15-441/641: Networking and the Internet
Project 0: Network Performance and Tools

TAs: Nirav Atre, Zili Meng, Hugo Sadok

Last update: September 2, 2022

1 Introduction: A Tale of Two Tools

Performance is an important consideration for many systems we encounter in the real world. As the world's largest and most complex distributed system, the Internet is no different! In class, you learned about two important metrics used to characterize network performance: *round-trip time* (RTT) and *throughput*. In this introductory project, you will gain hands-on experience with reasoning about network performance in the context of these metrics. The networking community has developed a rich set of tools to make this task easier, two of which you will use here: `ping` and `iperf`.

1. `ping` is a network utility commonly used to estimate the latency to different endpoints on the Internet.[1] Given a hostname or Internet Protocol (IP) address, `ping` uses Internet Control Message Protocol (ICMP) messages to measure the RTT between your machine and the one you're `ping`-ing, producing output that looks like this:

```
$ ping www.google.com
PING www.google.com (142.250.191.132) 56(84) bytes of data.
64 bytes from ord38s29-in-f4.1e100.net (142.250.191.132):  icmp_seq=1 ttl=114 time=205 ms
64 bytes from ord38s29-in-f4.1e100.net (142.250.191.132):  icmp_seq=2 ttl=114 time=146 ms
64 bytes from ord38s29-in-f4.1e100.net (142.250.191.132):  icmp_seq=3 ttl=114 time=251 ms
64 bytes from ord38s29-in-f4.1e100.net (142.250.191.132):  icmp_seq=4 ttl=114 time=275 ms

--- www.google.com ping statistics ---
5 packets transmitted, 4 received, 20% packet loss, time 4003ms
rtt min/avg/max/mdev = 146.493/219.334/274.660/49.018 ms
```

2. `iperf` is a userspace program used to estimate traffic throughput (using either TCP or UDP) between two machines running `iperf3` instances, producing output that looks like this:

   Server:

```
$ iperf3 -s
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
Accepted connection from 172.26.10.128, port 65475
[ 5] local 128.2.208.106 port 5201 connected to 172.26.10.128 port 65476
[ ID] Interval Transfer Bandwidth
[ 5] 0.00-1.00 sec 10.1 MBytes 85.1 Mbits/sec
[ 5] 1.00-2.00 sec 12.8 MBytes 107 Mbits/sec
[ 5] 2.00-3.00 sec 9.50 MBytes 79.7 Mbits/sec
[ 5] 3.00-4.00 sec 11.4 MBytes 95.8 Mbits/sec
[ 5] 4.00-5.00 sec 12.2 MBytes 103 Mbits/sec
[ 5] 5.00-6.00 sec 12.0 MBytes 100 Mbits/sec
```

[1] You may be familiar with the more traditional use for `ping`: checking whether nodes are responsive at all.

```
[ 5] 6.00-7.00 sec 8.60 MBytes 72.2 Mbits/sec
[ 5] 7.00-8.00 sec 11.8 MBytes 98.6 Mbits/sec
[ 5] 8.00-9.00 sec 10.1 MBytes 84.5 Mbits/sec
[ 5] 9.00-10.00 sec 9.94 MBytes 83.4 Mbits/sec
[ 5] 10.00-10.08 sec 1.05 MBytes 108 Mbits/sec
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval Transfer Bandwidth
[ 5] 0.00-10.08 sec 0.00 Bytes 0.00 bits/sec sender
[ 5] 0.00-10.08 sec 109 MBytes 91.1 Mbits/sec receiver
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
```

Client:

```
$ iperf3 -c <server-ip> -t 10 -i 2 -u -b 0
Connecting to host 128.2.208.106, port 5201
[ 5] local 172.26.10.128 port 65476 connected to 128.2.208.106 port 5201
[ ID] Interval Transfer Bitrate
[ 5] 0.00-2.00 sec 26.4 MBytes 111 Mbits/sec
[ 5] 2.00-4.00 sec 19.7 MBytes 82.8 Mbits/sec
[ 5] 4.00-6.00 sec 23.5 MBytes 98.3 Mbits/sec
[ 5] 6.00-8.01 sec 21.5 MBytes 89.9 Mbits/sec
[ 5] 8.01-10.00 sec 20.0 MBytes 84.2 Mbits/sec
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval Transfer Bitrate
[ 5] 0.00-10.00 sec 111 MBytes 93.2 Mbits/sec sender
[ 5] 0.00-10.00 sec 109 MBytes 91.8 Mbits/sec receiver

iperf Done.
```

An interesting complication that arises in large systems like the Internet is the sheer physical distance between communicating endpoints. To understand the performance implications of physical separation, you will run experiments with cloud-based Virtual Machines (VMs) in geographically different regions of the world. For this project, we will use Amazon's EC2 cluster. By the end of this project, you will gain familiarity with launching and managing cloud-based VM instances, making educated hypotheses about network performance, and using tools like `ping` and `iperf` to validate your hypotheses.

Note: This is a short, easy project that won't take you long to finish. However, one or more bonus subparts involve making long-running measurements (over 24 hours) that won't be humanly possible to finish the night before `:)` If you want to attempt the bonus points (you should, they're easy!), please start early.

