# Meng-Chieh Lin Lab3 Lin1055

1.(i)

After three tries, the result below shows that it takes around 3 seconds to transfer the dummy file with 33 MB size from pod1-1 to escher03 when there is only one client.

```
escher03 60 $ ./myfetchfile dummy 128.10.25.201 50122
Completion time (msec): 2945.789
Speed (bps): 93972807.964182
File size (bytes): 34603008
escher03 61 $ ./myfetchfile dummy 128.10.25.201 50122
Completion time (msec): 2944.978
Speed (bps): 93998686.577625
File size (bytes): 34603008
escher03 62 $ ./myfetchfile dummy 128.10.25.201 50122
Completion time (msec): 2945.076
Speed (bps): 93995558.688468
File size (bytes): 34603008
```

By observation, the throughput and completion time are very stable. The speed is always around 94Mbps, and the time is always around 3 second.

1.(ii) (numbers in brackets are results of 1-client)

When the number of clients is 2:

| | Completion time (msec) | Speed (bps) |
|---|---|---|
| escher03 | 2949.686 (2947.017) | 93848655.077184 |
| escher04 | 2944.054 (2947.716) | 94028188.341654 |

Based on the result above, the throughput and completion time are barely affected in 2-client circumstances, compared to 1-client.

When the number of clients is 6:

| | Completion time (msec) | Speed (bps) |
|---|---|---|
| escher03 | 2945.692 (2947.876) | 93975902.436507 |
| escher04 | 3075.333 (2944.910) | 90014337.959499 |
| escher05 | 2999.218 (2942.680) | 92298747.206772 |
| escher06 | 871.318 (500.476) | 317707271.053737 |
| escher07 | 1227.860 (491.340) | 225452465.264770 |
| escher08 | 1037.578 (556.344) | 266798316.849432 |

Based on the result above, the results from escher03 to escher05 are in a group (Group 1) while the other in another group (Group 2). By observation, the influence on Group1 is not obvious while the result of Group 2 is significantly affected. Overall, multi-client reduces the performance of each client.

When the number of clients is 10:

| | Completion time (msec) | Speed (bps) |
|---|---|---|
| escher03 | 2953.240 (2951.299) | 93735715.349921 |
| escher04 | 2980.860 (2949.437) | 92867180.612307 |
| escher05 | 3079.493 (2946.131) | 89892740.136120 |
| escher06 | 1137.623 (548.749) | 243335502.183061 |
| escher07 | 1487.900 (528.341) | 186050175.415014 |
| escher08 | 1404.964 (572.124) | 197032846.393217 |
| escher09 | 1364.849 (496.795) | 202823949.022932 |
| escher10 | 1126.894 (471.656) | 245652265.430466 |
| escher11 | 3027.633 (2944.135) | 91432503.212906 |
| escher12 | 720.332 (496.653) | 384300650.255715 |

By observation, all results are apparently affected by multi-client. By the way, the group, including escher03,04,05,11, are more stable than the rest.

1.(iii)

| Block size | Completion time (msec) | Speed (bps) |
|---|---|---|
| 500 | 2948.567 | 93884271.240911 |
| 1000 | 2951.994 | 93775280.031057 |
| 5000 | 2948.184 | 93896467.791698 |
| 10000 | 2946.468 | 93951152.362761 |

With the values given, it seems no optimal block size value.

2.

| Initial RTT: 5000um | Completion time (msec) | Speed (bps) |
|---|---|---|
| No estimation 1 | 6884.404 | 6623667.059632 |
| No estimation 2 | 7314.984 | 6233779.868828 |
| No estimation 3 | 7103.065 | 6419763.862502 |
| Adaptive estimation 1 | 8633.706 | 5281625.295093 |
| Adaptive estimation 2 | 8363.768 | 5452088.101918 |
| Adaptive estimation 3 | 8734.559 | 5220641.362661 |

With three times experiment, the result shows that employing adaptive estimation can evidently improve the throughput and reduce the completion time.

