

Summary

1. ITGSend
 - 1.1.Synopsis
 - 1.2.Options
 - 1.3.Type of meter
 - 1.4.Log File
 - 1.5.Remote Log
 - 1.6.Receiver Remote Log
 - 1.7.Application Level Protocols Options
 - 1.8.Script file
2. ITGRecv
 - 2.1.Synopsis
 - 2.2.Options
 - 2.3.Log File
 - 2.4.Remote Log
 - 2.5.Application Level Protocol Options
 - 2.6.Script file
3. ITGLog
 - 3.1.Synopsis
4. ITGDec
 - 4.1.Synopsis
 - 4.2.Options
5. ITGApi
6. WB::D-ITG Web Based Distributed Internet Traffic Generator
7. Getting started
 - 7.1. Command line usage examples
 - 7.2. Script file usage examples

1 - ITGSend

1.1 Synopsis:

In case of using a script file to generate multiple flows, type:

```
./ITGSend <script_file> [[-l <logfile>] | [-l <logfile>  
-L <log_server_addr> <protocol_type>]] [-Z <log_server_addr>  
<protocol_type> <receiver_logfile> ]
```

See Section 1.7 for details

If you want to remotely control the sender, launch it in daemon mode (see Section 4 for details):

```
./ITGSend -Q [[-l <logfile>] | [-l <logfile> -L <log_server_addr>  
<protocol_type>]] [-Z <log_server_addr> <protocol_type>  
<receiver_logfile> ]
```

Otherwise:

```
./ITGSend [-m <msr_type>] [-a <destination_address>] [-p  
<destination_port>] [-T <protocol_type>] [-f <TTL>] [-b <DS byte>]  
[-s <seed>] [[-l <logfile>] | [-l <logfile> -L <log_server_addr>  
<protocol_type>]] [-Z <log_server_addr> <protocol_type>  
<receiver_logfile> ] [-t <duration>] [-d <gen_delay>]  
[[ -C <pkts_per_sec> | -U <min_pkts_per_sec max_pkts_per_sec> |  
-E <average_pkts_per_sec> | -V <shape scale> | -Y <shape scale> | -N  
<mean std_dev> | -O <average_pkts_per_sec> | -G <shape scale>]  
[-c <pkt_size> | -u <min_pkt_size max_pkt_size> |  
-e <average_pkt_size> | -v <shape scale> | -y <shape scale> |  
-n <mean std_dev> | -o <average_pkt_size> | -g <shape scale>]] |  
[[Telnet][DNS][VoIP] [-x <codec_type>] [-h <protocol_type>  
-i <Voice_Activity_Detection>]]]
```

1.2 Options:

-m	type of meter See 1.3 for details
-a	destination address DEFAULT: localhost
-p	destination port DEFAULT: 8999
-T	protocol type VALUES: UDP, TCP, ICMP DEFAULT: UDP

If you choose ICMP you must specify the type of message.
Root privileges are needed under Linux.

-b	DS byte DEFAULT: 0	Set the DS byte for QoS tests. The value is interpreted as a decimal number, or as an hexadecimal number if the prefix “0x” is used. $DS \in [0, 255]$. (Note: DS option is disabled under Windows 2000 and XP, according to “Microsoft Knowledge Base Article – 248611” http://support.microsoft.com/default.aspx?scid=kb;EN-US;q248611)
-s	seed DEFAULT: Random	Set the seed for random number generator
-t	duration DEFAULT: 10000 msec	Set the generation duration. It's expressed in msec.
-d	gen_delay DEFAULT: 0 msec	Set the generation delay. It's expressed in msec.
-f	TTL DEFAULT: 64	Set the time to live (TTL). It's expressed in decimal notation. $TTL \in [0, 255]$.
-l	logfile See 1.4 for details.	Generate log file.
-L	remote log See 1.5 for details	
-Z	receiver remote log See 1.6 for details	

Inter-departure time options:

-C	pkts_per_sec	- Constant inter-departure with specified packet rate.
-U	min_pkts_per_sec max_pkts_per_sec	- Uniformly distributed inter-departure
-E	average_pkts_per_sec	- Exponentially distributed inter-departure
-V	shape scale	- Pareto distributed inter-departure
-Y	shape scale	- Cauchy distributed inter-departure
-N	mean std_dev	- Normal distributed inter-departure
-O	average_pkts_per_sec	- Poisson distributed inter-departure
-G	shape scale	- Gamma distributed inter-departure

NOTE: If you don't specify any inter_departure time option the default behaviour is: Constant inter-departure with 1000 packets per second.

