I Stop-n-Wait-n-Answer

1. **[10 points]:**

An application needs to send 100KB of data using a stop-and-wait reliable protocol. The protocol splits the data into segments that have a 1KB application data payload. Each segment fits in a single IP packet. The RTT is 50ms, there is no packetization delay, and no queueing delay. The protocol uses a fixed retransmission timeout of 200ms and has no retransmission limit.

(i) How long will the transmission take, in seconds, if the network does not drop, duplicate or corrupt any packets? You may assume the connection is established when you start your measurement, so there is no additional latency from connection setup and consists only of data transmissions. Your answer must be accurate to two decimal points (10ms).

50ms for one packet

⇒ 50ms × 100 = 5000 ms

⇒ 5 seconds

5 seconds

(ii) Let us now suppose that the network drops each segment with a probability of 10%, independently from segment to segment. The network drops both data and acknowledgements. What is the *expected* duration of the transmission? Show your calculations. Your answer must be accurate to two decimal points (10ms).

(9.6s if retransmits can be dropped too). 8

8.8 seconds

100 packets ⇒ ~~10 packets dropped~~

~~100 acks ⇒ 10 acks dropped~~

On average, 10 packets will be dropped

⇒ 90 packets won't be dropped but 10% acks will

⇒ we won't get acks for 9 of these 90 packets.

$$\Rightarrow 50ms \times 81 + \underbrace{200ms \times 19}_{19\ \text{timeouts}} + \underbrace{50ms \times 19}_{\text{retransmit}}$$

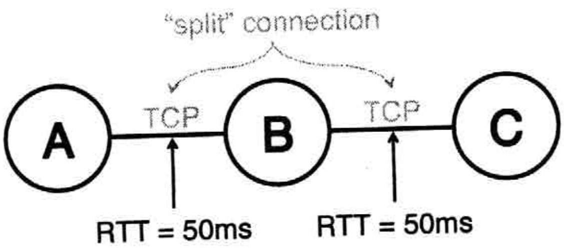
$= 5s + 3800ms$

$= 8.8$ seconds

[I assumed retransmitted packets won't be dropped. If they are also dropped, look on left]

II Without TCP The Internet Ain't Nuthin'

There are devices and services in the Internet, such as proxy servers, that "split" TCP connections. Suppose a host A wants to open a connection to a host C. A device somewhere along the path, B, can terminate A's connection at itself, and open a connection to C. So in this case there are now two TCP connections, A to B and B to C. A thinks it's sending data to C, but B is processing the TCP segments itself and sending acknowledgments back to A, spoofed from B's IP address. Simultaneously, B opens a TCP connection to C, pretending to be A.



Suppose you have the network above, where the RTT from A to B is 50ms, the RTT from B to C is 50ms, and there is no packetization, queueing, or processing delay, such that the RTT from A to C is 100ms. The maximum segment size is 1400 bytes. A is sending an infinite stream of bytes, such that every segment is the maximum segment size. Recall that a TCP flow's throughput can be approximated as

$$\text{MSS} \cdot \sqrt{\frac{3}{2}} \cdot \frac{1}{\text{RTT}\sqrt{p}}$$

where p is the packet drop rate.

Please write out answers numerically and do not leave radicals or variables in your solutions. You may leave fractions. If you do not have a calculator, you may approximate with the following values:

$$MSS \cdot \sqrt{\tfrac{3}{2}} = 13,717 \text{ bits}$$

$$\sqrt{0.1} = 0.32$$

$$\sqrt{0.19} = 0.44$$

$$\sqrt{0.2} = 0.45$$

$$\sqrt{0.21} = 0.46$$

$MSS \sqrt{\frac{3}{2}} \cdot \frac{1}{RTT\sqrt{p}}$

2. [5 points]:

Suppose that B does not split the TCP connection, such that packets flow directly from A to C, through B. The route between A and B drops 10% of data segments and does not drop acknowledgments, while the route between B and C does not drop any packet. What will the TCP throughput from A to C be?

