Lab 1 : Introdução ao ambiente de emulação de redes Mininet

Rodrigo Seiji Piubeli Hirao (186837)

16 de dezembro de 2021

Conteúdo

1 Introdução

Este relatório irá introduzir ao funcionamento da interface mininet, criando, assim 3 topologias diferentes por diferentes métodos e sendo analisado o resultado obtido.

2 Metodologia

2.1 Topologia 1

Foi criado o sistema inicial do mininet da Figura 1 e analisado a arquitetura criada.

Figura 1: Topologia 1: 2 hosts conectados por 1 switch

2.2 Topologia 2

Em seguida foi criado um script para simular a arquitetura da figura 2 usando o script a seguir.

```
import atexit
from mininet.net import Mininet
from mininet.topo import Topo
from mininet.cli import CLI
from mininet.log import info,setLogLevel
net = None

def createTopo():
        topo=Topo()
        #Create Nodes
        topo.addHost("h1")
        topo.addHost("h2")
        topo.addHost("h3")
        topo.addHost("h4")
        topo.addSwitch('s1')
        topo.addSwitch('s2')
        topo.addSwitch('s3')
        #Create links
        topo.addLink('s1','s2')
        topo.addLink('s1','s3')
        topo.addLink('h1','s2')
        topo.addLink('h2','s2')
        topo.addLink('h3','s3')
        topo.addLink('h4','s3')
        return topo

def startNetwork():
        topo = createTopo()
        global net
        net = Mininet(topo=topo, autoSetMacs=True)
        net.start()
        CLI(net)
```

```
def stopNetwork():
        if net is not None:
                net.stop()

if __name__ == '__main__':
        atexit.register(stopNetwork)
        setLogLevel('info')
        startNetwork()
```

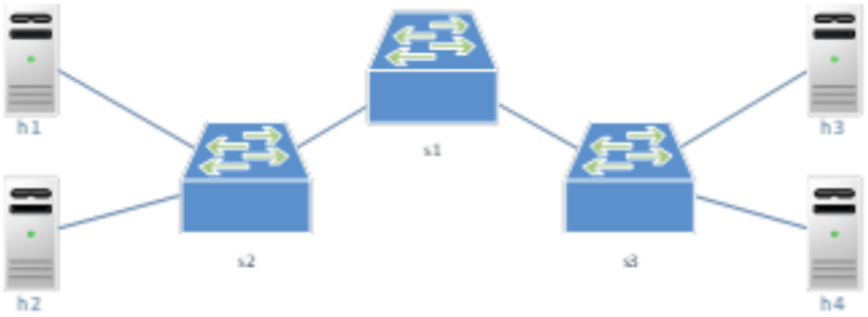

Figura 2: Topologia 2: com 4 hosts conectados por 3 switch

Então foi adicionado latência de 100ms, uma banda de 100Mbps e 10% de perda de pacotes no link do s1 com o s2 com o script a seguir e analisado seu desempenho.

```
import atexit
from mininet.net import Mininet
from mininet.topo import Topo
from mininet.cli import CLI
from mininet.log import info,setLogLevel
from mininet.link import TCLink
net = None

def createTopo():
        topo=Topo()
        #Create Nodes
        topo.addHost("h1")
        topo.addHost("h2")
        topo.addHost("h3")
        topo.addHost("h4")
        topo.addSwitch('s1')
        topo.addSwitch('s2')
        topo.addSwitch('s3')
        #Create links
        topo.addLink('s1','s2',bw=100,delay='100ms',loss=10)
        topo.addLink('s1','s3')
        topo.addLink('h1','s2')
        topo.addLink('h2','s2')
        topo.addLink('h3','s3')
        topo.addLink('h4','s3')
        return topo

def startNetwork():
        topo = createTopo()
        global net
```

```
        net = Mininet(topo=topo, autoSetMacs=True, link=TCLink)
        net.start()
        CLI(net)

def stopNetwork():
        if net is not None:
                net.stop()

if __name__ == '__main__':
        atexit.register(stopNetwork)
        setLogLevel('info')
        startNetwork()
```

