

# ECE 456

## Lab 07

Pablo Corona Alvarez

IOzone:

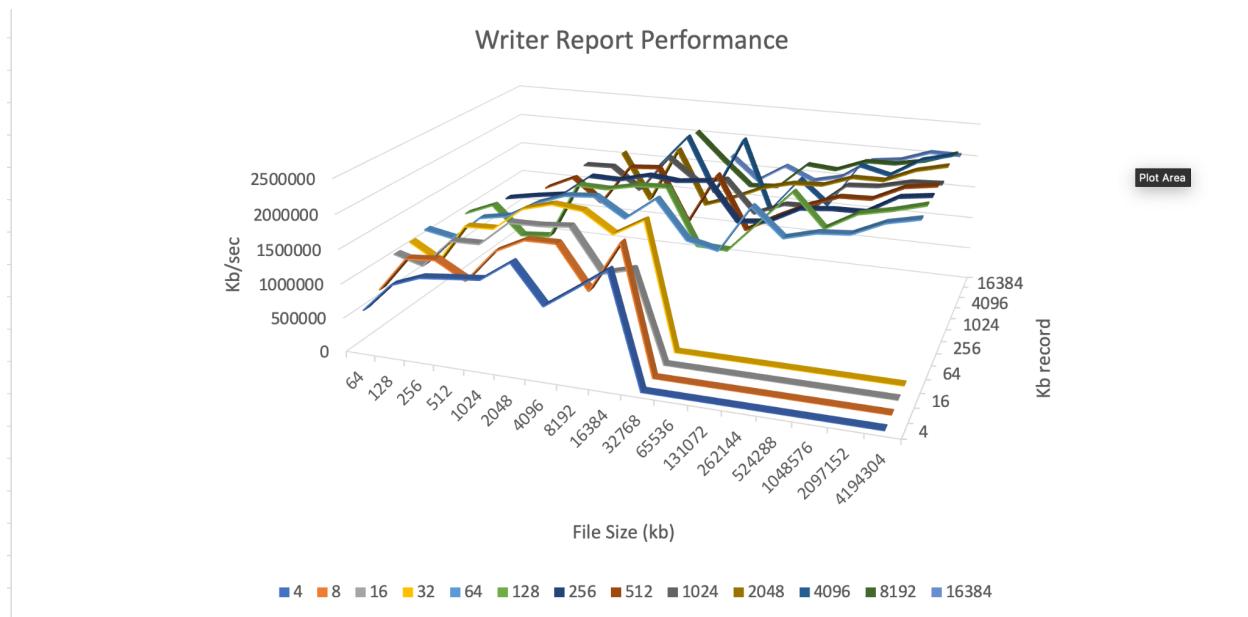


Figure 1: Writer Report Performance

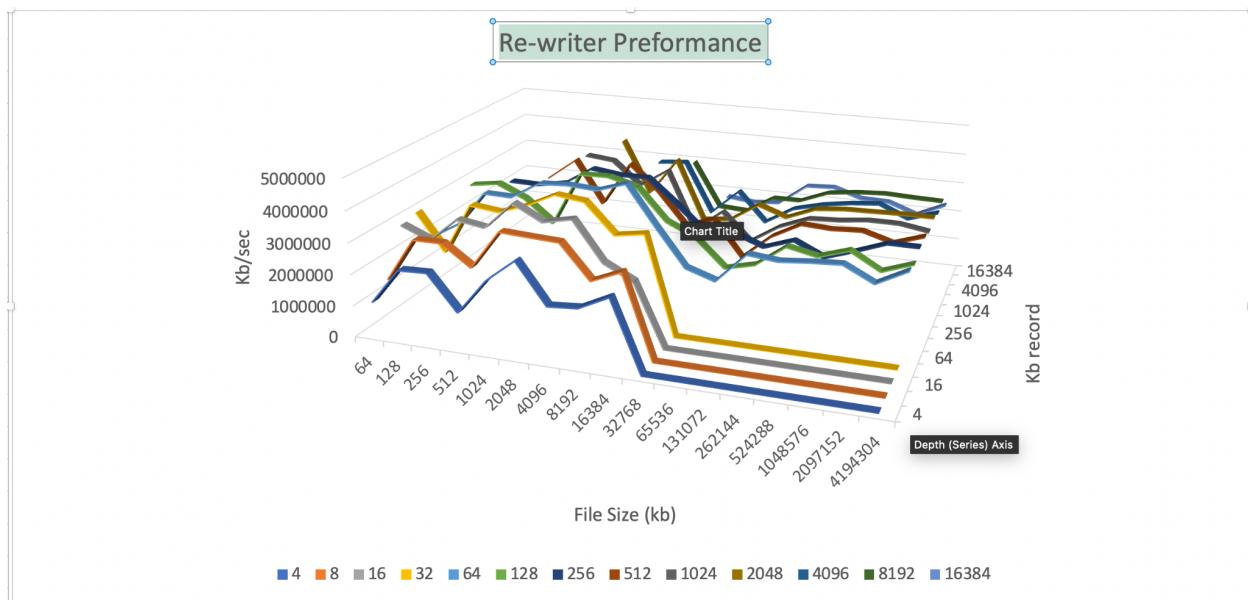


Figure 2: Re-writer Performance

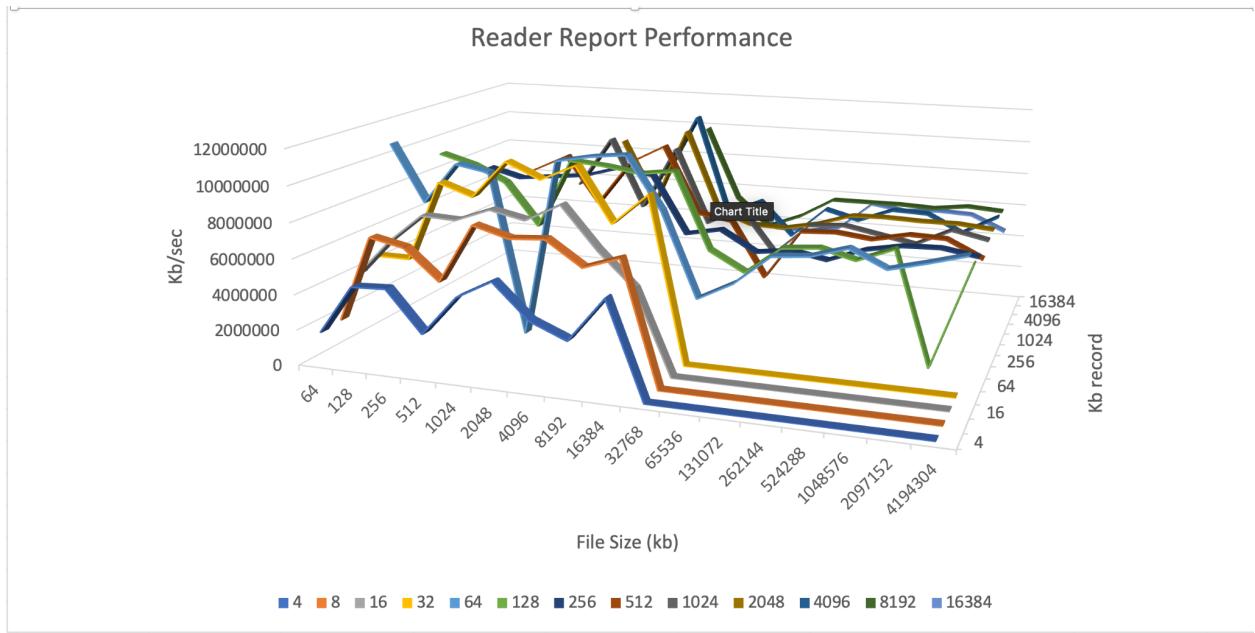


Figure 3: Reader Report Performance

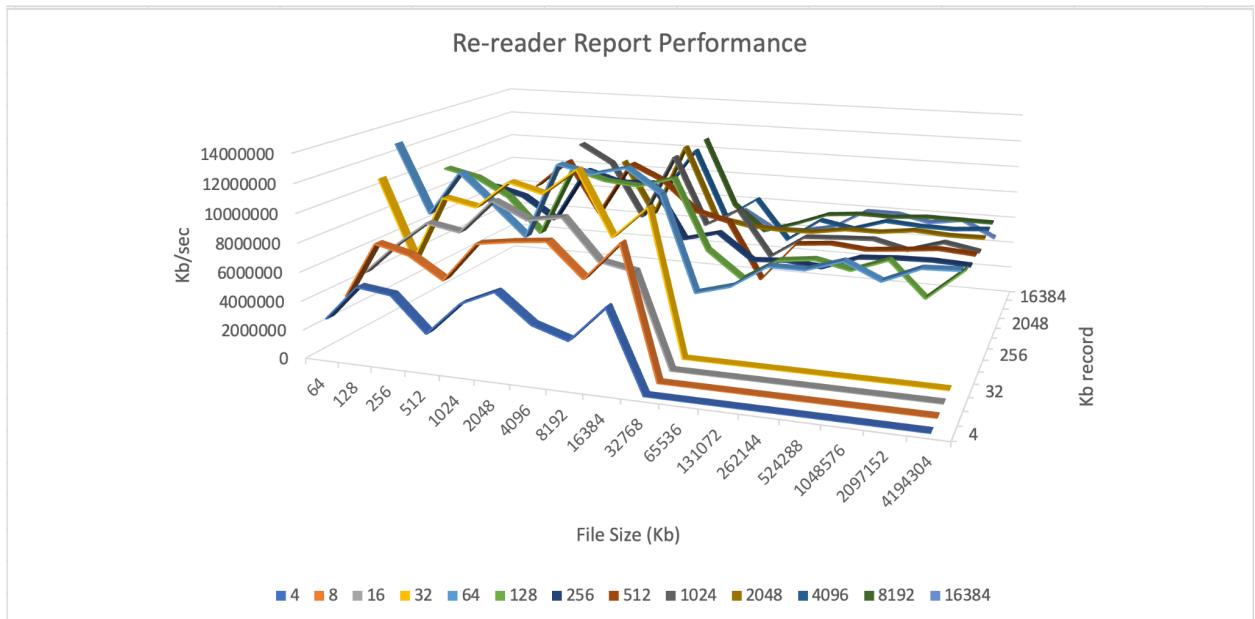


Figure 4: Re-reader Report Performance

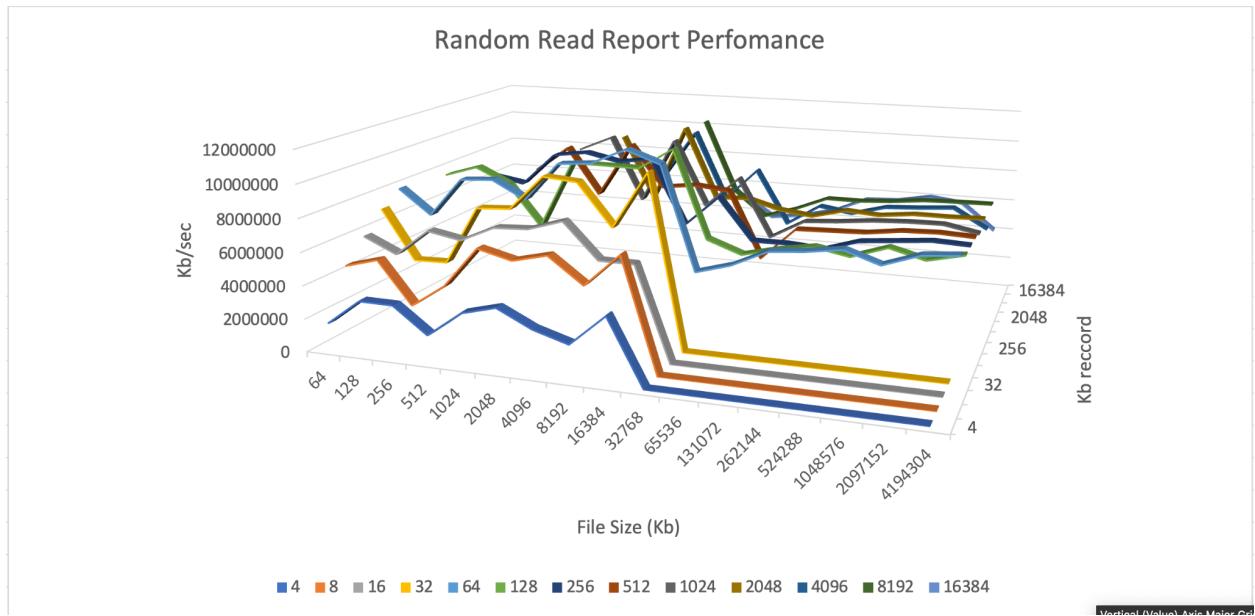


Figure 5: Random Read Report Performance

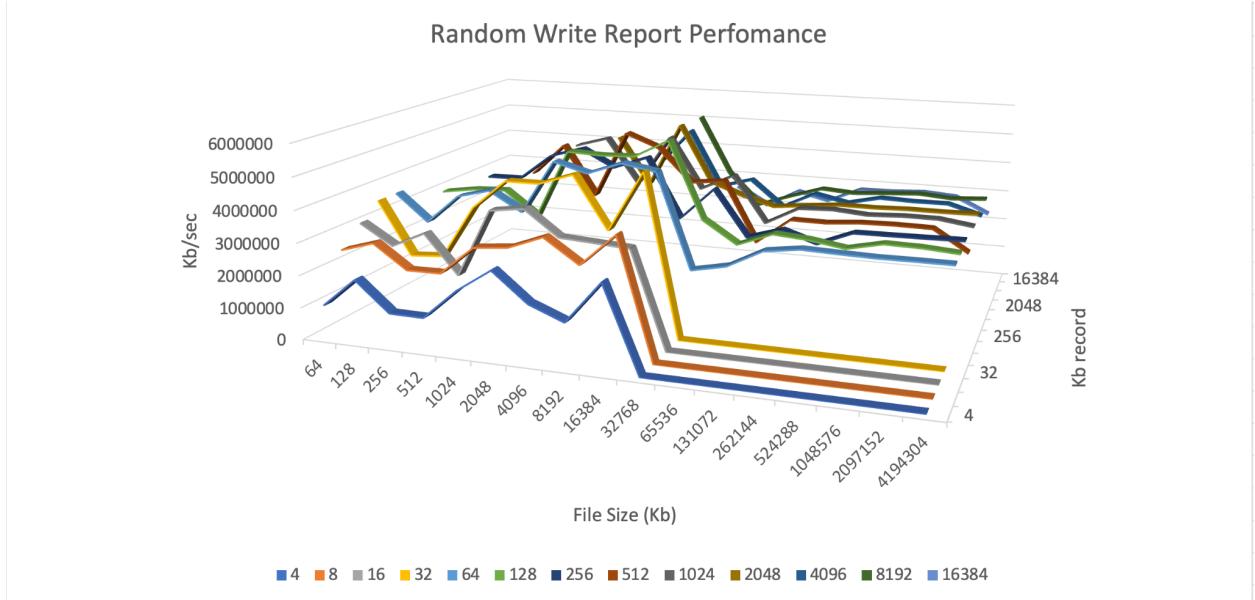


Figure 6: Random Write Report Performance

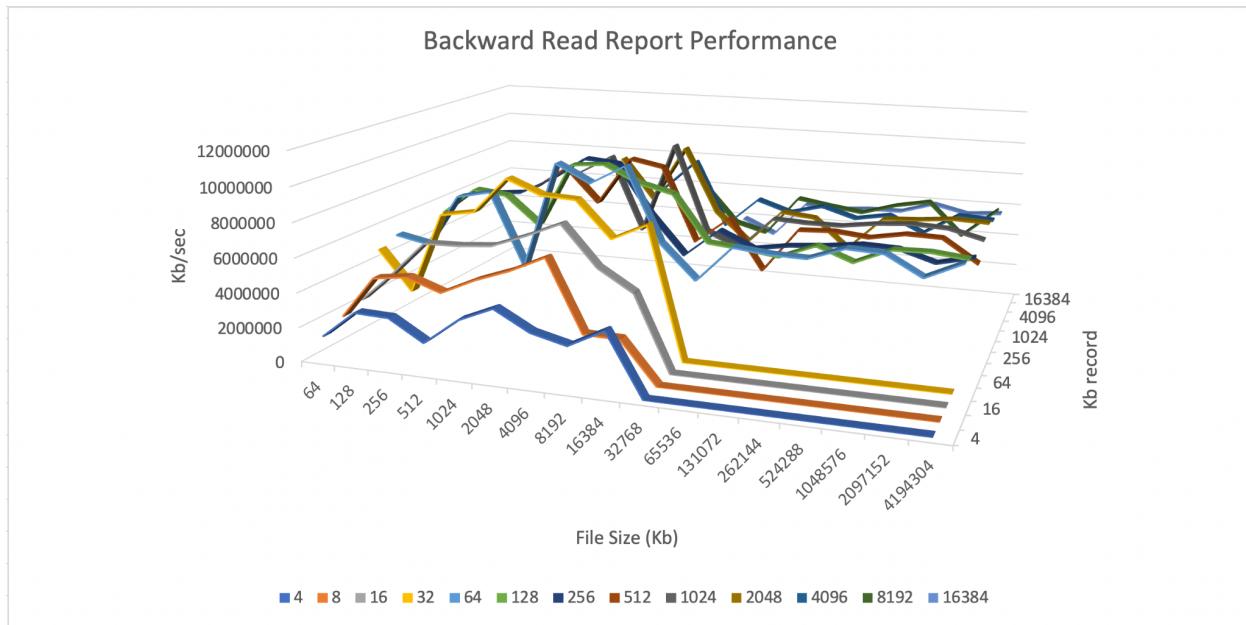


Figure 7: Backward Read Report Performance

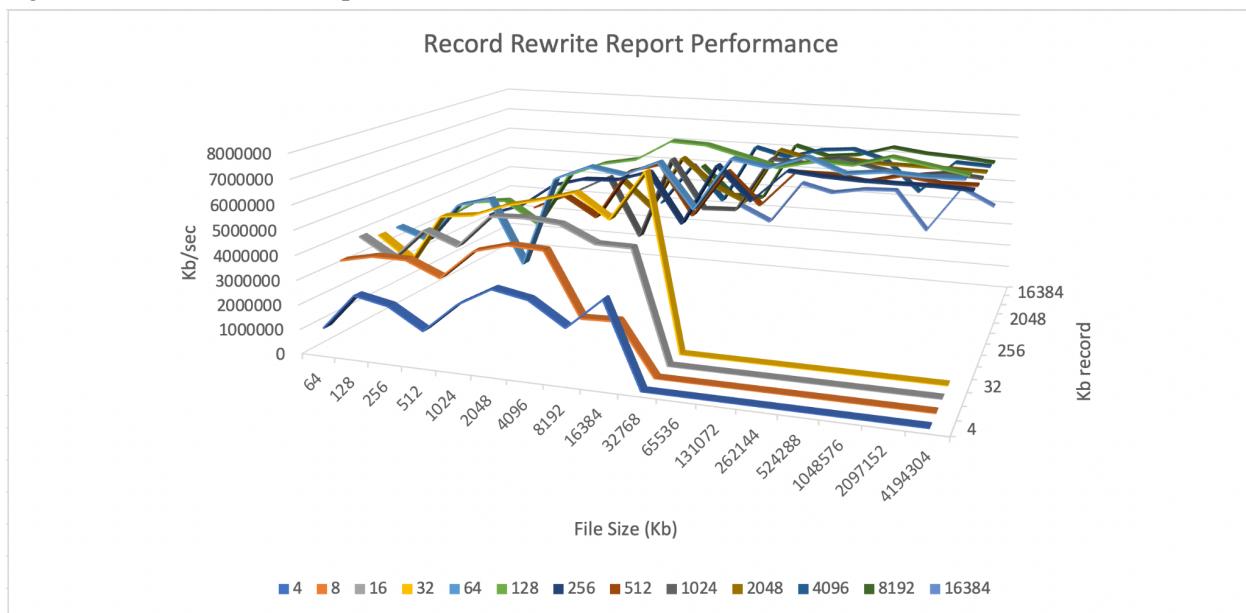


Figure 8: Record Rewrite Report Performance

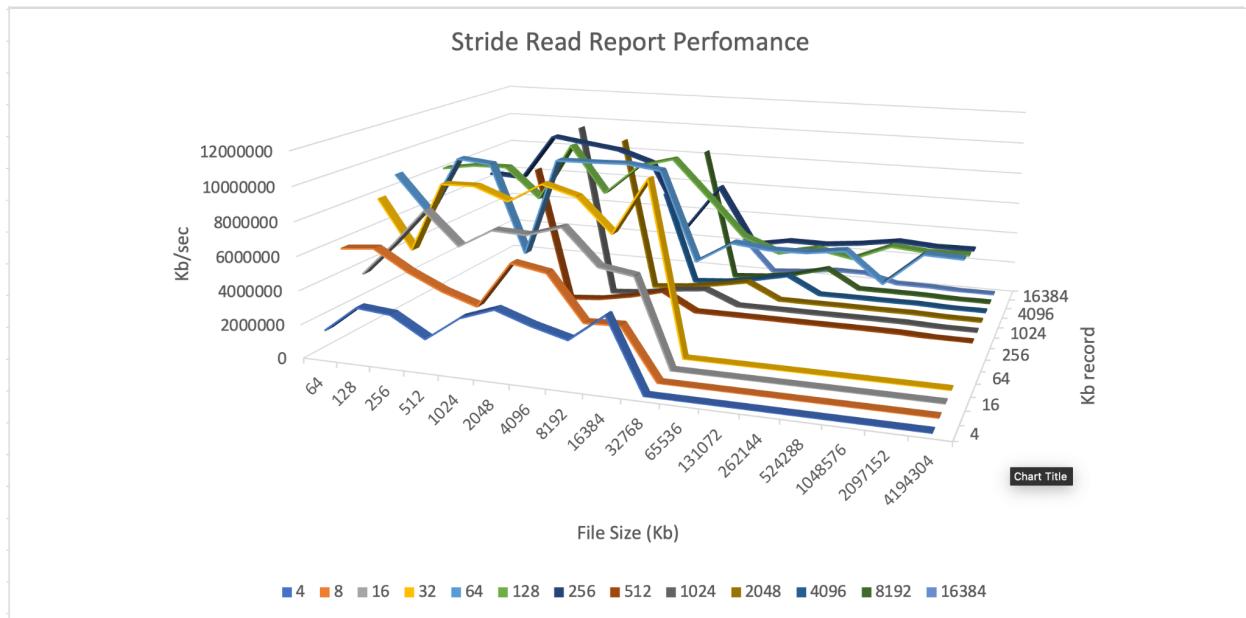


Figure 9: Stride Read Report Performance

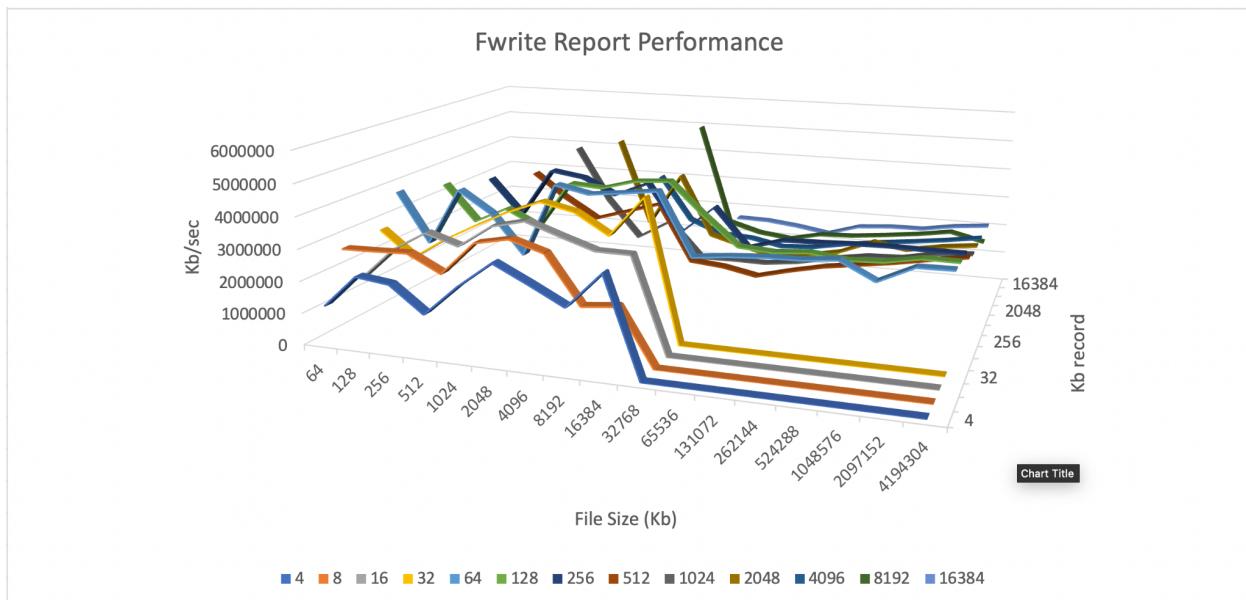


Figure 10: FWrite Report Performance

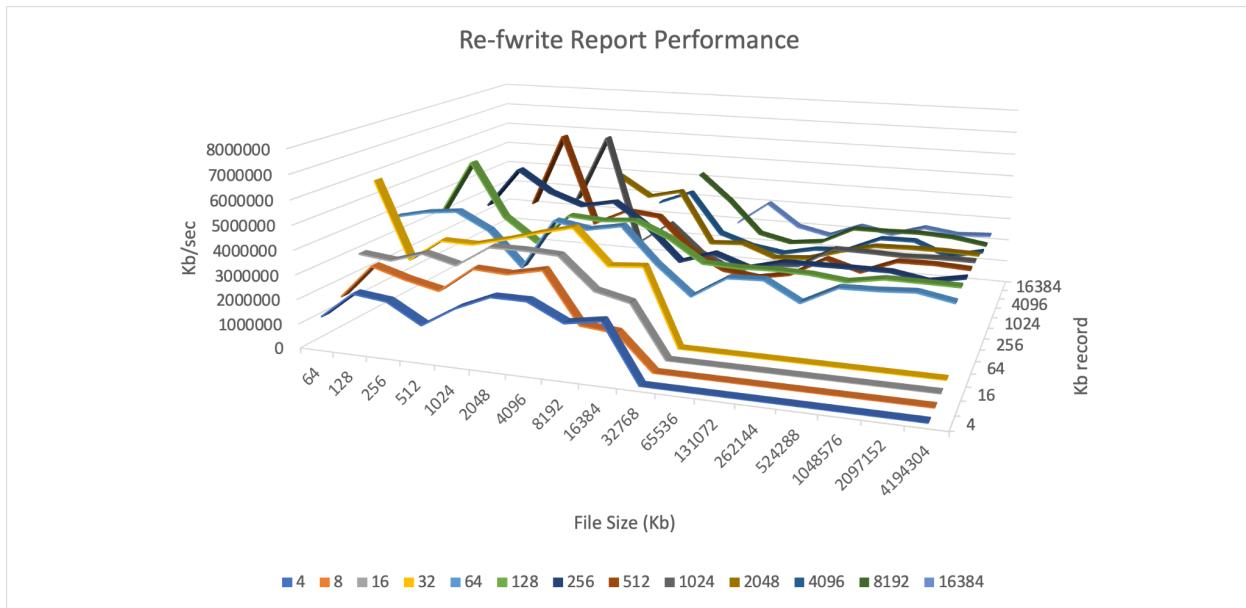


Figure 11: Re-fwrite Report Performance

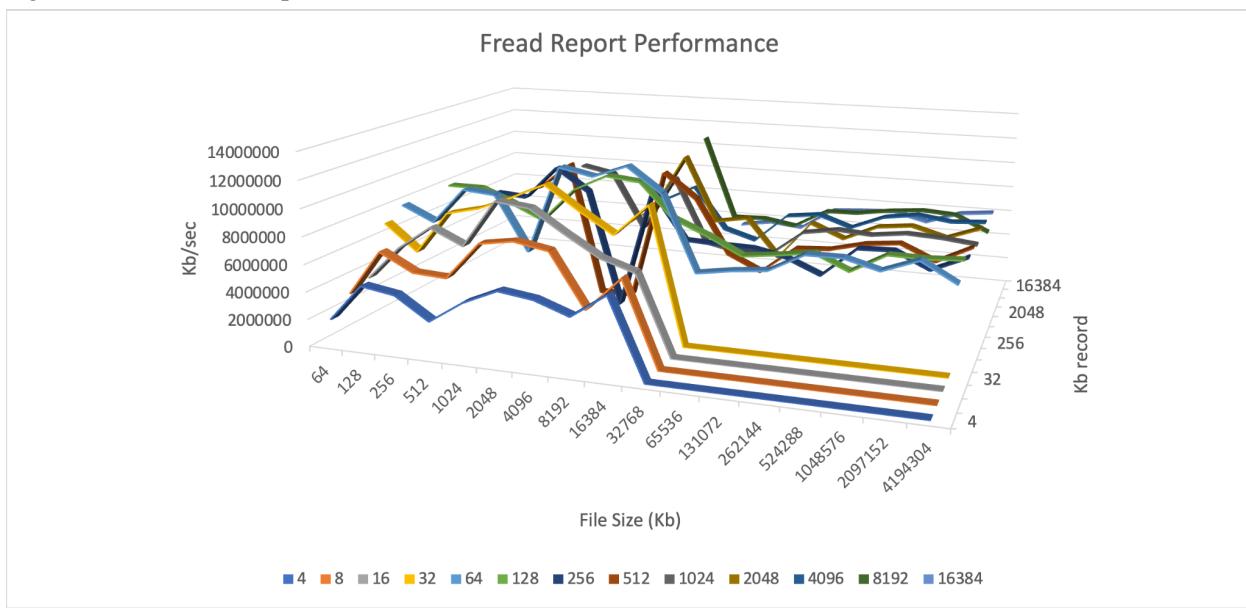


Figure 12: Fread Report Performance

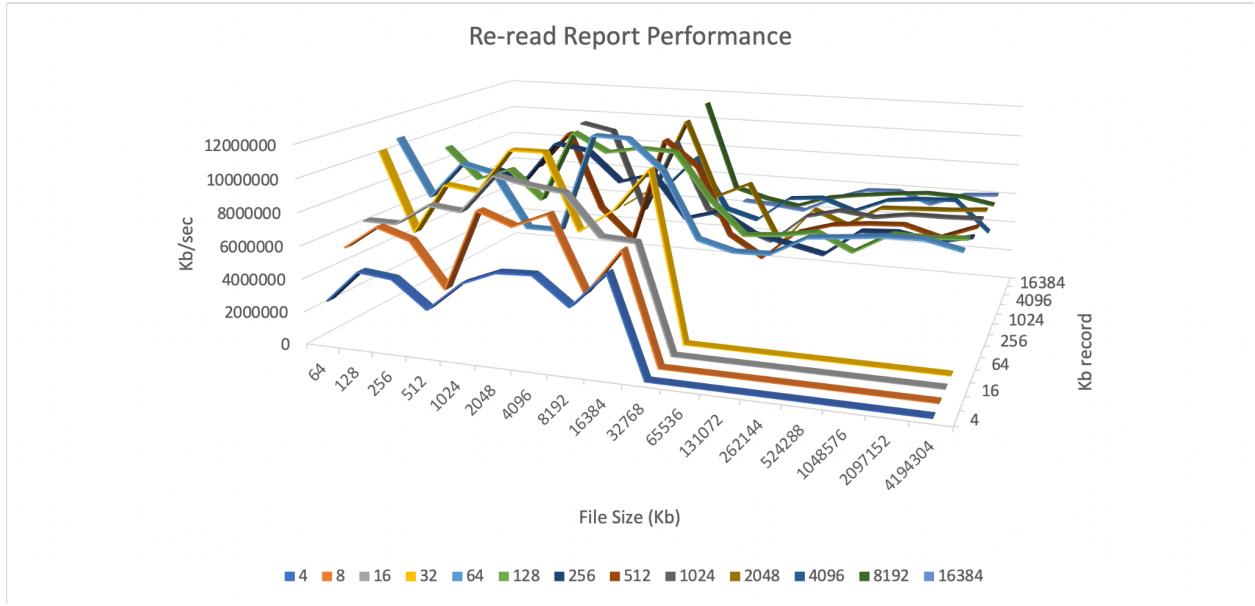


Figure 13: Re-Read Report Performance

Looking at all of the Reports that were generated from IOzone there is one thing that I noticed many had in common. All of the Reports seem to have dips in performance around the same file sizes. Particularly whenever the file size was 64, 512 and 8192. Though this didn't happen in all the scenarios and test that it ran i found it interesting that regardless of the test and Kb record they experience the dip in performance at those file sizes. This makes me believe that certain file size are more optimal for transferring data than