When NVMe over Fabrics Meets Arm: Performance and Implications

Yichen Jia[*] Eric Anger[†] Feng Chen[*]

[*]Louisiana State University [†]Arm Inc

MSST'19

May 23th , 2019

Table of Content

Background

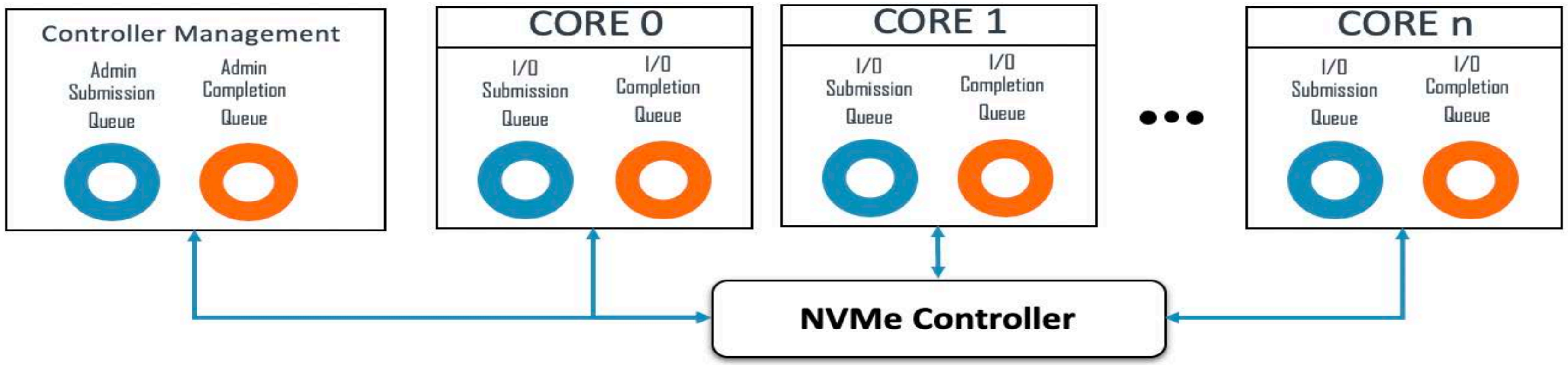
Arm, NVMe and NVMe over Fabrics

Background : Arm Processors

- Arm processors have become dominant in IoT and mobile phones, etc
- The recently released 64-bit ARM CPUs are suitable for cloud and data centers
  - Arm-based instances have been available in Amazon AWS since Nov, 2018
- One of its important applications is to be the storage server
  - Enhanced computing capability and power efficiency

Background : NVM Express

- Flash-based SSD is becoming cheaper and more popular
  - High throughput and low latency
  - Suitable for parallel I/Os
- Non-Volatile Memory Express (NVMe)
  - Supporting deep and paired queues
  - Scalable for the next generation NVM



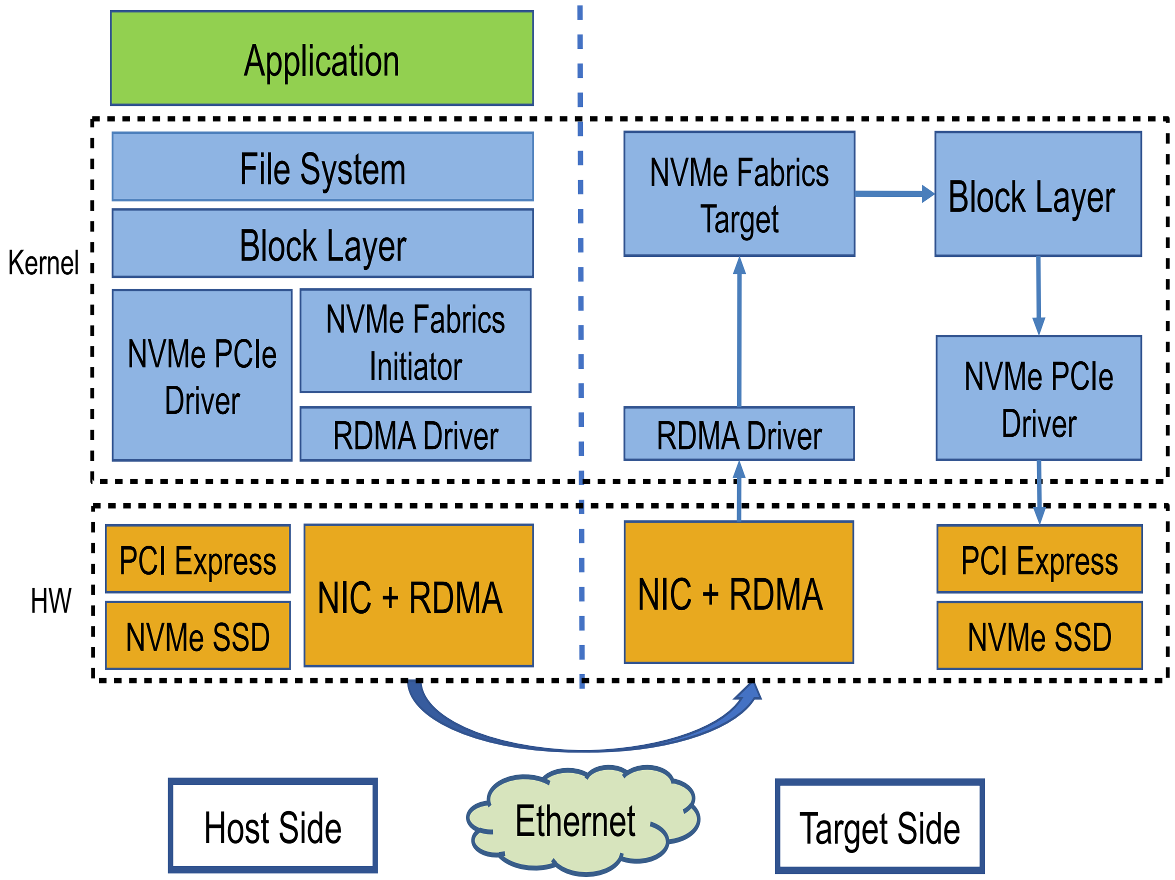
NVMe Structure*

*https://nvmexpress.org/about/nvm-express-overview/

Background: NVMe-over-Fabrics

- Direct Attached Storage (DAS)
  - Computing and storage in one box
  - Less flexible, hard to scale, etc

