CWM Programmable Networks

Exercise 2: Network Measurements

Introduction

As part of this exercise, you will practice measuring two types of network properties, *round trip time* and *effective bandwidth*.

A script is provided in the CWM's repository, under *assignment2*, for plotting graphs. It is recommended to complete all the measurements and collect the results before generating the graphs.
This exercise needs to be submitted as a pdf file via Canvas.

Setup

In this exercise you will use two machines: your lab machine, and a Raspberry Pi, connected directly using a single Ethernet cable. The cable should be connected from the lab machine's USB adapter to the Raspberry Pi's Ethernet port (no adapter).

Important! The lab machine's USB adapter should be connected to a USB 3.0 (blue) port, or the throughput will degrade.

To get started, open two terminals on the lab machine:

- One terminal will be used for measurements from the lab machine
- One terminal will be used for measurements from the Raspberry Pi
    - Connect to the Raspberry Pi using local monitor or ssh

Syntax

The following details for every command only the options used as part of the lab. See the command manual for the full list of options.

ping:

ping <address> [Optional: -c <count>] [Optional: -i <interval>] [Optional: -f]

`-i` (interval) - gap between pings in seconds, default=1.

`-f` (flood) - flooding. If an interval is not given, it sets interval to zero and outputs packets as fast as they come back or one hundred times per second, whichever is more.

`-c` (count) - number of ping measurements. If unspecified, ping runs until killed.

Example:

```
ping 192.168.10.2 -c 1000 -i 0.2
```

ping 1000 times with an interval of 200ms to address 192.168.10.2

iperf (Server):

```
iperf -s -B <server address> [Optional: -u]
```

`-u` (UDP) - use UDP rather than TCP.

Example (Pi => server, ENGS-LABxx => client):

```
iperf -s -B 192.168.10.
```

Run iperf server and bind it to IP address 192.168.10.2 (useful when multiple interfaces exist)

iperf (Client):

TCP:

`iperf -c <server address> [Opt: -i <interval>] [Opt: -t <time>] [Opt: -f <format>] [Opt: -d] [Opt: -w <TCP window size>]`

`-i` (interval) - how often to report bandwidth, in seconds.

`-t` (time) - length of test, in seconds. Default value is 10.

`-f` (format) - 'k' for Kbits/sec, 'm' for Mbits/sec, 'g' for Gbits/sec. Default is adaptive.

`-d` (dual test) - Do a bidirectional test simultaneously.

`-w` (TCP window size) - TCP window size (socket buffer size), e.g., -w 20KB.

Example (Pi is the server, and ENGS-LABxx is the client):

`iperf -c 192.168.10.2 -i 5 -t 20 -d`

Run iperf client for 20 seconds with the server running at IP address 192.168.10.2, report the results every 5 seconds. Run the test in both directions (client to server and server to client)

UDP:

```
iperf -c <server address> [Opt: -i <interval>] [Opt: -t <time>][Opt: -f <format>] [Opt: -d] [Opt: -b <bandwidth>] -u
```

`-i` (interval) - how often to report bandwidth, in seconds.

`-t` (time) - length of test, in seconds. Default value is 10.

`-f` (format) - 'k' for Kbits/sec, 'm' for Mbits/sec, 'g' for Gbits/sec. Default is adaptive.

`-d` (dual test) - Do a bidirectional test simultaneously.

`-b` (bandwidth) - bandwidth to send at in bits/sec, e.g. -b 20m (Mbits/sec).

-u for using UDP

Note: By default, iperf restricts bandwidth for UDP clients to a maximum of 1 Mbit/sec. There is no restriction for TCP clients.

Example (Pi => server, ENGS-LABxx => client):

```
iperf –c 192.168.10.2 –i 5 –t 20 –b 100k –u
```

Run iperf client for 20 seconds with the server running at IP address 192.168.10.2, report the results every 5 seconds. Run the test as UDP and restrict the bandwidth to 100kb/s (the server has to run as UDP as well)

iperf3 (Server):

TCP/UDP:

```
iperf3 -s -B <server address>
```

iperf3 (Client):

TCP:

```
iperf3 -c <server address> [Opt: -i <interval>] [Opt:
-t <time>][Opt: -f <format>] [Opt: -w <window size>]
```

UDP:

```
iperf3 -c <server address> [Opt: -i <interval>] [Opt:
-t <time>][Opt: -f <format>][Opt: -b <bandwidth>] -u
```

Plotting Graphs:

1. Save your data in space-delimited format, like in the example `test.data` file.

The first column is the x-axis (e.g., time), and the second column is the y-axis (e.g., packet drops).