2.3 Topologia 3

Por fim foi criado o script a seguir para a Figura 3 com 10Gbps de largura de banda e 1ms de latência entre a primeira camada (core) e a segunda, 1Gbps de larguira de banda e 3ms de latência entre a segunda e a terceira camada e por fim 100Mbps de largura de banda e 5ms de latência entre a terceira camada e os hosts:

```
import atexit
from mininet.net import Mininet
from mininet.topo import Topo
from mininet.cli import CLI
from mininet.log import info,setLogLevel
from mininet.link import TCLink
net = None

def createTopo():
        topo=Topo()
        #Create Nodes
        topo.addSwitch("c1")
        topo.addSwitch("d1")
        topo.addSwitch("d2")
        topo.addSwitch("a1")
        topo.addSwitch("a2")
        topo.addSwitch("a3")
        topo.addSwitch("a4")
        topo.addHost('h1')
        topo.addHost('h2')
        topo.addHost('h3')
        topo.addHost('h4')
        topo.addHost('h5')
        topo.addHost('h6')
        topo.addHost('h7')
        topo.addHost('h8')
        #Create links
        topo.addLink('c1','d1', bw=10000, delay='1ms')
        topo.addLink('c1','d2', bw=10000, delay='1ms')
        topo.addLink('d1','a1', bw=1000, delay='3ms')
        topo.addLink('d1','a2', bw=1000, delay='3ms')
        topo.addLink('d2','a3', bw=1000, delay='3ms')
        topo.addLink('d2','a4', bw=1000, delay='3ms')
        topo.addLink('a1','h1', bw=100, delay='5ms')
        topo.addLink('a1','h2', bw=100, delay='5ms')
        topo.addLink('a2','h3', bw=100, delay='5ms')
        topo.addLink('a2','h4', bw=100, delay='5ms')
        topo.addLink('a3','h5', bw=100, delay='5ms')
        topo.addLink('a3','h6', bw=100, delay='5ms')
        topo.addLink('a4','h7', bw=100, delay='5ms')
```

```
        topo.addLink('a4','h8', bw=100, delay='5ms', loss=15)
        return topo

def startNetwork():
        topo = createTopo()
        global net
        net = Mininet(topo=topo, autoSetMacs=True, link=TCLink)
        net.start()
        CLI(net)

def stopNetwork():
        if net is not None:
                net.stop()

if __name__ == '__main__':
        atexit.register(stopNetwork)
        setLogLevel('info')
        startNetwork(
```

