Name_____ Student ID_____ Department/Year_____

3rd Examination

Introduction to Computer Networks (Online)
Class#: EE 4020, Class-ID: 901E31110
Spring 2023

10:20-12:10 Wednesday
June 7, 2023

Cautions

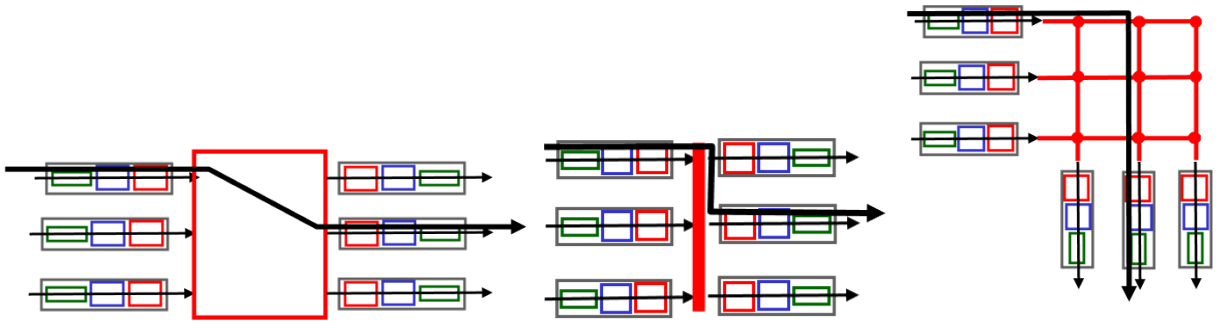
1. There are in total 100 points to earn. You have 90 minutes to answer the questions. Skim through all questions and start from the questions you are more confident with.
2. Use only English to answer the questions. Misspelling and grammar errors will be tolerated, but you want to make sure with these errors your answers will still make sense.
3. Grading policy in general: -1pt per error till 0pt left.
4. Grading policy for PA problem sets: pts for later subproblems will be given only when the formers are completed.

1. (ch41, 4pt) Tell which of the following actions happen at the data plane.
   (a) Route computation
   (b) Longest prefix matching
   (c) Generalized forwarding
   (d) Packet dropping
   (e) Packet retransmission
   (f) Route advertising

Sample Solution:

(b)(c)(d)

2. (ch42, 7pt) Below are 3 forms of switching fabric in a router.



(a) (b) (c)

   (1) Which one is the memory switching fabric? (1pt)
   (2) Which one is the bus switching fabric? (1pt)
   (3) Which one is the crossbar switching fabric? (1pt)
   (4) Tell one disadvantage of using the memory switching fabric. (1pt)
   (5) Tell one disadvantage of using the bus switching fabric. (1pt)
   (6) Tell one disadvantage of using the crossbar switching fabric. (2pt)

Sample Solution:

(1) (a)
(2) (b)
(3) (c)
(4) Memory read/writing slow
    Multiple bus crossings per packet
(5) Bus speed (although faster than memory) can be limited
    Bus contention
(6) Fixed length cells fragmentation to allow better contention control
    Path setup time

