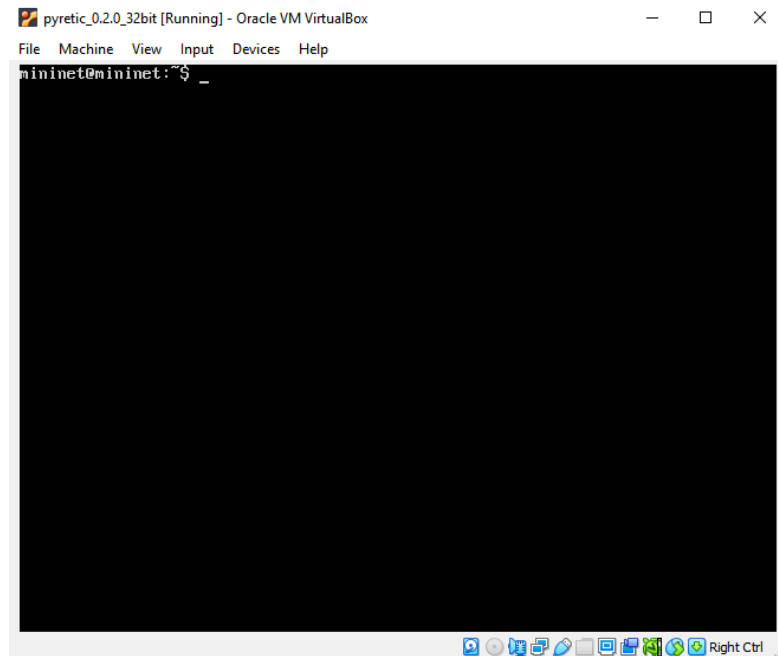


۱. در ابتدا، نرم افزار شبیه ساز VirtualBox به همراه فایل .ovf. دانلود و سیستم عامل راه اندازی شد.



۲. در این قسمت باید سیستم عامل مجازی را راه اندازی کنیم. ابتدا از وجود دو واسط مطمئن می شویم:

```
pyretic_0.2.0_32bit [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
mininet@mininet:~$ ifconfig -a
eth0      Link encap:Ethernet  HWaddr 08:00:27:04:78:8b
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe04:788b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:22116 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13139 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:30948807 (30.9 MB)  TX bytes:848878 (848.8 KB)

eth2      Link encap:Ethernet  HWaddr 08:00:27:61:db:a0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

mininet@mininet:~$ _
```

واسط eth2 آدرس ip ندارد؛ با دستور `sudo dhclient eth2` به آن آدرس می دهیم.

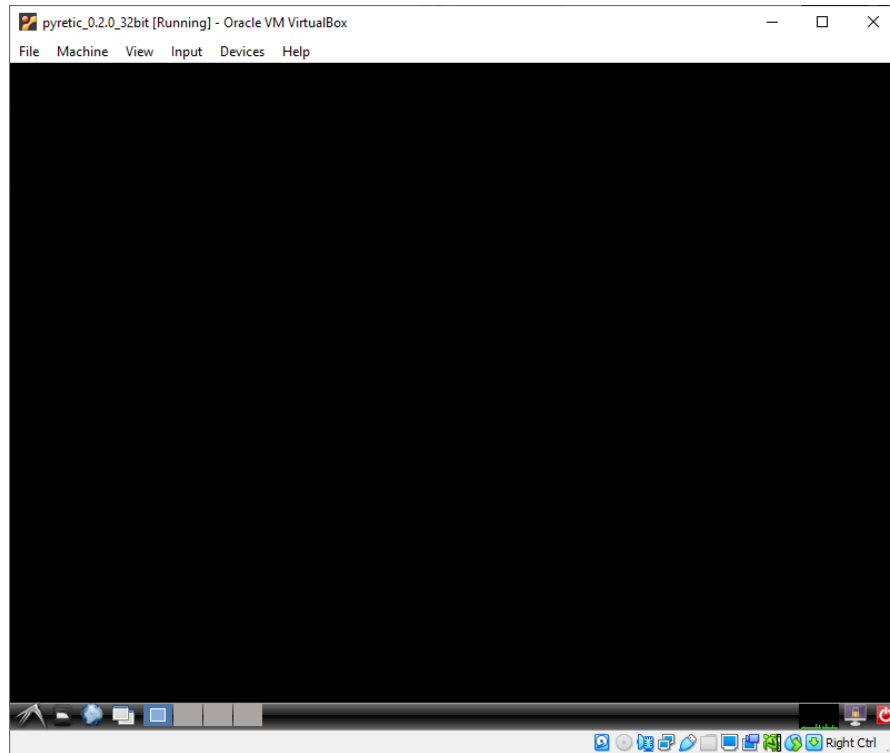
```
pyretic_0.2.0_32bit [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
mininet@mininet:~$ ifconfig -a
eth0      Link encap:Ethernet  HWaddr 08:00:27:04:78:8b
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe04:788b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:29 errors:0 dropped:0 overruns:0 frame:0
          TX packets:35 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3282 (3.2 KB)  TX bytes:3126 (3.1 KB)

eth2      Link encap:Ethernet  HWaddr 08:00:27:61:db:a0
          inet addr:192.168.56.101  Bcast:192.168.56.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe61:dbaa/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1180 (1.1 KB)  TX bytes:992 (992.0 B)

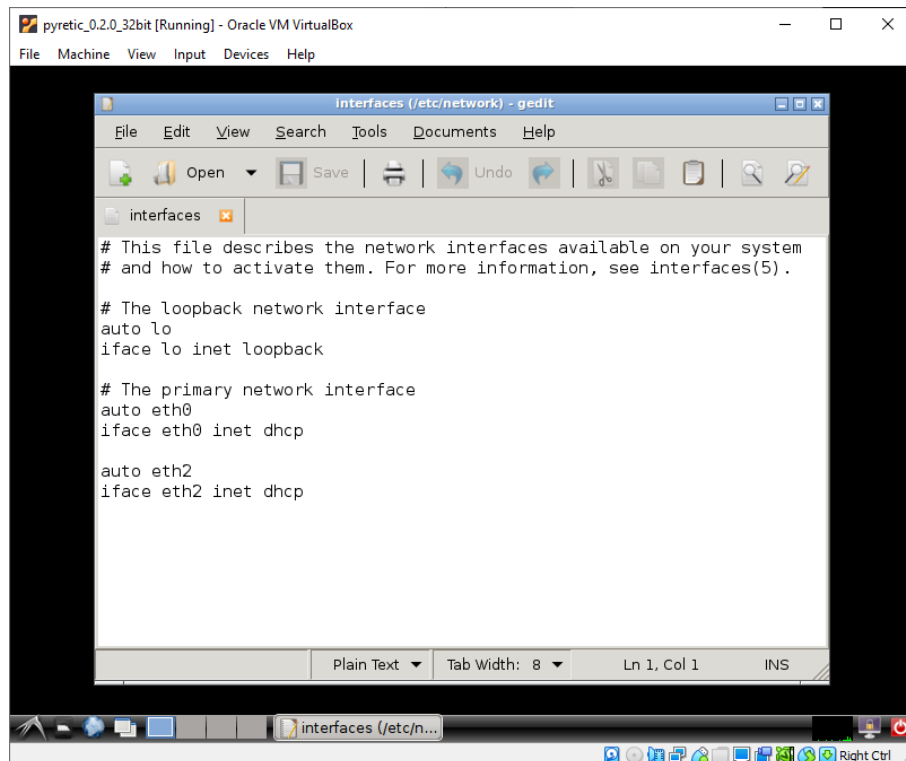
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

mininet@mininet:~$ _
```

