Handout #17

CSC458/2209
Computer Networks — Fall 2019

Programming Assignment 2: Bufferbloat

MohammadReza Ebrahimi

Bufferbloat

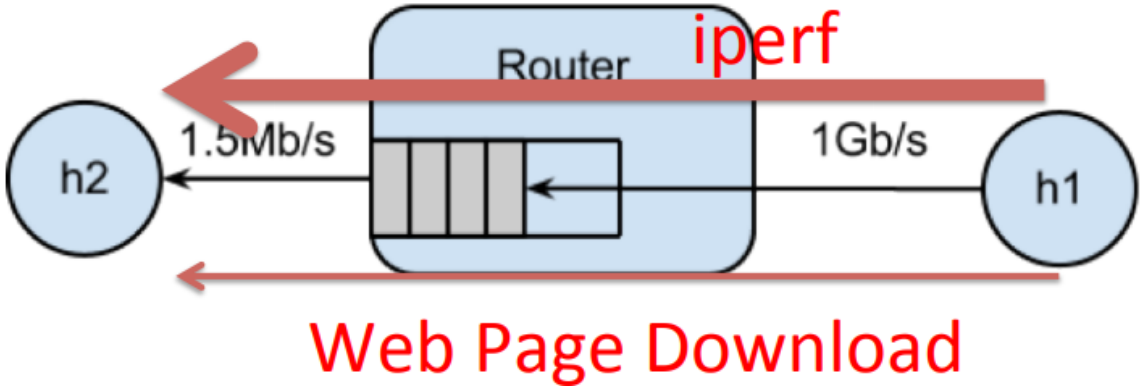
- Bufferbloat is a cause of high latency caused by **large buffering of packets** in bottleneck nodes of a network.
- Bufferbloat can also cause **packet delay variation** (also known as jitter), as well as reduce the overall network throughput.

- Oversized buffers can have a damaging effect!

- Bufferbloat test

Bufferbloat simple example

1. The host should be pinged continuously.
2. A several-seconds-long download from the host should be started and stopped a few times.
3. TCP congestion avoidance algorithm will rapidly fill up the bottleneck on the route.
4. If increase of the RTT is reported by ping, then it demonstrates that the buffer of the current bottleneck is bloated.

