electronics

Article

Performance Evaluation of NVMe-over-TCP Using Journaling File Systems in International WAN

Se-young Yu †

International Center for Advanced Internet Research, Northwestern University, Evanston, IL 60611, USA; young.yu@northwestern.edu
† Current address: 750 Lakeshore Drive, Chicago, IL 60611 USA.

Abstract: Distributing Big Data for science is pushing the capabilities of networks and computing systems. However, the fundamental concept of copying data from one machine to another has not been challenged in collaborative science. As recent storage system development uses modern fabrics to provide faster remote data access with lower overhead, traditional data movement using Data Transfer Nodes must cope with the paradigm shift from a store-and-forward model to streaming data with direct storage access over the networks. This study evaluates NVMe-over-TCP (NVMe-TCP) in a long-distance network using different file systems and configurations to characterize remote NVMe file system access performance in MAN and WAN data moving scenarios. We found that NVMe-TCP is more suitable for remote data read than remote data write over the networks, and using RAID0 can significantly improve performance in a long-distance network. Additionally, a fine-tuning file system can improve remote write performance in DTNs with a long-distance network.

Keywords: Data Transfer Nodes; high-performance networks; remote data access

Citation: Yu, S.-y. Performance Evaluation of NVMe-over-TCP Using Journaling File Systems in International WAN. *Electronics* **2021**, *10*, 2486. https://doi.org/10.3390/electronics10202486

Academic Editor: Antonio F. Díaz

Received: 31 August 2021
Accepted: 7 October 2021
Published: 13 October 2021

1. Introduction

Big data movement plays a critical role in science collaboration. Data-intensive science requires large data to be shared among researchers in different sites. As collaborations between researchers and institutions become more accessible using online collaboration tools, the amount of data shared among them often becomes too large. As a result, it becomes difficult to share the research data using traditional hardware and software.

A big data movement distributes a large amount of data collected from the research, often sourced at large instrumentation, and Data Transfer Nodes (DTNs) allow high-performance data distribution. To operate DTNs and keep up with scientific research, we need to continuously improve big data movement services and adopt new technologies that enhance data transfer performance. Recent developments in storage technology promise to access data remotely with efficient mechanisms.

Science DMZ [1] developed a high-speed data transfer network architecture with DTNs to exchange data efficiently at the boundaries of local network perimeters. Science DMZ allows a special network policy for large data flow between data sharing sites with DTNs operating at high speed, isolated from commodity network activities from the campus and lab networks.

Large-scale data transmission between DTNs requires specialized protocols, software, and hardware to move data over long-distance high-speed networks efficiently. Unfortunately, traditional network fabrics and systems are not optimized for large data flows. DTNs equipped with high-speed data transfer protocols, efficient transfer applications, and network interfaces allow DTNs to copy data from one site to another.

Many data transfer applications use traditional server-client architecture with various optimization techniques [2–5] to optimize DTNs for high-performance data copy because remote data access is often unreliable and difficult to optimize [6] especially for long-distance high-performance networks. However, the recent development of NVMe and NVMe-over-Fabrics (NVMeoF) provides remote data access at high-speed with lower overhead.

NVMeoF defined protocols encapsulating NVMe commands in network segments to exchange NVMe commands natively over network fabrics with low processing overhead and access latency. Since the encapsulation is protocol-independent, it is implemented in three major networks: RDMA, Fiber connection, and TCP. Unlike others, NVMe-over-TCP (NVMe-TCP) can be used widely over existing DTNs connecting Metropolitan Area Networks (MANs) and Wide Area Networks (WANs) as TCP is widely implemented and more robust to packet loss on the network.

NVMeoF exposes remote NVMe storage devices to allow remote storage access using network fabrics with low overhead. However, to access traditional files in remote NVMe storage, users have to use a file system to control data storage and mount to the remote file system. Modern journaling file systems manage system metadata to isolated journal space inside the file system, which incurs extra I/O on the storage device. Yet, the performance of these journaling file systems is not studied well with NVMeoF over a long-distance network with long latency.

This study defines research problems in advancing data transfer technologies as follows.

Problem 1: How can we integrate NVMe-TCP with file systems used in existing data transfer workflow? NVMe-TCP provides ways to access remote storage over the networks; however, there is a disparity of information on analyzing the performance of NVMe-TCP on remote data read and write over existing file systems.

Problem 2: How much impact the modern journaling file systems have on NVMe-TCP and remote data access? File systems such as Ext4 [7] and XFS [8] are journaling file systems that require frequent metadata access. Will there be any disadvantages in accessing data using the file system compared to the raw block device without journaling in the long-distance network, which added data access latency?

Problem 3: How can we optimize those file systems for NVMe-TCP in long-distance data transfer? NVMe-TCP provides multiple optimization mechanisms to provide efficient remote data access, but the effect of such mechanisms on different data transfer workflow is not studied.

Previous studies were characterizing the performance of NVMeoF in different aspects [9–11]; however, these studies are limited to studying NVMeoF in LAN. Furthermore, there is no previous study on characterizing the performance of NVMe-TCP in MANs and WANs. This work complements these works by understanding the performance of NVMe-TCP in a long-distance network and evaluating it using standard local file systems.

This study aims to reveal performance characteristics of NVMe-TCP in a long-distance network, focusing on DTNs' data transfer workflow. DTN uses a local file system to access and transfer data to remote places. We will look at how NVMe-TCP performs with different file systems commonly used in DTN and improve the data transfer process. We will also look at optimization options for NVMe-TCP with these file systems using different workload types and inline data sizes.

The main contribution of this study is three-fold. First, we provide a comprehensive understanding of NVMe-TCP operating on modern journaling file systems in a long-distance network. Second, we identify possible performance bottlenecks in the file systems used in DTNs with NVMe-TCP for a long-distance network. Lastly, we identify file system optimization techniques in NVMe-TCP over a long-distance network.

