سارا سلطانی _ ۹۸۲۳۰۳۳

سوال ۱:

```
acer@ubuntu:~/Desktop/net_project/newenv/lib/python2.7/site-packages$ sudo mn --controller=remote,ip=192.168.159.128,port=6633 --topo=single,3
*** 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
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

ابتدا توپولوژی single با سه هاست ایجاد شد که همگی به یک سوییچ متصل هستند.

دستور nodes:

```
mininet> nodes
available nodes are:
c0 h1 h2 h3 s1
mininet>
```

این دستور تمامی نودهای شبکه اعم از هاست ها و سوییچ و کنترلر را نشان میدهد. که در این سناریو سه هاست و یک کنترلر و یک سوییچ داریم.

دستور net

```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
h3 h3-eth0:s1-eth3
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0 s1-eth3:h3-eth0
c0
mininet>
```

این دستور اطلاعات مربوط به ارتباطات لینک ها را در اختیار ما قرار میدهد.

هاست h۱ از طریق رابط h۱_eth۰ به سوییچ s۱ از طریق s۱_eth۱ متصل میشود.

هاست h۲ از طریق رابط h۲_eth۰ به سوییچ s۱ از طریق s۱_eth۲ متصل میشود.

هاست h۳ از طریق رابط h۳_eth۰ به سوییچ s۱ از طریق s۱_eth۳ متصل میشود.

سوییچ s۱ دارای یک رابط loopback lo است که از طریق رابط s۱_eth۱ به h۱_eth۰ و همچنین از طریق s۱_eth۲ به h۲_eth۰ و از طریق s۱_eth۳ به h۳_eth۰ وصل میشود.

کنترلر c۰ نیز حامل اطلاعات مربوط به کل شبکه است و وظیفه بدست آوردن جداول روتینگ برای سوییچ ها و هدایت بسته ها را دارا ست.

دستور dump

```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=23181>
<Host h2: h2-eth0:10.0.0.2 pid=23183>
<Host h3: h3-eth0:10.0.0.3 pid=23185>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None pid=23190>
<RemoteController{'ip': '192.168.159.128', 'port': 6633} c0: 192.168.159.128:6633 pid=23175>
mininet>
```

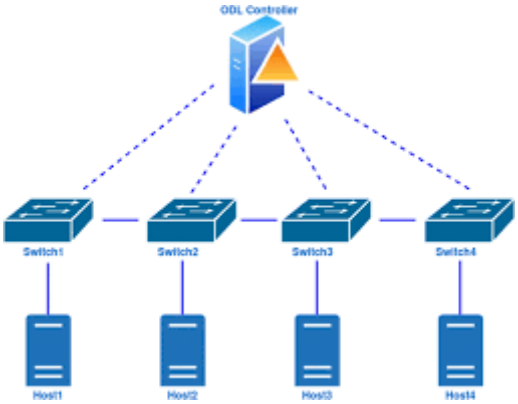
این دستور جزئیات بیشتری از نودهای شبکه به ما میدهد. برای هر سه هاستی که داریم اطلاعاتی از قبیل نام هاست ، اینترفیس آن و ادرس ایپی اختصاص یافته به آن و همچنین آیدی پروسس آن نشان داده میشود.

برای سوییچی که داریم نیز اطلاعاتی مانند نام آن و اینترفیس هایی که آن را به دیگر هاست ها مرتبط میکند و همچنین آدرس آیپی ان و ایدی پروسس آن به همراه وجود loopback lo نشان داده میشود.

برای کنترلر موجود نیز آیدی پروسس آن و همچنین شماره پورت اجرایی آن (که به دلیل استفاده از کنترلر pox پورت اجرایی به صورت دیفالت ۶۶۳۳ است). علاوه بر آن آیپی ادرسی که روی آن اجرا میشود(با توجه به اینکه روی سیستم لوکال اجرا میشود همان آیپی آدرس لوکال سیستم اجرای است.)

سوال دو :

Linear topology

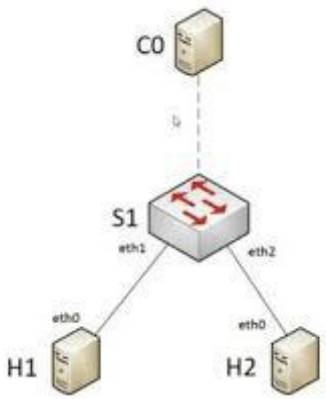


این توپولوژی شامل تعداد مساوی سوییچ و هاست است که هر هاست به یک سوییچ اختصاص یافته و سوییچ ها نیز به صورت خطی و پشت سر هم به یکدیگر متصل اند. در این شکل ۴ هاست و ۴ سوییچ وجود دارد.

