Software Defined Network
SDN

Ch 6

Data Traffic Test

Traffic Test

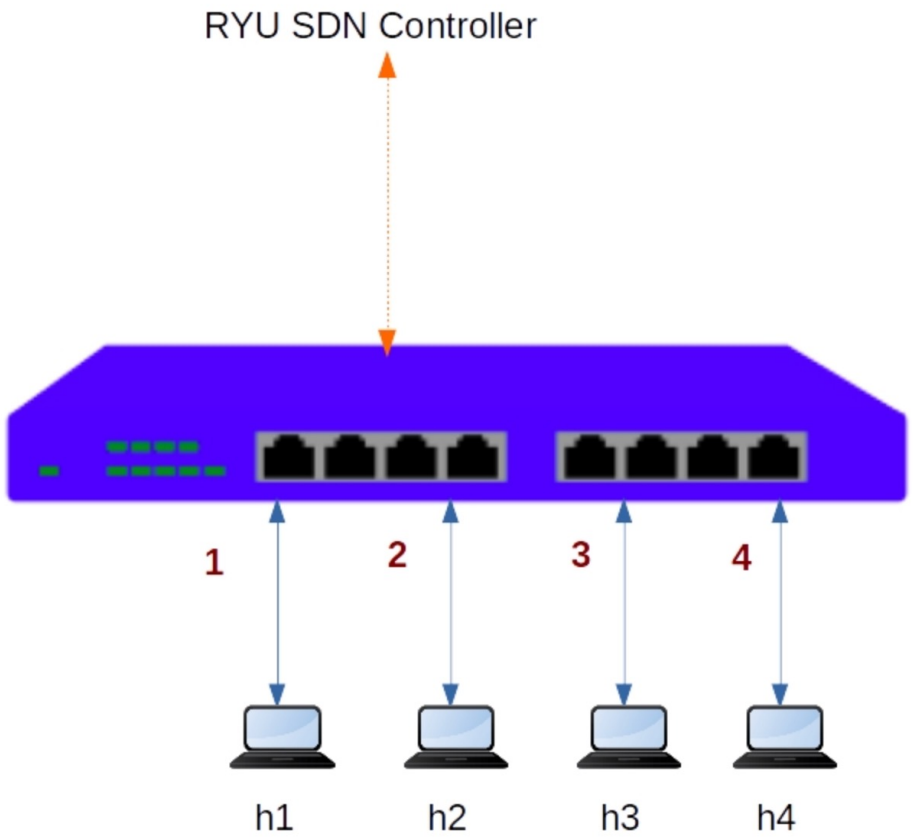
1. Introduction

Usually we need to perform various traffic tests between the hosts to test our network topology & SDN application performance.

Some of the test includes:

1. TCP Tests between two hosts
2. UDP Tests
3. UDP Tests with different Packet size (64 bytes, 1024 bytes etc), different packet rate (50 pkts/s, 100 pkts/s etc)
4. Variable Traffic rate (10Pkts/s for first 60s, then 100Pkts/s for rest of the test)
5. Multiple/Parallel Streams/Sessions
6. VoIP Traffic Test
7. Video Streaming mp4 video file will be used as video source. and will generate the video stream using VLC, and will be received by client using VLC Video Player.

Traffic Test



TCP Tests

IPERF is widely accepted traffic generation tool to perform TCP Tests.

- TCP tests generally used for measure the bandwidth.
- Latency, Jitter, Packet loss cannot be measured using TCP Tests.

For all our traffic tests, we will use Simple Topology and l4 switch application.

1. Create the topology

```
sudo mn --controller=remote,ip=127.0.0.1 --mac --switch=ovsk,protocols=OpenFlow13 --topo=single,4
```

2. Run the RYU Controller l4 switch applicaion.

```
ryu-manager l4_switch.py
```

TCP Tests

3. In mininet shell, do ping h1 to h2.

```
mininet>h1 ping -c 5 h2
```

4.check the flows

```
sudo ovs-ofctl -O OpenFlow13 dump-flows s1
```

5. understand the Host & Switch Port numbers.

```
mininet>links
mininet>ports
```

6. To check the switch statistics

```
sudo ovs-ofctl -O OpenFlow13 dump-ports s1
```

Now all set to run our traffic tests.

