

ENCAMINAMIENTO CON MININET

ALUMNO: MIGUEL CÓRDOBA ARANDA
CURSO: 1º ASIR
PLANIFICACIÓN Y ADMINISTRACIÓN DE REDES

ÍNDICE

INTRODUCCIÓN

ESCENARIO MININET

DESCARGAR MAQUINA VIRTUAL MININET

ACCEDER A LA MAQUINA MININET

CARGAR ESCENARIO Y ELEMENTOS

CONFIGURACIÓN Y COMPROBACIÓN

AÑADIR IP A LAS INTERFACES DE RED

AÑADIR RUTA DE ENCAMINAMIENTO A LOS HOSTS

CONECTIVIDAD Y CAPTURA DE TRAFICO EN LA RED

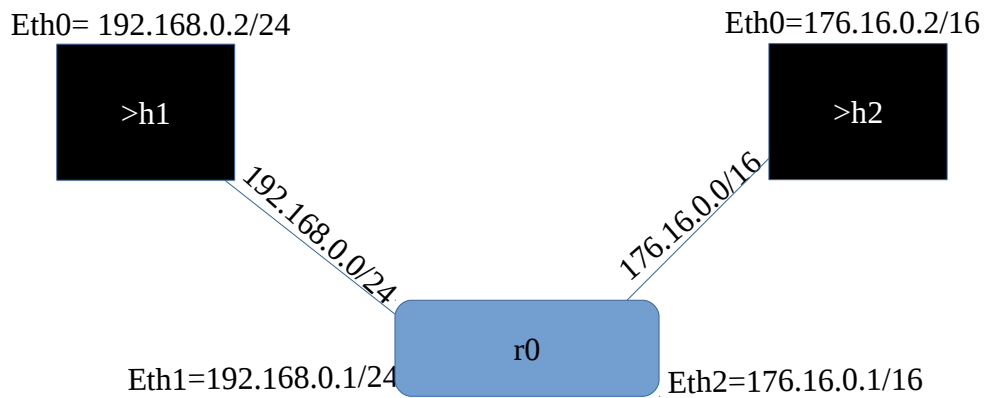
INTRODUCCIÓN

En este documento vamos a explicar paso a paso como configuraremos un escenario hecho en python que consta de cinco elementos (dos hosts, dos switch y un router). Para empezar mostraremos el esquema de este escenario y a continuación iremos mostrando paso a paso como configurarla hasta que todos los elementos estén enrutados (conectados) unos con otros, durante el proceso se mostraran capturas para que se vea que realmente la siguiente configuración es efectiva.

ESCENARIO MININET

Los elementos en el siguiente esquema son los siguientes:

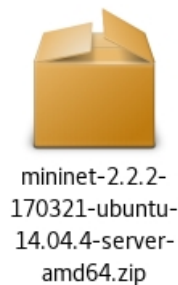
- 2 Host identificados como h1 y h2.
- 1 Router identificado como r0



DESCARGA DE MAQUINA VIRTUAL MININET

Para descargarte la maquina virtual Mininet vamos a la pagina web www.mininet.org y en el apartado **Get Started** hacemos click en **Download**.

Una vez descargada la maquina virtual nos saldrá un fichero comprimido llamado **mininet-2.2.2-170321-ubuntu-14.04.4-server-amd64.zip**, el cual hay que descomprimir y nos aparecerá un fichero con el nombre de **mininet-2.2.2-170321-ubuntu-14.04.4-server-amd64** en su interior se encuentra el fichero **mininet-2.2.2-170321-ubuntu-14.04.4-server-amd64.ovf** que sera importado en **VirtualBox** para que nos cree la máquina virtual mininet.



Una vez importada la maquina virtual MININET en **VirtualBox** te creará una maquina virtual con el nombre Mininet-VM con la que trabajaremos posteriormente para configurar nuestra red.

!!!IMPORTANTE!!!! → CONFIGURARLA EN MODO PUENTE

ACCEDER A LA MAQUINA MININET

Para acceder a la maquina Mininet-VM primero tenemos que inicializarla, una vez iniciada



Escenario1. MININET

Nos pedirá un usuario y una contraseña para acceder a la maquina. Por defecto el **usuario** es **mininet** y la **contraseña mininet**. Con este último paso tenemos la Mininet-VM totalmente iniciada.