The rest of the paper is organized as follows: Section 2 provides background information on NVMeoF, DTNs, and related technology to optimize data movement. Section 2.4 discusses related works in NVMeoF and DTN. Section 3 discusses network topology and system setup for evaluating modern file systems with NVMeoF in DTNs connected through MANs and WANs. Section 4 describes the result of the comparative analysis of NVMe-TCP in a long-distance network. Section 5 discusses the implication of the findings in the previous sections. Finally, Section 6 concludes the paper with the characterization of NVMe-TCP in a long-distance network and discusses the future work.

2. Background

2.1. Data Transfer Nodes

DTNs are specialized systems for high-performance data movement. DTNs are equipped with a high-performance network interface, local storage system, and processors to enhance network data transfer. Modern DTNs often have 100 Gbps or faster connection speed in the perimeters of campus or lab network. In addition, they are configured with a specific network policy that allows high-performance data exchange.

DTNs are a key component in data sharing and collaborative science. DTNs are widely deployed in science facilities such as Pacific Research Platform (PRP) (https://pacificresearchplatform.org/ (accessed on 6 October 2021)) and MREN Research Platform (MRP) (http://mren.org/ (accessed on 6 October 2021)).

Data transfer applications [2,3,5] use DTNs resources to move data from one storage system to another. Many of these applications use traditional server-client architecture in a distributed manner. A central controller manages file copies from a DTN to another. At the same time, they try various optimization techniques such as parallel file transfer and pipelining to improve end-to-end file transfer performance.

However, the recent development of NVMe-TCP allows direct remote storage access over TCP. This may enable a fundamental change in how DTNs operate in the future, as one can access the remote data without copying it to the local storage without performance degradation. Previous remote data access techniques such as iSCSI have much higher overhead compared to NVMeoF [9].

2.2. NVMe-over-TCP

NVMe is a storage protocol developed for high-performance SSDs to provide access to modern computer systems. With direct communication between the SSD and I/O stack through the PCI-express interface, NVMe provides higher I/O performance in high-performance data transfer systems such as DTNs. In addition, using multiple NVMe devices, DTNs can scale their storage performance to match with high-speed network capacity growing beyond 100 Gbps which was not possible with the traditional hard disk.

DTNs and data movement systems have evolved to provide better performance and usability in distributing and sharing data locally and remotely over the network, including long distances over WANs. NVMeoF is a recent development in storage systems that defines a common architecture supporting a range of storage networking fabrics for NVMe block storage protocol over a storage networking fabric [12]. NVMeoF allows remote NVMe devices to be exported and directly accessed over a network without using a server-client application, making it suitable for streaming data from the remote storage system and DTNs. Streaming data becomes more common in DTN applications [13,14].

NVMeoF currently supports three network fabrics: RDMA, Fiber channel, and TCP. However, RDMA and Fiber channel are inadequate to be used in high-speed WANs. RDMA implementations (RoCE and iWARP) suffer from a significant performance degradation with smaller packet losses than TCP. On the other hand, the Fiber channel does not scale as Ethernet does in terms of network capacity in WANs.

NVMe-TCP is a subset of NVMeoF, which encapsulating NVMe command to networking fabric. Traditional NVMes are attached to a PCI-express slot in the mainboard, only allowing the local system to access the storage. However, NVMeoF can disaggregate the NVMe storage from the local system and allowing the remote system to access the NVMe without translating NVMe protocols. This provides faster data access for DTNs with high-speed network connections with optimized TCP/IP stacks.

However, accessing remote storage will have longer latency when used in a long-distance network. It is impossible to reduce propagation delay due to the physical distance in data communication. It is commonly known that NVMe-TCP in intercontinental WANs can induce 200 ms propagation delay instead of 100 to 1000 ns scale latency in local PCI-express connection. In addition, modern file systems used in DTNs have journaling to ensure data reliability when the file system becomes unavailable without proper unmount.

However, journaling requires additional I/O to the storage device before actual modification of data in the storage, which may affect the performance of remote file access using NVMe-TCP in a long-distance network.

NVMe-TCP can use multiple submission and completion queues to issue parallel NVMe commands and configurable inline data sizes to buffer larger data sizes for improving performance. In the following sections, we will discover how these mechanisms affect the performance of NVMe-TCP in a long-distance network.

2.3. Local File Systems

DTNs use local file systems to store shared data from other DTNs and share their data from the file system. Typically, a large dataset is stored in a dedicated storage system and moved to DTN's local file system for transfer.

Modern local file systems used in DTNs use journaling features to maintain the integrity of stored data. Journaling keeps metadata of files stored in the file system to track structure changes, thus requiring additional data access to the storage device when a file is modified.

Data integrity in the remote file system is essential as the remote connection can be interrupted by link failure, congestion, and policy changes. Although the journaling on some file systems can be separated from data volume, writing journals of metadata on other storage than the remote NVMe can cause problems when the remote NVMes are shared among many NVMe-TCP initiators.

XFS [8] maintains its metadata using the b+tree algorithm [15] for efficient inode lookup and allocation [16]. It is focused on optimizing concurrent I/O operation in the file system and is suitable for handling a large number of files.

The Ext4 [7] file system allocates a region for journals within the file system and uses this space to keep track of file system changes (and optionally store actual data when $data = journal$ mount option is used). The file system metadata is first written to the reserved space before actual file modification and then updated after the file I/O finishes. Thus, writing actual data to the reserved space provides additional integrity protection. Yet, it has large performance degradation as it must read and write the actual data one more time within the file system.

It is common to use software RAID using mdadm [17] to improve performance and storage space for NVMe devices that support striping over multiple NVMe block devices [18]. There are other file systems such as BTRFS [19] and ZFS [20] that support Copy-on-Write and striping. However, these file systems are not actively deployed in DTNs due to their heavy resource utilization. Additionally, file systems for containerized application such as OverlayFS [21] and AuFS [22] are also avoided for lack of scalability.

2.4. Related Works

Data transfer protocols and applications have been studied in many existing works. Science DMZ [1] proposed architecture for high-speed data transfer, and DTN plays a critical role in sharing data from one site to another. As network capacity increases, many techniques are developed to accelerate the throughput of data transfer in DTNs. Legrand et al. [23] developed a TCP-based transfer system for data transfer with multiple I/O thread optimization. Allcock et al. [24] extended FTP protocol with multiple TCP connections, pipelining [25], and concurrent I/O processes [26]. Zhang et al. [3] optimized DTNs with a Non-Uniform Memory Access (NUMA) node by binding transfer processes to closer NUMA nodes.