857.3

~~0.43 kbps~~ ~~43.49 kbps~~ $\boxed{428.7 \text{ kbps}}$

~~42.87 kbps~~

~~0.86 kbps~~ ____________ kbps

$16 \overline{)13717}$, 128, 91, 80, 117, 112, 50

$\cancel{1400}\,128 \times 8 \text{ bits} \times \sqrt{\frac{3}{2}} \cdot \frac{1}{100 \cdot \sqrt{0.1}} + \min(\infty, \text{prev throughput})$

$= \cancel{2} \times 13717 \times \frac{1}{\cancel{50 \times 0.32}\ \cancel{16}\,32} = \frac{13717}{16} = 857.3 \text{ bps}$

$= 0.\frac{86}{2} \text{ kbps}$

$= 0.43 \text{ kbps}$

$13717 \times \frac{1}{0.15 \times 0.32} = \frac{13717}{32} \times 10^{3}$

$= 43.49 \times 10^{3} \text{ bits/s}$

$= 43.4$

$32 \overline{)13717}$ = 43.49; 128, 91, 96, 157, 128, 290

3. [5 points]:

Suppose that B does split the connection, such that packets flow from A to B, terminate at B, then are forwarded in separate flow from B to C. The route between A and B drops 10% of data segements and drops no acknowledgments, while the route between B and C does not drop any packet. What will the throughput from A to C be?

$32 \overline{)13717}$ = 42.87; 128, 91, 64, 277, 256, 210

~~0.86~~ ____________ kbps

2 × value of 2

= 2 × 42.88

~~= 86.98 kbps~~

~~= 85.73 kbps~~

$= \boxed{857.3 \text{ kbps}}$

4. **[5 points]**:

Suppose that B does split the connection, such that packets flow from A to B, terminate at B, then are forwarded in separate flow from B to C. The route between A and B drops 10% of packets, and the route between B and C also drops 10% of packets. What will the throughput from A to C be?

∵ initially, we had 100 packets, at B, we have 90 packets, and at C we have 81 packets

⇒ 19% drop rate

311.7 kbps

$$M.SS \sqrt{\tfrac{3}{2}} \times \frac{1}{RTT} \cdot \frac{1}{\sqrt{P}}$$

$$= 13717 \times \frac{1}{0.1} \times \frac{1}{\sqrt{0.19}} = \frac{13717}{0.1 \times 0.44} = \frac{13717}{1 \times 44} \times \cancel{10^3 \text{ bits}} \text{ kbps}$$

$$44 \overline{)13717} = 311.7$$

5. **[5 points]**:

Finally, suppose that B does not split the connection, such that packets flow from A to C passing through but not terminating at B. The route between A and B drops 10% of data segments, and the route between B and C also drops 10% of data segments. What will the throughput from A to C be?

311.7 kbps

III FTP Is For Old People

The File Transfer Protocol (FTP) is an older application protocol for transferring files. Like HTTP, it uses ASCII commands. Unlike HTTP, it uses a separate control and data channel. The protocol specification greatly predates STUN and other NAT probing/traversal approaches.

In normal operation, when a client requests a file (e.g., `RETRIEVE .cshrc`), the *FTP server* opens a TCP connection to the client to transfer the data. The client can specify the IP address and port to open a connection to with the `PORT` command. A client can alternatively tell the server to listen on a connection with the `PASSIVE` command (the server chooses the IP/port), such that the client can be the active opener.

Your client is behind a port-restricted NAT with no static mappings. The FTP server is outside the NAT and is not behind a NAT.

6. [**5 points**]:

Can your client use the `PORT` command to set up a successful file transfer?

Circle the best answer.

A ~~Yes~~ NO

B No

Briefly explain why:

~~The client sends a TCP message through the NAT. The NAT updates the IP and PORT in the TCP message. Now, when the server replies, it sends the message to this IP/port and the NAT will forward it to the client.~~

Suppose client tells server to open a connection to port 100, ~~and sp~~. When the server tries to open a connection to port 100, the NAT won't recognize this port and throw it away.

7. [5 points]:

Can your client use the `PASSIVE` command to set up a successful file transfer?

Circle the best answer.

~~A Yes~~

B No

NO

Briefly explain why:

~~YES. The client asks the server~~

Suppose the client asks server to have a PASSIVE connection. The server says "I'll talk on port 100"

~~NAT won't accept.~~

NAT won't accept.

IV Put It All Together, Now

8. [15 points]:

You type the following URL into your web browser:

`http://gradadmissions.stanford.edu/inquiry/onlineinq.htm`

Assuming that

- your DNS resolver is 171.64.7.77,
- neither your host nor your DNS resolver have any cached DNS entries,
- DNS never needs to fail over to TCP, and
- the HTML response returns 200 OK with a web page,
- the HTML request and response each fit in a single segment, and
- the web page requires loading no additional resources,

write down the series of packet exchanges that will occur for your host to receive the web page. Include packets sent by your DNS server as well as control packets for TCP connection setup and teardown. You need not include any ARP packets, and you do not need to write down message formats. Simple descriptions such as "X sends a UDP segment to the HTML server on the HTTP port" are sufficient. In the case of the HTTP request, clearly state the path of the file requested in the GET.

