

File IO: Evaluation of Changes

Michael Walker and Brian White
October 2, 2000

Testing

In gathering data for IO performance evaluation, we tested nets in five different stages:

- (1) Pre-1.7 Legion compiled, unoptimized;
- (2) Pre-1.7 Legion compiled with `-O3` optimization;
- (3) Pre-1.7 Legion with OpenSSL and `-O3` optimization;
- (4) Pre-1.7 Legion with OpenSSL, IO modifications, and `-O3` optimization;
- (5) Pre-1.7 Legion with OpenSSL, IO, and mmps modifications, and `-O3` on.

Nets (3-5) were originally compiled with `-O3` optimization, but this led to problems in the security layer (`LegionRSAKey.c`). We recompiled only the Legion library without optimization in order to reconcile the problem.

We aimed to quantify what contribution each stage of modification made to IO performance. The final net, (5), can be considered a summary of the cumulative changes in performance, and an indicator of future Legion IO performance (i.e. in 1.7).

We ran the testing as follows. Each 40-host net was set up on centurion during the evening, when the load on each node was less than .5. In general, the load on any given node was very close to zero. We ran tests for the 1 file and the N files cases on the net, recorded statistics, and shut it down. Each test executed 10 MB `BasicFileObject` reads (10 reads of 1 MB), and timed the latency. The tests were run for n readers, where $n = \{1, 5, 10, 20\}$. One test measured the latency of n readers accessing the same file, while the other measured the latency of n readers accessing n separate files on n hosts. The readers and files were always run on separate hosts. We repeated the process for every net.

All testing was done on the same evening – 9/28/00. This was intentionally done to ensure data integrity, as testing of this nature can be highly sensitive to changes in the network over time. This resulted in a low standard deviation, which is summarized in figure 1.

1 file, N readers	Average Standard Deviation for latency (sec)	
	1	0.108664425
	5	0.054387805
	10	0.082529112
	20	0.243861841
N files, N readers		
	1	0.246627424
	5	0.225307258
	10	0.334566393
	20	0.444914493

Figure 1. Average Standard Deviation for tests.

Trends

Although a group meeting would be appropriate to discuss these trends, we will make a few initial observations.

When there is a one-to-one correspondence of files to readers, the SSL and IO nets made moderate improvement to IO transfer rates, whereas the sliding window mmps protocol made very significant improvements in the same case. During runs with one reader on one file, for instance, the mmps sliding window net adds a total 73% relative increase in performance over the unoptimized net, and a 45% relative increase over its next competitor, the IO net. This can be attributed to the pipelining introduced by the sliding window implementation.

However, when multiple readers access one file, OpenSSL makes the major contribution to the performance boost, while the sliding window protocol adds relatively little. We theorize that multiple readers using the sliding window protocol saturate the server faster than other methods because it supports twice as many outgoing requests.

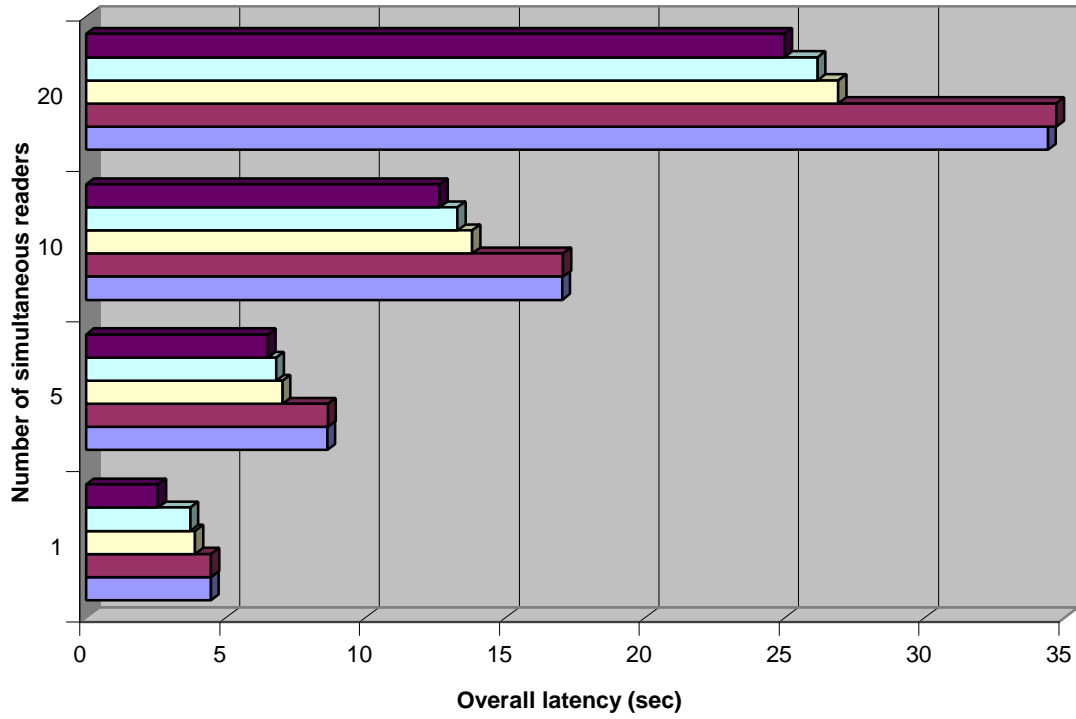
It is surprising to note that SSL seemed to contribute a greater change to IO read performance than our IO changes did; perhaps security was a larger source of latency than excessive memcopies. One might hypothesize that the OpenSSL performance increase will similarly influence many more operations in Legion, since it is fairly pervasive, although this has yet to be quantitatively shown.