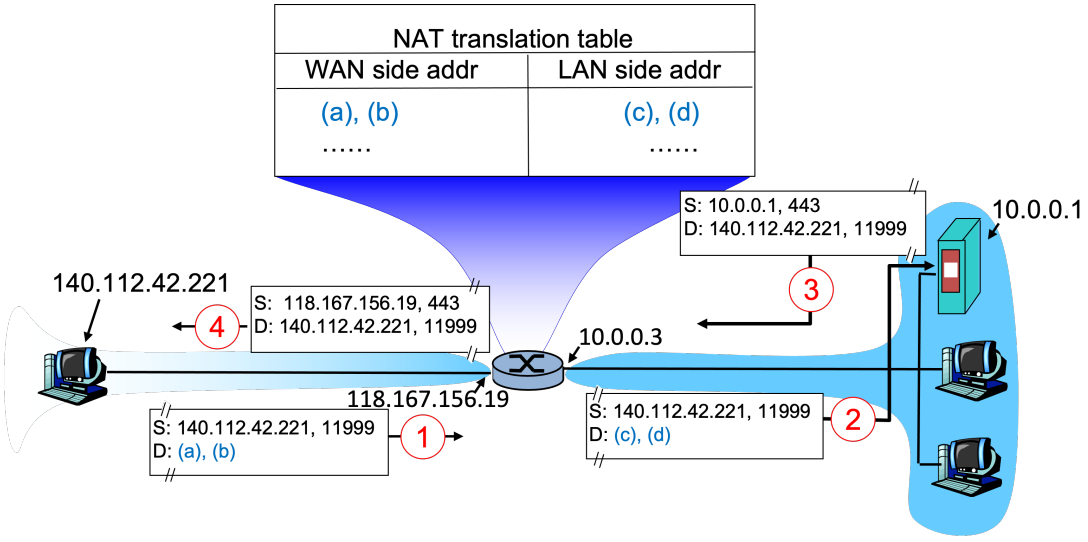
Hardware more expensive

3. (ch43, 4pt) Tell which of the followings are benefits of longest prefix matching.
   (a) Saving the forwarding table space. (1pt)
   (b) Speeding up the forwarding table lookup process. (1pt)
   (c) Reducing the amount of the route advertisements. (1pt)
   (d) Allowing division of an address block to multiple sub-blocks within an organization. (1pt)

Sample Solution:

(a)(b)(c)(d)

4. (ch43, 10pt) Most homes are allowed only 1 IP address for Internet access. To allow multiple smart devices accessing the Internet simultaneously at home, a common practice is to run NAT at the home gateway. This however makes it a little harder to run a server at home. Illustrated below is a Web server behind NAT at 10.0.0.1 port 443. To allow arbitrary users to access the server, the server admin needs to (1) set up an entry manually in the NAT translation table and (2) inform all users of the Web server's WAN side address (including the WAN side IP and port number). Provided the setting below, tell (a) the WAN side IP address, (b) the WAN side port number, (c) the LAN side IP address, and (d) the LAN side port number of the Web server.



(1) Tell the value of (a). (1pt)
(2) Tell the value of (b). (1pt)
(3) Tell the value of (c). (1pt)
(4) Tell the value of (d). (1pt)

In practice, the server admin will need to find the LAN side address as well as the WAN side address to set the entry up. Pretend that the server process is running on the machine you are currently using. The LAN side address is exactly the IP address of your machine and the port number the server process is running on.

(5) Tell the operating system (OS) of your machine. (1pt)

(6) Upload a photo or screenshot of your machine's OS setting where one can see the machine's IP address. (2pt)

The WAN side address is the IP address of the home gateway and the port number the server admin chooses to map to the internal server. Usually, one uses the port number default to the service (e.g., 80 for HTTP, 443 for HTTPS). To find the IP address of the home gateway, the server admin will need to log into the home gateway and look around its system settings. Another way is to google with phrases such as "what's my IP address" to find Web services that display your machine's WAN side IP address.

(7) Find one of these "what's my IP address" services and tell its URL. (1pt)

(8) Upload a photo or screenshot of the Web page that shows the WAN side IP address of your machine. (2pt)

Sample Solution:

(1) 118.167.156.19

(2) 443

(3) 10.0.0.1

(4) 11999

(5)-(8) whatever you've discovered.

5. (ch43, 12pt) Recall how fragmentation and reassembly is done in IPv4. When a 4000-byte IPv4 packet (ID=x) runs into a link with MTU = 1500 bytes. The network layer protocol will break the original IPv4 packet into 3 fragments as follows.