Packet size options:

-c	pkt_size	- Constant payload size
----	----------	-------------------------

-u	min_pkt_size max_pkt_size	- Uniformly distributed payload size
-e	average_pkt_size	- Exponentially distributed payload size
-v	shape scale	- Pareto distributed payload size
-y	shape scale	- Cauchy distributed payload size
-n	mean std_dev	- Normal distributed payload size
-o	average_pkt_size	- Poisson distributed payload size
-g	shape scale	- Gamma distributed payload size

NOTE: If you don't specify any packet size option the default behaviour is: Constant payload size c = 512 bytes

Application Level protocol indication:

Telnet	Generate traffic with Telnet traffic characteristics. No option is required. NOTE: Telnet traffic generation works with (i) OWDMeter and (ii) TCP transport layer protocol. Different settings will be ignored.
VoIP	Generate traffic with VoIP traffic characteristics. See 1.7 for further details. NOTE: VoIP traffic generation works with (i) OWDMeter and (ii) UDP transport layer protocol. Different settings will be ignored.
DNS	Generate traffic with DNS traffic characteristics. No option is required. NOTE: DNS traffic generation works with (i) OWDMeter using (ii) both UDP and TCP transport layer protocol. Different settings will be ignored.

NOTE: If you specify an application level protocol then you cannot specify any inter-departure time or packet size option. If you want to specify an application level protocol you must indicate it after any other options. Only the other options illustrated in Section 1.2 are allowed.

1.3 Type of meter:

VALUES: owdm (one way delay meter)
rttm (round trip time meter)

DEFAULT: owdm

In case of rttm, the sender log file is mandatory. You can specify the log file name using the option -l. The default name is ITGSend.log.

1.4 Log File:

log_file generates the log file

If the type of meter is OWDM, by omitting this option you will not generate log file. Otherwise, if the type of meter is RTTM, by omitting this option the program will generate a default log file as indicated below.

DEFAULT: /tmp/ITGSend.log

1.5 Remote Log :

<log_address> log server IP address
DEFAULT: localhost

<protocol_type> protocol used for communication between sender and log server
VALUES: UDP, TCP
DEFAULT: UDP

1.6 Receiver Remote Log:

The option -Z enables to remotely configure a log server for the ITGRecv receiver.

<log_address> log server IP address
DEFAULT: localhost

<protocol_type> protocol used for communication between receiver and log server
VALUES: UDP, TCP
DEFAULT: UDP

<receiver_logfile> receiver log file name

1.7 Application Level Protocols Options:

VoIP options:

-x	codec_type	Set the Codec type
	VALUES:	G.711.1 for G.711 codec with 1 sample per pkt
		G.711.2 for G.711 codec with 2 samples per pkt
		G.723.1 for G.723.1 codec
		G.729.2 for G.729 codec with 2 samples per pkt
		G.729.3 for G.729 codec with 3 samples per pkt

DEFAULT: G.711 with 1 sample per pkt

-h	protocol_type	Set the protocol type
	VALUES:	RTP for Real Time Protocol CRTP for Real Time Protocol with header compression
	DEFAULT:	RTP
-i	Voice_Activity_Detection	Set the Voice Activity Detection
	VALUES:	VAD for Voice Activity Detection on
	DEFAULT:	Voice Activity Detection off

In case of VoIP traffic generation, the indication of the options presented in Section 1.2 can not follow the indication of the VoIP options. Commands to generate VoIP traffic look like the following:

`./ITGSend [0 or more options present in the Section 1.2] VoIP [0 or more VoIP options]`

In no option is provided, the generation starts with default parameters (see example 1).