El siguiente paso es **comprobar la IP** que se le ha asignado mediante el comando **#ip a** para que posteriormente hagamos una **conexión ssh** para cargar un escenario y configurarlo. Nos muestra que la IP de nuestra Mininet-VM es 172.22.1.32/16, con esta IP desde una terminal en la maquina anfitriona haremos una conexión ssh (remota) a nuestra Mininet-VM.

Para conectarnos en la terminal de la maquina anfitriona introduciremos el siguiente comando **#ssh -X -v mininet@172.22.1.32** y seguidamente nos pedirá una (mininet) y ya estaremos dentro de nuestra Mininet-VM.

```
miguel@MiguelCordoba: ~  
miguel@MiguelCordoba: ~ 80x24  
miguel@MiguelCordoba:~$ ssh -X -v mininet@172.22.1.32  
debug1: Next authentication method: password  
mininet@172.22.1.32's password: 
```

CARGA DE ESCENARIO Y ELEMENTOS

Acto seguido tendremos un archivo python llamado “escenario1.py” en nuestra maquina anfitriona, el cual copiaremos y cargaremos en nuestra Mininet-VM.



Para copiarlo, lo haremos desde nuestra maquina anfitriona mediante el comando.

#scp ruta_del_escenario1.py mininet@ip:ruta_de_destino.

Para comprobar que se ha copiado satisfactoriamente no vamos a nuestra Mininet-VM y ejecutamos el comando **#ls**.

```
mininet@mininet-vm: ~  
mininet@mininet-vm: ~ 80x24  
mininet@mininet-vm:~$ ls  
escenario1.py  loxigen  oflops  openflow  zdf.mn  
install-mininet-vm.sh  mininet  oftest  pox  
mininet@mininet-vm:~$ 
```

Escenario1. MININET

Una vez comprobado procederemos a cargar nuestro escenario para configurar sus elementos. Para cargar el escenario usaremos el comando **# *sudo python escenario1.py***.

```
mininet@mininet-vm: ~  
mininet@mininet-vm: ~ 80x24  
mininet@mininet-vm:~$ sudo python escenario1.py  
*** Creating network  
*** Adding controller  
*** Adding hosts:  
h1 h2 r0  
*** Adding switches:  
s1 s2  
*** Adding links:  
(h1, s1) (h2, s2) (s1, r0) (s2, r0)  
*** Configuring hosts  
h1 h2 r0  
*** Starting controller  
c0  
*** Starting 2 switches  
s1 s2 ...  
*** Routing Table on Router:  
Kernel IP routing table  
Destination      Gateway          Genmask          Flags Metric Ref    Use Iface  
  
*** Starting CLI:  
mininet> 
```

A continuación cargaremos los dos hosts (h1 y h2) y el router (r0) con el comando **xterm**. Siendo el comando completo **#*xterm nombre_del_elemento***.

```
mininet> xterm h1  
mininet> debug1: client_input_channel_open: ctype x11 rchan 2 win 65536 max 1638  
4  
debug1: client_request_x11: request from 127.0.0.1 51842  
debug1: channel 1: new [x11]  
debug1: confirm x11
```

```
"Node: h1" (on mininet-vm) - □ x  
root@mininet-vm:~# 
```

Este cuadro de dialogo nos muestra que se ha cargado correctamente el elemento h1 y nos aparecerá una ventana como esta:

Escenario1. MININET

Cuando tengamos cargados todos los elementos procederemos a hacer las comprobaciones oportunas y configuraciones.

