LAB V: NETWORK PERFORMANCE MEASUREMENT

Objectives

After this lab, the student should be able to

- Perform measurements on the network performance metrics such as delay and throughput.
- Identify how the network performance is affected by different factors.

Equipment List

Equipment	Quantity
Linux computer with one Ethernet interface	4
Unmanaged Ethernet switch	2

Background

In Linux Raspberry Pi, the nuttcp command is a tool used to generate synthetic UDP and TCP traffic loads. Together with ping and traceroute, nuttcp is an essential utility program for debugging problems in IP networks. Running the *nuttcp* tool consists of setting up a *nuttcp* receiver (server) on one host and then a *nuttcp* sender on another host. Once the *nuttcp* sender is started, it sends the specified amount of data as fast as possible to the *nuttcp* receiver.

An *nuttcp* receiver process is started with the command

```
nuttcp -S --nofork [-lbuflen] [-pport] [-u]
```

An *nuttcp* sender process is started with the command

```
nuttcp -t [-lblocksize] [-nnumblock] [-pport] [-u] [-D] IPaddress
```

The options of the command are:

-S Make nuttcp wait for connections (server/receiver)

-t Specifies the transmit mode (client/transmitter)

-u Specifies to use UDP instead of TCP. By default, *nuttcp* uses TCP to send data.

-n*numblock* Number of data blocks to be transmitted (default value is 2048 blocks).

-l*blocksize* Size of the data blocks that are passed to UDP or TCP in bytes (default is 4096 bytes). When UDP is used, this value is the number of data bytes in UDP datagram.

-D Disables buffering of data in TCP and forces immediate transmission of the data at the *nttcp* sender. Used only in the context of TCP.

-p*port* Port number to send to or listen on. The port number must be identical at the sender and at the receiver. The default value is 5000.

IPaddress IP address of the *nttcp* receiver to send data to.

Use `nuttcp -h` or to verify the command usage option before using it to send/receive files.

Lab Procedures

Part I: Measuring RTT with Traceroute

1. Connect to KMUTT Wi-Fi networks and do traceroute
   a. If you use an MS Windows laptop, open Command prompt and enter

      ```
      tracert www.google.com
      tracert www.facebook.com
      tracert www.apple.com
      ```

   b. If you use a MacOS laptop, open Terminal and enter

      ```
      traceroute www.google.com
      traceroute www.facebook.com
      traceroute www.apple.com
      ```

   Save the screen outputs and separate Wireshark captures for each traceroute command to answer the lab questions. Traceroute sends out three packets in each round. For each packet sent in each round, look in the IP header and note the value of Time-to-live (TTL) field.

2. Continue from Step 1:
   a. For MS Windows, enter

      ```
      ping -n 10 www.google.com
      ping -n 10 www.facebook.com
      ping -n 10 www.apple.com
      ```

b. For MacOS, enter

```
ping -c 10 -i 3 www.google.com
ping -c 10 -i 3 www.facebook.com
ping -c 10 -i 3 www.apple.com
```

Save the screen outputs to answer the lab questions.

<u>Lab Questions:</u>

1.1 From Step 1,

(a) How many "hops" in the path to each destination?

(b) Are there any common intermediate routers among the three paths? What are they?

(c) Which one among the three destinations appear to have the highest average RTT and lowest average RTT? Are they related to any variable of the network ?

1.2 Examine the Wireshark captures in Step 1 and answer the following questions:

(a) What protocols are the payload of IP packet ? Do these packets have port numbers ?

(b) How are the values of TTL field in IP header changed by the source in each round?

(c) What packets were received to the source in each round? Who are the senders of those packets? Why those packets were generated ?

(d) From (b) and (c), Can you determine the role of TTL value and how it is related to the senders of the returned packets in each round?

(e) In the last round of packets, What packets were received by the source? Were they different from those in (c)? Explain.

(f) Do you think why traceroute needs to send out three packets, not just one, in each round?

1.3 From Step 2, among the three destinations, which one has the highest average RTT and lowest average RTT? Are the answers consistent with your answers in 1.1 ?

