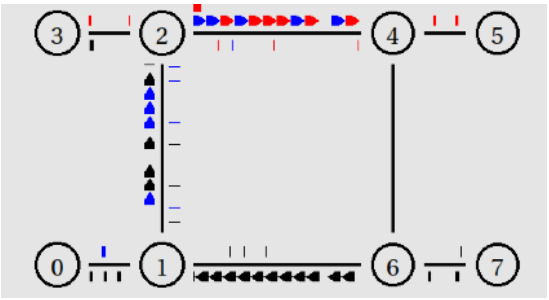
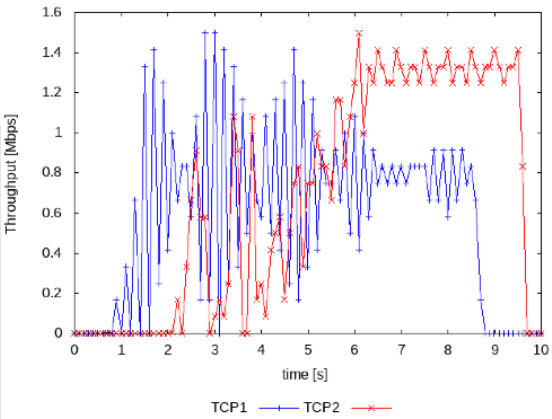
COMP9331 lab6 report

written by Heng-Chuan Lin z5219960

EX1Q1: Why the throughput achieved by flow tcp2 is higher than tcp1 between time span 6 sec to 8 sec?

Ans:





In the time period of 6 – 8 seconds, there are different flows competing each other in the link n1-n2 and n2-n4. TCP1 (n0-n5) compete the flow 4 (n7-n3) in the link n1-n2 and TCP2(n3-n5) compete the flow with TCP1 in link n2-n4. Hence, TCP1 suffered more competition throughout the entire path to the destination n5. Besides, TCP2 actually got less RTT than TCP1 (i.e. TCP1 should go through one more link to the destination). These reasons lead to the result that we've observed from the plot above.

EX1Q2: Why the throughput for flow tcp1 is fluctuating between time span 0.5 sec to 2 sec?

Ans:

At that time period, TCP 1 was in the slow-start stage and probing the maximum windows sizes it could get. Hence, the throughput would fluctuate because stages shifting among slow-start, fast-recovery and congestion avoidance.

EX1Q3: Why is the maximum throughput achieved by any one flow capped at around 1.5Mbps?

Ans:

FTP/TCP Source n0 -> TCP Sink n5 : start time: 0.5 sec End time: 8.5 sec

FTP/TCP Source n3 -> TCP Sink n5 : start time: 2.0 sec End time: 9.5 sec

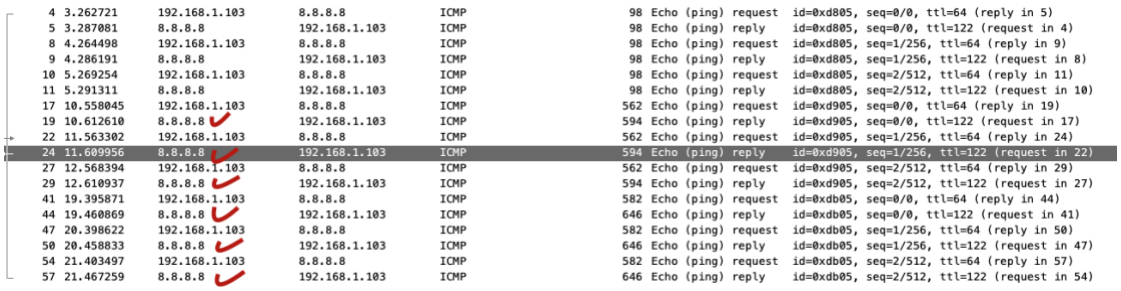
FTP/TCP Source n7 -> TCP Sink n0 : start time: 3.0 sec End time: 9.5 sec

FTP/TCP Source n7 -> TCP Sink n3 : start time: 4.0 sec End time: 7.0 sec

For TCP1, the maximum throughput that TCP1 could achieve during 0.5-2s is still around 1.5Mbps. It is noted that only TCP1 flow occupied the links throughout the entire path at that moment. However, it turned out that TCP1 couldn't use the entire capacity of link because TCP1 is on slow start stage. TCP1 still took times to probe the available bandwidth (but still not reach the maximum bandwidth 2.5 Mbps because running out of time) before TCP2 started competing with TCP1 in the link. The throughputs of TCP1 and TCP2 afterwards just the result of competition among different flows.