Examples:

- 1) `./ITGSend VoIP`
- 2) `./ITGSend -l logfilename -t 30000 -d 5000 VoIP -x G.711.2 -h CRTP -i VAD`
- 3) `./ITGSend -l logfilename -t 30000 -p 6666 -b 128 VoIP`

1.8 Script file:

By using the script mode for the ITGSend sender, it is possible to simultaneously generate several flows. Each flow is managed by a single thread, with a separate thread acting as a master and coordinating the other threads. To generate n flows, the script file must be made of n lines, each of which used to specify the characteristics of one flow. Each line can contain all the options specified in Section 1.2, but those regarding the logging process (-l, -L, -Z). Such options can be specified at the command line and will refer to all the flows.

2 - ITGRecv

2.1 Synopsis:

```
./ITGRecv [[-l <logfile>] | [-l <logfile> -L <log_server_addr>
<protocol _type>]]
```

2.2 Options:

-l	logfile See 2.3 for details.	Generate log file.
-L	remote log See 2.4 for details	

2.3 Log File:

log_file generates the log file

If the type of meter is RTTM, by omitting this option you will not generate a log file. Otherwise, if the type of meter is OWDM, by omitting this option the program will generate a default log file as indicated below.

DEFAULT: /tmp/ITGRecv.log

2.4 Remote Log :

<log_address> log server IP address
DEFAULT: localhost

<protocol_type> protocol used for communication between receiver and log server
VALUES: UDP, TCP
DEFAULT: UDP

3 - ITGLog

3.1 Synopsis:

`./ITGLog`

ITGLog is the log server, which receives log information from the ITGSend sender and the ITGRecv receiver. ITGLog listens on ports dynamically allocated in the range [9000-10000].

4 - ITGDec

4.1 Synopsis:

```
./ITGDec [ <logfile> [-v | -i] [-l <text_log_file>]
[-d <delay_interval_size>] [-j <jitter_interval_size>]
[-b <bitrate_interval_size> | -p <number_of_packets>] ] |
[ -h | --help] ]
```

The ITGDec decoder is the utility to analyze the results of the experiments conducted by using the D-ITG generation platform. ITGDec parses the log files generated by ITGSend and ITGRecv and calculates the average values of bitrate, delay and jitter either on the whole duration of the experiment or on variable-sized time intervals.

4.2 Options

-h	prints help
--help	prints help
-v	generates synthetic results for visualization
-i	generates synthetic results for import
-l <text_log_file>	generates decoded log file with name <text_log_file>
-d <delay_interval_size>	generates a file with the average delay on <delay_interval_size> millisecs windows
-j <jitter_interval_size>	generates a file with the average jitter on <jitter_interval_size> millisecs windows
-b <bitrate_interval_size>	generates a file with the average bitrate on <bitrate_interval_size> millisecs windows
-p <number_of_packet>:	generates a file with the average bitrate every <number_of_packet> packets

5 - ITGapi

ITGapi is a C++ API that enables to remotely control traffic generation. For this purpose, after having launched ITGSend in daemon mode (./ITGSend -Q) on one or more traffic source nodes, it is possible to use the following function to remotely coordinate the traffic generation from the different senders:

```
int DITGsend(char *sender, char *message);
```

sender is the IP address of ITGSend and message is the string you would type at command line (except the name of the ITGSend executable file). Returns 0 in case of success, -1 otherwise. ITGSend, when used in daemon mode, sends messages back to the application that issued the generation of the traffic flow. Two types of messages are used, one to acknowledge the start of the generation process and the other to signal its end. The manager application is able to catch those messages by using the function:

```
int catchManagerMsg(char **senderIP, char **msg);
```

the return value is -1 in case no message arrived (the function is non blocking), 1 to indicate the start of the flow and 2 to indicate the end of the flow; senderIP is a pointer to a string containing the IP address of the sender that sent the message and msg is a pointer to a string containing the command that the sender received.

These prototypes are declared in ITGapi.h

ITGManager.cpp is an example of application to remotely control the generation of traffic. To compile it, compile first ITGapi.cpp:

```
make -f Makefile.{Windows | Linux} ITGapi.o  
g++ ITGManager.cpp ITGapi.o -o ITGManager
```

6 – WB::D-ITG: Web Based Distributed Internet Traffic Generator

We are working on a Web Based version of D-ITG. Currently we are testing it before the official releasing. It will be soon available at D-ITG web site.