others. The other thing that is almost a universal trend among all reports is that the performance is almost the same universally for all tests except for the few outliers Being record rewrite performance and writer performance. We also see that as we add more Kb records that the overall performance decreases in most cases when going to larger file sizes.

## Netperf:

```

linux1.engr.colostate.edu
File Edit View Search Terminal Help
pcrona@linux1 src]$ ./netserver -p 10104 -4
Starting netserver with host 'IN(6)ADDR_ANY' port '10104' and family AF_INET
pcrona@linux1 src]$ ./netperf -h
sage: netperf [global options] -- [test options]

global options:
-a send,recv      Set the local send,recv buffer alignment
-A send,recv      Set the remote send,recv buffer alignment
-B brandstr       Specify a string to be emitted with brief output
-c [cpu_rate]     Report local CPU usage
-C [cpu_rate]     Report remote CPU usage
-d               Increase debugging output
-D time,[units]   Display interim results at least every time interval
                  using units as the initial guess for units per second
                  A negative value for time will make heavy use of the
                  system's timestamping functionality
-f G|M|K|g|m|k  Set the output units
-F lfill,rfill    Pre-fill buffers with data from specified file
-H name[ip,fam]  Display this text
-i max,min        Specify the max and min number of iterations (15,1)
-l lvl,[intvl]   Specify confidence level (95 or 99) (99)
                 and confidence interval in percentage (10)
-j               Keep additional timing statistics
-l testlen        Specify test duration (d>0 secs) (<0 bytes|trans)
