



دانشکده مهندسی کامپیوتر

آزمایشگاه شبکه‌های کامپیوتری

گزارش کار آزمایش *

گروه ۴

علی صدیقی ۹۷۵۲۱۳۷۸

دانیال بازمانده ۹۷۵۲۱۱۳۵

۱ سوال اول

دستور پینگ را برای مقاصد زیر اجرا می کنیم:

ping google.com

با توجه به اینکه دستور را در CMD وارد می کنیم ۵ بار بسته فرستاده می شود که میانگین آن ها را در نظر می گیریم. نتایج حالتی که VPN روی سرور هلند روشن بود:

Destination	RTT (ms)	TTL
google.com	179	57
yahoo.com	253	50
varzesh3.com	261	48
isna.ir	200	57
8.8.8.8	165	53

نتایج حالتی که VPN خاموش بود:

Destination	RTT (ms)	TTL
google.com	80	50
yahoo.com *	452	47
varzesh3.com	64	56
isna.ir	37	57
8.8.8.8	78	112

* With Iran IP we had 1 packet loss from 4 packets.

```
C:\Users\alise>ping google.com
Pinging google.com [172.217.18.142] with 32 bytes of data:
Reply from 172.217.18.142: bytes=32 time=77ms TTL=50
Reply from 172.217.18.142: bytes=32 time=85ms TTL=50
Reply from 172.217.18.142: bytes=32 time=85ms TTL=50
Reply from 172.217.18.142: bytes=32 time=73ms TTL=50

Ping statistics for 172.217.18.142:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 73ms, Maximum = 85ms, Average = 80ms

C:\Users\alise>ping yahoo.com
Pinging yahoo.com [98.137.11.163] with 32 bytes of data:
Reply from 98.137.11.163: bytes=32 time=591ms TTL=47
Request timed out.
Reply from 98.137.11.163: bytes=32 time=332ms TTL=47
Reply from 98.137.11.163: bytes=32 time=433ms TTL=47

Ping statistics for 98.137.11.163:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 332ms, Maximum = 591ms, Average = 452ms

C:\Users\alise>ping varzesh3.com
Pinging varzesh3.com [94.182.113.148] with 32 bytes of data:
Reply from 94.182.113.148: bytes=32 time=58ms TTL=56
Reply from 94.182.113.148: bytes=32 time=126ms TTL=56
Reply from 94.182.113.148: bytes=32 time=37ms TTL=56
Reply from 94.182.113.148: bytes=32 time=38ms TTL=56

Ping statistics for 94.182.113.148:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 37ms, Maximum = 126ms, Average = 64ms

C:\Users\alise>ping isna.ir
Pinging isna.ir [185.143.233.3] with 32 bytes of data:
Reply from 185.143.233.3: bytes=32 time=37ms TTL=57
Reply from 185.143.233.3: bytes=32 time=40ms TTL=57
Reply from 185.143.233.3: bytes=32 time=36ms TTL=57
Reply from 185.143.233.3: bytes=32 time=36ms TTL=57

Ping statistics for 185.143.233.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 36ms, Maximum = 40ms, Average = 37ms

C:\Users\alise>ping 8.8.8.8
Pinging 8.8.8.8 with 32 bytes of data:
Reply from 8.8.8.8: bytes=32 time=74ms TTL=112
Reply from 8.8.8.8: bytes=32 time=83ms TTL=112
Reply from 8.8.8.8: bytes=32 time=74ms TTL=112
Reply from 8.8.8.8: bytes=32 time=81ms TTL=112

Ping statistics for 8.8.8.8:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 74ms, Maximum = 83ms, Average = 78ms
```

مفهوم Round Trip Time: RTT مدت زمانی است که یک بسته از مبدا به مقصد رفته، سپس بسته حاوی خبر رسیدن آن (Acknowledge) از مقصد به مبدا رسیده است. به عبارت دیگر RTT مجموع زمان رفتن Packet و بازگشت Ack است.

مفهوم Time To Live: TTL بیانگر طول عمر و مدت زمان زنده بودن یک بسته در طول مسیر است. در واقع یک بسته پس از عبور از هر Hop در شبکه یک مقدار از TTL ش کاسته می‌شود. TTL از سرگردان بودن بسته در شبکه جلوگیری می‌کند. (حالتی را فرض کنید که بسته‌ای مدت زیادی بین سوییچ‌ها و روترها جابجا می‌شود بدون اینکه قرار باشد به مقصد برسد) TTL واحد ندارد زیرا بیانگر حداکثر تعداد Hop است. پیشینه مقدار TTL برای یک بسته 255 است (زیرا توسط ۸ بیت در Header بسته مشخص می‌شود).

