Práctica 5 – Overlays

1. Introducción y objetivos

El objetivo de esta práctica es construir escenarios donde el segmento de LAN empleado por contenedores se construya como una overlay.

2. Overlay empleando GRE

En este caso vamos a extender la LAN capa 2 entre los hosts para que los contenedores se encuentren todos en la misma LAN/subred IP pero lo vamos a hacer sin puentearla con la LAN externa a los hosts. Lo que haremos será introducir las tramas Ethernet en un túnel GRE para hacerlas llegar de un host al otro (Figura 1).

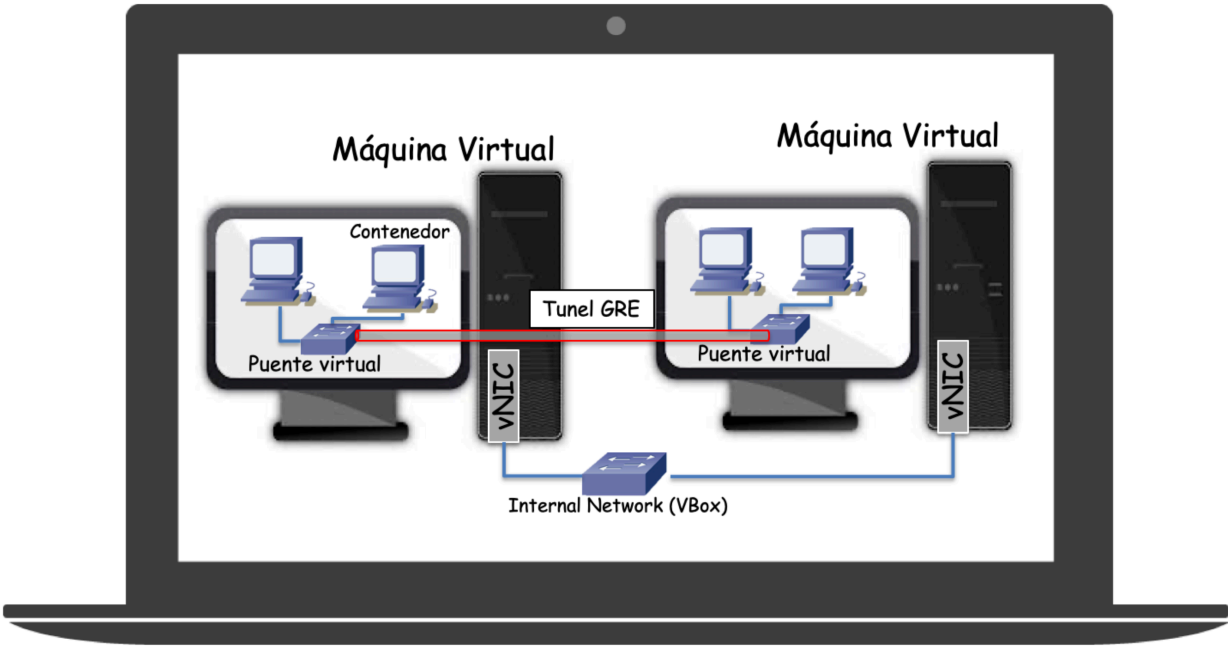

Figura 1 - Escenario Ethernet sobre GRE

Muchos túneles GRE se emplean para transportar IP sobre IP, pero eso no impide transportar otros protocolos sobre GRE y Ethernet es uno de ellos.

Empezamos con los hosts sin configurar.

Dado que la comunicación entre los hosts es con paquetes IP quiere decir que podría haber routers entre ellos (aunque en este ejemplo tengamos solo la internal network) y también significa que necesitamos asignar dirección IP a los interfaces externos de los hosts. Asigne direcciones a los interfaces de los hosts para que haya conectividad IP entre ellos. Para el resto de ejemplo supondremos que un host es 172.16.1.1 y el otro 172.16.1.2.

En cada host hay que crear un interfaz virtual que haga referencia al túnel GRE. El tipo de interfaz en el Kernel es “gretap” y lo crearíamos de la siguiente forma:

```
# ip link add eltunelgre type gretap remote 172.16.1.2 local 172.16.1.1
```

Evidentemente en el otro host las direcciones IP en el comando estarán en el orden contrario, dado que determinan los extremos local y remoto del túnel.