7 – Getting started

7.1 Command line generation

In the simplest case, you can generate just one flow and you can do it from the command line:

Example 1: Single UDP flow with constant inter-departure time between packets and constant packets size

1. start the receiver on the destination host (say it B):
.
./ITGRecv
2. start the sender on the source host (say it A):
./ITGSend -a B -p 9500 -C 100 -c 500 -t 20000

The resulting flow from A to B has the following characteristic:

- the destination port is 9500
- 100 packets per second are sent (with constant inter-departure time between packets)
- the size of each packet is equal to 500 bytes
- the duration of the generation experiment is 20 seconds (20000 milliseconds)

Example 2: Single TCP flow with constant inter-departure time between packets and uniformly distributed packet size between 500 and 1000 bytes with local sender/receiver log

1. start receiver on the destination host (10.0.0.3)
[donato@catarella tmp] ITGRecv -l recv_log_file
2. start the sender on the source host
[donato@otto donato]\$./ITGSend -a 10.0.0.3 -p 9501 -C 1000
-u 500 1000 -l send_log_file
3. close the ITGRecv by pressing Ctrl-C
4. decode the receiver log file on the destination host:

```
[donato@catarella tmp]ITGDec recv_log_file
-----
Flow Id : 1
-----
From          10.0.0.4:32810  ---> To          10.0.0.3:9501

Total time      = 9.999808 sec
Total packet    = 9958
Max delay       = 7288685.629000 msec
Min delay       = 7288636.261000 msec
Average delay    = 7288637.283738 msec
Average jitter   = 732.156510 msec
Delay variation  = 4.748347 msec
Byte received   = 7481719
Average bitrate  = 5985.490121 Kbps
Average packet rate = 995.819120 pkt/sec
packets dropped  = 42
-----
Total results
-----
Total number of flows = 1
Max delay             = 7288685.629000 msec
Min delay             = 7288636.261000 msec
Average delay         = 7288637.283738 msec
Average jitter        = 0.145226 msec
Delay variation       = 4.748347 msec
Byte received         = 7481719
Total time           = 9.999808 sec
Average bitrate       = 5985.490121 Kbit/sec
Average packets rate  = 995.819120 pkts/sec
Packets dropped       = 42
Packets wrong         = 0
```

Total packets received = 9958

1. decode the sender log file on the source host:

```
[donato@otto donato]$ ITGDec send_log_file
-----
Flow Id : 1
-----
From          10.0.0.4:32810  ---> To          10.0.0.3:9501

Total time      = 9.999000 sec
Total packet    = 10000
Max delay       = 0.000000 msec
Min delay       = 0.000000 msec
Average delay   = 0.000000 msec
Average jitter  = 0.000000 msec
Delay variation = 0.000000 msec
Byte received   = 7513374
Average bitrate = 6011.300330 Kbps
Average packet rate = 1000.100010 pkt/sec
packets dropped = 0
-----
Total results
-----
Total number of flows = 1
Max delay             = 0.000000 msec
Min delay             = 0.000000 msec
Average delay         = 0.000000 msec
Average jitter        = 0.000000 msec
Delay variation       = 0.000000 msec
Byte received         = 7513374
Total time           = 9.999000 sec
Average bitrate       = 6011.300330 Kbit/sec
Average packets rate  = 1000.100010 pkts/sec
Packets dropped       = 0
Packets wrong        = 0
Total packets received = 10000
-----
```

Example 3: Single TCP flow with constant inter-departure time between packets and uniformly distributed packet size between 500 and 1000 bytes with remote sender/receiver log

1. start the log server on the log host:

```
[donato@catarella tmp]$ ITGLog
```

2. start the receiver on the destination host:

```
[donato@catarella tmp] ITGRecv
```

3. start the sender on the source host:

```
[donato@otto donato]$ ITGSend -a 10.0.0.3 -p 9501
-C 1000 -u 500 1000 -l send_log_file -L 10.0.0.3
UDP -Z 10.0.0.3 UDP recv_log_file
```