(length, ID, flag, offset)	=	(1500, x, 1, 0)
		(1500, x, 1, 185)
		(1040, x, 0, 370)

(1) Suppose that the 4000-byte IPv4 packet (ID=x) runs into a link with MTU = 1300 bytes instead. How many fragments will the network layer send on the MTU = 1300 bytes link? (1pt)

(2) Continue from (1). Tell the (length, ID, flag, offset) of all these fragments. (2pt)

(3) Suppose that the 4000-byte IPv4 packet runs into a link with MTU = 1500 bytes first and then a link with MTU = 1300 bytes. How many fragments will the network layer send on the MTU = 1300 bytes link to deliver the content in the original 4000-byte IPv4 packet? (1pt)

(4) Continue from (3). Tell the (length, ID, flag, offset) of all these fragments. (3pt)

(5) Continue from (3) except that the network layer protocol is upgraded to IPv6. Note that IPv6 does not do fragmentation anymore. Tell how many IPv6 packets will be sent to deliver the content in the original 4000-byte IPv4 packet. (1pt)

(6) Continue from (5). Tell the length of all these IPv6 packets. (4pt)

Sample Solution:

(1) 4 fragments

(2) (length, ID, flag, offset) = (1300, x, 1, 0)
(1300, x, 1, 160)
(1300, x, 1, 320)
(160, x, 0, 480)

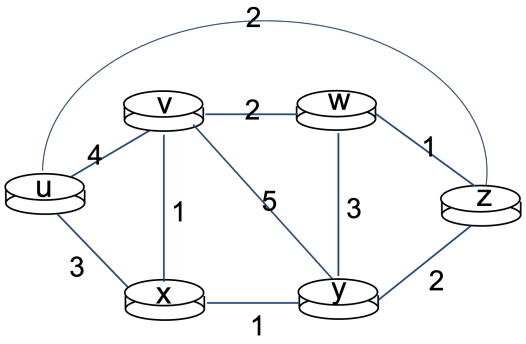
(3) 5 fragments

(4) (length, ID, flag, offset) = (1300, x, 1, 0)
(220, x, 1, 160)
(1300, x, 1, 185)
(220, x, 1, 345)
(1040, x, 0, 370)

(5) 4 IPv6 packets

(6) 1300, 1300, 1300, 240

6. (ch52, 10pt) Using the Dijkstra algorithm, one can compute the shortest path from a source node to all other nodes in the network. Consider the 6-node network illustrated below. Generate the table indicating the steps deriving the cost D(*) and previous hop p(*) from node u to all other nodes. Note that when the path costs are equal, add nodes to the traversed set N' in alphabetic order.



N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
u					

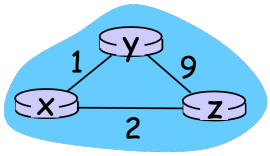
(1) Tell the traversed set N' (in order) when Dijkstra algorithm terminates. (2pt)

(2) Fill the derivation table till Dijkstra algorithm terminates. (8pt)

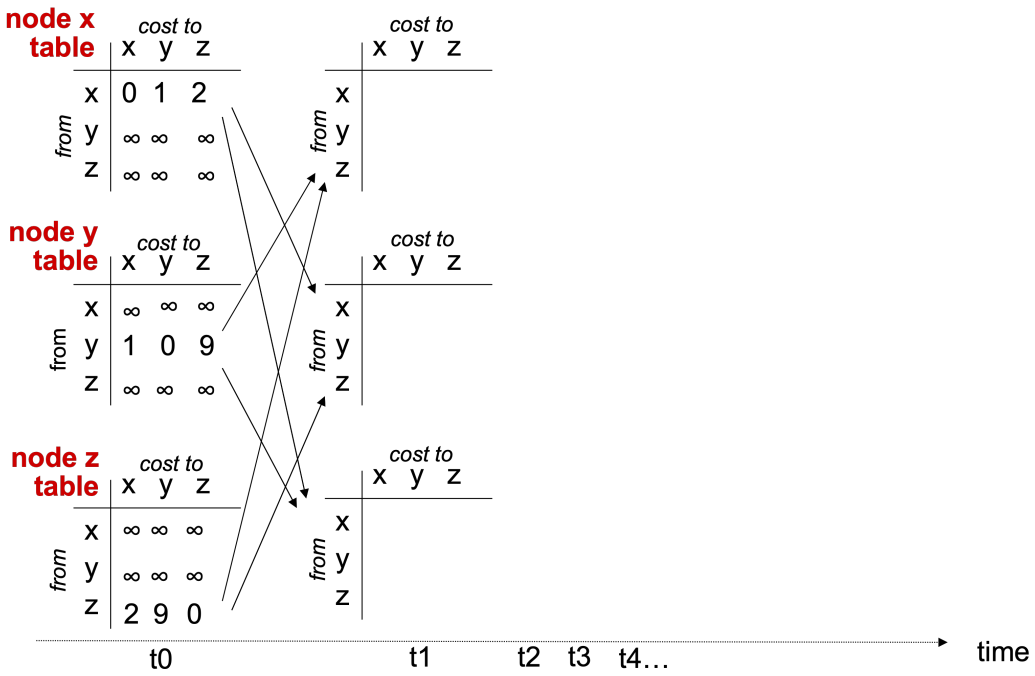
Sample Solution:

N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
u	4,u	inf	3,u	inf	2,u
uz	4,u	3,z	3,u	4,z	
uzw	4,u		3,u	4,z	
uzwx	4,u			4,z	
uzwxv				4,z	
uzwxvy					

7. (ch52, 22pt) Consider a simple network below. Develop the full DV tables at node x, y, and z until the DV routing algorithm converges (i.e., no more changes to propagate further).



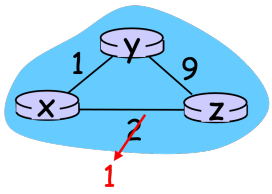
Please follow the derivation style as shown in the lecture. Assume the 3 nodes synchronously receive, compute, and send the DVs if there's any change. The initialization step at t0 is illustrated below. The first iteration happens at t1, the 2nd iteration at t2, and so on so forth.



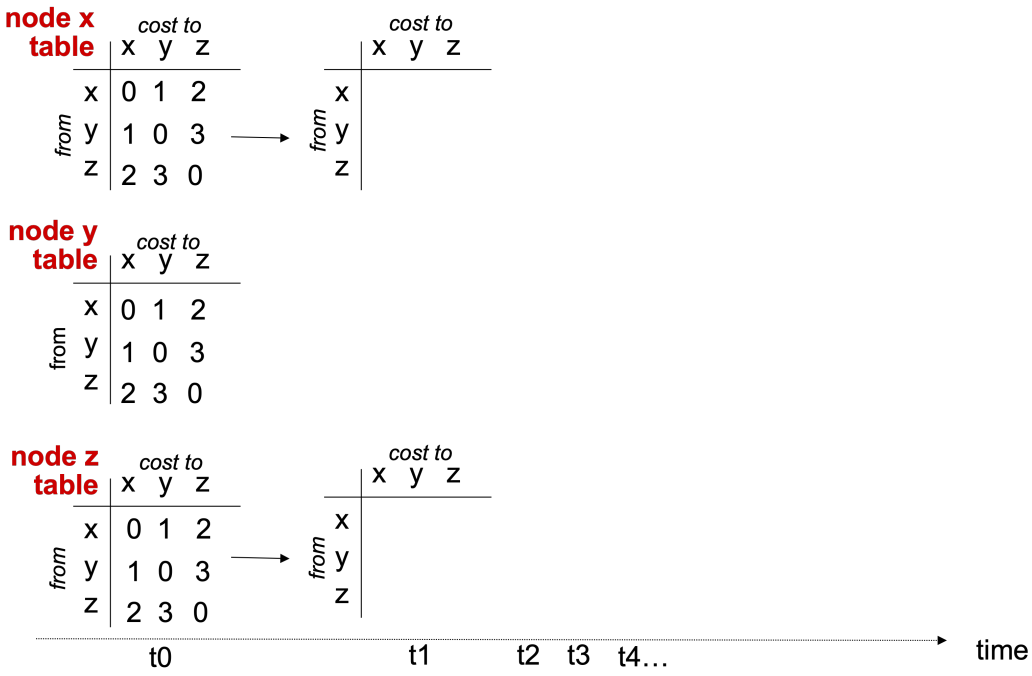
(1) Show the Bellman-Ford computation to derive Dx at node x at t1. Type 'inf' to represent infinity on the exam form. (3pt)

