



# TP N°2: Software-Defined Networks

[TA048] Redes  
Segundo cuatrimestre 2024  
*Grupo 3*

## Índice

<b>1. Introducción</b>	<b>2</b>
<b>2. Mininet</b>	<b>2</b>
<b>3. Firewall</b>	<b>2</b>
<b>4. Ejecución</b>	<b>3</b>
<b>5. Pruebas</b>	<b>3</b>
5.1. Se deben descartar todos los mensajes cuyo puerto destino sea 80 . . . . .	3
5.2. Se deben descartar todos los mensajes que provengan del host 1, tengan como puerto destino el 5001, y estén utilizando el protocolo UDP. . . . .	4
5.3. Se deben elegir dos hosts cualesquiera y los mismos no deben poder comunicarse de ninguna forma. . . . .	4
<b>6. Preguntas a responder</b>	<b>4</b>
6.1. ¿Cuál es la diferencia entre un Switch y un router? ¿Qué tienen en común? . . . .	4
6.2. ¿Cuál es la diferencia entre un Switch convencional y un Switch OpenFlow? . . . .	5
6.3. ¿Se pueden reemplazar todos los routers de la Internet por Switches OpenFlow? Piense en el escenario interASes para elaborar su respuesta . . . . .	5

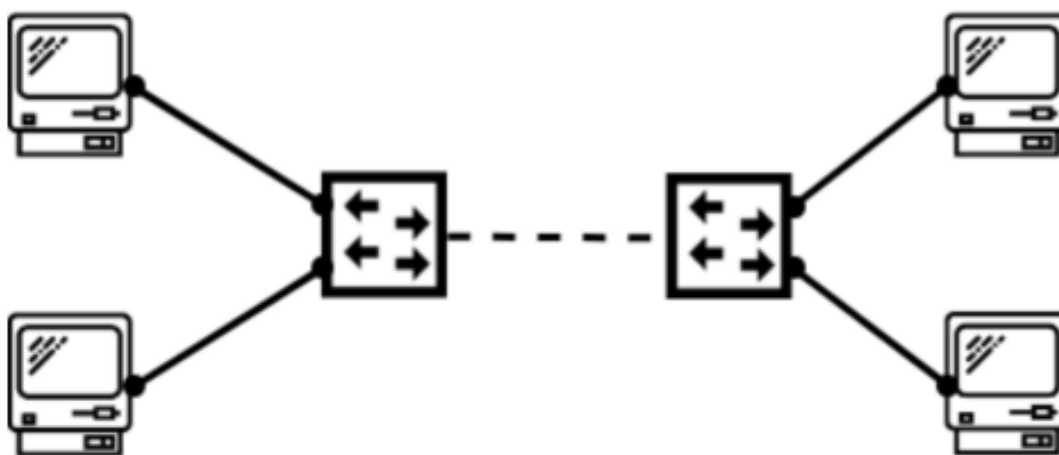
## 1. Introducción

El objetivo del trabajo es explorar los principios de las Software-Defined Networks (SDNs) y el protocolo OpenFlow mediante la construcción de una topología dinámica. Para ello, se empleará el simulador Mininet y se implementará un controlador capaz de gestionar un Firewall a nivel de capa de enlace. Este enfoque permitirá emular el comportamiento de dispositivos de red programables y aplicar políticas específicas de ruteo y seguridad.

## 2. Mininet

Se diseñó una topología en Mininet compuesta por dos hosts ubicados en los extremos, conectados mediante una cadena de switches de longitud  $n$ , siendo este un parámetro configurable al momento de iniciar la simulación.

```
sudo mn -custom mininet.py -topo=chain,n -mac -arp -switch ovsk -controller remote
```



## 3. Firewall

El firewall fue desarrollado utilizando POX y se encarga de aplicar las siguientes reglas:

- Se deben descartar todos los mensajes cuyo puerto destino sea 80.
- Se deben descartar todos los mensajes que provengan del host 1, tengan como puerto el 5001, y estén utilizando el protocolo UDP
- Se deben elegir dos hosts cualesquiera y los mismos no deben poder comunicarse de ninguna forma.