۳. با نصب یک رابط کاربری گرافیکی، سیستم عامل آماده است.

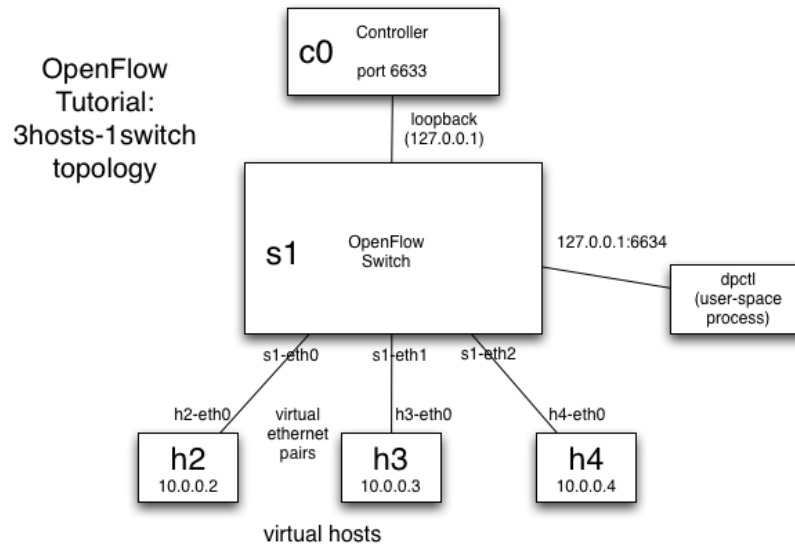


سپس فایل `/etc/network/interfaces` را تغییر می دهیم.



با اضافه کردن دو خط آخر، دیگر احتیاجی به اجرای دستور `sudo dhclient eth2`، هر بار که سیستم شروع به کار می کند، نیست.

۴. در این مرحله، می خواهیم یک شبکه به مانند شبکه زیر تولید کنیم:



با وارد کردن دستور `sudo mn --topo single,3 --mac --switch ovsk --controller remote` در ترمینال، شبکه‌ی مذکور تولید می شود.

```

pyretic_0.2.0_32bit [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

mininet@mininet: ~
mininet@mininet:~$ sudo mn --topo single,3 --mac --switch ovsk --controller remote
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6633
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
*** Starting 1 switches
s1
*** Starting CLI:
mininet>

```

سپس، از h1، h2 را ping می‌کنیم:

```

pyretic_0.2.0_32bit [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

mininet@mininet: ~
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6633
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
*** Starting 1 switches
s1
*** Starting CLI:
mininet> h1 ping -c3 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data:
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable

--- 10.0.0.2 ping statistics ---
3 packets transmitted, 0 received, +3 errors, 100% packet loss, time 2014ms
pipe 3
mininet>

```

از آن جایی که flow table خالی است، پاسخی دریافت نمی‌شود.
از طریق ovs-ofctl می‌توانیم جریان اطلاعات را مشخص کنیم.

```

pyretic_0.2.0_32bit [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

mininet@mininet: ~
mininet@mininet:~$ sudo ovs-ofctl add-flow s1 in_port=1,actions=output:2
mininet@mininet:~$ sudo ovs-ofctl add-flow s1 in_port=2,actions=output:1
mininet@mininet:~$ sudo ovs-ofctl dump-flows s1
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=6.097s, table=0, n_packets=0, n_bytes=0, in_port=1 actions=output:2
 cookie=0x0, duration=2.521s, table=0, n_packets=0, n_bytes=0, in_port=2 actions=output:1
mininet@mininet:~$

```

دوباره ping می‌کنیم؛ می‌بینیم که پاسخ دریافت می‌شود.

```

pyretic_0.2.0_32bit [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

mininet@mininet: ~
h1 h2 h3
*** Starting controller
*** Starting 1 switches
s1
*** Starting CLI:
mininet> h1 ping -c3 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data:
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable

--- 10.0.0.2 ping statistics ---
3 packets transmitted, 0 received, +3 errors, 100% packet loss, time 2016ms
pipe 3
mininet> h1 ping -c3 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data:
64 bytes from 10.0.0.2: icmp_req=1 ttl=64 time=0.226 ms
64 bytes from 10.0.0.2: icmp_req=2 ttl=64 time=0.058 ms
64 bytes from 10.0.0.2: icmp_req=3 ttl=64 time=0.057 ms

--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/ndev = 0.057/0.113/0.226/0.080 ms
mininet>

```

در ادامه، نرم‌افزار Wireshark را با دسترسی root باز می‌کنیم؛ برای فیلتر کردن ترافیک OpenFlow، فیلتر of را اعمال می‌کنیم.