TCP Tests

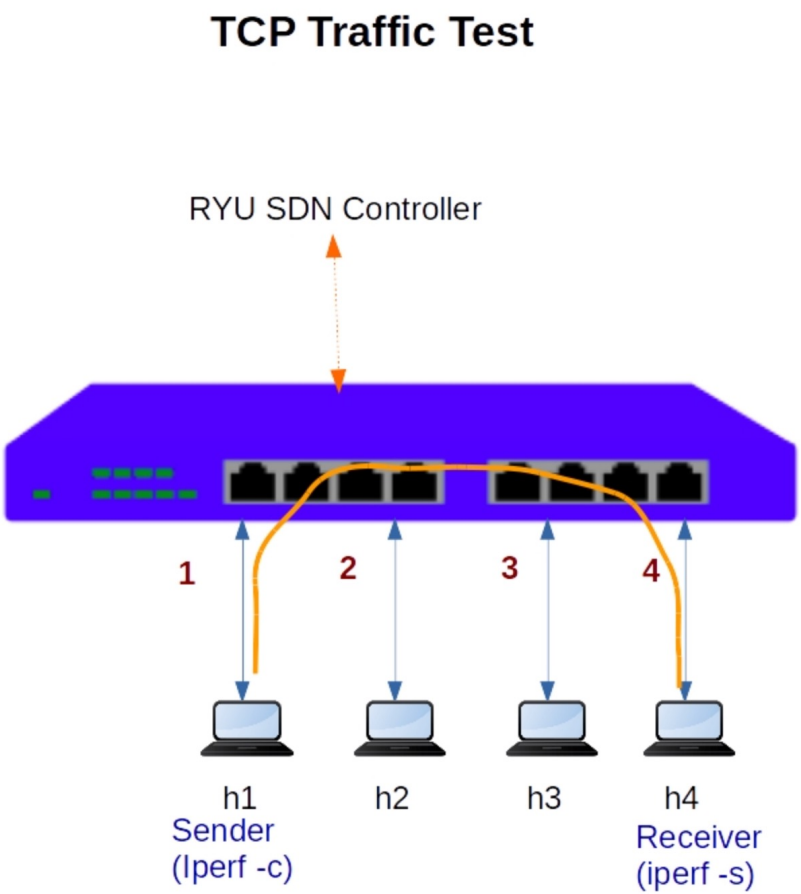
A. Traffic test from h1 to h4

Objective: Generate TCP Traffic from h1 to h4. (Measure bandwidth from h1 to h4)

h1 is sender.

h4 is receiver.

TCP Tests


TCP Traffic Test
RYU SDN Controller
1
2
3
4
h1
Sender
(Iperf -c)
h2
h3
h4
Receiver
(iperf -s)

TCP Tests

1. Start IPERF Server in h4

```
h4 iperf -s &
```

2. Start the IPERF Client in h1 and connecting to h4

```
h1 iperf -c h4
```

3. Analyze the results by flows

```
sudo ovs-ofctl -O OpenFlow13 dump-flows s1
```

we observe traffic in both directions.

h1 to h4 traffc is very high (in Gbps). This is data traffic. h4 to h1 traffic is very less. This is TCP Acknowledge traffic.

TCP Tests

4. Analyze the results by ports.

```
sudo ovs-ofctl -O OpenFlow13 dump-ports s1
```

h1 --------port1, port4--------h4

Forward traffic:

- h1 transimits. port1 receives.
- port4 transmits, h4 receives.

Acknowlege:

- h4 transmits. port4 receives.
- port1 transmits. h1 receives.

TCP Tests

B. Bidirectional Traffic test h1 to h4(sequentially).

1. Start IPERF Server in h4

```
h4 iperf -s &
```

2. Start the IPERF Client in h1 and connecting to h4

```
h1 iperf -c h4 -r
```

Its sequential test , once h1 to h4 traffic test is completed it will start h4 to h1 traffic test.