Dorigo et al. [5] developed a hierarchical data distribution system for scalable data access. Foster et al. [4] developed an online service for distributing data between data endpoints using GridFTP. Yu et al. [18] developed an optimization framework for DTNs and evaluated NUMA optimization for traditional data transfer and NVMeoF. Lu et al. [27] developed a platform for distributing data between sites using mdtmFTP with NUMA optimization. These works proposed the optimization in traditional data transfer using server-client architecture and are not suitable for direct data access that NVMe-TCP provides. However, some optimization techniques can be used in NVMe-TCP, such as using concurrent I/O process, yet its performance in NVMeoF is not studied.

On the other hand, NVMeoF is studied mainly in Local Area Networks (LANs). Guz et al. [9] evaluated NVMeoF performance and compared it with locally attached NVMe using synthetic and KV-store workload. The study is further extended [11] by assessing the overhead of using SPDK compared to the Linux kernel implementation. Xu et al. [10] evaluated the performance of overlay file systems with NVMeoF. However, these works are limited to using NVMeoF with RDMA in local area networks.

This study complements the above works by evaluating the file system performance of NVMe-TCP in the long-distance network for DTN with MAN and WAN. Since RDMA does not scale well in the long-distance network due to inefficient congestion control, NVMe-TCP is suitable for the lossy long-distance network. Therefore, this work will study NVMe-TCP performance in our MAN and WAN using remote data read and write and explore possible optimization options for NVMe-TCP.

3. Methodology

This section explains the experiment setup and evaluation methodology. The experiment used StarLight International/National Communications Exchange Facility (http://www.startap.net/starlight/ (accessed on 6 October 2021)) to connect to two remote DTNs. These DTNs are used to evaluate the performance of NVMe-TCP with journaling file systems in a long-distance network.

3.1. Network Topology

Figure 1 shows the experiment setup for evaluating NVMe-TCP in MAN and WAN. First, three DTNs are connected to a 100 Gbps network through StarLight International/National Communications Exchange Facility (http://www.startap.net/starlight/ (accessed on 6 October 2021)). StarLight provides 100 Gbps connections between Northwestern University (NVMe-TCP target) and the University of Illinois at Chicago (NVMe-TCP initiator) through the Metropolitan research and education network Research Platform (MRP) and Ciena in Canada (NVMe-TCP initiator) through Ciena Experimental Network for Innovation (CENI). The RTT between DTN in Northwestern University and the University of Illinois at Chicago is 0.15 ms, and DTNs between Northwestern University and Ciena is 15.0 ms. Both connections are routed through StarLight Metropolitan Research and Education Network (MREN) (http://mren.org/ (accessed on 6 October 2021)).

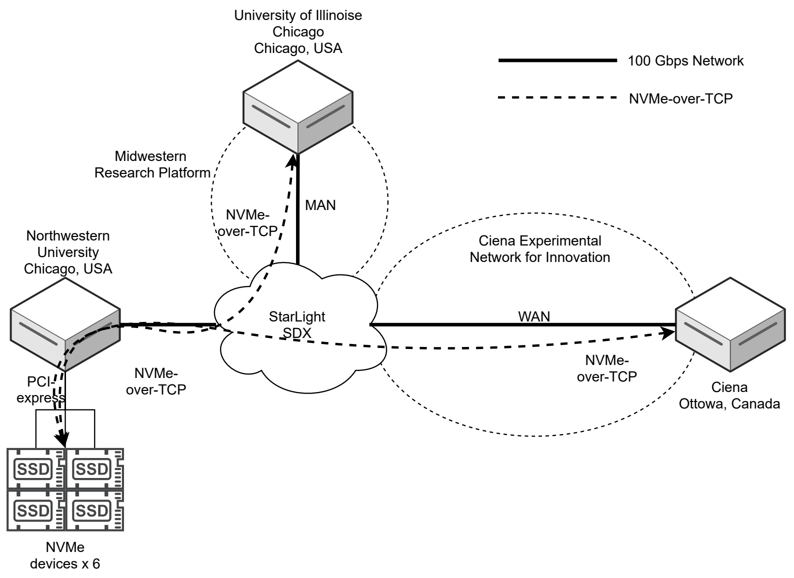


Figure 1. Experimental setup.

Each DTN is equipped with 100 GbE NIC, and each DTN is optimized to perform 100 Gbps memory-to-memory transfer end-to-end. Table 1 shows the configuration of DTNs used in this experiment. In addition, we measured the network capacity between DTNs using multiple TCP connections, and both MREN and CENI connections are capable of transferring more than 95 Gbps. NVMe devices are connected through the PCI-express link, and they can read and write 25.6 Gbps each sequentially using Flexible I/O tester [28].

Table 1. DTN configuration.

DTN	Northwestern University	University of Illinois at Chicago (UIC)	Ciena
CPU	2 x Intel(R) Xeon(R) Gold 5217 @ 3.00 GHz	2 x Intel(R) Xeon(R) E5-2690 0 @ 2.90 GHz	2 x Intel(R) Xeon(R) Gold 6136 @ 3.00 GHz
Memory	DDR4-2666 MT/s 192 GB	DDR3-1600 MT/s 64 GB	DDR4-2666 MT/s 192 GB
NIC		Mellanox MT27800 Family [ConnectX-5]	
NVME	6 x Micron 9300 7.68 TB	-	-
OS	GNU/Linux 5.4.0	GNU/Linux 5.13.9	GNU/Linux 5.4.0
RTT	-	0.46 ms	15.0 ms

There are two major NVMeoF implementations publicly available, one from Linux kernel and another from Storage Performance Development Kit (SPDK) (https://spdk.io/ (accessed on 6 October 2021)). However, in a preliminary experiment with both implementations in this setup, the SPDK implementation has significant performance degradation over increasing Round-Trip Time (RTT) and cannot produce meaningful throughput when RTT between DTNs is longer than 5 ms. Therefore, this study only uses the Linux kernel implementation in the evaluation.