We are surprised that both the IO and sliding window nets level off at different bandwidths. Out of simplicity and lack of knowledge, we assume that mmps ALWAYS sends 2 packets. Therefore, we would assume that the server would see the same number of packets per time unit under the 5 reader mmps net and the 10 reader IO net. Yet, these two nets do not show the same bandwidth. We realize our argument is simplified. We're interested in hearing more details about mmps.

The improvements of each net are amplified by the "N files, N readers" scalability tests. None of the nets show any signs of failure to scale, and the end result is proof: sliding windows mmps net achieved 64.3 MB aggregate bandwidth in the 20 reader case.

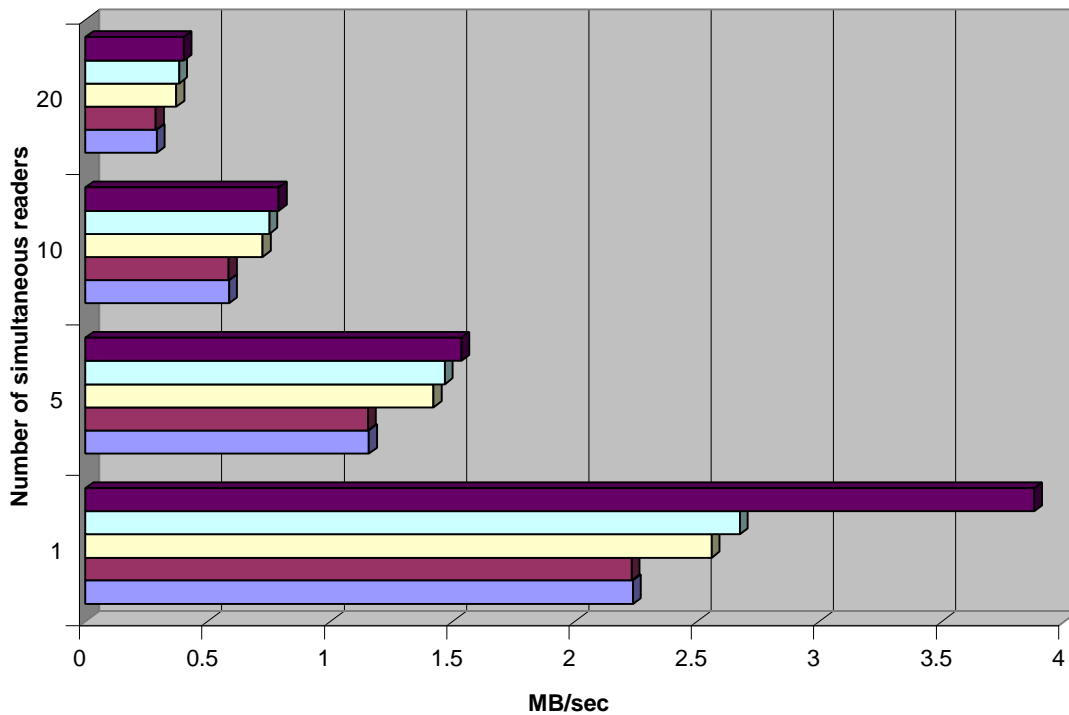
The "1 file, N readers: A different view" chart was put together to help approximate the theoretical limits of bandwidth on a particular file. In the best case (sliding window), bandwidth approaches slightly over 8 MB/sec. In our original experiments, the unoptimized original net approached 4.1 MB/sec. The same original net performed better this time, approaching 5.8 MB/sec. This may have been due to the excellent testing conditions of the more recent runs, allowing for more throughput.