رابطه TTL و RTT: میان این دو مقدار رابطه مطلق وجود ندارد. در جدول‌های بالا به صورت کلی می‌توان گفت هر چه RTT بیشتر باشد TTL کمتر است (رابطه عکس). هر چه TTL برای یک بسته بیشتر باشد به معنی این است که احتمالاً تعداد Hop بیشتری برای آن بسته وجود دارد. اما افزایش تعداد Hop الزاماً به معنی زیاد شدن زمان انتقال نیست. زیرا دو عامل طول مسیر و سرعت لینک Hop-to-Hop نیز تاثیر گذار است.

۲ سوال دوم

ابتدا ماشین مجازی را روشن می‌کنیم. سپس با کاربر mininet وارد می‌شویم. برای به دست آوردن آدرس ماشین مجازی دستور ip add را اجرا می‌کنیم. برای IP سیستم اصلی نیز دستور ipconfig را می‌زنیم.

```
mininet@mininet-vm:~$ ip add
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:9b:2e:f4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.83.3/24 brd 192.168.83.255 scope global dynamic eth0
        valid_lft 416sec preferred_lft 416sec
3: ovs-system: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 6e:1e:a9:12:81:f1 brd ff:ff:ff:ff:ff:ff
4: s1: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 36:6a:64:fb:2d:44 brd ff:ff:ff:ff:ff:ff
5: s3: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 4e:7a:f4:67:69:41 brd ff:ff:ff:ff:ff:ff
6: s2: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether a2:f3:59:e9:18:47 brd ff:ff:ff:ff:ff:ff
mininet@mininet-vm:~$ _
```

Ethernet adapter VirtualBox Host-Only Network:

```
Connection-specific DNS Suffix . :
Link-local IPv6 Address . . . . . : fe80::9c24:f0f5:f185:41f2%36
IPv4 Address. . . . . : 192.168.83.4
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . :
```

VM IP: 192.168.83.3

My Machine IP: 192.168.83.4

برنامه Wireshark را اجرا می‌کنیم. آداپتر مربوط به Virtual Box را انتخاب می‌کنیم. شروع به رکورد کردن بسته‌ها می‌کنیم.

با استفاده از دستور زیر پینگ می‌کنیم:

ping 192.168.83.3 -n 5

```
C:\Users\alise>ping 192.168.83.3 -n 5

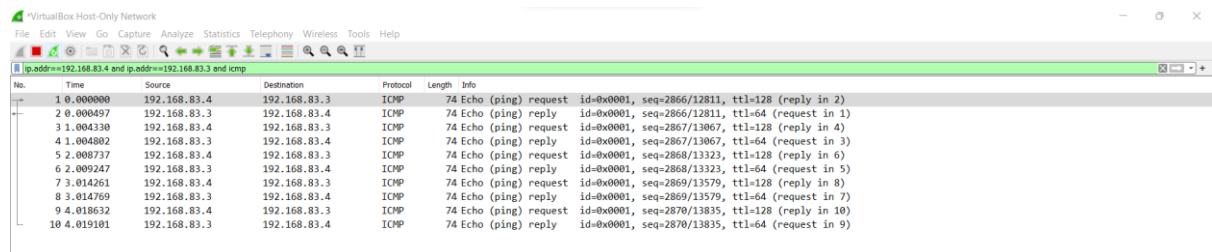
Pinging 192.168.83.3 with 32 bytes of data:
Reply from 192.168.83.3: bytes=32 time<1ms TTL=64
Reply from 192.168.83.3: bytes=32 time<1ms TTL=64
Reply from 192.168.83.3: bytes=32 time<1ms TTL=64
Reply from 192.168.83.3: bytes=32 time<1ms TTL=64
Reply from 192.168.83.3: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.83.3:
    Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

با استفاده از فیلتر زیر بسته‌ها مطلوب را جدا می‌کنیم. فیلتر باید تمام بسته‌های بین دو IP ماشین مجازی و لپ‌تاپ ما که از نوع icmp (پروتکل پینگ) هستند را شامل شود.

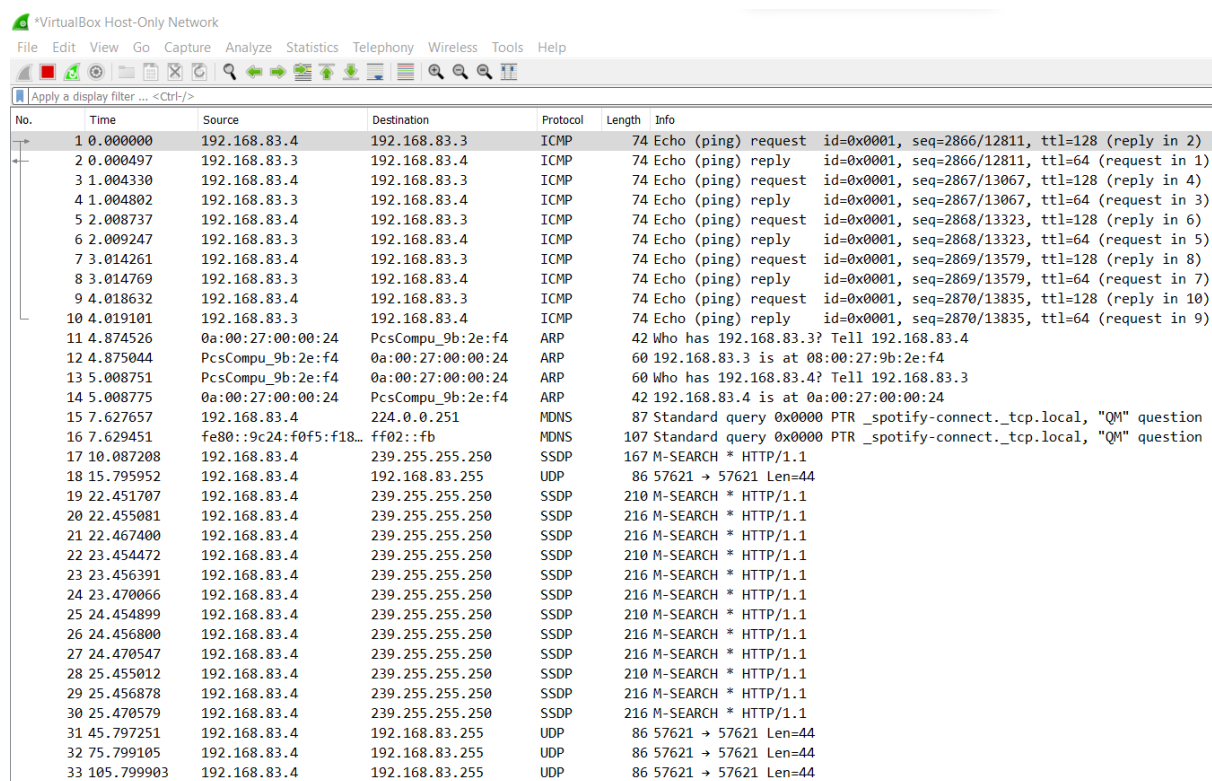
`ip.addr==192.168.83.4 and ip.addr==192.168.83.3 and icmp`

از `ip.addr` به جای `ip.src` و `ip.dst` استفاده کردیم زیرا هم ارسال و پاسخ بسته‌ها را می‌خواستیم.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.83.4	192.168.83.3	ICMP	74	Echo (ping) request id=0x0001, seq=2866/12811, ttl=128 (reply in 2)
2	0.000497	192.168.83.3	192.168.83.4	ICMP	74	Echo (ping) reply id=0x0001, seq=2866/12811, ttl=64 (request in 1)
3	1.004330	192.168.83.4	192.168.83.3	ICMP	74	Echo (ping) request id=0x0001, seq=2867/13067, ttl=128 (reply in 4)
4	1.004802	192.168.83.3	192.168.83.4	ICMP	74	Echo (ping) reply id=0x0001, seq=2867/13067, ttl=64 (request in 3)
5	2.008737	192.168.83.4	192.168.83.3	ICMP	74	Echo (ping) request id=0x0001, seq=2868/13323, ttl=128 (reply in 6)
6	2.009247	192.168.83.3	192.168.83.4	ICMP	74	Echo (ping) reply id=0x0001, seq=2868/13323, ttl=64 (request in 5)
7	3.014261	192.168.83.4	192.168.83.3	ICMP	74	Echo (ping) request id=0x0001, seq=2869/13579, ttl=128 (reply in 8)
8	3.014769	192.168.83.3	192.168.83.4	ICMP	74	Echo (ping) reply id=0x0001, seq=2869/13579, ttl=64 (request in 7)
9	4.018632	192.168.83.4	192.168.83.3	ICMP	74	Echo (ping) request id=0x0001, seq=2870/13835, ttl=128 (reply in 10)
10	4.019101	192.168.83.3	192.168.83.4	ICMP	74	Echo (ping) reply id=0x0001, seq=2870/13835, ttl=64 (request in 9)

تصویر بدون فیلتر:



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.83.4	192.168.83.3	ICMP	74	Echo (ping) request id=0x0001, seq=2866/12811, ttl=128 (reply in 2)
2	0.000497	192.168.83.3	192.168.83.4	ICMP	74	Echo (ping) reply id=0x0001, seq=2866/12811, ttl=64 (request in 1)
3	1.004330	192.168.83.4	192.168.83.3	ICMP	74	Echo (ping) request id=0x0001, seq=2867/13067, ttl=128 (reply in 4)
4	1.004802	192.168.83.3	192.168.83.4	ICMP	74	Echo (ping) reply id=0x0001, seq=2867/13067, ttl=64 (request in 3)