- Storage Disaggregation
  - Separated computing and storage
  - Reduced total cost of ownership (TCO)
  - Improved hardware utilization
  - Examples: NVMe over Fabrics, iSCSI

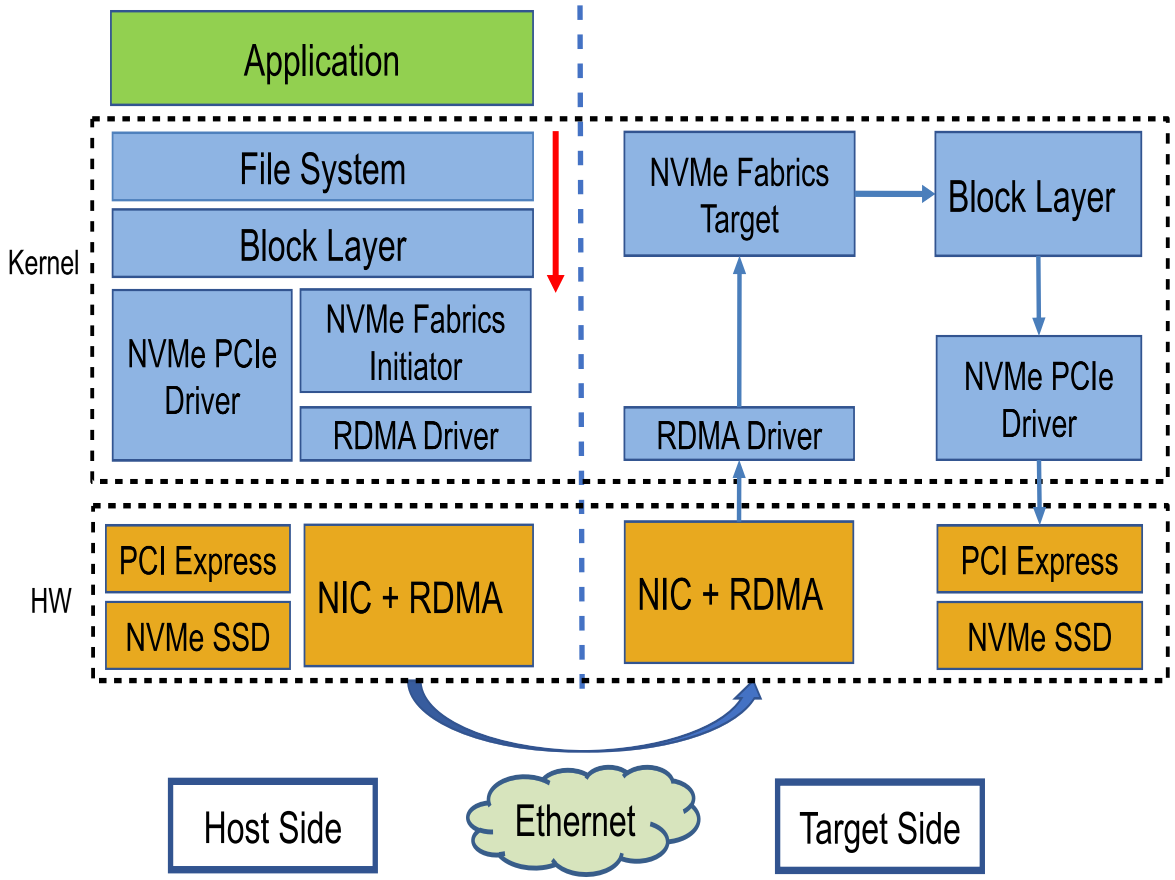


NVMe over Fabrics

Background: NVMe-over-Fabrics

- Direct Attached Storage (DAS)
  - Computing and storage in one box
  - Less flexible, hard to scale, etc

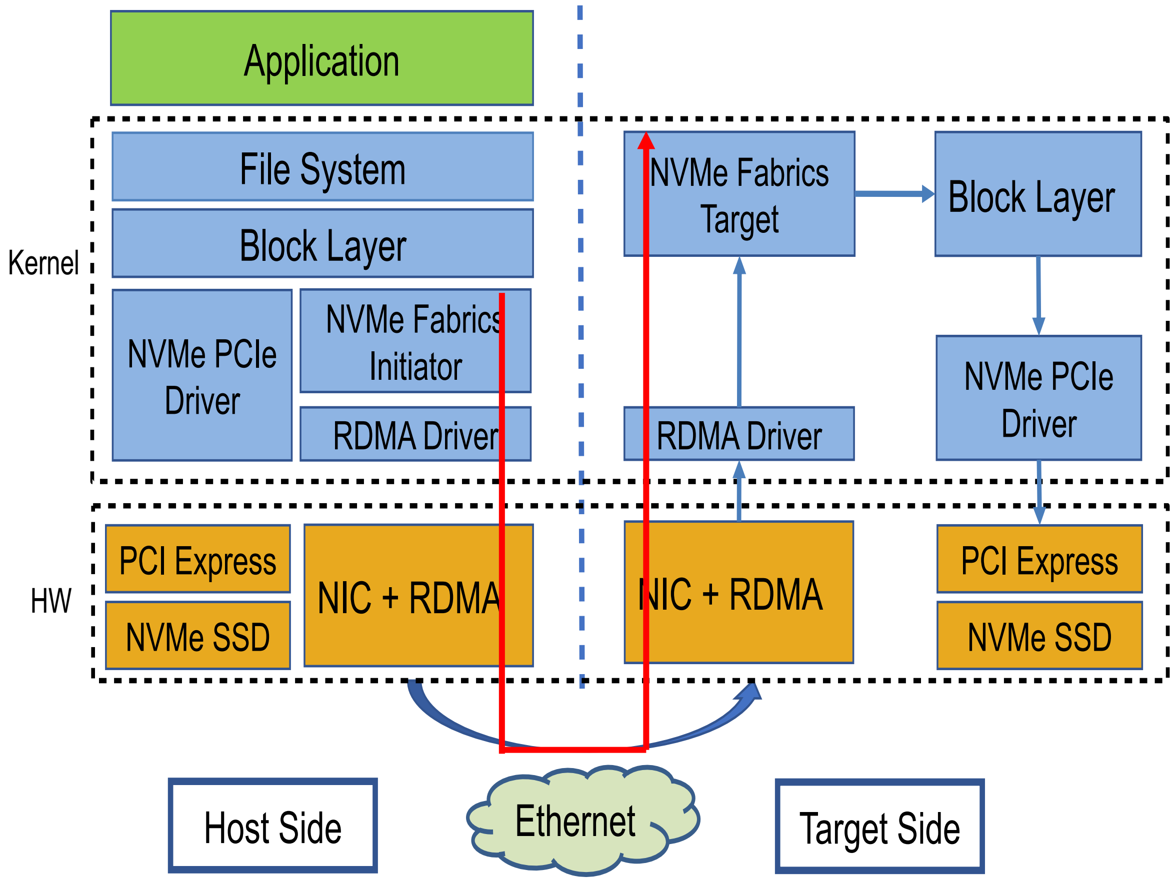
- Storage Disaggregation
  - Separated computing and storage
  - Reduced total cost of ownership (TCO)
  - Improved hardware utilization
  - Examples: NVMe over Fabrics, iSCSI