Filter: of

No.	Time	Source	Destination	Protocol	Length	Info
41	64.998822	127.0.0.1	127.0.0.1	ICMP	74	Echo Reply (SM) (60)
43	69.998456	127.0.0.1	127.0.0.1	ICMP	74	Echo Request (SM) (60)
44	69.998598	127.0.0.1	127.0.0.1	ICMP	74	Echo Reply (SM) (60)
46	74.998379	127.0.0.1	127.0.0.1	ICMP	74	Echo Request (SM) (60)
47	74.998515	127.0.0.1	127.0.0.1	ICMP	74	Echo Reply (SM) (60)
49	79.999349	127.0.0.1	127.0.0.1	ICMP	74	Echo Request (SM) (60)
50	79.999495	127.0.0.1	127.0.0.1	ICMP	74	Echo Reply (SM) (60)
52	84.998222	127.0.0.1	127.0.0.1	ICMP	74	Echo Request (SM) (60)
53	84.998367	127.0.0.1	127.0.0.1	ICMP	74	Echo Reply (SM) (60)
55	89.998159	127.0.0.1	127.0.0.1	ICMP	74	Echo Request (SM) (60)
56	89.998294	127.0.0.1	127.0.0.1	ICMP	74	Echo Reply (SM) (60)
58	94.998287	127.0.0.1	127.0.0.1	ICMP	74	Echo Request (SM) (60)
59	94.998428	127.0.0.1	127.0.0.1	ICMP	74	Echo Reply (SM) (60)

Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface
 Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
 Internet Protocol Version 4, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
 Transmission Control Protocol, Src Port: 39150 (39150), Dst Port: 6633 (6633), Seq: 1, Ack: 1, Len: 8
 OpenFlow Protocol

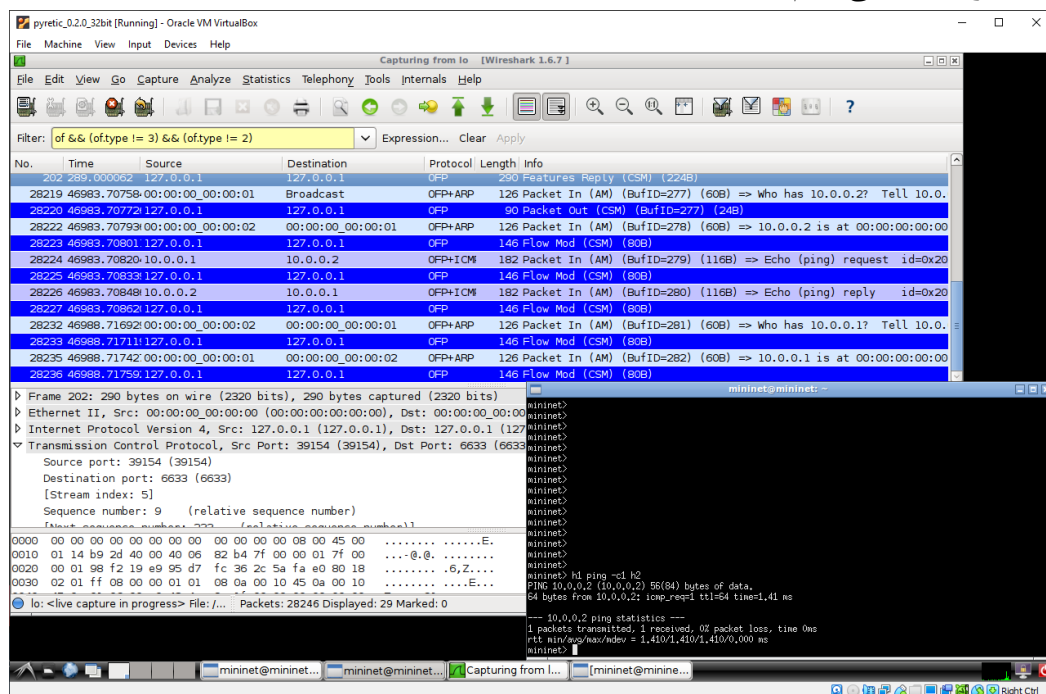
0000 00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00E..
 0010 00 3c 0a 4b 40 00 40 05 2e 6f 7f 00 00 01 7f 00 .<.K@.@..0.....
 0020 00 01 98 ee 19 e9 1b a9 9c 56 d0 cd 9e 06 80 18V.....
 0030 02 01 fe 30 00 00 01 01 08 0a 00 0f 2a d0 00 0f ...0....*....

lo: <live capture in progress> File: /... Packets: 60 Displayed: 40 Marked: 0

با دستور sudo controller tcp:6633 یک کنترلر ایجاد می کنیم. پیغام های رد و بدل شده عبارت اند از:

پیغام	نوع	توضیحات
Hello	از کنترلر به سوئیچ	بعد از TCP handshake، کنترلر شماره version خود را به سوئیچ ارسال می کند.
Hello	از سوئیچ به کنترلر	سوئیچ، شماره versionی که پشتیبانی می کند را به کنترلر ارسال می کند.
Features Request	از کنترلر به سوئیچ	کنترلر، پورتهای آزاد را درخواست می کند.
Set Config	از کنترلر به سوئیچ	کنترلر، flow expiration را درخواست می کند.
Features Reply	از سوئیچ به کنترلر	سوئیچ، لیست پورتهای همراه سرعت آنها، جداول و اعمالی که پشتیبانی می کند را برای کنترلر ارسال می کند.
Port Status	از سوئیچ به کنترلر	در صورت تغییر سرعت پورتهای یا اتصالات آنها، به کنترلر اطلاع می دهد. (به نظر می رسد که bug است.)