Part II: Measuring Network Throughput

1. Set up the network topology as shown in Fig. 1.

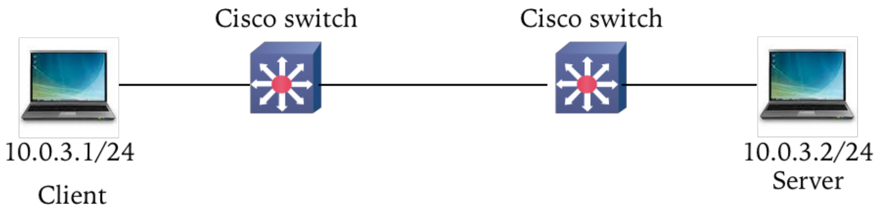

Figure 1: One-client download topology.

2. **Prepare downloading at Client:** Run nuttcp receiver with the command

```
Client% nuttcp -S -l1024
```

3. **Perform *TCP* transmission at Server**: To emulate the client download, run nuttcp sender to send 500 MB data with the command

```
Server% nuttcp -t -l1024 -n500000 10.0.3.1
```

   Record the download throughput in Megabits per second (Mbps) from the output display.

4. Repeat Step 3 with 2, 3, 4, and 5 parallel FTP downloads as following:
   a. Open 2, 3, 4, and 5 terminal windows at Server
   b. Type the nuttcp command in all terminals first and then enter so that the transfer will begin at the same time.
   c. Record the transfer throughputs in Mbps of all connections in the table format below.

Number of parallel TCP connections to client	Download throughputs (Mbps)	Average per-connection throughput
1	x1	x1
2	x1,x2	Average of x1, x2
3	x1,x2,x3	Average of x1, x2, x3
4	x1,x2,x3,x4	Average of x1, x2, x3, x4
5	x1,x2,x3,x4,x5	Average of x1, x2, x3, x4, x5

5. Add another client to the network as shown in Fig. 2 and run the nuttcp receiver as in Step 2.

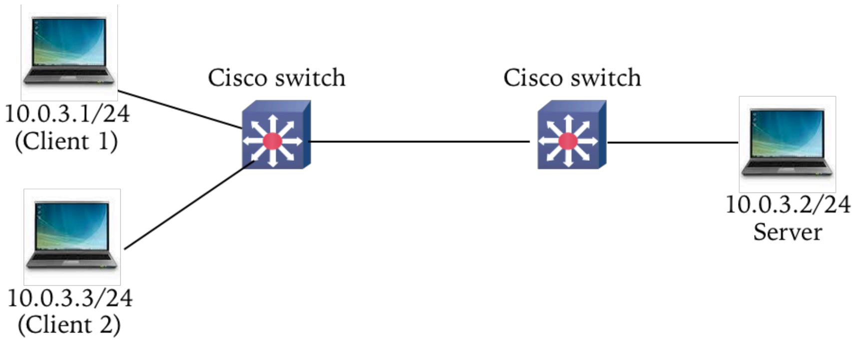

Figure 2: Two-client download topology

6. Repeat the nuttcp transmission in Steps 3 and 4 at Server with the number of TCP connections to each client as shown in the first two columns of the table below. Record the transfer throughput of all connections (x_i's for those of client 1 and y_i's for those of client 2) in the table. Note that the average per-connection transfer throughput is computed across two sources.

# TCP connections to client 1	# TCP connections to client 2	Number of TCP connections over the middle link	Download throughputs (Mbps)	Average per-connection throughput across two clients
1	1	2	x1 y1	Average of all xi and yi combined.
1	3	4	x1 y1, y2, y3	
2	4	6	x1,x2 y1,y2,y3,y4	
3	5	8	x1,x2,x3, y1,y2,y3,y4,y5	
4	6	10	x1,x2,x3,x4 y1,y2,y3,y4,y5,y6	

Report Questions:

2.1 From data in Steps 3 and 4, plot the per-connection throughputs (x_i) and its average as a function of the number of parallel TCP connections (n).

(a) For the case of one session (Step 3), Is the throughput close to the middle link capacity ?

What is the percentage of throughput to the link capacity?

(b) Explain how the per-connection throughputs you measured are related to the number of parallel TCP connections and the link capacity.

2.2 From data in Step 7, plot the per-connection throughputs (x and y) and its average as a function of the number of parallel TCP connections over the middle link (n).

(a) Do you observe the same behavior of per-connection throughput over the middle link as in 2.1(b) ? Explain.

(b) Which client gets higher total download throughput? Do they get a fair share of the middle link capacity? If not, Do you think this result will be a problem in a real-world situation?