CS3251 HW1

Wenqi He

September 3, 2018

P1

Assume that a reliable connection will be established between client and server, so the user only needs to authenticate at the beginning of each session. Also data should be encrypted before transmission and no packet loss should be allowed.

Login

ATM:

```
LOGIN [card number] [password]
```

Bank:
[Verifies user credentials and starts session timer]

```
LOGIN [card number] OK
```

[If authentication failed]

```
LOGIN [card number] ERROR
  Reason: Authentication failed
```

ATM: *[Displays error on screen]*

Query Account Balance

ATM:

```
QUERY [card number]
```

Bank:
[Looks up account balance in the database]

```
QUERY [card number] OK
  Balance: 3000.00
```

ATM: *[Displays account balance on screen]*

Withdraw From Account

ATM:

```
WITHDRAW [card number]
  Amount: 40.50
```

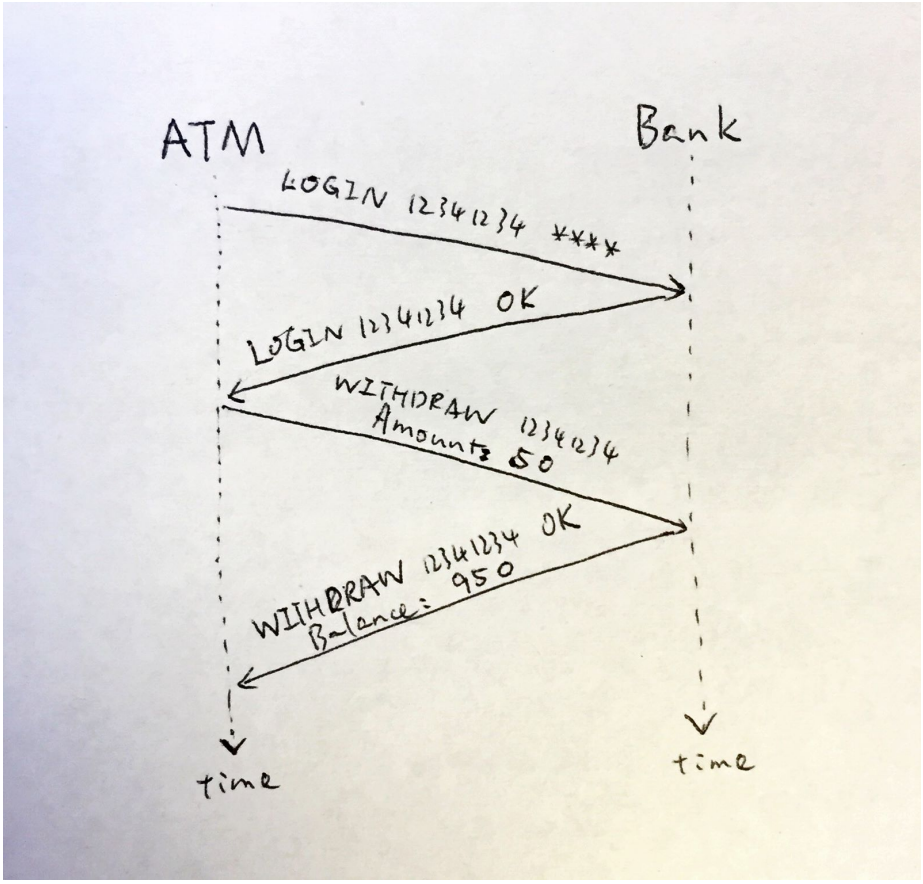
Bank:
[Queries database. If balance lower than amount requestsed]

```
WITHDRAW [card number] ERROR
  Reason: Insufficient fund
```

[Otherwise, updates the database]

```
WITHDRAW [card number] OK
  Balance: 2959.50
```

ATM: *[Dispenses cash]*



P7

The time for A to gather a packet is $\frac{56 \cdot 8}{64 \cdot 10^3} = 7 \cdot 10^{-3}s = 7ms$. The transmission delay is $\frac{56 \cdot 8}{2 \cdot 10^6} = 0.224 \cdot 10^{-3}s = 0.224ms$. So the total delay is

$$d_{proc} + d_{tran} + d_{prop} = 7 + 0.224 + 10 = 17.224ms$$

P8

a

$\frac{3 \cdot 10^6}{150 \cdot 10^3} = 20$ users.

b

The probability is 0.1.

c

$$P(N = n) = \binom{120}{n} \cdot 0.1^n \cdot 0.9^{120-n}$$

d

$$P(N \geq 21) = \sum_{n=21}^{120} P(N = n) = \sum_{n=21}^{120} \binom{120}{n} \cdot 0.1^n \cdot 0.9^{120-n}$$

P18

The three Traceroute's to www.google.com were performed on 12:46, 16:10 and 19:19 on September 1st. [See output on the last page]

a

- 12:46 - Average RTT: 6.164ms; Standard deviation: 5.774
- 16:10 - Average RTT: 7.910ms; Standard deviation: 15.608
- 19:19 - Average RTT: 5.766ms; Standard deviation: 5.153

b

- 12:46 - 10 routers
- 16:10 - 10 routers
- 19:19 - 10 routers

All paths are different except for the first 2-3 routers.

