Checkpoint 4: measuring the real world

Due: Sunday, October 26, 11:59 p.m.

0 Collaboration Policy

Collaboration Policy: Same as checkpoint 0. Please fully disclose any collaborators or any gray areas in your writeup—disclosure is the best policy.

1 Overview

By this point in the class, you've implemented the Transmission Control Protocol in an almost fully standards-compliant manner. TCP implementations are arguably the world's single most popular computer program, found in billions of devices. Most implementations use a different strategy from yours, but because all TCP implementations share a common **language**, they are all interoperable—every TCP implementation can be a peer with any other, across the whole Internet. This checkpoint won't use your TCP implementation: it's about **measuring** the long-term statistics of some real-world Internet paths.

To complete this checkpoint, we want you to choose and characterize at least **three** "interesting" Internet paths. Each path will be between one computer (e.g. your laptop, or a myth or cardinal machine) and another host. To make the path "interesting", it will either (1) go some significant distance (RTT greater than 100 ms) or will (2) include at least one "interesting" link along the way (e.g. a Wi-Fi or tethered cellular or satellite link). Ideally your final report will include a mix—at least one long-distance path (ideally without any wireless links) and at least one path that includes an "interesting" link.

For each path, please measure at least the below statistics:

2 Collecting data

1. Choose a remote host on the Internet to ping (measured by ping from your computer or VM). Some possibilities of faraway paths to get there:
    - `www.cs.ox.ac.uk` (Oxford University CS department webserver, United Kingdom)
    - `162.105.253.58` (Computer Center of Peking University, China)
    - `www.canterbury.ac.nz` (University of Canterbury webserver, New Zealand)
    - `41.186.255.86` (MTN Rwanda)
    - A friend's VM on the CS144 private network
    - *preferred:* an original choice with an RTT of at least 100 ms from you **or** that requires crossing a wireless link

2. Use the `mtr` or `traceroute` commands to trace the route between your VM and this host.
3. Run a `ping` **for at least an hour** to collect data on this Internet path. Use a command like `ping -D -n -i 0.2 hostname | tee data.txt` to save the data in the "data.txt" file. (The `-D` argument makes ping record the timestamp of every line, and `-i 0.2` makes it send one "echo request" ICMP message every 0.2 seconds. The `-n` argument makes it skip trying to use DNS to reverse-lookup the replying IP address to a hostname.)
4. Note: A default-sized ping every 0.2 seconds is fine, but please do not flood anybody with traffic faster than this for more than a few seconds.

3 Analyzing data

If you sent five pings per second for an hour, you will have sent approximately 3,600 echo requests ($= 5 \times 3600$), of which we expect the vast majority to have received a reply in the ping output. Using the programming language and graphing tools of your choice, please compute and graph at least the following information:

1. What was the overall delivery rate over the entire interval? In other words: how many echo replies were received, divided by how many echo requests were sent? (Note: ping on GNU/Linux doesn't print any message about echo replies that are **not** received. You'll have to identify missing replies by looking for missing sequence numbers.)
2. What was the longest consecutive string of successful pings (all replied-to in a row)?
3. What was the longest burst of losses (all **not** replied-to in a row)?
4. Can you figure out where the path went (geographically) from the names in the results of `mtr` or `traceroute`? Did it follow a path you expected or didn't expect?
5. Produce a graph showing the autocorrelation of "packet loss" over time. In other words:
   - Given that echo request #N received a reply, what is the probability that echo request #(N+k) was also successfully replied-to? In your graph, include a bar for each k between -10 and 10 (inclusive).
   - Given that echo request #N did **not** receive a reply, what is the probability that echo request #(N+k) was also not replied-to?
   - How do these figures (the **conditional** delivery rates) compare with the overall "unconditional" packet delivery rate in the first question? How independent or "bursty" were the losses?
6. What was the minimum RTT seen over the entire interval? (This is probably a reasonable approximation of the true MinRTT...)

7. What was the maximum RTT seen over the entire interval?

8. Make a graph of the RTT as a function of time. Label the x-axis with the actual time of day (covering the hour+ period), and the y-axis should be the number of milliseconds of RTT.

9. Graph the Cumulative Distribution Function of the distribution of RTTs observed. This is a graph where the x-axis is each observed value of RTT, and the y-axis is the proportion of samples that were less than or equal to this number (so the y-axis will go from 0 to 1). What rough shape is the distribution?

10. Make a scatter plot of the correlation between "RTT of ping #N" and "RTT of ping #N+1". The x-axis should be the number of milliseconds from the first RTT, and the y-axis should be the number of milliseconds from the second RTT. How correlated is the RTT over time?

11. Do some **brief** (less than 10 seconds) experiments where you send a higher data rate of pings, by increasing the packet size and frequency. On Linux, you can increase the **size** of an echo request (and reply) with the "-s" argument (do not use values greater than 1400), and you can increase the **frequency** of an echo request by reducing the interval given to the "-i" argument. You can tell ping to stop after a certain number of requests with the "-c" argument. Make a graph of the overall data rate of replies ($\frac{\text{packet size} \times \text{number of replies}}{\text{total duration}}$) as a function of the data rate of your echo requests. Does it level off at some value? What was the maximum throughput you were able to obtain?

12. What are your conclusions from the data? Did the network path behave the way you were expecting? What (if anything) surprised you from looking at the graphs and summary statistics?

13. What interesting comparisons can you make between this path and the other network paths you characterized?

Please submit your report as a PDF via Gradescope.