(2) Show the Bellman-Ford computation to derive Dz at node z at t1. Type 'inf' to represent infinity on the exam form. (3pt)

Continue from (2). Good news travel fast. Let's consider the following change to the network. Develop the full DV tables at node x, y, and z until the DV routing algorithm converges (i.e., no more changes to propagate further).



Please follow the derivation style as shown in the lecture. Assume also the 3 nodes synchronously receive, compute, and send the DVs if there's any change. The tables before the change are those shown at t0. The first iteration at t1 is triggered by the link cost change, the 2nd iteration at t2 is triggered by receiving the subsequent DV reports, and so on so forth.



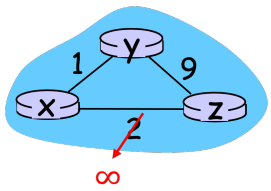
(3) Tell Dx at node x at t1. (1pt)

(4) Tell Dz at node z at t1. (1pt)

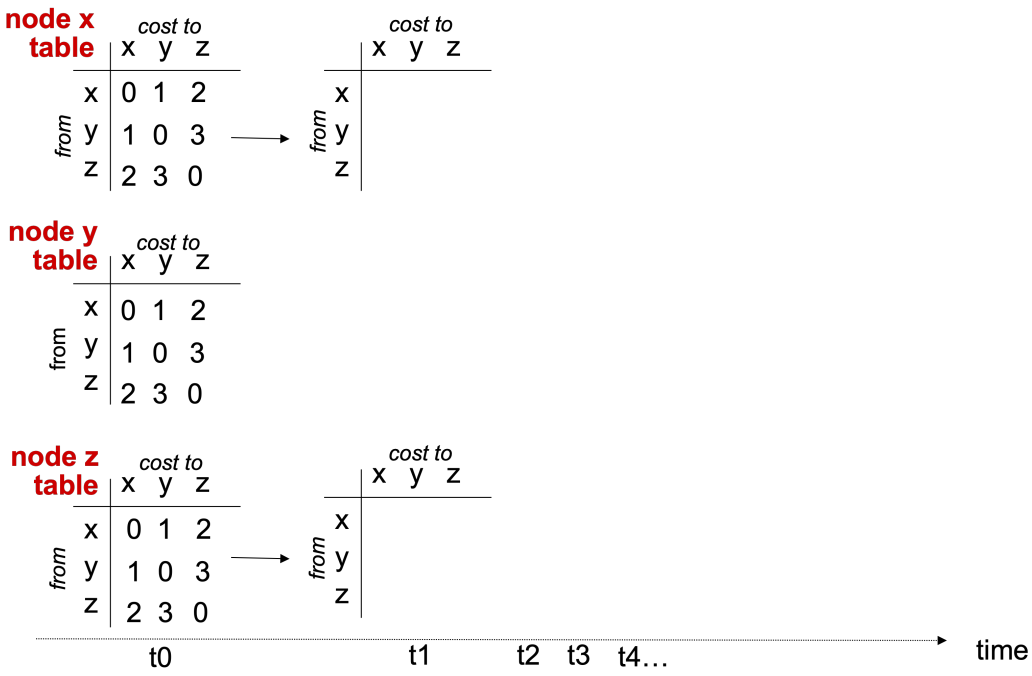
(5) Tell Dy at node y at t2. (2pt)

(6) Tell to which iteration (the n in the final tn) the DV tables converge. (2pt)

Continue from (2). Bad news travel slow. Let's consider the following change to the network. Develop the full DV tables at node x, y, and z until the DV routing algorithm converges (i.e., no more changes to propagate further).