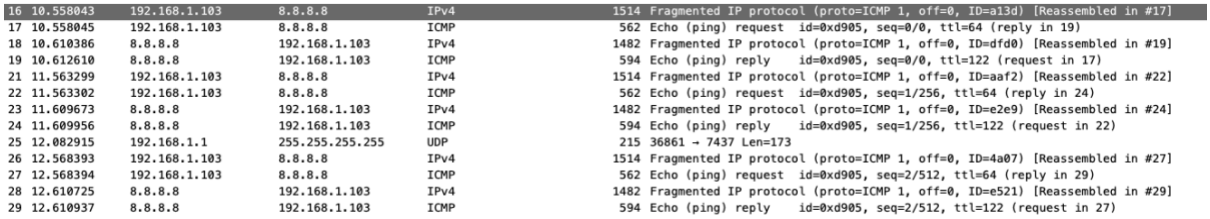
EX2Q1: Which data size has caused fragmentation and why? Which host/router has fragmented the original datagram? How many fragments have been created when data size is specified as 2000?

Ans:



4	3.262721	192.168.1.103	8.8.8.8	ICMP	98	Echo (ping) request id=0xd805, seq=0/0, ttl=64 (reply in 5)
5	3.287081	8.8.8.8	192.168.1.103	ICMP	98	Echo (ping) reply id=0xd805, seq=0/0, ttl=122 (request in 4)
8	4.264498	192.168.1.103	8.8.8.8	ICMP	98	Echo (ping) request id=0xd805, seq=1/256, ttl=64 (reply in 9)
9	4.286191	8.8.8.8	192.168.1.103	ICMP	98	Echo (ping) reply id=0xd805, seq=1/256, ttl=122 (request in 8)
10	5.269254	192.168.1.103	8.8.8.8	ICMP	98	Echo (ping) request id=0xd805, seq=2/512, ttl=64 (reply in 11)
11	5.291311	8.8.8.8	192.168.1.103	ICMP	98	Echo (ping) reply id=0xd805, seq=2/512, ttl=122 (request in 10)
17	10.558045	192.168.1.103	8.8.8.8	ICMP	562	Echo (ping) request id=0xd905, seq=0/0, ttl=64 (reply in 19)
19	10.612610	8.8.8.8	192.168.1.103	ICMP	594	Echo (ping) reply id=0xd905, seq=0/0, ttl=122 (request in 17)
22	11.563302	192.168.1.103	8.8.8.8	ICMP	562	Echo (ping) request id=0xd905, seq=1/256, ttl=64 (reply in 24)
24	11.609956	8.8.8.8	192.168.1.103	ICMP	594	Echo (ping) reply id=0xd905, seq=1/256, ttl=122 (request in 22)
27	12.568394	192.168.1.103	8.8.8.8	ICMP	562	Echo (ping) request id=0xd905, seq=2/512, ttl=64 (reply in 29)
29	12.610937	8.8.8.8	192.168.1.103	ICMP	594	Echo (ping) reply id=0xd905, seq=2/512, ttl=122 (request in 27)
41	19.395871	192.168.1.103	8.8.8.8	ICMP	582	Echo (ping) request id=0xdb05, seq=0/0, ttl=64 (reply in 44)
44	19.460869	8.8.8.8	192.168.1.103	ICMP	646	Echo (ping) reply id=0xdb05, seq=0/0, ttl=122 (request in 41)
47	20.398622	192.168.1.103	8.8.8.8	ICMP	582	Echo (ping) request id=0xdb05, seq=1/256, ttl=64 (reply in 50)
50	20.458833	8.8.8.8	192.168.1.103	ICMP	646	Echo (ping) reply id=0xdb05, seq=1/256, ttl=122 (request in 47)
54	21.403497	192.168.1.103	8.8.8.8	ICMP	582	Echo (ping) request id=0xdb05, seq=2/512, ttl=64 (reply in 57)
57	21.467259	8.8.8.8	192.168.1.103	ICMP	646	Echo (ping) reply id=0xdb05, seq=2/512, ttl=122 (request in 54)

The size of ICMP which larger than 1500 bytes would be fragmented according to the MTU setup of ethernet. Hence, the second and last ping commands with 2000 and 3500 bytes would be delivered in fragments. In this case, the sender would split the entire data to fragments and the receiver would finally assemble these fragments.



