

به نام خدا

پروژه 1

مرضیه علیدادی 9631983

1. شبیه ساز mininet را روی سیستم نصب کردم. از command ای که برای تست نصب است، استفاده کردم:

```
marzleh@ubuntu:~$ sudo mn --test pingall
*** 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 ...
*** Waiting for switches to connect
s1 ...
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
*** Stopping 1 controllers
c0
*** Stopping 2 links
...
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 5.733 seconds
marzleh@ubuntu:~$
```

2. از توپولوژی های آماده ی mininet می توان به linear ، minimal ، single و tree اشاره کرد:

1- توپولوژی minimal ، شامل 2 تا host و 1 switch است. که این switch به هر کدام از host ها متصل است.

برای ایجاد این توپولوژی، از command زیر استفاده می شود:

`sudo mn --topo minimal`

در تصویر زیر، این توپولوژی را ایجاد کردم:

```

marzieh@ubuntu:~/Desktop$ 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>
  
```

2- توپولوژی linear، به تعداد n ای که مشخص کردیم، host و به همان تعداد، switch می سازد. هر switch به یک host متصل است، و همه switch ها به هم متصل هستند.
 برای ایجاد این توپولوژی، از command زیر استفاده می شود:
`sudo mn --topo linear,n`

در تصویر زیر، این توپولوژی را ایجاد کردم:

```

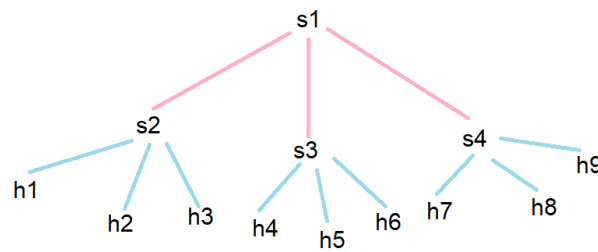
marzieh@ubuntu:~/Desktop$ sudo mn --topo linear,3
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1 s2 s3
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (s2, s1) (s3, s2)
*** Configuring hosts
h1 h2 h3
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet>
  
```

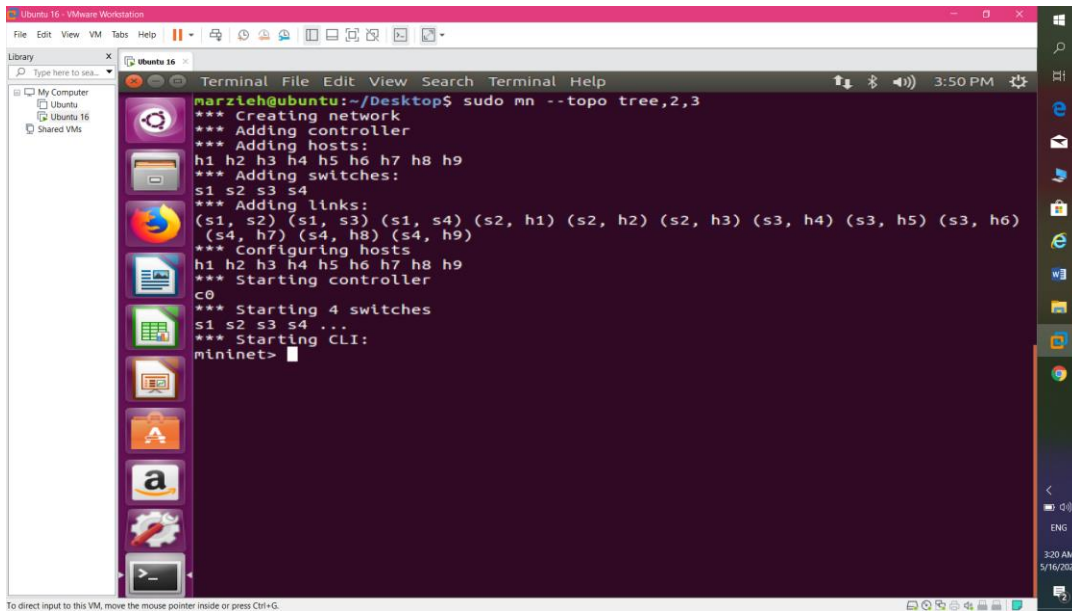
3- توپولوژی single ، شامل تعداد مشخص شده ی n تا host و 1 switch است.
 که این switch به هر کدام از host ها متصل است.
 برای ایجاد این توپولوژی، از command زیر استفاده می شود:
`sudo mn --topo single,n`
 در تصویر زیر، این توپولوژی را ایجاد کردم:

```

marzleh@ubuntu: ~/Desktop$ sudo mn --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>
    
```

