MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Mini Project Report
of
Computer Networks LAB

TITLE

NETWORK STATISTICS SUCH AS THROUGHPUT, AVERAGE RTT, TRANSMISSION SPEED

SUBMITTED BY

Laxman Malyala – 200905064 (Roll No.14)
A. Tarun Kumar Reddy – 200905028 (Roll No.8)

Computer Science & Engineering

Abstract

Throughput is the actual amount of traffic flowing from a specific individual source or group of sources to a specific destination or group of destinations at a specific point in time. This is an important point: Throughput is how much actual traffic is flowing when you do a real time measurement or the rate of data delivery over a specific period of time. It can be measured in packets per second, bytes per second, or bits per second.

Network bandwidth is a measurement indicating the maximum capacity of a wired or wireless communications link to transmit data over a network connection in a given amount of time. Typically, bandwidth is represented in the number of bits, kilobits, megabits or gigabits that can be transmitted in 1 second.

Transmission speed is the rate at which data packets cross a computer network from one server to another. It is not the same as bandwidth. Bandwidth describes the amount of data that a network can or does transmit at a given time. Transmission speed, on the other hand, describes the rate at which the data is sent over the network.

Round Trip Time (RTT) is the length time it takes for a data packet to be sent to a destination plus the time it takes for an acknowledgment of that packet to be received back at the origin.

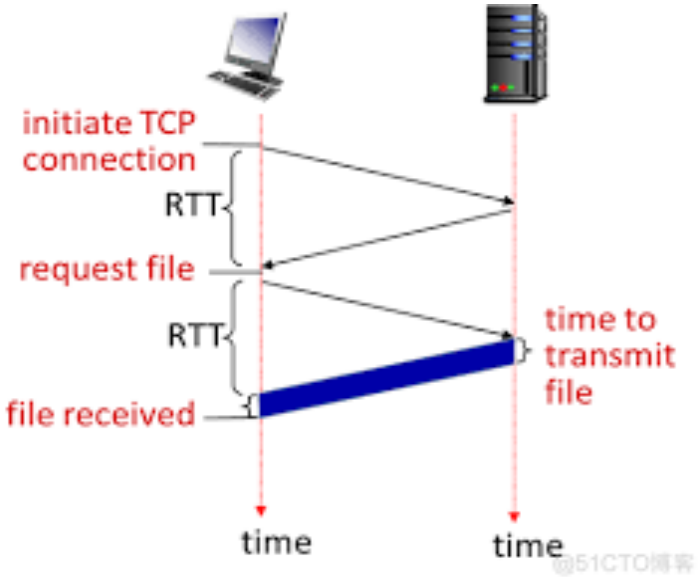


Table of Contents

Chapters

1. INTRODUCTION

The purpose of this project is to develop and implement programs and, where necessary, use tools that help us calculate the statistics of a network like throughput, network speed and the average round trip time.

Measurement of Throughput and RTT for a TCP client and Server program

2. PROBLEM DEFINITION

The program is a tcp client server program where first a connection is established from client to server.

The server then sends a segment of 10 bytes to the server.
The sending time and rtt are calculated *with a precision of nanoseconds* using struct timespec.

3. OBJECTIVES

On the server side, three struct timespec structures variables, t1, t2, and t3 are declared.
The fields in struct timespec are tv_sec and tv_nsec. tv_sec calculates the time elapsed since the epoch in seconds and tv_nsec calculates the time in nanoseconds after tv_sec value.
Struct timespec structure :
t1 calculates the precise time just before sending n packets from server to client.
t2 calculates the precise time just after sending the packets from server to client.
t3 calculates the precise time just after receiving an acknowledgement from the client.

The difference between t1 and t2 is the sending time of the server. Diving the bytes send by sending time gives throughput on the server side gives the sending throughput for 1 cycle.

The difference between t2 and t3 is the difference between the acknowledgement just being received and the bytes just being sent. It gives the RTT for one cycle. Similarly RTT is calculated for all 5 cycles and the average is found.

On the client side, struct timespec structure variables t1 and t2 calculates the receiving time of the bytes coming from server.

And dividing the n packets received by receiving time gives receiving throughput.