```
acer@ubuntu:~/Desktop/net_project/newenv/lib/python2.7/site-packages$ sudo mn --controller=remote,ip=192.168.159.128,port=6633 --topo=linear,4
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (h4, s4) (s2, s1) (s3, s2) (s4, s3)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet>
```

همانطوز که مشخص است ۴ هاست و ۴ سوییچ داریم که هر هاست به سوییچ متناظر متصل است و علاوه بر آن هر سوییچ به سوییچ مجاورش متصل است.

Minimal topology

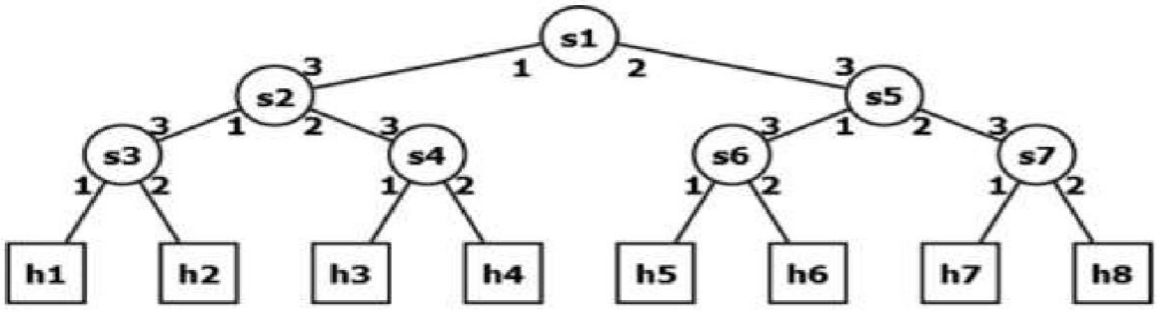


این توپولوژی شامل یک سوییچ هسته open flow است که متصل به دو هاست میباشد. همچنین یک کنترلر اصلی نیز سوییچ را کنترل میکند و به آن دستورات لازم را میدهد.

```
acer@ubuntu:~/Desktop/net_project/newenv/lib/python2.7/site-packages$ sudo mn --controller=remote,ip=192.168.159.128,port=6633 --topo=minimal
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

همانطور که اینجا نیز نشان داده شده دو هاست و یک سوییچ داریم که هردو به این سوییچ متصل هستند و یک کنترلر مرکزی داریم.

Tree topology



این توپولوژی یک درخت دودویی با عمق مشخص ایجاد میکند که در آن هر سوییچ برگ، دارای دو هاست متصل به آن است. در این شکل عمق درخت ۳ است یعنی سه لول سوییچ داریم و در نتیجه ۸ هاست داریم.

```
acer@ubuntu:~/Desktop/net_project/newenv/lib/python2.7/site-packages$ sudo mn --controller=remote,ip=192.168.159.128,port=6633 --topo=tree,depth=3,fanout=2
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7
*** Adding links:
(s1, s2) (s1, s5) (s2, s3) (s2, s4) (s3, h1) (s3, h2) (s4, h3) (s4, h4) (s5, s6) (s5, s7) (s6, h5) (s6, h6) (s7, h7) (s7, h8)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8
*** Starting controller
c0
*** Starting 7 switches
s1 s2 s3 s4 s5 s6 s7 ...
*** Starting CLI:
mininet>
```

در اینجا نیز با توجه به اینکه پارامتر depth=۳ است پس دو لول علاوه بر نود ریشه داریم که معادل با ۷ سوییچ میشود و از طرف دیگر با توجه به اینکه fanout=۲ قرار داده شده پس هر سوییچ برگ باید دو فرزند یا درواقع دو هاست متصل داشته باشد.

سوال سه:

میتوان به طور موثر از tcpdump در Mininet برای ضبط و تجزیه و تحلیل ترافیک شبکه استفاده کرد که به عیب یابی و درک رفتار شبکه کمک می کند. مثلا اگر پکت لاس در شبکه اتفاق بیفتد میتوان با ردیابی ترافیک متوجه آن شد و مجدد بسته را ارسال کرد و در واقع اینگونه از ارسال بسته های تکراری خودداری کرد.

