Evaluating Methods of Transferring Large Datasets

Jakub Kopeć(✉)

Interdisciplinary Centre for Mathematical and Computational Modelling, University of Warsaw, ul. Tyniecka 15/17, 02-630 Warsaw, Poland
jakubkopec1018@gmail.com
https://icm.edu.pl/en/

Abstract. Our society critically depends on data, Big Data. The humanity generates and moves data volumes larger than ever before and their increase is continuously accelerating. The goal of this research is to evaluate tools used for the transfer of large volumes of data. Bulk data transfer is a complex endeavour that requires not only sufficient network infrastructure, but also appropriate software, computing power and storage resources. We report on the series of storage benchmarks conducted using recently developed elbencho tool. The tests were conducted with an objective to understand and avoid I/O bottlenecks during data transfer operation. Subsequently Ethernet and InfiniBand networks performance was compared using Ohio State University bandwidth benchmark (OSU BW) and iperf3 tool. For comparison we also tested traditional (very inefficient) Linux scp and rsync commands as well as tools designed specifically to transfer large datasets more efficiently: bbcp and MDTMFTP. Additionally the impact of using simultaneous multi-threading and Ethernet jumbo frames on transfer rate was evaluated.

Keywords: I/O · File systems · Data management · Data transfer · File transfer protocols · Network evaluation · I/O benchmarking · Elbencho

1 The Outline of the Problem

The amount of data created and processed by humanity in the last few decades has grown exponentially. "Data explosion" is the term that is commonly used to describe this phenomenon. The Internet has evolved from the military project funded in 1965 [1] to the "place" where people spend large proportion of their time. Enormous amounts of data are generated which are sent over computer networks and stored in digital form on some sort of storage.

Historically Particle Physics and Astronomy were the major generators of Big Data generated by the particle accelerators and observatory equipment spread across the globe and beyond (e.g. The Hubble Telescope). Radioastronomy, with Square Kilometre Array (SKA) project - a radio telescope with a square kilometre collecting area located in Australia and South Africa will soon generate

D. K. Panda and M. Sullivan (Eds.): SCFA 2022, LNCS 13214, pp. 102–120, 2022.
https://doi.org/10.1007/978-3-031-10419-0_7

observational data in volumes never imagined before. A prototype project: Australian Square Kilometre Array Pathfinder (ASKAP) is designed to be 1% the size of the full Square Kilometre Array and is to planned produce 60 Tb per second of raw data when finished [2]. Currently the ASKAP acquires 7,5 TB of data per second and the overall storage requirements of astronomical sciences are estimated to 1 EB per year.

However, Genomics might steal the title of the biggest data generator with the predictions of using 40 EB of storage per year by 2025. Among the factors that are boosting the amount of the data created by Genomics are: - the constant increase in number of the new high-tech sequencers in production; - increasing number of new endeavours to catalogue all the DNA of plants, animals and part of microbes; - and, of course, human Genomics - sequencing the human genome for the scientific and medical purposes. The huge data generators are the endeavours of sequencing the genomes of cancers that are characterised by unimaginable levels of genetic variations and the personalized medicine that will focus on the genome of an individual person [3].

All of these projects generate unimaginable amount of data that need to be stored and in many cases transferred over the computer network for analysis as the necessary computing power may not be available in situ. As the data transfer may be realised using various hardware and software the author decided to evaluate the performance of the chosen data transfer solutions.

The common misconception is that moving enormous amounts of data requires only appropriate network with sufficient bandwidth. For efficient bulk data transfers over a high-bandwidth network one needs powerful servers, highly performant file system and storage. Servers that are able to provide data at the rate that allows to saturate the link efficiently. It is impossible to use 100 Gbps network efficiently if the server provides data at 10 Gbps. The crucial aspect is the read/write speed of the storage device that is used in data exchange - the sending side must be able to provide (read) the data at appropriate speed and the receiving side must be able to ingest (write) incoming data. In the first place storage medium is to provide necessary data rate, but the server must also posses sufficient computing power and memory to cope with the transmission task. This hardware aspect is frequently neglected by the developers of data transfer solution and they tend to focus on the network bandwidth and software solution [4,5]. As storage device performance is critical to the data transfer endeavor we chose to start the evaluation with conducting storage benchmark with recently created elbencho tool [6] and storage sweep script that allows to gain good understanding of the storage system I/O throughput and characteristics [7].

In the era of cloud and Internet of Things (IoT) the approach of moving compute to the data, in order to reduce access costs, becomes a common practice. The example of such action is using microcontrollers located in the devices that acquire data (sensors, cameras, detectors etc.) to preprocess data stream before it is passed to the main compute system. In larger systems, e.g. in cloud applications part of the computation may be performed outside the cloud (at

the edge) and only the aggregated or abnormal results may be uploaded to the cloud for further analysis. Such actions limit the amount of the data that needs to be transferred [8]. However, there are numerous scenarios where this approach cannot be efficiently employed. Advanced scientific instruments, such as radio telescopes, particle detectors etc., are able to generate tremendous amount of raw data so they are already coupled with complex systems that reduce the data flow. But even after such reduction these are still large datasets that need to be transferred as a whole to remain meaningful and transferring them is inevitable for the backup reason alone, not to mention further sharing or processing them. Due to that the author believes that the need to move large datasets cannot be completely eradicated by the approach of moving compute to the data, thus the evaluation of existing transfer methods and finding new solutions of this problem remain relevant.