NVMe over Fabrics

Background: NVMe-over-Fabrics

- Direct Attached Storage (DAS)
  - Computing and storage in one box
  - Less flexible, hard to scale, etc
- Storage Disaggregation
  - Separated computing and storage
  - Reduced total cost of ownership (TCO)
  - Improved hardware utilization
  - Examples: NVMe over Fabrics, iSCSI

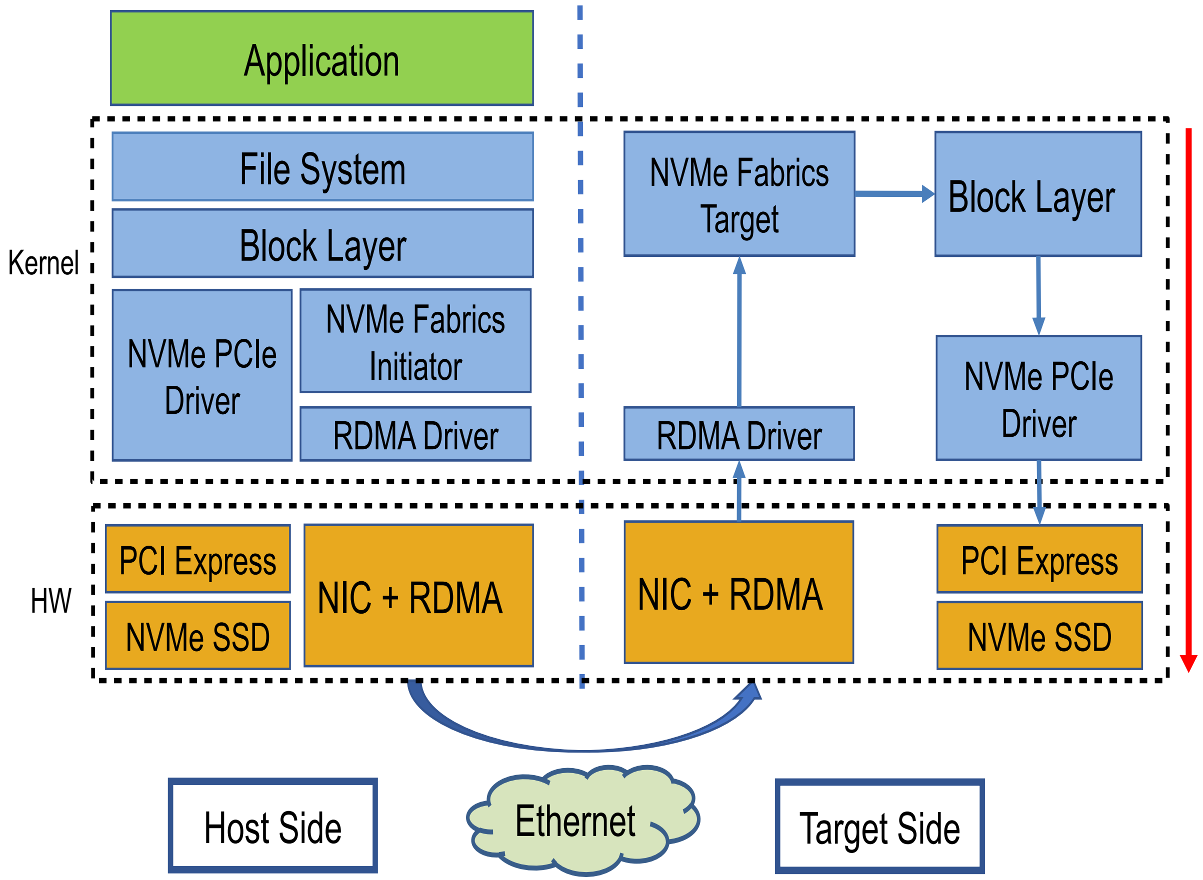


NVMe over Fabrics

Background: NVMe-over-Fabrics

- Direct Attached Storage (DAS)
  - Computing and storage in one box
  - Less flexible, hard to scale, etc

- Storage Disaggregation
  - Separated computing and storage
  - Reduced total cost of ownership (TCO)
  - Improved hardware utilization
  - Examples: NVMe over Fabrics, iSCSI



NVMe over Fabrics

Motivations

- Continuous investment in Arm-based solutions
- Increasingly popular NVMe over Fabrics
- Integrating Arm with NVMeoF is highly appealing
- However, the first-hand comprehensive experimental data is still lacking

Motivations

- Continuous investment in Arm-based solutions
- Increasingly popular NVMe over Fabrics
- Integrating Arm with NVMeoF is highly appealing
- However, the first-hand comprehensive experimental data is still lacking

A thorough performance study of NVMeoF on Arm is becoming necessary.