-L name[ip,fam]  Specify the local ipname and address family
-o send,recv      Set the local send,recv buffer offset
-s send,recv      Set the remote send,recv buffer offset
-n numcpu         Set the number of processors for CPU util
-N               Establish no control connection, do 'send' side only
-p port,lpport*  Set the netserver port number and/or local port
-P 0|1           Don't/Do display test headers
-r               Allow confidence to be hit on result only
-s seconds       Wait seconds between test setup and test start
-S set so keepalive on the data connection
-t testname      Specify test to perform
-T lcpu,rcpu    Request netperf/netserver be bound to local/remote cpu
-v verbosity     Specify the verbosity level
-W send,recv    Set the number of send,recv buffers
-v level        Set the verbosity level (default 1, min 0)
-V               Display the netperf version and exit
-Y local,remote  Set the socket priority
-Y local,remote  Set the IP TOS. Use hexadecimal.
-Z passphrase    Set and pass to netserver a passphrase

or those options taking two parms, at least one must be specified;
specifying one value without a comma will set both parms to that
value, specifying a value with a leading comma will set just the second
parm, a value with a trailing comma will set just the first. To set
ach parm to unique values, specify both and separate them with a
comma.

For these options taking two parms, specifying one value with no comma
will only set the first parms and will leave the second at the default

```

```

linux3.engr.colostate.edu
File Edit View Search Terminal Help
87388 16384 16384 10.03 925.64