2 Tests Workbench: Hardware Specification

2.1 Servers

It is essential to treat the bulk data transfer as a complex issue that needs a holistic solution - appropriate servers interconnected with high-bandwidth network and the software that is able to efficiently utilise underlying infrastructure [4,5]. To satisfy the balanced hardware requirements we used two HPE ProLiant DL385 Gen10 Plus servers that were customised to provide sufficient storage speed and compute power. The servers' specification is listed below:

- 2x AMD EPYC 7302 16-core (3.0 GHz) processors
- 16 HPE 1×32 GB Dual Rank x8 DDR4-3200 CAS-22-22-22 RAM memory (512 GB RAM memory in total)
- 8x HPE 3.84 TB NVMe Gen4 Mainstream Performance Read Intensive SFF SC U.3 CD6 SSD drives
- 2x 240 GB SATA SSD drives with HPE E208i-a SR GEN10 12G SAS controller
- Ethernet 100 Gb 2-port QSFP28 MCX516A-CCHT Adapter
- Intel I350-T4 Ethernet 1 Gb 4-port BASE-T OCP3 Adapter for HPE
- Mellanox MT27700 Family [ConnectX-4] InfiniBand Adapter

2.2 Mellanox InfiniBand Range Extenders

In the first part of the tests (storage benchmarking) the servers were interconnected directly with their 100 Gb InfiniBand Adpters, but in the second part of the test (network benchmarking) Mellanox's MetroX MTX6240 InfiniBand extenders were used. The vendor claims that this system is able to provide 40 Gbps throughput over 40 km of dark fiber [9] allowing to benefit from IB features (hardware-implemented RDMA) between geographically distributed sites [10–12]. MetroX system implements point-to-point communication - the two MTX6240 bundles, that consist of MEX6240 IB switch and MEX6200 DWDM

transponder, are located at the ends of the link - one bundle at each end. The MEX6240 IB switch may be connected to local IB network or directly to IB adapter in the server. During the tests the bundles were interconnected locally within one rack with short fiber (since longer dark fiber was not available at the time of tests) and the MEX6240 switches were connected directly to the IB adapters of HPE ProLiant servers.

3 Storage Benchmark - Elbencho

3.1 The Motivation for Storage Benchmarking

The storage performance is one of four crucial determinants that influence the overall data transfer performance - the storage device must be able to supply as well as ingest the transferred data at appropriate rate in order to saturate the connection and efficiently use available bandwidth. Vendors advertise their product's performance with the number of input/output operations per second (IOPS) or throughput (the amount of data transferred in a given time). Often these numbers may be quite meaningless as these may be the theoretical maximal values or the results obtained in the tests that simulate only narrow use scenario or even worse - are artificially designed to obtain high results without any respect to the real-life use of the drives. The hardware performance is closely coupled with the type of the executed workload - the same drive will perform differently when it will act as database with a lot of random writes/reads of small portions of data spanned across the whole medium than when it will read serial data sequence (i.e. streaming long video clip) saved in one physical location.

In order to evaluate the storage device one should run the tests that reflect the actual workload that will be run on a given device. As a production workload is not always available or possible to run as a test, people create benchmarks whose task is to simulate various workloads that may be spotted in real-life environments. Additionally the benchmarks may provide unified way to compare given devices or computer systems - the perfect example is the LINPACK benchmark that is used to measure performance and compare the computing power of the supercomputer systems for the TOP500 list [13].

In 2020 Sven Breuner created elbencho - a distributed storage benchmark for file systems and block devices with support for GPUs. Inspired by fio [14], mdtest and ior benchmarks [15] he wanted to create new, modern, easy to use and unified tool that may be used for testing storage systems performance. Elbencho allows testing the performance of GPU storage access. It has become essential nowadays as the deep learning and AI applications operate on GPUs [6]. The storage sweep script created by Chin Fang is invaluable part of elbencho toolbox as it provides the user with one-button-push ability to discover performance characteristics of a storage service with respect to the file size. It estimates the storage throughput by writing multiple hyperscale datasets (overall size bigger than 1 TB or number of files larger than 1 million) of different characteristics - a lots of small files (LOSF), an average amount of medium-sized files or a few of big files - and presents the results in a single csv file or on a graph

created with gnuplot [7]. There are other important aspects of benchmarking storage - the most noticeable are the testbed's configuration and the file size histogram of the test dataset. The hardware specification and the benchmark execution parameters should be as close to the projected production environment as possible - if the target application is host-oriented (e.g. database) then the benchmarking tool should be run on a single client host, analogically if the target application is cluster-oriented then the benchmark should be executed concurrently on multiple client hosts. Not only the servers specification should be identical to the target ones, but the whole system should be the same (e.g. storage, interconnects). The dataset structure (file size histogram) is important as transferring or processing a lot of small files is significantly slower than operating on smaller number of larger files [7] - this issue will be described in detail in the next section. "Storage sweeps", similarly to other benchmarks, should be carried out, as mentioned before, in a configuration that is intended for the target application. Nevertheless, as data transfer methods will be evaluated instead of production application, the "sweeps" will be performed using default options that are sufficient for comparison purposes [16].

3.2 Lots of Small Files Problem (LOSF)

Lots of small files (LOSF) issue is a common concern in any processing and transfer of the data. Every file is associated with the metadata - the data that describes the actual data that is valuable for the end-user, e.g. the location in the directory tree hierarchy (in case of file-based storage), the physical location on the storage device, the creation date, the last modification date, the owner, the access permissions etc. In the case of small files the metadata size may be comparable to the size of the actual data - hence the contribution of the overhead caused by the need of processing this metadata becomes significant part of any operation on the data. When the file size increases the ratio of the metadata size to the actual data size decreases and the excessive overhead diminishes. Other issues brought by the LOSF is the fact that any operation on file requires additional operations such as accessing the file, opening or closing it after processing. When dealing with the LOSF these operations are repeated frequently between the actual data reads/writes (that are rather short as there is not much data to process), but in the case of larger files these additional actions are less frequent and are separated with the significantly longer valuable data streams. The next problem may become less important with wider use of SSD drives, but it is still worth mentioning - a traditional storage devices (HDD) perform incomparably better operating on longer sequences of data that are stored in one physical location (track or adjacent sectors) than working on a lot of small files scattered randomly over the physical storage medium.

Here are a few examples of scientific domains in which the LOSF problem emerge as the datasets are most conveniently stored as independent files [17]:

– climatology - Community Climate System Model - 450k files with an average size of 61 MB [18],

– astronomy - Sloan Digital Sky Survey - 20 million files with an average size < 1MB [19],
– genomics - sequencing the human genome - 30 million files with an average size of 190 KB [20].