4. METHODOLOGY

Outputs

Server side output:

```
:~/200905028/CNL/project$ gcc ser.c -o ser
student@lplab-ThinkCentre-M71e:~/200905028/CNL/project$ ./ser
Socket successfully created..
Socket successfully binded..
server accept the client...

Sending throughput:
Send 1: 220.070423 Kb/s
Send 2: 359.634611 Kb/s
Send 3: 162.911555 Kb/s
Send 4: 165.889750 Kb/s
Send 5: 212.747851 Kb/s

Average rtt:0.000055 sec
student@lplab-ThinkCentre-M71e:~/200905028/CNL/project$
```

Client side output:

```
student@lplab-ThinkCentre-M71e:~/200905028/CNL/project$ gcc cli.c -o cli
student@lplab-ThinkCentre-M71e:~/200905028/CNL/project$ ./cli
Socket created successfully..
Successfully connected..

Receiving throughput:
Receive 1: 157.728707 Kb/s
Receive 2: 243.155182 Kb/s
Receive 3: 84.199182 Kb/s
Receive 4: 117.656748 Kb/s
Receive 5: 197.316496 Kb/s
student@lplab-ThinkCentre-M71e:~/200905028/CNL/project$
```

2. **Getting of RTT through ping command in python:**

The ping command shows as output the maximum, minimum and average RTT amongst the packets transmitted.

Code (in python)

Output:

```
ping.py > ...
1   import subprocess
2   def ping(host):
3       n = input('How many packets to transmit:')
4       command = ['ping', '-c', n, host]
5       return subprocess.call(command)
6   host = input('ENTER HOST NAME:')
7   ping(host)
8
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

/laxman/Documents/CN\ pro/ping.py
ENTER HOST NAME:twitter.com
How many packets to transmit:5
PING twitter.com (104.244.42.1): 56 data bytes
64 bytes from 104.244.42.1: icmp_seq=0 ttl=59 time=173.716 ms
64 bytes from 104.244.42.1: icmp_seq=1 ttl=59 time=218.190 ms
64 bytes from 104.244.42.1: icmp_seq=2 ttl=59 time=69.434 ms
64 bytes from 104.244.42.1: icmp_seq=3 ttl=59 time=55.603 ms
64 bytes from 104.244.42.1: icmp_seq=4 ttl=59 time=57.636 ms

--- twitter.com ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 55.603/114.916/218.190/67.809 ms
```

5. IMPLEMENTATION DETAILS

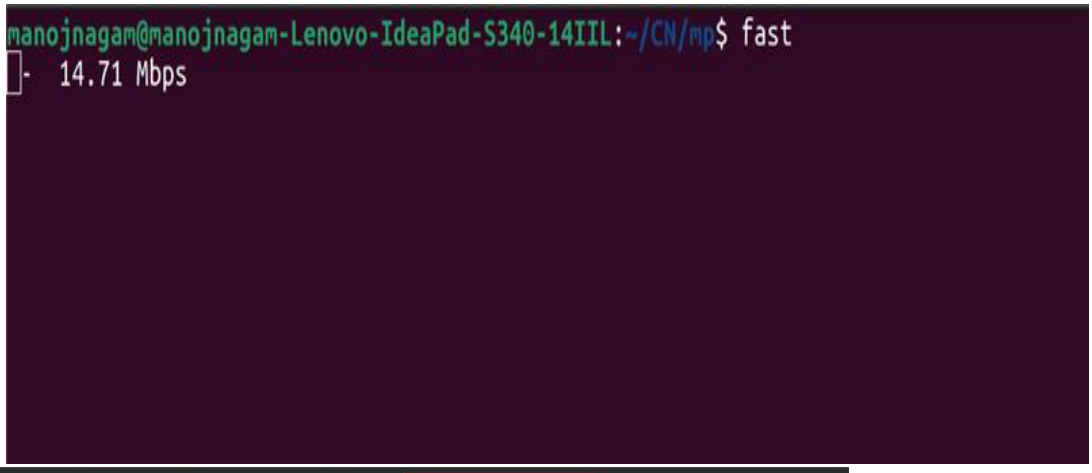
Measurement of network statistics in terminator:
Implementation