[pcrona@linux3 src]$ ./netperf -H linux1 -p 10104 -t UDP_STREAM -- -P,10105
MIGRATED UDP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Socket Message Elapsed Messages
Size Size Time Okay Errors Throughput
bytes bytes secs. # # 10^6bits/sec
212992 65507 10.00 18326 0 960.27
212992 10.00 17623 923.43

[pcrona@linux3 src]$ ./netperf -H linux1 -p 10104 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Time Throughput
bytes bytes secs. 10^6bits/sec
87388 16384 16384 10.03 929.09
[pcrona@linux3 src]$ ./netperf -H linux1 -p 10104 -t TCP_RR -- -P,10105
MIGRATED TCP REQUEST/RESPONSE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET : fi
rst burst 0
Local /Remote
Socket Request Resp. Elapsed Trans.
Send Recv Size Size Time Rate
bytes Bytes bytes bytes secs. per sec
16384 87380 1 1 10.00 4486.49
16384 87380
[pcrona@linux3 src]$ ./netperf -H linux1 -p 10104 -t UDP_RR -- -P,10105
MIGRATED UDP REQUEST/RESPONSE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET : fi
rst burst 0
Local /Remote
Socket Request Resp. Elapsed Trans.
Send Recv Size Size Time Rate
bytes Bytes bytes bytes secs. per sec
212992 212992 1 1 10.00 4882.24
212992 212992

[pcrona@linux3 src]$ ./netperf -H linux1 -p 10104 -t TCP_SENDFILE -- -P,10105
TCP SENDFILE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Time Throughput
bytes bytes secs. 10^6bits/sec
87388 16384 16384 10.03 925.76
[pcrona@linux3 src]$ ./netperf -H linux1 -p 10104 -t TCP_MAERTS -- -P,10105
MIGRATED TCP MAERTS TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Time Throughput
bytes bytes secs. 10^6bits/sec
87388 16384 16384 10.00 505.84
[pcrona@linux3 src]$ 

```

Figure 14: netperf Server

```