TCP Tests

C. Bidirectional Traffic test h1 to h4(parallel).

1. Start IPERF Server in h4

```
h4 iperf –s &
```

2. Start the IPERF Client in h1 and connecting to h4

```
h1 iperf –c h4 –d
```

Its parallel test , both direction h1 to h4 as well h4 to h1 traffic tests started.

TCP Tests

D. Traffic test from h1 to h4 with Multiple Sessions.

1. Start IPERF Server in h4

```
h4 iperf -s &
```

2. Start the IPERF Client in h1 and connecting to h4

```
h1 iperf -c h4 -P 5
```

UDP Tests with IPERF

UDP Tests are quite flexible in the nature of playing around packet sizes(64 bytes) and number of packets(10 Packets/s) you want to send. It means, user can control the bandwidth of UDP Traffic.

- Bandwidth, Latency, Jitter, Packet loss can be measured using UDP Tests.

IPERF doesnot provide provide much flexibility in UDP Tests.

1. Start IPERF UDP Server in h4

```
h4 iperf –u –s &
```

2. Start the IPERF Client in h1 and connecting to h4 and generate bandwidth of 10Mbsp

```
h1 iperf –u –c h4 –b 10m
```

UDP Tests with MGEN

Multi-Generator (MGEN) is open source traffic gen software

https://downloads.pf.itd.nrl.navy.mil/docs/mgen/mgen.html

Some important features

- variable packet rate
- pattern(Periodic, possion, burst, jitter)
- logs
- scriptable

It doesnot give test report (similar to iperf). we need to process the logs and prepare it.

Installation

```
sudo apt-get install mgen
```

UDP Tests with MGEN

A. Simple MGEN UDP Test (Variable Packet rate)

Objective:

we want to generate 10 packets per second for first 0th to 100 seconds. then will send 100 packets per secondfor 100th to 200 seconds. packet size is 1024 bytes.

receive.mgn

```
0.0 LISTEN UDP 5000-5001
```

send.mgn

```
0.0 ON 1 UDP SRC 5001 DST 10.1.1.3/5001 PERIODIC [10 1024]
100.0 MOD 1 PERIODIC [100 1024]
200.0 OFF 1
```

How to run mgen:

The default/minimum syntax is below,

> *mgen input script name*

UDP Tests with MGEN

Testing

1. Start MGEN Receiver in h3

```
h3 mgen input receive.mgn output mgenlog.txt &
```

2. Start the MGEN sender in h1

```
h1 mgen input send.mgn &
```

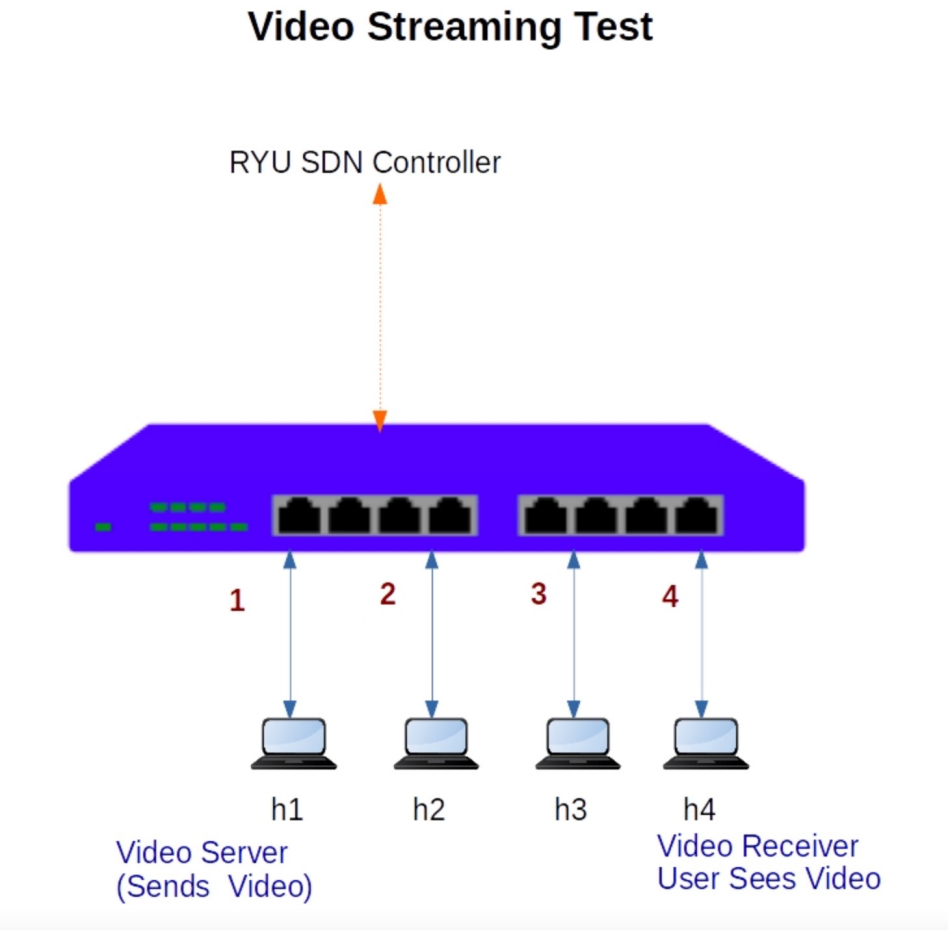
Video Stream Traffic Testing

we use VLC media player(https://www.videolan.org/index.html) for Video Streaming testing.

installation procedure

```
sudo apt-get install vlc vlc-nox
```

Video Stream Traffic Testing



Video Stream Traffic Testing

Testing

1. Create the topology

```
sudo mn --controller=remote,ip=127.0.0.1 --mac --switch=ovsk,protocols=OpenFlow13 --topo=single,4
```

2. Run the RYU Controller l4 switch applicaion.

```
ryu-manager l4_switch.py
```

3. Download the sample video from internet (or use your own video file)

4. h4 is a Video receiver. Start the video receiver in h4 xterm shell

```
mininet> xterm h4


vlc rtp://@10.0.0.4:9001
```

Video Stream Traffic Testing

5. h1 is video sender. Start the video streaming sender in h1 xterm shell

```
mininet> xterm h1

cvlc  /home/sureh/cat3.mp4  --sout '#rtp{proto=udp,mux=ts,dst=10.0.0.4,port=9001}'
```

VOIP Tests

VOIP Traffic is UDP Stream. This can be simulated thru IPERF UDP Tests.

Objective

1. Test the single Voip call 64Kbps
2. Test parallel voip calls

Testing

1. Create the topology

```
sudo mn --controller=remote,ip=127.0.0.1 --mac --switch=ovsk,protocols=OpenFlow13 --topo=single,4
```

2. Run the RYU Controller l4 switch applicaion.

```
ryu-manager l4_switch.py
```

VOIP Tests

A. Single 64Kbps VOICE CALL Test

start the 64Kbps VOIP Traffic Test between h1 to h4 for 60 seconds ,and vice versa.

1) Run the IPERF UDP server in h4

In mininet CLI

```
mininet>h4 iperf --server --udp --len 300 --tos 184 -fk --interval 5 &
```

Here we are starting udp server, and setting the TOS field to 184

2) Run the IPERF UDP Client in h1

In mininet CLI

```
h1 iperf -c 10.1.1.4 --udp --len 300 --bandwidth 67000 --dualtest --tradeoff --tos 184 -fk --interval 5
--time 60 --listenport 5002
```

VOIP Tests

B. Multiple Parallel calls VOIP calls test

1) Run the IPERF UDP server in h4

```
mininet>h4 iperf --server --udp --len 300 --tos 184 -fk --interval 5 --parallel 4
```

2) Run the IPERF UDP Client in h1

```
mininet>iperf -c 10.1.1.4 --udp --len 300 --bandwidth 67000 --dualtest --tradeoff --tos 184 -fk --
interval 5 --time 60 --listenport 5002 --parallel 4
```

Traffic Tests with DITG

D-ITG (Distributed Internet Traffic Generator) is a platform capable to produce traffic at packet level accurately replicating appropriate stochastic processes for both IDT (Inter Departure Time) and PS (Packet Size) random variables (exponential, uniform, cauchy, normal, pareto, ...).

