به نام خدا

امیررضا حسینی – ۹۸۲۰۳۶۳

شبکه ۲ – پروژه اول

سؤال اول) بله، همانطور که در خروجی گویاست، طبق آموزش به درستی نصب شده است.

```
amir@amir-VirtualBox:~$ git --version
git version 2.25.1
amir@amir-VirtualBox:~$ mn --version
2.3.1b1
amir@amir-VirtualBox:~$ mn --help
Usage: mn [options]
(type mn -h for details)

The mn utility creates Mininet network from the command line. It can create
parametrized topologies, invoke the Mininet CLI, and run tests.

Options:
  -h, --help            show this help message and exit
  --switch=SWITCH       default|ivs|lxbr|ovs|ovsbr|ovsk|user[,param=value...]
                        user=UserSwitch ovs=OVSSwitch ovsbr=OVSBridge
                        ovsk=OVSSwitch ivs=IVSSwitch lxbr=LinuxBridge
                        default=OVSSwitch
  --host=HOST           cfs|proc|rt[,param=value...] proc=Host
                        rt=CPULimitedHost{'sched': 'rt'}
                        cfs=CPULimitedHost{'sched': 'cfs'}
  --controller=CONTROLLER
                        default|none|nox|ovsc|ref|remote|ryu[,param=value...]
                        ref=Controller ovsc=OVSController nox=NOX
                        remote=RemoteController ryu=Ryu
                        default=DefaultController none=NullController
  --link=LINK           default|ovs|tc|tcu[,param=value...] default=Link
                        tc=TCLink tcu=TCULink ovs=OVSLink
  --topo=TOPO           linear|minimal|reversed|single|torus|tree[,param=value
                        ...] minimal=MinimalTopo linear=LinearTopo
                        reversed=SingleSwitchReversedTopo
                        single=SingleSwitchTopo tree=TreeTopo torus=TorusTopo
  -c, --clean           clean and exit
  --custom=CUSTOM       read custom classes or params from .py file(s)
  --test=TEST           pingall|pingpair|iperf|iperfudp|all|none|build
  -x, --xterms          spawn xterms for each node
  -i IPBASE, --ipbase=IPBASE
                        base IP address for hosts
  --mac                 automatically set host MACs
  --arp                 set all-pairs ARP entries
  -v VERBOSITY, --verbosity=VERBOSITY
                        debug|info|output|warning|warn|error|critical
  --innamespace         sw and ctrl in namespace?
  --listenport=LISTENPORT
                        base port for passive switch listening
  --nolistenport        don't use passive listening port
  --pre=PRE             CLI script to run before tests
  --post=POST           CLI script to run after tests
  --pin                 pin hosts to CPU cores (requires --host cfs or --host
```

در شکل بالا خروجی دو دستور git –version, mn --help , mn --version را مشاهده میکنیم.
نصب آن طبق دستورهای زیر انجام شده است.

```
sudo apt install git
sudo git clone https://github.com/mininet/mininet
cd mininet/util
./install -a
```

سؤال دوم) ایجاد توپولوژی:

```
amir@amir-VirtualBox:~$ sudo mn --topo single,3
[sudo] password for amir:
*** 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>
```

نتیجه دستورnodes:

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

دستورnodes برای نشان دادن تمام نودها در شبکه استفاده میشود؛ بدین صورت که تمامی نودها اعم از هاست‌ها، سوئیچ‌هاو کنترلرها را نشان میدهد. در توپولوژی بالا ما سه هاست داشتیم که از طریق یک سوئیچ به همدیگر متصل بودند، همچنین یک کنترلر برای ساختن Data plane و مشخص کردن روتینگ شبکه داریم که در نتیجه دستور هر۵ نود نمایش داده شده اند.

نتیجه دستورnet:

```
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>
```

دستور net برای نشان دادن روابط میان نودهاست. (هر نود به چه نودی و از طریق چه چیزی متصل است.) برای مثال در شبکه‌ی فوق سه هاست از طریق اینترفیس eth0 به سوئیچ متصل شده اند و سوئیچ نیز توسط eth1, eth2, eth3 به ترتیب به هاست یک تا سه متصل شده؛ در نتیجه سه هاست از طریق این سوئیچ به هم متصل‌اند.