1 file, N readers: Overall latency



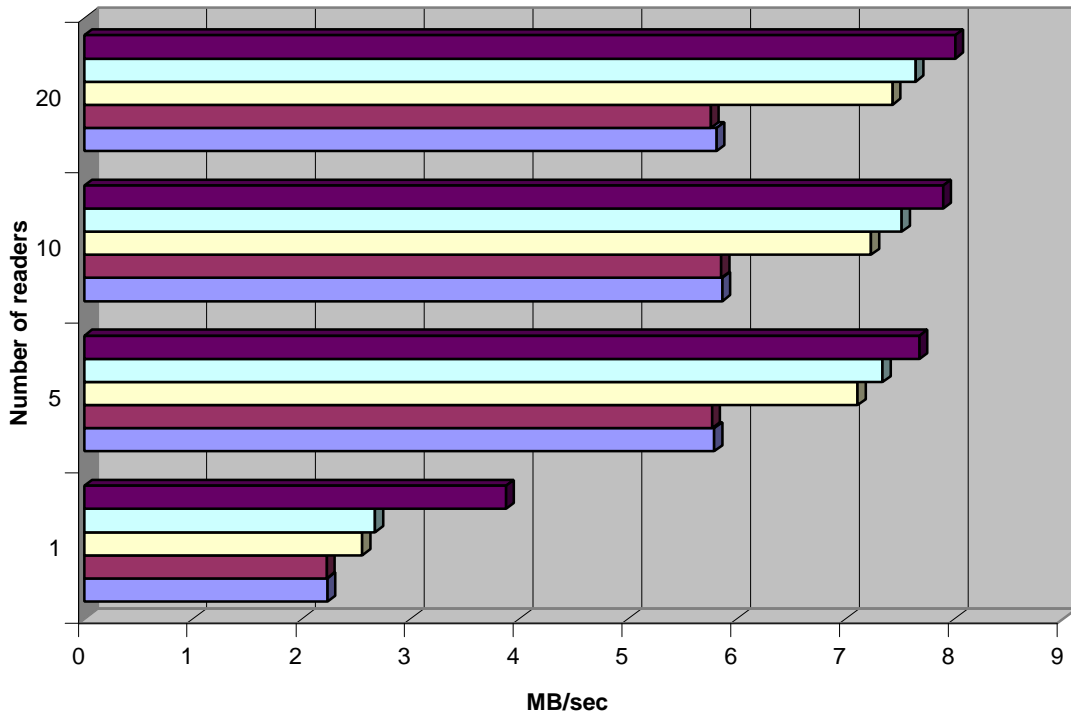
	1	5	10	20
■ Sliding window	2.58	6.51	12.66	24.97
□ IO additions	3.74	6.81	13.3	26.16
□ OpenSSL	3.91	7.03	13.82	26.91
■ Optimized	4.48	8.66	17.07	34.71
■ Unoptimized	4.47	8.63	17.04	34.4