5	2.008737	192.168.83.4	192.168.83.3	ICMP	74	Echo (ping) request id=0x0001, seq=2868/13323, ttl=128 (reply in 6)
6	2.009247	192.168.83.3	192.168.83.4	ICMP	74	Echo (ping) reply id=0x0001, seq=2868/13323, ttl=64 (request in 5)
7	3.014261	192.168.83.4	192.168.83.3	ICMP	74	Echo (ping) request id=0x0001, seq=2869/13579, ttl=128 (reply in 8)
8	3.014769	192.168.83.3	192.168.83.4	ICMP	74	Echo (ping) reply id=0x0001, seq=2869/13579, ttl=64 (request in 7)
9	4.018632	192.168.83.4	192.168.83.3	ICMP	74	Echo (ping) request id=0x0001, seq=2870/13835, ttl=128 (reply in 10)
10	4.019101	192.168.83.3	192.168.83.4	ICMP	74	Echo (ping) reply id=0x0001, seq=2870/13835, ttl=64 (request in 9)
11	4.874526	0a:00:27:00:00:24	PcsCompu_9b:2e:f4	ARP	42	Who has 192.168.83.3? Tell 192.168.83.4
12	4.875044	PcsCompu_9b:2e:f4	0a:00:27:00:00:24	ARP	60	192.168.83.3 is at 08:00:27:9b:2e:f4
13	5.008751	PcsCompu_9b:2e:f4	0a:00:27:00:00:24	ARP	60	Who has 192.168.83.4? Tell 192.168.83.3
14	5.008775	0a:00:27:00:00:24	PcsCompu_9b:2e:f4	ARP	42	192.168.83.4 is at 0a:00:27:00:00:24
15	7.627657	192.168.83.4	224.0.0.251	MDNS	87	Standard query 0x0000 PTR _spotify-connect_tcp.local, "QM" question
16	7.629451	fe80::9c24:f0f5:f18...	ff02::fb	MDNS	107	Standard query 0x0000 PTR _spotify-connect_tcp.local, "QM" question
17	10.087208	192.168.83.4	239.255.255.250	SSDP	167	M-SEARCH * HTTP/1.1
18	15.795952	192.168.83.4	192.168.83.255	UDP	86	57621 → 57621 Len=44
19	22.451707	192.168.83.4	239.255.255.250	SSDP	210	M-SEARCH * HTTP/1.1
20	22.455081	192.168.83.4	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
21	22.467400	192.168.83.4	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
22	23.454472	192.168.83.4	239.255.255.250	SSDP	210	M-SEARCH * HTTP/1.1
23	23.456391	192.168.83.4	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
24	23.470066	192.168.83.4	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
25	24.454899	192.168.83.4	239.255.255.250	SSDP	210	M-SEARCH * HTTP/1.1
26	24.456800	192.168.83.4	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
27	24.470547	192.168.83.4	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
28	25.455012	192.168.83.4	239.255.255.250	SSDP	210	M-SEARCH * HTTP/1.1
29	25.456878	192.168.83.4	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
30	25.470579	192.168.83.4	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
31	45.797251	192.168.83.4	192.168.83.255	UDP	86	57621 → 57621 Len=44
32	75.799105	192.168.83.4	192.168.83.255	UDP	86	57621 → 57621 Len=44
33	105.799903	192.168.83.4	192.168.83.255	UDP	86	57621 → 57621 Len=44

(الف) مورد الف همان توپولوژی minimal است.

`sudo mn --topo minimal`