linux5.engr.colostate.edu
File Edit View Search Terminal Help
Welcome to the Engineering Linux servers. See below for more info.
http://engineering.cs.colostate.edu/Engineering-Resources/linux-compute-servers
[pcrona@linux5 ~]$ cd netperf/install/
[pcrona@linux5 ~]$ ./netperf_install
netperf-netperf-2.7.0.tar.gz
[pcrona@linux5 ~]$ cd netperf-netperf-2.7.0
netperf-netperf-2.7.0.tar.gz
[pcrona@linux5 ~]$ cd netperf-netperf-2.7.0
[pcrona@linux5 ~]$ ./configure
[pcrona@linux5 ~]$ make
[pcrona@linux5 ~]$ make install
[pcrona@linux5 ~]$ ./netperf -p 10104 -t TCP_STREAM -- -P,10105
[pcrona@linux5 ~]$ ./netperf -H linux1 -p 10104 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Time Throughput
bytes bytes secs. 10^6bits/sec
87388 16384 16384 10.00 505.84
[pcrona@linux5 ~]$ ./netperf -H linux1 -p 10104 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Time Throughput
bytes bytes secs. 10^6bits/sec
87388 16384 16384 10.03 938.37
[pcrona@linux5 ~]$ ./netperf -H linux1 -p 10104 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Time Throughput
bytes bytes secs. 10^6bits/sec
87388 16384 16384 10.04 906.76
[pcrona@linux5 ~]$ ./netperf -H linux1 -p 10104 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Time Throughput
bytes bytes secs. 10^6bits/sec
87388 16384 16384 10.03 763.71
[pcrona@linux5 ~]$ ./netperf -H linux1 -p 10104 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Time Throughput
bytes bytes secs. 10^6bits/sec
87388 16384 16384 10.03 933.08
[pcrona@linux5 ~]$ ./netperf -H linux1 -p 10104 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Time Throughput
bytes bytes secs. 10^6bits/sec
87388 16384 16384 10.03 849.25
[pcrona@linux5 ~]$ 