3.

```
ee 4d 4d c2 a2 be 2e 5b  78 7c 1d e4 08 00 45 00   ·MM···.[ x|····E·
00 3b 8b ac 40 00 40 11  2b b2 c0 a8 01 02 c0 a8   ·;··@·@· +·······
01 01 ec 25 d7 ac 00 27  83 8c 30 33 33 33 33 33   ···%···' ··033333
33 33 33 33 33 33 33 33  33 33 33 33 33 33 33 33   33333333 33333333
33 33 33 33 33 33 33 33  33                        33333333 3
```

Destination address: ee:4d:4d:c2:a2:be

```
veth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.1  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::ec4d:4dff:fec2:a2be  prefixlen 64  scopeid 0x20<link>
        ether ee:4d:4d:c2:a2:be  txqueuelen 1000  (Ethernet)
```

Source address: 2e:5b:78:7c:1d:e4

```
veth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.2  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::2c5b:78ff:fe7c:1de4  prefixlen 64  scopeid 0x20<link>
        ether 2e:5b:78:7c:1d:e4  txqueuelen 1000  (Ethernet)
```

Type field: 08 00

```
ee 4d 4d c2 a2 be 2e 5b  78 7c 1d e4 08 00 45 00   ·MM···.[ x|····E·
00 3b 8b ac 40 00 40 11  2b b2 c0 a8 01 02 c0 a8   ·;··@·@· +·······
01 01 ec 25 d7 ac 00 27  83 8c 30 33 33 33 33 33   ···%···' ··033333
33 33 33 33 33 33 33 33  33 33 33 33 33 33 33 33   33333333 33333333
33 33 33 33 33 33 33 33  33                        33333333 3
```

Sequence number: 30 (hex)

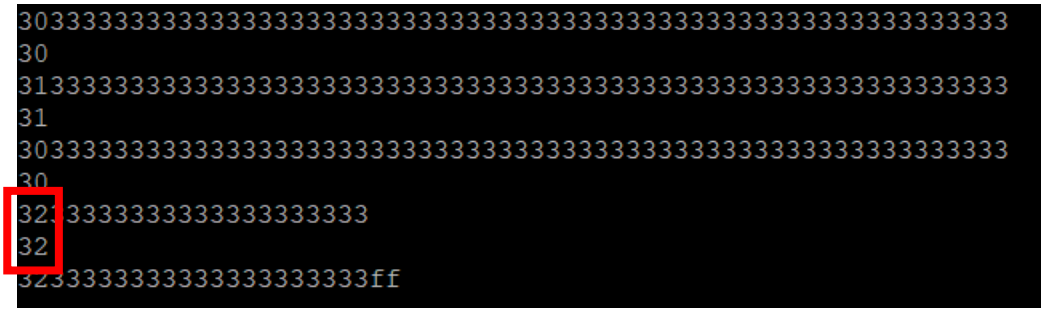
```
ee 4d 4d c2 a2 be 2e 5b  78 7c 1d e4 08 00 45 00   ·MM···.[ x|····E·
00 3b 8b ac 40 00 40 11  2b b2 c0 a8 01 02 c0 a8   ·;··@·@· +·······
01 01 ec 25 d7 ac 00 27  83 8c 30 33 33 33 33 33   ···%···' ··033333
33 33 33 33 33 33 33 33  33 33 33 33 33 33 33 33   33333333 33333333
33 33 33 33 33 33 33 33  33                        33333333 3
```

Rest of payload:

33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33

In ACK packets, payloads only contain one byte (sequence number), meaning that padding was not applied to Ethernet's payload.

```
2e 5b 78 7c 1d e4 ee 4d  4d c2 a2 be 08 00 45 00   .[x|···M M·····E·
00 1d b1 0b 40 00 40 11  06 71 c0 a8 01 01 c0 a8   ····@·@· ·q······
01 02 d7 ac ec 25 00 09  83 6e 30                  ·····%·· ·n0
```




For the last two stop-and-wait application packets, their sequence numbers are 32.