Práctico FCEFyN Redes de computadoras

Trabajo Práctico 3

Docente: Matías R. Cuenca del Rey

Mail: [mcuenca@unc.edu.ar](mailto:mcuenca@unc.edu.ar)

Ayudantes alumnos: Elisabeth Leonhard - Andrés Serjoy

Redes de computadoras

Facultad de Ciencias Exactas, Físicas y Naturales

Universidad Nacional de Córdoba

Práctico 3: Ruteo estático. Fragmentación

Presentación de consignas.

### Ejercicio 1: Ruteo estático. Fragmentación.

#### Recomendaciones

* Lea con cuidado las consignas.
* Tenga certeza de los comandos que ejecuta.
* Desactivar firewall en todos los puntos Sistema Operativos intervinientes.
* Utilizar cable ethernet para conectar las dos máquinas físicas.
* Para virtualización utilizar Virtualbox.

#### Esquema

* Se realizará bajo IPv6. No se usará IPv4.
* Realizar un shell script que pueda replicar el trabajo práctico.
* Se usarán las dos máquinas físicas por grupo. Se conectarán las dos máquinas físicas entre sí.
* Se usarán las máquinas virtuales Desktop o Server en cada máquina física, según preferencia del alumno.
* Se usarán namespaces dentro de las máquinas virtuales para simular routers y hosts.
* En cada computadora, las interfaces de red de las dos máquinas virtuales estarán conectadas mediante 'adaptador puente'.

#### Diagrama



#### Tabla de asignación de direcciones IPv6

Crear la tabla de asignación de direcciones IP

#### 

|  |  |  |
| --- | --- | --- |
| **Computadora** | **Interfaz de red** | **Dirección IP** |

#### Links de ayuda

Conceptos de namespaces: <https://www.toptal.com/linux/separation-anxiety-isolating-your-system-with-linux-namespaces>

#### Consignas

##### Creación y conexión de namespaces ns 1.1 y ns 2.1

1.- Crear los namespaces ns 1.1 y ns 2.1 en las correspondientes máquinas virtuales

2.- Crear un bridge llamado br-externo en cada máquina virtual

3.- Conectar cada namespace con el bridge creado en el punto anterior.

4.- Conectar la interfaz externa de la máquina virtual al bridge creado en el punto anterior.

5.- Configurar direcciones de red a los namespaces.

6.- Probar conectividad entre las dos puntas

7.- Configurar MTU a 500 para estas interfaces en los namespaces. No cambiar MTU en ningún otro punto de la red. Revisar RFC 2460.

##### Creación de redes y ruteo

8.- Crear los routers y host en cada computadora.

9.- Configurar ruteo estático para que todos los hosts desde todas las interfaces sean alcanzables unos a otros

##### Análisis de tráfico

10.- Mediante análisis de tráfico, verificar la fragmentación que se se debe producir cuando un datagrama IP va desde una máquina virtual a otra.

11.- Verificar cómo sucede la fragmentación en IPv6. Explicar cuál es la diferencia con IPv4.