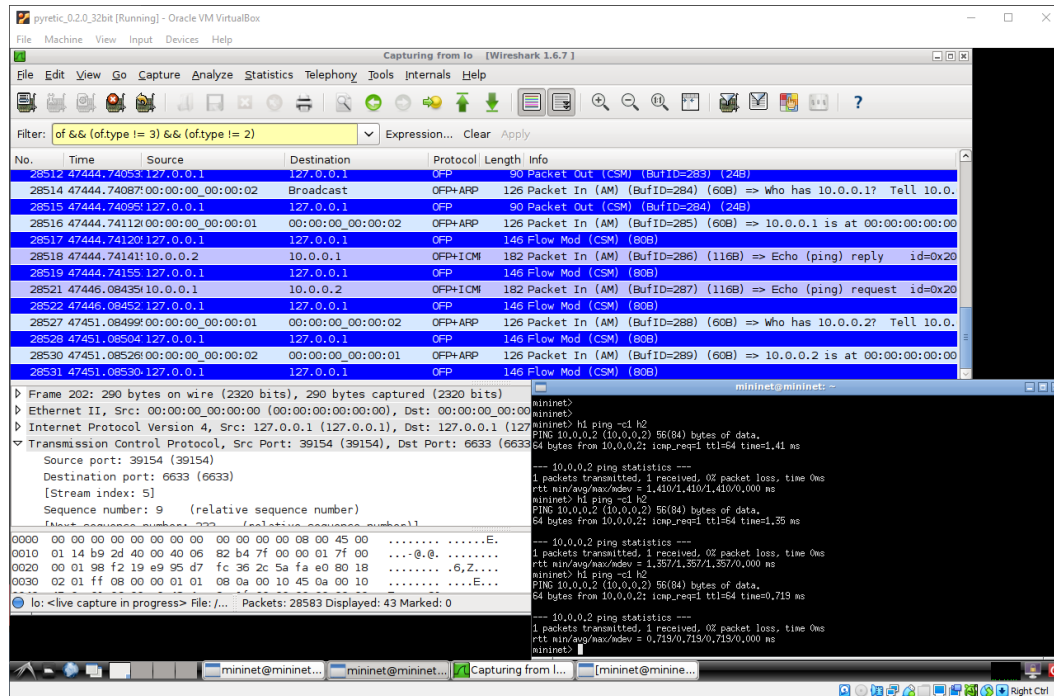
می‌توانیم پیغام‌های echo را با دستور `(of.type != 2) && (of.type != 3)` حذف کنیم. سپس از h1، h2 را ping می‌کنیم.



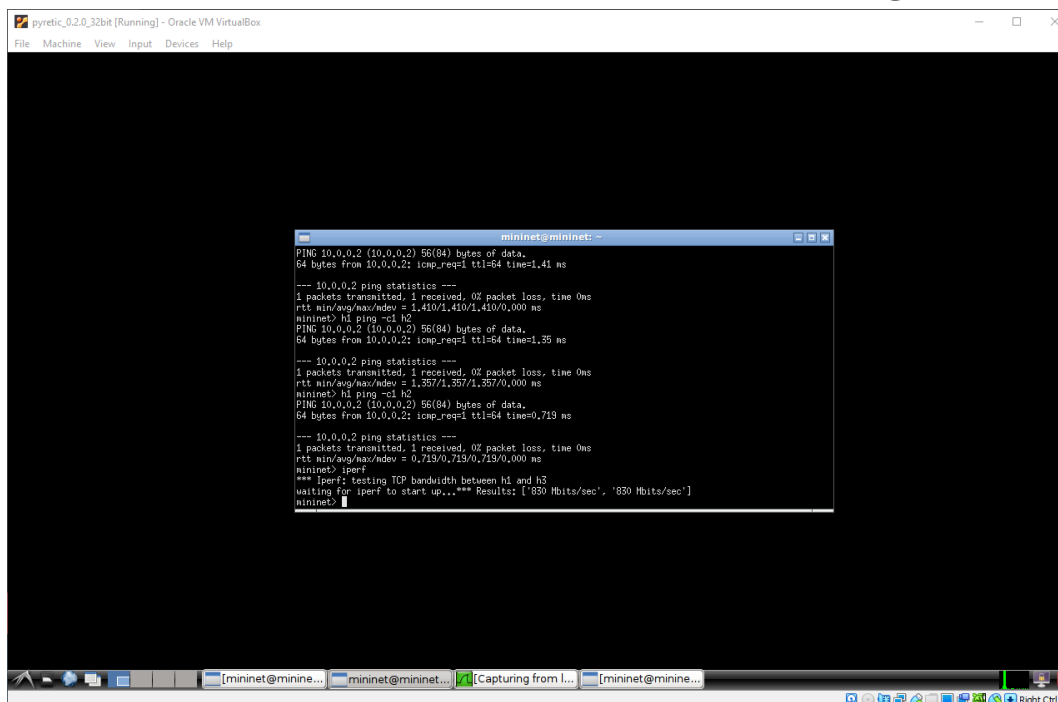
پیغام‌ها عبارت‌اند از:

پیغام	نوع	توضیحات
Packet-In	از سوئیچ به کنترلر	بسته دریافت شده با هیچ‌یک از ورودی‌های flow table تطابق ندارد و به کنترلر فرستاده می‌شود.
Packet-Out	از کنترلر به سوئیچ	کنترلر بسته‌ای را به یک یا چند پورت ارسال می‌کند.
Flow-Mod	از کنترلر به سوئیچ	اضافه کردن یک جریان خاص به flow table
Flow-Expired	از سوئیچ به کنترلر	Time out بعد از مدت زمانی خاص بدون استفاده ماندن یک جریان