P22

At each link, the probability that the packet is successfully transmitted is $1-p$. There are N links in total, so the probablity that all of them successfully transmit the packet is $(1-p)^N$.

The probability that a single packet needs to be retransmitted is $1-(1-p)^N$. In other words, on average, the server needs to retransmit each packet $1-(1-p)^N$ times.

P23

a

All bits experience the same propagation delay, and there won't be any queueing delay if $R_s < R_c$, so the inter-arrival time is determined only by the transmission delay of the bottleneck, which is L/R_s

b

Yes, because the rate of packets arriving is greater than the rate of packets leaving the switch.

The switch starts to send out the first packet when the second packet arrives. Sending out the first packets takes $T_c = \dfrac{L}{R_c}$ seconds, and receiving the second packets takes $T_s = \dfrac{L}{R_s}$ seconds. To ensure the first packet is completely sent out when when the second packet is fully received and ready to be sent out,

$$T \geq T_c - T_s = \frac{(R_s - R_c)L}{R_s R_c}$$

P31

a

It takes $\frac{8 \cdot 10^6}{2 \cdot 10^6} = 4s$.

The data need to be transmitted 3 times, so the total time is $4s \cdot 3 = 12s$

b

It takes $\frac{10^4}{2 \cdot 10^6} = 5 \cdot 10^{-3}s$.

It should be fully received $5 \cdot 10^{-3}s$ after the first packet starts being sent from the first switch to the second.

c

The first packet takes $3 \cdot 5 \cdot 10^{-3}s$ to reach the destination. After the first packet arrives, each of the following 799 packets only takes an additional $5 \cdot 10^{-3}s$, so the total time is

$$5 \cdot 10^{-3}s \cdot (3 + 799) = 4.01s$$

It takes significantly less time to move the file with segmentation, because each bit now only needs to wait for a small chunk of the file to arrive before it can move on to the next switch.

P33

Similar to P31(b), the first packet takes $3 \cdot \frac{S+80}{R}$, the rest $\frac{F}{S} - 1$ packets each takes an additional $\frac{S+80}{R}$. So in total, the delay is

$$T = \frac{S+80}{R}\left(3 + \frac{F}{S} - 1\right)$$

$$T'(S) = \frac{1}{R}\left(\frac{F}{S} + 2\right) + \frac{S+80}{R}\left(-\frac{F}{S^2}\right) = 0$$

$$S = \sqrt{40F}$$

Traceroute output

12:46pm

```
 1  128.61.16.1 (128.61.16.1)  7.205 ms  2.623 ms  2.406 ms
 2  143.215.253.218 (143.215.253.218)  2.584 ms  2.435 ms  2.296 ms
 3  143.215.254.91 (143.215.254.91)  3.124 ms  3.120 ms  3.449 ms
 4  143.215.194.109 (143.215.194.109)  3.373 ms  3.211 ms  5.348 ms
 5  74.125.48.33 (74.125.48.33)  3.115 ms  3.149 ms  4.042 ms
 6  108.170.249.76 (108.170.249.76)  4.132 ms  3.698 ms
    108.170.249.163 (108.170.249.163)  4.085 ms
 7  209.85.250.96 (209.85.250.96)  4.500 ms
    72.14.233.199 (72.14.233.199)  4.064 ms  4.386 ms
 8  216.239.40.131 (216.239.40.131)  31.525 ms  10.459 ms  10.559 ms
 9  74.125.37.222 (74.125.37.222)  10.461 ms
    72.14.234.55 (72.14.234.55)  10.510 ms
    72.14.234.53 (72.14.234.53)  10.811 ms
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  172.217.197.106 (172.217.197.106)  11.940 ms * *
```