Las reglas se definieron en un archivo .json con el siguiente formato, diseñado para facilitar la adición, eliminación o modificación de reglas existentes:

```
{  
  "switches": [switch ID],  
  "tp_proto": udp/tcp,  
  "tp_dst": puerto destino,  
  "dl_src": mac fuente,  
  "dl_dst": mac destino  
}
```

## 4. Ejecución

Es necesario tener instalados Mininet, Open vSwitch, Xterm y POX. Con todo configurado, se deben abrir dos terminales. En la primera, se inicia el controlador con el siguiente comando:

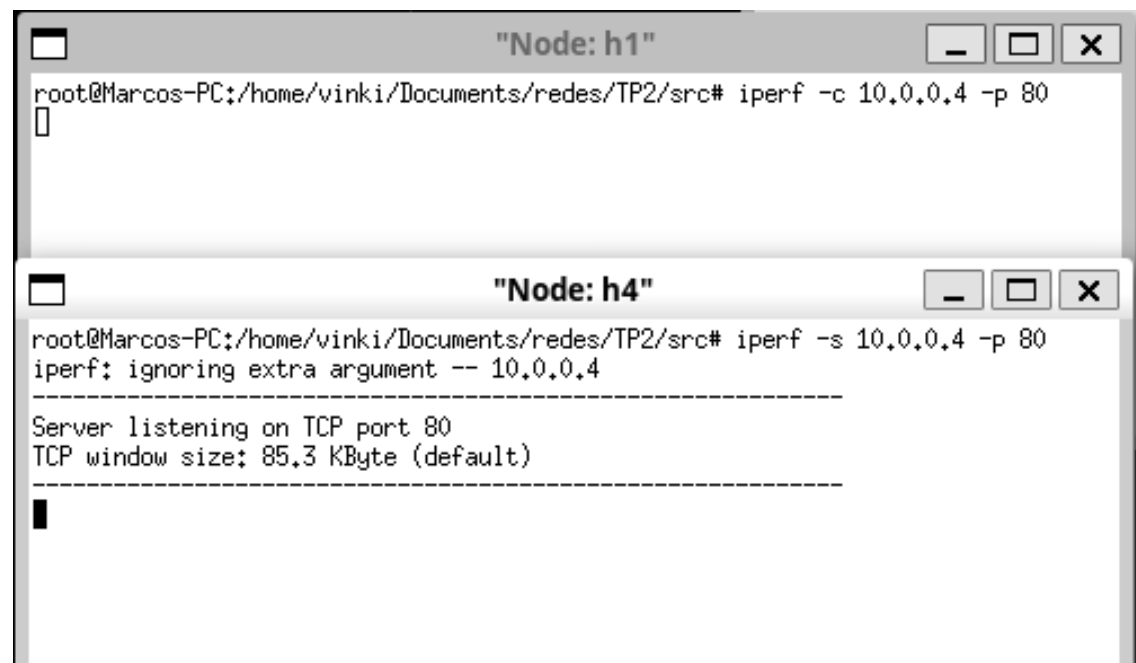
```
pox/pox.py log.level --DEBUG forwarding.l2_learning controller
```

Posteriormente, en la segunda terminal, se levanta la topología utilizando la siguiente línea (sustituyendo n por la cantidad de switches que debe contener):