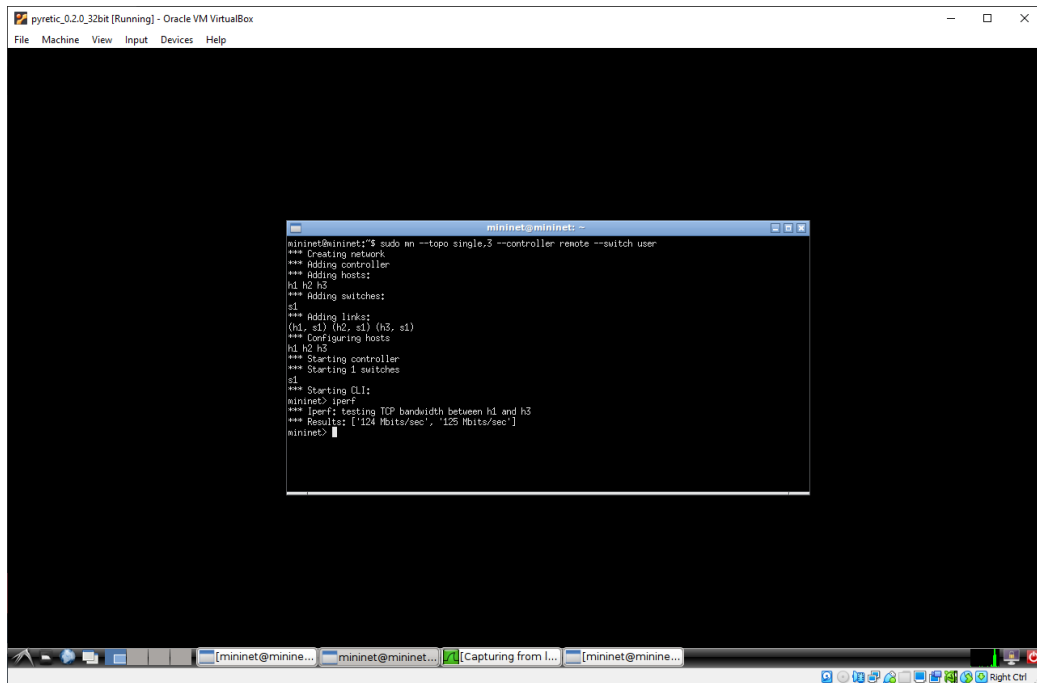
اگر بلافاصله دوباره ping کنیم، می بینیم که مدت زمان کمتری طول می کشد؛ چرا که flowها آماده هستند و منقضی نشده اند.



با استفاده از iperf می توان سرعت بین دو دستگاه را اندازه گیری کرد.



در حالت قبلی هر دو دستگاه روی ماشین مجازی اجرا می‌شوند و بسته‌ها را برای یکدیگر ارسال می‌کنند. با استفاده از دستور `sudo mn --topo single,3 --controller remote --switch user` توپولوژی دیگر ایجاد می‌کنیم؛ در این حالت، بسته‌ها باید از `user-space` به `kernel-space` بروند و برعکس که منجر به کند شدن ارسال اطلاعات می‌شود.

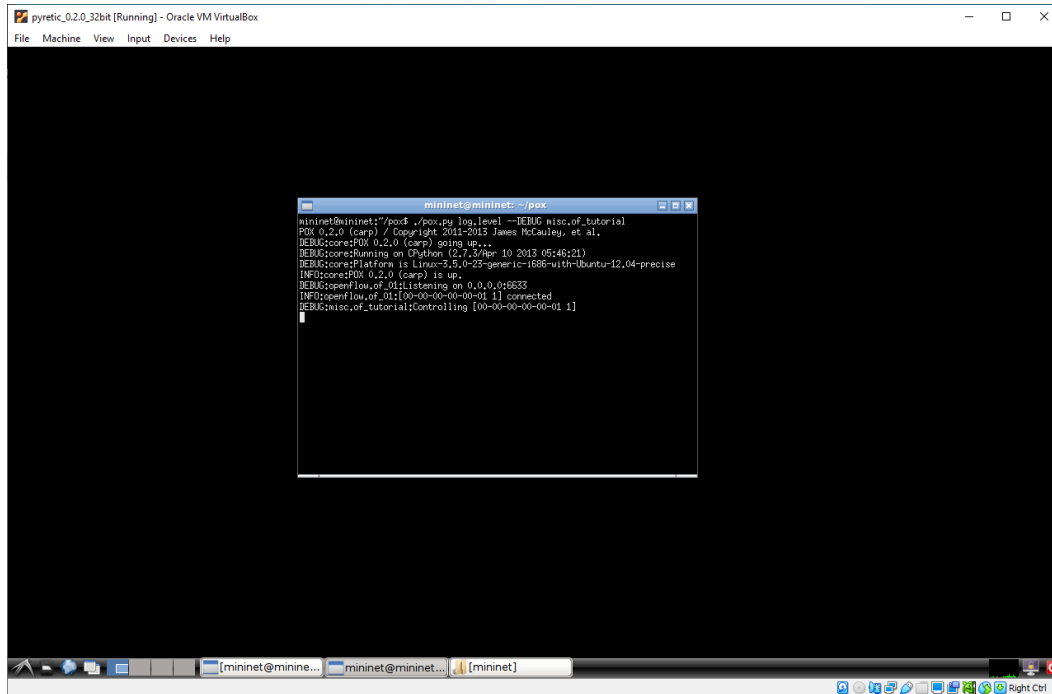


```

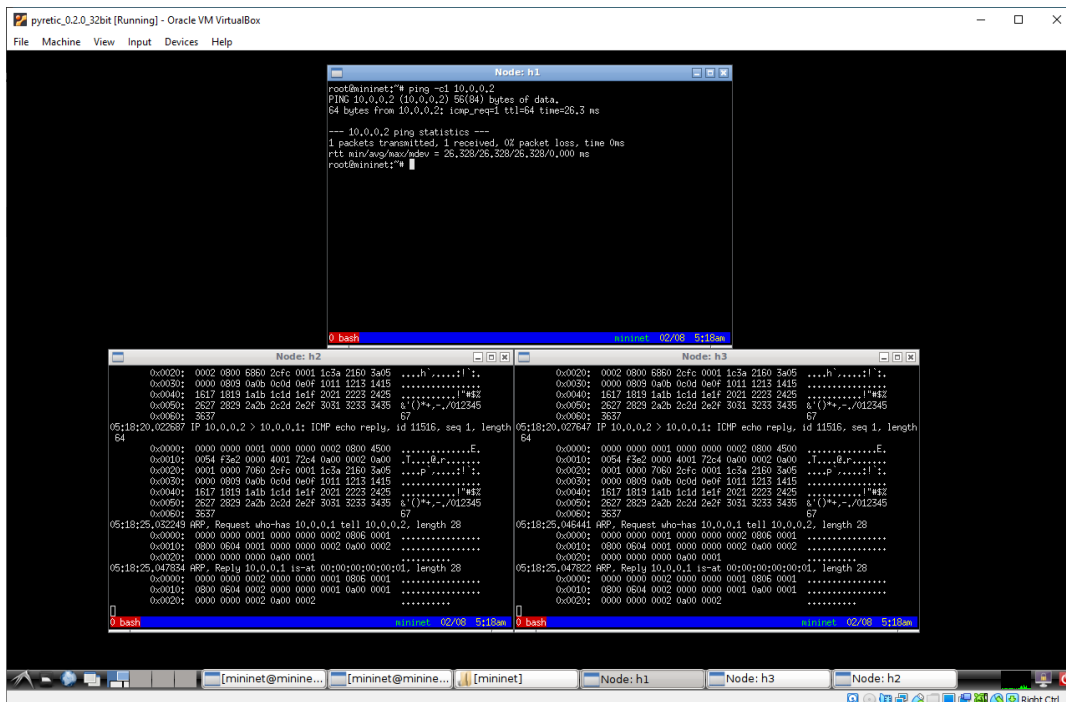
mininet@mininet:~$ sudo mn --topo single,3 --controller remote --switch user
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
*** Starting 1 switches
s1
*** Starting CLI:
mininet> iperf
*** iperf: testing TCP bandwidth between h1 and h3
*** Result: [ 124 Mbits/sec, 125 Mbits/sec ]
mininet>