We used the default CUBIC as our TCP congestion control algorithm. We were able to saturate available network capacity using memory-to-memory TCP transfer. We achieved more than 90 Gbps on StarLight-UIC DTNs and StarLight-Ciena DTN using CUBIC TCP with iperf3.

Due to the time it takes to complete each transfer and the availability of the remote DTN, we had to repeat the experiment five times and calculate the average throughput and latency in each experiment scenario.

3.2. Evaluation Scenarios

There are optimization techniques available in NVMeoF with journaling file systems to improve performance in DTNs with a high-speed, long-distance network. For example, increasing inline data size from the default of 8 k to 16 k reduces overhead in encapsulating data in NVMe command and response capsules. Another technique is to use software RAID0 to improve the data rate using multiple NVMe devices. Lastly, use a file system with less journal access to reduce I/O overhead.

This study will explore the effectiveness of these optimizations in NVMe-TCP to demystify their benefit at a longer distance. To evaluate these, we set up an NVMe-TCP target on the DTN at Northwestern University and use the DTN at the University of Illinois at Chicago.

First, baseline NVMe performance in four different workload types commonly seen on DTNs is measured, and the performance of the optimizations in two different DTNs is investigated. The workload type A-D represents a typical data transfer scenario where users read and write 1, 10, 100, and 1000 files concurrently in a single remote NVMe device and RAID0 device with striping data over six NVMe devices. Next, Ext4 and XFS performance over the workloads are compared against a raw device that reads and writes to the NVMe block device directly without a file system and journaling. RAID0 with six remote NVMe devices is also compared to see how NVMe-TCP workloads are handled by data striping over multiple NVMes. This allows us to observe how each file system handles system calls created by multiple file I/O simultaneously and data striping, and NVMe-TCP handles such system calls. We used XFS for RAID0 because it is most commonly used in DTNs and supports automatic strip size and width.

At the end of each experiment, the files at both NVMes are deleted, and NVMe TRIM is executed. We waited 2 min for the TRIM command to finish and started the following experiment.

This study will provide optimization recommendations for NVMe-TCP when used in DTNs with long-distance WAN through this evaluation. Both throughput and latency of each workload using different file systems are measured, and this study will discuss the implications of their performance in high-speed data transfer.

4. Result

4.1. Baseline Performance

This experiment measures the throughput, submission latency, and completion latency of NVMe-TCP in MAN and WAN testbeds.

Figure 2 shows the read and write throughput of NVMe-TCP with different file systems using UIC DTN as an initiator. We can see there is no significant difference between the raw device, ext4, and XFS in both read and write throughput, except RAID0. RAID0 does provide significantly higher read throughput with workloads with multiple files as striping data over multiple remote NVMe devices. Yet, it becomes unstable and loses connection with too many files reads at once. Thus, it does not produce meaningful data transfer for workload D in using RAID0.

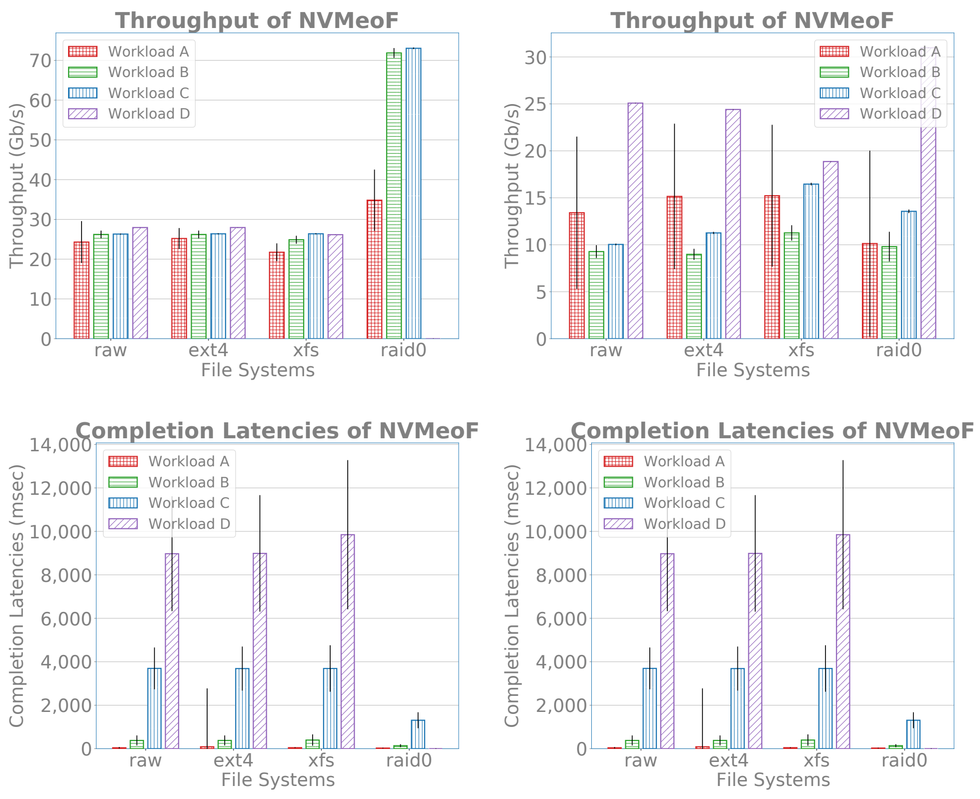
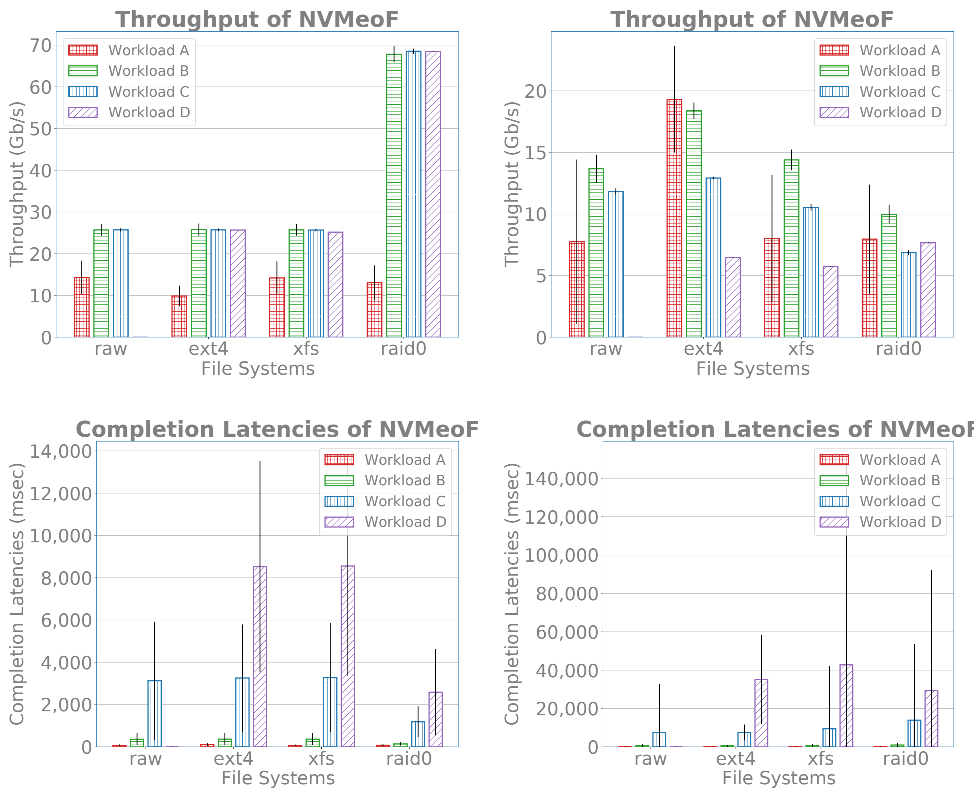