There are various attempts to efficiently address the LOSF issue, e.g. in machine learning some training sets are packed into single file. In the domain of data transfer one of the solutions is to compress the files into single archive and decompress it after transmission. Nevertheless, such solution is rather imperfect as compression and decompression require CPU time.

3.3 The Reasons for Conducting "Storage Sweeps"

There are three main reasons why we conducted "storage sweeps" in advance of the test of the actual data transfer methods. First of all, the storage I/O performance may be the one of the bottlenecks in the pipeline of data transmission tasks - the "sweeps" allow verification if the throughput of the storage service is sufficient to saturate the available network bandwidth. The results will also be useful to select the best file system to conduct the further research. Secondly, these "sweeps" are the ideal way of estimating the performance of the network file systems (NFS) implemented in various network configurations. Lastly, the "sweeps" are the good start for the tests of data transfer protocols as they bring useful information on how the successive tests may be carried out, for example which datasets should be used etc.

3.4 Storage Benchmarks Results

Local File Systems. The first series of tests concerned the local file systems and the results are plotted in Fig. 1. They helped to decide that for the successive tests the XFS file system will be used, as it's performance was better in comparison with EXT4 - even though the maximal throughput was similar in both cases, with XFS the maximal value was obtained for the broader range of the datasets. The throughput of 8 SSD coupled in linux software RAID0 (using mdadm tool) was almost 8 times larger than the throughput of single SSD drive. We infer that the overhead of creating software RAID in this case was negligible. We conducted some tests of the BeeGFS distributed file system that is widely implemented on HPC systems [21]. As one may expect the BeeGFS Converged System Setup, which essentially means that all the required services (storage server, metadata server and client) are hosted within one physical server [22], cannot compete with the performance of the local XFS file system. It is caused by the overhead that results from the additional tasks required from distributed file system such as metadata synchronization or data replication. Moreover as all the services are launched on the single machine they may compete for resources which may create further overhead. In the conducted tests beeGFS gave approximately one

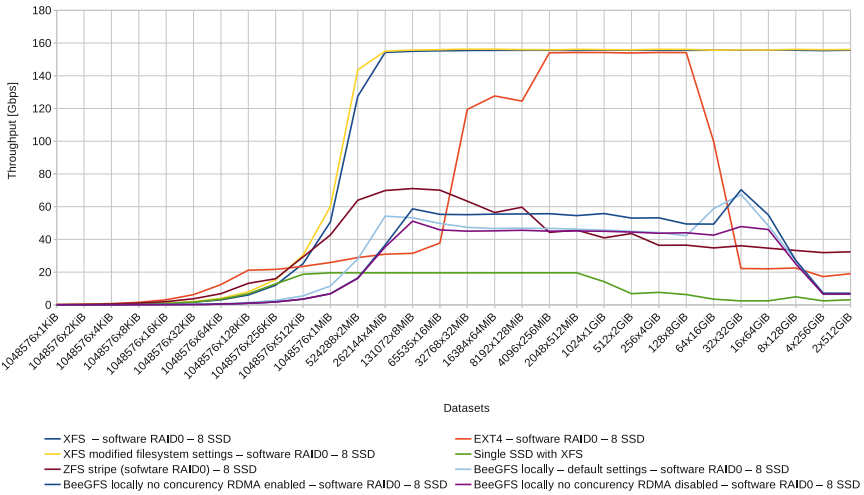

Fig. 1. Results of the "storage sweeps" conducted on local file systems

third of the throughput of XFS. Nevertheless, the distributed file systems are not designed to achieve maximal throughput per server, but to scale, i.e. to scale the performance proportionally to the large number of severs. In beeGFS documentation [23] ZFS file system was suggested as the appropriate base for beeGFS when using software RAID. The "sweeps" showed that ZSF-based software RAID's performance is far below the performance of linux software RAID. In the case of locally accessed file system the RDMA feature did not influence the throughput which is not surprising. Hence the XFS will be the best choice of file system for the further tests of data transfer protocols/data movers and that the storage service in case of medium and large files will not be the bottleneck as 155 Gbps throughput will easily saturate the 100 Gbps network adapters installed in the servers.

Network File Systems. The first "network storage sweep" was run using 1 Gbps Ethernet adapter instead of 100 Gbps one. It showed that 1 Gbps link could be utilised in 100% without any effort and tuning using default settings. Further analysis of the results plotted in Fig. 2 show that traditional NFS cannot even obtain the throughput of 10 Gbps. The results obtained using NFS over 100 Gb Ethernet and NFS over IPoIB (IP packets encapsulated into InfiniBand packets and sent over IB network) are similar. For BeeGFS "sweeps" the elbencho was run on the server that was the BeeGFS client and the second server, that was connected to the first one with 100 Gbps Ethernet, acted as the BeeGFS storage and metadata server. The BeeGFS without using the RDMA feature performed slightly better than traditional NFS, but still it barely saturated 20% of the available bandwidth. The best results were obtained using the BeeGFS with RDMA feature enabled and NFS over RDMA (NFS using RoCE feature) [24].

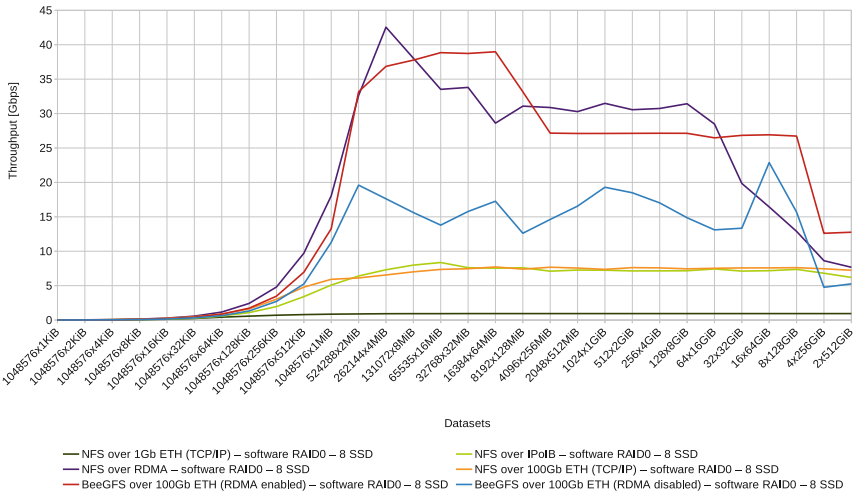

Fig. 2. Results of the "storage sweeps" conducted on network/distributed file systems

The RoCE-based NFS seems to perform slightly better, but still the results are comparable. However, any of the tested protocols/file systems did not allow to saturate so much as 50% of the link bandwidth. It is worth to notice that in all the tests the the throughput decreased for the datasets comprised of large files.