Tabla de asignación de direcciones IPv6

|  |  |  |  |
| --- | --- | --- | --- |
| Namespace | Interface | Address |  |
| ns1.1 | fa1.0 | 2001:aaaa:aaaa:1::1 |  |
| fa1.1 | 2001:aaaa:aaaa:2::1 |  |
| fa1.2 | 2001:aaaa:aaaa:3::1 |  |
| fa1.3 | 2001:aaaa:aaaa:7::1 | 2001:aaaa:aaaa:7::2 |
| ns1.2 | fa2.0 | 2001:aaaa:aaaa:1::2 |  |
| ns1.3 | fa3.0 | 2001:aaaa:aaaa:2::2 |  |
| fa3.1 | 2001:aaaa:aaaa:4::1 |  |
| fa3.2 | 2001:aaaa:aaaa:5::1 |  |
| ns1.4 | fa4.0 | 2001:aaaa:aaaa:3::2 |  |
| fa4.1 | 2001:aaaa:aaaa:4::2 |  |
| fa4.2 | 2001:aaaa:aaaa:6::1 |  |
| ns1.5 | eth5 | 2001:aaaa:aaaa:5::2 |  |
| ns1.6 | eth6 | 2001:aaaa:aaaa:6::2 |  |
| ns2.1 | fa1.0 | 2001:aaaa:bbbb:1::1 |  |
| fa1.1 | 2001:aaaa:bbbb:2::1 |  |
| fa1.2 | 2001:aaaa:bbbb:3::1 |  |
| fa1.3 | 2001:aaaa:aaaa:7::3 | 2001:aaaa:aaaa:7::4 |
| ns2.2 | fa2.0 | 2001:aaaa:bbbb:1::2 |  |
| ns2.3 | fa3.0 | 2001:aaaa:bbbb:2::2 |  |
| fa3.1 | 2001:aaaa:bbbb:4::1 |  |
| fa3.2 | 2001:aaaa:bbbb:5::1 |  |
| ns2.4 | fa4.0 | 2001:aaaa:bbbb:3::2 |  |
| fa4.1 | 2001:aaaa:bbbb:4::2 |  |
| fa4.2 | 2001:aaaa:bbbb:6::1 |  |
| ns2.5 | eth5 | 2001:aaaa:bbbb:5::2 |  |
| ns2.6 | eth6 | 2001:aaaa:bbbb:6::2 |  |

Script *create\_ns1\_X:*

|  |
| --- |
| echo Eliminando componentes anteriores ip -all netns del ip link set dev br-externo down brctl delbr br-externo  #Creacion namespaces/ Activacion loopbacks en ns/ Activacion fordwarding echo Creando namespaces: for i in $(seq 1 6); do  ip netns add ns1.$i  ip netns exec ns1.$i ip link set dev lo up  ip netns exec ns1.$i sysctl -w net.ipv6.conf.all.forwarding=1  echo ns1.$i done  echo Creando y conectando todas las interfaces virtuales #Creo interfaces para unir ns1.1 con ns1.2 ip link add fa1.0 type veth peer name fa2.0  #Conecto interfaz a ns1.1 ip link set fa1.0 netns ns1.1 ip netns exec ns1.1 ip link set dev fa1.0 up  #Conecto interfaz a ns1.2 ip link set fa2.0 netns ns1.2 ip netns exec ns1.2 ip link set dev fa2.0 up #############################################################  #Creo interfaces para unir ns1.1 con ns1.3 ip link add fa1.1 type veth peer name fa3.0  #Conecto interfaz a ns1.1 ip link set fa1.1 netns ns1.1 ip netns exec ns1.1 ip link set dev fa1.1 up  #Conecto interfaz a ns1.3 ip link set fa3.0 netns ns1.3 ip netns exec ns1.3 ip link set dev fa3.0 up #############################################################  #Creo interfaces para unir ns1.1 con ns1.4 ip link add fa1.2 type veth peer name fa4.0  #Conecto interfaz a ns1.1 ip link set fa1.2 netns ns1.1 