```
mininet@mininet-vm:~$ 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> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo: s1-eth1:h1-eth0 s1-eth2:h2-eth0
c0
mininet> echo ALI SEDAGHI
*** Unknown command: echo ALI SEDAGHI
mininet> _
```

اتصالات (همگی دو طرفی هستند، فقط یک طرف نوشته شده است):

- هاست h1 از طریق پورت eth0 به سویچ s1 از طریق پورت eth1
- هاست h2 از طریق پورت eth0 به سویچ s1 از طریق پورت eth2

(ب) این مورد توپولوژی خطی (Linear) با ۲ سویچ که به هر سویچ ۲ هاست وصل است.

`sudo mn --topo linear,2,2`

```
mininet@mininet-vm:~$ sudo mn --topo linear,2,2
*** Creating network
*** Adding controller
*** Adding hosts:
h1s1 h1s2 h2s1 h2s2
*** Adding switches:
s1 s2
*** Adding links:
(h1s1, s1) (h1s2, s2) (h2s1, s1) (h2s2, s2) (s2, s1)
*** Configuring hosts
h1s1 h1s2 h2s1 h2s2
*** Starting controller
c0
*** Starting 2 switches
s1 s2 ...
*** Starting CLI:
mininet> net
h1s1 h1s1-eth0:s1-eth1
h1s2 h1s2-eth0:s2-eth1
h2s1 h2s1-eth0:s1-eth2
h2s2 h2s2-eth0:s2-eth2
s1 lo: s1-eth1:h1s1-eth0 s1-eth2:h2s1-eth0 s1-eth3:s2-eth3
s2 lo: s2-eth1:h1s2-eth0 s2-eth2:h2s2-eth0 s2-eth3:s1-eth3
c0
mininet> echo SEDAGHI
*** Unknown command: echo SEDAGHI
mininet> _
```

اتصالات (همگی دو طرفی هستند، فقط یک طرف نوشته شده است):

- هاست h1s1 از طریق پورت eth0 به سویچ s1 از طریق پورت eth1
- هاست h1s2 از طریق پورت eth0 به سویچ s2 از طریق پورت eth1
- هاست h2s1 از طریق پورت eth0 به سویچ s1 از طریق پورت eth2
- هاست h2s2 از طریق پورت eth0 به سویچ s2 از طریق پورت eth2
- سویچ s1 از طریق پورت eth3 به سویچ s2 از طریق پورت eth3

(ج) توپولوژی درختی (tree) با عمق ۲ است که هر پدر ۳ فرزند دارد (fanout)

`sudo mn --topo tree,depth=2,fanout=3`

```
mininet@mininet-vm:~$ sudo mn --topo tree,depth=2,fanout=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> net
h1 h1-eth0:s2-eth1
h2 h2-eth0:s2-eth2
h3 h3-eth0:s2-eth3
h4 h4-eth0:s3-eth1
h5 h5-eth0:s3-eth2
h6 h6-eth0:s3-eth3
h7 h7-eth0:s4-eth1
h8 h8-eth0:s4-eth2
h9 h9-eth0:s4-eth3
s1 lo: s1-eth1:s2-eth4 s1-eth2:s3-eth4 s1-eth3:s4-eth4
s2 lo: s2-eth1:h1-eth0 s2-eth2:h2-eth0 s2-eth3:h3-eth0 s2-eth4:s1-eth1
s3 lo: s3-eth1:h4-eth0 s3-eth2:h5-eth0 s3-eth3:h6-eth0 s3-eth4:s1-eth2
s4 lo: s4-eth1:h7-eth0 s4-eth2:h8-eth0 s4-eth3:h9-eth0 s4-eth4:s1-eth3
c0
mininet> _
```

اتصالات (همگی دو طرفی هستند، فقط یک طرف نوشته شده است):

- هاستهای h1، h2 و h3 از طریق پورت eth0 خود به سویچ s2 از طریق پورتهای eth1، eth2 و eth3
- هاستهای h4، h5 و h6 از طریق پورت eth0 خود به سویچ s3 از طریق پورتهای eth1، eth2 و eth3
- هاستهای h7، h8 و h9 از طریق پورت eth0 خود به سویچ s4 از طریق پورتهای eth1، eth2 و eth3
- سویچهای s2، s3 و s4 از طریق پورت eth4 خود به سویچ s1 از طریق پورتهای eth1، eth2 و eth3

۴ سوال چهارم

برای ایجاد شبکه جدول اول دستور زیر را وارد می‌کنیم:

```
sudo mn --topo minimal --link tc,bw=100,delay=0.01ms
```

مقدار delay را به ازای هر ۱۰ مقدار عوض می‌کنیم.

برای محاسبه پینگ میان دو هاست از دستور زیر استفاده می‌کنیم (میانگین):

```
pingallfull
```

برای محاسبه پهنای باند میان دو هاست از دستور زیر استفاده می‌کنیم:

```
iperf
```

```
mininet@mininet-vm:~$ sudo mn --topo minimal --link tc,bw=100,delay=0.01ms
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(100.00Mbit 0.01ms delay) (100.00Mbit 0.01ms delay) (h1, s1) (100.00Mbit 0.01ms delay) (100.00Mbit 0.01ms delay) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ... (100.00Mbit 0.01ms delay) (100.00Mbit 0.01ms delay)
*** Starting CLI:
mininet> pingallfull
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results:
h1->h2: 1/1, rtt min/avg/max/mdev 4.866/4.866/4.866/0.000 ms
h2->h1: 1/1, rtt min/avg/max/mdev 2.209/2.209/2.209/0.000 ms
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['93.4 Mbits/sec', '111 Mbits/sec']
mininet> _
```

کار بالا را به ازای هر ۱۰ تاخیر انجام می‌دهیم. نتایج را در جدول ذکر می‌کنیم و دیگر تصاویر خروجی را در گزارش قرار نمی‌دهیم. (پهنای باند ثابت = 100Mbps)

Delay (ms)	RTT (ms)	Measured BW (Mb/s)
0.01	4.8 - 2.2	93.4 - 111
0.05	5.7 - 2.1	92.4 - 109
0.1	5.0 - 2.0	92.7 - 109
0.5	12.8 - 3.4	92.7 - 109
1.0	11.0 - 5.9	91.5 - 108
5.0	45.0 - 22.3	91.4 - 108
10.0	85.6 - 41.9	90.3 - 105
50.0	404.9 - 202.5	72.3 - 82.4
100.0	813.8 - 403.3	44.3 - 49.1
500.0	4006.4 - 2002.4	0.314 - 0.523

✓ مقدار سمت چپ (L): h1 -> h2

✓ مقدار سمت راست (R): h2 -> h1

برای ایجاد شبکه جدول دوم دستور زیر را وارد می‌کنیم:

```
sudo mn --topo minimal --link tc,bw=0.01,delay=1ms
```

```
mininet@mininet-vm:~$ sudo mn --topo minimal --link tc,bw=0.05,delay=1ms
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(0.05Mbit 1ms delay) (0.05Mbit 1ms delay) (h1, s1) (0.05Mbit 1ms delay) (0.05Mbit 1ms delay) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ... (0.05Mbit 1ms delay) (0.05Mbit 1ms delay)
*** Starting CLI:
mininet> pingallfull
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results:
  h1->h2: 1/1, rtt min/avg/max/mdev 11.338/11.338/11.338/0.000 ms
  h2->h1: 1/1, rtt min/avg/max/mdev 5.496/5.496/5.496/0.000 ms
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['48.3 Kbits/sec', '354 Kbits/sec']
mininet>
```

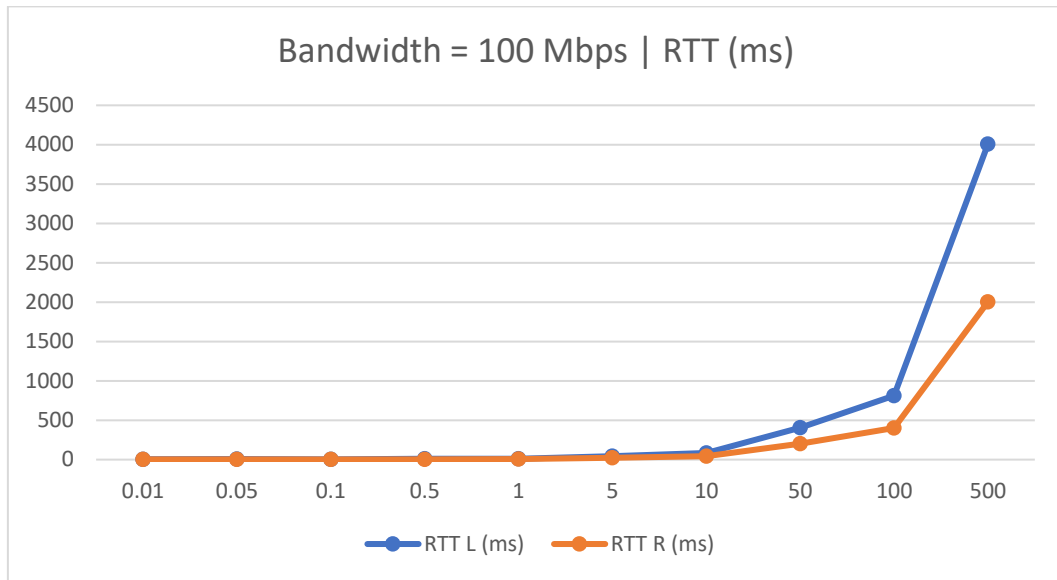
کار بالا را به ازای هر ۱۰ پهنای باند انجام می‌دهیم. نتایج را در جدول ذکر می‌کنیم و دیگر تصاویر خروجی را در گزارش قرار نمی‌دهیم. (تاخیر ثابت = 1ms)

Bandwidth (Mb/s)	RTT (ms)	Measured BW (Mb/s)
0.01	11.0 - 5.7	0.009 - 0.220
0.05	11.3 - 5.4	0.048 - 0.354
0.1	11.3 - 5.0	0.096 - 0.514
0.5	11.5 - 5.3	0.480 - 1.01
1.0	17.0 - 5.4	0.959 - 1.61
5.0	19.4 - 5.7	4.77 - 6.26
10.0	11.1 - 5.2	9.54 - 12.2
50.0	16.0 - 5.2	47.4 - 56.5
100.0	11.9 - 5.3	92.2 - 109
500.0	16.7 - 17.3	388 - 406

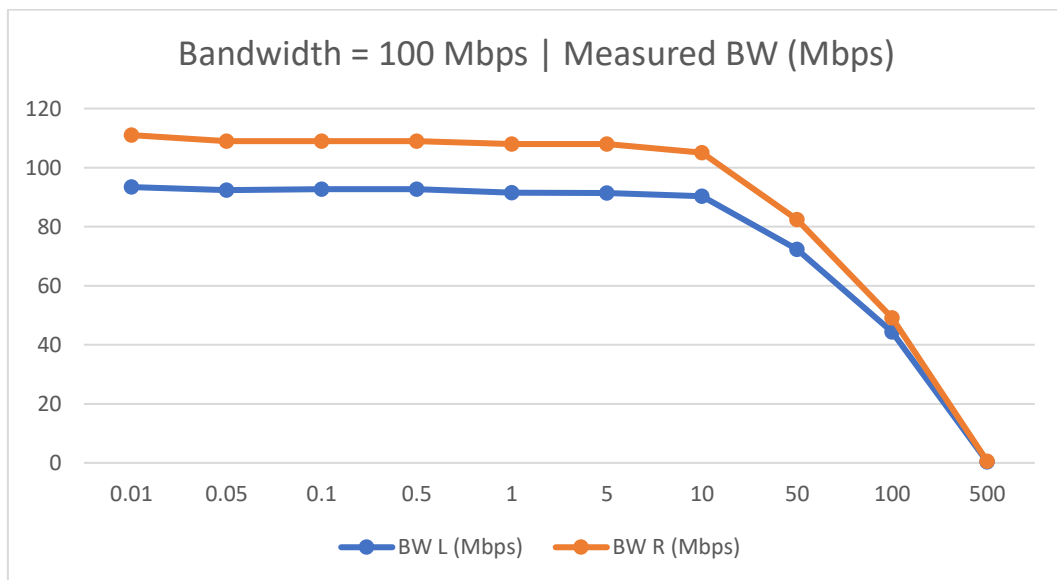
✓ مقدار سمت چپ (L): h1 -> h2

✓ مقدار سمت راست (R): h2 -> h1

ثابت بودن پهنای باند و تغییر Delay:

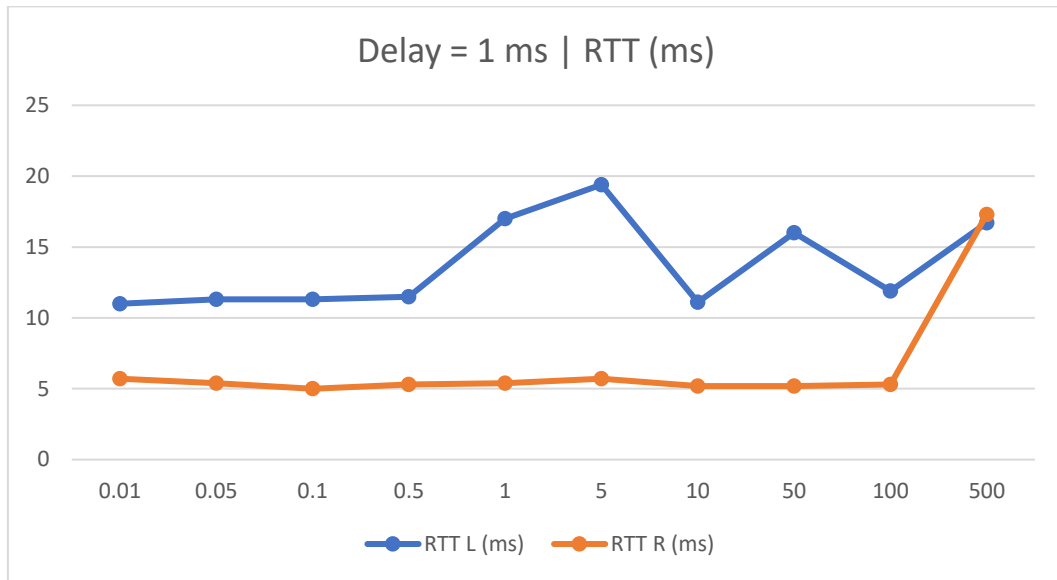