Con esto se habrá creado un nuevo interfaz de nombre *eltunegre*.

Cree un bridge para la comunicación entre los contenedores en cada host y conecte algún contenedor en cada host a su bridge. Levante el interfaz de túnel GRE en cada host y añádalo al bridge correspondiente.

No hace falta activar el reenvío de paquetes IP en los hosts pues no van a reenviar paquetes IP. El puente virtual actúa en capa 2 y uno de sus interfaces será el túnel. Ese interfaz especial introduce la trama Ethernet que se le entregan en un paquete IP dirigido al otro extremo del túnel. En ningún momento el host está haciendo *forwarding* de un paquete IP por una tabla de rutas.

Ahora configure la dirección IP del interfaz de cada contenedor de forma que estén en la misma subred aunque estén en hosts diferentes. Con eso ya deberían alcanzarse el uno al otro sin saltos IP enrutados aparentes y las tramas Ethernet se transportarán sobre GRE si las miramos con *tcpdump* en el interfaz de uno de los hosts. La red virtual es simplemente la que se muestra en la Figura 2.

Figura 2 - Red virtual

¿Qué direcciones MAC aprenderán ahora los conmutadores de la infraestuctura física entre los hosts?

¿Qué tendría que hacer para extender la red virtual a contenedores de otro host?

Punto de control 1 (45%): Muestre al profesor el escenario en funcionamiento, así como una traza de tráfico con paquetes con encapsulado GRE.

3. Overlay empleando VXLAN

En este caso configuraremos un escenario muy similar al anterior pero ahora la comunicación de las tramas Ethernet entre los hosts se hará sobre UDP, en concreto empleando VXLAN. Esta solución emplea un grupo multicast para el tráfico BUM (Broadcast, Unknown unicast, Multicast, más información en los materiales de teoría).

Empezamos de nuevo con los hosts sin configurar.

Dado que la comunicación entre los hosts es con paquetes IP quiere decir que de nuevo podría haber routers entre ellos (aunque en este ejemplo tengamos solo la internal network) y también significa que necesitamos asignar dirección IP a los interfaces del host. Asigne direcciones a los interfaces de los hosts para que haya conectividad IP entre ellos.

Creamos el interfaz encargado del tráfico VXLAN:

```
# ip link add vxlan0 type vxlan id <VNI> group <ip_multicast> dev <interfazInternalNetwork> dstport 4789
```

donde el significado de los parámetros es:

> *"id <grupoDePracticas>"* : VXLAN permite crear LANs sobre UDP. Cada LAN necesita su identificador numérico que se llama el VNI (VXLAN Network Identifier) y es un número de 24 bits (muchos más que las VLANs de 802.1Q)

"group <ip_multicast>" : VXLAN emplea un grupo multicast. Seleccione uno dentro del rango privado (239.0.0.0/8).

"<interfazInternalNetwork>" : El nombre del interfaz (ej: eth0) conectado a la internal network (por donde podrá comunicarse con el resto de hosts con contenedores de esta VXLAN)

El valor del puerto 4789 es el reservado por IANA para VXLAN[1].

Cree un bridge para la comunicación entre los contenedores en cada host y conecte algún contenedor en cada host a su bridge. Añada el interfaz VXLAN creado (en este caso lo hemos llamado vxlan0) al puente correspondiente y configure la dirección IP de los contenedores en la misma subred.

Repita la configuración en el otro host, asignando a los contenedores dirección IP en la misma subred que en el otro host. Pruebe a comunicar un contenedor de un host con otro de otro host (un ping por ejemplo) y vea en una de las VMs el intercambio de paquetes en el interfaz que va a la internal network. Es interesante que lance *tcpdump* antes de empezar la comunicación para poder ver los paquetes ARP sobre el grupo multicast.

¿Qué direcciones MAC aprenderán ahora los conmutadores de la infraestuctura física entre los hosts?

¿Qué ventajas tenemos frente el caso de túneles GRE?

Punto de control 2 (50%): Muestre al profesor el escenario en funcionamiento, así como una traza de tráfico donde se vea el ARP sobre multicast.

4. Enrutado entre VXLANs

Construya un escenario donde haya dos overlays VXLAN y un contenedor tenga interfaces en las dos, enrutando entre ellas.

Punto de control 3 (5%): Muestre al profesor el escenario en funcionamiento.

[1] https://tools.ietf.org/html/rfc7348