ip netns exec ns1.1 ip link set dev fa1.2 up  #Conecto interfaz a ns1.4 ip link set fa4.0 netns ns1.4 ip netns exec ns1.4 ip link set dev fa4.0 up ############################################################  #Creo interfaces para unir ns1.3 con ns1.4 ip link add fa3.1 type veth peer name fa4.1  #Conecto interfaz a ns1.3 ip link set fa3.1 netns ns1.3 ip netns exec ns1.3 ip link set dev fa3.1 up  #Conecto interfaz a ns1.4 ip link set fa4.1 netns ns1.4 ip netns exec ns1.4 ip link set dev fa4.1 up ############################################################  #Creo interfaces para unir ns1.3 con ns1.5 ip link add fa3.2 type veth peer name eth5  #Conecto interfaz a ns1.3 ip link set fa3.2 netns ns1.3 ip netns exec ns1.3 ip link set dev fa3.2 up  #Conecto interfaz a ns1.5 ip link set eth5 netns ns1.5 ip netns exec ns1.5 ip link set dev eth5 up ############################################################  #Creo interfaces para unir ns1.4 con ns1.6 ip link add fa4.2 type veth peer name eth6  #Conecto interfaz a ns1.4 ip link set fa4.2 netns ns1.4 ip netns exec ns1.4 ip link set dev fa4.2 up  #Conecto interfaz a ns1.6 ip link set eth6 netns ns1.6 ip netns exec ns1.6 ip link set dev eth6 up ############################################################  #Creo interfaces para unir ns1.1 con bridge ip link add fa1.3 type veth peer name brside  #Conecto interfaz a ns1.1 ip link set fa1.3 netns ns1.1 ip netns exec ns1.1 ip link set dev fa1.3 up  #Creacion Bridge  echo Creando bridge externo brctl addbr br-externo brctl stp br-externo off ip link set dev br-externo up brctl addif br-externo brside brctl addif br-externo enp0s3 ip link set dev brside up   #Configuro MTU para todas las interfaces echo Configurando MTU a 1280 en todas las interfaces sudo ip netns exec ns1.1 ifconfig fa1.0 mtu 1280 sudo ip netns exec ns1.1 ifconfig fa1.1 mtu 1280 sudo ip netns exec ns1.1 ifconfig fa1.2 mtu 1280 sudo ip netns exec ns1.1 ifconfig fa1.3 mtu 1280 sudo ip netns exec ns1.2 ifconfig fa2.0 mtu 1280 sudo ip netns exec ns1.2 ifconfig lo mtu 1280 sudo ip netns exec ns1.3 ifconfig fa3.0 mtu 1280 sudo ip netns exec ns1.3 ifconfig fa3.1 mtu 1280 sudo ip netns exec ns1.3 ifconfig fa3.2 mtu 1280 sudo ip netns exec ns1.4 ifconfig fa4.0 mtu 1280 sudo ip netns exec ns1.4 ifconfig fa4.1 mtu 1280 sudo ip netns exec ns1.4 ifconfig fa4.2 mtu 1280 sudo ip netns exec ns1.5 ifconfig eth5 mtu 1280 sudo ip netns exec ns1.6 ifconfig eth6 mtu 1280 ifconfig brside mtu 1280 ifconfig enp0s3 mtu 1280 |