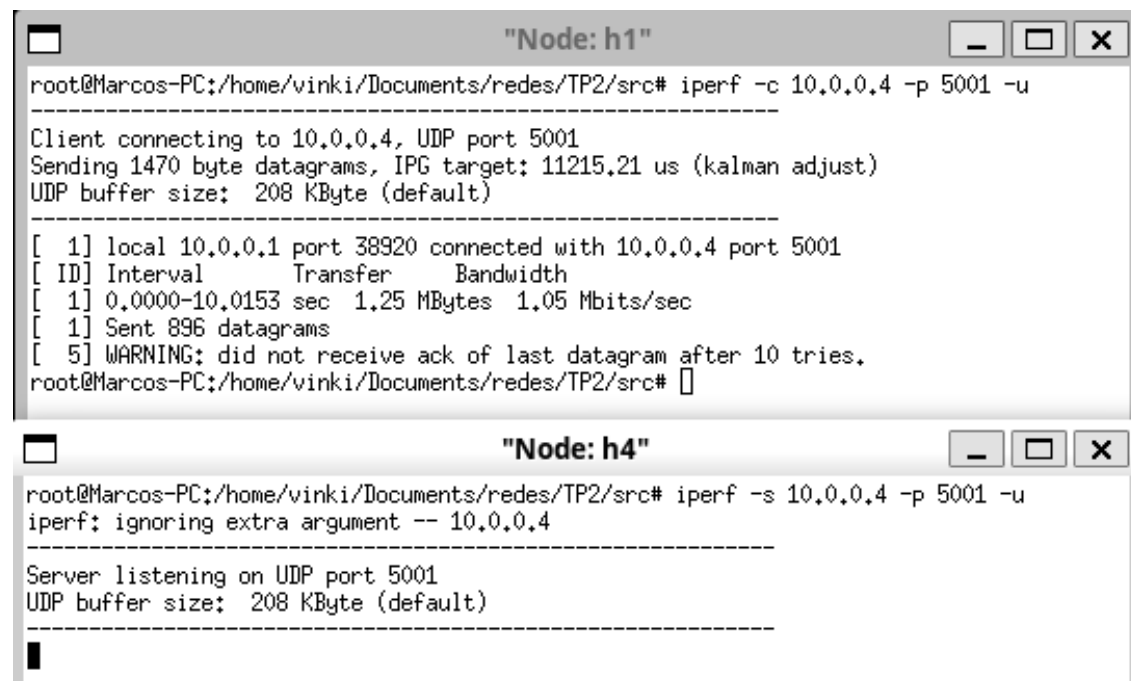
```
mininet.py --topo=chain,n --mac --arp --switch ovsk --controller remote
```

## 5. Pruebas

### 5.1. Se deben descartar todos los mensajes cuyo puerto destino sea 80



- 5.2. Se deben descartar todos los mensajes que provengan del host 1, tengan como puerto destino el 5001, y estén utilizando el protocolo UDP.



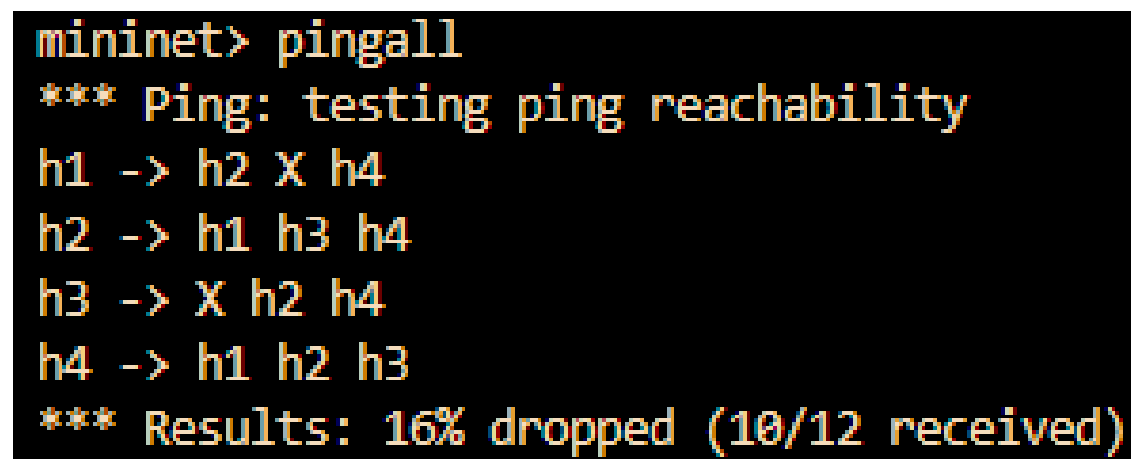
```

"Node: h1"
root@Marcos-PC:/home/vinki/Documents/redes/TP2/src# iperf -c 10.0.0.4 -p 5001 -u
-----
Client connecting to 10.0.0.4, UDP port 5001
Sending 1470 byte datagrams, IPG target: 11215.21 us (kalman adjust)
UDP buffer size: 208 KByte (default)
-----
[ 1] local 10.0.0.1 port 38920 connected with 10.0.0.4 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 1] 0.0000-10.0153 sec 1.25 MBytes 1.05 Mbits/sec
[ 1] Sent 896 datagrams
[ 5] WARNING: did not receive ack of last datagram after 10 tries.
root@Marcos-PC:/home/vinki/Documents/redes/TP2/src#

"Node: h4"
root@Marcos-PC:/home/vinki/Documents/redes/TP2/src# iperf -s 10.0.0.4 -p 5001 -u
iperf: ignoring extra argument -- 10.0.0.4
-----
Server listening on UDP port 5001
UDP buffer size: 208 KByte (default)
-----