4 Network Benchmarks and Data Transfer Tests

Three types of the tests of data transmission methods were used:

- Ohio State University Bandwidth Test (OSU BW)
  The OSU BW is a part of the OSU Micro-Benchmarks bundle developed at the Ohio State University in Network-Based Computing Laboratory. This benchmark may be used to test maximum data rate that is sustained in the network, it measures the bandwidth based on the transmission time of various-sized messages passed with non-blocking MPI functions [25].
- iperf3 network speedtest tool
  Iperf3 is intended to measure the maximum achievable bandwidth on Internet Protocol-based networks and it's development is mainly driven by the the U.S. Department of Energy Sciences Network (ESNet) and the Lawrence Berkeley National Laboratory [26].
- Sending the test elbencho-created datasets between the servers
  The last series of test will consist of sending the test datasets, created using elbencho, between the servers installed in the testbed using various programs and comparing their performance.

4.1 Tests Methodology

During the tests one server acted as a host while the other one acted as a client. We decided to verify the impact of the AMD Simultaneous Multi-Threading (AMD SMT) feature on the data transfer rate - in Intel processors this feature is known as Hyper-threading and allows more efficient utilisation of CPU cycles. During the first test series the AMD SMT was disabled, and it was enabled in servers' BIOS for the second series.

We also wanted to verify how the utilisation of jumbo frames (the Ethernet frames with larger data payload) influence the transfer rates over the Ethernet. Since majority of the used applications is IP-based it was necessary to employ in such cases the Internet Protocol over InfiniBand (IPoIB) protocol that encapsulates IP packets into IB packets [27]. The use of the IPoIB is a major drawback, as it imposes additional overhead and almost completely eliminates the benefits brought by InfiniBand, but there is no other option of evaluating the performance of these applications with the MetroX IB extenders. Since the testbed operating system and IB adapter configuration did not allow the change of the IB frame size the influence of using jumbo frames with IPoIB protocol was not tested. As the out of the box server configuration usually is not able to provide full 100 Gbps throughput of the network interfaces the servers were appropriately tuned. The CPU governor was set to "performance" such as the power saving settings would not limit the CPU frequency. TCP maximal buffer size was extended to 2 GB, the maximum value possible in the Linux OS. Additionally the "fair queuing" (FQ) packet scheduler was used and the network interface interrupts were bound to the appropriate CPU socket using NIC vendor supplied script [28].

Ohio State University Bandwidth Test (OSU BW). The OSU BW benchmark is launched on two interconnected nodes simultaneously using MPI. We used MVAPICH [29] implementation of MPI to run the test. In each tested configuration the benchmark was launched three times and the average throughput of these runs is treated as an end result.

Iperf3. The tests with iperf3 measure overall network bandwidth. We also check if "zero copy" sending data method has any significant effect on the throughput. We checked if assigning the iperf3 process to appropriate CPU socket, so the affinity between the NIC and user process would be guaranteed, would impact the results. The test consisted of launching 6 instances of iperf3 simultaneously. The bandwidth of all iperf3 instances was aggregated and used as a test result. The use of single iperf3 instance was rather poor as each run in the same conditions resulted in significantly different outcomes. It is probably a result of the fact that iperf3 is single-threaded process and in the case of using high speed networks the CPU core frequency may become the bottleneck. The solution of this problem is launching multiple iperf3 instances that may utilise more than one CPU core [30]. Running 6 instances of iperf3 was optimal since adding more instances did not increase the aggregated throughput. The average of three consecutive runs is reported. This average is compared to the nominal bandwidth of the link.

Data Transfer of Elbencho-Created Datasets. This test comprised of transferring three test datasets using scp, rsync (standard Linux' tools to transfer data), bbcp [31] and MDTMFTP [32] in various testing configurations. The datasets were created using elbencho. It allows to create datasets containing random data with user-specified hierarchy (number of directories and number of files within them) and size. We created three test datasets that will correspond to three data structures:

- A lots of small files (LOSF) (1024×1 MiB = 1 GiB)
- An average number of medium files (40×256 MiB = 10 GiB)
- A few large files (10×10 GiB = 100 GiB)

In case of scp and rsync the default parameters were used. bbcp enables user to tune many transfer parameters hence we decided to check if increasing the number of parallel TCP streams to 4 or using fixed-size optimal TCP window instead of auto-tuned one would improve the throughput. The suggestion of using 4 parallel TCP streams and the formula for optimal TCP window size - (bandwidth in Gbps)/8*(round-trip time between the source and target) - were found in the bbcp documentation [31]. The MDTMFTP and OSU BW are the only solutions implemented in this study that support InfiniBand natively (without the use of IPoIB) and are able to utilise it's full potential to saturate the links effectively. Moreover the MDTMFTP is multicore software that is able to utilise multiple cores [32]. The tests with MDTM required using larger LOSF and medium datasets as their transfer took few seconds which prevented throughput measurement - the transmission was so short that the MDTMFTP process was killed by the OS before reporting any results. For MDTMFTP tests the LOSF dataset and medium dataset were increased to approximately 10 GiB (10000×1 MiB) and 30 GiB (120×256 MiB) respectively. During the tests we noticed that using jumbo frames did not impact the transfer rates significantly, but instead it caused the MDTMFTP to crash repeatedly which precluded the transfer completion. For this reason we abandoned conducting the tests in the last configuration (AMD SMT enabled, transfer using jumbo frames) as this phase of tests was too time-consuming.