- Client makes recursive UDP request to DNS resolver
- DNS resolver makes non-recursive UDP request to root server, TLD (.edu), stanford.edu and finally gradadmissions.stanford.edu
- DNS resolver sends UDP packet to client with IP address of gradadmissions.stanford.edu to client on port 83 (-1)
- Client ~~[illegible]~~ sends SYN TCP message to HTML server on HTTP port on port 80 (-1) (TCP) client doesn't necessa[rily] Bind it
- Server responds with SYN/ACK to client on port 80
- Client sends ACK to ~~server~~ HTML server on port 80 and also ~~request~~ makes a GET request for /inquiry/onlineinq.htm (TCP) end o[f] 80
- Server responds with TCP 200 OK message with a web page
- Client sends FIN TCP message to HTML server on port 80
- ~~Client~~ Server sends FIN/ACK TCP to client on port 80
- Client sends ACK to server and goes into TIMEWAIT, keeping socket around for 2MSL before closing.

V A Rose By Any Other Name..

The command "`dig stanford.edu A @a.edu-servers.net`" asks the machine `a.edu-servers.net` (name server for the `.edu` zone) for the IPv4 address (DNS type `A` record) of domain name `stanford.edu`. The command's output might look something like this:

```
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31366
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 4, ADDITIONAL: 4
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;stanford.edu.                    IN      A

;; AUTHORITY SECTION:
stanford.edu.            172800   IN      NS      avallone.stanford.edu.
stanford.edu.            172800   IN      NS      atalante.stanford.edu.
stanford.edu.            172800   IN      NS      argus.stanford.edu.
stanford.edu.            172800   IN      NS      aerathea.stanford.edu.

;; ADDITIONAL SECTION:
avallone.stanford.edu.   172800   IN      A       171.64.7.88
atalante.stanford.edu.   172800   IN      A       171.64.7.61
argus.stanford.edu.      172800   IN      A       171.64.7.115
aerathea.stanford.edu.   172800   IN      A       152.3.104.250
```

9. **[5 points]:**

Assuming `stanford.edu` has an IP address (i.e., DNS resource record of type `A`), why is the answer section empty in this reply?

The a.edu-servers.net can only do non-recursive lookups whereas to full resolve stanford.edu, it will need to do recursive lookups which TLDs don't do.

But .edu could have a cached reply.

-1

10. **[5 points]:**

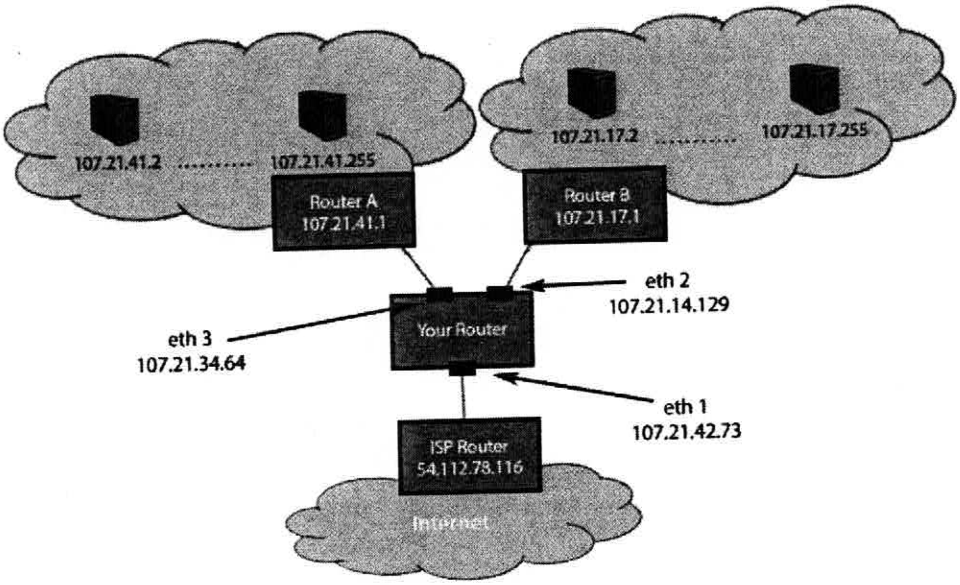
What is the purpose of the records in the additional section—How would DNS break if replies did not contain an additional section or any of the records usually placed here?

The additional section contains the A records of the NS (name server) records associated with the domain name being requested.

