



iPerf3 User Documentation

Support Team - 2021-10-21 - Comments (0) - General

iPerf 3 user documentation

GENERAL OPTIONS

Command line option	Description
-p, --port <i>n</i>	The server port for the server to listen on and the client to connect to. This should be the same in both client and server. Default is 5201.
--cport<i>n</i>	Option to specify the client-side port. (new in iPerf 3.1)
-f, --format[<i>kmKM</i>]	A letter specifying the format to print bandwidth numbers in. Supported formats are 'k' = Kbits/sec 'K' = KBytes/sec 'm' = Mbits/sec 'M' = MBytes/sec The adaptive formats choose between kilo- and mega- as appropriate.
-i, --interval <i>n</i>	Sets the interval time in seconds between periodic bandwidth, jitter, and loss reports. If non-zero, a report is made every <i>interval</i> seconds of the bandwidth since the last report. If zero, no periodic reports are printed. Default is zero.
-F, --file name	client-side: read from the file and write to the network, instead of using random data; server-side: read from the network and write to the file, instead of throwing the data away.
-A, --affinity<i>n/n,m-F</i>	Set the CPU affinity, if possible (Linux and FreeBSD only). On both the client and server you can set the local affinity by using the <i>n</i> form of this argument (where <i>n</i> is a CPU number). In addition, on the client side you can override the server's affinity for just that one test, using the <i>n,m</i> form of argument. Note that when using this feature, a process will only be bound to a single CPU (as opposed to a set containing potentially multiple CPUs).
-B, --bind <i>host</i>	Bind to <i>host</i> , one of this machine's addresses. For the client this sets the outbound interface. For a server this sets the incoming interface. This is only useful on multihomed hosts, which have multiple network interfaces.
-V, --verbose	give more detailed output

-J, --json	output in JSON format
--logfile file	send output to a log file. (new in iPerf 3.1)
--d, --debug	emit debugging output. Primarily (perhaps exclusively) of use to developers.
-v, --version	Show version information and quit.
-h, --help	Show a help synopsis and quit.

SERVER SPECIFIC OPTIONS

Command line option	Description
-s, --server	Run iPerf in server mode. (This will only allow one iperf connection at a time)
-D, --daemon	Run the server in background as a daemon.
-I, --pidfile <i>file</i>	write a file with the process ID, most useful when running as a daemon. (new in iPerf 3.1)

CLIENT SPECIFIC OPTIONS

Command line option	Description
-c, --client <i>host</i>	Run iPerf in client mode, connecting to an iPerf server running on <i>host</i> .
--sctp	Use SCTP rather than TCP (Linux, FreeBSD and Solaris). (new in iPerf 3.1)
-u, --udp	Use UDP rather than TCP. See also the -b option.
-b, --bandwidth <i>n</i>[<i>KM</i>]	Set target bandwidth to <i>n</i> bits/sec (default 1 Mbit/sec for UDP, unlimited for TCP). If there are multiple streams (-P flag), the bandwidth limit is applied separately to each stream. You can also add a '/' and a number to the bandwidth specifier. This is called "burst mode". It will send the given number of packets without pausing, even if that temporarily exceeds the specified bandwidth limit.
-t, --time <i>n</i>	The time in seconds to transmit for. iPerf normally works by repeatedly sending an array of <i>len</i> bytes for <i>time</i> seconds. Default is 10 seconds. See also the -l , -k and -n options.

-n, --num <i>n</i>[<i>KM</i>]	The number of buffers to transmit. Normally, iPerf sends for 10 seconds. The -n option overrides this and sends an array of <i>len</i> bytes <i>num</i> times, no matter how long that takes. See also the -l , -k and -t options.
-k, --blockcount<i>n</i>[<i>KM</i>]	The number of blocks (packets) to transmit. (instead of -t or -n) See also the -t , -l and -n options.
-l, --length<i>n</i>[<i>KM</i>]	The length of buffers to read or write. iPerf works by writing an array of <i>len</i> bytes a number of times. Default is 128 KB for TCP, 8 KB for UDP. See also the -n , -k and -t options.
-P, --parallel <i>n</i>	The number of simultaneous connections to make to the server. Default is 1.
-R, --reverse	Run in reverse mode (server sends, client receives).
-w, --window<i>n</i>[<i>KM</i>]	Sets the socket buffer sizes to the specified value. For TCP, this sets the TCP window size. (this gets sent to the server and used on that side too)
-M, --set-mss <i>n</i>	Attempt to set the TCP maximum segment size (MSS). The MSS is usually the MTU - 40 bytes for the TCP/IP header. For ethernet, the MSS is 1460 bytes (1500 byte MTU).
-N, --no-delay	Set the TCP no delay option, disabling Nagle's algorithm. Normally this is only disabled for interactive applications like telnet.
-4, --version4	only use IPv4.
-6, --version6	only use IPv6.
-S, --tos <i>n</i>	The type-of-service for outgoing packets. (Many routers ignore the TOS field.) You may specify the value in hex with a '0x' prefix, in octal with a '0' prefix, or in decimal. For example, '0x10' hex = '020' octal = '16' decimal. The TOS numbers specified in RFC 1349 are: IPTOS_LOWDELAY minimize delay 0x10 IPTOS_THROUGHPUT maximize throughput 0x08 IPTOS_RELIABILITY maximize reliability 0x04 IPTOS_LOWCOST minimize cost 0x02
-L, --flowlabel <i>n</i>	Set the IPv6 flow label (currently only supported on Linux).
-Z, --zerocopy	Use a "zero copy" method of sending data, such as sendfile(2), instead of the usual write(2). This uses much less CPU.
-O, --omit <i>n</i>	Omit the first <i>n</i> seconds of the test, to skip past the TCP TCP slowstart period.
-T, --title <i>str</i>	Prefix every output line with this string.

-C, --linux-congestion Set the [congestion control algorithm](#) (Linux only for iPerf
algo 3.0, Linux and FreeBSD for iPerf 3.1).

See also <https://github.com/esnet/iperf>