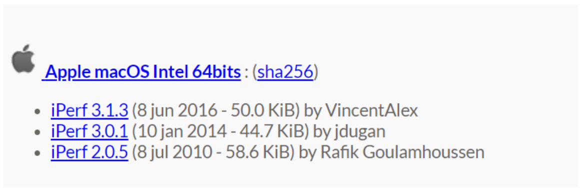
How to perform an IPERF test - MAC

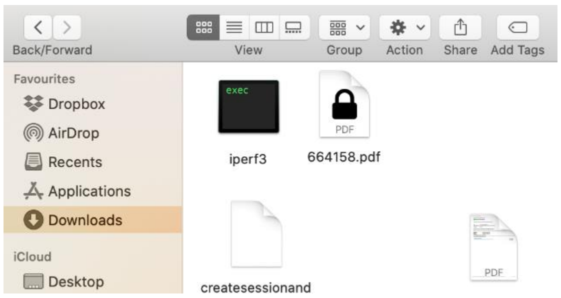
1. Download IPERF for MacOS: https://iperf.fr/iperf-download.php

You should see the file in your 'Downloads' folder.

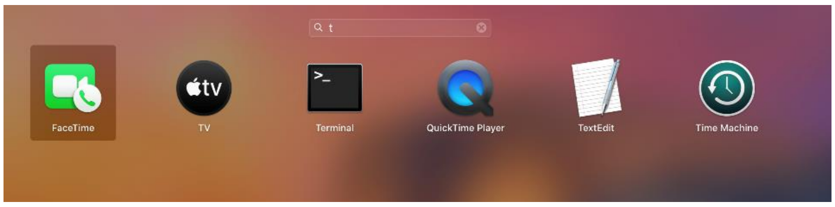


Apple macOS Intel 64bits : (sha256)

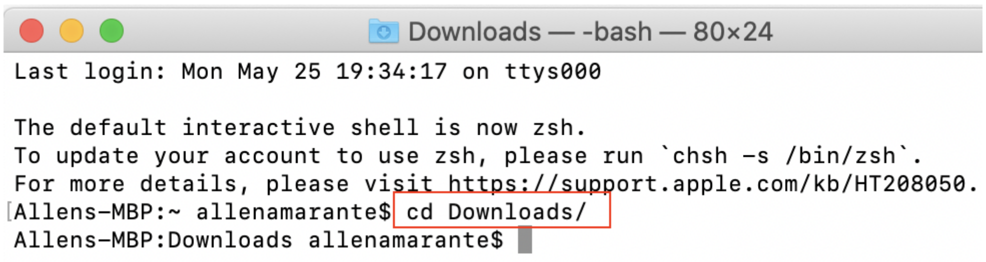
- iPerf 3.1.3 (8 jun 2016 - 50.0 KiB) by VincentAlex
- iPerf 3.0.1 (10 jan 2014 - 44.7 KiB) by jdugan
- iPerf 2.0.5 (8 jul 2010 - 58.6 KiB) by Rafik Goulamhoussen



2. Go to Launchpad and search for 'Terminal'. Click to Open application.



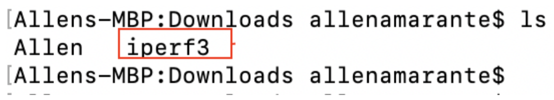
3. From the terminal app, type the command "cd Downloads/". You are now in the Downloads folder.



```
Downloads — -bash — 80×24
Last login: Mon May 25 19:34:17 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
[Allens-MBP:~ allenamarante$ cd Downloads/
Allens-MBP:Downloads allenamarante$
```

4. From the Downloads folder, type the command "ls" to open file in the Download folder. You should be able to see the "Iperf3" program similar below.