```

Figure 15: Net perf Client

### Throughput ( $10^6$ ) vs length

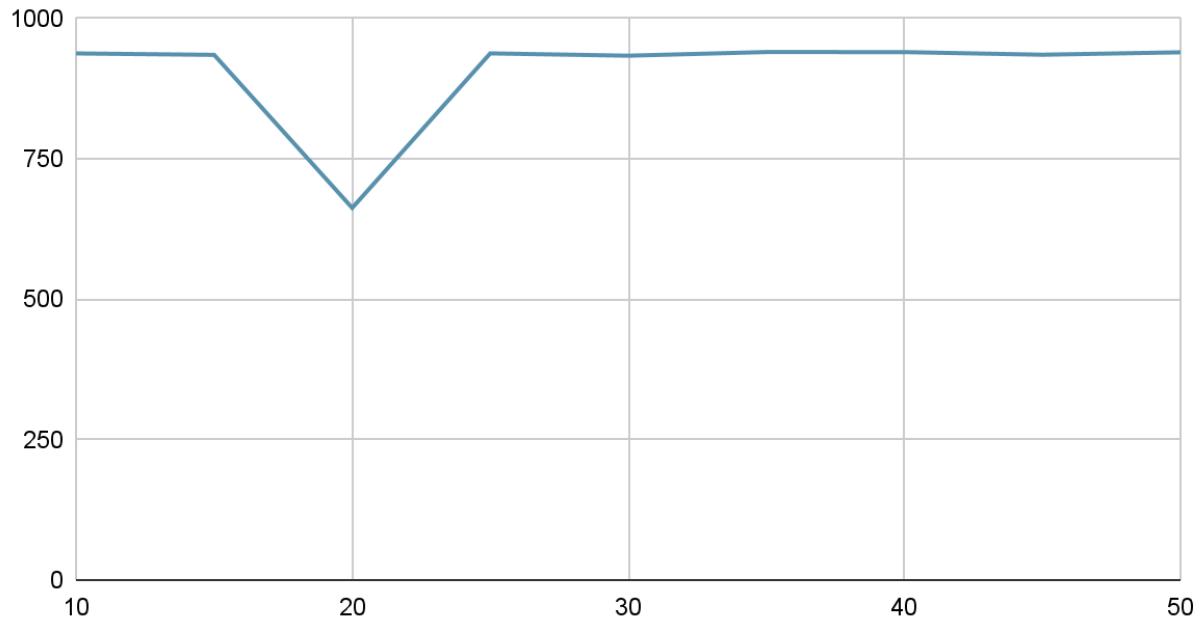


Figure 16: Netperf Throughput vs length

### Throughput ( $10^6$ ) vs buffer alignment

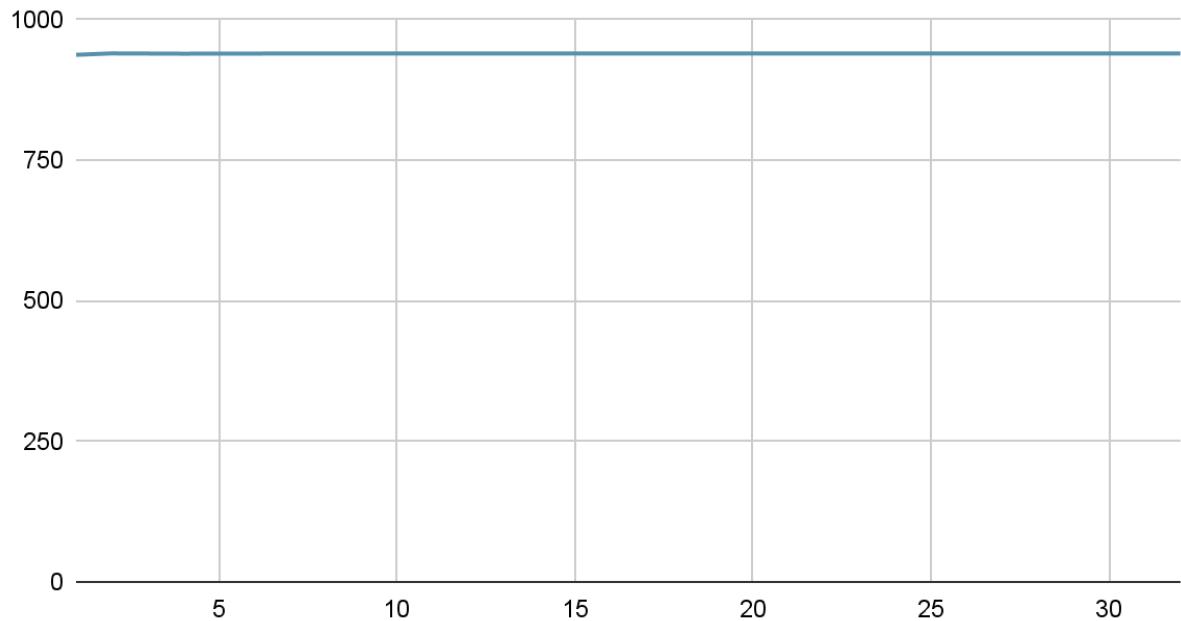


Figure 17: Netperf Throughput vs buffer alignment

```

*                                         Mate Terminal
File Edit View Search Terminal Help
Socket Socket Message Elapsed
Size  Size  Size  Time   Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 10.03 937.78
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -l 15 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linuxa1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size  Size  Size  Time   Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 15.03 935.23
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -l 20 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linuxa1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size  Size  Size  Time   Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 20.05 662.66
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -l 25 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linuxa1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size  Size  Size  Time   Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 25.03 937.89
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -l 30 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linuxa1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size  Size  Size  Time   Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 30.03 933.76
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -l 35 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linuxa1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size  Size  Size  Time   Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 35.03 940.25
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -l 40 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linuxa1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size  Size  Size  Time   Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 40.02 940.16
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -l 46 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linuxa1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size  Size  Size  Time   Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 46.02 935.49
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -l 50 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linuxa1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size  Size  Size  Time   Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 50.03 939.80

```

Figure 18: Netperf Throughput by varying -l

```

MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linuxa1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size  Size  Size  Time   Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 40.02 940.16
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -l 46 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linuxa1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size  Size  Size  Time   Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 46.02 935.49
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -l 50 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linuxa1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size  Size  Size  Time   Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 50.03 939.80

```

Figure 19:Netperf Throughput by varying -l part 2

```

[pcorona@linuxa3 src]$
[pcorona@linuxa3 src]$
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -a 1 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Size Time Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 10.03 937.50
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -a 2 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Size Time Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 10.02 940.01
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -a 4 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Size Time Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 10.03 939.49
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -a 8 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Size Time Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 10.02 939.94
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -a 16 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Size Time Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 10.03 939.82
[pcorona@linuxa3 src]$ ./netperf -H linux1 -p 10104 -a 32 -t TCP_STREAM -- -P,10105
MIGRATED TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to linux1.engr.colostate.edu () port 10105 AF_INET
Recv Send Send
Socket Socket Message Elapsed
Size Size Size Time Throughput
bytes bytes bytes secs. 10^6bits/sec

87380 16384 16384 10.02 939.88
[pcorona@linuxa3 src]$ █

```

Figure 20: Netperf Throughput by varying -a

One of the things that I came to notice about netperf is that it doesn't really matter which test is being run TCP or UDP, in either case they are about the same in the rate of throughput. The thing that made a difference in the throughput was the test that was being run. Interestingly enough we can see that both the buffer alignment and pause for a certain amount of running options don't affect the throughput of the test being run, interestingly only one of the pauses affected the throughput. The one thing which reliably brought the throughput down was attempting to use the same port and server from another machine. While this would fail on the new machine it reduced the throughput of the test which was running