```

۵. در این قسمت می‌خواهیم یک برنامه شبکه تولید کنیم. ابتدا با دستور `poxy.py log.level --/.` `misc.of_tutorial` یک `hub` اجرا می‌کنیم:



با استفاده از tcpdump از عملکرد hub مطمئن می‌شویم؛ با استفاده از دستور `tcpdump -XX -n -i h1 -s 0` و `h1` را چاپ می‌کنیم و از طریق `h1` ping می‌کنیم:



The screenshot shows a Pyretic VM running mininet. The terminal window displays the following commands and output:

```

mininet@mininet: ~
$ ./mininet.py
Unable to contact the remote controller at 127.0.0.1:6633
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
*** Starting 1 switches
s1
*** Starting CLI:
mininet> xterm h1 h2 h3
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3
h2 -> h1 h3
h3 -> h1 h2
*** Results: 0% dropped (0/6 lost)
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h3
*** Results: ['13.5 Mbits/sec', '13.6 Mbits/sec']
mininet>

```

The taskbar at the bottom shows several open windows, including 'mininet@mininet...', '[mininet@minine...', '[mininet]', '[Node: h1]', '[Node: h3]', and '[Node: h2]'. The system tray on the right includes icons for network, volume, and other system utilities, along with the text 'Right Ctrl'.

The image displays a Kali Linux virtual machine environment with a network simulation. The main terminal window shows the setup of a Mininet network topology. The user has created a network with three hosts (h1, h2, h3) and a switch (s1). The IP addresses assigned to the hosts are 10.0.0.1 for h1, 10.0.0.2 for h2, and 10.0.0.3 for h3. The terminal output shows the results of a ping test from h1 to h2, which was successful with 100% success rate.

Below the main terminal, there are two smaller terminal windows showing the output of a tcpdump command on node h2. The top window shows the capture of a ping request packet (ICMP Echo Request) from h1 to h2. The bottom window shows the capture of a ping response packet (ICMP Echo Reply) from h2 to h1.

The status bar at the bottom of the image shows the active window is 'mininet@mininet...' and the current directory is '/home/mininet/pox'.

برای آزمون کنترلر باید ببینیم آیا بسته‌هایی که آدرس مقصد آن‌ها شناخته‌شده نیستند روی دو hub دیگر نمایش داده می‌شوند یا خیر:

```

mininet@mininet:~/pox
mininet@mininet:~/pox$ ./pox.py log.level --DEBUG misc.of_tutorial
POX 0.2.0 (carp) / Copyright 2011-2013 James H. Cole, et al.
DEBUG:core:POX 0.2.0 (carp) going up...
DEBUG:core:Running on Python (2.7.3 Apr 10 2013 05:46:21)
DEBUG:core:Platform is Linux-3.10.0-23-generic-i686-with-Ubuntu-12.04-precise
INFO:core:POX 0.2.0 (carp) is up.
DEBUG:openflow.of_01:listening on 0.0.0.0:6633
INFO:openflow.of_01:[00:00:00:00:00:00:01] connected
DEBUG:core:of_tutorial:Controlling [00:00:00:00:00:01]
Learning that 00:00:00:00:00:01 is attached at port 1
FF:FF:FF:FF:FF:FF not known, resend to everybody
Learning that 00:00:00:00:00:02 is attached at port 2
00:00:00:00:00:01 destination known, only send message to it
00:00:00:00:00:02 destination known, only send message to it
00:00:00:00:00:01 destination known, only send message to it
00:00:00:00:00:02 destination known, only send message to it
FF:FF:FF:FF:FF:FF not known, resend to everybody
FF:FF:FF:FF:FF:FF not known, resend to everybody
FF:FF:FF:FF:FF:FF not known, resend to everybody
FF:FF:FF:FF:FF:FF not known, resend to everybody
[]

Node: h1
root@mininet:~# ping -c 1 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data:
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=23.9 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/ndev = 23.941/23.941/23.941/0.000 ms
root@mininet:~# ping -c 1 10.0.0.5
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data:
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable

--- 10.0.0.5 ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms
root@mininet:~#

Node: h2
0x0000: 2627 2629 2a2b 2c2d 2e2f 3031 3233 3435 &()*...-/012345
0x0060: 3637
10:46:35.551141 ARP, Request who-has 10.0.0.1 tell 10.0.0.2, length 28
0x0000: 0000 0000 0001 0000 0000 0002 0806 0001 .....
0x0010: 0800 0604 0001 0000 0000 0002 0a00 0002 .....
0x0020: 0000 0000 0000 0a00 0001 .....
10:46:35.702450 ARP, Reply 10.0.0.1 is-at 00:00:00:00:00:01, length 28
0x0000: 0000 0000 0002 0000 0000 0001 0806 0001 .....
0x0010: 0800 0604 0002 0000 0000 0001 0a00 0001 .....
0x0020: 0000 0000 0002 0a00 0002 .....
10:47:32.424134 ARP, Request who-has 10.0.0.5 tell 10.0.0.1, length 28
0x0000: ffff ffff ffff 0000 0000 0001 0806 0001 .....
0x0010: 0800 0604 0001 0000 0000 0001 0a00 0001 .....
0x0020: 0000 0000 0000 0a00 0005 .....
10:47:33.407443 ARP, Request who-has 10.0.0.5 tell 10.0.0.1, length 28
0x0000: ffff ffff ffff 0000 0000 0001 0806 0001 .....
0x0010: 0800 0604 0001 0000 0000 0001 0a00 0001 .....
0x0020: 0000 0000 0000 0a00 0005 .....
10:47:34.336589 ARP, Request who-has 10.0.0.5 tell 10.0.0.1, length 28
0x0000: ffff ffff ffff 0000 0000 0001 0806 0001 .....
0x0010: 0800 0604 0001 0000 0000 0001 0a00 0001 .....
0x0020: 0000 0000 0000 0a00 0005 .....

Node: h3
root@mininet:~# tcpdump -XX -n -i h3-eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on h3-eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
10:46:35.1644037 ARP, Request who-has 10.0.0.2 tell 10.0.0.1, length 28
0x0000: ffff ffff ffff 0000 0000 0001 0806 0001 .....
0x0010: 0800 0604 0001 0000 0000 0001 0a00 0001 .....
0x0020: 0000 0000 0000 0a00 0002 .....
10:47:32.424135 ARP, Request who-has 10.0.0.5 tell 10.0.0.1, length 28
0x0000: ffff ffff ffff 0000 0000 0001 0806 0001 .....
0x0010: 0800 0604 0001 0000 0000 0001 0a00 0001 .....
0x0020: 0000 0000 0000 0a00 0005 .....
10:47:33.407445 ARP, Request who-has 10.0.0.5 tell 10.0.0.1, length 28
0x0000: ffff ffff ffff 0000 0000 0001 0806 0001 .....
0x0010: 0800 0604 0001 0000 0000 0001 0a00 0001 .....
0x0020: 0000 0000 0000 0a00 0005 .....
10:47:34.336590 ARP, Request who-has 10.0.0.5 tell 10.0.0.1, length 28
0x0000: ffff ffff ffff 0000 0000 0001 0806 0001 .....
0x0010: 0800 0604 0001 0000 0000 0001 0a00 0001 .....
0x0020: 0000 0000 0000 0a00 0005 .....

