CPSC 3600 section 002 HW Shortened #1
Spring 2020
Last revision: 1/29/2020

You do not have to turn this homework in – it is not graded. You can work with a partner (or in a group). But I highly recommend that you use your partner/group to help understand the questions and the background knowledge required to answer the questions and then you complete the homework on your own. I'll release the solution after the HW due date. The text assumes you are to submit the homework, so you will see things like, 'your submission should include …. '. I kept these statements in the homework to help fully understand the context of the questions.

Update 1/26/2020: The difference between this shortened HW1 and the original HW1 is that I am replacing Question 3 replacing it with a similar question but that works with starting code that is much cleaner and easier to use. Anyone that would like to complete HW1 as originally described please come and talk to me - you will need to show me you understand what is being asked. If you do understand, you will be considered an advanced student and we can talk about an alternative course direction for you.

Update 1/29/2020: I've extended the solution to Exercise 2 which caused me to more precisely define what you need to do for Question 3. You should complete question 3 using the posted solution to Exercise 2 (located in git ./exercises/ex2).

Section 1 (40 points) Short Answer Questions

Question 1.1 (5) List the layers from the OSI seven layer network model that apply in a TCP/IP network. Provide a 1sentence summary of each layer in a TCP/IP context.

Question 1.2 (5) Assume two nodes communicate directly over a physical layer. Explain the difference between the channel bandwidth and the baud rate in a context where the two nodes communicate directly over a physical layer.

Question 1.3 (5) New automobiles sold in the United States will soon be required to support wireless communications using a form of WiFi. There are seven channels available to support vehicle-to-vehicle communications. The bandwidth of each channel is 10Mhz. In the 'best conditions', it uses a modulation/demodulation based on QAM-64 which uses 64 signal levels per symbol. What is the maximum data rate in these 'best conditions' ?   Please state any assumptions you might make for your answer. Hint: look up the Nyquist theorem.

Question 1.4 (5) In a connected vehicle scenario as described in the previous question, applications running on vehicles can chose to use higher order modulation (64QAM) or lower order modulation (BPSK or QPSK). Why might a connected vehicle application transmit application messages using a lower order modulation such as QPSK rather than a higher order modulation (which offers higher data rates than lower modulations) such as 64 QAM?

Question 1.5 (6) When transmitting a 32-bit 2's complement integer in big-endian order, when is the sign bit transmitted (relative to other bits in the 32bit Integer). Please state any assumptions you make.

Question 1.6 (5) From the kurose text, Review question R12. In the 7th edition, this question is on page 68 and begins "What advantage does a circuit-switched network have…"

Question 1.7 (10) From the Kurose text, review question R13. In the 7th edition, this question is on page 68 and begins "Suppose users share a 2 Mbps link…."

Section 2 (30) Analyzing ping results

Use the ping program on your VM with one of the following hosts: 202.58.60.194 or 202.58.60.194. You can use any other host as long as it is located outside of the US (or….to the best of your ability, you think it is located outside the US). Use traceroute or other tools to help estimate the physical location. I'll refer to the destination node as the 'server'. Please do these two parts of this question.

Question 2.1 (10 points)

- Run ping between your VM and the server for at least 1 hour. Redirect standard output to a file.
- Do this twice. Once in the evening (normally the Internet's busy time) and once earlier in the day (or late at night). So you will collect two data files: ping1dat and ping2.dat.
- Ping invocation example (NOTE: this involves a very short path- you are to find/use a path that goes outside the US): 'ping -D 152.1.0.36> ping1.dat'
    - Specify the destination address as an IPV4 address in dotted decimal notation.
    - Specify the –D parameter so that a timestamp is appended to each sample

- Results:
  - Copy and paste the ping summary statistics to your homework submission.
  - Copy and paste the results from traceroute to your homework submission.
  - Run 'traceroute server', if that does not appear to work, try 'sudo traceroute –I server'. Cut and paste your results to your homework submission. The purpose is simply to verify that the path between the ping program endpoints appears to be outside the US.
  - Example result 'traceroute –I 152.1.0.36' (Running traceroute 152.1.0.36 does not work- just see wildcards for 64 iterations before giving up). Note- you can also ask traceroute to show the domain name of each router- this might give further clues as to if the path travels outside the US.
  - traceroute to www1.csc.ncsu.edu (152.1.0.36), 64 hops max

```
o   1   10.0.2.2  0.005ms  0.003ms  0.002ms
o   2   130.127.3.114  14.690ms  25.637ms  15.252ms
o   3   130.127.3.6  13.278ms  17.736ms  18.978ms
o   4   205.186.62.9  19.742ms  14.194ms  15.338ms
o   5   205.186.63.2  18.224ms  15.144ms  31.125ms
o   6   143.215.193.2  17.234ms  15.278ms  16.049ms
o   7   198.71.46.170  22.189ms  38.032ms  22.835ms
o   8   128.109.9.217  25.090ms  24.523ms  26.389ms
o   9   128.109.18.110  26.197ms  24.992ms  30.577ms
o   10   152.1.6.253  27.822ms  28.429ms  27.553ms
o   11   *  *  *
o   12   *  *  *
o   13   152.1.0.36  43.842ms  39.613ms  32.689ms
```

  - We can interpret the traceroute result as follows
      - There are at least 13 hops between my VM and the server.
      - Each traceroute line shows the response from the next hop router. By default, traceroute contacts each router (hop by hop) three times. The three times in each line represent the RTT from the three contact attempts with the router.
      - We do see an increase in the avg RTT as we move further away from the client. Not that sometimes we might see a hop reflect a larger RTT than the next hop router. This usually is due to the fact that

some routers handle packets that require special processing (such as when the traceroute client contacts the router) on a low priority basis. Routers typically give priority to forwarding traffic and then respond to IP control/signaling messages such as ICMP or IP options at a lower priority.