16	10.558043	192.168.1.103	8.8.8.8	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a13d) [Reassembled in #17]
17	10.558045	192.168.1.103	8.8.8.8	ICMP	562	Echo (ping) request id=0xd905, seq=0/0, ttl=64 (reply in 19)
18	10.610386	8.8.8.8	192.168.1.103	IPv4	1482	Fragmented IP protocol (proto=ICMP 1, off=0, ID=dfd0) [Reassembled in #19]
19	10.612610	8.8.8.8	192.168.1.103	ICMP	594	Echo (ping) reply id=0xd905, seq=0/0, ttl=122 (request in 17)
21	11.563299	192.168.1.103	8.8.8.8	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=aaf2) [Reassembled in #22]
22	11.563302	192.168.1.103	8.8.8.8	ICMP	562	Echo (ping) request id=0xd905, seq=1/256, ttl=64 (reply in 24)
23	11.609673	8.8.8.8	192.168.1.103	IPv4	1482	Fragmented IP protocol (proto=ICMP 1, off=0, ID=e2e9) [Reassembled in #24]
24	11.609956	8.8.8.8	192.168.1.103	ICMP	594	Echo (ping) reply id=0xd905, seq=1/256, ttl=122 (request in 22)
25	12.082915	192.168.1.1	255.255.255.255	UDP	215	36861 → 7437 Len=173
26	12.568393	192.168.1.103	8.8.8.8	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=4a07) [Reassembled in #27]
27	12.568394	192.168.1.103	8.8.8.8	ICMP	562	Echo (ping) request id=0xd905, seq=2/512, ttl=64 (reply in 29)
28	12.610725	8.8.8.8	192.168.1.103	IPv4	1482	Fragmented IP protocol (proto=ICMP 1, off=0, ID=e521) [Reassembled in #29]
29	12.610937	8.8.8.8	192.168.1.103	ICMP	594	Echo (ping) reply id=0xd905, seq=2/512, ttl=122 (request in 27)

For 2000 bytes ICMP, there are # (16,17), # (21,22), # (26,27) have been created.

#(IPv4, IPv4 with ICMP)

EX2Q2: Did the reply from the destination 8.8.8.8. for 3500-byte data size also get fragmented? Why and why not?

Ans:

Yes, it did. The reason is that these response packets still be delivered through different routers. Each router may use different network protocol. Even if your data got chopped into fragments in a router on your path to destination, it was still possible that your fragments would be chopped again into smaller fragments due to different network protocol in other routers.

EX2Q3: Give the ID, length, flag and offset values for all the fragments of the first packet sent by 192.168.1.103 with data size of 3500 bytes?

First packet # 39 IPv4 Offset = 0	Total Length: 1500 Identification: 0x7a7b (31355) Flags: 0x2000, More fragments 0... .... = Reserved bit: Not set .0.. .... = Don't fragment: Not set ..1. .... = More fragments: Set ...0 0000 0000 0000 = Fragment offset: 0

Second packet # 40 IPv4 Offset = 185 * 8 = 1480	Total Length: 1500 Identification: 0x7a7b (31355) Flags: 0x20b9, More fragments 0... .... = Reserved bit: Not set .0.. .... = Don't fragment: Not set ..1. .... = More fragments: Set ...0 0000 1011 1001 = Fragment offset: 185
Third packet # 41 IPv4 w/ ICMP Offset = 370 * 8 = 2960	Total Length: 568 Identification: 0x7a7b (31355) Flags: 0x0172 0... .... = Reserved bit: Not set .0.. .... = Don't fragment: Not set ..0. .... = More fragments: Not set ...0 0001 0111 0010 = Fragment offset: 370

EX2Q4: Has fragmentation of fragments occurred when data of size 3500 bytes has been used? Why and why not?

Ans:

For sending ICMP request to 8.8.8.8, we didn't observe any fragmentation of fragments occurred when data of size 3500 bytes has been used. Because we only know the packet from our source in the trace file. We wouldn't know if there's any fragmentation occurred through the routers on the entire path to 8.8.8.8.

For receiving ICMP response from 8.8.8.8, there's three fragments received. In this case, there's no fragmentation of fragments occurred.

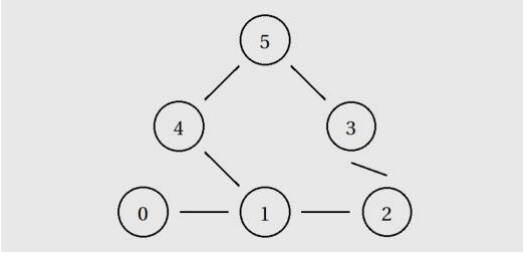
EX2Q5: What will happen if for our example one fragment of the original datagram from 192.168.1.103 is lost?