4. close the receiver by pressing Ctrl-C

5. close the log server by pressing Ctrl-C

6. decode the receiver log file on the log host:

```
[donato@catarella tmp]ITGDec recv_log_file
-----
Flow Id : 1
-----
From          10.0.0.4:32811  ---> To          10.0.0.3:9501

Total time      = 9.999074 sec
Total packet    = 10000
Max delay       = 7288657.341000 msec
Min delay       = 7288655.495000 msec
Average delay   = 7288655.641950 msec
Average jitter  = 728.986754 msec
Delay variation = 0.152799 msec
Byte received   = 7499571
Average bitrate = 6000.212420 Kbps
```

```

Average packet rate = 1000.092609 pkt/sec
packets dropped      = 0
-----
Total results
-----
Total number of flows = 1
Max delay              = 7288657.341000 msec
Min delay              = 7288655.495000 msec
Average delay          = 7288655.641950 msec
Average jitter         = 0.048301 msec
Delay variation        = 0.152799 msec
Byte received          = 7499571
Total time             = 9.999074 sec
Average bitrate        = 6000.212420 Kbit/sec
Average packets rate   = 1000.092609 pkts/sec
Packets dropped        = 0
Packets wrong          = 0
Total packets received = 10000
-----

```

7. decode the sender log file on the log host:

```

[donato@otto donato]$ ITGDec send_log_file
-----
Flow Id : 1
-----
From          10.0.0.4:32811  ---> To          10.0.0.3:9501

Total time      = 9.999000 sec
Total packet    = 10000
Max delay       = 0.000000 msec
Min delay       = 0.000000 msec
Average delay   = 0.000000 msec
Average jitter  = 0.000000 msec
Delay variation = 0.000000 msec
Byte received   = 7499571
Average bitrate = 6000.256826 Kbps
Average packet rate = 1000.100010 pkt/sec
packets dropped = 0
-----
Total results
-----
Total number of flows = 1
Max delay              = 0.000000 msec
Min delay              = 0.000000 msec
Average delay          = 0.000000 msec
Average jitter         = 0.000000 msec
Delay variation        = 0.000000 msec
Byte received          = 7499571
Total time             = 9.999000 sec
Average bitrate        = 6000.256826 Kbit/sec
Average packets rate   = 1000.100010 pkts/sec
Packets dropped        = 0
Packets wrong          = 0
Total packets received = 10000
-----

```

7.2 Script file generation

If you want to simultaneously generate more than one flow, you have to prepare a script file like those shown in the following examples:

Example 4: Three UDP flows with different constant bit rate and remote log

1. start the log server on the log host:

```
[donato@otto tmp]$ ITGLog
```

2. start the receiver on the destination host:

```
[donato@catarella tmp] ITGRecv
```

3. start the sender:

```

[donato@otto donato]$ cat script_file
-a 10.0.0.3 -p 10001 -C 1000 -c 512 -T UDP
-a 10.0.0.3 -p 10002 -C 2000 -c 512 -T UDP
-a 10.0.0.3 -p 10003 -C 3000 -c 512 -T UDP

```

- ```
[donato@otto tmp]$ ITGSend script_file -l
send_log_file -L 10.0.0.4 UDP -Z 10.0.0.4 UDP
recv_log_file
```
4. close the receiver by pressing Ctrl-C:
  5. close the log server by pressing Ctrl-C:
  6. decode the receiver log file on the log host:

```
[donato@otto donato]$ ITGDec recv_log_file
```

```

Flow Id : 1

From 10.0.0.4:32822 ---> To 10.0.0.3:10001

Total time = 9.975128 sec
Total packet = 9976
Max delay = 7288691.402000 msec
Min delay = 7288682.630000 msec
Average delay = 7288683.525721 msec
Average jitter = 730.813785 msec
Delay variation = 1.161023 msec
Byte received = 5107712
Average bitrate = 4096.358062 Kbps
Average packet rate = 1000.087417 pkt/sec
packets dropped = 0

Flow Id : 2

From 10.0.0.4:32823 ---> To 10.0.0.3:10002

Total time = 9.999648 sec
Total packet = 19665
Max delay = 7288716.313000 msec
Min delay = 7288682.630000 msec
Average delay = 7288683.763405 msec
Average jitter = 0.126182 msec
Delay variation = 1.539421 msec
Byte received = 10068480
Average bitrate = 8055.067538 Kbps
Average packet rate = 1966.569223 pkt/sec
packets dropped = 335

Flow Id : 3

From 10.0.0.4:32824 ---> To 10.0.0.3:10003

Total time = 9.999814 sec
Total packet = 29266
Max delay = 7288713.811000 msec
Min delay = 7288682.628000 msec
Average delay = 7288684.060495 msec
Average jitter = 0.101156 msec
Delay variation = 1.519769 msec
Byte received = 14984192
Average bitrate = 11987.576569 Kbps
Average packet rate = 2926.654436 pkt/sec
packets dropped = 734

Total results

Total number of flows = 3
Max delay = 7288716.313000 msec
Min delay = 7288682.628000 msec
Average delay = 7288683.870752 msec
Average jitter = 0.112491 msec
Delay variation = 1.486200 msec
Byte received = 30160384
Total time = 10.098591 sec
Average bitrate = 23892.746226 Kbit/sec
Average packets rate = 5833.189997 pkts/sec
Packets dropped = 1069
Packets wrong = 0
Total packets received = 58907

