Departamento de Informática
Universidad Técnica Federico Santa María
Campus San Joaquín

Mininet y Openflow

Sebastián Campos
201773517-1

Axel Reyes
201773502-3

Agosto 2020

1 Primera Topología

La primera topología se encuentra en el archivo tp1.py, para cargarla a mininet se debe agregar el archivo a la carpeta custom, luego con la consola entrar a la carpeta custom y utilizar:

```
sudo mn --custom tp1.py --topo tp1
--controller=remote,ip=127.0.0.1,port=6633
```

En la topología, los host tienen los mismos nombres que los señalados en la tarea (Host a = ha, Host b = hb, etc), en cambio, los switches tienen los siguientes cambios: switch x = s1, switch y = s2, switch z = s3.

Para manejar el controlador OpenFlow se utilizará la implementación de POX realizada en Python, en caso de no tenerlo, se debe ejecutar la siguiente linea de comandos:

```
git clone http://github.com/noxrepo/pox
cd pox
```

Una vez en la carpeta de pox, se debe ejecutar pox.py con los siguientes flags:

```
python pox.py log.level --DEBUG misc.full_payload
    forwarding.l2_learning openflow.discovery openflow
    .spanning_tree --no-flood --hold-down
```

Donde:

- log.level –DEBUG : Habilita los mensajes de debugeo de POX, de forma que se pueda seguir el flujo del código.
- misc.full_payload : Esto ejecuta un script de POX el cual obliga a enviar los paquetes de forma completa. No es necesario para el funcionamiento general del programa, pero evita que se muestren por consola un spam de mensajes.

- forwarding.l2_learning : El algoritmo de forwarding que utilizará POX, para esta instancia se utilizará un algoritmo que viene implementado por defecto en POX llamado l2_learning, posteriormente se cambiará por otros algoritmos para las demás pruebas.
- openflow.discovery: Script de openflow necesario para la detección de switches.
- openflow.spanning-tree –no-flood –hold-down: Script que evita que los mensajes se queden atrapados en redes en las cuales las conexiones formen ciclos.

Ya con esto, se podría ejecutar el comando pingall y ver que todos los host pueden alcanzar a todos los host. (Nota: A veces el primer ping puede tardarse u otras veces no llegar, asi que se recomienda probar cada ping dos veces si la primera vez no se logró alcanzar el objetivo.)

Luego, para bajar un link se puede ejecutar el comando de mininet:

```
link s1 s2 down
```

Posteriormente, al hacer nuevamente pingall, podemos notar que todos los host aún puede llegar al resto de hosts, esto se debe a que al romperse el link, las tablas de forwarding buscan actualizarse enviando paquetes por el resto de links disponibles (que no sea el de entrada), por lo que si existe otro camino que comunique los switches, el controlador Openflow lo descubrirá y comenzará a enviar los paquetes por este nuevo camino. (Nota: Hay veces que el script l2_learning tarda en mostrar este comportamiento, por lo que si se ha intentado varias veces y no se logra comunicar todos los host, se recomienda utilizar el script l2_multi en su reemplazo, ejecutando el mismo código pox cambiando forwarding.l2_learning por forwarding.l2_multi.)

Ahora, para probar la red en un solo sentido, se debe agregar el script llamado "l2_learningAH.py" a la carpeta de forwarding de pox y ejecutar (NOTA: Antes de ejecutar se debe bajar la instancia del controlador anterior utilizando ctrl+c en la consola):

```
python pox.py log.level --DEBUG misc.full_payload
   forwarding.l2_learningAH openflow.discovery
   openflow.spanning_tree --no-flood --hold-down
```

Con esta versión del controlador, todos los mensajes se envían en sentido antihorario. Para comprobarlo, en la consola donde se está ejecutando POX hay unos mensajes DEBUG que señalan los puertos de entrada y de salida de los paquetes, para ver si está siguiendo el camino correcto. Se recomienda guiarse con el siguiente mapa:

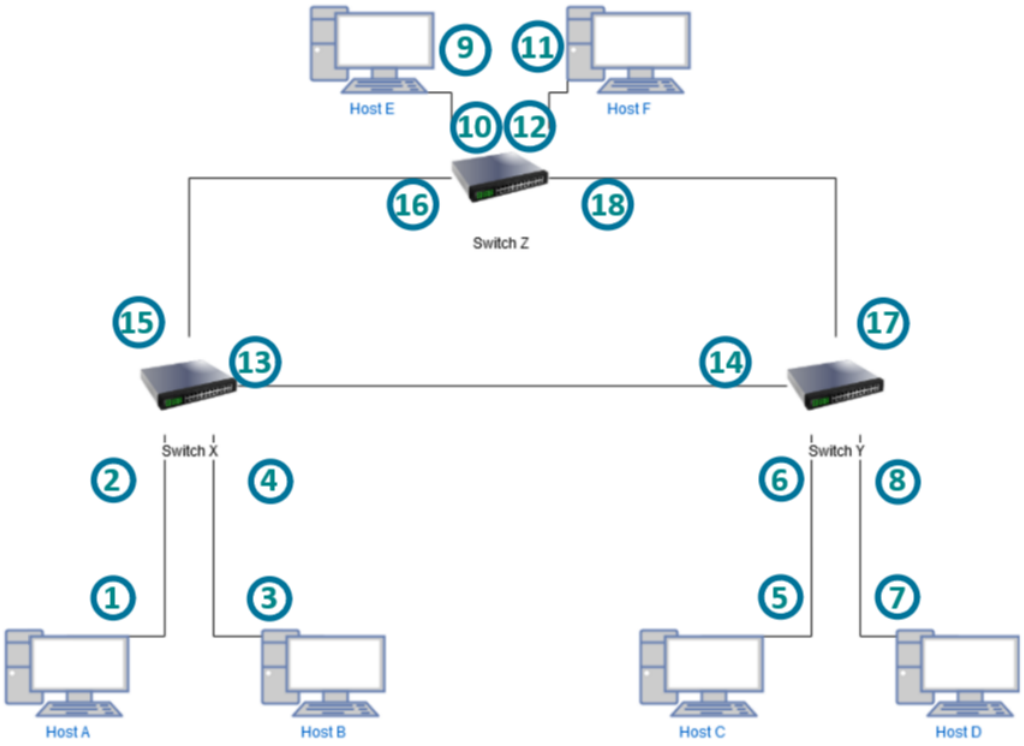

Figure 1: Puertos topología 1

Ahora, para maximizar el flujo entre los switches X e Y, se debe agregar el controlador llamado "l2_learningXY.py" a la carpeta de forwarding de pox y ejecutar el siguiente comando (NOTA: Antes de ejecutar se debe bajar la instancia del controlador anterior utilizando ctrl+c en la consola):

```
python pox.py log.level --DEBUG misc.full_payload
    forwarding.l2_learningXY openflow.discovery
    openflow.spanning_tree --no-flood --hold-down
```

Con esta versión, todos los mensajes pasan la conexión entre el switch X y el switch Y, a menos que el mensaje sea entre host del mismo switch (Por ejemplo, Host A con Host B). Para comprobarlo, en la consola de POX hay unos mensajes de DEBUG que señala los puertos de entrada y salida de los paquetes. Se recomienda utilizar el mapa de puertos señalado anteriormente

para guiarse.

Explicación: Para que los paquetes tomaran los caminos correspondiente, se modificó el reenvío de paquetes de forma que al querer llegar a un destino, tomara el puerto que queríamos en vez del puerto que llevara al camino más corto. Este nuevo forwarding se añadió en el script original de l2_learning al final del método _handle_PacketIn, creando los dos nuevos script llamados l2_learningAH y l2_learningXY.

Para evitar la conexión de host externos, se añadió una 'whitelist' dentro del código de cada controlador mencionado anteriormente, que contiene todas las mac address permitidas en la red (desde la 00:00:00:00:00:01 a la 00:00:00:00:00:06), de forma que si un paquete proviene o va dirigido a una mac que no esté en la lista, se dropea.

2 Topología 2

La segunda topología se encuentra en el archivo tp2.py. Para cargarla a mininet, se debe agregar el archivo a la carpeta custom, luego con la consola se debe entrar a la carpeta custom y utilizar:

```
sudo mn --custom tp2.py --topo tp2
--controller=remote,ip=127.0.0.1,port=6633
```

En la topología, los hosts tienen los mismos nombres que los señalados en la tarea (Host a = ha, Host b = hb, etc), en cambio los switches tienen los siguientes cambios:

switch $v = s1$
switch $x = s2$
switch $z = s3$
switch $y = s4$
switch $w = s5$
switch $u = s6$
switch $t = s7$