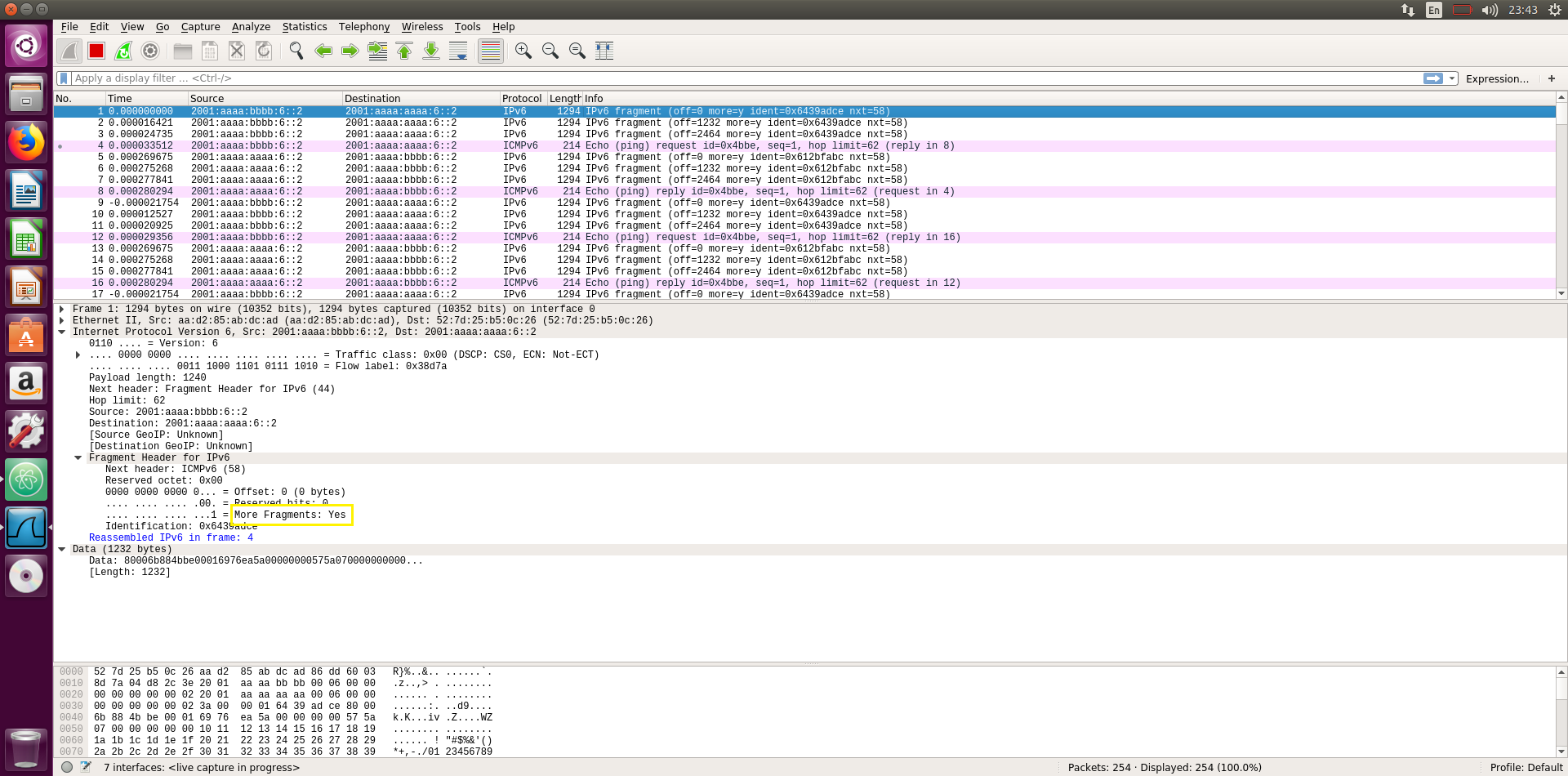
Script *configure\_IPs\_ns1\_X*

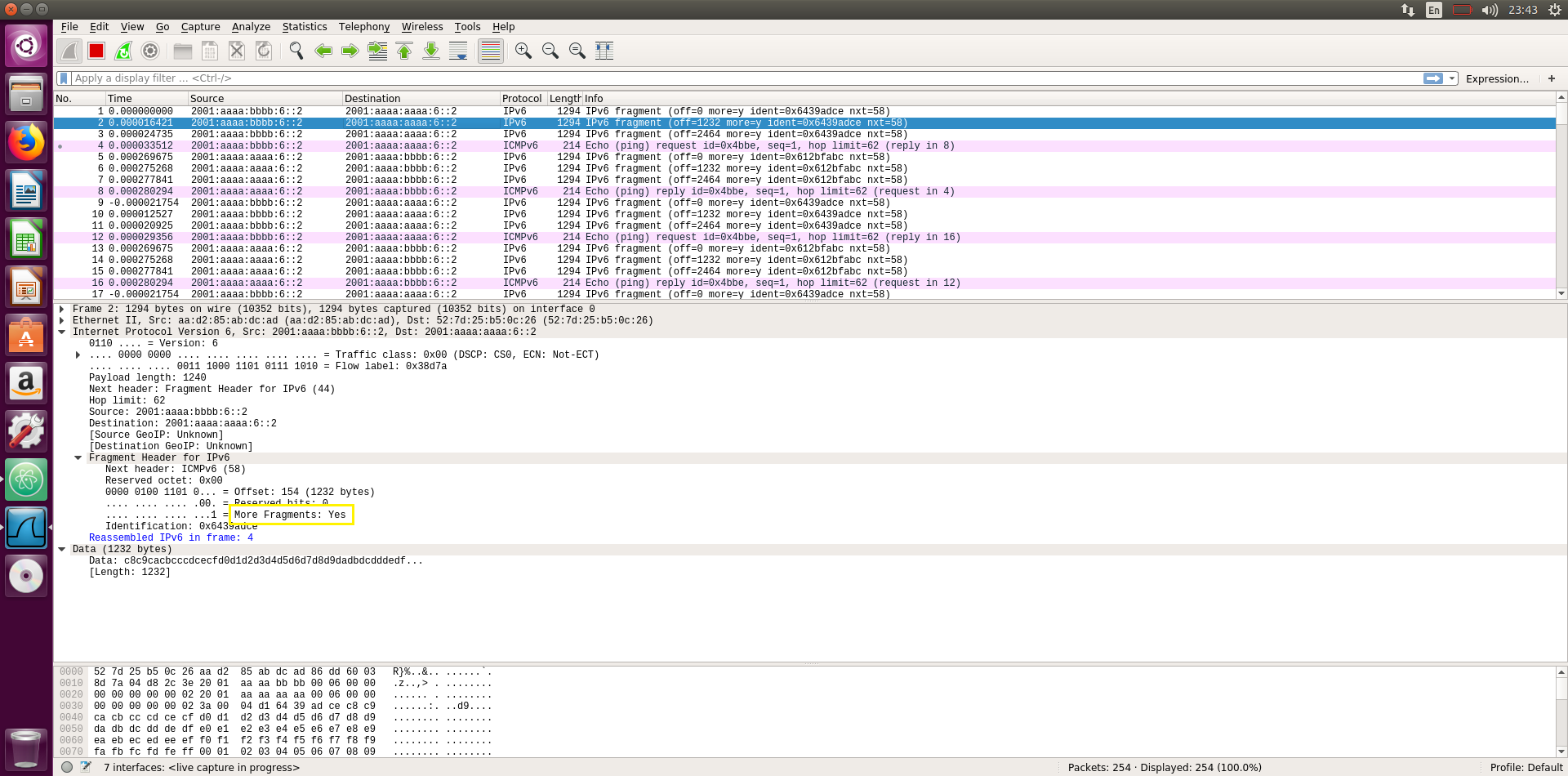
|  |
| --- |
| #Agrego direcciones IPv6  ip netns exec ns1.1 ip -6 addr add 2001:aaaa:aaaa:1::1/64 dev fa1.0 ip netns exec ns1.1 ip -6 addr add 2001:aaaa:aaaa:2::1/64 dev fa1.1 ip netns exec ns1.1 ip -6 addr add 2001:aaaa:aaaa:3::1/64 dev fa1.2 ip netns exec ns1.1 ip -6 addr add 2001:aaaa:aaaa:7::1/64 dev fa1.3 ip netns exec ns1.2 ip -6 addr add 2001:aaaa:aaaa:1::2/64 dev fa2.0 ip netns exec ns1.3 ip -6 addr add 2001:aaaa:aaaa:2::2/64 dev fa3.0 ip netns exec ns1.3 ip -6 addr add 2001:aaaa:aaaa:4::1/64 dev fa3.1 ip netns exec ns1.3 ip -6 addr add 2001:aaaa:aaaa:5::1/64 dev fa3.2 ip netns exec ns1.4 ip -6 addr add 2001:aaaa:aaaa:3::2/64 dev fa4.0 ip netns exec ns1.4 ip -6 addr add 2001:aaaa:aaaa:4::2/64 dev fa4.1 ip netns exec ns1.4 ip -6 addr add 2001:aaaa:aaaa:6::1/64 dev fa4.2 ip netns exec ns1.5 ip -6 addr add 2001:aaaa:aaaa:5::2/64 dev eth5 ip netns exec ns1.6 ip -6 addr add 2001:aaaa:aaaa:6::2/64 dev eth6  ifconfig br-externo inet6 