2 Performance Measurement

> "Alright," said Deep Thought. "The Answer to the Great Question... Of Life, the Universe and Everything... Is... Forty-two" said Deep Thought, with infinite majesty and calm. "Forty-two!" yelled Loonquawl. "Is that all you've got to show for seven and a half million years' work?" "I checked it very thoroughly," said the computer, "and that quite definitely is the answer. I think the problem, to be quite honest with you, is that you've never actually known what the question is."
>
> Douglas Adams, *The Hitchhiker's Guide to the Galaxy*

2.1 Hypotheses

In this and subsequent projects, we will ask you to develop hypotheses (i.e., make *educated guesses*) as a first step towards answering various experimental questions about networked systems. Hypotheses are an important part of the scientific process because they help us *ask the right questions* and *put our experimental findings into context*. For each of the following questions, develop a plausible hypothesis about the experimental outcome based on what you have learned in class; don't run any experiments yet!

Given an arbitrary pair of nodes A and B on the Internet:

1. If the throughput from A to B is X Megabits/second (Mbps), would you expect the throughput from B to A to also be X Mbps? Why or why not?

2. If the RTT experienced by A going to B is Y milliseconds (ms), would you expect the RTT experienced by B going to A to also be Y ms? Why or why not?

3. Consider two scenarios. In Scenario 1, A is transmitting to B as fast as possible. In Scenario 2, A is not transmitting to B. B uses a ping to measure the RTT to A. Do you expect the measured RTT to be the same or different between these two scenarios? How so, and why?

2.2 Experiments

Next, we will perform a series of experiments to answer (empirically) our original questions. Create a set of 3 VMs on the AWS cloud, making sure that each VM is in a different AWS region. For each *directed pair* of VMs (i.e., A to B is distinct from B to A, so 6 pairs in total), perform the following exercises:

1. Measure RTT from each VM to every other VM using `ping`.[2] Next, create a bar plot with the *Average RTT* in milliseconds on the Y-axis, and each directed pair on the X-axis. You should further group the bars in the plot so that pairs containing the same set of VMs are adjacent to each other (e.g., A/B is next to B/A).

2. Measure UDP throughput from each VM to every other VM using `iperf`. Next, create a bar plot with the *Average Throughput* on the Y-axis, and each directed pair on the X-axis. You should further group the bars in the plot so that pairs containing the same set of VMs are adjacent to each other (e.g., A/B is next to B/A).

3. Measure RTT from each VM to every other VM using `ping` *while* `iperf` sends traffic in the reverse direction (e.g., if A is `ping`-ing B, then B should be sending background traffic to A using `iperf`). Next, create a bar plot with the *RTT* on the Y-axis, and each directed pair on the X-axis. You should further group the bars in the plot so that pairs containing the same set of VMs are adjacent to each other (e.g., A/B is next to B/A).

2.3 Inferences

For each of the three questions posed in §2.1, state briefly (in 3-4 sentences) whether the experimental evidence either corroborates or disproves your original hypothesis, and why. If your hypothesis was wrong, propose an alternative hypothesis that might explain the results you observed.[3]

3 Bonus

In this section, we will extend our understanding of the relationship between network latency and geographic separation. This part is optional (for bonus points), but it's fun, easy, and instructive, so we highly encourage you to attempt it!

[2] You should gather several samples to account for natural variance in packet transmission.

[3] Ordinarily, the scientific method would require us to repeat all of the above steps to validate a new hypothesis, but we'll skip this for the sake of time.

3.1 RTT and Geo-distance

Important: We recommend performing the following experiment by SSH'ing into an Andrew lab machine. You may use your own machine, but you must ensure that you're not connected to a Virtual Private Network (VPN) server in a different geographic location than your machine.

1. Collate a list of 10-15 domain names (the more the better!). This list should contain a few websites that you visit often (e.g., www.google.com or ~~www.netflix.com~~ www.cs.cmu.edu), as well as several non-regional ones (e.g., www.casa-museumedeirosealmeida.pt belongs to a popular Musuem in Lisbon, Portugal). If you don't know any foreign domains, get creative!
2. Use an IP-to-geolocation service (e.g., https://tools.keycdn.com/geo) to find the (approximate) geographic coordinates of each domain in list.[4]
3. Use the same service to find your own geographic coordinates (once again, beware of VPNs!), and use a geo-distance service (e.g., https://www.nhc.noaa.gov/gccalc.shtml) to compute the pairwise geographic distance in miles between yourself and each domain in your list.
4. Create a scatter-plot with the *Average RTT* in milliseconds on the Y-axis, and the geo-distance on the X-axis. Next, fit a linear function to it.

Using the scatter-plot and linear fit you generated, answer the following questions:

(a) Are RTT and geo-distance correlated positively, negatively, or not at all? If applicable, also comment on the strength of correlation (weak vs. strong).

(b) Why do you think you observe this trend or lack thereof?

3.2 RTT and Time-of-day

Measure the RTT between your (or an Andrew lab) machine and www.google.com over a period of 24 hours, and create a line plot with the *RTT* on the Y-axis, and *Time of day* on the X-axis. Do you observe any interesting trends here? Why?

[4] You might notice a few websites which you expected to be non-regional to have coordinates in/around Pittsburgh. This is because they're using Content Distribution Networks (CDNs) to cater to users located far from their own servers. You'll hear more about CDNs in class!

A Resources

You may find the following resources to be helpful:

- `ping`[1]
- `iperf`[2]
- man pages
  - Installed locally (e.g., `$ man ping`)
  - Available online: The Single Unix Specification[3]

References

[1] https://en.wikipedia.org/wiki/Ping_(networking_utility)

[2] https://iperf.fr/

[3] http://www.opengroup.org/onlinepubs/007908799/