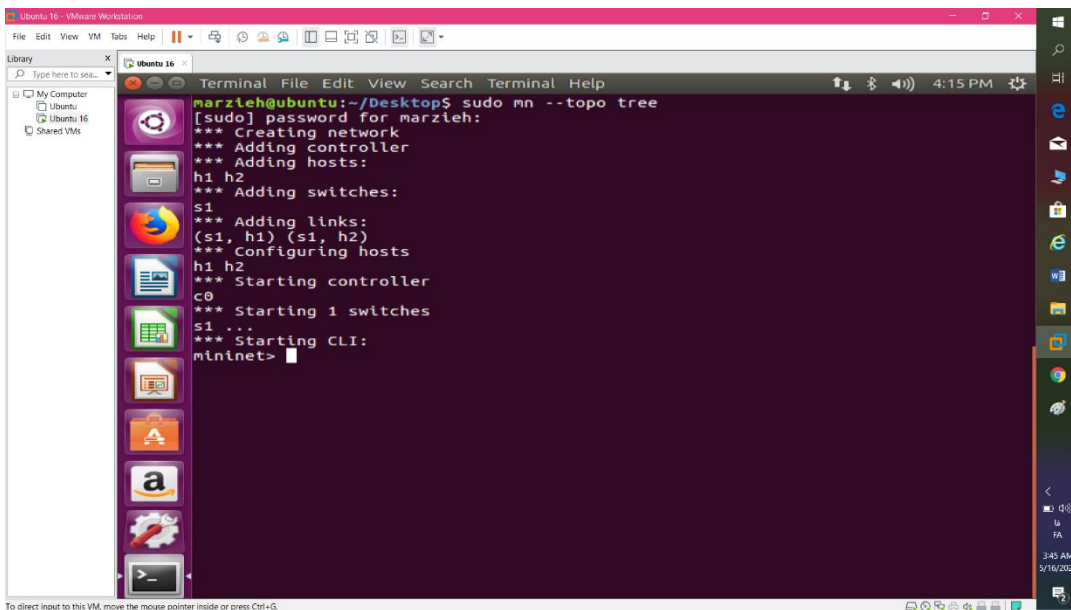
4- توپولوژی tree ، بر مبنای ساختار درختی است. و شامل تعداد مشخص
 n تا level از host ها و switch ها است.
 برای ایجاد این توپولوژی، از command زیر استفاده می شود:
`sudo mn --topo single,depth=n,fanout=n`
 در تصویر زیر، این توپولوژی را ایجاد کردم:
 (شبکه ای که می سازد، ساختارش مثل شکلی ست که در زیر کشیدم:)





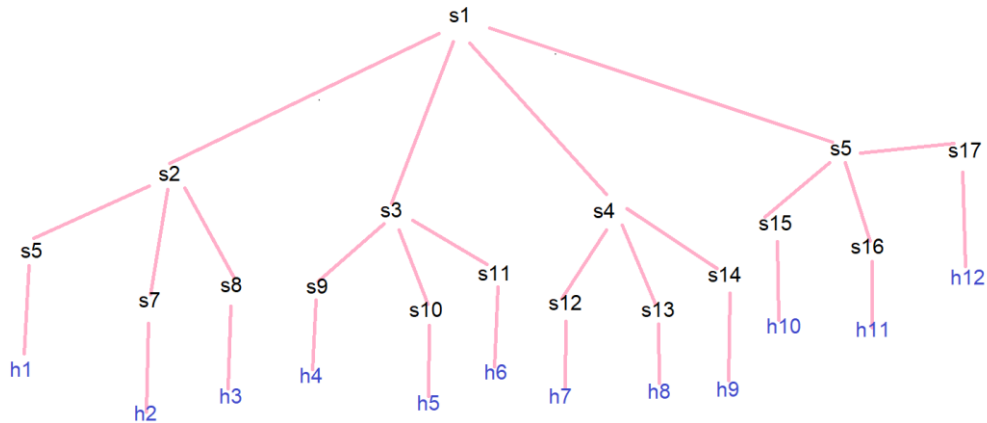
```
marzleh@ubuntu:~/Desktop$ sudo mn --topo tree,2,3
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(s1, s2) (s1, s3) (s1, s4) (s2, h1) (s2, h2) (s2, h3) (s3, h4) (s3, h5) (s3, h6)
(s4, h7) (s4, h8) (s4, h9)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet>
```