1 tile, N readers: Single reader bandwidth



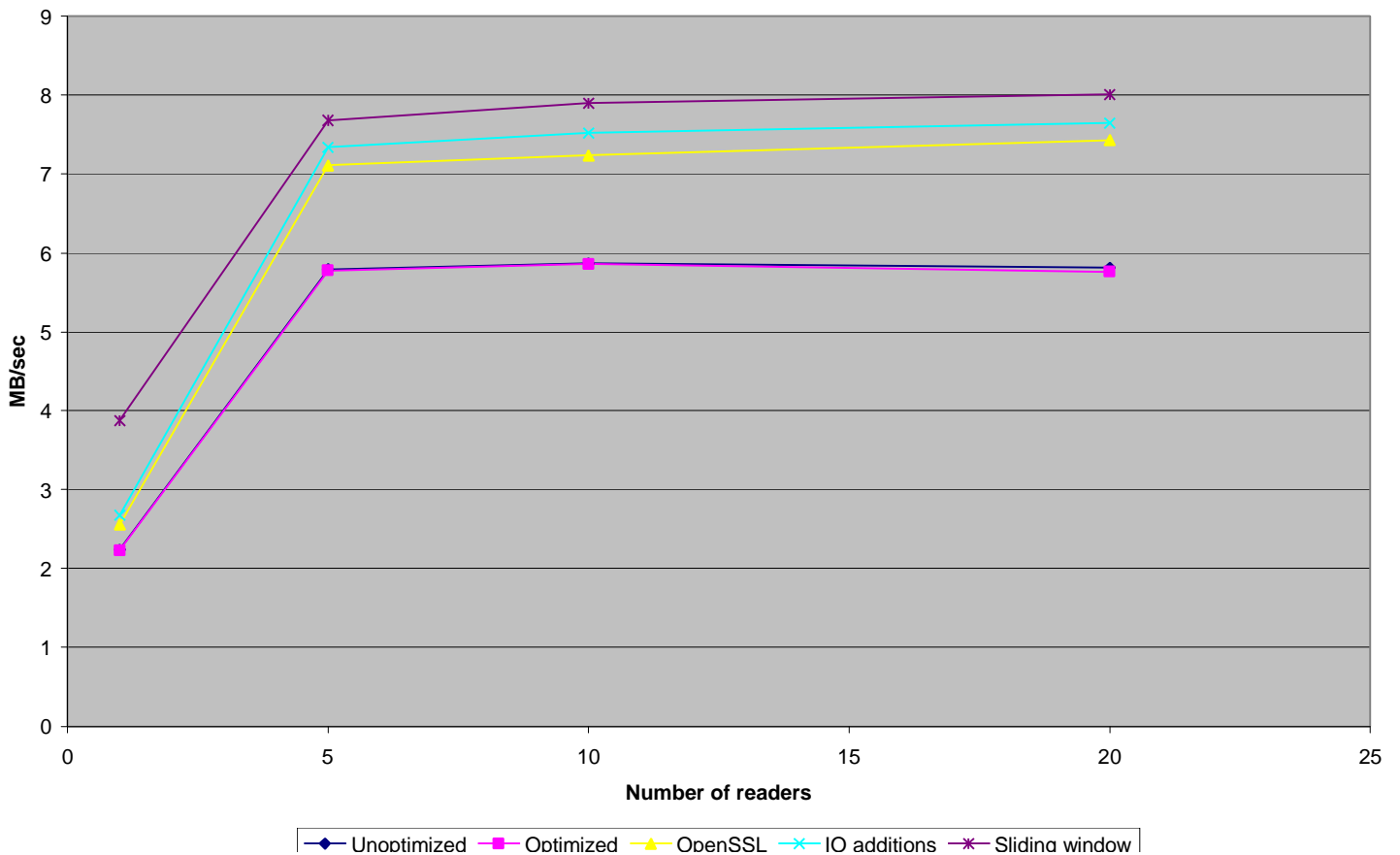
	1	5	10	20
■ Sliding window	3.875968992	1.53609831	0.789889415	0.400480577
■ IO additions	2.673796791	1.468428781	0.751879699	0.382262997
■ OpenSSL	2.557544757	1.422475107	0.723589001	0.371609067
■ Optimized	2.232142857	1.154734411	0.585823081	0.288101412
■ Unoptimized	2.237136465	1.158748552	0.58685446	0.290697674