```

- 5.3. Se deben elegir dos hosts cualesquiera y los mismos no deben poder comunicarse de ninguna forma.



```

mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 X h4
h2 -> h1 h3 h4
h3 -> X h2 h4
h4 -> h1 h2 h3
*** Results: 16% dropped (10/12 received)

```

## 6. Preguntas a responder

- 6.1. ¿Cuál es la diferencia entre un Switch y un router? ¿Qué tienen en común?

Los switches y los routers son dispositivos aislados fundamentales para el funcionamiento del internet dado que un switch opera en la capa de enlace de datos (capa 2) y se encarga de interconectar dispositivos dentro de una misma red local (LAN) utilizando direcciones MAC para forwardear

paquetes, mientras que por otro lado, un router opera en la capa de red (capa 3) y tiene la función de interconectar diferentes redes, utilizando direcciones IP para determinar la ruta óptima para forwardear datos, lo que permite la comunicación entre redes locales e incluso con internet. Ambos tienen conceptos similares, como el uso de tablas internas para tomar decisiones de forwarding y la capacidad de manejar múltiples dispositivos, pero difieren en su propósito y alcance. Mientras que los switches optimizan el tráfico dentro de una red local, los routers gestionan el tráfico entre redes distintas, siendo complementarios en la configuración de una infraestructura de red completa.

## **6.2. ¿Cuál es la diferencia entre un Switch convencional y un Switch OpenFlow?**

Un switch convencional y un switch OpenFlow se diferencian principalmente en su arquitectura y funcionalidad. Los switches convencionales combinan el plano de control y el plano de datos en el mismo dispositivo, lo que significa que las decisiones de forwarding y las operaciones de conmutación, como la construcción de tablas MAC, se realizan internamente de manera automática y están determinadas por el firmware del fabricante, limitando así su flexibilidad. En contraste, los switches OpenFlow separan el plano de control del plano de datos, permitiendo que un controlador centralizado gestione las decisiones de forwarding y configure dinámicamente las reglas en las tablas de flujo del switch, lo que proporciona una alta programabilidad y adaptabilidad a las necesidades específicas de la red. Esta separación en los switches OpenFlow permite una administración centralizada y facilita la implementación de arquitecturas de redes definidas por software (SDN), en las cuales la red se puede controlar de forma dinámica y global. Aunque los switches convencionales son más simples y económicos, los switches OpenFlow ofrecen mayor flexibilidad y escalabilidad, a expensas de una mayor complejidad inicial y costos asociados. La elección entre ambos depende de los requisitos específicos y la escala de la red en cuestión.

## **6.3. ¿Se pueden reemplazar todos los routers de la Internet por Switches OpenFlow? Piense en el escenario interASes para elaborar su respuesta**

No es posible reemplazar todos los routers de Internet por switches OpenFlow debido a que los switches OpenFlow están diseñados para operar bajo un paradigma centralizado de control, donde el plano de control está separado del plano de datos y aunque OpenFlow permite una gran flexibilidad para definir reglas de forwarding, su enfoque está orientado principalmente a la gestión de redes dentro de un dominio controlado, como un único centro de datos o red privada, y no para escenarios complejos de enrutamiento global como los que existen en Internet. Además, en un escenario interAS, la separación administrativa entre diferentes ASes hace que sea inviable delegar el control del tráfico a un controlador OpenFlow centralizado. Por lo tanto, aunque los switches OpenFlow pueden complementar ciertas partes de la infraestructura de red, no pueden reemplazar la funcionalidad y las capacidades de los routers en el contexto de Internet y el enrutamiento interAS.