Advanced Networking 2018

Lab #1: TCP Congestion Control

Total points: 25 pts

Assignment

Lab date: Feb.09 and Feb.13, 2018

Submission date: Feb. 15, 2018 @8PM CET

Authors: Ralph Koning, Paola Grosso, Arno Bakker

Emails: r.koning@uva.nl, p.grosso@uva.nl, Arno.Bakker@os3.nl

University of Amsterdam

Introduction

This assignment helps to learn more about congestion control in the TCP protocol through emulation and simulation of various TCP implementations.

- This lab consists of two mandatory tasks, one for simulation (task1) and one for emulation (task2). Completing perfectly Task1 and Task2 gives 20 points (8/10). Task3 is an advanced task, meant for further exploration and deepening in the subject (and full points for the lab).
- In this lab you will work in pairs. Choose a colleague to work with.
- You have two afternoons to complete this lab.

Related documents

- ns-3 manual: `http://www.nsnam.org/doxygen/index.html`
- ns-3 tutorial: `http://www.nsnam.org/docs/release/3.27/tutorial/ns-3-tutorial.pdf`
- gnuplot reference card: `http://www.gnuplot.info/docs_4.0/gpcard.pdf`
- `google.com` :-)

Submission

You should create a single overall submission for all the tasks in lab 1 that must contain the following:

1. Only **one report in PDF format** containing all answers and graphs with the name, generated using the **supplied .tex template**:
2. The **.tex file** with the answers from which you generated the pdf.

   `lab1-report-${groupnumber}.pdf`

   `lab1-report-${groupnumber}.tex`

3. An archive file (.tar.gz or .zip) for all source codes, data logs and plots. The file name must be:

   `lab1-source-${groupnumber}.tar.gz`

Any other kind of submission will not be taken into account.

Task 1: TCP congestion simulation (6 points)

Task1 is about network simulation; you will use the NS3 simulator to simulate data transfers using different congestion algorithms and circumstances.

Requirements for Your Submission of Task1

- Use *gnuplot* to make your plots.

  Note: if you want to use *gnuplot* for plotting graphs from a data log, you can refer to example in the `plot-lab1-task1.gnu` script.
- Use the supplied tex template to write a small report.
- Provide plots and raw data logs.
- The file names (for the 3 scenarios) must be named (lowercase):

  `lab1-group${groupnumber}-task1-question${questionnumber}.[png,jpg,log]`

  e.g.

  `lab1-group6-task1-question1.1.log`

Preparation

Follow the instructions at: `http://amiens.studlab.os3.nl/an2018/lab1/instructions.html`.

You are now ready to run NS3 simulations. You will work with the example in: *examples/tcp/tcp-variants-comparison.cc* Read the code and try to understand the various options available when running the simulation.

You can run this simulation by issuing the command:

```
./waf --run tcp-variants-comparison --command-template="%s --tracing=1 --pcap_tracing=1 \
    --sack=0 --duration=100"
```

Note that by setting the *tracing* and *pcap_tracing* options to 1 (true) you will get a number of output files in your working directory in the Vagrant VM. Make sure you copy this data to your machine for further analysis.

Scenario 1: TcpNewReno

In this first scenario you will investigate the behaviour of the TCP New Reno implementation. To select the New Reno implementation you have to explicitly indicate this as command line option:

```
./waf --run tcp-variants-comparison --command-template="%s --tracing=1  --pcap_tracing=1 \
    --sack=0 --duration=100 --prefix_name=TcpNewReno --transport_prot=TcpNewReno"
```

Table 1: Example Table

Time (s)	Current CWND (bytes)	New CWND (bytes)	New State	Event
...				
...				
...				
...				

- Q1.1 Plot a graph showing CWND versus time from 0.0s to 100.0s.
- Q1.2 Plot a graph showing SSTH versus time from 0.0s to 100.0s.
- Q1.3 Find the points where the slow-start, congestion-avoidance, fast retransmit/fast recovery states begin. Provide a table with time, CWND value, and state (precision of 0.001s).

When analyze the CWND changes use the layout of Table 1. This will allow you to identify where state-change points occurred and fill in following fields in the table below: the time, CWND before and after change, the new state and the event that caused the change. Provide also the initial states.

Scenario 2: Another TCP variant

Choose another TCP implementation variant available in the simulation. Repeat the previous step:

- Q1.4 Plot a graph showing CWND versus time from 0.0s to 100.0s.
- Q1.5 Plot a graph showing SSTH versus time from 0.0s to 100.0s.
- Q1.6 Find the points where the slow-start, congestion-avoidance, fast retransmit/fast recovery states begin. Provide a table with time, CWND value, and state (precision of 0.001s).
- Q1.7 Discuss and motivate the differences you observe between the NewReno and this algorithm.

Task 2: Emulation of TCP Congestion Window (14 pts)

Task2 is about network emulation; you will use Netem and the traffic control (*tc*) facilities on the linux host to emulate different network circumstances, change congestion algorithms and analyze them.

Requirements for Your Submission of Task2

- Use *gnuplot* to make your plots.
- Use the supplied tex template to write a small report.
- The file names (for the 3 scenarios) must be named (lowercase):

  `lab1-group${groupnumber}-task2-question${questionnumber}.[png,jpg,log]`

  e.g.