Experimental Setup

Experimental Setup

- **Target Side**: Broadcom 5880X Stingray.
  - CPU: 8-core 3GHz ARMv8 Coretx-A72 CPU
  - Memory: 48GB
  - Storage: Intel Data Center P3600 SSD
  - Network: Broadcom NetXtreme NIC
- **Host Side**: Lenovo ThinkCentre M910s
  - CPU: Intel(R) 4-core (HT) i7-6700 3.40GHz CPU
  - Memory: 16GB
  - Network: Broadcom NetXtreme NIC
- The host and target machines are connected by a Leoni ParaLink@23 cable
- Speed on both host and target sides is configured to be 50Gb/s
- Benchmarking tool: FIO

Server/Client	**Arm/x86**	**x86/Arm**
Bandwidth(Gb/s)	45.42	45.40
Latency (us)	3.26	3.17

RoCEv2 Performance

Experimental Results

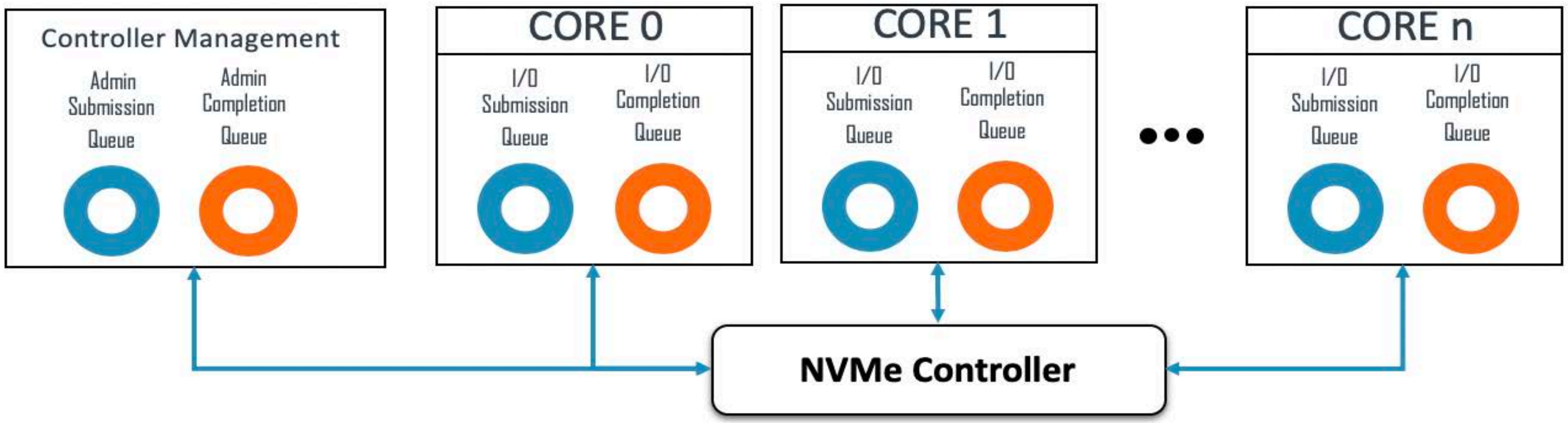
Experiments

- Effect of Parallelism
- Study of Computational Cost
- Effect of IODepth
- Effect of Request Sizes

Experiments

- **Effect of Parallelism**
- **Study of Computational Cost**
- **Effect of IODepth**
- Effect of Request Sizes