Para el caso del controlador, se debe añadir el archivo l2_learning2.py a la carpeta forwarding de pox y ejecutar:

```
python pox.py log.level --DEBUG misc.full_payload
   forwarding.l2_learning2 openflow.discovery
   openflow.spanning_tree --no-flood --hold-down
```

Tal como se señaló en la tarea, solo se permite intercambiar mensajes de tipo HTTP entre alguno de los hosts y el servidor (Host G). Para montar el servidor HTTP, se debe realizar el siguiente comando en la consola de mininet:

```
hg python -m SimpleHTTPServer 80 &
```

De esta forma se montará un servidor HTTP en el Host G. Para realizar una petición GET HTTP se debe ejecutar:

```
ha wget -O - hg
```

Donde "ha" puede ser remplazado por cualquier de los host de la red que no sea "hg" (hb, hc, hd, etc). Para ver si la direccionalidad está bien, en la consola de POX se señalan los puertos de entrada y salida de los paquetes en forma de mensajes DEBUG. Para guiarse de mejor manera, se recomienda utilizar el siguiente mapa:

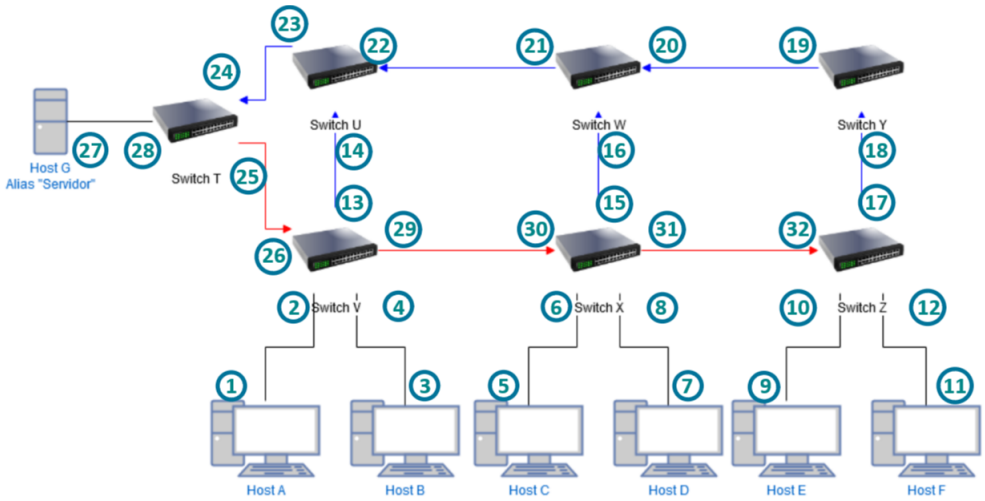

Figure 2: Puertos topología 2

Al igual que la topología anterior, se realizó una whitelist con todas las mac address permitidas en la red (desde la 00:00:00:00:00:01 a la 00:00:00:00:00:07), dropeando los mensajes que van dirigidos o son enviados por una mac address que esté fuera de esa lista.

Explicación: Para hacer los links unidireccionales, se realizó algo similar a lo realizado en la primera topología, redireccionando los mensajes por el puerto del switch correcto para llegar al destino. Para evitar que los hosts se comuniquen entre sí, todo mensaje que no sea enviado por la mac address del servidor o hacia la mac address del servidor, es dropeado. Finalmente, el firewall se construyó revisando si el paquete que recibe el servidor corresponde a un mensaje de tipo TCP (SYN o ACK) y si el mensaje recibido corresponde a una petición GET. Si cumple ambas condiciones, el mensaje es aceptado y responde con un mensaje HTTP.

Como se entendió en la tarea, los hosts no deberían poder hacer ping al servidor ni este de vuelta, por lo que si se desea realizar ping (para comprobar que los host se comunican con quien deberían), se puede comentar desde la línea

234 hasta las 247 del archivo llamado l2_learning2.py, de esta forma se desactiva el firewall del server y podrá recibir pings.

3 Consideraciones

- Para comprobar el funcionamiento de la whitelist, se debe agregar algún host externo con mac y número de puertos distintos a los ya registrados.
- En ciertos casos, los pings envían dos o más veces.
- Si se utiliza el controlador l2.multi, tener en cuenta de que el primer pingall fallará en un host, luego para el segundo ya habrá aprendido el nuevo recorrido.
- En el comprimido se enviará tambien el script l2_multi y l2_learning, en principio no deberían ser necesarios ya que al clonar el repositorio de pox estos vienen incluidos en la carpeta forwarding, pero se envían por si acaso.