add 2001:aaaa:aaaa:7::2/64  #Agrego rutas estáticas y gateway por defecto a los namespaces ip netns exec ns1.1 ip -6 route add 2001:aaaa:bbbb:1::/64 via 2001:aaaa:aaaa:7::3 ip netns exec ns1.1 ip -6 route add 2001:aaaa:bbbb:2::/64 via 2001:aaaa:aaaa:7::3 ip netns exec ns1.1 ip -6 route add 2001:aaaa:bbbb:3::/64 via 2001:aaaa:aaaa:7::3 ip netns exec ns1.1 ip -6 route add 2001:aaaa:aaaa:4::/64 via 2001:aaaa:aaaa:2::2 ip netns exec ns1.1 ip -6 route add 2001:aaaa:aaaa:5::/64 via 2001:aaaa:aaaa:2::2 ip netns exec ns1.1 ip -6 route add 2001:aaaa:aaaa:6::/64 via 2001:aaaa:aaaa:3::2 ip netns exec ns1.1 ip -6 route add default via 2001:aaaa:aaaa:7::3 ip netns exec ns1.2 ip -6 route add default via 2001:aaaa:aaaa:1::1 ip netns exec ns1.3 ip -6 route add default via 2001:aaaa:aaaa:2::1 ip netns exec ns1.4 ip -6 route add default via 2001:aaaa:aaaa:3::1 ip netns exec ns1.5 ip -6 route add default via 2001:aaaa:aaaa:5::1 ip netns exec ns1.6 ip -6 route add default via 2001:aaaa:aaaa:6::1  ip addr flush dev enp0s3 ifconfig enp0s3 down ifconfig enp0s3 up |