Parallelism Feature in NVMe

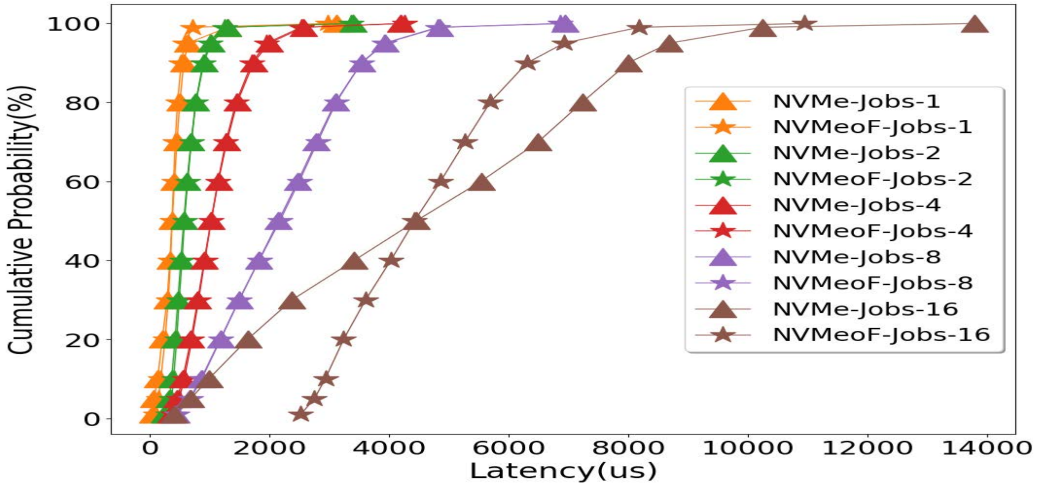


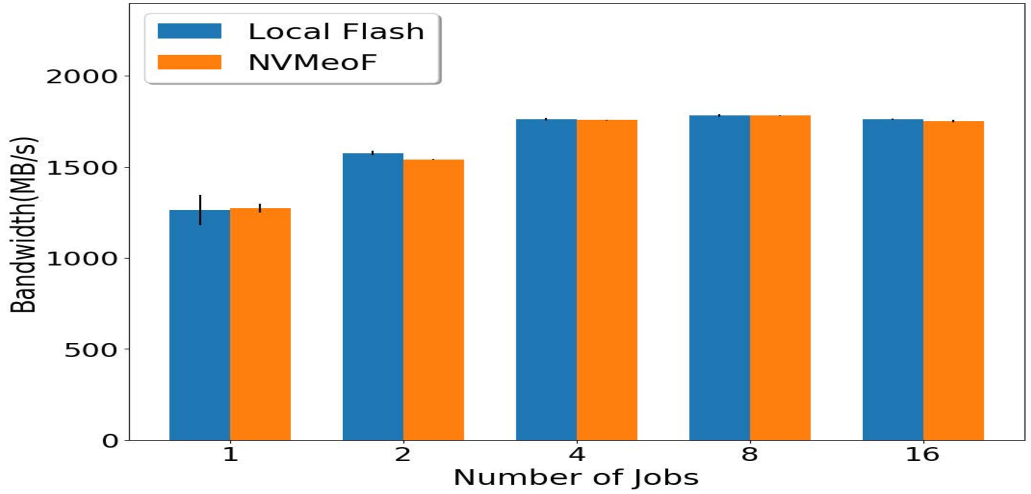
NVMe Structure*

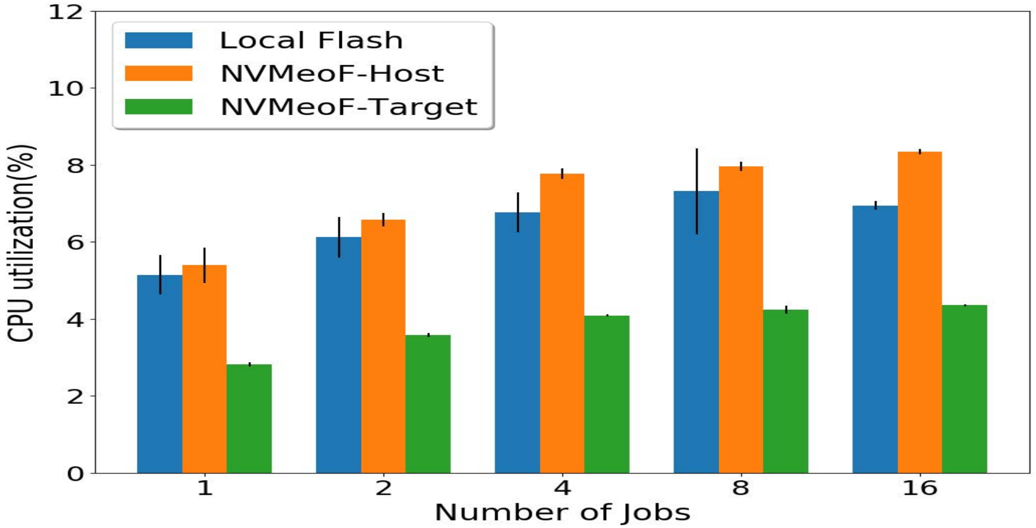
- Parallel I/Os play an important role in NVMe to fully exploit hardware potentials
- I/O parallelism will also have a great impact on NVMe-over-Fabrics

*https://nvmexpress.org/about/nvm-express-overview/

Finding #1: Effect of Parallelism



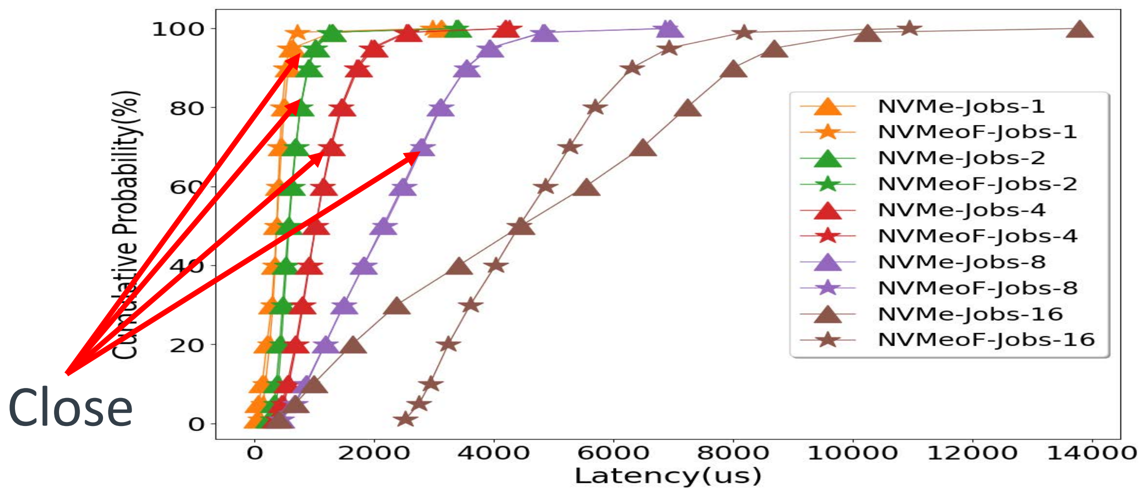


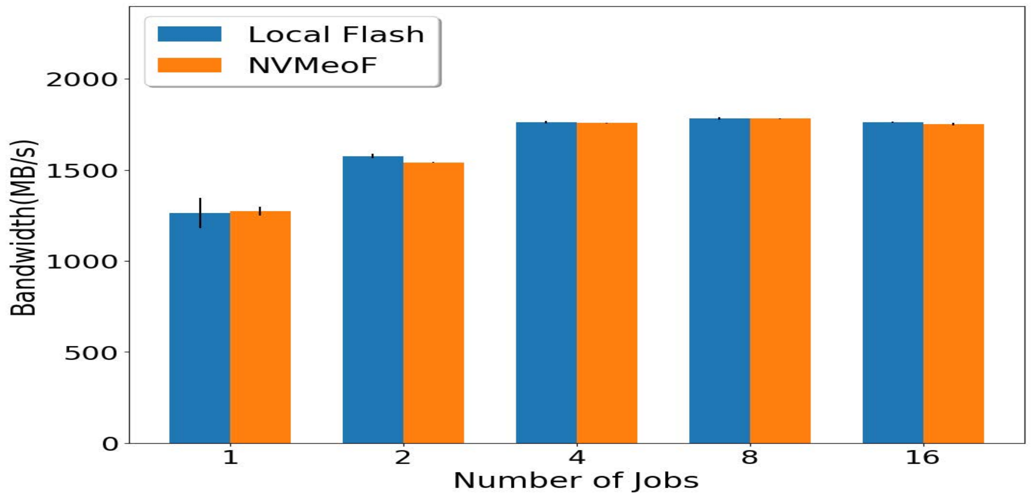


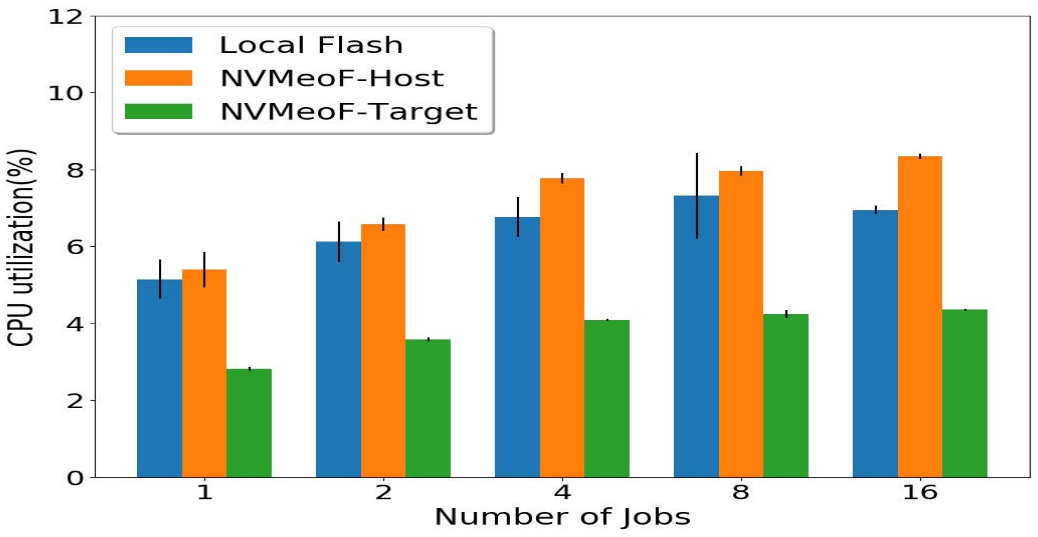
1. Latency increases as the number of jobs increases
2. NVMeoF has a close or shorter tail latency for seq read
3. BW reaches plateau when job number reaches 4
4. CPU utilization on target side is much lower
5. Arm is powerful enough to be storage server