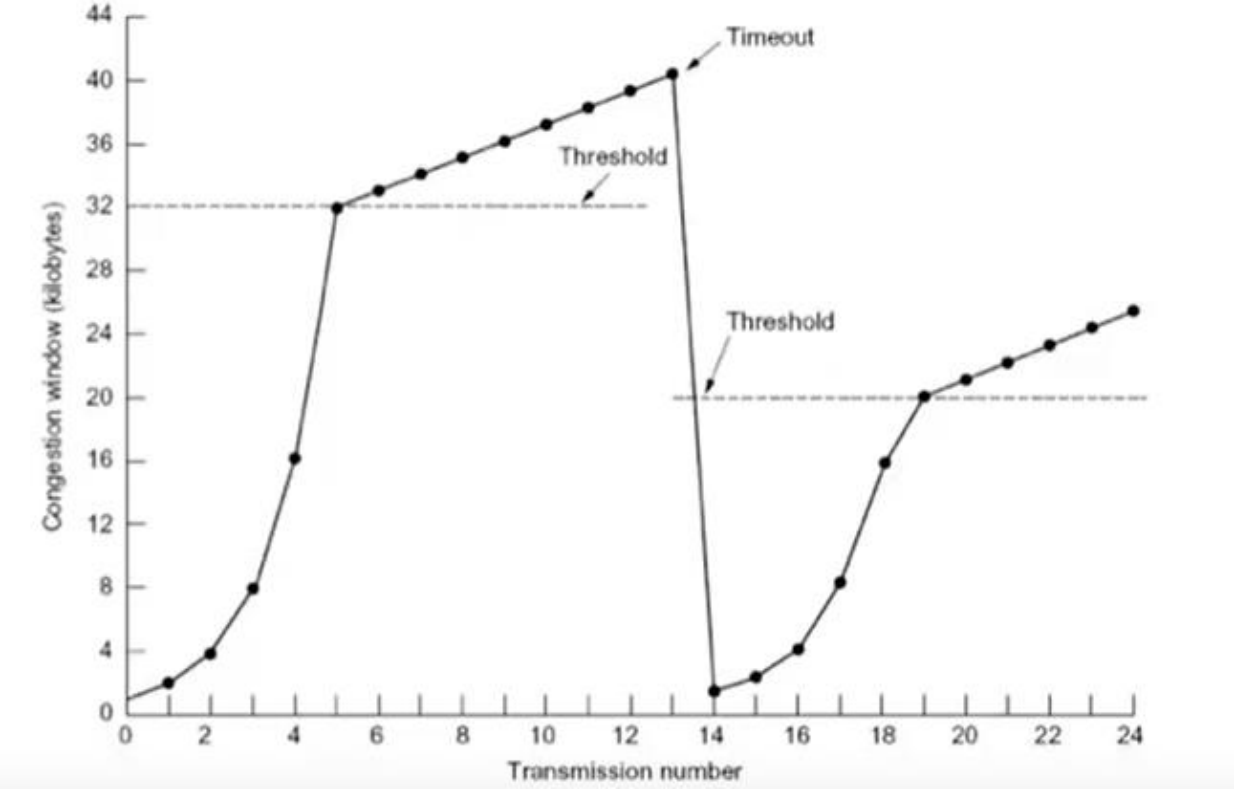


Goals of the Assignment

- Learn first-hand the dynamics of TCP sawtooth and router buffer occupancy in a network.
- Learn why large router buffers can lead to poor performance.
- Learn how to use Mininet to run traffic generators, collect statistics and plot them.
- Learn how to package your experiments so it’s easy for others to run your code.

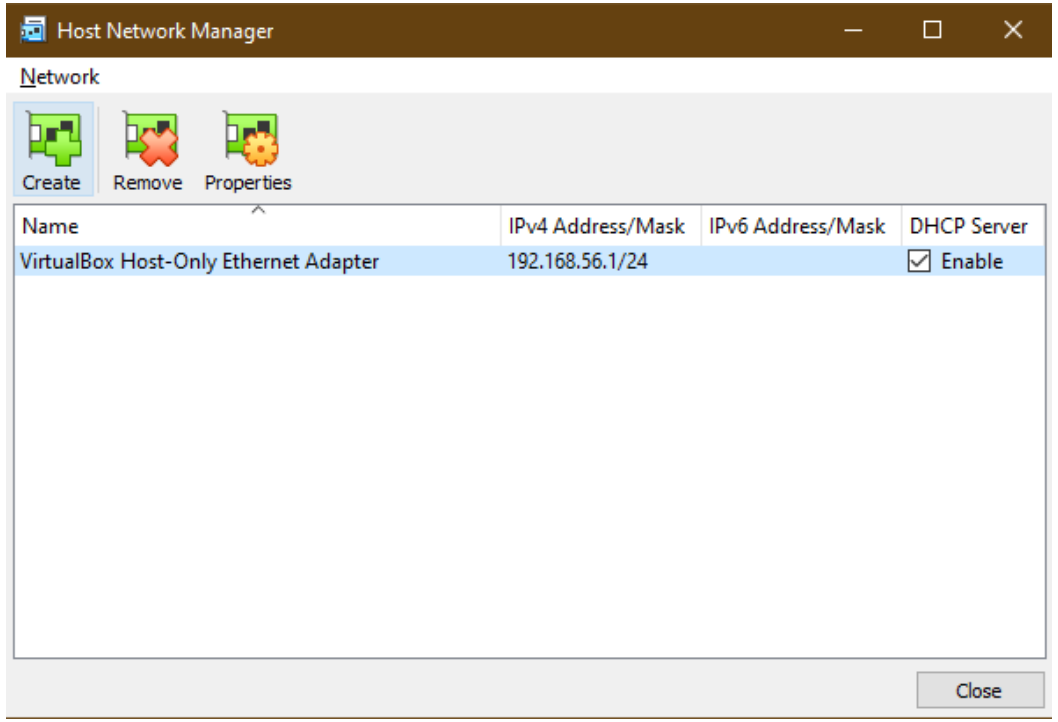
TCP Reno

- Modes:
  - Slow Start: cwnd < ssthresh
  - Congestion Avoidance: cwnd > ssthresh
- Loss:
  - Triple Duplicate ACK: cwnd, ssthresh ← cwnd/2
  - Timeout: ssthresh ← cwnd/2 & cwnd ← 1MSS