16:10pm

```
 1  128.61.16.1 (128.61.16.1)  4.142 ms  2.832 ms  2.385 ms
 2  143.215.253.218 (143.215.253.218)  2.402 ms  2.556 ms  2.278 ms
 3  143.215.254.91 (143.215.254.91)  3.351 ms  3.075 ms  3.347 ms
 4  143.215.194.109 (143.215.194.109)  83.651 ms  3.736 ms  3.340 ms
 5  74.125.48.33 (74.125.48.33)  3.297 ms  8.067 ms  3.196 ms
 6  108.170.249.162 (108.170.249.162)  3.274 ms
    108.170.249.163 (108.170.249.163)  3.674 ms
    108.170.249.44 (108.170.249.44)  3.625 ms
 7  72.14.233.199 (72.14.233.199)  28.006 ms
    209.85.246.223 (209.85.246.223)  4.060 ms
    72.14.233.199 (72.14.233.199)  5.883 ms
 8  209.85.244.143 (209.85.244.143)  3.919 ms
    216.239.50.104 (216.239.50.104)  3.909 ms
    108.170.230.67 (108.170.230.67)  4.313 ms
 9  216.239.51.195 (216.239.51.195)  7.589 ms
    108.170.231.171 (108.170.231.171)  5.257 ms
    216.239.50.57 (216.239.50.57)  8.419 ms
10  * * *
```

```
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  yb-in-f104.1e100.net (64.233.185.104)  5.423 ms  3.624 ms  3.838 ms
```

19:19pm

```
 1  128.61.16.1 (128.61.16.1)  3.985 ms  1.876 ms  1.824 ms
 2  143.215.253.218 (143.215.253.218)  1.893 ms  1.904 ms  1.726 ms
 3  143.215.254.91 (143.215.254.91)  2.447 ms  2.751 ms  24.431 ms
 4  143.215.194.109 (143.215.194.109)  2.909 ms  2.783 ms  2.698 ms
 5  74.125.48.33 (74.125.48.33)  2.657 ms  2.675 ms  2.770 ms
 6  108.170.249.67 (108.170.249.67)  3.291 ms  3.331 ms  3.904 ms
 7  108.170.229.80 (108.170.229.80)  4.615 ms
    209.85.246.223 (209.85.246.223)  5.562 ms
    216.239.63.88 (216.239.63.88)  9.736 ms
 8  216.239.40.119 (216.239.40.119)  10.408 ms
    216.239.40.133 (216.239.40.133)  10.510 ms
    216.239.40.119 (216.239.40.119)  10.160 ms
 9  209.85.251.243 (209.85.251.243)  14.793 ms
    209.85.243.184 (209.85.243.184)  10.042 ms
    72.14.234.53 (72.14.234.53)  10.003 ms
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  172.217.197.105 (172.217.197.105)  12.190 ms * *
```

CS 3251 Homework 2

Wenqi He

October 2, 2018

2.4

a. `gaia.cs.umass.edu/cs453/index.html`. From the `GET` line and the `Host` header.

b. HTTP 1.1. From the `GET` line.

c. Persistent. From the `Connection:keep-alive` header.

d. It's not in the HTTP message.

e. Netscape. From the `User-Agent` header. Because not all Web standards are fully supported by all browsers, and the source files intended for different devices (desktop, tablet, phones) might be maintained separately. The server can use this information to send appropriate files to different browsers.

2.5

a. Yes. (from the status code). 12:39:45 GMT on March 7, 2008 (from the `Date` header).

b. 18:27:46 GMT on December 10, 2005 (from the `Last-Modified` header).

c. 3874 bytes (from the `Content-Length` header).

d. `<!doc` (following the double cr-lf). Yes (from the `Connection:Keep-Alive` header).

2.10

Since the link is short, propagation delay is negligible. However, the link rate is extremely low, so even a small control packet can have a significant impact on transmission delay. For example, a three-way handshake to establish TCP connection would take $3 \cdot 200/150 = 4$ sec. If referenced objects are downloaded in parallel, each instance must establish a separate TCP connection, which is very time-consuming under this network condition. Furthermore, even if we ignore the time to establish connections, parallel instances wouldn't speed up download in this network, because all traffic goes through a single link with fixed finite bandwidth that is shared by everyone. The more instances there are, the lower the transmission rate is for each intance, and the overall transmission rate stays the same. Therefore it would always take the same amount of time to tranmit data, regardless of whether parallel download is used or not. On the other hand, if persistent HTTP is used, the client only needs to establish one TCP connection, which would reduce the transmission delay by as much as $10 \cdot 4 = 40$ sec.

2.19

a

For `cc.gatech.edu`:

```
a.root-servers.net -> f.edu-servers.net -> dns1.gatech.edu
```

b

For `google.com`:

```
b.root-servers.net -> c.gtld-servers.net -> ns2.google.com
```

For `yahoo.com`:

```
b.root-servers.net -> h.gtld-servers.net -> ns1.yahoo.com
```

For `amazon.com`:

```
c.root-servers.net -> f.gtld-servers.net -> pdns1.ultradns.net
```

2.22

	10	100	1000
300 Kbps	7500s	25000s	45454.55s
700 Kbps	7500s	15000s	20547.95s
2 Mbps	7500s	7500s	7500s

3.2

From server to process 1 on host C:
source port: 80, dest. port: 7532
source IP: B, dest. IP: C

