e8:1c:23:a3:21:f4**

e8 → 11101000 → 11101010 → EA

ea1c:23ff:fea3:21f4

<https://ben.akrin.com/mac-address-to-ipv6-link-local-address-online-converter/>

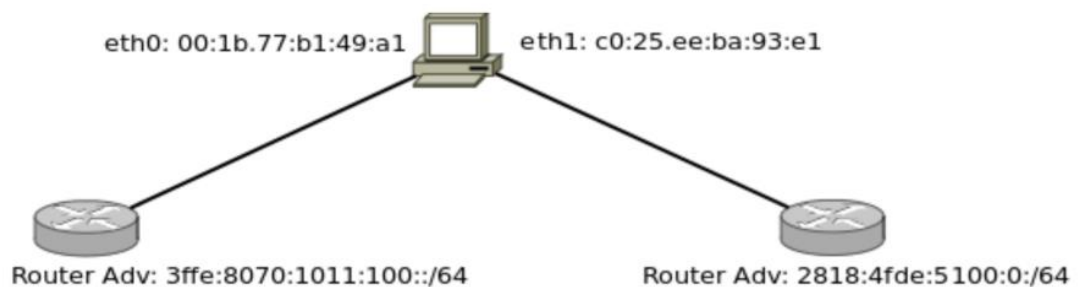
8. **¿Cuál de las siguientes direcciones IPv6 no son válidas?**
- **2001:0:1019:afde::1** → Es valida
  - **2001::1871::4** → No es valida :: no puede estar dos veces
  - **3ffg:8712:0:1:0000:aede:aaaa:1211** → No es válida, g no está la notación hexadecimal
  - **3::1** → Valida
  - **::** → Valida
  - **2001::** → Valida
  - **3ffe:1080:1212:56ed:75da:43ff:fe90:affe** → Valida
  - **3ffe:1080:1212:56ed:75da:43ff:fe90:affe:1001** → Invalida, tiene más de 128 bits
9. **¿Cuál sería una abreviatura correcta de 3f80:0000:0000:0a00:0000:0000:0000:0845?**

- 3f80::a00::845
- 3f80::a:845
- 3f80::a00:0:0:0:845:4567
- 3f80:0:0:a00::845 → Esta es la valida
- 3f8:0:0:a00::845}

10. Indique si las siguientes direcciones son de link-local, global-address, multicast, etc.

- fe80::1/64 → link-local
- 3ffe:4543:2:100:4398::1/64 → global
- :: → anycast
- ::1 → loopback
- ff02::2 → multicast
- 2818:edbc:43e1::8721:122 → global
- ff02::9 → multicast

11. Dado el siguiente diagrama, ¿qué direcciones IPv6 será capaz de autoconfigurar el nodo A en cada una de sus interfaces?



Todos tienen al menos dos direcciones IP, una local y una global.

- 00:1b:77:b1:49:a1 → fe80::21b:77ff:feb1:49a1 local
- 3ffe:8070:1011:100:21b:77ff:feb1:49a1 /64 global
- c0:25:ee:ba:93:e1 → fe80::c225:eeff:feba:93e1 local
- 2818:4fde:5100:0:c225:eeff:feba:93e1 /64 global

12. Al autogenerarse una dirección IPv6 sus últimos 64 bits en muchas ocasiones no se deducen de la dirección MAC, se generan de forma random, ¿por qué sucede esto? ¿Qué es lo que se intenta evitar? (Ver direcciones temporarias, RFC 8981)

Es una medida de privacidad y seguridad. Esta práctica se utiliza para evitar la asociación directa y constante entre la dirección MAC de un dispositivo y su dirección IPv6 ya que cuando se utilizan las direcciones MAC en la construcción de direcciones IPv6, se puede rastrear la identidad de un dispositivo de manera más fácil y constante a lo largo del tiempo.

Para abordar este problema, se introdujo la idea de las "direcciones temporales" en la RFC 4941 y, posteriormente, se actualizó con la RFC 8981. Las direcciones temporales permiten generar los últimos 64 bits de una dirección IPv6 de forma aleatoria, lo que dificulta el rastreo constante de un dispositivo a través de su dirección MAC.