If we did not have an additional section, we won't get the IP address of the name server ~~rec~~ records and may not be able to resolve a domain name any further.

11. [10 points]:

A web site hosted on a single server becomes extremely popular. The administrators decide to replace the server with two sets of 200 servers (i.e. 400 servers in total), each set of servers connected to the Internet via a different router (for fault tolerance). The figure shows the topology. The two routers (Router A and Router B) connect to the Internet via a router that you manage ("Your Router"). The administrator of Router A assigns 200 host IP addresses in the range `107.21.41.2`–`107.21.41.255`, and the administrator of Router B assigns 200 host IP addresses in the range `107.21.17.2`–`107.21.17.255`. You decide to manually insert routing table entries into "Your Router" to correctly route packets between the servers and the rest of the Internet.



Write down the routing table entries for "Your Router" to correctly route packets between the Internet and the four hundred servers. You need not include routes to "Your Router"'s interfaces. Use as few table entries as possible.

107.21.41.0

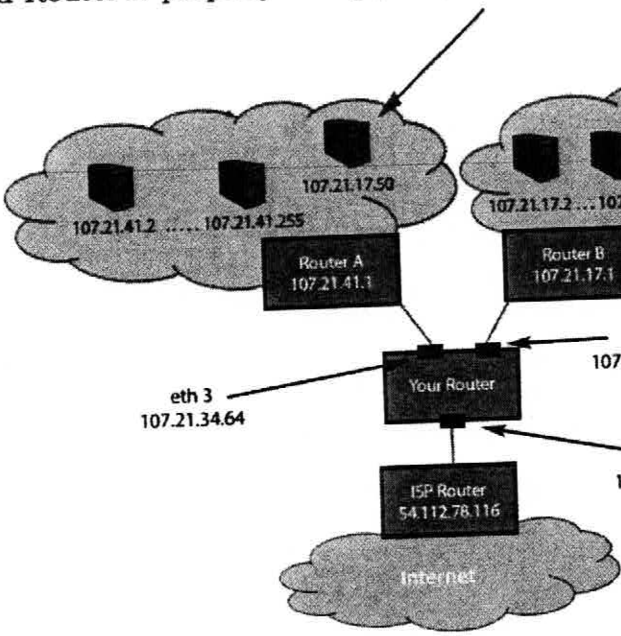
destination prefix (e.g. 10.0.0.0)	net mask (e.g. 255.0.0.0)	next hop (e.g. 10.0.0.1)	interface (e.g. eth0)
107.21.41.0	255.255.255.0	107.21.41.1	eth 3
107.21.17.0	255.255.255.0	107.21.17.1	eth 2
0.0.0.0	0.0.0.0	54.112.78.116	eth 1
107.21.41.0	255.255.255.255	"	"
107.21.17.0	255.255.255.255	"	"

These 2 would otherwise go to router A or router B but belong to neither of them.

Nice catch!

12. [**5 points**]:

The administrators decide to move the server with IP address `107.21.17.50` from the network behind Router B to the network behind Router A. Assuming that they change no server IP addresses (the moved server keeps the *same* IP address), how do you need to modify the routing tables of "Your Router" to correctly route packets between all servers and the Internet? You may assume the administrators will update Router A and Router B properly to support your change. The new topology looks like this:



Write down any new or modified entries you need:

destination prefix	net mask	next hop	interface
107.21.17.50	255.255.255.255	107.21.41.1	eth 3

Explain your answer:

Because of $\boxed{\text{longest prefix match}}$ 107.21.17.50 will ~~be~~ have longest match with above entry and any packets to it will be routed through Router A.

VI Playback Buffers and Queues

13. [**10 points**]:

Two endpoints in a voice-over-IP session are connected by a path of 4 routers. All links are running at 1Mb/s and the hosts are separated by 3000km. All packets are of size 1500Bytes. Assume the bit propagation speed is 2×10^8 m/s. Note that 1KB of data is 1024Bytes, but 1Mb/s is 10^6 bits/s.

(i) What is the minimum round trip time (RTT), assuming there is no queueing delay and assuming processing time at each host is negligible?

$\Rightarrow$ e2e delay = 63ms

$\Rightarrow$ RTT = 2×63 = 126

126 -1 milliseconds

prop delay = $\frac{3000 \times 10^3 \text{ m}}{2 \times 10^8 \text{ ms}^{-1}} = \frac{3}{2} \times 10^{-2}$ s $= \frac{1.5}{100} = 0.015$ s $= 15$ms

pack. delay = 4 × $\frac{1500 \times 8 \text{ bits}}{10^6 \text{ bits/s}}$ = $\frac{48000}{10^6}$ s $= 48 \times 10^{-3}$ s $= 48$ms

5

(ii) For this part only let us assume that one router on the path has a steady queue occupancy of 5 packets. What is the end-to-end delay (one way, not round trip) in this case?