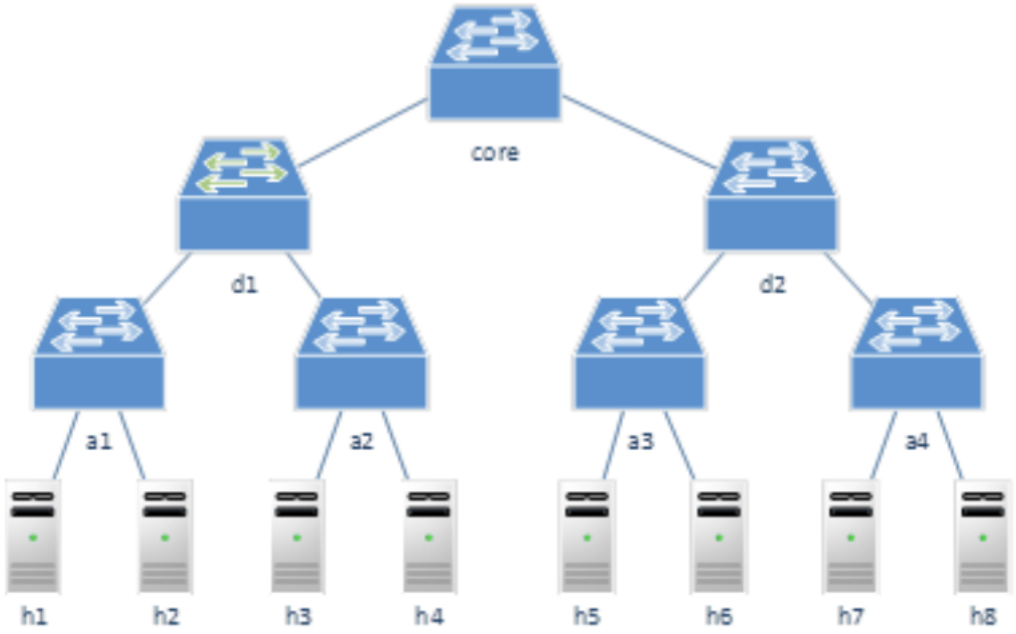

Figura 3: Topologia 3: datacenter

Assim, ao introduzir uma perda de 15% entre h8 e a4, a rede foi analisada e seu ping calculado da forma da equação 1, e medido as diversas larguras de banda do sistema.

$$ping_{a4->h8} = \frac{ping_{h6->h8} - ping_{h6->h7}}{2} \quad (1)$$

3 Resultados

3.1 Topologia 1

3.1.1 Dados básicos da rede

Nome	Tipo	Interface	IPv4	MAC
h1	Host	h1-et0	10.0.0.1	00:00:00:00:00:01
h2	Host	h2-et0	10.0.0.2	00:00:00:00:00:02
s1	Switch	enp0s3	10.0.2.15	08:00:27:49:2f:c8

Figura 4: Dados de cada aparelho

3.1.2 Tabelas de roteamento

Destino	Gateway	Mask	Interface
0.0.0.0	10.0.2.2	0.0.0.0	enp0s3
10.0.2.0	0.0.0.0	255.255.255.0	enp0s3

Figura 5: Tabela de roteamento (s1)

Destino	Gateway	Mask	Interface
10.0.0.0	0.0.0.0	255.0.0.0	h1-eth0

Figura 6: Tabela de roteamento (h1)

Destino	Gateway	Mask	Interface
10.0.0.0	0.0.0.0	255.0.0.0	h2-eth0

Figura 7: Tabela de roteamento (h2)

3.1.3 Tabelas ARP

Endereço	MAC	Interface
10.0.0.2	00:00:00:00:00:02	h1-eth0

Figura 8: Tabela ARP (h1)

Endereço	MAC	Interface
10.0.0.1	00:00:00:00:00:01	h2-eth0

Figura 9: Tabela ARP (h2)

3.2 Topologia 2

3.2.1 pingall

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
```

3.2.2 ping individual

```
mininet> h1 ping h4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=233 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=201 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=201 ms
--- 10.0.0.4 ping statistics ---
4 packets transmitted, 3 received, 25% packet loss, time 3083ms
rtt min/avg/max/mdev = 200.587/211.546/233.460/15.495 ms

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=0.364 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.077 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.212 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.110
--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3095ms
rtt min/avg/max/mdev = 0.077/0.190/0.364/0.111 ms
```

3.2.3 iperf

```
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['2.31 Gbits/sec', '2.33 Gbits/sec']

mininet> iperf h1 h4
*** Iperf: testing TCP bandwidth between h1 and h4
*** Results: ['373 Kbits/sec', '491 Kbits/sec']
```

3.3 Topologia 3

3.3.1 latência entre h8 e a4