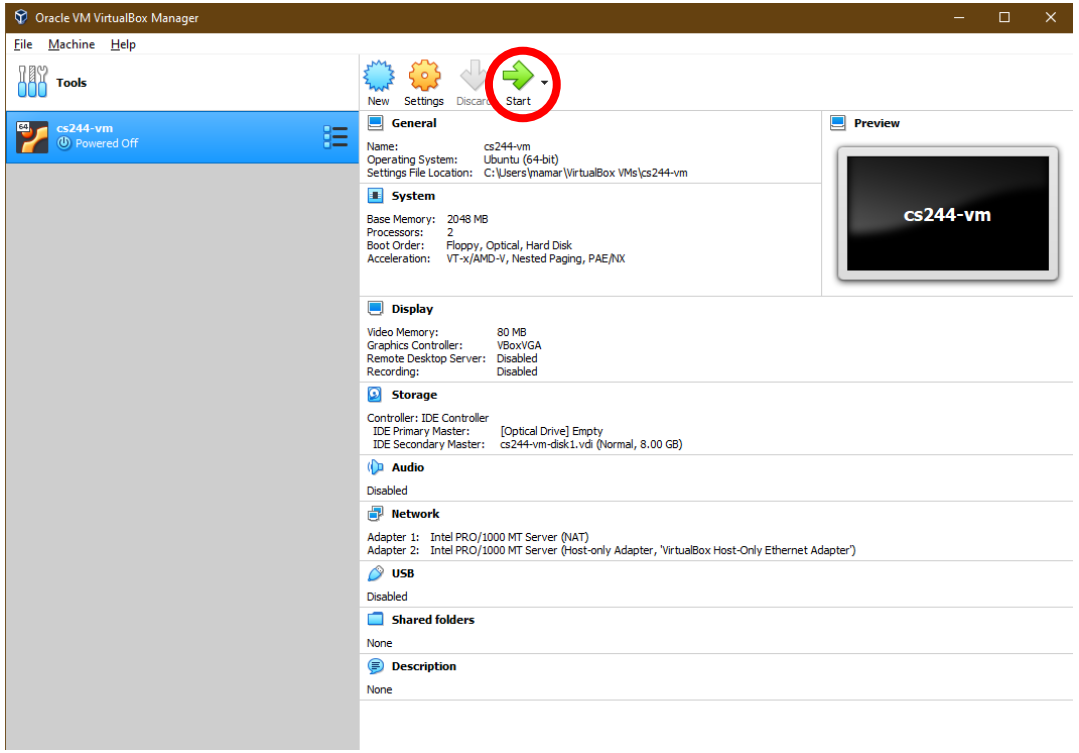
Installation and Setup

- Download and Install Virtual Box.
- Go to File -> **Host Network Manager** and add a host-only network adapter.



