Bufferbloat on Network Emulator (Mininet)

Assignment Due: Aug, 25th 2023

This document contains all the instructions necessary for completing the assignment, including setting up the simulation environment. Some sections may be unnecessary depending on students' experience. Yet, it is highly recommended that the document is read in full before attempting anything. This minimizes the chances for missing hints that may be helpful while finding the solutions.

Bufferbloat

In this exercise we will study the dynamics of TCP in home networks. Take a look at the figure below which shows a "typical" home network with a Home Router connected to an end host. The Home Router is connected via DSL/GPON to a Headend router at the Internet access provider's office. We are going to study what happens when we download data from a remote server to the End Host in this home network.

In a real network it's hard to measure cwnd (because it's private to the server) and the buffer occupancy (because it's private to the router). To ease our measurement, we are going to emulate the network in Mininet.

Goals

- Learn the dynamics of TCP sawtooth and router buffer occupancy in a network.
- Learn why large router buffers can lead to poor performance which is often called "**bufferbloat**."
- Learn how to use Mininet to create network topologies, run traffic generators, collect statistics and plot them.
- Learn how to package your experiments so it's easy for others to run your code.
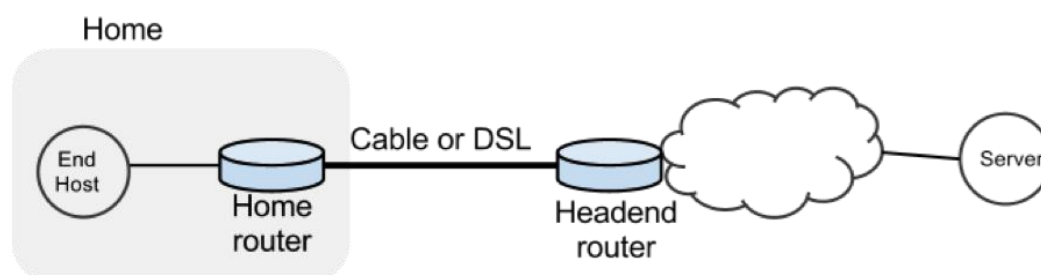
Tasks

Within Mininet, create the following topology. Here h1 is a remote server (ie google.com) on the Internet that has a fast connection (1Gb/s) to your home router with a slow downlink connection (10Mb/s). The round-trip propagation delay, or the minimum RTT between h1 and h2 is 6ms. The router buffer size can hold 100 full sized ethernet frames (about 150kB with an MTU of 1500 bytes).

Then do the following:

- Start a long lived TCP flow sending data from h1 to h2. Use iperf.
- Send pings from h1 to h2 10 times a second and record the RTTs.
- Plot the time series of the following:
  - The long lived TCP flow's cwnd
  - The RTT reported by ping
  - Queue size at the bottleneck
- Spawn a webserver on h1. Periodically download the index.html web page (three times every five seconds) from h1 and measure how long it takes to fetch it (on average). The starter code has some hints on how to do this. Make sure that the webpage download data is going in the same direction as the long-lived flow.
- The long lived flow, ping train, and web server downloads should all be happening simultaneously.

  Repeat the above experiment and replot all three graphs with a smaller router buffer size (Q=20 packets).

Minimum RTT: 4ms
Queue: 150kB (100 packets)
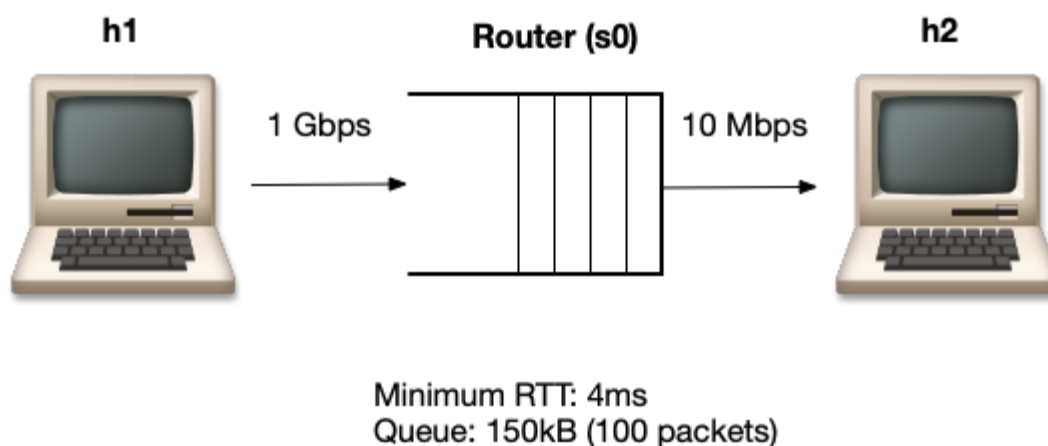
Setting up the Emulation Environment

Mininet is a network emulator which creates a virtual network on your computer that communicates actual packets between virtually independent nodes. This allows researchers to test different scenarios without having access to a large (possibly expensive) network.

You can check the Mininet web page (http://mininet.org) and follow the instructions to install it on your own environment if requirements are satisfied. However there is an easier method. Below we describe how to setup a VM that would have all the necessary modules installed for this class. You can even use it for the second assignment of this class, and maybe even for your final project.

Setting up a VM

In order to give you a more comprehensive "Computer Networks Researcher" VM on Baadal (because you are one), we will describe setting up a VM with some useful networking research tools installed.

- Visit https://baadal.iitd.ac.in/baadal and you can login to Baadal interface using your IITD credentials.