Al generar direcciones temporales de forma aleatoria, se mejora la privacidad del usuario al hacer que sea más difícil vincular de manera persistente la actividad en línea con una identidad específica basada en la dirección MAC.

**13. Utilizando la máquina virtual abrir la topología llamada 3-ruteo-OSPF.imn para realizar las siguientes pruebas:**

- a. **Habilitar la vista de las direcciones IPv6 en la topología (View ->show ->IPv6 Addresses).**
- b. **Esperar a que la red converja. Verificar, mediante ping6, la comunicación entre n6 y n7.**

```
root@n6:/tmp/pycore.40373/n6.conf# ping6 2001:4::22
PING 2001:4::22(2001:4::22) 56 data bytes
From 2001::1 icmp_seq=1 Destination unreachable: No route
From 2001::1 icmp_seq=2 Destination unreachable: No route
From 2001::1 icmp_seq=3 Destination unreachable: No route
From 2001::1 icmp_seq=4 Destination unreachable: No route
From 2001::1 icmp_seq=5 Destination unreachable: No route
```

**c. Observar la configuración IPv6:**

**i. De la PC n6**

```
root@n6:/tmp/pycore.40373/n6.conf# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.10 netmask 255.255.255.0 broadcast 0.0.0.0
    inet6 fe80::200:ff:feaa:1 prefixlen 64 scopeid 0x20<link>
    inet6 2001::10 prefixlen 64 scopeid 0x0<global>
    ether 00:00:00:aa:00:01 txqueuelen 1000 (Ethernet)
    RX packets 81 bytes 9064 (8.8 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 20 bytes 1880 (1.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

**ii. De la PC n7.**

```
vcmd
root@n7:/tmp/pycore.40373/n7.conf# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.4.22 netmask 255.255.255.0 broadcast 0.0.0.0
    inet6 2001:4::22 prefixlen 64 scopeid 0x0<global>
    inet6 fe80::200:ff:feaa:f prefixlen 64 scopeid 0x20<link>
    ether 00:00:00:aa:00:0f txqueuelen 1000 (Ethernet)
    RX packets 89 bytes 9300 (9.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 11 bytes 962 (962.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@n7:/tmp/pycore.40373/n7.conf#
```

### iii. Del router n1.

```
root@n1:/tmp/pycore.40373/n1.conf# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.1 netmask 255.255.255.0 broadcast 0.0.0.0
    inet6 2001::1 prefixlen 64 scopeid 0x0<global>
    inet6 fe80::200:ff:feaa:0 prefixlen 64 scopeid 0x20<link>
    ether 00:00:00:aa:00:00 txqueuelen 1000 (Ethernet)
    RX packets 75 bytes 8266 (8.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 109 bytes 9230 (9.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.1.2 netmask 255.255.255.0 broadcast 0.0.0.0
    inet6 fe80::200:ff:feaa:3 prefixlen 64 scopeid 0x20<link>
    inet6 2001:1::2 prefixlen 64 scopeid 0x0<global>
    ether 00:00:00:aa:00:03 txqueuelen 1000 (Ethernet)
    RX packets 91 bytes 10314 (10.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 113 bytes 9918 (9.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.1 netmask 255.255.255.0 broadcast 0.0.0.0
    inet6 2001:2::1 prefixlen 64 scopeid 0x0<global>
    inet6 fe80::200:ff:feaa:4 prefixlen 64 scopeid 0x20<link>
    ether 00:00:00:aa:00:04 txqueuelen 1000 (Ethernet)
    RX packets 96 bytes 10736 (10.4 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 112 bytes 9792 (9.5 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

### iv. La tabla de rutas tanto de las PCs como de los routers.

n1

```

root@n1:/tmp/pycore.40373/n1.conf# ip -6 route show
2001::/64 dev eth0 proto kernel metric 256 pref medium
2001:1::/64 dev eth1 proto kernel metric 256 pref medium
2001:2::/64 dev eth2 proto kernel metric 256 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth1 proto kernel metric 256 pref medium
fe80::/64 dev eth2 proto kernel metric 256 pref medium