## TTCP:

```
[pcorona@linuxb1 ttcp-1.0]$ ./ttcp -r -p 10120
tcp>r: buflen=8192, nbuf=2048, align=16384/0, port=10120  tcp
ttcp>r: socket
ttcp>r: accept from 129.82.20.172
hello
WOW
ttcp>r: 10 bytes in 12.19 real seconds = 0.00 KB/sec ===+
ttcp>r: 3 I/O calls, msec/call = 4160.80, calls/sec = 0.25
ttcp>r: 0.0user 0.0sys 0:12real 0% 0i+0d 472maxrss 0+1pf 3+0csw
[pcorona@linuxb1 ttcp-1.0]$ ./ttcp -r -p 10120
tcp>r: buflen=8192, nbuf=2048, align=16384/0, port=10120  tcp
ttcp>r: socket
ttcp>r: accept from 129.82.20.172
hello
this one is interesting
we can work on typing things out
[pcorona@linuxb1 ttcp-1.0]$ ./ttcp -r -p 10120 -u
tcp>r: buflen=8192, nbuf=2048, align=16384/0, port=10120  udp
ttcp>r: socket
hello

```

The terminal window title is "Mate Terminal". It shows the output of several ttcp processes. One process fails with "Permission denied". Another process succeeds and prints "Hello" and "this one is interesting". A third process succeeds and prints "Hello". The final command "./ttcp -r -p 10120 -u" is shown at the bottom.

Figure 21: TTCP Connection with TCP and UDP

```
linux6.engr.colostate.edu
ttcp -r [-options > out]
Common options:
  -l ## length of bufs read from or written to network (default 8192)
  -u use UDP instead of TCP
  -p # port number to send to or listen at (default 5001)
  -s # source port to use
    -r: sink (discard) all data from network
  -i #: "interval": number of seconds to run the test, rather than number of buffers
  -A align the start of buffers to this modulus (default 16384)
  -O start buffers at this offset from the modulus (default 0)
  -v verbose: print more statistics
  -d set DEBUG socket option
  -b #: set socket buffer size (if supported)
    -r: for rate: b,B = (bit,byte); K,M = kilo(bit,byte); m,M = mega; g,G = giga; r,R = raw (bit,byte)
Options specific to -t:
  -n## number of source bufs written to network (default 2048)
  -D don't buffer TCP writes (sets TCP_NODELAY socket option)
Options specific to -r:
  -B for -s, only output full blocks as specified by -l (for TAR)
  -T "touch": access each byte as it's read
  -c "collect call": initiate the TCP connection but then expect data sent from the other end
    -q "quiet": only print the measured throughput, emit no other chatty output.
[pcorona@linuxb1 ttcp-1.0]$ ./ttcp -r -p 10120
tcp>r: buflen=8192, nbuf=2048, align=16384/0, port=10120  tcp
ttcp>r: socket
ttcp>r: accept from 129.82.20.172
Hello
what is going on my dude
Hello
ttcp>r: 31 bytes in 112.70 real seconds = 0.00 KB/sec ===+
ttcp>r: 3 I/O calls, msec/call = 38837.51, calls/sec = 0.03
ttcp>r: 0.0user 0.0sys 0:53real 0% 0i+0d 472maxrss 0+1pf 3+0csw
[pcorona@linuxb1 ttcp-1.0]$ hello
Hello: Command not found.
[pcorona@linuxb1 ttcp-1.0]$ ./ttcp -r -p 10120
tcp>r: buflen=8192, nbuf=2048, align=16384/0, port=10120  tcp
ttcp>r: socket
ttcp>r: accept from 129.82.20.172
sdhfhahdh;fdhsjkfndch
sdnfvjadsbfkjdsbjfjabsdf
dsanfjkabgjhkadsbfjghadfjkdsnfjs
dfanvjkdsfbqjhkdfabkhvckdahvb sdhj
sadfjkdsfhjkasdhlfjahsfjf
sadnfjkdsabfjhdsbjfhksahnjdfghksda
http>r: 1000 bytes in 25.62 real seconds = 0.00 KB/sec ===+
ttcp>r: 1 I/O calls, msec/call = 3747.51, calls/sec = 0.02
ttcp>r: 0.0user 0.0sys 0:25real 0% 0i+0d 472maxrss 0+1pf 7+0csw
[pcorona@linuxb1 ttcp-1.0]$ ./ttcp -r -p 10120
tcp>r: buflen=8192, nbuf=2048, align=16384/0, port=10120  tcp
ttcp>r: socket
ttcp>r: accept from 129.82.20.172
Hello
who is the master
ttcp>r: 26 bytes in 37.60 real seconds = 0.00 KB/sec ===+
ttcp>r: 3 I/O calls, msec/call = 12835.56, calls/sec = 0.08
ttcp>r: 0.0user 0.0sys 0:37real 0% 0i+0d 472maxrss 0+1pf 3+0csw
[pcorona@linuxb1 ttcp-1.0]$ 
```

The left terminal window shows a successful connection attempt with a throughput of 0.00 KB/sec. The right terminal window shows a failed connection attempt with an error message "errno104" and "IO: Connection reset by peer".

Figure 22: TTCP Connection fails with two TCP connections

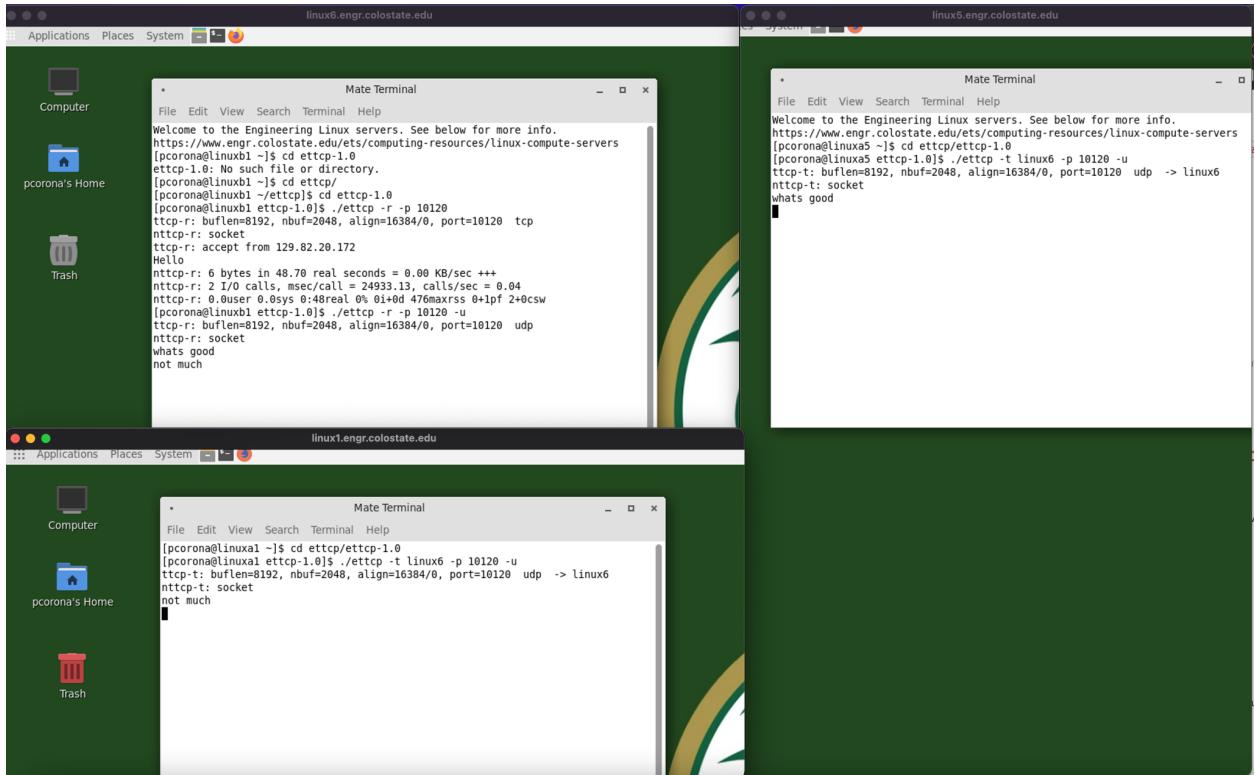


Figure 23: TTCP UDP functioning with UDP two client

The Things that I found most notable about running the tests for TTCP was that it was relatively similar to some of the programs which we have coded. It essentially works as a message board and is sending strings which are then displayed on the server. When this was run the thing in particular that I found interesting was that TCP was incapable of having two connections at the same time, however, attempting to connect from another client did not seem to have any kind of effect on the overall performance of the system. The client which tried to connect to the server while it was busy simply could not connect and would have its connection reset once the other client finished. UDP on the other hand was able to have multiple clients sending information and messages at the same time.