From server to process 2 on host C:
source port: 80, dest. port: 26145
source IP: B, dest. IP: C

From server to process on host A:
source port: 80, dest. port: 26145
source IP: B, dest. IP: A

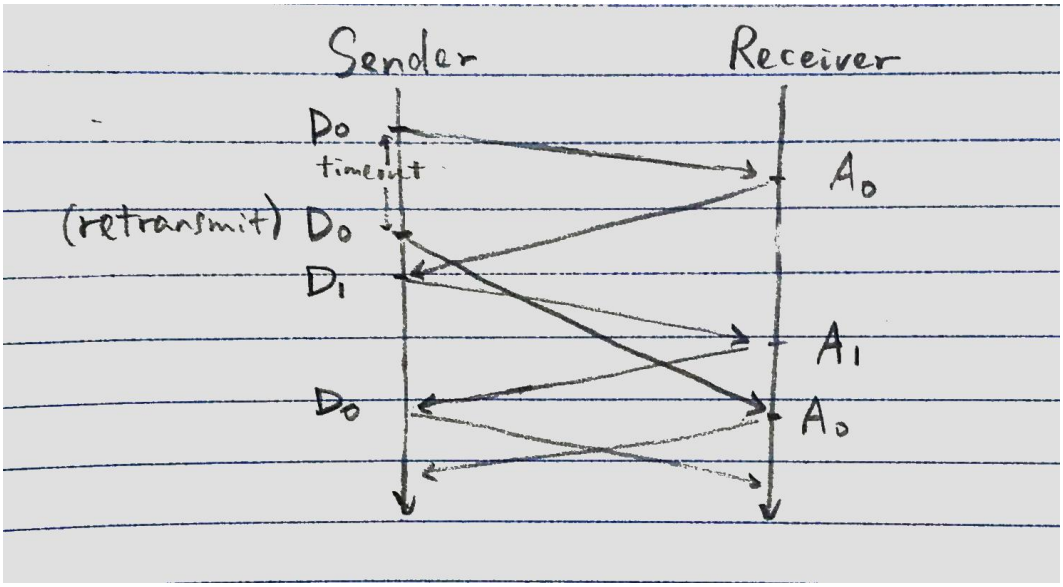
3.6

Suppose the receiver successfully receives a packet with sequence number 0, but the acknowledgement gets corrupted on its way back to the sender. The sender, according to rdt2.1, would resend packet 0, but the receiver is now expecting packet 1. According to Figure 3.57, upon receiving packet 0 it would respond with a NAK packet, which might be successfully received by the sender or also get corrupted. In either situation, the sender would resend packet 0 yet again. And the cycle continues.