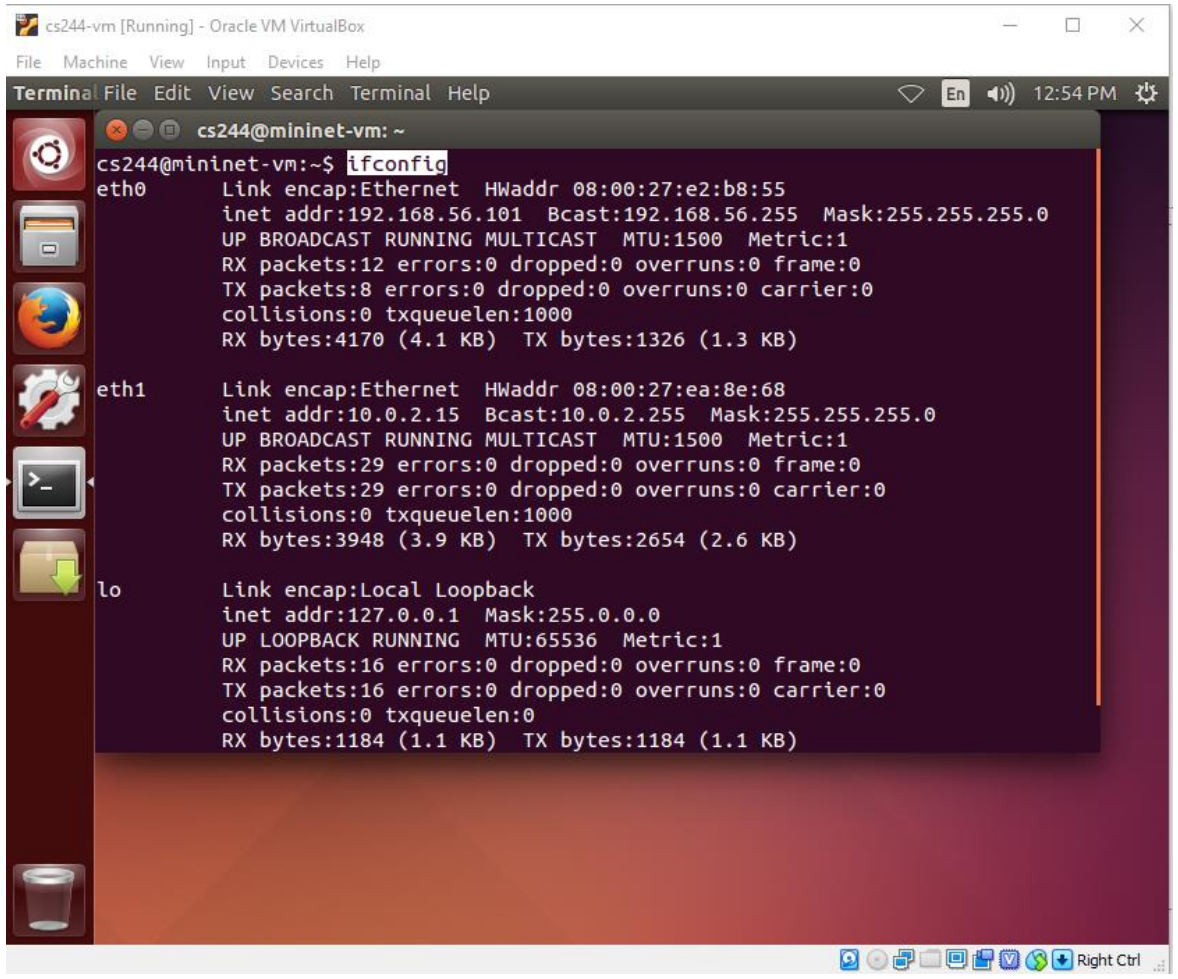
Installation and Setup

- Download the PA2 OVM File.
- Import VM: On VirtualBox, go to File->Import Appliance and select the .ova file from your unzipped folder.
- Start and login in your VM