## Iperf:

The figure displays three terminal windows from different Linux hosts (linux1, linux2, and linux3) performing a UDP connection test to a common server port (10102). The output shows the transfer of 1470 byte datagrams over UDP with a buffer size of 208 KByte (default).

```

[pcorona@linux1 ~]$ iperf -s -p 10102 -u
Server listening on UDP port 10102
Receiving 1470 byte datagrams
UDP buffer size: 208 KByte (default)
[ 3] local 129.82.20.167 port 10102 connected with 129.82.20.168 port 60882
[ ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams
[ 3] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec 0.010 ms 0/ 892 (0%)
[ 4] local 129.82.20.167 port 10102 connected with 129.82.20.169 port 54771
[ 4] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec 0.016 ms 0/ 892 (0%)
[ 3] local 129.82.20.167 port 10102 connected with 129.82.20.169 port 55466
[ 4] local 129.82.20.167 port 10102 connected with 129.82.20.168 port 53742
[ 3] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec 0.021 ms 0/ 892 (0%)
[ 4] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec 0.014 ms 0/ 892 (0%)
[pcorona@linux2 ~]$ iperf -c linux1.engr.colostate.edu -p 10102
Client connecting to linux1.engr.colostate.edu, UDP port 10102
Sending 1470 byte datagrams, IPG target: 11215.21 us (Kalman adjust)
UDP buffer size: 208 KByte (default)
[ 3] local 129.82.20.168 port 60882 connected with 129.82.20.167 port 10102
[ ID] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec
[ 3] Sent 892 datagrams
[ 3] Server Report:
[ 3] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec 0.009 ms 0/ 892 (0%)
[pcorona@linux2 ~]$ iperf -c linux1.engr.colostate.edu -p 10102
connect failed: Connection refused
[pcorona@linux2 ~]$ iperf -c linux1.engr.colostate.edu -p 10102 -u
Client connecting to linux1.engr.colostate.edu, UDP port 10102
Sending 1470 byte datagrams, IPG target: 11215.21 us (Kalman adjust)
UDP buffer size: 208 KByte (default)
[ 3] local 129.82.20.168 port 60882 connected with 129.82.20.167 port 10102
[ ID] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec
[ 3] Sent 892 datagrams
[ 3] Server Report:
[ 3] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec 0.014 ms 0/ 892 (0%)
[pcorona@linux2 ~]$ 

[pcorona@linux3 ~]$ iperf -c linux1.engr.colostate.edu -p 10102
Client connecting to linux1.engr.colostate.edu, UDP port 10102
Sending 1470 byte datagrams, IPG target: 11215.21 us (Kalman adjust)
UDP buffer size: 208 KByte (default)
[ 3] local 129.82.20.169 port 54771 connected with 129.82.20.167 port 10102
[ ID] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec
[ 3] Sent 892 datagrams
[ 3] Server Report:
[ 3] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec 0.015 ms 0/ 892 (0%)
[pcorona@linux3 ~]$ iperf -c linux1.engr.colostate.edu -p 10102 -u
Client connecting to linux1.engr.colostate.edu, UDP port 10102
Sending 1470 byte datagrams, IPG target: 11215.21 us (Kalman adjust)
UDP buffer size: 208 KByte (default)
[ 3] local 129.82.20.169 port 55466 connected with 129.82.20.167 port 10102
[ ID] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec
[ 3] Sent 892 datagrams
[ 3] Server Report:
[ 3] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec 0.020 ms 0/ 892 (0%)
[pcorona@linux3 ~]$ 

```

Figure 24: Iperf UDP Server connection with two clients

The figure displays three terminal windows from different Linux hosts (linux1, linux2, and linux3) performing a TCP connection test to a common server port (10102). The output shows the transfer of 1470 byte datagrams over TCP with a window size of 85.3 KByte (default).

```

[pcorona@linux1 ~]$ iperf -s -p 10102
Server listening on TCP port 10102
TCP window size: 85.3 KByte (default)
[ 4] local 129.82.20.167 port 10102 connected with 129.82.20.168 port 35362
[ ID] Interval Transfer Bandwidth
[ 4] 0.0-10.0 sec 1.16 GBytes 941 Mbits/sec
[ 4] local 129.82.20.167 port 10102 connected with 129.82.20.169 port 42118
[ 5] local 129.82.20.167 port 10102 connected with 129.82.20.168 port 35426
[ 4] 0.0-10.1 sec 854 MBytes 711 Mbits/sec
[ 5] 0.0-10.0 sec 686 MBytes 574 Mbits/sec
[pcorona@linux2 ~]$ iperf -c linux1.engr.colostate.edu -p 10102
Client connecting to linux1.engr.colostate.edu, TCP port 10102
TCP window size: 714 Kbyte (default)
[ 3] local 129.82.20.168 port 35362 connected with 129.82.20.167 port 10102
[ ID] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 1.10 GBytes 943 Mbits/sec
[pcorona@linux2 ~]$ iperf -c linux1.engr.colostate.edu -p 10102
Client connecting to linux1.engr.colostate.edu, TCP port 10102
TCP window size: 272 Kbyte (default)
[ 3] local 129.82.20.168 port 35426 connected with 129.82.20.167 port 10102
[ ID] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 686 MBytes 575 Mbits/sec
[pcorona@linux2 ~]$ iperf -c linux1.engr.colostate.edu -p 10102
Client connecting to linux1.engr.colostate.edu, UDP port 10102
Sending 1470 byte datagrams, IPG target: 11215.21 us (Kalman adjust)
UDP buffer size: 208 KByte (default)
[ 3] local 129.82.20.168 port 37383 connected with 129.82.20.167 port 10102
read failed: Connection refused
[ 3] WARNING: did not receive ack of last datagram after 1 tries.
[ ID] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec
[ 3] Sent 892 datagrams
[pcorona@linux2 ~]$ 

[pcorona@linux3 ~]$ iperf -c linux1.engr.colostate.edu -p 10102
Client connecting to linux1.engr.colostate.edu, UDP port 10102
TCP window size: 586 KByte (default)
[ 3] local 129.82.20.169 port 49431 connected with 129.82.20.167 port 10102
read failed: Connection refused
[ 3] WARNING: did not receive ack of last datagram after 1 tries.
[ ID] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 1.25 MBytes 1.05 Mbits/sec
[ 3] Sent 892 datagrams
[pcorona@linux3 ~]$ iperf -c linux1.engr.colostate.edu -p 10102
Client connecting to linux1.engr.colostate.edu, TCP port 10102
TCP window size: 586 KByte (default)
[ 3] local 129.82.20.169 port 42118 connected with 129.82.20.167 port 10102
[ ID] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 854 MBytes 715 Mbits/sec

```

Figure 25: Iperf TCP Server connection with client

One of the more interesting things that I noticed when using Iperf to test the bandwidth of the connection between the server and client was that the UDP connection did not experience much performance degradation when two clients connected at the same time and remained at a relatively similar performance as when only one connection was present. However this is not the case with TCP. While Iperf can handle two TCP connections at the same time it does experience some performance degradation when there were two connections. While the first client experiences a decent performance losing about 20% the second client experiences a drastic performance loss of about 40%.