Figure 2. Baseline average and standard deviation of read (**a**) and write (**b**) performance for UIC DTN.

Compared to the read throughput, write throughput with smaller concurrent file writes are significantly slower even with RAID0, suggesting that we need multiple file I/O to get the optimal throughput with NVMe-TCP. In addition, the completion latency for remote write is higher than the remote read, which results in a lower overall remote write throughput.

Interestingly, the write throughput for workload D is higher than workload B and C, although the completion latency or workload D is higher. The lower throughput may be a result of having more concurrent I/O overcomes the increase in latency. This suggests NVMe-TCP may be well suited for streaming data from remote NVMe without remote write.

Figure 3 shows the read and write throughput of NVMe-TCP with different file systems using Ciena DTN as an initiator. As RTT increase, the throughput of single file I/O decreases significantly in both read and write, especially reading from Ext4 and writing to XFS. Using multiple file I/O improves read throughput on all the file systems, but not for remote write. NVMe remote write has significantly higher latency with an increasing number of files than the UIC DTN and caused the NVMe submission queues to fill up faster. However, workload D could not be completed using raw NVMe device in remote read and write because it made the NVMe-TCP connections unstable. Additionally, as workload increases and the submission queues fill up, it increases completion latencies high for I/O in all file systems, as manifested as large variabilities in latencies in Figures 3–5.



(**a**) (**b**)

Figure 3. Baseline average and standard deviation of read (**a**) and write (**b**) performance for Ciena DTN.

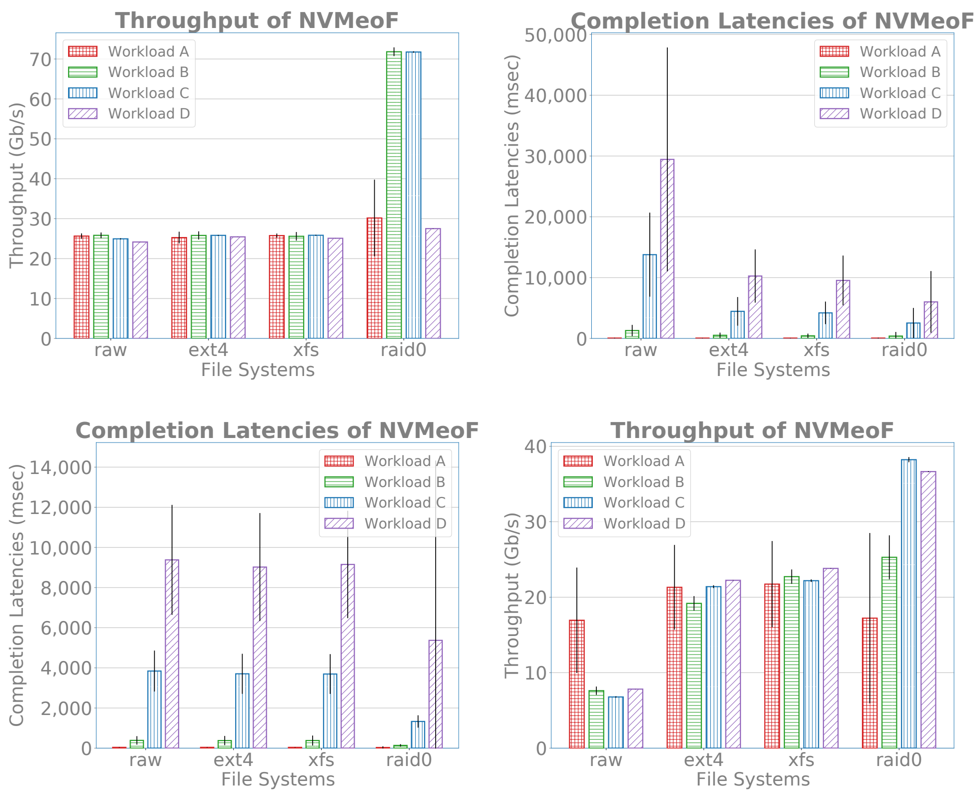
Compared to Ext4, XFS shows significantly low remote write performance with Ciena DTN. This indicates that XFS may not suit applications that write few files simultaneously to remote NVMe devices using the default configuration.