- هاست h1 با سوئیچ S1 از طریق ارتباط ethernet متصل هستند.
- هاست h2 با سوئیچ S1 از طریق ارتباط ethernet متصل هستند.
- هاست h3 با سوئیچ S1 از طریق ارتباط ethernet متصل هستند.
- سوئیچ s1 از طریق کابل ethernet به هاستهای h1, h2 و h3 متصل است.

نتیجه دستورdump:

```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=3081>
<Host h2: h2-eth0:10.0.0.2 pid=3083>
<Host h3: h3-eth0:10.0.0.3 pid=3085>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None pid=3090>
<OVSController c0: 127.0.0.1:6653 pid=3074>
mininet>
```

دستور dump نیز برای نمایش نام نود، network interface به همراه آدرس ip آن به‌علاوه‌ی پروسس آیدی آن در سیستم را نشان میدهد.

- نود اول که h1 نام دارد و یک هاست است، آیپی ۱۰.۰.۰.۱گرفته است و آیدی پروسس مربوطه برابر است با۳۰۸۱. همچنین مشخص است که یک ارتباط اترنت به نام h1-eth0 دارد.
- نود دوم که h2 نام دارد و یک هاست است، آیپی ۱۰.۰.۰.۲گرفته است و آیدی پروسس مربوطه برابر است با۳۰۸۳. همچنین مشخص است که یک ارتباط اترنت به نام h2-eth0 دارد.
- نود سوم که h3 نام دارد و یک هاست است، آیپی ۱۰.۰.۰.۳گرفته است و آیدی پروسس مربوطه برابر است با ۳۰۸۵. همچنین مشخص است که یک ارتباط اترنت به نام h3-eth0 دارد.
- نود چهارم که s1 نام دارد و یک OVSSwitch است، آیپی ۱۲۷.۰.۰.۱دارد، یک ارتباط اترنت دارد و آیپی پروسس مربوطه برابر است با ۳۰۹۰.
- نود پنجم که c0 نام دارد و یک کنترلر است. همچنین مشخص شده است که آیپی آن برابر با ۱۲۷.۰.۰.۱است و روی پورت ۶۶۵۳در حال شنود است. همچنین مشخص شده است که آیدی پروسس مربوط به آن برابر است با ۳۰۷۴.

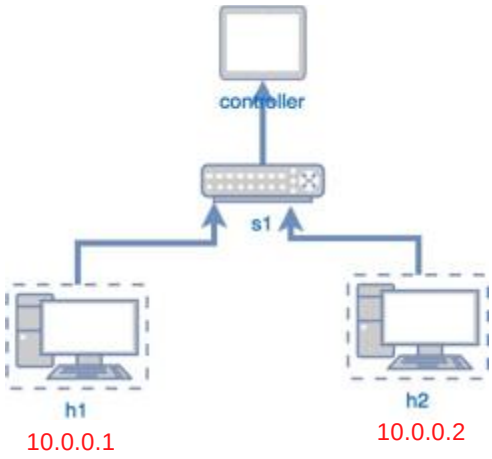
سؤال سوم) با توجه به داکیومنت مینینت طبق عکس زیر توپولوژیها مشخص شده درتصویر را میتوان استفاده کرد.

```
amir@amir-VirtualBox:~$ mn --help
Usage: mn [options]
(type mn -h for details)

The mn utility creates Mininet network from the command line. It can create
parametrized topologies, invoke the Mininet CLI, and run tests.

Options:
  -h, --help            show this help message and exit
  --switch=SWITCH       default|ivs|lxbr|ovs|ovsbr|ovsk|user[,param=value...]
                        user=UserSwitch ovs=OVSSwitch ovsbr=OVSBridge
                        ovsk=OVSSwitch ivs=IVSSwitch lxbr=LinuxBridge
                        default=OVSSwitch
  --host=HOST           cfs|proc|rt[,param=value...] proc=Host
                        rt=CPULimitedHost{'sched': 'rt'}
                        cfs=CPULimitedHost{'sched': 'cfs'}
  --controller=CONTROLLER
                        default|none|nox|ovsc|ref|remote|ryu[,param=value...]
                        ref=Controller ovsc=OVSController nox=NOX
                        remote=RemoteController ryu=Ryu
                        default=DefaultController none=NullController
  --link=LINK           default|ovs|tc|tcu[,param=value...] default=Link
                        tc=TCLink tcu=TCULink ovs=OVSLink
  --topo=TOPO           linear|minimal|reversed|single|torus|tree[,param=value
                        ...] minimal=MinimalTopo linear=LinearTopo
                        reversed=SingleSwitchReversedTopo
                        single=SingleSwitchTopo tree=TreeTopo torus=TorusTopo
  -c, --clean           clean and exit
```