  `lab1-group6-task2-question1.1.log`

  or when a range of a plot is requested:

  `lab1-group6-task2-question1.1-xrange-0-5.png:w`

Preparation

Follow the instructions for task 2 at: `http://amiens.studlab.os3.nl/an2018/lab1/instructions.html`.

These are the necessary tools to complete this lab:

1. "TCP probe": In our experiments you can used tcp_probe along with iperf to record the state of TCP connection in response to incoming packet. You can read the output of tcp_probe from procnettcp_prob.

   Tcp probe can be loaded as a kernel module:

   ```
   modprobe tcp_probe port=5001 full=1
   ```

   In order to make the lab not affected by other factors, please remove the tcp probe module every time before starting a new emulation using `modprobe -r tcp_probe` and then add it again.

2. "IPerf(2)": Iperf is tool which you can use to generate the network traffic. The advantage of using iperf is that it allows us to tune various network parameters. Iperf reports bandwidth and jitter loss (In case of TCP transmission). Iperf has to be run at server and client side, also iperf has its own TCP version (it is not dependant on TCP version of the underlying system). This can be selected by -Z flag (e.g. Iperf -c -Z cubic).

To make sure that the system you use is in sync with the experimental set up, you must select the same TCP congestion algorithm at system level as used with iperf. You can use the command "cat /proc/sys/net/ipv4/tcp_available_congestion_control" to know how many congestion algorithm the machine can support. Use the command (echo "cubic" > /proc/sys/net/ipv4/tcp_congestion_control) to change the congestion algorithm for the entire machine.

3. "Netem": Netem is a command line network emulator for testing network protocols. You can use this emulator in combination with traffic control (tc) tool to set network parameters like delay and loss by choosing the appropriate interface device.

   Possible command: "tc qdisc add dev eth0 root handle 1:0 netem delay 100ms loss 2%".

4. "gnuplot": You can use the sample `plot-lab1-task2.gnu` to make a graph of time in seconds in x-axis versus segments (cwnd, sshtresh) on y-axis.

   Note: Before plotting the congestion window, please remove the last line of the result data file you get. Because sometime the last line is not completed, it may cause error when plotting.

Scenario 1: Reno, low latency

Configure the machine to use the congestion algorithm of "TCP reno" and use "iperf" to emulate a TCP flow for 200 seconds. In this scenario, set the latency of this link to 50ms and the packet loss rate as 0%.

- Q2.1 Plot a graph showing the CWND and ssthresh versus time with all the data you get. These two metrics are in one graph.
- Q2.2 Briefly discuss the changing process.

Scenario 2: Reno high latency

Use the congestion algorithm of "TCP reno" and use "iperf" to emulate a TCP flow for 200 seconds. Here, set the latency to 250ms and the packet loss rate as 0%.

- Q2.3 Plot a graph showing CWND versus time with all the data you get.
- Q2.4 Compare this graph with the one from Q1.1, show the difference between these two graphs.

Scenario 3: Reno, packet loss

Use the congestion algorithm of "TCP reno" and use "iperf" to emulate a TCP flow for 200 seconds. Here, set the latency to 50ms and the packet loss rate as 3%. After completing the emulation, make a snapshot of the iperf to show the throughput.

- Q2.5 Plot a graph showing CWND and ssthresh versus time with all the data you get.

- Q2.6 Compare this graph with the graph of Q1.1 and show the differences.
- Q2.7 Zoom in the graph of this scenario (plot some parts of this scenario in a short duration, 10 or 20 seconds). Briefly explain the changing process.
- Q2.8 Show a screen capture of the real throughput in this scenario.

Scenario 4: Cubic, low latency

Use the congestion algorithm of "TCP cubic" and use "iperf" to emulate a TCP flow for 200 seconds. Here, set the latency to 50ms and the packet loss rate as 0%.

- Q2.9 Plot a graph showing CWND and ssthresh versus time with all the data you get.
- Q2.10 Compare this graph with the graph of Q1.1 and show the differences.

Scenario 5: Cubic, packet loss

Use the congestion algorithm of "TCP cubic" and use "iperf" to emulate a TCP flow for 200 seconds. Here, set the latency to 50ms and the packet loss rate as 3%. After completing the emulation, make a snapshot of the iperf to show the throughput.

- Q2.11 Plot a graph showing CWND and ssthresh versus time with all the data you get.
- Q2.12 Compare this graph with the graph of scenario three and show the differences.
- Q2.13 Zoom in the graph of this scenario (plot some parts of this scenario in a short duration, 10 or 20 seconds). Briefly explain the changing process and compare it with the graph of Q3.2.
- Q2.14 Show a screen capture of the real throughput and compare it with throughput of Q3.4. Tell the differences.

Task 3 - Advanced scenarios: TCP performance (5 points)

If times remains, return back to the simulation environment of Task1. In this task you are asked to run as many experiments you want with varying parameters to be able to answer/address the following questions. You can choose which TCP variants to use.

- Q3.1 Explain what an LFN network is. Change the simulation parameters to your likings and demonstrate that TcpNewReno is not suitable for LFN networks.
- Q3.2 Explain SACK does. Change the simulation parameters to your likings and demonstrate the performance improvement with SACK.
- Q3.3 Explain with TCP fairness is. Show the effect of multiple flows in the simulation.
- Q3.4 Replicate scenario 3 of the emulation: packet loss of 3%, delay of 50 ms and transfer duration of 200sec. Use TcpNewReno. Compare the two results.
- Q3.5 After these experiments, please briefly describe the difference between simulation and emulation?