Use awk, cut or other tools to get the data that you need, for example:

```
ping 192.168.10.2 –c 100 –i 0.2 > ping_test.txt
```

```
cat ping_text.txt | grep 'time='| awk -F '[=, ]'
'{print $10}'
```

The first command will run a ping test and save the results into the file `ping_test.txt`. The second command will get all the RTT lines in the file and extract the RTT value.

A different example:

```
ping 192.168.10.1 -c 10 -i 0.1|grep 'time='|awk -F
'[=, ]' '{print 0.1*NR, $10}'> processed_ping.txt
```

This measure ping with an interval of 0.1s, and then save to the file `processed_ping.txt` only the measurement time and RTT. `NR` indicates record number, therefore `0.1*NR` is the time since the start of the measurement.

Note: it is strongly recommended to save the raw results to a file and then process it, and not to measure & process directly, as this may lead to loss of time (require repeated measurements) and data.

2. Modify parameters within the file `plot.py`.

The parameters to be modified are:

- `filename` – name of the data file

- `label` – label to be used in the legend, e.g., “iperf2” or “iperf3”.

- `xlabel` – x-axis label

- `ylabel` – y-axis label

- `title` – title of the plotted graph

- `fig_name` – name of the saved figure file

3. Run the python script to plot your figure

```
python3 plot.py
```

Ping

1. Ping from the lab machine to the Raspberry Pi, 10 times, interval 0.2 seconds.
   Report the results.
2. Ping from the Raspberry Pi to the lab machine, 10 times, interval 0.2 seconds.
   Report the results.
3. Ping from the Raspberry Pi to the lab machine, 100 times, interval 0.001 seconds (use sudo).
   Report the results, and discuss the differences between minimum, mean and maximum results.
4. Ping from the Raspberry Pi to the lab machine, 10000 times using flooding (use sudo).
   Report the results.
5. Ping from the Raspberry Pi to the lab machine. Run measurements with 3 different intervals (0.01, 0.001, 0.0001) and at least 1000 measurements, and draw a cdf of your measurements results (one graph per interval).
   The following command will help you:
   Writing to a file called ping.log:

   ping <address> [Optional: -c <count>] [Optional: -i <interval>] [Optional: -f] > ping.log

   example:

   ping 192.168.10.2 -c 1000 > ping.log

   You can copy the generated file from the Raspberry Pi to the lab machine using the scp command on the lab machine:

```
scp pi@192.168.10.2:filename .
```

Note that this will copy the file to your current directory.

6. Can you speculate why different intervals lead to different round trip results? What do you estimate is the most accurate measured parameter (e.g., min, max, mean) that can be used to estimate propagation time between the two machines?

iperf

1. Set the lab machine as iperf server, and the Raspberry Pi as the client, and use iperf to measure the effective bandwidth between the two, using TCP and a 10 seconds long experiment.
   Report the results.
2. Set the Raspberry Pi as iperf server, and the lab machine as the client, and use iperf to measure the effective bandwidth between the two, using TCP, a 10 seconds long experiment, and 1 second interval.
   Plot the measured bandwidth.
   The following command will help you:
   Writing to a file called iperf.log:
   ```
   iperf -s -B <server address> [Optional: -u] > iperf.log
   ```
   Example:
   ```
   iperf -s -B 192.168.10.2 > iperf.log
   ```
3. Set the Raspberry Pi as iperf server, and the lab machine as the client, and use **bi-directional** iperf to measure the effective

bandwidth between the two, using TCP, a 10 seconds long experiment, and 1 second interval.
Plot the measured bandwidth in each direction.

4. Run one way iperf using UDP, from the lab machine to the Raspberry Pi, 5 sec long, with varying bandwidth (100Kb/s, 1Mb/s, 100Mb/s).
    a. Plot the percentage of packets dropped for each bandwidth.
    b. Report the iperf results of each configuration.

iperf3

1. Set the Raspberry Pi as iperf3 server, and the lab machine as the client, and use iperf3 to measure the effective bandwidth between the two, using TCP, a 10 seconds long experiment, and 1 second interval.
   Plot the measured bandwidth.
2. Run one way iperf3 using UDP, from the lab machine to the Raspberry Pi, 5 sec long, with varying bandwidth (100Kb/s, 1Mb/s, 100Mb/s).
    a. Plot the percentage of packets dropped for each bandwidth.
    b. Report the iperf3 results of each configuration.
3. Discuss any observed differences between iperf and iperf3 results.

Submit a pdf on Canvas with your answers, graphs and discussions.

You should include a link to your repository containing the raw files of your measurements.