In Ciena DTN, the write throughput of workload D is much lower than the other workloads in most file systems. Furthermore, with the completion latency increased almost twice as much compared to the UIC DTN, the benefit of multiple file I/O cannot overcome the increased completion latency and degraded throughput.

4.2. Larger Inline Data Size

In this section, the inline data size of the NVMe-TCP target is increased to 16 KB from the default of 8 KB, and read and write performance is measured.

Figures 4 and 5 show the read and write performance of NVMe-TCP in UIC and Ciena DTNs using a 16 KB inline data size. The read performance was not affected by increasing inline data size in both throughput and completion latency. It improved the performance of remote write throughput and reduced completion latency for all file systems and RAID0. However, it degrades performances of multiple file writes on raw block devices on UIC DTN, especially for the workload D.



(**a**) (**b**)

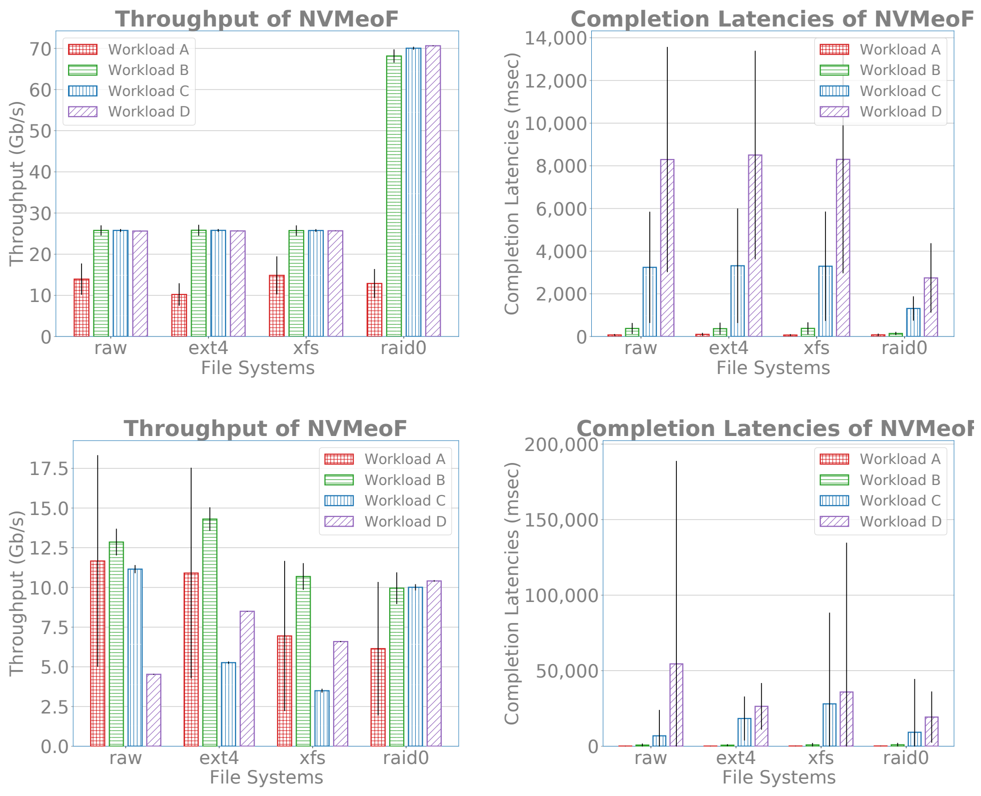
Figure 4. Average and standard deviation of read (**a**) and write (**b**) performance for UIC DTN using 16 KB inline data size.

On the other hand, increasing inline data size to 16 KB had an adverse effect on DTNs with longer RTT. It increased the variation of remote write throughput in all file systems in Ciena DTN and widened the standard deviation of write throughput and completion latency.

The result shows that increasing inline data size may benefit remote write for DTNs with relatively shorter RTT but not for DTNs with longer RTT.

4.3. Large Journal Size

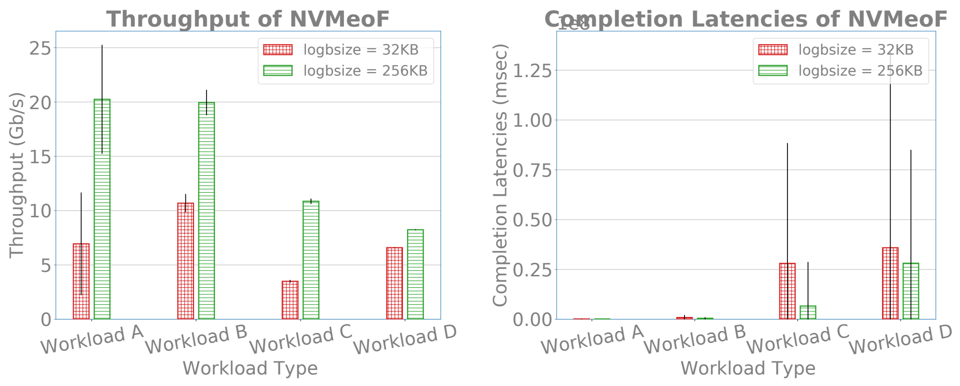
To improve how XFS handles journaling in remote NVMe-TCP devices, the journal log size is increased to 256 KB from the default 32 KB in Ciena DTN. Increasing journal log size increases in-memory log buffer size, reducing the number of journal writes in the XFS file system.



(**a**) (**b**)

Figure 5. Average and standard deviation of read (**a**) and write (**b**) performance for Ciena DTN using 16 KB inline data size.

In Figure 6, we can see a significant improvement of throughput and completion latency in Ciena DTN. A larger log buffer size benefits workloads with fewer concurrent files, which implies minor improvement when the NVMe queues are filled more by file I/O. Yet, it shows it is possible to resolve the performance issue of XFS with a small number of concurrent remote writes by fine-tuning file system options. However, there was no significant performance improvement from other file system mount options or format options from XFS, Ext4, and mdadm.



(**a**) (**b**)

Figure 6. Average and standard deviation of throughput (a) and completion latency (b) of write performance in XFS for Ciena DTN.