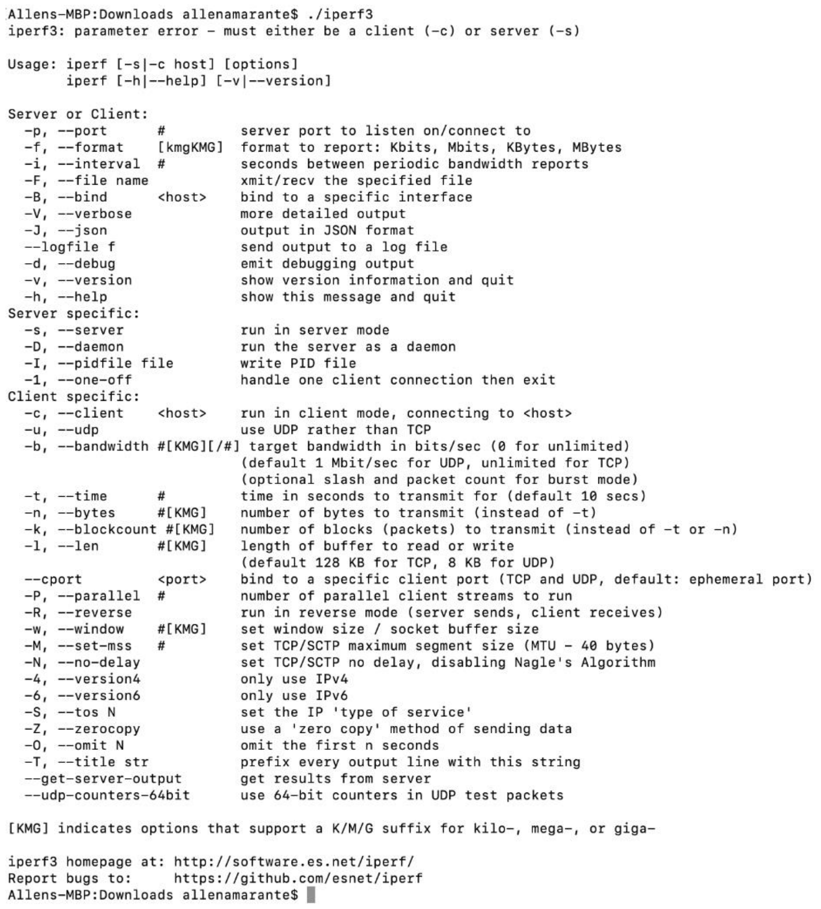
```
[Allens-MBP:Downloads allenamarante$ ls
 Allen    iperf3
[Allens-MBP:Downloads allenamarante$
```

5. Now type the command "./iperf3"and hit 'Enter'. You should now be able to run the program.

If this doesn't work and your Mac reports an error message saying "…Cannot be opened because developer cannot be verified..."

You will need to change the setting on your Mac:
Go to System Preference > Security & Privacy > General
*Enable any blocked app from Allow apps download pane at the bottom of the window.

Once done, you should now be able to run the command ./iperf3



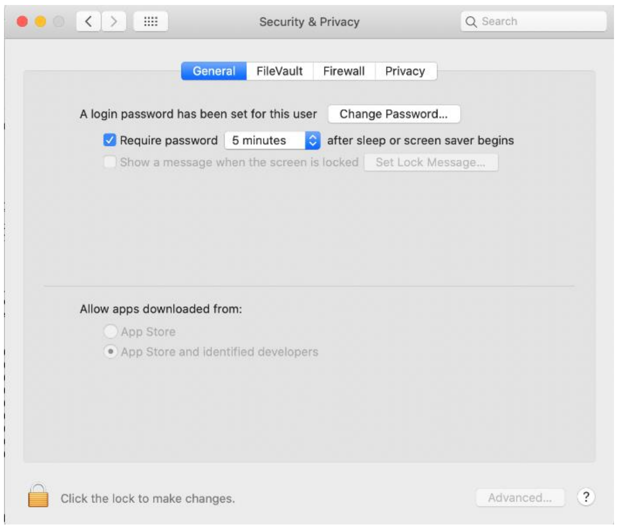
```
Allens-MBP:Downloads allenamarante$ ./iperf3
iperf3: parameter error - must either be a client (-c) or server (-s)

Usage: iperf [-s|-c host] [options]
       iperf [-h|--help] [-v|--version]

Server or Client:
  -p, --port      #         server port to listen on/connect to
  -f, --format    [kmgKMG]  format to report: Kbits, Mbits, KBytes, MBytes
  -i, --interval  #         seconds between periodic bandwidth reports
  -F, --file name           xmit/recv the specified file
  -B, --bind      <host>    bind to a specific interface
  -V, --verbose             more detailed output
  -J, --json                output in JSON format
  --logfile f               send output to a log file
  -d, --debug               emit debugging output
  -v, --version             show version information and quit
  -h, --help                show this message and quit
Server specific:
  -s, --server              run in server mode
  -D, --daemon              run the server as a daemon
  -I, --pidfile file        write PID file
  -1, --one-off             handle one client connection then exit
Client specific:
  -c, --client    <host>    run in client mode, connecting to <host>
  -u, --udp                 use UDP rather than TCP
  -b, --bandwidth #[KMG][/#] target bandwidth in bits/sec (0 for unlimited)
                            (default 1 Mbit/sec for UDP, unlimited for TCP)
                            (optional slash and packet count for burst mode)
  -t, --time      #         time in seconds to transmit for (default 10 secs)
  -n, --bytes     #[KMG]    number of bytes to transmit (instead of -t)
  -k, --blockcount #[KMG]   number of blocks (packets) to transmit (instead of -t or -n)
  -l, --len       #[KMG]    length of buffer to read or write
                            (default 128 KB for TCP, 8 KB for UDP)
  --cport         <port>    bind to a specific client port (TCP and UDP, default: ephemeral port)
  -P, --parallel  #         number of parallel client streams to run
  -R, --reverse             run in reverse mode (server sends, client receives)
  -w, --window    #[KMG]    set window size / socket buffer size
  -M, --set-mss   #         set TCP/SCTP maximum segment size (MTU - 40 bytes)
  -N, --no-delay            set TCP/SCTP no delay, disabling Nagle's Algorithm
  -4, --version4            only use IPv4
  -6, --version6            only use IPv6
  -S, --tos N               set the IP 'type of service'
  -Z, --zerocopy            use a 'zero copy' method of sending data
  -O, --omit N              omit the first n seconds
  -T, --title str           prefix every output line with this string
  --get-server-output       get results from server
  --udp-counters-64bit      use 64-bit counters in UDP test packets

[KMG] indicates options that support a K/M/G suffix for kilo-, mega-, or giga-

iperf3 homepage at: http://software.es.net/iperf/
Report bugs to:     https://github.com/esnet/iperf
Allens-MBP:Downloads allenamarante$
```