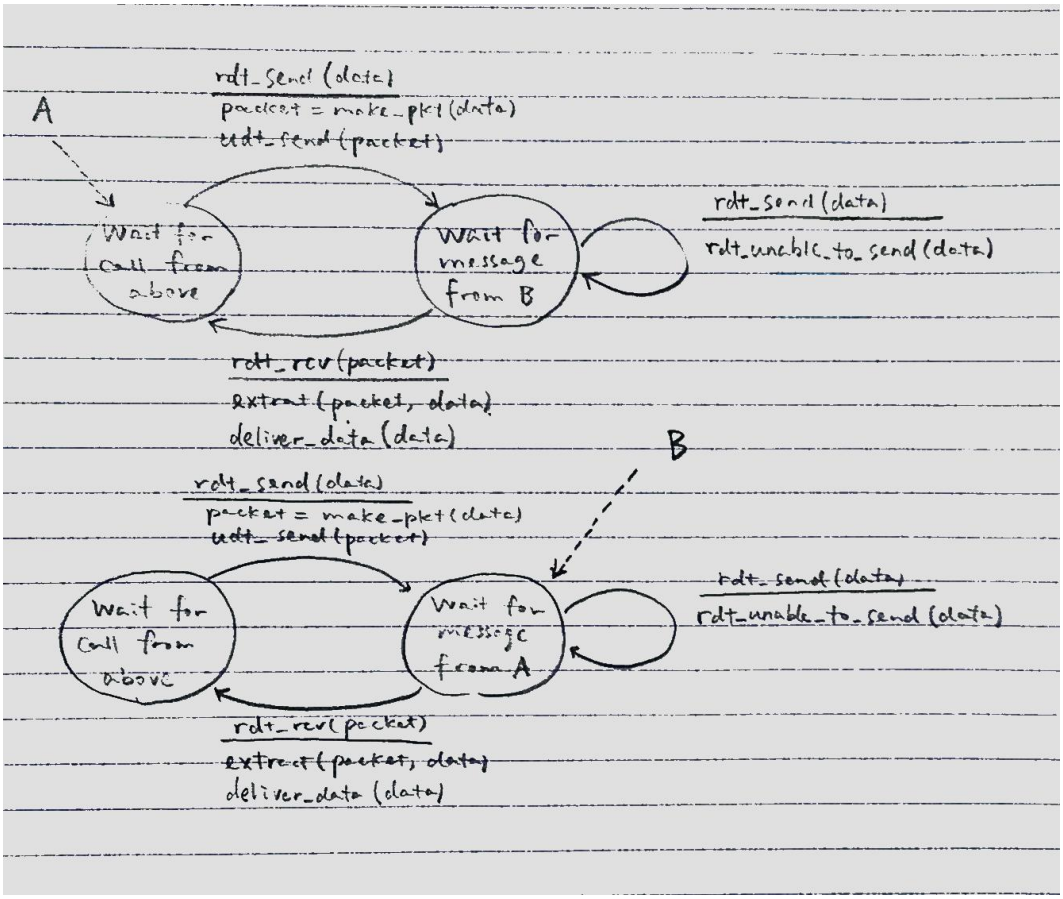
3.10

- Now the sender should start/restart a timer whenever a packet is sent/resent. (Whenever `udt_send(sndpkt)` is called.)
- When the sender is waiting for ACK/NAK, if timeout occurs, it should resend packet (call `udt_send(sndpkt)` again) and keep waiting. If it receives a corrupt packet or a NAK packet, instead of resending the packet immediately, it should simply ignore it, since the packet will eventually be retransmitted if its acknowledgement is not received before timeout.

3.13



3.17



3.23

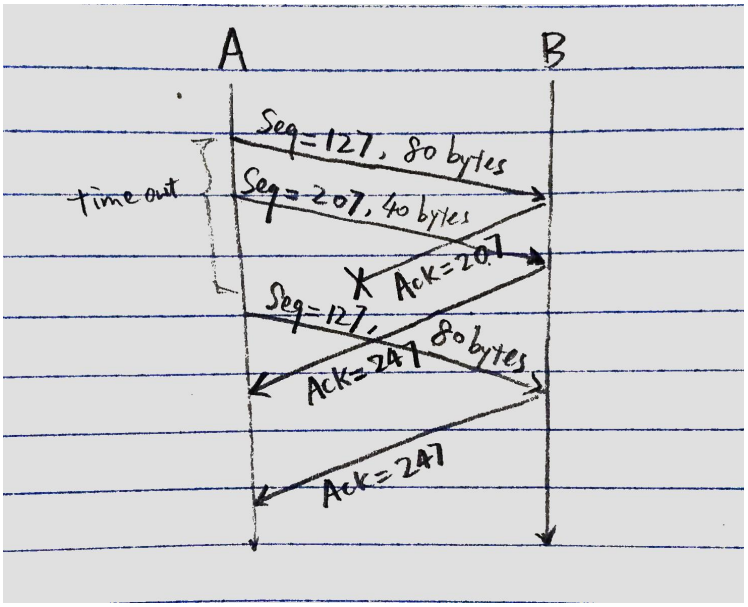
The largest allowable sender window size is $\lfloor \frac{k}{2} \rfloor$.

3.27

a. Sequence number: 207. Source port: 302. Destination port: 80.

b. Acknowledgement number: 207, Source por: 80. Destination port: 302.

c. Acknowledgement number: 127.



3.28

Host B will constantly inform Host A of how much spare room it has in its buffer (`rwnd`) through the receive window header in TCP packets. Meanwhile host A will keep track of `rwnd` and make sure the total size of unacknowledged packets is always less than `rwnd` so that it won't overflow B's buffer. Therefore, even though A has a much higher link rate, its sending speed will be throttled by TCP flow control in order to match B's receiving speed.

CS 3251 Homework 3

Wenqi He

November 1, 2018

3.40

(a) $[1, 6]$ and $[23, 26]$. Because the window size is growing exponentially.

(b) $[6, 16]$, $[17, 22]$. Because the window size is growing linearly.

(c) Triple duplicate ACK. Because the window size drops to roughly half of the original size.

(d) Timeout. Because the window size drops to 1.

(e) 32. That's where congestion avoidance begins.

(f) 21. It's half of the `cwnd` at the 16th round.

(g) 14. It's half of the `cwnd` at the 22nd round.

(h) The 7th round.

(i) The `cwnd` at the 26th round is 8.
So `ssthresh = cwnd/2 = 4` and `cwnd = ssthreash + 3 = 7`.

