TTCP Feature

Teldat Dm831-I

Legal Notice

Warranty

This publication is subject to change.

Teldat offers no warranty whatsoever for information contained in this manual.

Teldat is not liable for any direct, indirect, collateral, consequential or any other damage connected to the delivery, supply or use of this manual.

Table of Contents

Chapter 1 Introduction

Test TCP (TTCP) is a feature that times the transmission and reception of data between two systems using the TCP protocol for measuring network throughput. It differs from common "blast" tests, which tend to measure the remote inetd as much as the network performance.

When testing, the transmitter should be started with *tx* after the receiver has been started with *rx*. In order to obtain accurate measurements, tests must last, at least, tens of seconds. Graphical presentations of throughput versus buffer size for buffers ranging from tens of bytes to several "pages" can help define bottlenecks.

Chapter 2 Options

To use it, enter **feature ttcp** from the general monitoring menu.

Syntax:

```
+feature ttcp
```

Example:

```
+feature ttcp


-- TTCP --
Ttcp+
```

Once inside, you have to choose the role that the device is going to play in the test: **sender (client)** or **receiver (server)**. If you choose the "sender" role, you must indicate the IP address of the receiving device.

If no parameter is entered (**ttcp <cr>** option), they are requested.

Syntax:

```
Ttcp+ttcp ?
  tx      client
  rx      server
  <cr>    ask for epittcp options
Ttcp+ttcp
```

```
Ttcp+ttcp tx ?
  <a.b.c.d>    destination host: IP address
Ttcp+ttcp tx
```

Depending on the role chosen, the options that can be specified are:

- **nsrcb** (client option): number or source buffers transmitted. The default value is 2048.
- **blen**: length of buffers in bytes read from, or written to, the network. The default value is 8192.
- **port**: port number to send or listen at. Default is port 5001. On some systems, this port may be allocated to another network daemon.
- **timeout** (server option): sets the timeout value (in milliseconds) to exit if no data has been received or the TCP connection is not working. The default value is 0, which means that no timeout is set.
- **verbose**: print more statistics.
- **phmarks**: print hash marks to indicate how the transfer process is progressing, one per buffer.
- **nbhmarks**: number of buffers to send between hash marks.
- **sbsize**: set socket buffer size. The default value is 65535 bytes. This value times two determines the size of the TCP window receive size of the server (up to a maximum of 262144 bytes).
- **nsrcbb**: number of source buffers per burst. The default value is 50.
- **random** (client option): use random data in TCP frames.
- **iRNG** (client option): init/seed value for RNG when sending random size buffers. The default value is 1.
- **frate**: specify, using one of the following characters, the format of the throughput rates. v,B = bits,bytes; k,K = kilo{bits,bytes}; m,M = mega{bits,bytes}; g,G = giga{bits,bytes}.
- **rlimit** (client option): client transmit rate limit in Kbps. No limit by default.
- **perc** (client option): test completed progress shown as a percentage.
- **mss**: configures the maximum segment size for the TCP protocol. The default value is 1340 bytes.
- **time** (client option): test duration in seconds. The client transmits until the duration has expired.

Note

The above commands will run in both modes unless specified (i.e., unless it is stated, between brackets, that they are a "client" or "server" option)

Note

Be careful when configuring the number of buffers to send between hash marks (**nbhmarks**). A low value can severely affect the throughput measured. Instead, we recommend using the percentage (**perc**) option to track the transfer progress.

Command history:

Release	Modification
11.01.07	Option *-t* has been renamed and is now option *tx* as of version 11.01.07.
11.01.07	Option *-r* has been renamed and is now option *rx* as of version 11.01.07.
11.01.07	Option *-l* has been renamed and is now option *blen* as of version 11.01.07.
11.01.07	Option *-p* has been renamed and is now option *port* as of version 11.01.07.
11.01.07	Option *-n* has been renamed and is now option *nsrcb* as of version 11.01.07.
11.01.07	Option *-x* has been renamed and is now option *random* as of version 11.01.07.
11.01.07	Option *-H* has been renamed and is now option *phmarks* as of version 11.01.07.
11.01.07	Option *-#* has been renamed and is now option *nbhmarks* as of version 11.01.07.
11.01.07	Option *-I* has been renamed and is now option *iRNG* as of version 11.01.07.
11.01.07	Option *-Ns* has been renamed and is now option *nsrcbb* as of version 11.01.07.
11.01.07	Option *-v* has been renamed and is now option *verbose* as of version 11.01.07.
11.01.07	Option *-b* has been renamed and is now option *sbsize* as of version 11.01.07.
11.01.07	Option *-f* has been renamed and is now option *frate* as of version 11.01.07.
11.01.07	Option *-w* has been renamed and is now option *timeout* as of version 11.01.07.
11.01.07	Option *-c* has been renamed and is now option *rlimit* as of version 11.01.07.
11.01.07	The *perc* option was introduced as of version 11.01.07.
11.01.07	The *mss* option was introduced as of version 11.01.07.
11.01.07	The *time* option was introduced as of version 11.01.07.
11.01.07	The *-Vr* option is obsolete as of version 11.01.07.
11.01.07	The *-A* option is obsolete as of version 11.01.07.
11.01.07	The *-O* option is obsolete as of version 11.01.07.
11.01.07	The *-R* option is obsolete as of version 11.01.07.
11.01.07	The *-De* option is obsolete as of version 11.01.07.
11.01.07	The *-d* option is obsolete as of version 11.01.07.
11.01.07	The *-Bf* option is obsolete as of version 11.01.07.
11.01.07	The *-T* option is obsolete as of version 11.01.07.
11.01.07	The *-u* option is obsolete as of version 11.01.07.
11.01.07	The *-S* option is obsolete as of version 11.01.07.

Chapter 3 Usage

This feature is designed to be tested with multiple operating systems and ttcp tools. It is no synchronous by default, meaning that no control threads are open for each new connection. This allows the application to send or receive data from the remote host as soon as possible.

For it to be compatible with multiple operating systems and ttcp tools, the latter must support the no synchronous operation mode. The **NTttcp** software, available for Linux and Windows, supports the no-sync operation mode (**-N** flag for Linux and **-ns** for Windows) and has been tested against the ttcp tool developed. This software can be downloaded from the following links:

https://github.com/Microsoft/ntttcp-for-linux (Linux)

https://gallery.technet.microsoft.com/NTttcp-Version-528-Now-f8b12769 (Windows)

To properly measure the bandwidth of the desired network, some parameters from this tool have to be adjusted. The default values are set to measure the bandwidth in a 100Mbps network, but this test may take too long or too short (depending on the network speed) and give inaccurate results. The basic parameters to adjust are:

- **Buffer length**: Length in bytes of the buffer to be read or written.
- **Number of buffers** (sender only): This parameter determines the total number of bytes to be sent, which is the result of the number of buffers times the buffer length.
- **Maximum Segment Size or MSS** (TCP only): Maximum amount of data that can be encapsulated in each TCP packet. The larger it is, the better (results-wise).
- **Socket receive/send buffer**: It configures how many bytes the socket can write or read each time. A very short value will imply in a large number of write and read operations, slowing down the test.

Warning

This ttcp tool is not guaranteed to be compatible with other OS ttcp tools that do not support the no synchronous operation mode.

Note

The output given by the application must be interpreted as an estimate of the measured network bandwidth at a given moment in time. This information might not always represent the reality because multiple factors can affect the output of this test. For example, network congestion or a large number of applications running on the device can lower the throughput measured. Device licenses and hardware components also play a big role when it comes to this test.

Note

Bear in mind that test results will be better if the device is processing as little extra information as possible. For example, if traces are configured to be displayed from a subsystem like IP or TCP, they will slow down the test and affect the output results.

3.1 CIT

Two devices can easily measure the network bandwidth they are connected to using this tool. A basic example where the buffer length is set to 4096 bytes and the number of buffers to be sent is 2048 is shown down below. In this case, the network bandwidth is limited to a maximum of 100Mbps.