4.2 Results

Ohio State University Bandwidth Test (OSU BW). The results of OSU BW benchmark are shown in the graphs in Fig. 3. The first obvious observation is the fact that the results for all combination of parameters are practically the same - as this benchmark is supposed to test the maximum data rate in the network it is not surprising that servers' configuration (AMD SMT enabled/disabled, Ethernet frame size) does not affect the obtained results. Nevertheless, these graphs are an ideal way of visualisation the benefits of using InfiniBand as a transport protocol. The maximum data rate in 100 Gbps Ethernet is approximately 55 Gbps which means that the remaining 45% of the bandwidth is used by the service of the Ethernet protocol. On the other side

one may notice that 4xQDR InfiniBand maximum data rate is approximately 30 Gbps which means that almost 94% of the available bandwidth may be used to transfer valuable data. And that is the exact reason of the InfiniBand's superiority over the Ethernet - it is able to efficiently saturate the network with meaningful data instead of congesting the fabrics with surplus control data.

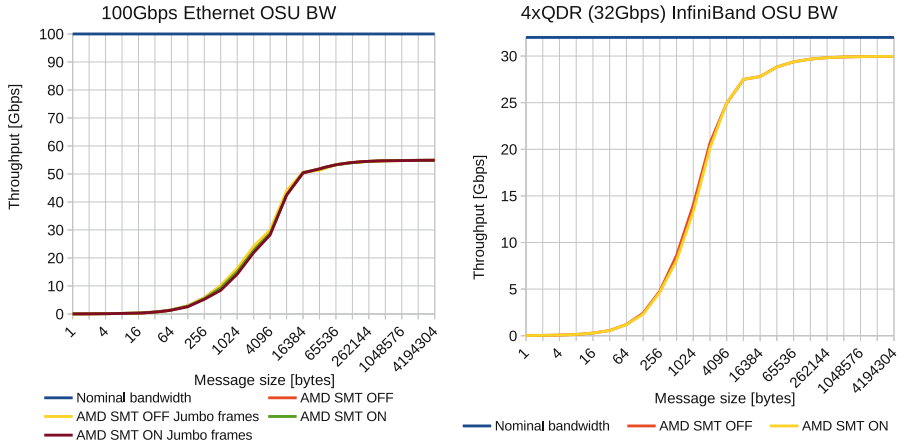

Fig. 3. Results of the OSU BW benchmarks with the additional plots of the links nominal bandwidths.

Table 1. The results of iperf3 benchmarks with AMD SMT disabled and with standard-sized Ethernet frames. All results are given in Gbps

	100 Gbps ethernet					4xQDR (32 Gbps) InfiniBand				
	Run 1	Run 2	Run 3	Average	% of 100 Gbps	Run 1	Run 2	Run 3	Average	% of 32 Gbps
No additional parameters	35,08	39,18	42,04	38,77	38,8%	14,18	14,18	14,19	14,18	44,3%
Zero copy	42,63	49,13	41,34	44,37	44,4%	11,84	11,84	11,85	11,84	37,0%
CPU affinity set	42,07	35,31	40,52	39,30	39,3%	9,52	9,52	11,90	10,31	32,2%

Iperf3. The results of iperf3 benchmarks are shown in Tables 1, 2, 3 and 4. By comparing the Tables 1 and 2 we notice that setting the CPU affinity does not seem to influence the throughput while using zero copy method of sending data boost up the throughput over a few Gbps. Turning on AMD SMT caused a drop in obtained throughputs. Such phenomenon is not seen in the results of the test with jumbo frames as in all cases the aggregated throughput achieve approximately the maximum network data rate of 55 Gbps.

In the iperf3 InfiniBand tests we do not find any significant correlations except for the fact that the achieved throughputs are a combinations of 2.38 Gbps

Table 2. The results of iperf3 benchmarks with AMD SMT enabled and with standard-sized ethernet frames. All results are given in Gbps.

	100 Gbps ethernet					4xQDR (32 Gbps) InfiniBand				
	Run 1	Run 2	Run 3	Average	% of 100 Gbps	Run 1	Run 2	Run 3	Average	% of 32 Gbps
No additional parameters	32,38	34,26	31,44	32,69	32,7%	14,22	9,52	11,90	11,88	37,1%
Zero copy	41,40	38,86	38,21	39,49	39,5%	14,23	14,10	14,23	14,19	44,3%
CPU affinity set	34,04	33,56	33,38	33,66	33,7%	9,52	11,90	14,25	11,89	37,2%

Table 3. The results of iperf3 benchmarks with AMD SMT disabled and with jumbo frames. All results are given in Gbps.

	100 Gbps ethernet				
	Run 1	Run 2	Run 3	Average	% of 100 Gbps
No additional parameters	54,50	54,87	54,81	54,73	54,7%
Zero copy	54,12	54,31	54,14	54,19	54,2%
CPU affinity set	55,46	55,34	55,50	55,43	55,4%

Table 4. The results of iperf3 benchmarks with AMD SMT enabled and with jumbo frames. All results are given in Gbps.

	100 Gbps ethernet				
	Run 1	Run 2	Run 3	Average	% of 100 Gbps
No additional parameters	54,54	54,86	54,96	54,79	54,8%
Zero copy	54,12	54,92	53,87	54,30	54,3%
CPU affinity set	55,42	55,25	55,20	55,29	55,3%

Table 5. The results of data transfer tests with AMD SMT disabled and with standard-sized Ethernet frames. All results are given in Gbps.