- After login, goto *USER MENU > Request VM*,

  with template ID : Ubuntu 16.04 Server amd64 30GB (*will share updated template ID*)

- After VM is created, you will receive a mail with all the details. Baadal interface has VNC option can be access the VM desktop directly.
- By default VMs are created with a private IP; hence can be accessed from within IITD network only. User can seperately request for VPN access for connecting to the IITD LAN from outside IITD.
  To access Linux VM using assigned private IP; SSH to baadal machine using default credentials. Once successfully logged in; user is enforced to change the password.
- You can install mininet using apt install mininet.
- For more info, check https://baadal.iitd.ac.in/faq

Manual Setup:

- Visit http://mininet.org/download/ and download Mininet VM
- (mininet-2.3.0-210211-ubuntu-16.04.7-server-amd64-ovf.zip)
- Note that we are downloading Ubuntu 16.04.7 (not the most recent one)
- VM is about 1 GB, so it might take a moment to download
- Follow VM Setup Notes (http://mininet.org/vm-setup-notes/) to import the downloaded

  VM with default settings

- Before starting the VM, set up the network on Virtualbox Download/Get Started With Mininet - Mininet. Below, we list the steps for

  creating a NAT based network adapter.

    - ○ Setting>Network>Adapter 1 then select "NAT"
    - ○ Choose Advanced > Port Forwarding
    - ○ Add a rule to forward host port 2222 to guest port 22
    - ○ To SSH from the local machine to the VM, run:

      - ssh-p2222-L8000:localhost:8000mininet@localhost

- Both the username and password for the VM are mininet.
- If you encounter a "Kernel driver not installed" error, you can follow the steps at

  https://medium.com/@Aenon/mac-virtualbox-kernel-driver-error-df39e7e10cd8 to solve it

- There is a step in Mininet Walkthrough where Wireshark is tested. If it doesn't work for you don't worry, we will not use Wireshark for our assignments.
- For Python 2 : **"sudo PYTHON=python2 mininet/util/install.sh –n"**
- Since the Linux (and Python) versions of the VM are a little bit old we will need to

  upgrade some modules for our assignments. Run the following commands:

    - ○ pip install --upgrade pip==9.0.1
    - ○ pip install termcolor --user
    - ○ pip install matplotlib --user

- o ○ sudo apt-get update
- o ○ sudo apt-get install curl
- • ● Now the VM is ready for running assignment programs.

Framework code

To help you get started, we provide a basic skeleton code. Once you have your VM instance running, run a Mininet sanity check (sudo mn --test pingall) and then continue with this assignment. Your instance should have Mininet and most tools pre-installed.

Clone the starter code for this project:

git clone https://github.com/stanford-cs244/cs244-bufferbloat.git
cd cs244-bufferbloat

Look for TODOs in the starter code. The following are important files you will find in the repository. Ignore other files.

File	Purpose
bufferbloat.py	Creates the topology, measures cwnd, queue sizes and RTTs and spawns a webserver.
plot_queue.py	Plots the queue occupancy at the bottleneck router
plot_ping.py	Parses and plots the RTT reported by ping
plot_tcpprobe.py	Plots the cwnd time-series for a flow specified by its destination port
run.sh	Runs the experiment and generates all graphs in one go.

Note: We recommend you using a version control system to track changes to your assignment and back it up regularly on the "cloud." Both Github and Bitbucket host private git repositories for free. Do NOT use a public repo[1]; you will run the risk of violating the Honor Code. This applies for all the assignments; feel free to use a public repo for the final project.

Submission/Deliverables

- • **Plots**: There should be 6 plots in total, 3 each for router buffer sizes 100 and 20 packets. They MUST have the following names and be present in your submission folder

  ○ bb-q100/cwnd-iperf.png,bb-q100/q.png,bb-q100/rtt.png. ○ bb-q20/cwnd-iperf.png,bb-q20/q.png,bb-q20/rtt.png.

  ## Questions

Include your answers to the following questions in your README file. Remember to keep answers brief.

1. Why do you see a difference in webpage fetch times with short and large router buffers?
2. Bufferbloat can occur in other places such as your network interface card (NIC). Check the output of `ifconfig eth0` on your VirtualBox VM. What is the (maximum) transmit queue length on the network interface reported by ifconfig? For this queue size, if you assume the queue drains at 100Mb/s, what is the maximum time a packet might wait in the queue before it leaves the NIC?
3. How does the RTT reported by ping vary with the queue size? Describe the relation between the two.
4. Identify and describe two ways to mitigate the bufferbloat problem.
5. Describe how and why your results change when you re-run the emulation.

Notes

● When running the curl command to fetch a web page, please fetch webserver_ip_address/http/index.html.

Grading

Total: 10 pts

- ● 3 points: Producing working code that generates the graphs
- ● 5 points: Answering the questions
- ● 1 point: Producing correct graphs
- ● 1 point: Design and code quality

Mininet tutorial

The "Introduction to Mininet" tutorial on github has a number of examples to help you with common tasks you will encounter in this assignment: creating topologies, spawning programs (ping, iperf, etc.) and collecting output.

https://github.com/mininet/mininet/wiki/Introduction-to-Mininet

Useful Hints

To look at your output, `run.sh` script starts a web server in the `cs244-bufferbloat` directory using the command below. Then you can point a web browser to `<vm-ip>:8000` and browse your results.

```
python -m SimpleHTTPServer.
```

If your Mininet script does not exit cleanly due to an error (or if you pressed Control-C), you may want to issue a clean command `sudo mn -c` before you start Mininet again. Take a look at the `run.sh` script to see how it does that.