Ans:

If receiver noticed that there's some fragments lost/corrupted, it would drop the entire data. Finally, the sender would retransmit these packets to destination again.

EX3Q1: Which nodes communicate with which other nodes? Which route do the packets follow? Does it change over time?

Ans:



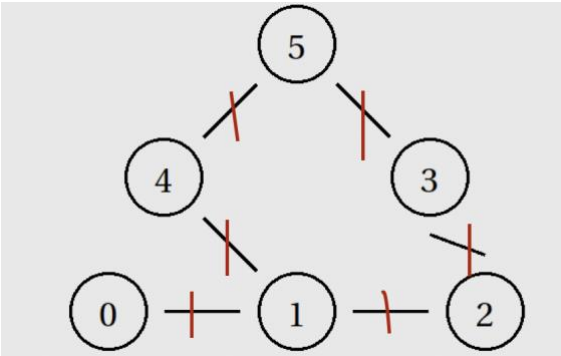
According to the script, n0 was transmitting to n5 and n2 was transmitting to n5 as well. The packets followed 2 paths, n0-n1-n4-n5 and n2-n3-n5 and the paths didn't change over time.

EX3Q2: What happens at time 1.0 and at time 1.2? Does the route between the communicating nodes change as a result of that?

Ans:

At time 1.0, the direction of n1-n4 changed to down. The packets from source cannot be delivered from n0 to n5.
At time 1.2, the direction of n1-n4 finally changed to up. The packets could be delivered through this node again from n0 to n5.

EX3Q3: Did you observe any additional traffic as compared to Step 3 above? How does the network react to the changes that take place at time 1.0 and time 1.2 now?



Ans:

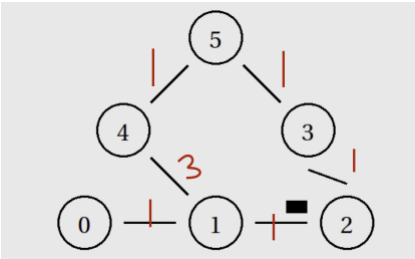
Yes, using DV allows routers establishing connection-oriented connection among these nodes and sharing their forwarding information to the different interfaces which connected to the interface of another node.

At time 1.0, the direction of n1-n4 changed to down. The packets from source cannot be delivered from n0 to n5 through this node. This would force packets being delivered through n0-n1-n2-n3-n5. At this time, DV routing noticed that routing path changed and update this information to each router.

At time 1.2, the direction of n1-n4 finally changed to up. DV would notice the path changed and then update to each router. Finally, it would transmit through the original path n0-n1-n4-n5 which with the lowest cost (3).

EX3Q4: How does this change affect the routing? Explain why.

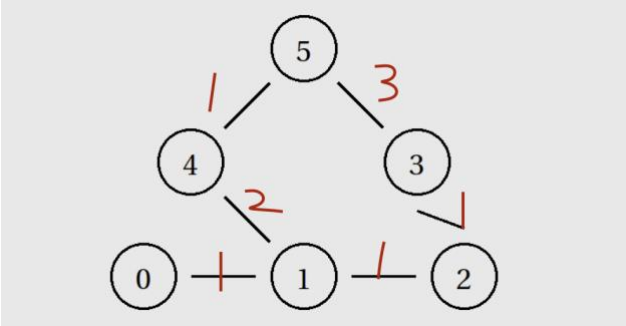
Ans:



The DV routing protocol is based on the least cost path. By adding cost of n1-n4 to 3, we observed that it travelled through n0-n1-n2-n3-n5(cost 1+1+1+1 = 4) with lower cost.

EX3Q5: Describe what happens and deduce the effect of the line you just uncommented.

Ans:



Cost of paths

N0 to N5	N0-N1-N4-N5 cost:3	N0-N1-N2-N3-N5 cost:6
N2 to N5	N2-N-N4-N5 cost:4	N2-N3-N5 cost:4

For N0 to N5, DV routing protocol would choose the lowest cost path which is N0-N1-N4-N5.

For N2 to N5, DV routing protocol would choose the lowest cost path. However, there are two paths with the same costs which are both lowest. Since the multipath model is enable, N2 could spend packet through N2-N-N4-N5 and N2-N3-N5 to N5.