- Server role.

```
Ttcp+ttcp rx frate m

--- TTCP  ---
TTCP running (Press any key to abort)...

ttcp-rx: buflen=8192, nbuf=2048, align=16384/0, port=5001
sockbufsize=65535

# tcp receiver #
ttcp-rx: accept from <client ip address>
```

```
ttcp-rx: 16777216 bytes in 1.485143 real seconds = 90.374 Mbit/sec +++
ttcp-rx: 2048 I/O calls, 0.725 msec(real)/call
ttcp done.

--- TTCP finished ---

Ttcp+
```

- Client role.

```
Ttcp+ttcp tx <server ip address> frate m perc blen 8192 nsrcb 2048

--- TTCP  ---
TTCP running (Press any key to abort)...

ttcp-tx: buflen=8192, nbuf=2048, align=16384/0, port=5001
sockbufsize=65535

# tcp sender -> <server ip address> #
# Completed... 100% #

ttcp-tx: 16777216 bytes in 1.473657 real seconds = 91.078 Mbit/sec +++
ttcp-tx: 2048 I/O calls, 0.720 msec(real)/call
ttcp done.

--- TTCP finished ---

Ttcp+
```

3.2 Linux

The NTttcp program is available for common Linux distributions like Ubuntu or Debian. This tool can operate in no-sync mode, thus allowing the test to run properly. Two possible tests are described: one where the Linux device acts as client and another where it acts as server.

3.2.1 Linux client

In this scenario, the network bandwidth is limited to a maximum of 100Mbps. The test lasts 10 seconds and the receiver side sets port 5002 to exchange data.

- Server role.

```
Ttcp+ttcp rx frate m port 5002

--- TTCP  ---
TTCP running (Press any key to abort)...

ttcp-rx: buflen=8192, nbuf=2048, align=16384/0, port=5002
sockbufsize=65535

# tcp receiver #
ttcp-rx: accept from <client ip address>


ttcp-rx: 117702656 bytes in 10.022581 real seconds = 93.950 Mbit/sec +++
ttcp-rx: 14368 I/O calls, 0.698 msec(real)/call
ttcp done.

--- TTCP finished ---

Ttcp+
```

- Client role (Linux machine).

```
$ ntttcp -s <server ip address> -N -p 5002 -n 1 -R -t 10 -V -P 1
NTTTCP for Linux 1.3.5
```

```
------------------------------------------------------------
*** sender role
*** no sender/receiver synch
connections: 1 X 1 X 1
cpu affinity: *
server address: <server ip address>
domain: IPv4
protocol: TCP
server port starting at: 5002
sender socket buffer (bytes): 131072
test warm-up (sec): no
test duration (sec): 10
test cool-down (sec): no
show system tcp retransmit: yes
verbose mode: enabled
------------------------------------------------------------
09:13:37 DBG : user limits for maximum number of open files: soft: 1024; hard: 4096
09:13:37 INFO: Starting sender activity (no sync) ...
09:13:37 INFO: 1 threads created
09:13:37 DBG : New connection: local:41819 [socket:4] --> <server ip address>:5002
09:13:47 INFO: Test run completed.
09:13:47 INFO: Test cycle finished.
09:13:47 INFO: ThreadTime(s)Throughput
09:13:47 INFO: ======================
09:13:47 INFO: 0 10.00 94.05Mbps
09:13:47 INFO: 1 connections tested
09:13:47 INFO: #####  Totals:  #####
09:13:47 INFO: test duration:10.00 seconds
09:13:47 INFO: total bytes:117571584
09:13:47 INFO:  throughput:94.05Mbps
09:13:47 INFO: tcp retransmit:
09:13:47 INFO:  retrans_segments/sec:0.50
09:13:47 INFO:  lost_retrans/sec:0.00
09:13:47 INFO:  syn_retrans/sec:0.20
09:13:47 INFO:  fast_retrans/sec:0.00
09:13:47 INFO:  forward_retrans/sec:0.00
09:13:47 INFO:  slowStart_retrans/sec:0.00
09:13:47 INFO:  retrans_fail/sec:0.00
09:13:47 INFO: cpu cores:8
09:13:47 INFO:  cpu speed:3549.859MHz
09:13:47 INFO:  user:0.42%
09:13:47 INFO:  system:0.23%
09:13:47 INFO:  idle:99.32%
09:13:47 INFO:  iowait:0.00%
09:13:47 INFO:  softirq:0.04%
09:13:47 INFO:  cycles/byte:16.45
09:13:47 INFO: cpu busy (all):1.02%
09:13:47 INFO: tcpi rtt:11040 us
------------------------------------------------------------
```

The parameters configured for the client test are the following:

- **-s**: Sender mode.
- **-N**: No synchronous operation mode.
- **-p**: Destination port.
- **-n**: Number of threads per each receiver port. We set this parameter to one because the developed ttcp tool only opens one listening thread per connection.
- **-R**: Give extra information about TCP protocol in the test.
- **-t**: Duration of the test. In this case it is set to last 10 seconds.
- **-V**: Enable verbose mode.
- **-P**: Number of ports listening on the server side. This parameter is set to one for the same reason as the number of threads.