3.45

Assuming that the window size grows by 1 every RTT, then the total number of packets is

$$\frac{W}{2} + \left(\frac{W}{2} + 1\right) + \left(\frac{W}{2} + 2\right) + \cdots + W$$
$$= \frac{1}{2}\left(\frac{W}{2} + W\right)\left(W - \frac{W}{2} + 1\right) = \frac{3}{8}W^2 + \frac{3}{4}W$$

So the loss rate is

$$L = \frac{\#\text{ packets lost}}{\text{total }\#\text{ packets}} = \frac{1}{\frac{3}{8}W^2 + \frac{3}{4}W}$$

3.46

(a) In one RTT, the link can transmit at most $10\text{Mbps} \cdot 150\text{msec} = 1500\text{kbit}$, which translates to

$$W = \frac{1500 \cdot 10^3}{1500 \cdot 8} = 125 \text{ segments}$$

(b) The average window size is $3W/4 = 93.75$ segments. The average throughput is $93.75 \cdot 1500 \cdot 8\text{bit}/(1500 \cdot 10^{-3}\text{sec}) = 7.5\text{Mbps}$

(c) Assuming the window size increases by 1 segment every RTT, then it takes $(W/2) \cdot RTT = (125/2) \cdot 150 \cdot 10^{-3}\text{sec} = 9.375\text{sec}$

4.4

The minimum number of time slots needed is 2. If we use a sequence of 3-tuples to denote the packets being transfered from input ports to output ports, then we can achieve this by $(X, Y, Z), (null, X, Y)$.

The maximum number of time slots needed is 4, which happens in the following senario: First (X, X, Z) and the second X is blocked, then $(null, Y, Y)$ and the first Y is blocked. Now there are still two packets X, Y in the second input queue, which will take 2 more time slots to transfer.

4.5

(a)

11100000 00****** ******** ********	0
11100000 01000000 ******** ********	1
11100000 ******** ******** ********	2
11100001 0******* ******** ********	2
otherwise	3

(b) The first one doesn't match any prefix, so it will be forwarded to link 3. The second one matches the 4th entry, so it will be forwarded to link 2. The third one doesn't match any prefix, so it will be forwarded to link 3.

4.8

Subnet 1: 223.1.17.0/26
Subnet 2: 223.1.17.128/25
Subnet 3: 223.1.17.64/28

CS 3251 Homework 4

Wenqi He

November 30, 2018

4.1

(1)

The rows in each table are the distance vectors from each router's view:

w:

	w	x	y	z
w	0	5	1	1
y	1	4	0	∞
z	1	∞	∞	0

, **x:**

	w	x	y	z
x	5	0	4	6
y	1	4	0	2
z	1	6	2	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	∞	0	4	∞
y	1	4	0	2
z	1	6	2	0

, **z:**

	w	x	y	z
w	0	5	1	1
x	5	0	4	6
y	1	4	0	2
z	1	6	2	0

(2)

There will still be a "count-to-infinity" problem, because initially z routes to x through z-w-y-x, but y is unaware of it even though poison reverse is used. After y updates its route to y-z-x, it forms a closed loop of w, y and z. Wrong information will propagate along this loop repeatedly and increase the cost of each link slowly until they exceed the actual link costs, then the distance vectors will finally converge. As shown below, it takes 30 iterations in total.

1st iteration: (*x converges immediately*)

w:

	w	x	y	z
w	0	5	1	1
y	1	4	0	∞
z	1	∞	∞	0

, **x:**

	w	x	y	z
x	51	0	52	50
y	1	4	0	2
z	1	6	2	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	∞	0	4	∞
y	1	9	0	2
z	1	6	2	0

, **z:**

	w	x	y	z
w	0	5	1	1
x	5	0	4	6
y	1	4	0	2
z	1	6	2	0

2nd iteration:

w:

	w	x	y	z
w	0	10	1	1
y	1	9	0	∞
z	1	∞	∞	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	51	0	52	50
y	1	9	0	2
z	1	6	2	0

, **z:**

	w	x	y	z
w	0	5	1	1
x	∞	0	∞	50
y	1	∞	0	2
z	1	6	2	0

3rd iteration:

w:

	w	x	y	z
w	0	10	1	1
y	1	9	0	∞
z	1	∞	∞	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	51	0	52	50
y	1	9	0	2
z	1	6	2	0

, **z:**

	w	x	y	z
w	0	10	1	1
x	∞	0	∞	50
y	1	∞	0	2
z	1	11	2	0

4th iteration:

w:

	w	x	y	z
w	0	10	1	1
y	1	9	0	∞
z	1	∞	∞	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	51	0	52	50
y	1	14	0	2
z	1	11	2	0

, **z:**

	w	x	y	z
w	0	10	1	1
x	∞	0	∞	50
y	1	∞	0	2
z	1	11	2	0

5th iteration:

w:

	w	x	y	z
w	0	15	1	1
y	1	14	0	∞
z	1	∞	∞	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	51	0	52	50
y	1	14	0	2
z	1	11	2	0