A login password has been set for this user [Change Password...]

[✓] Require password [5 minutes] after sleep or screen saver begins

[ ] Show a message when the screen is locked [Set Lock Message...]

Allow apps downloaded from:

( ) App Store

(●) App Store and identified developers

Click the lock to make changes.

```
./iperf3 -c 45.64.51.193 -u -R -b 500M
./iperf3 -c 45.64.51.193 -u -R -b 1000M
./iperf3 -c 45.64.51.193 -b 1000M -P 20 -R
./iperf3 -c 45.64.51.193 -u -R -b 500M
./iperf3 -c 45.64.51.193 -b 1000M -P 20
```

<u>Download Testing:</u>

500 Mbps *UDP* Download: Server sends at a rate of 500 Mbps UDP, client receives UDP:

```
iperf3.exe -c 45.64.51.193 -u -R -b 500M
```

1 Gbps *UDP* Download: Server sends at a rate of 1 Gbps UDP, client receives UDP:

```
iperf3.exe -c 45.64.51.193 -u -R -b 1000M
```

1 Gbps *TCP* Download: Server sends at a rate of 1 Gbps TCP, client receives TCP:

```
iperf3.exe -c 45.64.51.193 -b 1000M -P 20 -R
```

<u>Upload Testing:</u>

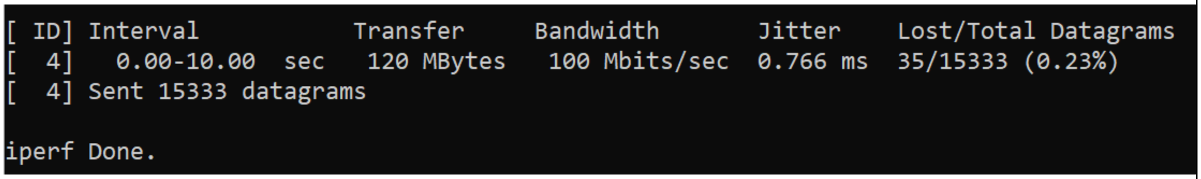
500 Mbps *UDP* Upload: Server sends at a rate of 500 Mbps UDP, client receives UDP:

```
iperf3.exe -c 45.64.51.193 -u -R -b 500M
```

500 Mbps *TCP*: Client sends TCP at undefined Mbps rate, server receives TCP - (extra load due to TCP, which guarantees delivery of data and also guarantees that packets will be delivered in the same order in which they were sent.):

```
iperf3.exe -c 45.64.51.193 -b 1000M -P 20
```

An example of an iperf result:



```
[ ID] Interval           Transfer     Bandwidth       Jitter    Lost/Total Datagrams
[  4]   0.00-10.00  sec   120 MBytes   100 Mbits/sec  0.766 ms  35/15333 (0.23%)
[  4] Sent 15333 datagrams

iperf Done.
```

In the above example you will see that in this case an 100Mb iperf test was able to transfer 120 MBytes at a **<u>speed of 100 Mbits per second</u>**.