D-ITG supports both IPv4 and IPv6 traffic generation and it is capable to generate traffic at network, transport, and application layer.

```
sudo apt-get install d-itg
```

Once installed, Quick verification

```
suresh@suresh-Latitude-6430U:~$ ITGSend
ITGSend version 2.8.1 (r1023)
Compile-time options: sctp dccp bursty multiport

Missing argument!!!
Try ITGSend -h or --help for more information
```

Traffic Tests with DITG

Quick Start

ITGRecv command for receiver

ITGSend command for sender

ITGSend/Recv generate the logs. which needs to be ananlzed by **ITGDec** command to see the result.

UDP

1. start the ryu l4_switch app

```
ryu-manager l4_switch.py
```

2. run the mininet topology

sudo mn --controller=remote,ip=127.0.0.1 --mac -i 10.1.1.0/24 --switch=ovsk,protocols=OpenFlow13 --topo=single,4

Traffic Tests with DITG

3. start the receiver (or server) on h2

```
xterm h2
ITGRecv -l /tmp/receiver.log
```

Here the log file is stored in /tmp/receiver.log, which can be decoded via ITGDec.

4. start the sender

```
h1 ITGSend -T UDP -a h2 -c 100 -C 10 -t 15000
```

The options are explained below,

```
-t 15000   is test duration in milliseconds  (15 seconds)

-a h2    target ip

-c 100   packet size

-C 10    packet per second (pps)

-T UDP   Protocol - UDP/TCP/
```

Traffic Tests with DITG

5. verify the flows:

You can observe 1 UDP flows and 2 TCP flows(server port 9000). Those TCP flows are used for communicating between ITGSend and ITGRecv for control information.

$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
OFPST_FLOW reply (OF1.3) (xid=0x2):
cookie=0x0, duration=4.876s, table=0, n_packets=5, n_bytes=358,
priority=1,tcp,nw_src=10.1.1.1,nw_dst=10.1.1.2,tp_src=60580,tp_dst=9000 actions=output:2
cookie=0x0, duration=4.874s, table=0, n_packets=3, n_bytes=206,
priority=1,tcp,nw_src=10.1.1.2,nw_dst=10.1.1.1,tp_src=9000,tp_dst=60580 actions=output:1
cookie=0x0, duration=4.871s, table=0, n_packets=45, n_bytes=6390,
priority=1,udp,nw_src=10.1.1.1,nw_dst=10.1.1.2,tp_src=39783,tp_dst=8999 actions=output:2
cookie=0x0, duration=47.834s, table=0, n_packets=19, n_bytes=1554, priority=0 actions=CONTROLLER:65535
(ry) suresh@suresh-Latitude-6430U:~/projects/ditg$

Traffic Tests with DITG

5. To Check the result

```
cd /tmp

ITGDec receiver.log

(ry) suresh@suresh-Latitude-6430U:~/projects/ditg$ ITGDec /tmp/receiver.log
ITGDec version 2.8.1 (r1023)
Compile-time options: sctp dccp bursty multiport
\------------------------------------------------------------
Flow number: 1
From 10.1.1.1:39783
To    10.1.1.2:8999
------------------------------------------------------------
Total time                =     14.923949 s
Total packets             =           150
Minimum delay             =      0.000057 s
Maximum delay             =      0.001506 s
Average delay             =      0.000162 s
Average jitter            =      0.000040 s
Delay standard deviation  =      0.000118 s
Bytes received            =         15000
Average bitrate           =      8.040767 Kbit/s
Average packet rate       =     10.050959 pkt/s
Packets dropped           =             0 (0.00 %)
Average loss-burst size   =      0.000000 pkt
------------------------------------------------------------
```

Traffic Tests with DITG

```
_____________________________________________________________
****************  TOTAL RESULTS   ******************

_____________________________________________________________
Number of flows          =            1
Total time               =     14.923949 s
Total packets            =          150
Minimum delay            =      0.000057 s
Maximum delay            =      0.001506 s
Average delay            =      0.000162 s
Average jitter           =      0.000040 s
Delay standard deviation =      0.000118 s
Bytes received           =         15000
Average bitrate          =      8.040767 Kbit/s
Average packet rate      =     10.050959 pkt/s
Packets dropped          =             0 (0.00 %)
Average loss-burst size  =             0 pkt
Error lines              =             0
_____________________________________________________________
```

The result contain two parts. First part is flow specific,

we ran only one UDP flow, this is mapped with FLOW Number. if we have multiple flows, we can see each flow result.