*Sequential Read, 4KB, 128 IODepth, 1-16 Concurrent Jobs

Finding #1: Effect of Parallelism



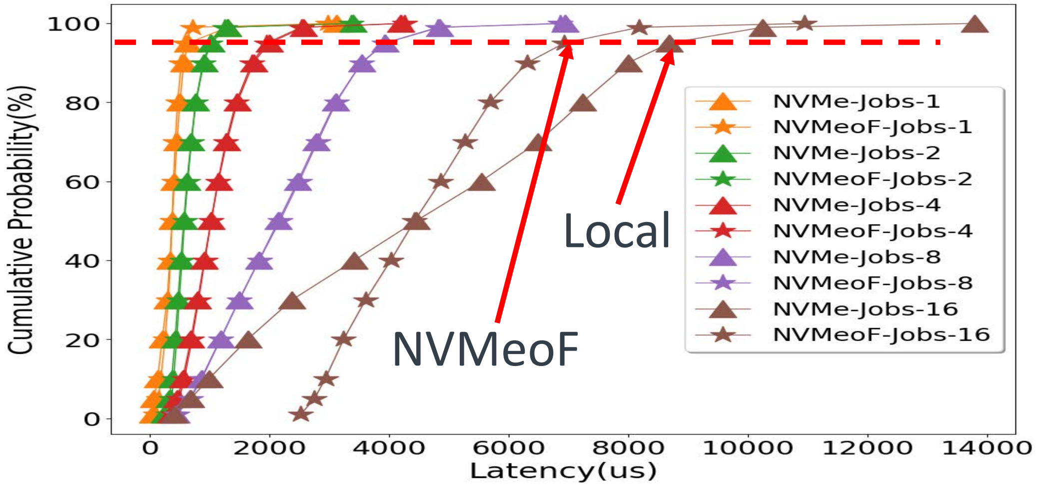


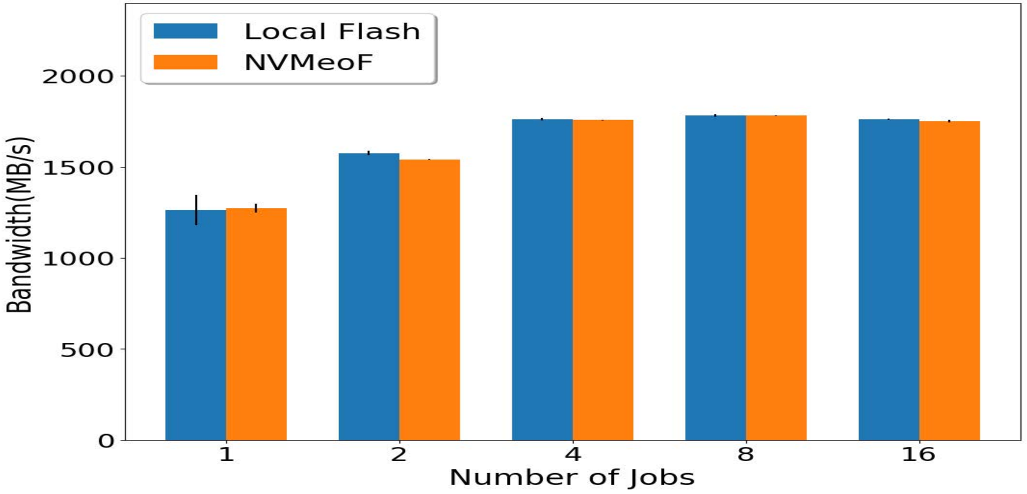


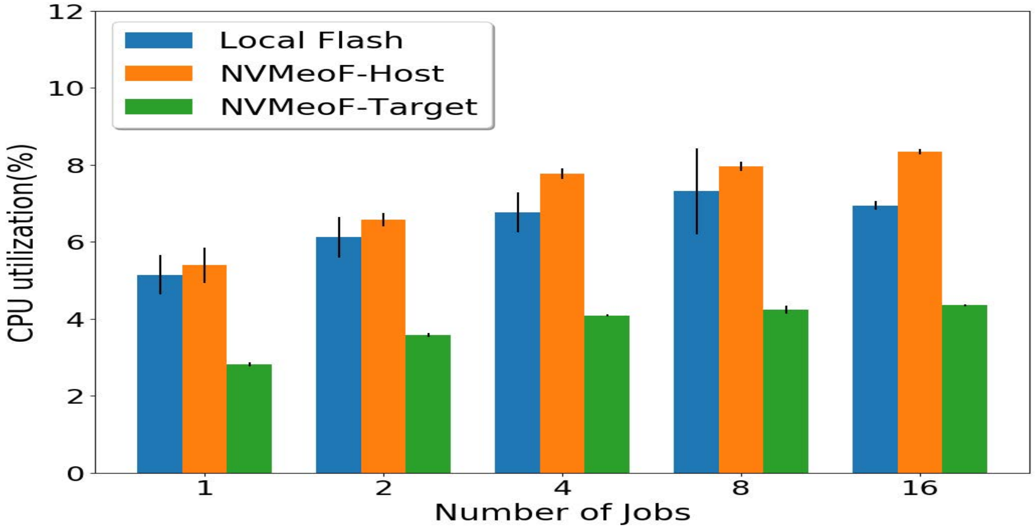
1. Latency increases as the number of jobs increases
2. NVMeoF has a close or shorter tail latency for seq read
3. BW reaches plateau when job number reaches 4
4. CPU utilization on target side is much lower
5. Arm is powerful enough to be storage server