```

هم‌چنین، لازم است سرعت کنترلر را اندازه‌گیری کنیم. باید برابر حالت کنترلر قبلی باشد:

```

mininet@mininet:~
mininet@mininet:~$ ping 10.0.0.2 (10.0.0.2) 56(84) bytes of data:
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=26.5 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/ndev = 26.565/26.565/26.565/0.000 ms
mininet> h1 ping -c 1 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data:
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=37.6 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/ndev = 37.561/37.561/37.561/0.000 ms
mininet> h1 ping -c 1 h3
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data:
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=11.5 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/ndev = 11.517/11.517/11.517/0.000 ms
mininet> iperf
*** iperf: testing TCP bandwidth between h1 and h3
*** Results: [13.0 Mbits/sec], 13.3 Mbits/sec]
mininet>

Node: h1
root@mininet:~# ping -c 1 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data:
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=27.2 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/ndev = 27.287/27.287/27.287/0.000 ms
root@mininet:~# ping -c 1 10.0.0.5
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data:
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable

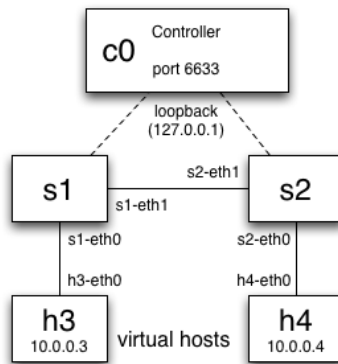
--- 10.0.0.5 ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms
root@mininet:~#

Node: h2
10:02:39.562802 IP 10.0.0.1 > 10.0.0.2: ICMP echo request, id 14909, seq 1, length 64
0x0000: 0000 0000 0002 0000 0000 0001 0800 4500 .....E.
0x0010: 0004 0000 4000 4001 26a7 0a00 0001 0a00 .....L..R..
0x0020: 0002 0800 2472 3a3d 0001 b77c 2160 c56f ...&:..!..o
0x0030: 0800 0808 0a06 0c0d 0e0f 1011 1213 1415 .....
0x0040: 1517 1819 1a1b 1c1d 1e1f 2021 2223 2425 .....*#
0x0050: 2627 2829 2a2b 2c2d 2e2f 3031 3233 3435 &()*...-/012345
0x0060: 3637
10:02:39.562820 IP 10.0.0.2 > 10.0.0.1: ICMP echo reply, id 14909, seq 1, length 64
0x0000: 0000 0000 0001 0000 0000 0002 0800 4500 .....E.
0x0010: 0004 33e0 0000 4001 72ba 0a00 0002 0a00 .....L..R..
0x0020: 0001 0000 2272 3a3d 0001 b77c 2160 c56f .....&:..!..o
0x0030: 0800 0809 0a06 0c0d 0e0f 1011 1213 1415 .....
0x0040: 1517 1819 1a1b 1c1d 1e1f 2021 2223 2425 .....*#
0x0050: 2627 2829 2a2b 2c2d 2e2f 3031 3233 3435 &()*...-/012345
0x0060: 3637
tcpdump: pcap_loop: The interface went down
36 packets captured
0 packets dropped by kernel
root@mininet:~#

Node: h3
0x0040: 3031 3233 3435 3637 3839 3031 3233 3435 0123456789012345
0x0050: 3637 3839 3031 3233 3435 3637 3839 3031 6789012345678901
0x0060: 3233 3435 3637 3839 3031 3233 3435 3637 2345678901234567
0x0070: 3839 3031 3233 3435 3637 3839 3031 3233 8901234567890123
0x0080: 3435 3637 3839 3031 3233 3435 3637 3839 4567890123456789
0x0090: 3031 3233 3435 3637 3839 3031 3233 3435 0123456789012345
0x00a0: 3637 3839 3031 3233 3435 3637 3839 3031 6789012345678901
0x00b0: 3233 3435 3637 3839 3031 3233 3435 3637 2345678901234567
0x00c0: 3839 3031 3233 3435 3637 3839 3031 3233 8901234567890123
0x00d0: 3435 3637 3839 3031 3233 3435 3637 3839 4567890123456789
0x00e0: 3031 3233 3435 3637 3839 3031 3233 3435 0123456789012345
0x00f0: 3637 3839 3031 3233 3435 3637 3839 3031 6789012345678901
0x0100: 0034 0000 4000 4006 26c1 0a00 0001 0a00 ...4..R..B...
0x0110: 0003 c4ab 1389 77ff 77d9 e21d 1f65 8010 ...K.....N...
0x0120: 00a5 d6d2 0000 0101 000a 014d 15f5 014d ...B.....N...
0x0130: 19ef
tcpdump: pcap_loop: The interface went down
254 packets captured
2162 packets received by filter
8828 packets dropped by kernel
root@mininet:~#