همانطور که مشخص است، عدد اول، عمق درخت شبکه، و عدد دوم، تعداد شاخه هایی که به هر node متصل است، را تعیین میکند. اگر این اعداد تعیین نشوند، به ترتیب 1 و 2 در نظر گرفته میشوند: (در نتیجه شبکه ای مثل شبکه ی minimal را تولید میکند)



```
marzleh@ubuntu:~/Desktop$ sudo mn --topo tree
[sudo] password for marzleh:
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(s1, h1) (s1, h2)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

3. در این قسمت، شبکه ای با ساختار زیر ایجاد کردم:
(که کد آن با نام topoOfMine.py ، ضمیمه شده است).



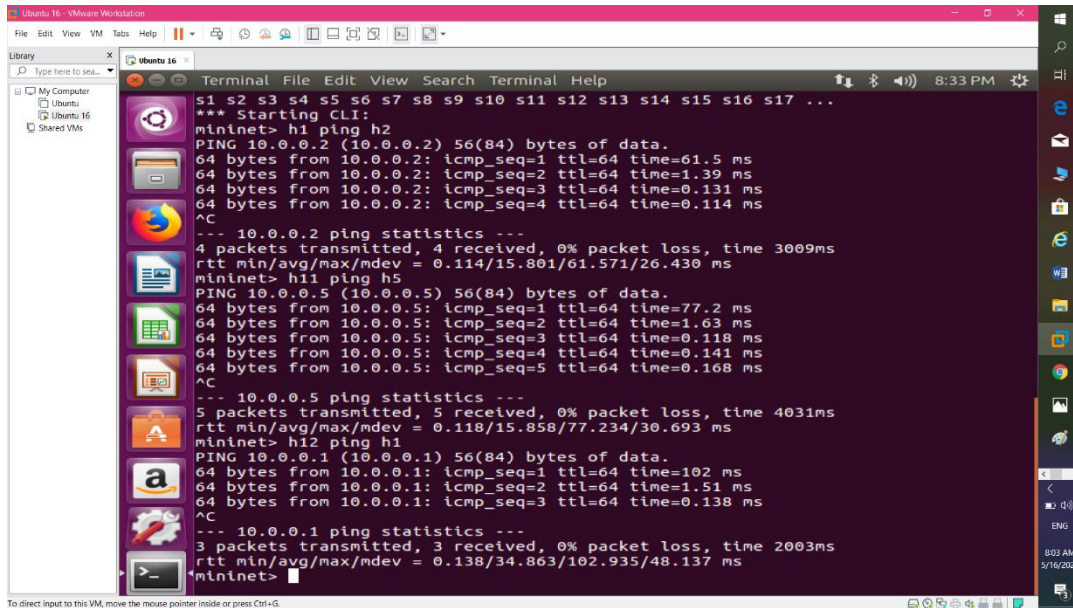
با command زیر آن، را اجرا کردم:
`sudo mn --custom topoOfMine.py --topo topoofmine`

نتیجه ی اجرا:

```

marziah@ubuntu:~/Desktop$ python topoOfMine.py
marziah@ubuntu:~/Desktop$ sudo mn --custom topoOfMine.py --topo topoofmine
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 s12 s13 s14 s15 s16 s17
*** Adding links:
(s1, s2) (s1, s3) (s1, s4) (s1, s5) (s2, s6) (s2, s7) (s2, s8) (s3, s9) (s3, s10)
(s3, s11) (s4, s12) (s4, s13) (s4, s14) (s5, s15) (s5, s16) (s5, s17) (s6, h1)
(s7, h2) (s8, h3) (s9, h4) (s10, h5) (s11, h6) (s12, h7) (s13, h8) (s14, h9) (s15, h10)
(s16, h11) (s17, h12)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12
*** Starting controller
c0
*** Starting 17 switches
s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 s12 s13 s14 s15 s16 s17 ...
*** Starting CLI:
mininet>
  
```

براساس موقعیت های مختلف مکانی host ها، برای چند نمونه از آنها از دستور ping برای اطمینان از درستی شبکه و برقراری ارتباط بین آنها استفاده کردم:



```
s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 s12 s13 s14 s15 s16 s17 ...
*** Starting CLI:
mininet> h1 ping 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=61.5 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=1.39 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.131 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.114 ms
^C
--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3009ms
rtt min/avg/max/mdev = 0.114/15.801/61.571/26.430 ms
mininet> h11 ping h5
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data:
64 bytes from 10.0.0.5: icmp_seq=1 ttl=64 time=77.2 ms
64 bytes from 10.0.0.5: icmp_seq=2 ttl=64 time=1.63 ms
64 bytes from 10.0.0.5: icmp_seq=3 ttl=64 time=0.118 ms
64 bytes from 10.0.0.5: icmp_seq=4 ttl=64 time=0.141 ms
64 bytes from 10.0.0.5: icmp_seq=5 ttl=64 time=0.168 ms
^C
--- 10.0.0.5 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4031ms
rtt min/avg/max/mdev = 0.118/15.858/77.234/30.693 ms
mininet> h12 ping h1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data:
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=102 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=1.51 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.138 ms
^C
--- 10.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.138/34.863/102.935/48.137 ms
mininet>
```

4. این کنترلر یک کنترلر SDN و open source است، که از پروتکل openFlow برای هدایت ترافیک های مربوط به هر flow استفاده می کند. این کنترلر قابلیت هایی دارد؛ از جمله اینکه: با application هایی که به java نوشته شده باشند، سازگار است؛ و شامل یک سری REST API است، که ارتباط برقرار کردن با برنامه ها را آسان تر می کند. معماری آن به صورت ماژولار است. همچنین دارای یک سری application است که به صورت built-in درون آن قرار گرفته اند.