```
mininet> h6 ping h7
PING 10.0.0.7 (10.0.0.7) 56(84) bytes of data.
64 bytes from 10.0.0.7: icmp_seq=1 ttl=64 time=68.7 ms
64 bytes from 10.0.0.7: icmp_seq=2 ttl=64 time=43.7 ms
64 bytes from 10.0.0.7: icmp_seq=3 ttl=64 time=34.8 ms
64 bytes from 10.0.0.7: icmp_seq=4 ttl=64 time=34.7 ms
64 bytes from 10.0.0.7: icmp_seq=5 ttl=64 time=33.7 ms
64 bytes from 10.0.0.7: icmp_seq=6 ttl=64 time=33.8 ms
64 bytes from 10.0.0.7: icmp_seq=7 ttl=64 time=34.1 ms
64 bytes from 10.0.0.7: icmp_seq=8 ttl=64 time=37.1 ms
64 bytes from 10.0.0.7: icmp_seq=9 ttl=64 time=34.7 ms
64 bytes from 10.0.0.7: icmp_seq=10 ttl=64 time=34.3 ms

--- 10.0.0.7 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9014ms
rtt min/avg/max/mdev = 33.663/38.946/68.742/10.335 ms
mininet> h6 ping h8
PING 10.0.0.8 (10.0.0.8) 56(84) bytes of data.
64 bytes from 10.0.0.8: icmp_seq=1 ttl=64 time=76.1 ms
64 bytes from 10.0.0.8: icmp_seq=3 ttl=64 time=34.7 ms
64 bytes from 10.0.0.8: icmp_seq=5 ttl=64 time=35.5 ms
64 bytes from 10.0.0.8: icmp_seq=7 ttl=64 time=33.9 ms
64 bytes from 10.0.0.8: icmp_seq=8 ttl=64 time=34.4 ms
64 bytes from 10.0.0.8: icmp_seq=9 ttl=64 time=34.2 ms
64 bytes from 10.0.0.8: icmp_seq=10 ttl=64 time=33.4 ms
```

```
--- 10.0.0.8 ping statistics ---
10 packets transmitted, 7 received, 30% packet loss, time 9056ms
rtt min/avg/max/mdev = 33.414/40.303/76.067/14.613 ms
```

Logo: $ping_{a4->h8} = \frac{30\%-0}{2} = 15\%$

3.3.2 Banda do Sistema

```
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['90.9 Mbits/sec', '106 Mbits/sec']
mininet> iperf h1 h3
*** Iperf: testing TCP bandwidth between h1 and h3
*** Results: ['91.5 Mbits/sec', '107 Mbits/sec']
mininet> iperf h1 h5
*** Iperf: testing TCP bandwidth between h1 and h5
*** Results: ['91.3 Mbits/sec', '106 Mbits/sec']
```

4 Discussão

Na primeira topologia pode ser visto pela tabela arp que h1 consegue enxergar h2 e vice-versa pois estão na mesma subnet, o que pode ser visto em suas tabelas de roteamento.

Na segunda topologia é possível ver a perda de pacotes introduzido na rede ao comparar conexões que passam pelo switch s2 e conexões que não precisam passar pelo switch s2. Além disso também é possível perceber, pelo mesmo método, a perda de banda devido à baixa banda entre s1 e s2.

Na terceira topologia é possível ver a eficiência da rede em árvore montada e que a velocidade máxima alcançada foi a dos nós. Também foi estudado a introdução de uma perda em um dos nós, que foi calculado como 15% e teve um resultado prático de mesmo valor.

Como pode ser visto no arquivo de tcpdump do wireshark anexado sobre a topologia 3, pode ser visto o mesmo comportamento pelo ping dos hosts, e também pode ser visto requisições de broadcast para disponibilizar os MAC dos IPs presentes na subrede.

5 Conclusão

A interface mininet é uma interface de fácil programação e simulação, com uma grande coleção de ferramentas e opções para tentar aproximar a simulação de um caso real.

Os sistemas criadas mostraram diversas maneiras de criar uma topologia, bem como as diversas ferramentas do linux para analizar a rede. Todas as topologias mostraram resultados de acordo com os esperados teoricamente, provavelmente por serem apenas simulações.