```

می‌خواهیم نشان دهیم که کنترلر، از چندین سوئیچ پشتیبانی می‌کند. یک توپولوژی مانند توپولوژی زیر طراحی کنیم:



با استفاده از دستور `sudo mn --topo linear --switch ovsk --controller remote` ، توپولوژی

را ایجاد می‌کنیم.

```

pyretic_0.2.0_32bit [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

mininet@mininet: ~
mininet@mininet:~$ sudo mn --topo linear --switch ovsk --controller remote
*** Creating network
*** Adding controller
*** Adding hosts:
h1, h2
*** Adding switches:
s1, s2
*** Adding links:
(h1, s1) (h2, s2) (s1, s2)
*** Configuring hosts
h1, h2
*** Starting controller
*** Starting 2 switches
s1, s2
*** Starting CLI:
mininet>
  
```

پس از تغییر دادن کنترلر، با استفاده از دستور pingall، از درستی کارکرد توپولوژی مطمئن می‌شویم:

The screenshot shows a VirtualBox window titled 'pyretic_0.2.0_32bit [Running] - Oracle VM VirtualBox'. It contains two terminal windows. The left terminal, titled 'mininet@mininet: ~/pox', displays a series of log messages from the POX controller, including 'not known, resend to everybody' and 'destination known, only send message to it'. The right terminal, titled 'mininet@mininet: ~', shows the execution of the command 'sudo mn --topo linear --switch ovsk --controller remote', followed by network creation steps, host configuration, and a successful 'pingall' test result: 'Results: 0% dropped (0/2 lost)'.

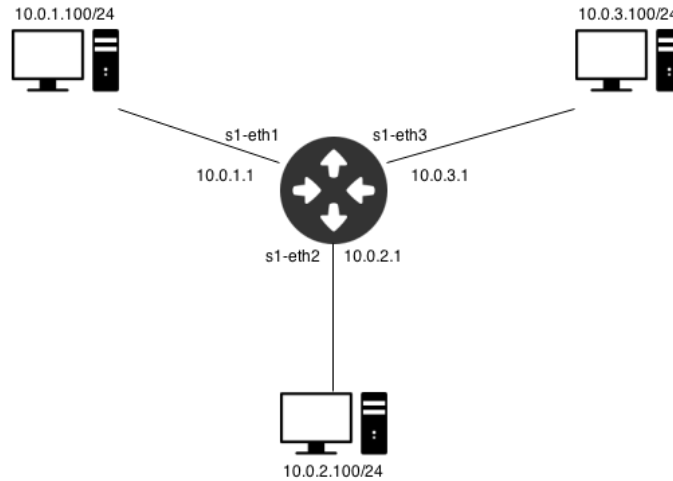
```

mininet@mininet: ~/pox
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
ff:ff:ff:ff:ff:ff not known, resend to everybody
ff:ff:ff:ff:ff:ff not known, resend to everybody
82:a6:bb:e2:eb:4a destination known, only send message to it
82:a6:bb:e2:eb:4a destination known, only send message to it
52:ec:75:9f:ce:30 destination known, only send message to it
52:ec:75:9f:ce:30 destination known, only send message to it
82:a6:bb:e2:eb:4a destination known, only send message to it
82:a6:bb:e2:eb:4a destination known, only send message to it
82:a6:bb:e2:eb:4a destination known, only send message to it
82:a6:bb:e2:eb:4a destination known, only send message to it
52:ec:75:9f:ce:30 destination known, only send message to it
52:ec:75:9f:ce:30 destination known, only send message to it
82:a6:bb:e2:eb:4a destination known, only send message to it
82:a6:bb:e2:eb:4a destination known, only send message to it
52:ec:75:9f:ce:30 destination known, only send message to it
52:ec:75:9f:ce:30 destination known, only send message to it

mininet@mininet: ~
mininet@mininet:~$ sudo mn --topo linear --switch ovsk --controller remote
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1 s2
*** Adding links:
(h1, s1) (h2, s2) (s1, s2)
*** Configuring hosts
h1 h2
*** Starting controller
*** Starting 2 switches
s1 s2
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (0/2 lost)
mininet>
  
```

به تمام درخواست‌ها پاسخ داده می‌شود.

۷. در این مرحله، باید یک توپولوژی مانند توپولوژی زیر ایجاد کنیم.



فایلی به عنوان مثال در `mininet/custom/topo-2sw-2host.py/` آمده است. این فایل را باید طوری تغییر دهیم تا مطابق شکل شود.

سپس، با دستور `sudo mn --custom mytopo.py --topo mytopo --mac`، آن را اجرا می‌کنیم. سپس با دستور `pingall` از درستی آن مطمئن می‌شویم.

```

mininet@mininet:~$ sudo mn --custom /home/mininet/mininet/custom/mytopo.py --topo mytopo --mac
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding link:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
*** Starting 1 switches
s1
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3
h2 -> h1 h3
h3 -> h1 h2
*** Results: 0% dropped (0/6 lost)
mininet>
  
```