1 file, N readers: Overall bandwidth

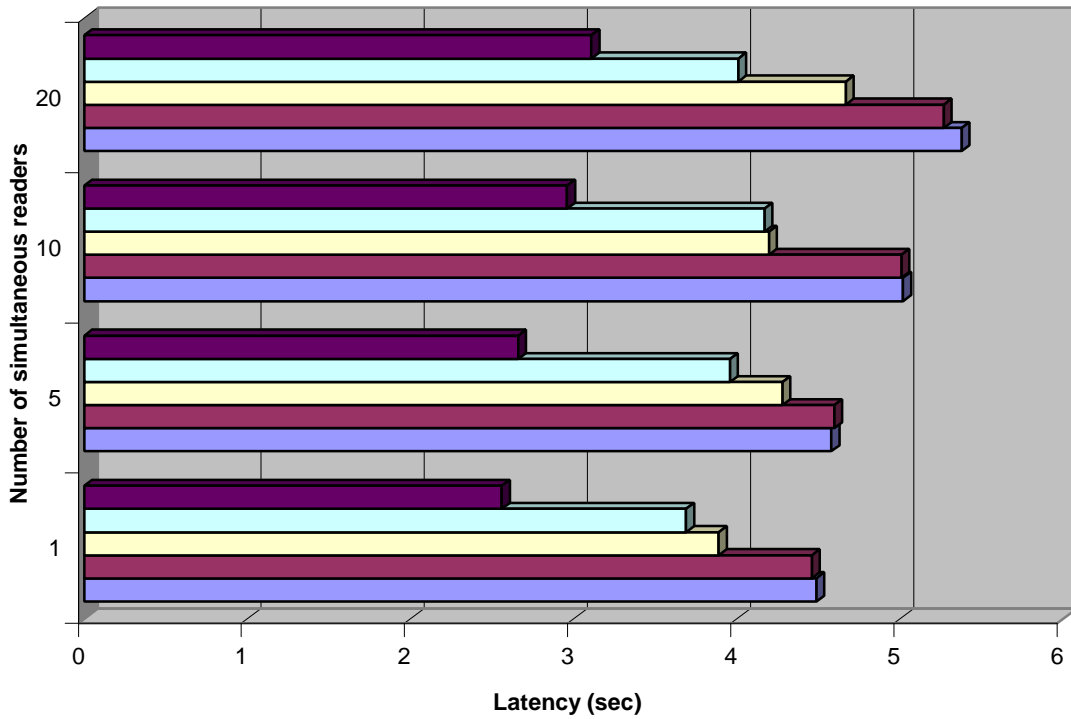


	1	5	10	20
■ Sliding window	3.875968992	7.680491551	7.898894155	8.009611534
□ IO additions	2.673796791	7.342143906	7.518796992	7.645259939
□ OpenSSL	2.557544757	7.112375533	7.235890014	7.432181345
■ Optimized	2.232142857	5.773672055	5.858230814	5.762028234
■ Unoptimized	2.237136465	5.793742758	5.868544601	5.813953488