123 milliseconds

63ms + $\frac{5 \times 1500 \times 8}{10^6}$ (size of queue)

$= \frac{6 \times 10^4}{10^6} = 6 \times 10^{-2}$ s $= 60 \times 10^{-3}$ s $= 60$ms

$= 63 + 60 = 123$

(iii) Now let us assume the maximum queue occupancy for *every* router queues is 5 packets. What is the maximum end-to-end delay?

1500
~~20~~ 160

13
16
240000

303 milliseconds.

size in bits

~~2400~~
$24 \times 10^4 \times 10^6$
$= 24 \times 10^{-2}$
$= 240$ ms

$$1 + \underbrace{4}_{\text{no. of routers}} \times \underbrace{5}_{\text{no. of packets in buffer}} \times \frac{\overbrace{1500 \times 8 \text{ bits}}^{\text{size in bits}}}{10^6 \text{ bits/s}}$$

$= 63 + 240 = 303$ ms

(iv) For part (iii), how long should the playback buffer be at the destination voice-over-IP client be if each packet arrives successfully, without being dropped? Express your answer in bytes.

30000 = 20 packets bytes.

min e2e delay = all queues are empty
= 63 ms

max e2e delay = 303 ms
⇒ buffer size = 240 ms
= 240×10^{-3} s

⇒ in ~~bytes~~ bits = $\frac{10^6 \text{ bits}}{s} \times 240 \times 10^{-3} \cancel{s}$

$= 240 \times 10^3$ bits
$= 30 \times 10^3$ bytes
=

(v) For part (iii), how long should the playback buffer be at the destination voice-over-IP client if each packet either arrives successfully when first transmitted, or is dropped the first time and then arrives successfully on the second attempt? Assume the sender retransmits a packet only after it finds out that the packet was dropped, which is after twice the maximum end to end delay. Express your answer in bits.

843×10^3 bits.

min e2e = 63ms

max e2e = ~~634~~ $63 + \underbrace{303 \times 2}_{\text{max e2e but dropped}} + 240$

= max e2e × 3

= 909 ms

⇒ difference = 909 − 63

= 846 ms

⇒ size = 846ms × $\frac{10^6 \text{ bits}}{s}$

= 843×10^3 bits

size = $\sigma\left(1 - \frac{\rho}{R}\right)$

$\frac{10}{500} - \frac{10}{600}$

$\frac{\sigma}{e} - \frac{\sigma}{R} =$

14. [10 points]:

Recall that a (σ, ρ) leaky-bucket traffic regulator ensures that the number of bytes departing the regulator in any interval of time (t_1, t_2) is bounded by $B(t_1, t_2) \le \sigma + \rho(t_2 - t_1)$, where σ and ρ are non-negative constants.

Now, consider a queue which drains at rate R. The queue has an infinite buffer size. Answer the following questions.

(i) Which of the following is correct?
Circle all that apply. (There may be multiple answers.)

A If the source feeding the queue is leaky-bucket constrained, then the delay of a bit through the queue is always less than 2σ, regardless of the source's average sending rate, ρ.

B The delay of any bit through the queue is finite only when $\sigma < R$.

C If several leaky-bucket constrained sources (σ_i, ρ_i) feed the queue, and when combined $\sum_i \sigma_i = \sigma$ and $\sum_i \rho_i = \rho < R$, then the delay through the queue will be bounded and finite. Assume σ is finite.

D If several leaky-bucket constrained sources (σ_i, ρ_i) feed the queue, and $\sum_i \sigma_i = 10$, $\sum_i \rho_i = 500\text{b/s}$, and $R = 600\text{b/s}$, then no bit will ever experience a delay through the queue of more than 100ms.

~~B is false. If $\sigma = R$, the delay is still finite. (a queue~~

B is FALSE because dimensions don't match.

D $\Rightarrow$ max bytes departing $= \sigma + \rho t = 10 + 500t$

departure rate $= 600 \Rightarrow 600t = 10 + 500t$

$\Rightarrow t = 100\text{ms}$

max delay $= \frac{10}{600} = 16.7$

-1

A doesn't make sense since delay is in seconds whereas 2σ is in size, so A is FALSE.

C is trivially true.

insufficient explanation -1

$\boxed{8}$