CONFIGURACIÓN Y COMPROBACIÓN

El primer paso sera comprobar las interfaces de red de los elementos utilizando el comando `#ip a`. Y vemos que no tiene IP en la interfaz **h1-eth0@if12**.

```
"Node: h1" (on mininet-virtual-machine)
root@mininet-virtual-machine:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: h1-eth0@if12: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 3a:4c:81:c2:17:ef brd ff:ff:ff:ff:ff:ff
root@mininet-virtual-machine:~#
```

```
"Node: h2" (on mininet-virtual-machine)
root@mininet-virtual-machine:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: h2-eth0@if13: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 36:ba:a0:9d:21:fd brd ff:ff:ff:ff:ff:ff
root@mininet-virtual-machine:~#
```

```
"Node: r0" (on mininet-virtual-machine)
root@mininet-virtual-machine:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: r0-eth1@if14: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether c2:a5:2b:de:36:31 brd ff:ff:ff:ff:ff:ff
3: r0-eth2@if15: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether de:6f:51:d0:86:bf brd ff:ff:ff:ff:ff:ff
root@mininet-virtual-machine:~#
```


AÑADIR IP A LAS INTERFACES DE RED

Para asignarle una IP a esa interfaz utilizaremos el comando **#ip a add** con la IP que queramos asignarle, en mi caso utilizare la 192.168.0.2/24 . Como resultados obtendremos el comando **#ip a add 192.168.0.2/24 dev h1-eth0** .

```
"Node: h1" (on mininet-virtual-machine)
root@mininet-vm:~# ip a add 192.168.0.2/24 dev h1-eth0
root@mininet-vm:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: h1-eth0@if12: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 3a:4c:81:c2:17:ef brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.2/24 scope global h1-eth0
        valid_lft forever preferred_lft forever
root@mininet-vm:~#
```

#ip a add 176.16.0.2/16 dev h2-eth0 en el h2.

```
"Node: h2" (on mininet-virtual-machine)
root@mininet-vm:~# ip a add 176.16.0.2/16 dev h2-eth0
root@mininet-vm:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: h2-eth0@if13: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 36:ba:a0:9d:21:fd brd ff:ff:ff:ff:ff:ff
    inet 176.16.0.2/16 scope global h2-eth0
        valid_lft forever preferred_lft forever
root@mininet-vm:~#
```

En el caso del router(r0) tenemos dos interfaces de red [r0-eth1@if14](#) y [r0-eth2@if15](#) . Por lo que utilizaremos: **#ip a add 192.168.0.1/24 dev r0-eth1** y **#ip a add 176.16.0.1/16 dev r0-eth2**

```
"Node: r0" (on mininet-virtual-machine)
root@mininet-vm:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: r0-eth1@if14: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether c2:a5:2b:de:36:31 brd ff:ff:ff:ff:ff:ff
3: r0-eth2@if15: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether de:6f:51:d0:86:bf brd ff:ff:ff:ff:ff:ff
root@mininet-vm:~#
```

Escenario1. MININET

Una vez asignadas las IP's tanto a los dos hosts como al router procederemos a asignarle una ruta de encaminamiento para que h1 se comunique con h2 y viceversa.

AÑADIR RUTA DE ENCAMINAMIENTO A LOS HOSTS

Para empezar con este apartado lo primero que haremos sera comprobar las rutas de encaminamiento que se le asignan por defecto a los diferentes elementos por el simple echo de asignarles una IP. Utilizaremos el comando `#ip r` para ver la tabla de encaminamiento.

```
"Node: h1" (on mininet-virtual-machine)
root@mininet-virtual-machine:~# ip r
192.168.0.0/24 dev h1-eth0 proto kernel scope link src 192.168.0.2
root@mininet-virtual-machine:~#
```

Esta linea nos indica que nuestro **h1** a través de **eth0** esta conectada a una red 192.168.0.0/24 y nosotros le añadiremos una nueva ruta de encaminamiento para que tengamos **conectividad con h2**.

Para añadirle una nueva ruta de encaminamiento pondremos `#ip r add 176.16.0.2 via 192.168.0.1` .

```
"Node: h1" (on mininet-virtual-machine)
root@mininet-virtual-machine:~# ip r
176.16.0.2 via 192.168.0.1 dev h1-eth0
192.168.0.0/24 dev h1-eth0 proto kernel scope link src 192.168.0.2
root@mininet-virtual-machine:~#
```

Con ese comando le hemos indicado a nuestro **h1** que queremos ir a la 176.16.0.2 por la 192.168.0.1 que es la **interfaz de red del router** con la que se comunica **h1**.