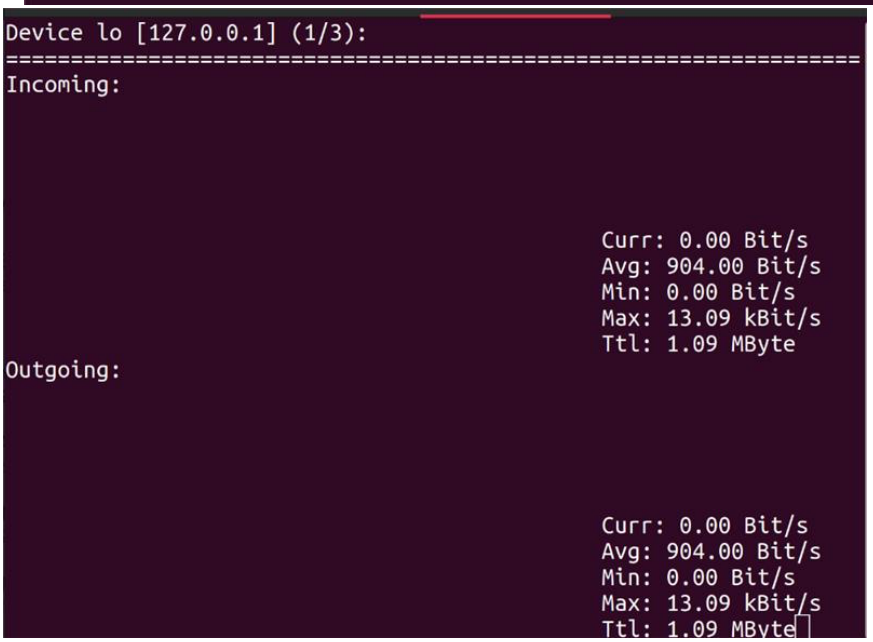
nload: nload is another simple, easy-to-use command-line tool for monitoring network traffic and bandwidth usage in real-time. It uses graphs to help you monitor inbound and outbound traffic. In addition, it also displays information such as the total amount of transferred data and min/max network usage.
fast: This is a file which makes some network traffic continuosly over a period of time.

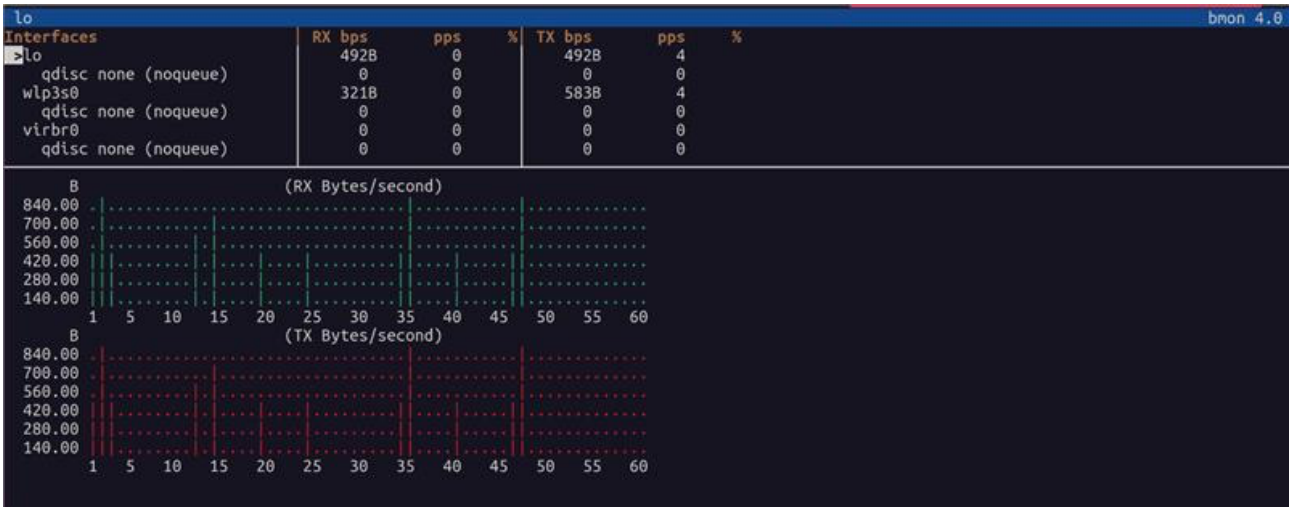
sudo iftop -i lo: iftop is a simple, easy-to-use, real-time top-like command line-based network bandwidth monitoring tool, used to get a quick overview of network activities on an interface. It displays network usage bandwidth updates every 2, 10, and 40 seconds on average.
" lo is the name of the interface on which the network holds on."