برای نشان دادن این ویژگی ابتدا یک توپولوژی linear با ۴ هاست ایجاد کرده. سپس با استفاده از xterm برای هر یک از هاست ها ترمینالی جداگانه ایجاد شده تا ترافیک مربوط به هرکدام قابل دریابی باشد.

سپس هاست های h۱ , h۲, h۳ را به عنوان شنونده یا دریافت کننده در نظر گرفته و هاست h٤ را ارسال کننده در نظر میگیریم و با دستور زیر برای هر سه هاست دریافت و گوش دادن به ارسال کننده را فراهم میکنیم:

```
tcpdump -i h۱-eth۰ -w h۱.pcap
tcpdump -i h۲-eth۰ -w h۲.pcap
tcpdump -i h۳-eth۰ -w h۳.pcap
```

در سمت ارسال کننده (هاست h٤) نیز ابتدا برای هاست های h۳ , h۲ به تعداد ۴ پکت پینگ کرده تا ترافیک برای هر دو ارسال شود.

```
ping -c ٤ ١٠,٠,٠,٢
ping -c ٤ ١٠,٠,٠,٣
```

سپس با دستور زیر نرخ ریت پکت لاس ۱۰٪ را برای این هاست تنظیم کرده و سپس به تعداد ۴ بسته نیز برای هاست h۱ پینگ میکنیم:

```
sudo tc qdisc add dev h٤-eth٠ root netem loss ١٠٪
```

سپس در سمت دریافت کنندگان tcpdump را متوقف کرده و با دستورات زیر ترافیک موجود روی هر هاست را ردیابی مکنیم.

```
tcpdump -r h١.pcap
tcpdump -r h٢.pcap
tcpdump -r h٣.pcap
```

```
root@ubuntu:/home/acer/Desktop/net_project/newenv/lib/python2.7/site-packages# p
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=9.37 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=1.53 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.185 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.330 ms

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3029ms
rtt min/avg/max/mdev = 0.185/2.854/9.371/3.798 ms
root@ubuntu:/home/acer/Desktop/net_project/newenv/lib/python2.7/site-packages# p
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=9.84 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=1.08 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.062 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=0.174 ms

--- 10.0.0.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3022ms
rtt min/avg/max/mdev = 0.062/2.788/9.838/4.089 ms
```

پینگ هاست های ۲ و ۳ بدون پکت لاس_۱

```
root@ubuntu:/home/acer/Desktop/net_project/newenv/lib/python2.7/site-packages# s
root@ubuntu:/home/acer/Desktop/net_project/newenv/lib/python2.7/site-packages# p
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=25.6 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=1.92 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.208 ms

--- 10.0.0.1 ping statistics ---
4 packets transmitted, 3 received, 25% packet loss, time 3013ms
rtt min/avg/max/mdev = 0.208/9.234/25.574/11.575 ms
```

پینگ هاست ۱ با یک پکت لاس_۲

در این دو تصویر همانطور که مشاهده میکنید سمت هاست های h۳ , h۲ هر ۴ پکت به درستی دریافت شده اند. که sequence number بسته ها گواه این ادعاست.

```
root@ubuntu:/home/acer/Desktop/net_project/newenv/lib/python2.7/site-packages# tcpdump -r h3.pcap
reading from file h3.pcap, link-type EN10MB (Ethernet)
02:28:01.863169 ARP, Request who-has 10.0.0.1 tell 10.0.0.4, length 28
02:30:33.474003 ARP, Request who-has 10.0.0.2 tell 10.0.0.4, length 28
02:31:52.170543 ARP, Request who-has 10.0.0.3 tell 10.0.0.4, length 28
02:31:52.170560 ARP, Reply 10.0.0.3 is-at 46:a2:ee:2c:01:33 (oui Unknown), length 28
02:31:52.175834 IP 10.0.0.4 > 10.0.0.3: ICMP echo request, id 5328, seq 1, length 64
02:31:52.175885 IP 10.0.0.3 > 10.0.0.4: ICMP echo reply, id 5328, seq 1, length 64
02:31:53.171039 IP 10.0.0.4 > 10.0.0.3: ICMP echo request, id 5328, seq 2, length 64
02:31:53.171099 IP 10.0.0.3 > 10.0.0.4: ICMP echo reply, id 5328, seq 2, length 64
02:31:54.172106 IP 10.0.0.4 > 10.0.0.3: ICMP echo request, id 5328, seq 3, length 64
02:31:54.172123 IP 10.0.0.3 > 10.0.0.4: ICMP echo reply, id 5328, seq 3, length 64
02:31:55.190449 IP 10.0.0.4 > 10.0.0.3: ICMP echo request, id 5328, seq 4, length 64
02:31:55.190497 IP 10.0.0.3 > 10.0.0.4: ICMP echo reply, id 5328, seq 4, length 64
02:31:57.270274 ARP, Request who-has 10.0.0.4 tell 10.0.0.3, length 28
02:31:57.283857 ARP, Reply 10.0.0.4 is-at fa:01:05:20:08:a0 (oui Unknown), length 28
root@ubuntu:/home/acer/Desktop/net_project/newenv/lib/python2.7/site-packages#
```

