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

检查点4：测量真实世界

截止日期：10月26日星期日，晚上11:59

0 协作政策

协作政策：与检查点0相同。请充分披露任何协作者或你在报告中存在的任何灰色地带——公开透明是最好的政策。

1 概述

到课程这个阶段，你已经以几乎完全符合标准的方式实现了传输控制协议。TCP 实现可以说是世界上最受欢迎的计算机程序，存在于数十亿设备中。大多数实现使用与你不同的策略，但由于所有 TCP 实现都共享一种通用语言，它们都是互操作的——任何 TCP 实现都可以与任何其他实现成为对等体，遍及整个互联网。这个检查点不会使用你的 TCP 实现：它关于测量一些真实世界互联网路径的长期统计数据。

要完成这个检查点，我们希望你能选择并描述至少三条“有趣的”互联网路径。每条路径将连接一台计算机（例如你的笔记本电脑，或一个 myth 或 cardinal 机器）和另一个主机。为了让路径“有趣”，它将要么 (1) 跨越相当长的距离（往返时间大于 100 毫秒），要么将 (2) 在途中包含至少一条“有趣的”链路（例如一个 Wi-Fi 或连接的蜂窝网络或卫星链路）。理想情况下，你的最终报告将包含一个混合——至少一条长距离路径（理想情况下没有任何无线链路）和至少一条包含“有趣的”链路的路径。

对于每条路径，请至少测量以下统计数据：

2 收集数据

1. 选择一个互联网上的远程主机进行ping测试（可通过您的计算机或虚拟机进行ping测试）。一些可能的远距离路径：
   - `www.cs.ox.ac.uk` (牛津大学计算机系服务器，英国)
   - `162.105.253.58` (北京大学计算机中心，中国)
   - `www.canterbury.ac.nz` (坎特伯雷大学服务器，新西兰)
   - `41.186.255.86` (卢旺达MTN)
   - 一位朋友在CS144私有网络上的虚拟机
   - 优先：一个原始选择，其往返时间至少为100毫秒（从你那里测量）或需要跨越无线链路

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...)

2. 使用 `mtr` 或 `traceroute` 命令来追踪您的虚拟机与该主机的路由。

3. 运行 `ping` 至少一小时以收集该互联网路径的数据。使用类似 `ping -D -n -i 0.2 hostname | tee data.txt` 的命令将数据保存到"data.txt"文件中。（`-D` 参数使 ping 记录每行的时间戳，`-i 0.2` 使其每0.2秒发送一次"回显请求"ICMP消息。`-n` 参数使其跳过尝试使用DNS将回复的IP地址反向解析为主机名。）

4. 注意：每0.2秒发送一次默认大小的 ping 是可以的，但请不要在几秒钟内向任何人发送比这更快的流量。

3 分析数据

如果你每秒发送五个回显请求，持续一小时，你将发送大约 3,600 个回显请求 ($= 5 \times 3600$)，其中我们预计绝大多数都会在 ping 输出中收到回复。请使用你选择的编程语言和绘图工具，计算并绘制至少以下信息：

1. 整个时间间隔内的整体交付率是多少？换句话说：收到的回显回复数量除以发送的回显请求数量是多少？（注意：在GNU/Linux上执行ping时，不会打印未收到的回显回复的任何消息。您需要通过查找缺失的序列号来识别缺失的回复。）

2. 最长的连续成功ping字符串是多少（所有回复都在一行中）？

3. 最长的连续丢包突发是多少（所有未回复都在一行中）？

4. 能否根据{v1}或{v2}结果中的名称推断出路径的地理位置？
   它是否遵循了您预期或未预期的路径？

5. 生成一个图表，展示"丢包率"随时间的变化的自相关情况。换句话说：
   - 已知回显请求#N收到了回复，那么回显请求 #(N+k) 成功回复的概率是多少？在您的图表中，为-10到10（含）之间的每个k包含一个条形图。
   - 既然回显请求 #N 没有收到回复，那么回显请求 #(N+k) 没有收到回复的概率是多少？
   - 这些数据（条件交付率）与第一题中的整体"无条件"数据包交付率相比如何？损失是独立还是"突发"的？

6. 在整个时间间隔内观察到的最小往返时间（RTT）是多少？（这可能是真实最小往返时间 trueMinRTT 的一个合理近似...）

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.

7. 整个时间间隔内观察到的最大往返时间(RTT)是多少?

8. 绘制RTT随时间变化的图表。将x轴标记为实际时间(覆盖小时+ 段)，y轴应为往返时间的毫秒数。

9. 绘制观察到的往返时间(RTT)分布的累积分布函数(CDF)图表。这是一个x轴为每个观察到的RTT值，y轴为小于或等于该数值的样本比例(因此y轴将从0到1)的图表。分布的形状大致如何?

10. 绘制“第N次ping的RTT”与“第N+1次ping的RTT”之间的相关性散点图。x轴应为从第一个RTT开始经过的毫秒数，y轴应为从第二个RTT开始经过的毫秒数。RTT随时间的相关性如何?

11. 进行一些简短的（少于10秒）实验，通过增加数据包大小和频率来发送更高的数据速率的ping。在Linux中，您可以使用“-s”参数增加回显请求（和回复）的大小（不要使用大于1400的值），并通过减少分配给“-i”参数的间隔来增加回显请求的频率。您可以使用“-c”参数指示ping在收到一定数量的请求后停止。绘制回复的整体数据速率（packet 大小×回复总数总持续时间）作为回显请求数据速率的函数的图表。它是否在某个值上趋于平稳? 您能够获得的最大吞吐量是多少?

12. 您从数据中得出的结论是什么? 网络路径是否按您的预期运行? 从查看图表和摘要统计数据中，您是否有什么（或没有）感到惊讶的事情?

13. 您可以在这条路径和您表征的其他网络路径之间进行哪些有趣的比较?

请通过 Gradescope 提交您的报告（PDF 格式）。