Again, use the derivation style shown in the lecture. Assume the 3 nodes synchronously receive, compute, and send the DVs if there's any change. The tables before the change are those shown at t0. The first iteration at t1 is triggered by the link cost change, the 2nd iteration at t2 is triggered by the subsequent DV changes, and so on so forth. Assume also that the poisoned reverse mechanism is disabled.



(7) Tell Dx at node x at t1. (1pt)

(8) Tell Dz at node z at t1. (1pt)

(9) Tell Dy at node y at t2. (2pt)

(10) Tell Dx at node x at t3. (2pt)

(11) Tell Dy at node y at t4. (2pt)

(12) Tell to which iteration (the n in the final tn) the DV tables converge. (2pt)

Sample Solution:

(1) Dx at node x at t1 = (0, 1, 2)

dx(y) = min{c(x,y)+dy(y), c(x,z)+dz(y)} = min{1+0, 2+9}=1

dx(z) = min{c(x,y)+dy(z), c(x,z)+dz(z)} = min{1+9, 2+0}=2

(2) Dz at node z at t1 = (2, 3, 0)

dz(x) = min{c(z,x)+dx(x), c(z,y)+dy(x)} =min{2+0, 9+1}=2

dz(y) = min{c(z,x)+dx(y), c(z,y)+dy(y)} =min{2+1, 9+0}=3

(3) Dx at node x at t1 = (0, 1, 1)

(4) Dz at node z at t1 = (1, 2, 0)

(5) Dy at node y at t2 = (1, 0, 2)

(6) t3

(7) Dx at node x at t1 = (0, 1, 4)

(8) Dz at node z at t1 = (10, 9, 0)

(9) Dy at node y at t2 = (1, 0, 5)

(10) Dx at node x at t3 = (0, 1, 6)

(11) Dy at node y at t4 = (1, 0, 7)

(12) t8

8. (PA, 11pt) Please go on the PA workstation and work under the exam3 subdirectory for this problem set. Create the exam3 subdirectory if you haven't done so.
   (1) Develop exam3-p8-1.go such that it works as a Web server – accepting HTTP requests for files and returning the files requested in the HTTP responses. Use server-test.html for testing (the test file for PA7-PA9). (2pt)
   (2) Develop exam3-p8-2.go such that it works as a Web server – accepting HTTP requests for files, returning the files requested in the HTTP responses if the files are found or returning "Sorry, not here." if the files are not found. (3pt)
   (3) Move the server.key and server.cer file in your home directory to the exam3 subdirectory. (1pt)
   (4) Develop exam3-p8-3.go such that it extends exam3-p8-2.go to allow HTTPS requests and responses. Use the server.key and server.cer for the secure Web server. (2pt)
   (5) Develop exam3-p8-4.go such that it extends exam3-p8-3.go and the Web server returns "Howdy!" when the client requests with this URL – "https://<server IP>:<port#>/greetings". (3pt)

Sample Solution:

Whatever that works.

9. (PA, 10pt) Please go on the PA workstation and work under the exam3 subdirectory for this problem set. Create the exam3 subdirectory if you haven't done so. Move the client.key and client.cer file in your home directory to exam3 subdirectory.

   One can implement a pair of secure client and server to exchange encrypted messages without HTTPS. You will work towards a secure "Price is Right!" game client in this problem set. This client will interact with the secure "Price is Right!" game server running on port 12000. Note that the server.key and server.cer used to configure the server are paired with the client.key and client.cer now in the exam3 subdirectory. Therefore, do use the client.key and client.cer to configure your client. Otherwise, they won't connect.

```
package main

import "crypto/tls"

func check(e error) {
    if e != nil {
        panic(e)
    }
}

func main() {
    cert, err := tls.LoadX509KeyPair("client.cer", "client.key")
    check(err)
    //skip checking the certificate
    config := tls.Config{Certificates: []tls.Certificate{cert}, \
    InsecureSkipVerify: true}
    conn, _ := tls.Dial("tcp", ":12000", &config)
    defer conn.Close()
}
```