1 file, N readers: Overall bandwidth: A different view

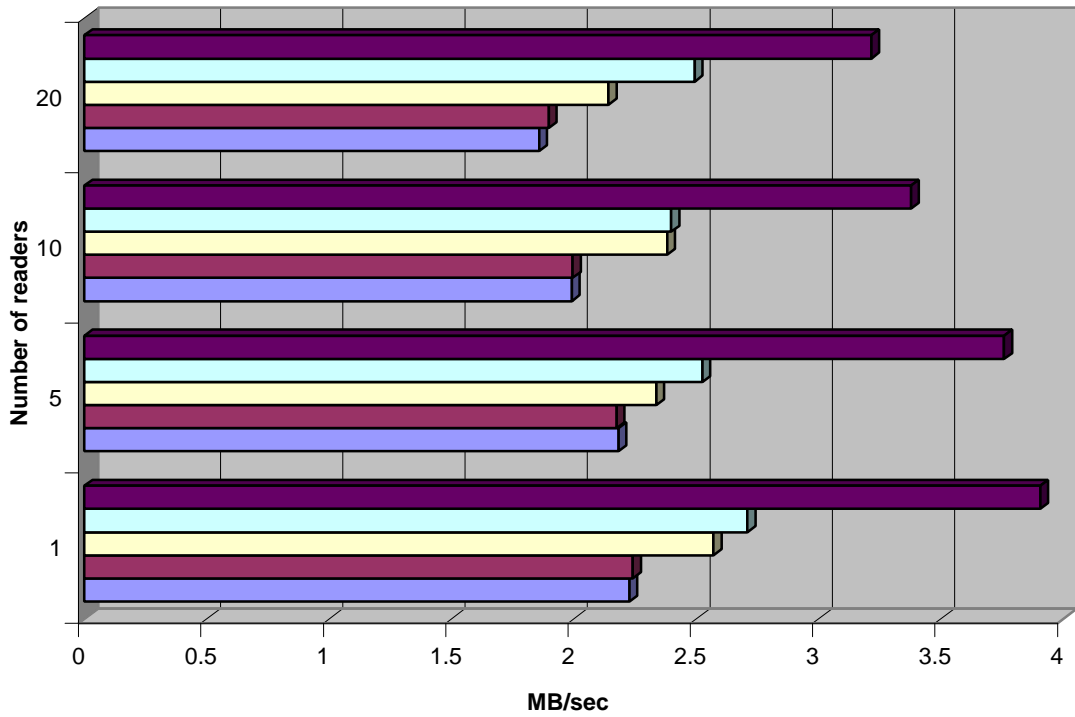


N files, N readers: Overall latency



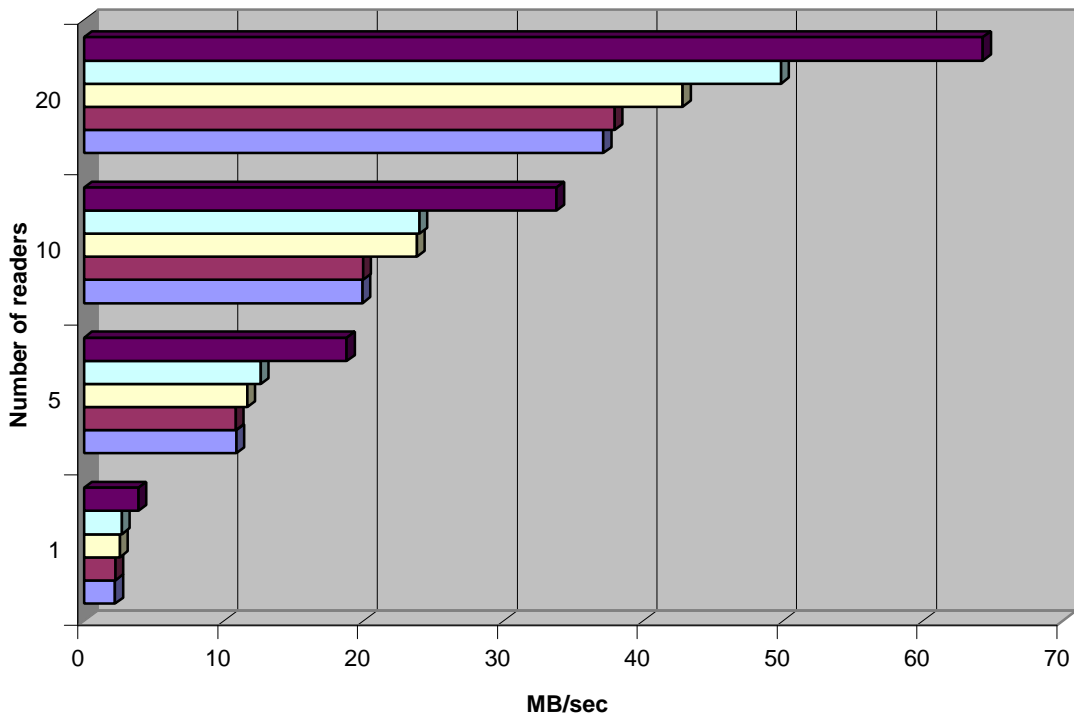
	1	5	10	20
■ Sliding window	2.56	2.66	2.96	3.11
□ IO additions	3.69	3.96	4.17	4.01
□ OpenSSL	3.89	4.28	4.2	4.67
■ Optimized	4.46	4.6	5.01	5.27
■ Unoptimized	4.49	4.58	5.02	5.38

N files, N readers: Single reader bandwidth



	1	5	10	20
■ Sliding window	3.90625	3.759398496	3.378378378	3.215434084
□ IO additions	2.7100271	2.525252525	2.398081535	2.493765586
□ OpenSSL	2.570694087	2.336448598	2.380952381	2.141327623
■ Optimized	2.242152466	2.173913043	1.996007984	1.897533207
■ Unoptimized	2.227171492	2.183406114	1.992031873	1.858736059

N files, N readers: Overall bandwidth



	1	5	10	20
■ Sliding window	3.90625	18.79699248	33.78378378	64.30868167
■ IO additions	2.7100271	12.62626263	23.98081535	49.87531172
■ OpenSSL	2.570694087	11.68224299	23.80952381	42.82655246
■ Optimized	2.242152466	10.86956522	19.96007984	37.95066414
■ Unoptimized	2.227171492	10.91703057	19.92031873	37.17472119

N files, N readers: Overall bandwidth: A different view