	100 Gbps ethernet				4xQDR (32 Gbps) InfiniBand			
	LOSF	Medium	Large	% of 100 Gbps	LOSF	Medium	Large	% of 32 Gbps
scp	0,95	1,11	1,56	1,6%	0,92	1,36	1,71	5,3%
rsync	1,12	1,55	1,42	1,5%	1,12	1,58	1,44	4,9%
bbcp	2,30	13,60	16,80	16,8%	1,92	7,91	8,80	27,5%
Bbcp - optimal window size	1,96	13,60	16,00	16,0%	1,51	7,88	8,80	27,5%
bbcp - 4 streams	2,30	12,80	16,00	16,0%	1,88	7,92	8,80	27,5%
MDTMFTP	28,23	27,87	28,41	28,4%	17,08	20,19	25,60	80,0%

and 1.19 Gbps (a half of 2.38) obtained by the individual iperf3 instances. It may suggest that IPoIB imposes some limit on encapsulated frames that causes the repetitiveness of per-thread results. There is no evident impact of setting he CPU core affinity between the NIC and user process as the obtained throughput was similar to the outcomes of the tests conducted with default iperf3 parameters. Possibly the obtained throughput was too small to benefit or bring loss from the CPU affinity settings.

Table 6. The results of data transfer tests with AMD SMT enabled and with standard-sized Ethernet frames. All results are given in Gbps.

	100 Gbps ethernet				4xQDR (32 Gbps) InfiniBand			
	LOSF	Medium	Large	% of 100 Gbps	LOSF	Medium	Large	% of 32 Gbps
scp	1,11	1,34	1,47	1,5%	0,89	1,44	1,40	4,5%
rsync	1,12	1,52	1,61	1,6%	1,12	1,74	1,58	5,4%
bbcp	2,19	12,80	18,40	18,4%	1,91	7,91	8,80	27,5%
bbcp - optimal window size	1,93	13,60	15,20	15,2%	1,48	7,89	8,80	27,5%
bbcp - 4 streams	2,24	13,60	16,00	16,0%	1,85	7,92	8,80	27,5%
MDTMFTP	27,40	29,95	30,65	30,6%	16,71	18,99	27,96	87,4%

Table 7. The results of data transfer tests with AMD SMT disabled and with jumbo frames. All results are given in Gbps.

	100 Gbps ethernet			
	LOSF	Medium	Large	% of 100 Gbps
scp	0,95	1,18	1,63	1,6%
rsync	1,12	1,40	1,69	1,7%
bbcp	2,21	12,80	16,00	16,0%
bbcp - optimal window size	2,10	12,80	16,00	16,0%
bbcp - 4 streams	0,22	12,80	16,00	16,0%
MDTMFTP	—	—	—	—

Data Transfer Tests. The results of the data transfer tests are listed in Tables 5, 6 and 7 and the first conspicuous fact is that how poorly the standard Linux data transfer tools (scp, rsync) utilise available bandwidth - in all configurations they were not able to provide as much as 2 Gbps throughput. While using the 100 Gb Ethernet they used approximately 1,6% of the bandwidth. In the case of the InfiniBand they used approximately 5% of the bandwidth, but that fact is meaningless as it does not result from the increase in the achieved throughput, but from the decrease of the available bandwidth. Regardless of the testbed configuration the results obtained by scp and rsync were similar and oscillated around 1,34 Gbps. This poor performance of these tools is probably caused by the fact that these tools use OpenSSH with built-in 1 MB buffer to encrypt the transferred data [33]. In order to remove that bottleneck one should look for tool that uses another encryption protocol or change the "data mover" to one that encrypt only control channel and sends the actual data unencrypted (which is acceptable in some applications) - for example bbcp [34]. The result obtained with bbcp shows that auto-tuned parameters are optimal as any attempt of manual tuning caused the slight decrease of the throughput or did not bring any positive effect. The bbcp results show perfectly the issue of processing the LOSF as the throughput obtained when transferring large files was approximately 8–9 times bigger than in the case of the LOSF. In all test conducted with bbcp we noticed that there seems to be a limit on the maximal throughput that may be obtained using this program - 16 Gbps on the Ethernet and 8,8 Gbps on the InfiniBand. We believe that this limitation may arise from the fact that bbcp is single-thread program and CPU frequency is the factor that limits

the throughput. However, in the case of InfiniBand this limit may be a result of the IPoIB encapsulation. The most interesting are the results obtained with MTDMFTP - it's mechanism of dealing with the LOSF problem proved to be successful as the differences between the LOSF and large files throughput were not only significantly smaller, but on the Ethernet it seemed to disappear completely. While transferring the large files using IB the MDTMFTP was able to saturate approximately 80–90% of the available bandwidth that was impossible to achieve with any other tested software. The change of the size of the Ethernet frame did not result in any major change of the achievable throughput, but it only caused the instability of MDTMFTP software - the numerous errors prevented obtaining any reliable results of MDTMFTP performance with the use of the jumbo frames. Enabling AMD SMT feature revealed slight improvement of the throughput obtained with the MDTMFTP, no other changes were noticed.

4.3 Additional Comments on the Tests Results

We were not able to notice any significant impact of enabling AMD SMT feature (except for small decrease of throughput in the iperf3 tests and slight improvement in MDTMFTP transfer rates) that would allow drawing unambiguous conclusions on its influence on transfer rates. The usage of the jumbo frames undoubtedly improved the obtained throughput (what was observable in iperf3 tests), but none of the evaluated "data movers" can benefit from that increase as single-threaded applications were not able to utilise such bandwidth and MDTMFTP became unstable and the jumbo frames caused numerous errors and crashes.

5 Conclusions

The tests revealed striking inefficiency of the most popular Linux transfer tools on high-bandwidth networks. These tools were developed when the volume of transferred data and network bandwidths were incomparably smaller, thus they are not able to perform efficiently with the current volumes of transmitted data. Their design and underlying protocols are not able to saturate modern high-speed network links - approximately 98% of the bandwidth was wasted. This software may still perform well in the situations it was designed for. Scp is a useful tool to transfer few gigabytes over 100Mbps residential network, but it is by far not sufficient and outright wasteful to transfer hundreds of terabytes of scientific data across intercontinental 100Gbps link. The test has also shown how much bandwidth capacity may be spared by using the InfiniBand fabrics instead of the Ethernet. The InfiniBand is able to utilise efficiently more than 90% of the bandwidth while the Ethernet barely uses half of it after thorough tuning and effort. But the most importantly this research allows to understand how complex an issue the efficient transfer of digital data is and, that the network bandwidth is only one part of the mix and to transfer data efficiently one needs also appropriate storage, file system and computing resources, not to mention the suitable software.

