Department of Computer Engineering

Faculty of Engineering, University of Peradeniya

CO 323 Computer Communication Networks

Laboratory session 2

Semester 5, 2017

Iperf tutorial

Iperf is a tool to measure bandwidth and quality of a link, that is available for both Linux and Windows platforms. It also has a graphical frontend known as Jperf, written in Java. However, for this lab you will only use Iperf (i.e. the command line tool)

The network link is delimited by two hosts running Iperf.

Figure 1: Example network setup

The quality of a link can be tested using the following parameters:

Table 1: Parameters and related tools

Parameter	**Tool to measure**
Latency	Ping command
Jitter	Iperf UDP test
Datagram loss	Iperf UDP test
Bandwidth	Iperf TCP test

Iperf uses the different capacities of TCP and UDP to provide statistics about network links.

When using Iperf, one host must be set as client, the other one as server. In our lab, we'll use a lab machine as the client and the Tesla server as the server.

Using iperf

By default, the Iperf client connects to the Iperf server on the TCP port 5001 and the bandwidth displayed by Iperf is the bandwidth from the client to the server.

On the server side, first start a server instance using the following command:

iperf -s

It should give an output similar to the following.

Server listening on TCP port 5001

TCP window size: 85.3 KByte (default)

On the client side, type in a console

iperf -c <*server-ip*>

If the serve is on, an output similar to the following can be seen.

Client connecting to <server-ip>, TCP port 5001

TCP window size: 85.0 KByte (default)

(The TCP window size is the amount of data that can be buffered during a connection without a validation from the receiver)

[3] local <client-ip> port *60919 connected with <server-ip> port 5001*

[ID] Interval Transfer Bandwidth

[3] 0.0-10.0 sec 1.08 GBytes 931 Mbits/sec

When the connection is established, the server should print the following:

[4] local <server-ip> port 5001 connected with <client-ip> port 60918

[ID] Interval Transfer Bandwidth

[4] 0.0-10.0 sec 1.08 GBytes 931 Mbits/sec

> The above output clearly shows the bandwidth of this connection.

Describing the output:

Now let us check some more useful features of iperf.

Data formatting: (-f argument)

The -f argument can display the results in the desired format:

bits(b), bytes(B), kilobits(k), kilobytes(K), megabits(m), megabytes(M), gigabits(g) or gigabytes(G).

Generally the bandwidth measures are displayed in bits (or Kilobits, etc ...) and an amount of data is displayed in bytes (or Kilobytes, etc ...).

On the client side, type in a console

iperf -c *<server-ip>* -f *M*

If the serve is on, an output similar to the following can be seen.

Client connecting to 10.40.18.9, TCP port 5001

TCP window size: 0.08 MByte (default)

---

[3] local 10.40.18.78 port 60925 connected with 10.40.18.9 port 5001

[ID] Interval Transfer Bandwidth

[3] 0.0-10.0 sec 1097 MBytes 110 MBytes/sec

Note that all the parameters are now in MBytes instead of Mbits.

The same can be performed at the server side.

Bi-directional bandwidth measurement: (-r argument)

The Iperf server connects back to the client allowing the bi-directional bandwidth measurement. By default, only the bandwidth from the client to the server is measured.

When using -r, the bandwidths are tested sequentially. That is, it first checks the communication properties of the connection from client to server, then from server to client.

iperf -c <*server ip*> -r

Simultaneous bi-directional bandwidth measurement: (-d argument)

To measure the bi-directional bandwidths simultaneously, use the -d argument.

iperf -c <*server ip*> -d

TCP Window size: (-w argument)

The TCP window size can be between 2 and 65,535 bytes. The following sets the window size to 200 bytes.

iperf -c <*server ip*> -w 200

Communication port (-p), timing (-t) and interval (-i):

The Iperf server communication port can be changed with the -p argument. It must be configured on the client and the server with the same value, default is TCP port 5001.

The -t argument specifies the test duration time in seconds, default is 10 secs.

The -i argument indicates the interval in seconds between periodic bandwidth reports.

iperf -c <*server ip*> -p 12000 -t 20 -i 2

UDP tests: (-u), bandwidth settings (-b)

The jitter is basically the latency variation and does not depend on the latency. The UDP tests with the -u argument will give invaluable information about the jitter and the packet loss. If you don't specify the -u argument, Iperf uses TCP.

To keep a good link quality, the packet loss should not go over 1%. A high packet loss rate will generate a lot of TCP segment retransmissions which will affect the bandwidth.

The -b argument allows the allocation if the desired bandwidth.

iperf -c <*server ip*> -u -b 10m

In the server:

iperf -s -u -i 1

Maximum Segment Size (-m argument) display:

The Maximum Segment Size (MSS) is the largest amount of data, in bytes, that a computer can support in a single, unfragmented TCP segment.

It can be calculated as follows:

MSS = MTU - TCP & IP headers

The TCP & IP headers are equal to 40 bytes.

The MTU or Maximum Transmission Unit is the greatest amount of data that can be transferred in a frame.

Here are some default MTU size for different network topology:

Ethernet - 1500 bytes: used in a LAN.

PPPoE - 1492 bytes: used on ADSL links.

Token Ring (16Mb/sec) - 17914 bytes: old technology developed by IBM.

Dial-up - 576 bytes

Generally, a higher MTU (and MSS) brings higher bandwidth efficiency

iperf -c <*server ip*> -m

Maximum Segment Size (-M argument) settings:

Use the -M argument to change the MSS. (See the previous test for more explanations about the MSS)

iperf -c <*server ip*> -M 1300 -m

Parallel tests (-P argument):

Use the -P argument to run parallel tests.

iperf -c <*server ip*> -P 2

Iperf help:

iperf -h