Second Part is , as a Whole receiver summary.

6. Stop the ITGRecv on h2

Traffic Tests with DITG

UDP with three flows(Multiflow mode).

Syntax is below

```
ITGSend <script_file> [log_opts]
```

1. Create a script file /tmp/send.sh

```
-T UDP -a 10.1.1.2 -c 100 -C 30 -t 15000
-T UDP -a 10.1.1.2 -c 500 -C 20 -t 20000
-T UDP -a 10.1.1.2 -c 1024 -C 10 -t 25000
```

Copy

Here we generate 3 flows

1st flow generates 30 pkts/per second, with packet size(payload) 100 bytes for 15 seconds

2nd flow generates 20 pkts/per second, with packet size(payload) 500 bytes for 20 seconds

3rd flow generates 10 pkts/per second, with packet size(payload) 1024 bytes for 25 seconds

Traffic Tests with DITG

2. start the receiver (or server) on h2

```
xterm h2
ITGRecv -l /tmp/receiver1.log
```

Here the log file is stored in /tmp/receiver.log, which can be decoded via ITGDec.

3. start the sender

```
h1 ITGSend  /tmp/send.sh

Logs:
mininet> h1 ITGSend /tmp/send.sh
ITGSend version 2.8.1 (r1023)
Compile-time options: sctp dccp bursty multiport
Started sending packets of flow ID: 1
Started sending packets of flow ID: 2
Started sending packets of flow ID: 3
Finished sending packets of flow ID: 1

Finished sending packets of flow ID: 2

Finished sending packets of flow ID: 3
```

Copy

Traffic Tests with DITG

4. verify the flows:

you can see 3 UDP flows and 2 TCP Flows (control channel)

5. stop the receiver in h2

6. result verification

we see 3 flows, each flow represents the one which we sent.

Traffic Tests with DITG

```
$ ITGDec /tmp/receiver1.log
ITGDec version 2.8.1 (r1023)
Compile-time options: sctp dccp bursty multiport
/------------------------------------------------------------
Flow number: 1
From 10.1.1.1:58990
To    10.1.1.2:8999
------------------------------------------------------------
Total time                =     14.984081 s
Total packets             =           449
Minimum delay             =      0.000056 s
Maximum delay             =      0.004455 s
Average delay             =      0.000173 s
Average jitter            =      0.000041 s
Delay standard deviation  =      0.000207 s
Bytes received            =         44900
Average bitrate           =     23.972107 Kbit/s
Average packet rate       =     29.965134 pkt/s
Packets dropped           =             0 (0.00 %)
Average loss-burst size   =      0.000000 pkt
------------------------------------------------------------
```

Traffic Tests with DITG

```
------------------------------------------------------------
Flow number: 2
From 10.1.1.1:39348
To    10.1.1.2:8999
------------------------------------------------------------
Total time                =     19.994380 s
Total packets             =           400
Minimum delay             =      0.000056 s
Maximum delay             =      0.004048 s
Average delay             =      0.000174 s
Average jitter            =      0.000053 s
Delay standard deviation  =      0.000199 s
Bytes received            =        200000
Average bitrate           =     80.022486 Kbit/s
Average packet rate       =     20.005622 pkt/s
Packets dropped           =             0 (0.00 %)
Average loss-burst size   =      0.000000 pkt
------------------------------------------------------------
```