5.1 Future Work

This research by no means did exhaust a topic of bulk data moving as it is very broad and complex problem that exists as the computer networks evolve and the number of links and their bandwidth increase rapidly. Moreover the storage technology is advancing rapidly as the new media are being developed. This study was focused on evaluation of common hardware and widely available software performance, however, there are several areas where the further research may be conducted.

New Storage Media. As the new non-volatile memory technologies emerge, such as spin-torque transfer RAM (STT-RAM), phase-change (PCM) and resistive (ReRAM) memory [36,37] or Intel's 3D XPoint [37,38] their performance could be assessed. For instance, the new Intel Optane SSD drives that employ 3D XPoint memory technology could be benchmarked using elbencho or its influence on the data transfer rate may be evaluated.

NVM-adapted File Systems. With the advent of fast non-volatile memories with the DRAM-like performance and byte-addressability current file systems are becoming the new performance bottleneck. Modern journaling file systems are designed to use whole data block as the basic unit of the journal what may cause significant overhead. As the actual price of the next genaration NVM prohibits it to be used as standalone large-capacity memory device it may be beneficial to use it in hybrid DRAM/NVMM (non-volatile main memory) or NVDIMM solutions where NVMM may act as the external journal device for journaling files systems [37,39,40] or the journaling strategies may be altered to be more suitable for NVM dvices [41]. On the other hand maybe the new principles of designing file systems are required as in [42,43] to fully utilize the cutting-edge NVM media. The evaluation of mentioned solutions may be the further extension of this research.

Alleviate OpenSSH Bottleneck Using HPN-SSH. The poor performance of SSH-backed tools caused by the limited size of OpenSSH buffer and the significant encryption overhead may be improved by using HPN-SSH - a research project that consists of a series of modifications to OpenSSH created at the Pittsburg Supercomputing Center. Authors of the modifications reported that the throughput of SSH was at least doubled while using their tuning [44]. The data transfer tests of SSH-backed tools could be repeated with the use of HPN-SSH for the comparison purposes.

Creating Geographically Distributed Testbed. Developing further geographically distributed systems enabled with the MetroX InfiniBand or Vcinity range extenders [35] and comparing its performance with Ethernet would allow for validation of obtained results in productional environment.

Evaluating Other Software Solutions' Performance. During this research only the widely available software was evaluated. In the upcoming studies the performance of other software, for instance XRootD [45] may be evaluated. If possible the performance of proprietary solutions, such as Zettar's zx [46] or Obsydian Strategics' dsync+ [10] could be investigated.

Acknowledgments. The work reported here constituted research part of my Master of Science degree in Computational Engineering undertaken at the Interdisciplinary Centre for Mathematical and Computational Modelling,

University of Warsaw (ICM UW) under supervision of Dr Marek Michalewicz who recommended this topic and guided me throughout. I would like to express my gratitude to people who contributed to the realisation of this project: Chin Fang (Zettar Inc.) and Marcin Semeniuk (ICM UW) for sharing their knowledge with me and their valuable comments on my work; and to Bartosz Drogosiewicz (ICM UW) and Jarosław Skomiał (ICM UW) for helping me build the testbed for this project. I also wish to thank Hewlett Packard Enterprise Polska Sp. z o.o. for providing two test HPE ProLiant DL385 Gen10 Plus servers used in this study.

References

1. Kleinrock, L.: An early history of the internet [History of Communications]. IEEE Commun. Mag. **48**(8), 26–36 (2010)
2. Newman, R., Tseng, J.: Memo 134 Cloud Computing and the Square Kilometre Array (2011)
3. Stephens, Z., et al.: Big data: astronomical or genomical? PLOS Biol. **13**(7), 1–11 (2015)
4. Fang, C.: Moving massive amounts of data across any distance efficiently, Talk on 2020 Rice Oil & Gas HPC Conference. https://www.youtube.com/watch?v=8PCjMSKMyRw. Accessed 17 Apr 2021
5. Zettar Inc. white paper: Understanding moving data at scale & speed (2020). https://www.zettar.com/white-paper/. Accessed 01 Mar 2021
6. Breuner, S.: elbencho github repository. https://github.com/breuner/elbencho/. Accessed 29 May 2021
7. Fang, C.: Storage Sweep github repository. https://github.com/breuner/elbencho/tree/master/contrib/storage_sweep. Accessed 29 May 2021
8. Near-Memory Computing. https://semiengineering.com/knowledge_centers/compute-architectures/near-memory-computing/. Accessed 02 Dec 2021
9. Mellanox TX6240 product brief. https://www.mellanox.com/related-docs/prod_long_haul_systems/MetroX_TX6240.pdf. Accessed 29 Mar 2021
10. Michalewicz M., et al.: InfiniCortex: concurrent supercomputing across the globe utilising trans-continental infiniband and galaxy of supercomputers. In: Supercomputing 2014: The International Conference for High Performance Computing, Networking, Storage and Analysis, New Orleans (2014). https://doi.org/10.13140/2.1.3267.7444
11. Michalewicz, M.: InfiniCortex: present and future (2020). https://doi.org/10.1145/2903150.2912887
12. Niedzielewski, K., et al.: Long distance geographically distributed InfiniBand based computing (2020). https://doi.org/10.14529/jsfi200202