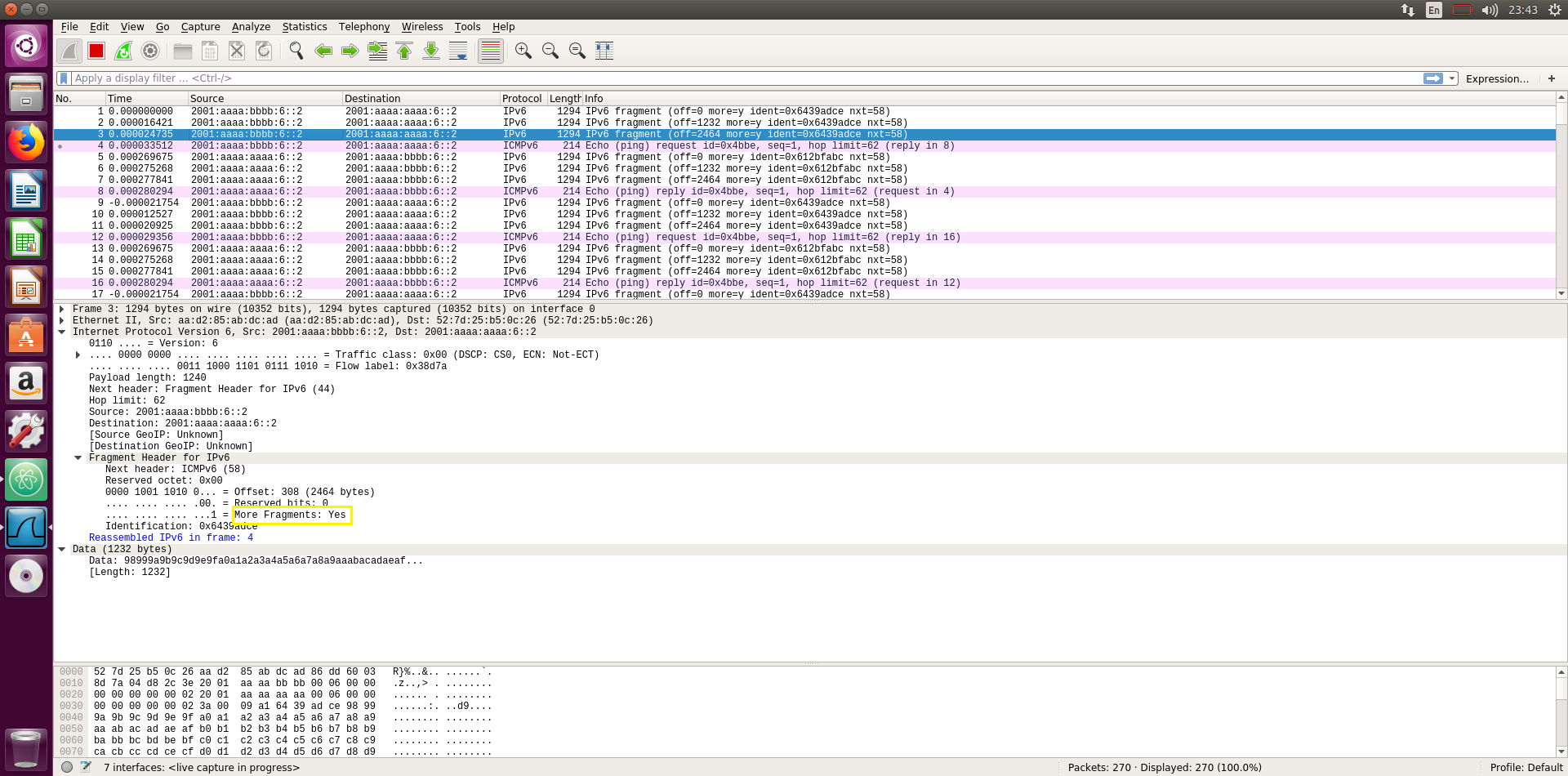
##### Análisis de tráfico

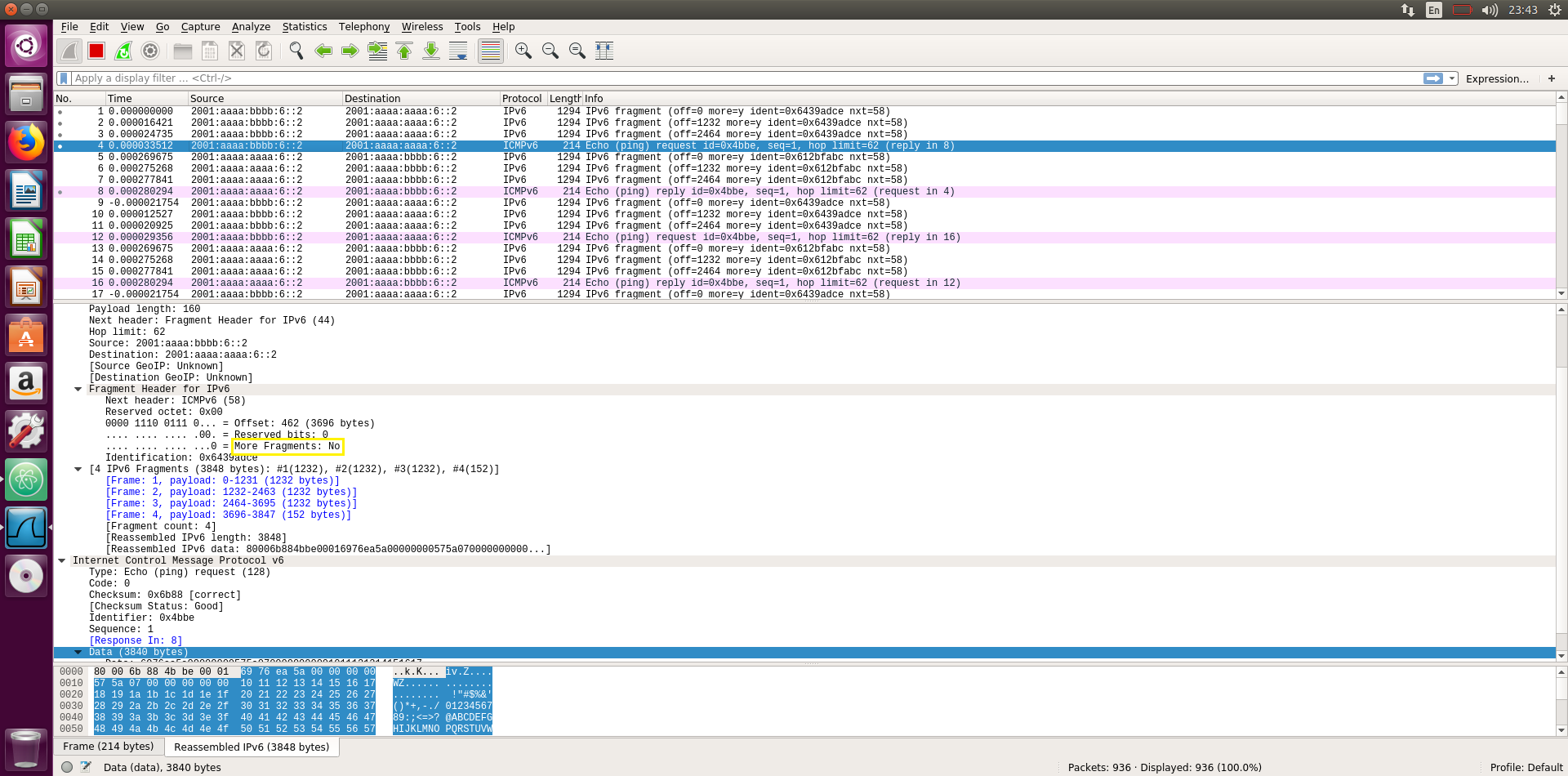
10.- Mediante análisis de tráfico, verificar la fragmentación que se debe producir cuando un datagrama IP va desde una máquina virtual a otra.

sudo ip netns exec ns2.6 ping6 2001:aaaa:aaaa:6::2 -s 3840



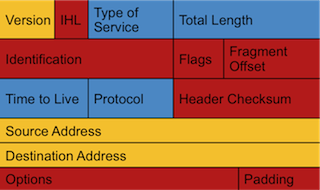






11.- Verificar cómo sucede la fragmentación en IPv6. Explicar cuál es la diferencia con IPv4.