*Sequential Read, 4KB, 128 IODepth, 1-16 Concurrent Jobs

Finding #1: Effect of Parallelism



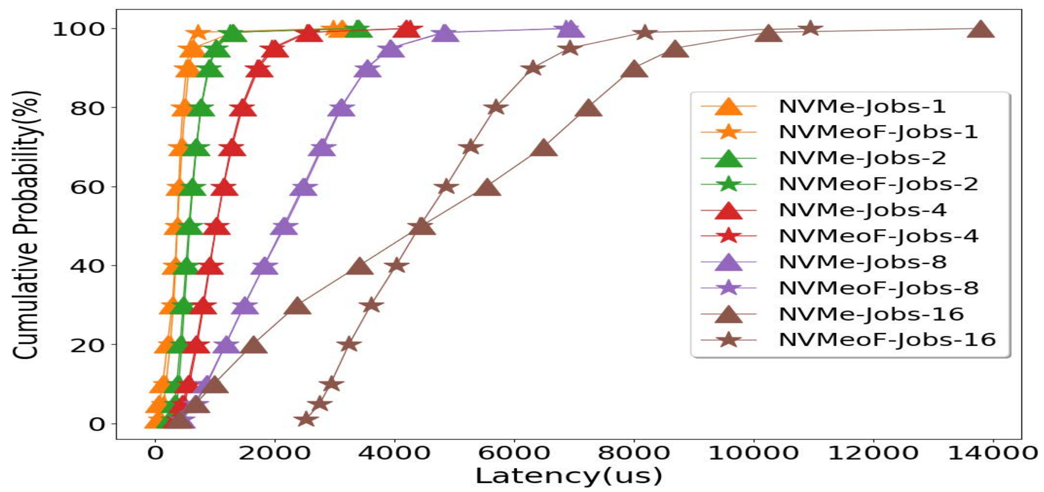


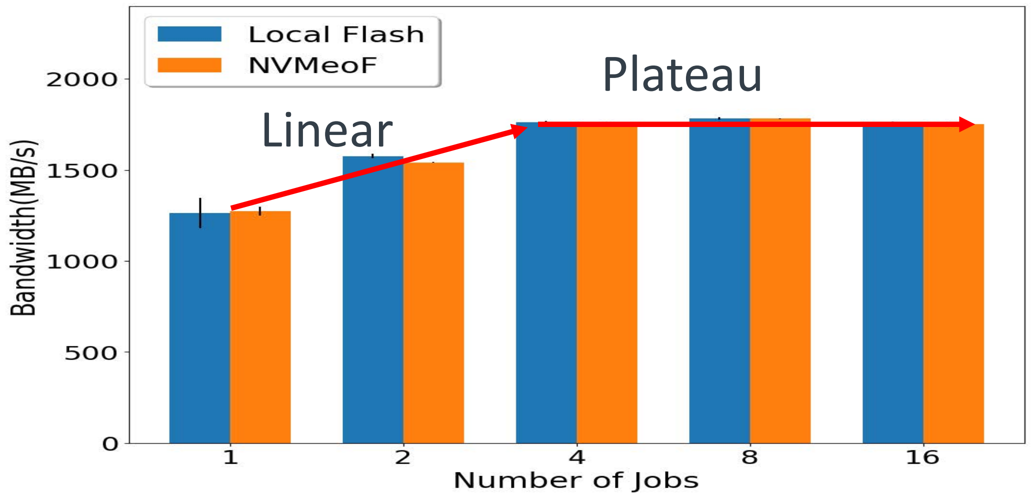


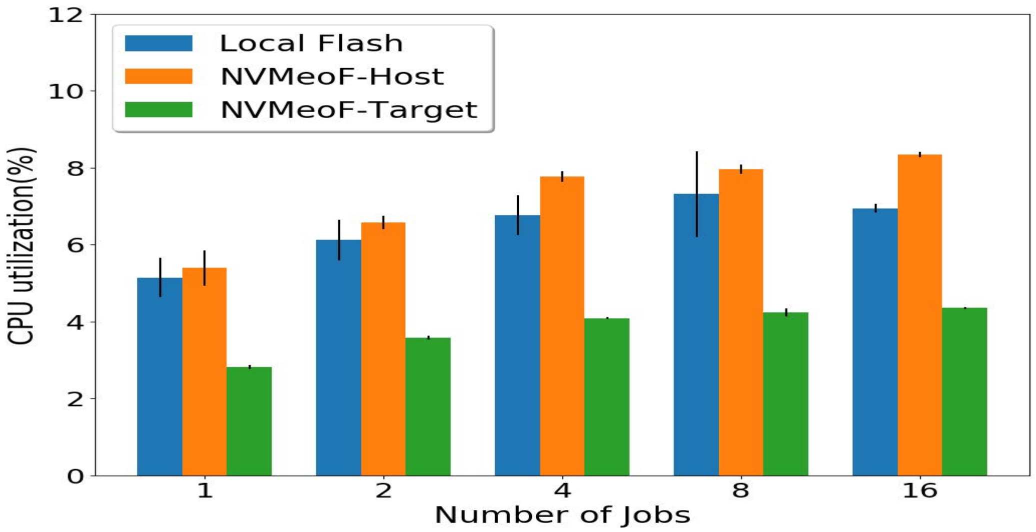
1. Latency increases as the number of jobs increases
2. NVMeoF has a close or shorter tail latency for seq read
3. BW reaches plateau when job number reaches 4
4. CPU utilization on target side is much lower
5. Arm is powerful enough to be storage server