3.2.2 Linux server

In this case, the network bandwidth is configured to have a maximum speed of 1Gbps. The server in this case is the linux device.

- Server role (Linux machine).

```
$ ntttcp -r -m 1,*,<server ip address> -N -V
NTTTCP for Linux 1.3.5
---------------------------------------------------------
*** receiver role
*** no sender/receiver synch
ports: 1
cpu affinity: *
server address: <server ip address>
domain: IPv4
protocol: TCP
server port starting at: 5001
receiver socket buffer (bytes): 65536
test warm-up (sec): no
test duration (sec): 60
test cool-down (sec): no
show system tcp retransmit: no
verbose mode: enabled
---------------------------------------------------------
11:03:02 DBG : user limits for maximum number of open files: soft: 1024; hard: 4096
11:03:02 INFO: 1 threads created
11:03:02 DBG : ntttcp server is listening on 192.168.212.35:5001
11:03:03 DBG : New connection: <client ip address>:1039 --> local:5001 [socket 4]
11:03:03 DBG : socket closed: 4
^C11:03:08 INFO: Interrupted by Ctrl+C
11:03:08 INFO: Test run completed.
11:03:08 INFO: Test cycle finished.
11:03:08 INFO: ThreadTime(s)Throughput
11:03:08 INFO: ========================
11:03:08 INFO: 0 4.80 27.97Mbps
11:03:08 INFO: #####  Totals:  #####
11:03:08 INFO: test duration:4.80 seconds
11:03:08 INFO: total bytes:16777216
11:03:08 INFO:  throughput:27.97Mbps
11:03:08 INFO: cpu cores:8
11:03:08 INFO:  cpu speed:800.012MHz
11:03:08 INFO:  user:2.25%
11:03:08 INFO:  system:1.36%
11:03:08 INFO:  idle:96.30%
11:03:08 INFO:  iowait:0.03%
11:03:08 INFO:  softirq:0.05%
11:03:08 INFO:  cycles/byte:67.66
11:03:08 INFO: cpu busy (all):3.42%
---------------------------------------------------------
```

The server is configured as a receiver with the "-r" parameter and a 3-tuple is given indicating that only one thread that listens to port 5001 at the server IP address is created. The no-sync and verbose modes are enabled. The server output does not match the client output because the user must manually end the test at the server side and the test duration does not match.

- Client role.

```
Ttcp+ttcp tx <server ip address> frate m perc

--- TTCP  ---
TTCP running (Press any key to abort)...

ttcp-tx: buflen=8192, nbuf=2048, align=16384/0, port=5001
sockbufsize=65535

# tcp sender -> <server ip address> #
# Completed... 100% #
```

```
ttcp-tx: 16777216 bytes in 0.152508 real seconds = 880.070 Mbit/sec +++
ttcp-tx: 2048 I/O calls, 0.074 msec(real)/call
ttcp done.

--- TTCP finished ---

Ttcp+
```

3.3 Windows

The NTttcp program is available for the latest Windows versions, starting with Windows 7. There are not as many parameters as in the Linux version but the overall network bandwidth can still be measured. Below there are two usage examples: one where the Windows machine acts a client and a second one where it acts as a server.