5. Discussion

This study revealed performance comparisons between different DTNs using different file system settings to find optimal configurations for NVMe-TCP with modern file systems used in DTN. NVMe-TCP provides low-overhead direct access to remote data without server-client application architecture. Yet, the performance of remote data access over existing file systems in DTNs with a long-distance network has not been studied well.

Our research problems require a comprehensive understanding of NVMe-TCP performance with journaling file systems to optimize efficient data transfer in DTNs connected to MAN and WAN. Therefore, this paper set up experiments to answer the research questions derived from NVMe-TCP with a journaling file system in a long-distance network. Furthermore, this study aims to find optimal ways to use a local file system with NVMe-TCP for existing DTN workflows in a long-distance network.

The experimental result explains how NVMe-TCP can be used with existing local file systems in DTN with the long-distance network. In addition, NVMe-TCP provides significantly faster remote data read than data write in both DTNs, regardless of the file system. This suggests that the journaling files system can be used with NVMe-TCP for streaming data from remote DTN to local DTN using networks with long RTTs using remote data reads.

There are minimal performance differences between different file systems typically used in DTN and raw block devices. Using RAID0 provides significant performance improvements for both remote read and write, especially for concurrent file access, as NVMe-TCP works well with data striping. However, accessing too many files make the NVMe-TCP connection unstable.

By optimizing file system options, NVMe-TCP performance can be improved further. For example, increasing log buffer size in XFS provide significant remote write performance in the long-distance network.

The results indicate that journaling file systems provide insignificant overhead with NVMe-TCP even with long-distance WAN. As a typical data transfer scenario using NVMe-TCP uses journaling file systems to access remote data, the result ensures the current journaling file systems can be used with NVMe-TCP. Furthermore, metadata operations that support RAID0 can also lead to better performance in NVMe-TCP with long-distance WAN.

A journaling file system over NVMe-TCP allows DTNs to transfer remote data without server-client applications without additional metadata optimization. In addition, users can mount the remote journaling system file system directly without copying the data to the local file system before they need to access it, which significantly reduces remote data access delay. NVMe-TCP with direct file system mounts also improves the user experience as they can access remote data in the same way they do in local data.

6. Conclusions and Future Works

Big data movement plays a critical role in science collaboration. Data-intensive science requires large data to be shared among researchers in different sites. The recent development of NVMe and NVMeoF provides remote data access at high speed with lower overhead without server-client architecture. However, to our best knowledge, there is no existing study in characterizing the performance of NVMe-TCP with journaling file systems in DTNs with MAN and WAN connections.

This study revealed how we could use existing file systems with NVMe-TCP for data transfer using DTNs in a long-distance network. We found NVMe-TCP is well-suited for remote data streaming with journaling file systems and relatively slower remote write performance. We also found a slight performance difference between file systems used in DTN with NVMe-TCP, but we can improve overall performance with RAID0. In addition, we can further optimize some file system performance by fine-tuning the journaling options.

We can recommend using NVMe-TCP for remote data read and streaming, with RAID0 for optimal performance through this knowledge. We can do this by configuring the NVMe-TCP target at the DTNs near the data source and configuring the DTN initiator near the data sink. There is almost no difference between raw block device, Ext4, and XFS in DTNs in such cases. XFS has lower remote write performance with NVMe-TCP when few files are written at a time, but the increasing size of the journal log can help increase NVMe-TCP write performance.

In future works of NVMeoF, we will include various RAID levels to improve data integrity in DTN with remote data access and address unstable connection issues with workloads with many files. Various RAID levels (other than RAID0) can be used to provide data integrity over multiple storage devices as they provide redundancy over unstable NVMe-TCP connections between DTNs with long-distance networks.

Funding: This work was supported by the National Science Foundation (NSF) under grant no. 1450871.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: http://github.com/youf3/NVMeoF_fs_performance (accessed on 6 October 2021).

Acknowledgments: We would like to thank our sponsors, including the StarLight International/National Communications Exchange Facility, Metropolitan Research and Education Network (MREN), National Science Foundation (NSF), Northwestern University, Dell Technologies Inc., CANARIE, Ciena Inc., and, more specifically, Rodney Wilson, who has been a very valuable supporter and contributor. In addition, we would like to thank Jim Chen and Joe Mambretti for their support for this research.

Conflicts of Interest: The author declares no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DTN	Data Transfer Nodes
NVMe	Non-Volatile Memory Express
NVMeoF	NVMe-over-Fabrics
NVMe-TCP	NVMe-over-TCP
MAN	Metropolitan Area Network
WAN	Wide Area Network
LAN	Local Area Network
RDMA	Remote Direct Memory Access
iWARP	internet Wide Area RDMA Protocol
RoCE	RDMA over Converged Ethernet
RAID	Redundant Array of Inexpensive Disks
PRP	Pacific Research Platform
MREN	Midwest Research and Education Network
MRP	MREN Research Platform

References

1. Dart, E.; Rotman, L.; Tierney, B.; Hester, M.; Zurawski, J. The Science DMZ: A network design pattern for data-intensive science. In Proceedings of the 2013 SC-International Conference for High Performance Computing, Networking, Storage and Analysis (SC), Denver, CO, USA, 17–22 November 2013; pp. 1–10.
2. Globus. Globus Connect Server v5.3 Installation Guide. Globus, 2019. Available online: https://docs.globus.org/globus-connect-server/v5.3/ (accessed on 6 October 2021)
3. Zhang, L.; Wu, W.; DeMar, P.; Pouyoul, E. mdtmFTP and Its Evaluation on ESNET SDN Testbed. *Future Gener. Comput. Syst.* **2018**, *79*, 199–204. [CrossRef]
4. Foster, I. Globus Online: Accelerating and Democratizing Science through Cloud-Based Services. *IEEE Internet Comput.* **2011**, *15*, 70–73. [CrossRef]
5. Dorigo, A.; Elmer, P.; Furano, F.; Hanushevsky, A. XROOTD/TXNetFile: A Highly Scalable Architecture for Data Access in the ROOT Environment. In Proceedings of the 4th WSEAS International Conference on Telecommunications and Informatics, TELE-INFO'05, Prague, Czech Republic, 13–15 February 2005; World Scientific and Engineering Academy and Society (WSEAS): Stevens Point, WI, USA; pp. 46:1–46:6.