- The asterisks at hops 11 and 12 are probably due to a firewall rule.

Question 2.2. (20 points) Plot the probability distribution of each data set.   There are different ways to do this. You do not need to submit the scripts or tools you used- just document in your HW submission how you did this. Your submission should show each distribution separately.   One method is:

- Step 1: Using a script/program, process each ping output data file (ping1.dat and ping2.dat) and create two files that contain two fields: the timestamp and the RTT. Let's refer to these files as RTT1.dat and RTT2.dat.   (UPDATE: If you want, you can create a single column data file that contains just the RTT sample).
- Step 2: We want to plot the distribution of the RTT from each of the reduced data files. You can do this with excel, matlab, R, ….
    - The plotDataPDF.m matlab program will work (link is on our web page). Make sure the data file is in the same directory as the matlab script.   You will need to edit the matlab file to load the correct data file and to adjust if the data is one or two columns. In a matlab command window"
        - plotDataPDF(1,500000,.5)
        - The parameters are: plotmode: value 1 is to plot PDF, maxXValue sets the max RTT sample to 500000 microseconds, maxY sets scale of Y axis to [0,0.50]
- Results: Include the two plots in your HW submission. Very briefly, interpret the results. In your opinion, what conclusions can you draw from each figure? As best you can, explain any significant differences between the two data sets (based on the visualizations)

Section 3 Sockets Programming (30 points)

Please get familiar with the solution I posted to Exercise 2 Question 3.   The state of the program is as follows:

- It accepts new parameters including messageSize and numberIterations
- It will send and receive a message that includes messageSize number of bytes for numberIterations number of times. If numberIterations is 0, the client runs forever.
- A CNT-C will cause the client to display summary stats and terminate
- The server loops forever, simply echoes back any message it receives.
- When the server receives a CNT-C, it displays summary stats and terminates.

Currently the client and server exchange a 'dummy' message. A CNT-C issues a SIGINT to the client (and the server) programs which handles it by computing and displaying final summary stats.

```
Client                    Server
 ---------------------→
←------------------------
(delay)
---------------------→
←---------------------
```

You are to add the following capabilities to the solution provided for exercise 2.

- Validate/test the starting code and fix any errors you find. If you find anything strange, let me know ASAP so I can announce the issue to the class.
- Each iteration, the client should obtain an RTT sample. It should display the result as follows:
    - WallclockTime RTTsample totalSent totalReceived totalTimeouts
- The client should add a message header to each message:

```
        typedef struct {
            uint32_t sequenceNum;
            uint32_t timeSentSeconds;
            uint32_t timeSentNanoSeconds;
          } messageHeaderDefault;
```

- The minimum messageSize must be 12 bytes (large enough to hold the 3 uint32_t's. Please ensure the maximum messageSize is <= 5028 bytes.
- The time fields in the message structure should be based on a call to clock_gettime specifying CLOCK_REALTIME that should occur just before the client's sendto().
- The client should maintain a sequenceNumber (defined as uint32_t). The very first message, the sequence number that should be contained in the message

header is 1. The sequenceNumber is incremented by 1 with each new transmission.

- The client's RTT calculation should involve obtaining a time stamp before the sendto and after a valid recvfrom. The timestamps should be obtained using clock_gettime specifying the clock id of CLOCK_MONOTONIC_RAW.
- All time that is computed and displayed should be to nanosecond precision.
- When the client receives the echoed message it should interpret the received message header and ensure the sequence number is correct. It the sequence number received differs from the expected, the client should increment a counter numberOfSequenceNumberGaps. When computing the RTT you can chose to use the sending timestamp that is in the received message header or you can keep a variable that stores the timestamp prior to the transmission.
- The client should start a 2 second timeout with each transmission. If it pops the client will return from the recvfrom with an error – the code should ignore the error and move to the next iteration. The client's alarm handler should simply increment the numberOfTimeouts counter.
- When the client is issued a CNT-C, it should compute summary statistics. It should output a line to standard out that holds the following 7 fields of information. The times should be in units of seconds with nanosecond precision.

  WallclockTime averageRTT numberTransmissions
  numberReceivedReplies numberOfTimeouts
  numberOfSequenceNumberGaps averageLossRate

- The server does not need to look at or interpret the message header that it receives. It simply echoes the entire message.
- The server runs forever or until it receives a CNT-C from the user. If a CNT-C is issued the server should print out a line that contains the following information and then terminate:

  WallclockTime numberOfArrivals

The server parameters:

- Server service information-either a port number or a well known service name.

The client parameters:

- Server name – it can be a domain name, an IPV4 address in dotted quad format, or an IPV6 address in hex format.
- Server port : specifies the server port
- Iteration delay: Number of microseconds between samples. A 0 is allowed – this would be equivalent to ping's adaptive mode (ping -A destinationAddress, try it!!!)
- Message size: specifies the number of application bytes to place in each message. In order to ensure a message fits in one IP packet, you should assume a message size can not exceed 1472 bytes.
- Number iterations: specifies the number of iterations (i.e., data samples) to perform (or obtain). A 0 implies run forever (or until a CNT-C is entered).