```

7. decode the sender log file on the log host:

```
[donato@otto donato]$ ITGDec send_log_file
```

```

Flow Id : 1

From 10.0.0.4:32822 ---> To 10.0.0.3:10001

Total time = 9.975000 sec
Total packet = 9976
Max delay = 0.000000 msec
Min delay = 0.000000 msec
Average delay = 0.000000 msec
Average jitter = 0.000000 msec
Delay variation = 0.000000 msec
Byte received = 5107712
Average bitrate = 4096.410627 Kbps
Average packet rate = 1000.100251 pkt/sec
packets dropped = 0

Flow Id : 2

From 10.0.0.4:32823 ---> To 10.0.0.3:10002

Total time = 9.999501 sec
Total packet = 20000
Max delay = 0.000000 msec
Min delay = 0.000000 msec
Average delay = 0.000000 msec
Average jitter = 0.000000 msec
Delay variation = 0.000000 msec
Byte received = 10240000
Average bitrate = 8192.408801 Kbps
Average packet rate = 2000.099805 pkt/sec
packets dropped = 0

Flow Id : 3

From 10.0.0.4:32824 ---> To 10.0.0.3:10003

Total time = 9.999668 sec
Total packet = 30000
Max delay = 0.000000 msec
Min delay = 0.000000 msec
Average delay = 0.000000 msec
Average jitter = 0.000000 msec
Delay variation = 0.000000 msec
Byte received = 15360000
Average bitrate = 12288.407975 Kbps
Average packet rate = 3000.099603 pkt/sec
packets dropped = 0

Total results

Total number of flows = 3
Max delay = 0.000000 msec
Min delay = 0.000000 msec
Average delay = 0.000000 msec
Average jitter = 0.000000 msec
Delay variation = 0.000000 msec
Byte received = 30707712
Total time = 10.098462 sec
Average bitrate = 24326.644592 Kbit/sec
Average packets rate = 5939.122215 pkts/sec
Packets dropped = 0
Packets wrong = 0
Total packets received = 59976

```

### Example 5: VoIP, Telnet and DNS flows towards two distinct destinations

1. start the reciver on the first destination host:  

```
[donato@catarella donato]$ ITGRecv -l
recv1_log_file
```
2. start the receiver on the second destination host:  

```
[donato@otto donato]$ ITGRecv -l recv2_log_file
```
3. start the sender on the source host:



```
[donato@otto donato]$ cat script_file
-a 10.0.0.3 -p 10001 VoIP -x G.711.2 -h RTP -i VAD
-a 10.0.0.4 -p 10002 Telnet
-a 10.0.0.4 -p 10003 DNS
[donato@otto donato]$ ITGSend script_file -l
sender_log_file
```