A continuación haremos lo mismo en la h2 pero esta vez usando :
`#ip r add 192.168.0.2 via 176.16.0.1` .

```
"Node: h2" (on mininet-virtual-machine)
root@mininet-virtual-machine:~# ip r
176.16.0.0/16 dev h2-eth0 proto kernel scope link src 176.16.0.2
192.168.0.2 via 176.16.0.1 dev h2-eth0
root@mininet-virtual-machine:~#
```

El siguiente paso sería configurar **r0** pero como este conoce las dos redes pues no es necesario configurarle nada más para que haya conectividad entre **h1** y **h2**.

```
"Node: r0" (on mininet-virtual-machine)
root@mininet-virtual-machine:~# ip r
176.16.0.0/16 dev r0-eth2 proto kernel scope link src 176.16.0.1
192.168.0.0/24 dev r0-eth1 proto kernel scope link src 192.168.0.1
root@mininet-virtual-machine:~#
```

CONECTIVIDAD Y CAPTURA DE TRAFICO EN LA RED

Para comprobar la conectividad entre **h1** y **h2** nos vamos al terminal de una de las dos maquinas y utilizamos el comando ping. Si estamos en la **h1** el comando seria **#ping 176.16.0.2** (ip de h2) y si estamos en la **h2** el comando seria **#ping 192.168.0.2** (ip de h1).

```
"Node: h1" (on mininet-vm)
root@mininet-vm:~# ping 176.16.0.2 -c 5
PING 176.16.0.2 (176.16.0.2) 56(84) bytes of data.
64 bytes from 176.16.0.2: icmp_seq=1 ttl=63 time=1.97 ms
64 bytes from 176.16.0.2: icmp_seq=2 ttl=63 time=0.167 ms
64 bytes from 176.16.0.2: icmp_seq=3 ttl=63 time=0.123 ms
64 bytes from 176.16.0.2: icmp_seq=4 ttl=63 time=0.132 ms
64 bytes from 176.16.0.2: icmp_seq=5 ttl=63 time=0.124 ms

--- 176.16.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4001ms
rtt min/avg/max/mdev = 0.123/0.503/1.970/0.733 ms
root@mininet-vm:~#
```

Vemos que **h1** y **h2** tienen conectividad porque el parámetro “**time**” nos indica el tiempo que tarda nuestra **h1** en enviar un **paquete ICMP** y recibirlo. Con la opción **-c** indico cuantos paquetes quiero enviar.

Para capturar el tráfico de la red nos hemos ido a nuestro **router (r0)** y hemos ejecutado el comando **#tcpdump** para ver el tráfico que existe en la red.

```
"Node: r0" (on mininet-vm)
length 64
17:37:40.504301 IP 192.168.0.2 > 176.16.0.2: ICMP echo reply, id 2725, seq 43, l
length 64
17:37:41.178468 IP 192.168.0.2 > 176.16.0.2: ICMP echo request, id 2611, seq 297
, length 64
17:37:41.178562 IP 176.16.0.2 > 192.168.0.2: ICMP echo reply, id 2611, seq 297,
length 64
17:37:41.505598 IP 176.16.0.2 > 192.168.0.2: ICMP echo request, id 2725, seq 44,
length 64
17:37:41.505656 IP 192.168.0.2 > 176.16.0.2: ICMP echo reply, id 2725, seq 44, l
length 64
17:37:42.181209 IP 192.168.0.2 > 176.16.0.2: ICMP echo request, id 2611, seq 298
, length 64
17:37:42.181283 IP 176.16.0.2 > 192.168.0.2: ICMP echo reply, id 2611, seq 298,
length 64
17:37:42.508052 IP 176.16.0.2 > 192.168.0.2: ICMP echo request, id 2725, seq 45,
length 64
17:37:42.508110 IP 192.168.0.2 > 176.16.0.2: ICMP echo reply, id 2725, seq 45, l
length 64
17:37:43.180199 IP 192.168.0.2 > 176.16.0.2: ICMP echo request, id 2611, seq 299
, length 64
17:37:43.180271 IP 176.16.0.2 > 192.168.0.2: ICMP echo reply, id 2611, seq 299,
length 64
```