6. Aiken, S.; Grunwald, D.; Pleszkun, A.; Willeke, J. A performance analysis of the iSCSI protocol. In Proceedings of the 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies, (MSST 2003), San Diego, CA, USA, 7–10 April 2003; pp. 123–134. [CrossRef]
7. Cao, M.; Bhattacharya, S.; Ts'o, T. Ext4: The Next Generation of Ext2/3 Filesystem. In Proceedings of the Linux Storage and Filesystem Workshop, San Jose, CA, USA, 12–13 February 2007.
8. Holton, M.; Das, R. XFS: A Next Generation Journalled 64-Bit Filesystem with Guaranteed Rate I. SGI Corp. 1994. Available online: http://www.sgi.com/Technology/xfs-whitepaper.html (accessed on 6 October 2021).
9. Guz, Z.; Li, H.H.; Shayesteh, A.; Balakrishnan, V. NVMe-over-fabrics Performance Characterization and the Path to Low-overhead Flash Disaggregation. In Proceedings of the 10th ACM International Systems and Storage Conference (SYSTOR '17), Haifa, Israel, 22–24 May 2017.
10. Xu, Q.; Awasthi, M.; Malladi, K.T.; Bhimani, J.; Yang, J.; Annavaram, M. Performance analysis of containerized applications on local and remote storage. *Proc. MSST* **2017**, *3*, 24–28.
11. Guz, Z.; Li, H.H.; Shayesteh, A.; Balakrishnan, V. Performance Characterization of NVMe-over-Fabrics Storage Disaggregation. *ACM Trans. Storage* **2018**, *14*, 1–18. [CrossRef]
12. Minturn, D. NVM Express Over Fabrics. In Proceedings of the 11th Annual OpenFabrics International OFS Developers' Workshop, Monterey, CA, USA, 15 March 2015.
13. Kettimuthu, R. Autonomous Infrastructure for Science: Challenges and Opportunities, Argonne National Lab, 29 November 2018. Available online: https://www.mcs.anl.gov/ kettimut/talks/2018-11-29-BDSDE-Keynote.pdf (accessed on 6 October 2021)
14. Habib, S. *Streaming Data in Cosmology*; Streaming Systems, 2016. Available online: http://streamingsystems.org/Slides/streaming_data_cosmology_habib_stream_2016.pdf (accessed on 6 October 2021)
15. Rodeh, O. B-Trees, Shadowing, and Clones. *ACM Trans. Storage* **2008**, *3*, 1–26. [CrossRef]
16. Hellwig, C. XFS: The big storage file system for Linux. *Login Mag. Usenix Sage* **2009**, *34*, 10–18.
17. Vadala, D. *Managing RAID on Linux: Fast, Scalable, Reliable Data Storage*, 1st ed.; O'Reilly Media: Sebastopol, CA, USA, 2002
18. Yu, S.; Chen, J.; Yeh, F.; Mambretti, J.; Wang, X.; Giannakou, A.; Pouyoul, E.; Lyonnais, M. SCinet DTN-as-a-Service Framework. In Proceedings of the 2019 IEEE/ACM Innovating the Network for Data-Intensive Science (INDIS), Denver, CO, USA, 17 November 2019; pp. 1–8.
19. Rodeh, O.; Bacik, J.; Mason, C. BTRFS: The Linux B-Tree Filesystem. *Trans. Storage* **2013**, *9*, 1–32. [CrossRef]
20. Bonwick, J.; Ahrens, M.; Henson, V.; Maybee, M.; Shellenbaum, M. The zettabyte file system. In Proceedings of the 2nd Usenix Conference on File and Storage Technologies, San Francisco, CA, USA, March 31–April 2 2003.
21. Mizusawa, N.; Nakazima, K.; Yamaguchi, S. Performance Evaluation of File Operations on OverlayFS. In Proceedings of the 2017 Fifth International Symposium on Computing and Networking (CANDAR), Aomori, Japan, 19–22 November 2017; pp. 597–599. [CrossRef]
22. Merkel, D. Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux J.* **2014**, *2014*, 2.
23. Legrand, I.; Newman, H.; Voicu, R.; Cirstoiu, C.; Grigoras, C.; Dobre, C.; Muraru, A.; Costan, A.; Dediu, M.; Stratan, C. MonALISA: An agent based, dynamic service system to monitor, control and optimize distributed systems. *Comput. Phys. Commun.* **2009**, *180*, 2472–2498. [CrossRef]
24. Allcock, W. GridFTP: Protocol Extensions to FTP for the Grid, GGF Document Series DFG. 20. 2003. Available online: http://www.ggf.org/documents/GFD.20.pdf (accessed on 6 October 2021).
25. Bresnahan, J.; Link, M.; Kettimuthu, R.; Fraser, D.; Foster, I. Gridftp pipelining. In Proceedings of the TeraGrid Conference, Madison, WI, USA, 17 October 2007.
26. Yildirim, E.; Kim, J.; Kosar, T. How GridFTP Pipelining, Parallelism and Concurrency Work: A Guide for Optimizing Large Dataset Transfers. In Proceedings of the 2012 SC Companion: High Performance Computing, Networking Storage and Analysis, Salt Lake City, UT, USA, 10–16 November 2012; pp. 506–515. [CrossRef]
27. Lu, Q.; Zhang, L.; Sasidharan, S.; Wu, W.; DeMar, P.; Guok, C.; Macauley, J.; Monga, I.; Yu, S.; Chen, J.H.; et al. BigData Express: Toward Schedulable, Predictable, and High-Performance Data Transfer. In Proceedings of the 2018 IEEE/ACM Innovating the Network for Data-Intensive Science (INDIS), Dallas, TX, USA, 11 November 2018; pp. 75–84. [CrossRef]
28. Axboe, J. Fio-Flexible io Tester. 2014. Available online: http://freecode.com/projects/fio (accessed on 6 October 2021).