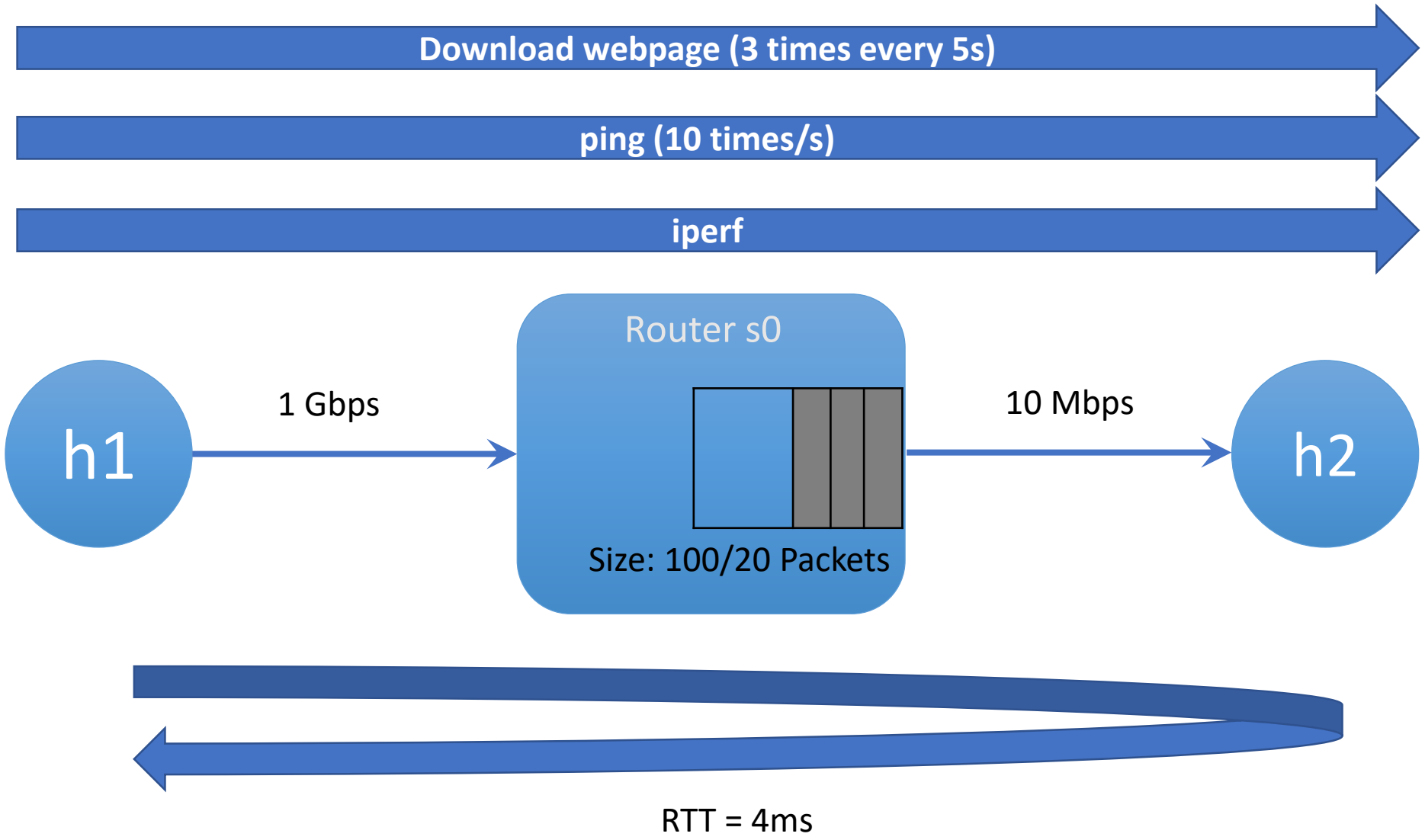


Installation and Setup

- Check the available networks in your VM:
  - ifconfig
- Check Internet connectivity:
  - ping google.com
- All the needed packages are already installed.



Topology


Download webpage (3 times every 5s)
ping (10 times/s)
iperf
Router s0
h1
1 Gbps
10 Mbps
h2
Size: 100/20 Packets
RTT = 4ms

Plots and Results

For Queue size of 20 and 100:

- Plot
  - The long lived TCP flow's cwnd: bb-q[qsize]/cwnd-iperf.png
  - The RTT reported by ping: bb-q[qsize]/rtt.png
  - Queue size at the bottleneck: bb-q[qsize]/q.png

  } 6 plots in total.
- Report
  - avg and std of webpage fetch time. → Report in readme file, question 1

Files

- **bufferbloat.py**:
  - Creates the topology and starts the network.
  - Starts iperf, webserver, ping.
  - Measures cwnd, queue size, RTT, and fetch time.

- **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.

- **run.sh**: Calls scripts above to run experiments.

Files

- The project is almost complete!
- You just need to implement few TODOs in *bufferbloat.py*
- You need to read *bufferbloat.py* carefully first, it helps you complete TODOs.

bufferbloat.py TODOs

- Topology: add links
  - bw, delay, max_queue_size
- Start iperf (client)
  - Specify the experiment time.
- Start ping
  - Read the ping man page to see how to ping every 0.1s for a specific time.
- Start monitoring queue size
  - Uncomment ☺
- Fetch webpage and measure the time.
  - Curl
- Look the code carefully.

Deliverables

A single file: pa2.tar.gz

- **Final Code**:
  - MUST be runnable as a single shell command (sudo ./run.sh)
- **README:**
  - Instructions to run the code and reproduce the results.
  - Answers to the questions.
- **Plots**: There should be 6 plots in total, 3 each for router buffer sizes 100 and 20 packets.
  - 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**
  - Identify your answers with the question number
  - **Please keep your answers brief.**