13. About The Linpack Benchmark webpage. https://www.top500.org/project/linpack/. Accessed 29 May 2021
14. Flexible I/O Tester (fio) github repository. https://github.com/axboe/fio. Accessed 29 May 2021
15. IOR and mdtest github repository. https://github.com/hpc/ior. Accessed 29 May 2021
16. Fang, C., Cottrell, R., Kissel, E.: When to use rsync - DOE Technical Report. https://slac.stanford.edu/pubs/slactns/tn06/slac-tn-21-001.pdf. Accessed 29 May 2021
17. Carns P., Lang S., Ross R., Vilayannur M., Kunkel J., Ludwig T.: Small-file access in parallel file systems. In: 2009 IEEE International Symposium on Parallel & Distributed Processing, pp. 1–11. IEEE, Rome (2009). https://doi.org/10.1109/IPDPS.2009.5161029
18. Chervenak A., et al.: Monitoring the Earth System Grid with MDS4. In: Proceedings of the Second IEEE International Conference on e-Science and Grid Computing, p. 69. IEEE, Amsterdam (2006). https://doi.org/10.1109/E-SCIENCE.2006.261153
19. Neilsen, E.H., Jr.: The Sloan digital sky survey data archive server. Comput. Sci. Eng. **10**(1), 13–17 (2008)
20. Bonfield, J.K., Staden, R.: ZTR: a new format for DNA sequence trace data. Bioinformatics **18**(1), 3–10 (2002)
21. Why Use BeeGFS webpage. https://www.beegfs.io/c/home/why-use-beegfs/. Accessed 29 May 2021
22. BeeGFS Documentation, Architecture overview. https://doc.beegfs.io/latest/architecture/overview.html. Accessed 29 May 2021
23. Tips and Recommendations for BeeGFS Storage Server Tuning webpage. https://www.beegfs.io/wiki/StorageServerTuning. Accessed 29 May 2021
24. HowTo Configure NFS over RDMA (RoCE) webpage. https://community.mellanox.com/s/article/howto-configure-nfs-over-rdma-roce-x. Accessed 07 Apr 2021
25. OSU Micro-Benchmarks webpage. http://mvapich.cse.ohio-state.edu/benchmarks/. Accessed 09 Jun 2021
26. iperf3 homepage. https://iperf.fr/. Accessed 09 Jun 2021
27. Kashyap, V.: RFC 4392 - IP over InfiniBand (IPoIB) Architecture. https://www.rfc-editor.org/rfc/rfc4392.html. Accessed 09 Jun 2021
28. ESnet Fasterdata Knowledge Base - 40G/100G Tuning webpage. https://fasterdata.es.net/host-tuning/linux/100g-tuning/. Accessed 09 Jun 2021
29. MVAPICH homepage. http://mvapich.cse.ohio-state.edu/. Accessed 09 Jun 2021
30. ESnet Fasterdata Knowledge Base - iperf3 at 40Gbps and above webpage. https://fasterdata.es.net/performance-testing/network-troubleshooting-tools/iperf/multi-stream-iperf3/. Accessed 09 Jun 2021
31. BBCP homepage. https://www.slac.stanford.edu/~abh/bbcp/. Accessed 16 Mar 2021
32. MDTM Homepage. https://mdtm.fnal.gov/index.html. Accessed 17 Mar 2021
33. ESnet Fasterdata Knowledge Base - scp and sftp webpage. https://fasterdata.es.net/data-transfer-tools/scp-and-sftp/. Accessed 09 Jun 2021
34. ESnet Fasterdata Knowledge Base - Data transfer tools background webpage. https://fasterdata.es.net/data-transfer-tools/background/. Accessed 09 Jun 2021
35. WAN Interoperability Overview - Vcinity Application Note. https://vcinity.io/sites/default/files/WAN_Interop_AN_RevA.pdf. Accessed 29 Mar 2021

36. Suzuki, K., Swanson, S.: The Non-Volatile Memory Technology Database (NVMDB). Technical Report CS2015-1011, Department of Computer Science & Engineering, University of California, San Diego (2015)
37. Xu J., Swanson, S.: NOVA: a log-structured file system for hybrid volatile/Non-volatile main memories. In: 14th USENIX Conference on File and Storage Technologies (FAST 16), pp. 323–338. USENIX Association, Santa Clara (2016)
38. Intel and Micron Produce Breakthrough Memory Technology. https://newsroom.intel.com/news-releases/intel-and-micron-produce-breakthrough-memory-technology/. Accessed 16 Feb 2022
39. Chen, C., Yang, J., Wei, Q., Wang, C., Xue, M.: Fine-grained metadata journaling on NVM. In: 32nd International Conference on Massive Storage Systems and Technology (MSST 2016), pp. 1–13 (2016). https://doi.org/10.1109/MSST.2016.7897077
40. Xu J., et al.: NOVA-Fortis: a fault-tolerant non-volatile main memory file system. In: Proceedings of the 26th Symposium on Operating Systems Principles (SOSP 2017), pp. 478–496. Association for Computing Machinery, New York (2017). https://doi.org/10.1145/3132747.3132761
41. Chen, C., Wei, Q., Wong, W.-F., Wang, C.: NV-journaling: locality-aware journaling using byte-addressable non-volatile memory. IEEE Trans. Comput. **69**(2), 288–299 (2020). https://doi.org/10.1109/TC.2019.2948004
42. Kwon, Y., Fingler, H., Hunt, T., Peter, S., Witchel, E., Anderson, T.: Strata: a cross media file system. In: Proceedings of the 26th Symposium on Operating Systems Principles (SOSP 2017), pp. 460–477. Association for Computing Machinery, New York (2017). https://doi.org/10.1145/3132747.3132770
43. Lu, Y., Shu, J., Chen, Y., Li, T.: Octopus: an RDMA-enabled distributed persistent memory file system. In: 2017 USENIX Annual Technical Conference (USENIX ATC 17), pp. 773–785. USENIX Association, Santa Clara (2017)
44. Rapier, C., Bennett, B.: High speed bulk data transfer using the SSH protocol. In: Proceedings of the 15th ACM Mardi Gras conference (MG 2008), Association for Computing Machinery, New York (2008). https://doi.org/10.1145/1341811.1341824
45. XRootD Homepage. https://xrootd.slac.stanford.edu/index.html. Accessed 16 Mar 2021
46. A Software Engine for Moving Data at Scale & Speed - Zettar product brief. https://www.zettar.com/wp-content/uploads/2020/10/Zettar_product_brief.pdf. Accessed 01 Mar 2021