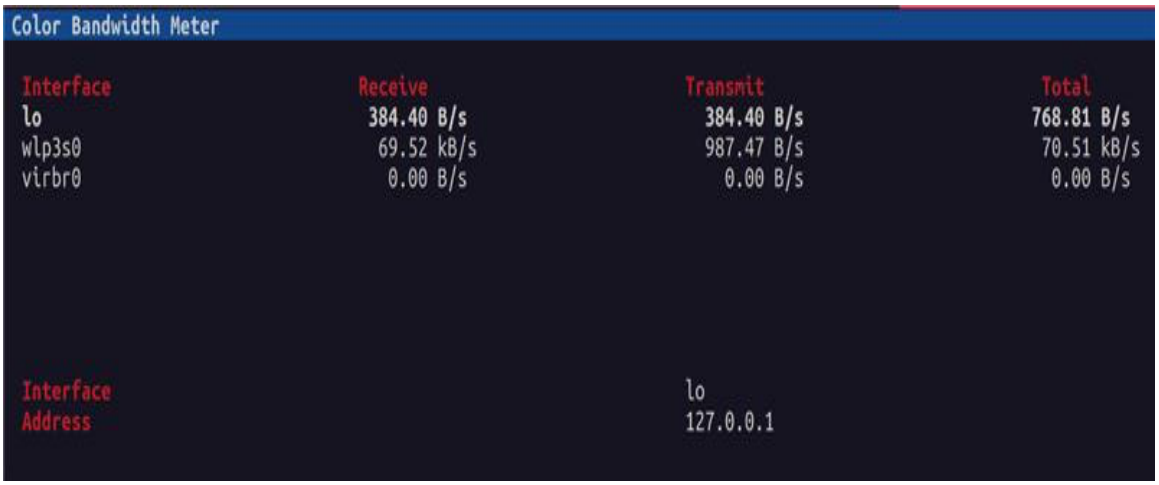
cbm: cbm is a tiny command-line utility for displaying current network traffic on all connected devices in coloured output in Ubuntu Linux and its derivatives such as Linux Mint, ubuntu, and many others. It shows each connected network interface, bytes received, bytes transmitted, and total bytes, allowing you to monitor network bandwidth.

bmon: bmon is also a straightforward command-line tool for monitoring network bandwidth utilization and a rate estimator, in Linux. It captures network statistics and visualizes them in a human-friendly format so that you can keep an eye on your system.


manojnagam@manojnagam-Lenovo-IdeaPad-S340-14IIL:~/CN/mp$ fast
14.71 Mbps


Device lo [127.0.0.1] (1/3):
Incoming:
Curr: 0.00 Bit/s
Avg: 904.00 Bit/s
Min: 0.00 Bit/s
Max: 13.09 kBit/s
Ttl: 1.09 MByte
Outgoing:
Curr: 0.00 Bit/s
Avg: 904.00 Bit/s
Min: 0.00 Bit/s
Max: 13.09 kBit/s
Ttl: 1.09 MByte


lo
bmon 4.0
Interfaces
RX bps
pps
%
TX bps
pps
%
>lo
492B
0
492B
4
qdisc none (noqueue)
0
0
0
0
wlp3s0
321B
0
583B
4
qdisc none (noqueue)
0
0
0
0
virbr0
0
0
0
0
qdisc none (noqueue)
0
0
0
0
B
(RX Bytes/second)
840.00
700.00
560.00
420.00
280.00
140.00
1 5 10 15 20 25 30 35 40 45 50 55 60
B
(TX Bytes/second)
840.00
700.00
560.00
420.00
280.00
140.00
1 5 10 15 20 25 30 35 40 45 50 55 60


Color Bandwidth Meter
Interface
Receive
Transmit
Total
lo
384.40 B/s
384.40 B/s
768.81 B/s
wlp3s0
69.52 kB/s
987.47 B/s
70.51 kB/s
virbr0
0.00 B/s
0.00 B/s
0.00 B/s
Interface
lo
Address
127.0.0.1

6. CONTRIBUTION DETAILS

Code related to server side – Laxman
Code related to client side – Tarun
Design layout and Documentation – both

7. REFERENCES

1. James F. Kurose and Keith W. Ross COMPUTER NETWORKING: A Top-Down Approach
2. tcpdump.org
3. stackoverflow.com
4. youtube.com