3.3.1 Windows client

- Server role.

```
Ttcp+ttcp rx frate m

--- TTCP  ---
TTCP running (Press any key to abort)...

ttcp-rx: buflen=8192, nbuf=2048, align=16384/0, port=5001
sockbufsize=65535

# tcp receiver #
ttcp-rx: accept from <client ip address>


ttcp-rx: 236847104 bytes in 20.014945 real seconds = 94.668 Mbit/sec +++
ttcp-rx: 28912 I/O calls, 0.692 msec(real)/call
ttcp done.

--- TTCP finished ---

Ttcp+
```

- Client role (Windows machine).

```
C:\Users\pc>C:\Users\pc\Desktop\NTttcp.exe -s -m 1,*,<server ip address> -t 10 -ns -p 5001
Copyright Version 5.33
Network activity progressing...


Thread  Time(s) Throughput(KB/s) Avg B / Compl
======  ======= ================ =============
     0    9.999        11559.556     65536.000


#####  Totals:  #####


   Bytes(MEG)    realtime(s) Avg Frame Size Throughput(MB/s)
================ =========== ============== ================
      112.875000      10.000       1455.729           11.288


Throughput(Buffers/s) Cycles/Byte       Buffers
===================== =========== =============
              180.600      24.104      1806.000


DPCs(count/s) Pkts(num/DPC)   Intr(count/s) Pkts(num/intr)
============= ============= ============== ==============
```

```
     3048.900         2.608        4870.400          1.633


Packets Sent Packets Received Retransmits Errors Avg. CPU %
============ ================ =========== ====== ==========
       81305            79529           0      0      2.145

C:\Users\pc>
```

The NTttcp client is set to operate in no-synchronous (**-ns** flag) mode for 10 seconds. Only one thread is open to connect to the server that is listening to port 5002.

3.3.2 Windows server

- Server role (Windows machine).

```
C:\Users\pc>C:\Users\pc\Desktop\NTttcp.exe -r -p 5002 -m 1,*,<server ip address> -ns

Copyright Version 5.33
Network activity progressing...
ERROR: DoSendsReceives failed: Unexpected disconnect
ERROR: DoSendsReceives failed: error in send/recv
ERROR: StartSenderReceiver in thread: 0 failed: error in send/recv
ERROR: DoWork failed: one or more worker threads failed
ERROR: main failed: one or more worker threads failed


Thread  Time(s) Throughput(KB/s) Avg B / Compl
======  ======= ================ =============
     0    1.416        11570.621      1975.184


#####  Totals:  #####


   Bytes(MEG)    realtime(s) Avg Frame Size Throughput(MB/s)
================ =========== ============== ================
       16.000000       1.417       1456.735           11.291


Throughput(Buffers/s) Cycles/Byte       Buffers
===================== =========== =============
              180.663      74.201       256.000


DPCs(count/s) Pkts(num/DPC)   Intr(count/s) Pkts(num/intr)
============= ============= ============== ==============
     3078.335         2.640        4908.257          1.656


Packets Sent Packets Received Retransmits Errors Avg. CPU %
============ ================ =========== ====== ==========
        4250            11517           0      0      6.606

C:\Users\pc>
```

The server runs one thread that listens on port 5002 and operates in no-sync mode.

- Client role.

```
Ttcp+ttcp tx <server ip address> frate m perc port 5002

--- TTCP  ---
TTCP running (Press any key to abort)...

ttcp-tx: buflen=8192, nbuf=2048, align=16384/0, port=5002
sockbufsize=65535
```

```
# tcp sender -> <server ip address> #
# Completed... 100% #

ttcp-tx: 16777216 bytes in 1.406833 real seconds = 95.404 Mbit/sec +++
ttcp-tx: 2048 I/O calls, 0.687 msec(real)/call
ttcp done.

--- TTCP finished ---

Ttcp+
```

Chapter 4 Performance

The aim of this section is to explain the scope of this tool. Several devices have been tested using the ttcp tool to establish the maximum throughput that can be measured in the best-case scenario. The device that runs the test against these devices is a Linux machine with Ubuntu 18.04 and an Ethernet port of 1Gbps.

The network of this scenario has an estimated bandwidth of 1Gbps, to match the Linux device ethernet port speed. The MSS is set to be the highest possible and TCP receive window size scaling is active. The parameters used in this test (which devices send and receive) are the default ones. The Linux commands used match those of the previous section.

The results shows that, even though conditions are the same, each device tested measures different bandwidth values. Therefore, we can conclude that bandwidth measurements can be more or less accurate depending on the specs of the device tested.

Note

The throughput measured by this tool cannot be larger than a third of the value obtained from the IMIX traffic with services test. This information can be checked at the device performance table.

Note

Each device may output a different throughput value in the same network. Hardware, licenses and configuration options may have an impact on results.