Traffic Tests with DITG

```
------------------------------------------------------------
Flow number: 3
From 10.1.1.1:60097
To    10.1.1.2:8999
------------------------------------------------------------
Total time                =     24.938133 s
Total packets             =           235
Minimum delay             =      0.000058 s
Maximum delay             =      0.003840 s
Average delay             =      0.000189 s
Average jitter            =      0.000054 s
Delay standard deviation  =      0.000247 s
Bytes received            =        240640
Average bitrate           =     77.195835 Kbit/s
Average packet rate       =      9.423320 pkt/s
Packets dropped           =            15 (6.00 %)
Average loss-burst size   =      3.000000 pkt
------------------------------------------------------------
```

Traffic Tests with DITG

```
_____________________________________________________________
****************  TOTAL RESULTS   ******************

_____________________________________________________________
Number of flows          =            3
Total time               =     24.953222 s
Total packets            =          1084
Minimum delay            =      0.000056 s
Maximum delay            =      0.004455 s
Average delay            =      0.000177 s
Average jitter           =      0.000056 s
Delay standard deviation =      0.000214 s
Bytes received           =        485540
Average bitrate          =    155.664066 Kbit/s
Average packet rate      =     43.441284 pkt/s
Packets dropped          =            15 (1.36 %)
Average loss-burst size  =      0.000000 pkt
Error lines              =            0
_____________________________________________________________
```

Traffic Tests with DITG

TCP Test

DITG TCP Test is more powerful, we can configure the Packet Size, Number of packets/s etc.

1. start the receiver (or server) on h2

```
xterm h2
ITGRecv -l /tmp/receiver1.log
```

2. start the sender

```
h1 ITGSend -T TCP  -a h2  -t 15000
h1 ITGSend -T TCP -a h2 -c 1000 -C 100 -t 15000
```

3. check the logs as above test.

Traffic Tests with DITG

Packet distribution (Rate & Size)

Inter-departure time options (Time factor)

Constant distribution

-C rate Constant

```
h1 ITGSend –T TCP –a h2 –c 1000 –C 100 –t 15000
```

Uniform distribution

-U min_rate max_rate Uniform distribution.

```
h1 ITGSend –T TCP –a h2 –c 1000 –U 10 100 –t 15000
```

Traffic Tests with DITG

Poisson distribution

-O mean Poisson distribution.

```
h1 ITGSend -T TCP -a h2 -c 1000 -O 50 -t 15000
```

Packet size variation

-c pkt_size Constant (default: 512 bytes).

-u min_pkt_size max_pkt_size Uniform distribution.

-o mean Poisson distribution.

```
h1 ITGSend -T TCP -a h2 -c 1000 -O 50 -u 10 1000 -t 15000
h1 ITGSend -T TCP -a h2 -c 1000 -O 50 -o 100 -t 15000
```

Traffic Tests with DITG

VOIP,Telnet,DNS

For VOIP, Telnet, DNS Applications, Receiver method will be same as above examples. only sender option will change as below

1. VOIP

```
h1 ITGSend  -t 15000 -a h2 -rp 10001 VoIP -x G.711.2 -h RTP -VAD
```

2. Telnet

```
h1 ITGSend -t 15000 -a h2 -rp 10002 Telnet
```

3. DNS

```
h1 ITGSend -t 15000 -a h2 -rp 10003 DNS
```

Traffic Tests with DITG

3. FTP

Read payload from file

udp

```
h1 ITGSend  -a h2 -Fp test.dat
```

tcp

```
h1 ITGSend  -T TCP -a h2 -Fp test.dat
```

Traffic Tests with DITG

Video streaming

1. receiver

```
xterm h2
ITGRecv -l /tmp/receiver1.log
```

2. Sender

Script file (/tmp/send.sh)

```
-C 24 -a 10.1.1.2 -n 27791 6254 -t 720000  -T UDP -m rttm -sp 10101 -rp 10001
-C 24 -a 10.1.1.2 -n 27791 6254 -t 720000  -T UDP -m rttm -sp 10102 -rp 10002
```

Option details are below -C 24 pkts/second -n 27791 6254 (normal distribution options - packet size) -t 720000 (test time 720s) -m rttm (meter round-trip time meter) -sp 10101 srcport -rp 10001 receiver port -H NAT Traversal

```
h1 ITGSend  /tmp/send.sh
```