, **z:**

	w	x	y	z
w	0	10	1	1
x	∞	0	∞	50
y	1	∞	0	2
z	1	11	2	0

8th iteration:

w:

	w	x	y	z
w	0	20	1	1
y	1	19	0	∞
z	1	∞	∞	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	51	0	52	50
y	1	19	0	2
z	1	16	2	0

, **z:**

	w	x	y	z
w	0	15	1	1
x	∞	0	∞	50
y	1	∞	0	2
z	1	16	2	0

14th iteration:

w:

	w	x	y	z
w	0	30	1	1
y	1	29	0	∞
z	1	∞	∞	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	51	0	52	50
y	1	29	0	2
z	1	26	2	0

, **z:**

	w	x	y	z
w	0	25	1	1
x	∞	0	∞	50
y	1	∞	0	2
z	1	26	2	0

20th iteration:

w:

	w	x	y	z
w	0	40	1	1
y	1	39	0	∞
z	1	∞	∞	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	51	0	52	50
y	1	39	0	2
z	1	36	2	0

, **z:**

	w	x	y	z
w	0	35	1	1
x	∞	0	∞	50
y	1	∞	0	2
z	1	36	2	0

26th iteration:

w:

	w	x	y	z
w	0	50	1	1
y	1	49	0	∞
z	1	∞	∞	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	51	0	52	50
y	1	49	0	2
z	1	46	2	0

, **z:**

	w	x	y	z
w	0	45	1	1
x	∞	0	∞	50
y	1	∞	0	2
z	1	46	2	0

27th iteration: (*z finally converges*)

w:

	w	x	y	z
w	0	50	1	1
y	1	49	0	∞
z	1	∞	∞	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	51	0	52	50
y	1	49	0	2
z	1	46	2	0

, **z:**

	w	x	y	z
w	0	50	1	1
x	∞	0	∞	50
y	1	∞	0	2
z	1	50	2	0

28th iteration:

w:

	w	x	y	z
w	0	50	1	1
y	1	49	0	∞
z	1	50	∞	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	51	0	52	50
y	1	53	0	2
z	1	50	2	0

29th iteration: (*x converges*)

w:

	w	x	y	z
w	0	51	1	1
y	1	53	0	∞
z	1	50	∞	0

, **y:**

	w	x	y	z
w	0	∞	1	1
x	51	0	52	50
y	1	53	0	2
z	1	50	2	0

30th iteration: (*y converges*)

		w	x	y	z
	w	0	51	1	1
y:	x	51	0	52	50
	y	1	52	0	2
	z	1	50	2	0

4.2

(1)

W would use WABX or WACX to reach X and WACY to reach Y. X would use XBAW or XCAW to reach W and XCY to reach Y.

(2)

A should advertise path AW to B only, and AV to both B and C. C receives AS routes: CY, CX, CB, CBX, CBA, CBAW, CBAV, CA, CAV

4.17

a

First create a variable a_0 and assign to it the identification number of the first datagram. Then, whenever a new packet is captured, if the identification number equals $a_i + 1$ for any a_i, increment that variable: $a_i = a_i + 1$, otherwise assign the number to a new variable. Because the identification numbers of the packets from each host is contiguous, each variable keeps track of the datagrams coming from a single host. Therefore, the number of variables is exactly the number of unique hosts.

b

No, it would not work. Because the above technique relies entirely on the identification numbers being sequential. If the number is random, then there would be no way to tell if a new datagram comes from the same host as any previously received datagrams.

6.11

a

For a node to succeed in any given time slot, only that node can transmit in that time slot, that is

$$P(\text{A succeeds}) = P(\text{A transmitting}) \cdot P(\text{B, C, D not transmitting}) = p(1-p)^3$$

If slot 5 is the first time it succeeds, it must have failed in slots 1 - 4, therefore the probability is

$$\boxed{P(\text{success}) \cdot P(\text{fail})^4 = p(1-p)^3\Big[1 - p(1-p)^3\Big]^4}$$

b

Since the events that any of the nodes succeeds are disjoint, the probablity is simply

$$P = P(A) + P(B) + P(C) + P(D) = 4p(1-p)^3$$

c

$$P = 4p(1-p)^3\left[1-4p(1-p)^3\right]^2$$

d

The probablity that the first success occurs in slot n is a geometric distribution

$$P(n) = 4p(1-p)^3\left[1-4p(1-p)^3\right]^{n-1}$$

The expected number of time slots needed to transmit 1 packet is the expected number of time slots needed for the first success:

$$E[n] = \frac{1}{4p(1-p)^3}$$

If no collision occured, each time slot could allow one packet to transmit. So the efficiency is

$$\frac{\#\text{ packets actually sent}}{\#\text{ packets could have been sent}} = \frac{1}{E(n)} = 4p(1-p)^3$$

6.15

a

No. F is in the same subnet as E, so E can get F's MAC address through ARP and send the datagram to F directly. The source IP and MAC addresses are E's IP and MAC addresses, and the destination IP and MAC addresses are F's.

b

No, because B is not in the same subnet as E. The source and destination IP are E and B's IP. The source MAC address is E's MAC address, the destination MAC address is the router R1's MAC address.

c

Since S1's forwarding table contains an entry for B, it will just forward the message through the corresponding port instead of broadcasting it, therefore R1 will not receive the message.