   (1) Copy the secure client example above into exam3-p9-1.go. Make sure that it connects to the secure server on port 12000 and then closes the connection. (2pt)
   (2) Extend exam3-p9-1.go to exam3-p9-2.go such that it connects to the secure server on port 12000, sends "play\n", receives a line of message from the server, prints the message on the screen, and then closes the connection. (2pt)

   You should see the response from the server on port 12000. It says it is the secure version of the "Price is Right!" game engine. In the game, the player guesses the price of

a merchandize. The true price lies in the range 1-10000. Given a guess, the game engine responses with either “Too high”, “Too low”, or “Bingo”. “Too high” means the guess is higher than the true price and “Too low” means the guess is lower. In either case, the game engine will continue to wait for the next guess until the guess is right. “Bingo” means the guessed price is right and the game ends at this point.

(3) Extend exam3-p9-2.go to exam3-p9-3.go such that it connects to the secure server on port 12000, sends “play\n”, receives a line of message from the server, prints the message on the screen. Then, the secure client prompts the user for a guess, sends the number in one line, receives a line of message from the server, prints the line of message on the screen, and then closes the connection. (3pt)

(4) To complete the secure game client, extend exam3-p9-3.go to exam3-p9-4.go such that it connects to the secure server on port 12000, sends “play\n”, receives a line of message from the server, prints the message on the screen. Then, the client repeatedly prompts the user for a guess, sends the number in one line, receives a line of message from the server and prints the line of message on the screen until the server response is “Bingo\n”. After receiving the “Bingo\n” response, the client closes the connection. (3pt)

Sample Solution:

Whatever that works.

10. (PA, 10pt) Please go on the PA workstation and work under the exam3 subdirectory for this problem set. Create the exam3 subdirectory if you haven't done so. Move the server.key and server.cer file in your home directory to exam3 subdirectory if you have not yet done so.

    Let's build a secure "Price is Right!" game server mimicking how one listens on a secure socket as shown in the sec-string-Response.go example in PA9. The end goal is to implement a secure server that works just like the secure "Price is Right!" game server on port 12000. Use the server.key and server.cer for the secure server.

    (1) Develop exam3-p10-1.go running on your exam port # such that it allows multiple clients such as your exam3-p9-1.go to connect and then close. (2pt)
    (2) Develop exam3-p10-2.go such that it allows multiple clients such as your exam3-p9-2.go to connect, send "play\n", receive a line of response from this exam3-p10-2.go saying "Welcome to The SECURE Price is Right! Let's guess the price of a 12-person tent.\n" and then close. (2pt)
    (3) Develop exam3-p10-3.go such that it allows multiple clients such as your exam3-p9-3.go to connect, send "play\n", receive a short response from this exam3-p10-3.go saying "Welcome to The SECURE Price is Right! Let's guess the price of a 12-person tent.\n". Afterwards, have this exam3-p10-3.go set the true price to 9053, allow your exam3-p9-3.go to send a number in 1-10000 followed by "\n", receive a short response from this exam3-p10-3.go saying the guess is "Too high\n" if the guess is higher than the true answer, "Too low\n" if the guess is lower, or "Bingo\n" if the guess is right. Then, have this exam3-p10-3.go close the connection. (3pt)
    (4) Develop exam3-p10-4.go such that it allows multiple clients such as your exam3-p9-4.go to connect, send "play\n", receive a short response from this exam3-p10-4.go saying "Welcome to The SECURE Price is Right! Let's guess the price of a 12-person tent.\n". Afterwards, have this exam3-p10-4.go set the true price to 9053, allow your exam3-p9-4.go to repeatedly send a number in 1-10000 followed by "\n", receive a short response from this exam3-p10-4.go saying the guess is "Too high\n" if the guess is higher than the true answer, "Too low\n" if the guess is lower, or "Bingo\n" if the guess is right. The repetition stops when the guess is right. Then, have this exam3-p10-4.go close the connection with your exam3-p9-4.go. (3pt)

Sample Solution:

Whatever that works.