4. close the first receiver by pressing Ctrl-C
5. close the second receiver by pressing Ctrl-C
6. decode the sender log file:

```
[donato@otto donato]$ ITGDec sender_log_file

Flow Id : 2

From 10.0.0.4:32822 ---> To 10.0.0.4:10002

Total time = 9.994707 sec
Total packet = 220
Max delay = 0.000000 msec
Min delay = 0.000000 msec
Average delay = 0.000000 msec
Average jitter = 0.000000 msec
Delay variation = 0.000000 msec
Byte received = 449
Average bitrate = 0.359390 Kbps
Average packet rate = 22.011651 pkt/sec
packets dropped = 0

Flow Id : 1

From 10.0.0.4:32825 ---> To 10.0.0.3:10001

Total time = 9.985236 sec
Total packet = 500
Max delay = 0.000000 msec
Min delay = 0.000000 msec
Average delay = 0.000000 msec
Average jitter = 0.000000 msec
Delay variation = 0.000000 msec
Byte received = 56000
Average bitrate = 44.866241 Kbps
Average packet rate = 50.073929 pkt/sec
packets dropped = 0

Flow Id : 3

From 10.0.0.4:32826 ---> To 10.0.0.4:10003

Total time = 8.934449 sec
Total packet = 6
Max delay = 0.000000 msec
Min delay = 0.000000 msec
Average delay = 0.000000 msec
Average jitter = 0.000000 msec
Delay variation = 0.000000 msec
Byte received = 1144
Average bitrate = 1.024350 Kbps
Average packet rate = 0.671558 pkt/sec
packets dropped = 0

Total results

Total number of flows = 3
Max delay = 0.000000 msec
Min delay = 0.000000 msec
Average delay = 0.000000 msec
Average jitter = 0.000000 msec
Delay variation = 0.000000 msec
Byte received = 57593
Total time = 8.934707 sec
Average bitrate = 51.567891 Kbit/sec
Average packets rate = 81.256162 pkts/sec
Packets dropped = 0
Packets wrong = 0
Total packets received = 726
```

7. decode the first receiver log file:

```
[donato@catarella src]# ITGDec recv1_log_file
```

```

Flow Id : 1

From 10.0.0.4:32825 ---> To 10.0.0.3:10001

Total time = 9.985378 sec
Total packet = 500
Max delay = 7288741.060000 msec
Min delay = 7288709.880000 msec
Average delay = 7288710.158406 msec
Average jitter = 14606.995768 msec
Delay variation = 1.745491 msec
Byte received = 56000
Average bitrate = 44.865602 Kbps
Average packet rate = 50.073217 pkt/sec
packets dropped = 0

Total results

Total number of flows = 1
Max delay = 7288741.060000 msec
Min delay = 7288709.880000 msec
Average delay = 7288710.158406 msec
Average jitter = 0.362713 msec
Delay variation = 1.745491 msec
Byte received = 56000
Total time = 9.985378 sec
Average bitrate = 44.865602 Kbit/sec
Average packets rate = 50.073217 pkts/sec
Packets dropped = 0
Packets wrong = 0
Total packets received = 500

```

8. decode the second receiver log file:

```
[donato@otto donato]$ ITGDec recv2_log_file
```

```

Flow Id : 2

From 10.0.0.4:32822 ---> To 10.0.0.4:10002

Total time = 9.994668 sec
Total packet = 220
Max delay = 1.041000 msec
Min delay = 0.013000 msec
Average delay = 0.198005 msec
Average jitter = 0.130447 msec
Delay variation = 0.304595 msec
Byte received = 449
Average bitrate = 0.359392 Kbps
Average packet rate = 22.011737 pkt/sec
packets dropped = 0

Flow Id : 3

From 10.0.0.4:32826 ---> To 10.0.0.4:10003

Total time = 8.934390 sec
Total packet = 6
Max delay = 0.086000 msec
Min delay = 0.025000 msec
Average delay = 0.040000 msec
Average jitter = 0.034200 msec
Delay variation = 0.022196 msec
Byte received = 1144
Average bitrate = 1.024356 Kbps
Average packet rate = 0.671562 pkt/sec
packets dropped = 0

Total results

Total number of flows = 2
Max delay = 1.041000 msec
```

```
Min delay = 0.013000 msec
Average delay = 0.193810 msec
Average jitter = 0.127449 msec
Delay variation = 0.301617 msec
Byte received = 1593
Total time = 8.934671 sec
Average bitrate = 1.426354 Kbit/sec
Average packets rate = 25.294720 pkts/sec
Packets dropped = 0
Packets wrong = 0
Total packets received = 226

```

---

**For additional information send an e-mail to the Authors:**

Stefano Avallone ([stavallo@unina.it](mailto:stavallo@unina.it))

Donato Emma ([demma@napoli.consorzio-cini.it](mailto:demma@napoli.consorzio-cini.it))

Antonio Pescapè ([pescape@unina.it](mailto:pescape@unina.it))