با افزایش Delay لینک‌ها RTT افزایش می‌یابد: هر بسته در مسیر رفت و برگشت ۴ بار روی لینک قرار می‌گیرد (دو لینک، ارسال و پاسخ). اگر تاخیر لینک‌ها افزایش یابد بدیهی است که RTT نیز حداقل به اندازه ۴ برابر Delay افزایش می‌یابد. از نمودار بالا پیداست که نحوه افزایش RTT نسبت به Delay به صورت نمایی است (۴ برابر شدن در هر مرحله)

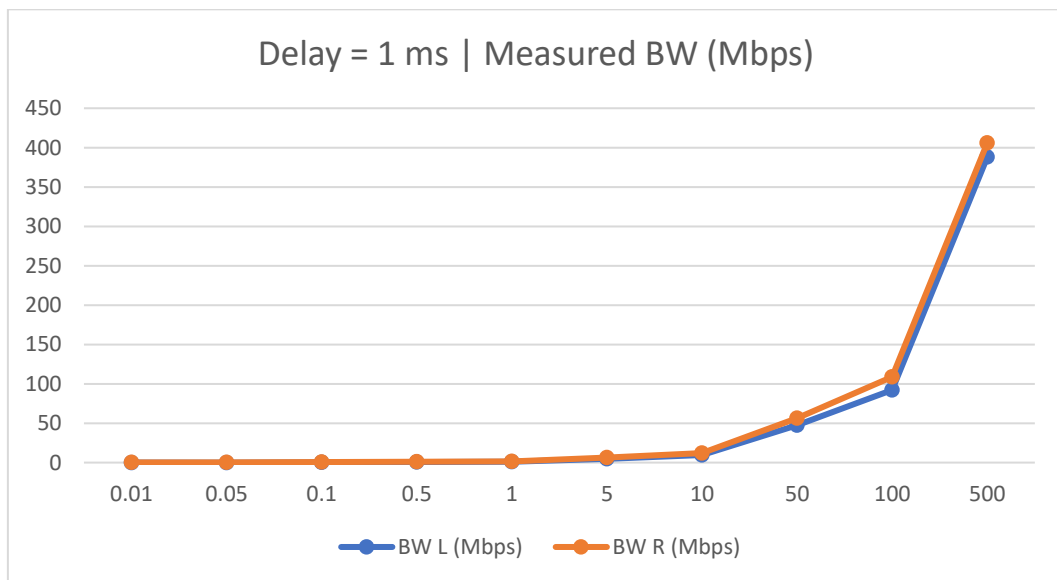


با افزایش Delay لینک‌ها، پهنای باند کاهش می‌یابد: هنگامی که تاخیر لینک زیاد می‌شود، زمان بیشتری طول می‌کشد که داده روی لینک منتقل شود. با افزایش زمان (مخرج کسر پهنای باند) پهنای باند واقعی کاهش می‌یابد. به عبارت دیگر به دلیل وجود تاخیر، حجم مشخصی از داده در زمان طولانی تری منتقل می‌شود. البته کاهش پهنای باند هنگامی که تاخیر حدود 10ms است ناچیز است. اما با افزایش ناگهانی تاخیر میزان پهنای باند به طرز فزاینده‌ای کاهش می‌یابد. (کاهش نمایی)

ثابت بودن Delay و تغییر پهنای باند:



هنگامی که Delay ثابت است تغییر پهنای باند، تاثیری روی RTT ندارد: هنگامی که طول بسته (۶۴ بیت پینگ) در مقایسه با پهنای باند لینک بسیار کوچک است و لینک تنها توسط یک بسته اشغال شده است، تغییر پهنای باند لینک تاثیری روی RTT ندارد. جاده‌ای را در نظر بگیرید که هیچ خودرویی در آن وجود ندارد و خودروی ما با هر سرعتی می‌تواند در آن حرکت کند. اکنون تعداد باندهای این جاده (پهنای باند) در زمان رفت و برگشت (RTT) ما تاثیری ندارد.



با افزایش پهنای باند، پهنای باند اندازه گرفته شده افزایش می‌یابد: اگر به مقادیر درون جدول نگاه کنیم متوجه می‌شویم که پهنای باند لینک تقریباً با پهنای باند اندازه گیری شده برابر است (اندازه گیری شده اندکی کمتر از واقعی است) این موضوع بدیهی بوده و در توضیح آن می‌توان گفت هر دو مؤلفه در این جا یک چیز (پهنای باند) می‌باشند و بدیهی است که رابطه میان آن‌ها مستقیم است.