No, because B has already learned A's MAC address from the ARP request message it just received from A. S1 would just forward the message to A, because it has also learned A's MAC address from the ARP request it received earlier.

6.21

A: IP: 111.111.111.111, MAC: 1A-1A-1A-1A-1A-1A
R1 (Subnet 1): IP: 111.111.111.110, MAC: 2A-2A-2A-2A-2A-2A
R1 (Subnet 2): IP: 222.222.222.222, MAC: 1B-1B-1B-1B-1B-1B
R2 (Subnet 2): IP: 222.222.222.220, MAC: 2B-2B-2B-2B-2B-2B
R2 (Subnet 3): IP: 333.333.333.330, MAC: 2C-2C-2C-2C-2C-2C
F: IP: 333.333.333.333, MAC: 1C-1C-1C-1C-1C-1C

i

source MAC: 1A-1A-1A-1A-1A-1A, destination MAC: 2A-2A-2A-2A-2A-2A
source IP: 111.111.111.111, destination IP: 333.333.333.333

ii

source MAC: 1B-1B-1B-1B-1B-1B, destination MAC: 2B-2B-2B-2B-2B-2B
source IP: 111.111.111.111, destination IP: 333.333.333.333

iii

source MAC: 2C-2C-2C-2C-2C-2C, destination MAC: 1C-1C-1C-1C-1C-1C
source IP: 111.111.111.111, destination IP: 333.333.333.333

6.28

Suppose the EE hosts are in subnet 111.111.111.111/24, and the CS hosts are in subnet 222.222.222.222/24. The router's interface in EE subnet has IP addresss 111.111.111.000 and MAC address AA-AA-AA-AA-AA-AA, and its interface in CS subnet has IP 222.222.222.000 and MAC address BB-BB-BB-BB-BB-BB.

Suppose EE host A with IP 111.111.111.111 and MAC address 1A-1A-1A-1A-1A-1A wants to send an IP datagram to CS hsot B with IP 222.222.222.222 and MAC address 2B-2B-2B-2B-2B-2B. The steps taken are:

1. Host A's network layer creates an IP datagram with source IP 111.111.111.111 and destination IP 222.222.222.222.
2. A's link layer broadcasts an ARP request for the MAC address of the router, and the router responds with its MAC address AA-AA-AA-AA-AA.
3. A's link layer encapsulates the IP datagram in an Ethernet frame with source MAC address 1A-1A-1A-1A-1A-1A and destination MAC address AA-AA-AA-AA-AA-AA, and sends out the frame.
4. The router receives the frame, extracts the IP datagram and passes it up to the network layer. From the forwarding table, the network layer determines that it should forward the datagram to the port corresponding to the CS subnet.
5. The router's link layer broadcasts an ARP request in the CS subnet for B's MAC address, and B responds with 2B-2B-2B-2B-2B-2B.
6. The router's link layer then encapsulates the IP datagram in a new Ethernet frame with source MAC address BB-BB-BB-BB-BB-BB and destination MAC address 2B-2B-2B-2B-2B-2B, and sends out the frame.
7. Finally B receives the frame, extracts the IP datagram and passes it up to its network layer.

When device connects to the Internet:

1. Host broadcasts a DHCP request encapsulated in UDP packet.
2. DHCP server responds with an IP address, name and IP of a DNS server and the first-hop router's IP.

When navigating to `https://example.com/some/path?key=value`

1. Browser creates a DNS query, encapsulated in UDP packet, encapsulated in IP datagram.
2. Host broadcasts a ARP query for the MAC address of the first-hop router.
3. Router responds with its MAC address; Host stores the MAC addresses in its ARP cache.
4. Host sends out Ethernet/802.11 frame containing the query, which is then forwarded to the DNS server.
5. DNS server responds with the IP address of `example.com`.
6. Host performs a 3-way handshake (SYN, SYN-ACK, ACK) to establish TCP connection with server.
7. Host performs TLS key exchange with HTTP server and starts an encrypted session.
8. Browser sends out HTTP reqeust `GET /some/path?key=value HTTP/1.1\r\n Host:example.com...`
9. HTTP server handles the path and query string and sends back a HTTP response.
10. Browser parses the source code contained in the HTTP response and renders the page.