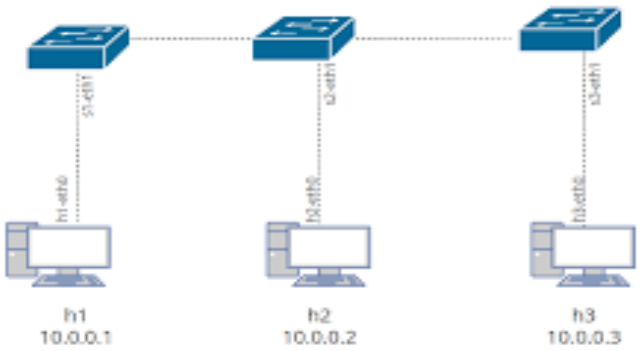
۱) توپولوژی minimal: در این سناریو یک توپولوژی بسیار ساده و مینیمال ساخته میشود. این توپولوژی شامل دو هاست است که ازطریق یک سوئیچ به یکدیگر متصل اند. همچنین این سناریو یک کنترلر نیز دارد. این توپولوژی آرگومان ورودی ندارد و بجز حالت توضیح داده شده نمیتوان توپولوژی دیگری با استفاده از این دستور ساخت.



دستورات برای ساخت این توپولوژی:

```
amir@amir-VirtualBox:~$ sudo mn --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> nodes
available nodes are:
c0 h1 h2 s1
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0
c0
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=6379>
<Host h2: h2-eth0:10.0.0.2 pid=6381>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=6386>
<OVSController c0: 127.0.0.1:6653 pid=6372>
mininet>
```

۲) توپولوژی linear: در این توپولوژی سوئیچ ها بصورت خطی به یکدیگر متصل اند و هرهاست هم به یک سوئیچ متصل است. اگر تعداد هاست ها بیشتر شود همین الگو ادامه پیدا میکند، به ازای هر هاست یک سوئیچ اضافه میشود در حالیکه هاست جدید به سوئیچ جدید متصل است سوئیچ جدید به سوئیچ قبلی متصل میشود و این فرآیند ادامه مییابد دراینجا به عنوان مثال توپولوژی ای رسم شده است که سه هاست دارد.



دستورات برای ساخت این توپولوژی:

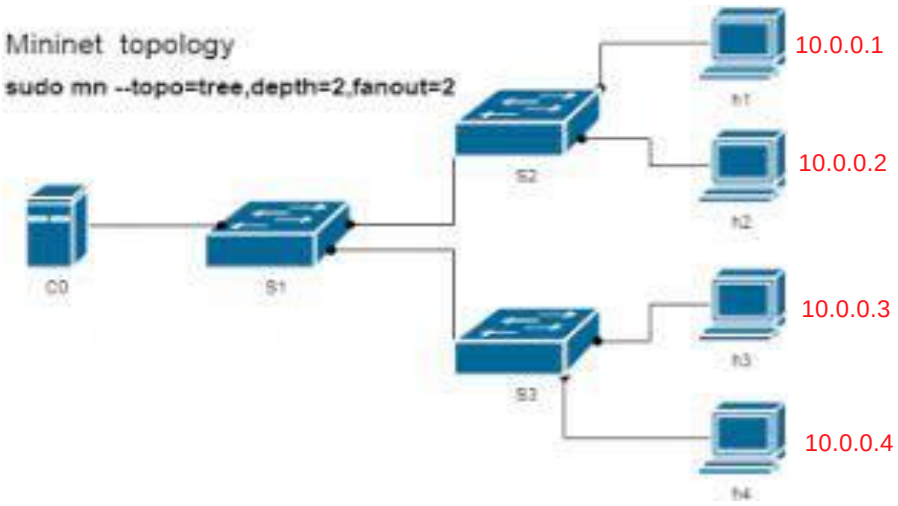
```
amir@amir-VirtualBox:~$ sudo mn --topo linear,3
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1 s2 s3
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (s2, s1) (s3, s2)
*** Configuring hosts
h1 h2 h3
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet> nodes
available nodes are:
c0 h1 h2 h3 s1 s2 s3
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s2-eth1
h3 h3-eth0:s3-eth1
s1 lo:  s1-eth1:h1-eth0 s1-eth2:s2-eth2
s2 lo:  s2-eth1:h2-eth0 s2-eth2:s1-eth2 s2-eth3:s3-eth2
s3 lo:  s3-eth1:h3-eth0 s3-eth2:s2-eth3
c0
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=6055>
<Host h2: h2-eth0:10.0.0.2 pid=6057>
<Host h3: h3-eth0:10.0.0.3 pid=6059>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=6064>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None pid=6067>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None pid=6070>
<OVSController c0: 127.0.0.1:6653 pid=6048>
mininet>
```

۳) توپولوژی tree: در این سناریو بعد از مشخص کردن نوع توپولوژی دو آرگومان دریافت میشود. اولین آرگومان عمق درخت است که مشخص میکند چند لایه سوئیچ داشته باشیم. آرگومان دوم تعداد هاست‌ها است که برابر است با تعداد برگهای درخت. به عنوان مثال در پایین توپولوژی با عمق ۲و fanout = 2 نمایش داده شده است.اگر درخت دودویی باشد خواهیم داشت:

تعداد سوئیچ‌ها در درخت در سطح(L) برابر است با 2^{L-1} . (سطح اول شامل ۱ نود است)

تعداد سوئیچ‌ها در درخت با عمق N برابر است با : $\sum_{i=1}^{N} 2^{i-1} = 2^N - 1$

تعداد هاست‌ها در درخت با عمق N برابر است با : 2^N



دستورات برای ساخت این توپولوژی:

```
amir@amir-VirtualBox:~$ sudo mn --topo tree,fanout=2,depth=2
[sudo] password for amir:
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3
*** Adding links:
(s1, s2) (s1, s3) (s2, h1) (s2, h2) (s3, h3) (s3, h4)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet> nodes
available nodes are:
c0 h1 h2 h3 h4 s1 s2 s3
mininet> net
h1 h1-eth0:s2-eth1
h2 h2-eth0:s2-eth2
h3 h3-eth0:s3-eth1
h4 h4-eth0:s3-eth2
s1 lo:  s1-eth1:s2-eth3 s1-eth2:s3-eth3
s2 lo:  s2-eth1:h1-eth0 s2-eth2:h2-eth0 s2-eth3:s1-eth1
s3 lo:  s3-eth1:h3-eth0 s3-eth2:h4-eth0 s3-eth3:s1-eth2
c0
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=6574>
<Host h2: h2-eth0:10.0.0.2 pid=6576>
<Host h3: h3-eth0:10.0.0.3 pid=6578>
<Host h4: h4-eth0:10.0.0.4 pid=6580>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=6585>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None pid=6588>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None,s3-eth3:None pid=6591>
<OVSController c0: 127.0.0.1:6653 pid=6567>
mininet>
```