ا

```
root@ubuntu:/home/acer/Desktop/net_project/newenv/lib/python2.7/site-packages# tcpdump -r h2.pcap
reading from file h2.pcap, link-type EN10MB (Ethernet)
02:28:01.863935 ARP, Request who-has 10.0.0.1 tell 10.0.0.4, length 28
02:30:33.474293 ARP, Request who-has 10.0.0.2 tell 10.0.0.4, length 28
02:30:33.474306 ARP, Reply 10.0.0.2 is-at 0e:3b:e8:28:a3:6d (oui Unknown), length 28
02:30:33.478910 IP 10.0.0.4 > 10.0.0.2: ICMP echo request, id 5320, seq 1, length 64
02:30:33.478956 IP 10.0.0.2 > 10.0.0.4: ICMP echo reply, id 5320, seq 1, length 64
02:30:34.476321 IP 10.0.0.4 > 10.0.0.2: ICMP echo request, id 5320, seq 2, length 64
02:30:34.476379 IP 10.0.0.2 > 10.0.0.4: ICMP echo reply, id 5320, seq 2, length 64
02:30:35.477023 IP 10.0.0.4 > 10.0.0.2: ICMP echo request, id 5320, seq 3, length 64
02:30:35.477074 IP 10.0.0.2 > 10.0.0.4: ICMP echo reply, id 5320, seq 3, length 64
02:30:36.502085 IP 10.0.0.4 > 10.0.0.2: ICMP echo request, id 5320, seq 4, length 64
02:30:36.502133 IP 10.0.0.2 > 10.0.0.4: ICMP echo reply, id 5320, seq 4, length 64
02:30:38.678148 ARP, Request who-has 10.0.0.4 tell 10.0.0.2, length 28
02:30:38.683936 ARP, Reply 10.0.0.4 is-at fa:01:05:20:08:a0 (oui Unknown), length 28
02:31:52.171634 ARP, Request who-has 10.0.0.3 tell 10.0.0.4, length 28
root@ubuntu:/home/acer/Desktop/net_project/newenv/lib/python2.7/site-packages#
```

اما با توجه به اینکه در پینگ به هاست یک یک پکت لاس رخ داده بود همانطور که درشکل زیر نیز مسخص است با ردیابی ترافیک هاست یک بسته با sequence number ۱ گم شده:

```
root@ubuntu:/home/acer/Desktop/net_project/newenv/lib/python2.7/site-packages# tcpdump -r h1.pcap
reading from file h1.pcap, link-type EN10MB (Ethernet)
03:13:21.959387 IP 10.0.0.4 > 10.0.0.1: ICMP echo request, id 5562, seq 2, length 64
03:13:21.959431 IP 10.0.0.1 > 10.0.0.4: ICMP echo reply, id 5562, seq 2, length 64
03:13:22.944930 IP 10.0.0.4 > 10.0.0.1: ICMP echo request, id 5562, seq 3, length 64
03:13:22.945000 IP 10.0.0.1 > 10.0.0.4: ICMP echo reply, id 5562, seq 3, length 64
03:13:23.945267 IP 10.0.0.4 > 10.0.0.1: ICMP echo request, id 5562, seq 4, length 64
03:13:23.945318 IP 10.0.0.1 > 10.0.0.4: ICMP echo reply, id 5562, seq 4, length 64
03:13:27.125736 ARP, Request who-has 10.0.0.4 tell 10.0.0.1, length 28
03:13:27.145136 ARP, Request who-has 10.0.0.1 tell 10.0.0.4, length 28
03:13:27.145141 ARP, Reply 10.0.0.1 is-at f2:d6:34:1a:8d:49 (oui Unknown), length 28
03:13:27.148020 ARP, Reply 10.0.0.4 is-at fa:01:05:20:08:a0 (oui Unknown), length 28
```