Please type in your answers and turn in your homework before 11:59pm on Monday, April 22, 2019. Homework must be done individually.

1 Understanding TCP retransmission timers (10 pts)

TCP computes an average round-trip time (RTT) for a connection using an exponential weighted moving average (EWMA) estimator:

$$srtt_n = \alpha \cdot RTT_n + (1-\alpha) \cdot srtt_{n-1} \tag{1}$$

$$rttvar_n = \beta \cdot (|RTT_n - srtt_{n-1}|) + (1-\beta) \cdot rttvar_{n-1} \tag{2}$$

$$RTO_n = srtt_n + \gamma \cdot rttvar_n \tag{3}$$

where RTT_n is the nth RTT measurement, $srtt_n$ is the nth estimate of RTT, $rttvar_n$ is the nth estimate of RTT variance, and RTO_n is the nth retransmission timeout value. In a typical TCP implementation, the parameters are set to: $\alpha = 1/4$, $\beta = 1/8$, and $\gamma = 4$.

Ben Bitdiddle considers this design a bit convoluted, and proposes to use a simple arithmetic average to estimate the RTT over a fixed number of past samples. That is,

$$srtt_n = \frac{\sum_{i=n-k}^{n} RTT(i)}{k+1}; k \geq 0 \tag{4}$$

$$RTO_n = \mu \cdot srtt_n \tag{5}$$

where $k = 99$, and $\mu = 2$. Suppose that at time before $n = 0$, a TCP's RTT time is a constant 10ms. Both RTT estimates in EQ (1) and (4) converge to 10ms. At $n = 0$, the RTT of the TCP connection experiences a sudden change, and goes up to 1 second, and does not change afterwards. Answer the following questions:

1. (2 pts) With TCP's RTO estimate, what's the RTO at time when $n = 1$?
2. (2 pts) With Ben's RTO estimate, what's the RTO at time when $n = 1$?
3. (3 pts) Will TCP suffer spurious (premature) retransmissions? If no, explain why. If yes, compute when the RTO estimate will be sufficiently accurate to prevent spurious retransmission. (Hint: you may want to write a small script to do this.)
4. (3 pts) Will Ben's TCP suffer spurious (premature) retransmission? If no, explain why. If yes, compute when the RTO estimate will be sufficiently accurate to prevent spurious retransmission.

2 Go deep with Wireshark (20 pts)

This problem helps you understand the TCP congestion control algorithm. Wireshark `http://www.wireshark.org/` is a useful tool for network traffic analysis. In this problem, you will use Wireshark to look at packet traces from a TCP connection, and answer a few questions. First, download the tcp trace from this link `http://www2.cs.duke.edu/courses/spring19/compsci356/hws/tcp-trace`. Then start Wireshark using the command "wireshark tcp-trace". You may install wireshark on your own machine, or use the one we provided in the virtual machine image used for all our labs. Answer the following questions.

1. (1 pt) Click on the first packet. The window below will show you the packet details. What protocol headers are included in this packet?
2. (1 pt) What TCP flags are set in the first packet?
3. (1 pt) What TCP options are included in the first packet?
4. (1 pt) Which host does the active open?

5. (1 pt) Go to packet 91. Why is a Dup ACK sent?
6. (1 pt) Why does the sender send packet 96?
7. (1 pt) Go to packet 119. What might cause the sender to send this packet?
8. (1 pt) Go to packet 425. Which host does an active close?
9. (1 pt) Go to the "Statistics" menu, and follow the sub-menu "TCP Stream Graph." Click the "Time-Sequence Graph (tcptrace)." Zoom this graph so that you can see 1-second time ticks. From this graph, can you tell how many times the TCP sender does slow start, and when each slow start phase finishes?
10. (1 pt) What (duplicate acks or timeout) is likely to trigger the retransmission around time 2.5s?
11. (1 pt) What is likely to limit the sender's sending window size at time 2s (cwnd, or receiver advertise window)?
12. (1 pt) How many packets are likely to be lost between time 2s and 7s? Circle the correct answer.
    A. One B. Two C. Three D. > Three
13. (1 pt) Around time 9.5s, another retransmission occurs. What is likely to trigger the retransmission?
14. (1 pt) When does the sender receive the ACK for the packet retransmitted at around time 9.5?
15. (1 pt) What limits the sender's sending window size at time 12s?
16. (1 pt) Which congestion mode is the sender in at time 13s?
17. (1 pt) Can you estimate the sender's congestion window size at time 14s?
18. (2 pt) Click the "Round Trip Time Graph." Can you explain why the measured RTT values fluctuate so much over time from what you learned from the Time-Sequence graph?
19. (1 pt) Similarly, Click the "Throughput Graph." Can you explain why the TCP throughput fluctuates a lot between 5-10 second, but stabilizes afterwards?

3 Understanding Buffer Bloat (10 pts)

This problem will help you understand how a router's buffer size affects TCP performance. First, please install a python library on the virtual machine image used in our labs by using the command "sudo apt-get install python-matplotlib." Then following the instructions at this link (`https://github.com/mininet/mininet/wiki/Bufferbloat`) to complete the assignment.

4 BGP in the Wild (10 pts)

The Route Views project (`http://www.routeviews.org/`) has an archive that stores periodic BGP table dumps from several routers. Each Route Views router peers with a bunch of other routers located all over the world and receives their BGP updates in real time. So we can discover a lot about Internet routing by examining the data Route Views routers collect, or login to the routers. In this problem, we will telnet to a Route Views router and learn a bunch about its internal state.

1. (2 pt) Login to the routeviews' server by typing `telnet route-views.routeviews.org 23`. The first parameter after telnet is the router's DNS name, and the second is the port we telnet into. Enter the username `rviews` as prompted.
2. (0 pt) Type `?` to see the commands supported by the router. You can also type `?` to complete a partial command.
3. (2 pt) Now run the command `show ip bgp 152.3.0.0/16`. The IP prefix 152.3.0.0/16 is Duke's network prefix. This command tells you the route advertisements the router receives from its peers. How many routes are available from this router to Duke? (The answer will be displayed at the 2nd line of the command's output.)

4. (2 pt) Now look at the first route output by the command. What's the IP address of the next hop router that advertises the route? What's the AS path from the next hop router to Duke? What's the local preference of the route?

5. (2 pt) Open another terminal, and login to linux.cs.duke.edu. Run the command `traceroute -A nextHopRouter`, where `nextHopRouter` is the IP address of the router obtained from the previous step. The `-A` option outputs the AS path information. What is the AS path from Duke to the next hop router? Is the path the reverse AS path from the next hop router to Duke? If not, explain why this may happen.

6. (2 pt) Now go back to the router login window, and scroll down until you find the line ending with the word "best". This is the best route. What's the next hop router that advertises this best route? What's the best AS path? Can you explain why this route is chosen as the best path?

5 Digging DNS for fun (10 pts)

Learn how to use dig by typing "man dig" on a Unix machine. Then use dig to figure out the DNS server hierarchy that leads to the domain name `www.cnn.com.`. Turn in each dig commands you used and their outputs. Is the website `www.cnn.com` served by a CDN? Explain why.