```

n2

```

root@n2:/tmp/pycore.40373/n2.conf# ip -6 route show
2001:1::/64 dev eth0 proto kernel metric 256 pref medium
2001:3::/64 dev eth1 proto kernel metric 256 pref medium
2001:4::/64 dev eth2 proto kernel metric 256 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth1 proto kernel metric 256 pref medium
fe80::/64 dev eth2 proto kernel metric 256 pref medium

```

n3

```

root@n3:/tmp/pycore.40373/n3.conf# ip -6 route show
2001:2::/64 dev eth0 proto kernel metric 256 pref medium
2001:3::/64 dev eth1 proto kernel metric 256 pref medium
2001:5::/64 dev eth2 proto kernel metric 256 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth1 proto kernel metric 256 pref medium
fe80::/64 dev eth2 proto kernel metric 256 pref medium

```

n6

```

root@n6:/tmp/pycore.40373/n6.conf# ip -6 route show
2001::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
default via 2001::1 dev eth0 metric 1024 pref medium

```

n7

```

root@n7:/tmp/pycore.40373/n7.conf# ip -6 route show
2001:4::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
default via 2001:4::1 dev eth0 metric 1024 pref medium

```

n8

```

root@n8:/tmp/pycore.40373/n8.conf# ip -6 route show
2001:4::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
default via 2001:4::1 dev eth0 metric 1024 pref medium

```

n9

```

root@n9:/tmp/pycore.40373/n9.conf# ip -6 route show
2001:4::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
default via 2001:4::1 dev eth0 metric 1024 pref medium

```

n10

```

root@n10:/tmp/pycore.40373/n10.conf# ip -6 route show
2001:5::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
default via 2001:5::1 dev eth0 metric 1024 pref medium

```

n11

```

root@n11:/tmp/pycore.40373/n11.conf# ip -6 route show
2001:5::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
default via 2001:5::1 dev eth0 metric 1024 pref medium
root@n11:/tmp/pycore.40373/n11.conf#

```

n12

```

root@n12:/tmp/pycore.40373/n12.conf# ip -6 route show
2001:5::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
default via 2001:5::1 dev eth0 metric 1024 pref medium
root@n12:/tmp/pycore.40373/n12.conf#

```

d. Responda:

- i. ¿Cuántas direcciones IPv6 se observan tanto en la PC n6 como en la PC n7?

Se observan 2, la global y la link-local

- ii. ¿Es posible desde la PC n7 hacer un ping6 a cada una de las direcciones IPv6 de la PC n6? ¿Por qué?

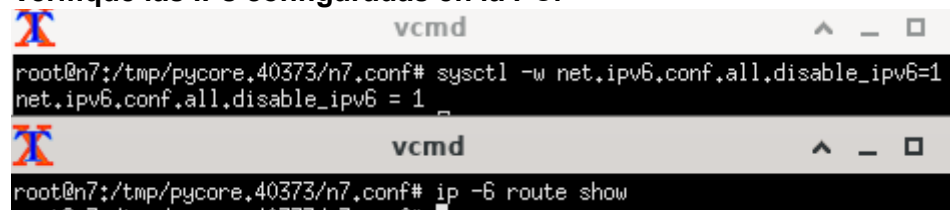
No, porque una de las IPv6 de la PC n6 es de tipo link-local que tienen un alcance limitado a la red local. n6 esta en una red local distinta a n7.

- e. Cuando se quiere hacer ping6 a una dirección link-local es necesario especificar la interfaz que se quiere utilizar (ping6 -I eth0 <IPv6-address>) ¿Por qué?

Porque las direcciones link-local son específicas de una interfaz de red. Cada interfaz de red puede tener su propia dirección link-local única.

- f. Deshabilite la configuración de IPv6 en la PC n7 mediante el comando:  
sysctl -w net.ipv6.conf.all.disable\_ipv6=1

- i. Verifique las IPs configuradas en la PC.



```

vcmd
root@n7:/tmp/pycore.40373/n7.conf# sysctl -w net.ipv6.conf.all.disable_ipv6=1
net.ipv6.conf.all.disable_ipv6 = 1
vcmd
root@n7:/tmp/pycore.40373/n7.conf# ip -6 route show
root@n7:/tmp/pycore.40373/n7.conf#

```

- ii. Luego de deshabilitarse IPv6, ¿puede comunicarse con la PC n6? ¿Cómo?

Si, pero debe ser a través de IPv4.