Con el fin de entender un poco más la forma en la que IPv6 trabaja a diferencia de IPv4, debemos ver cómo es que se agrega información en un paquete de IPv6 por medio de su encabezado. Revisemos primero un encabezado de IPv4 que es con lo que estamos más familiarizados:



Todas las secciones que aparecen en color ROJO son elementos que se eliminaron del encabezado de IPv6. Lo que está en AZUL, cambió de nombre y posición. Lo que está en AMARILLO se mantiene igual.

Veamos qué secciones se eliminaron:

* IHL (IP Header Length - 4 bits). En IPv4 el tamaño del encabezado es variable. Este valor combinado con el Total Length del paquete le permite al equipo saber cuántos bytes hay de carga útil o payload (Payload = Total Length - IHL). Con un tamaño de encabezado variable, se requiere de alguna rutina o algoritmo para que los equipos determinen en dónde empieza la información dentro del paquete lo que hace un poco más complejo el manejo y proceso de paquetes.
* Identification/Flags/Fragment Offset (32 bits). Estos elementos se utilizan para el manejo de fragmentación de paquetes.
* Header Checksum (16 bits). Utiliza un algoritmo para determinar si el encabezado no está corrupto o ha sido modificado incorrectamente. Tecnologías como NAT que modifican parámetros del encabezado también deben modificar esta sección de manera adecuada. IPv6 elimina el uso de esta sección y deja la revisión de la integridad de los paquetes a CRC en capa 2 y checksums de capa 4.
* Options. Es una sección de tamaño variable y contiene datos sobre la información del paquete que no se pudieron integrar en otros campos del encabezado. Es esta sección la que hace que el encabezado de IPv4 sea de tamaño variable.
* Padding. Es una serie de bits con un valor de '0' que sirve para alinear el encabezado a un múltiplo de palabras de 32 bits.

Esta es la forma de un encabezado de IP versión 6:



Como se puede observar, las secciones de Versión (4 bits) y Direcciones de Origen y Destino (128 bits) permanecen igual. Además existe una sección nueva de 20 bits, Flow Label, que es para un uso futuro basado en el RFC 3697. Esta sección servirá para identificar un flujo específico y hacerle un tratamiento (QoS) idéntico a todos los paquetes que pertenezcan al flujo (aún no está implementado).

Veamos ahora las secciones modificadas:

* Traffic Class (8 bits). Tiene la misma funcionalidad que el Type of Service, sirve para almacenar la información de precedencia, DSCP o clase de servicio dependiendo de la implementación de QoS.
* Payload Length (16 bits). En IPv6 tenemos un encabezado fijo de 40 bytes por lo que solo necesitamos saber el tamaño de la información útil del paquete. Esto también nos quita la necesidad de implementar rutinas de búsqueda como en IPv4. Ya estamos utilizando 4 veces el tamaño de buffers para el manejo de direcciones de 128 bits, cualquier cosa que disminuya el procesamiento extensivo de los paquetes es preferible.
* Next Header (8 bits). Esta sección nos puede indicar el protocolo al que pertenece la información que estamos acarreando (TCP, UDP, ICMPv6, etc.). También se utiliza para indicarnos si existe información de opciones para los datos que se transportan. En lugar de manejar Opciones en el encabezado como IPv4, éstas las integramos dentro de la carga útil de los paquetes para mantener el encabezado en un tamaño fijo. Esta sección nos indicaría si hay opciones que revisar dentro del paquete.
* Hop Limit (8 bits). Sirve exactamente la misma función que el TTL de IPv4. Solo se cambió el nombre ya que este parámetro es un contador de saltos (equipos capa 3) y no un contador de tiempo como parece indicar el nombre en IPv4