*Sequential Read, 4KB, 128 IODepth, 1-16 Concurrent Jobs

Finding #1: Effect of Parallelism



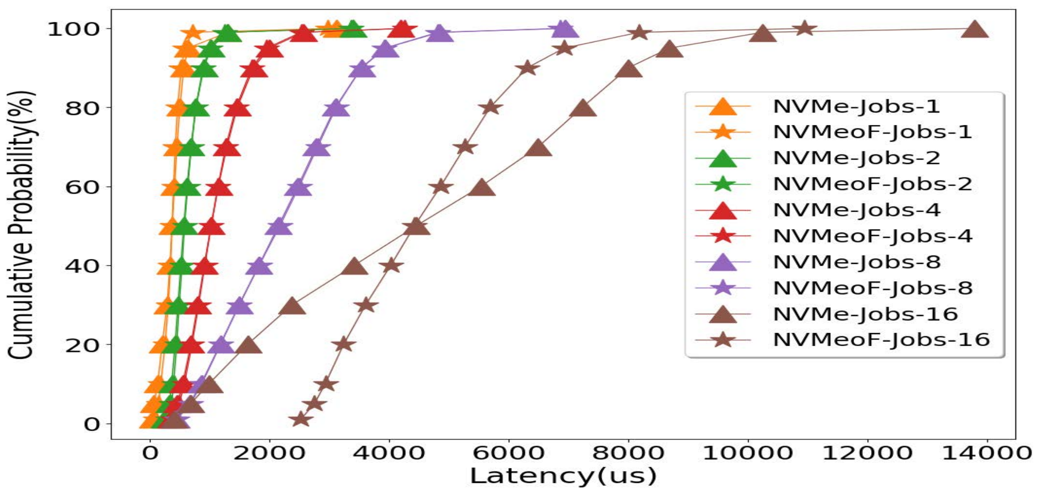


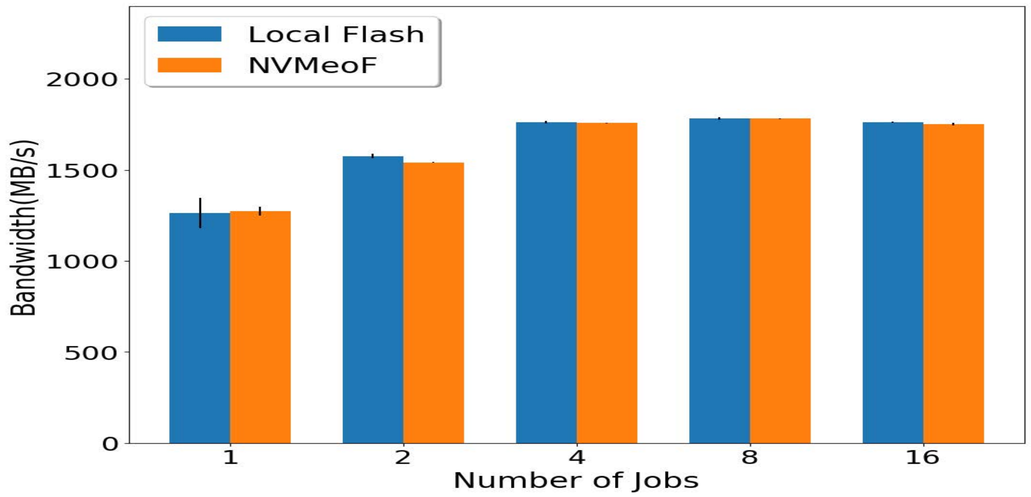


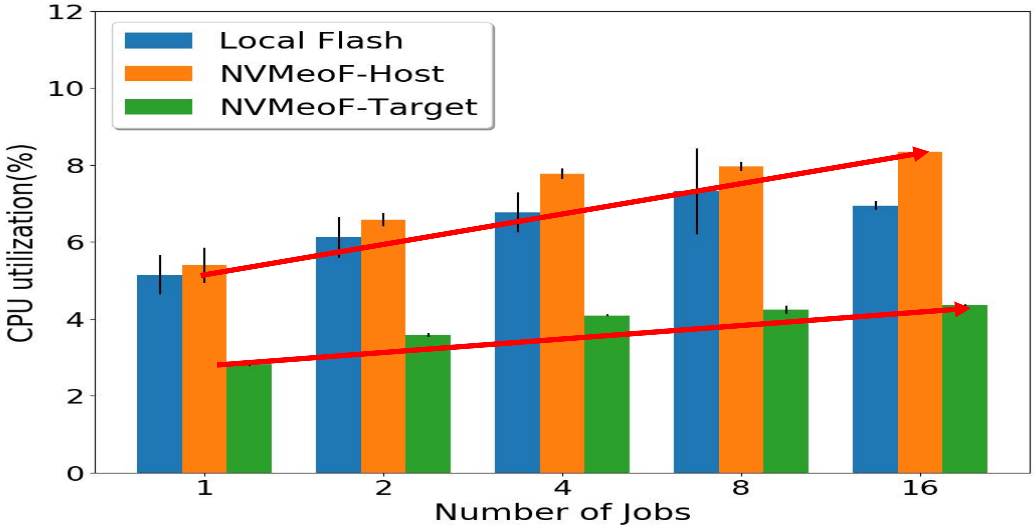
1. Latency increases as the number of jobs increases
2. NVMeoF has a close or shorter tail latency for seq read
3. BW reaches plateau when job number reaches 4
4. CPU utilization on target side is much lower
5. Arm is powerful enough to be storage server

*Sequential Read, 4KB, 128 IODepth, 1-16 Concurrent Jobs

Finding #1: Effect of Parallelism



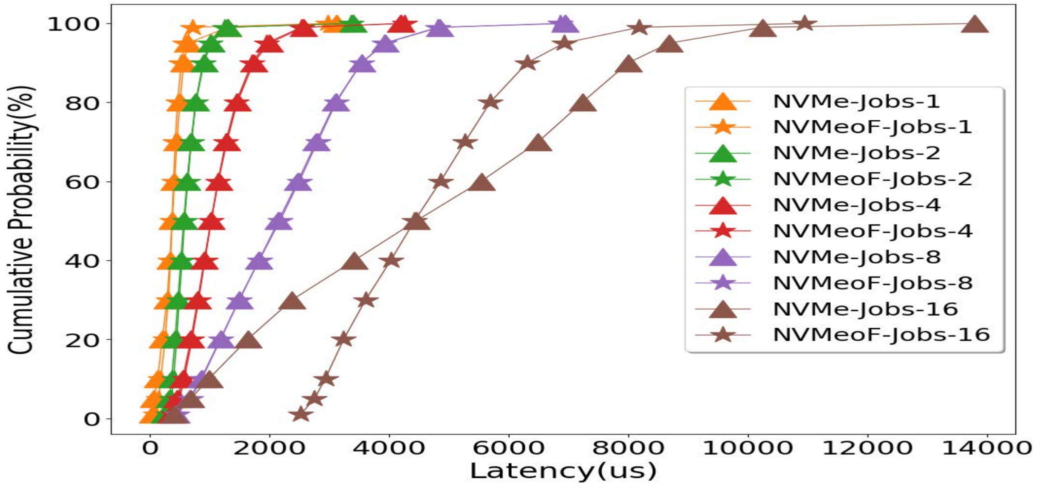


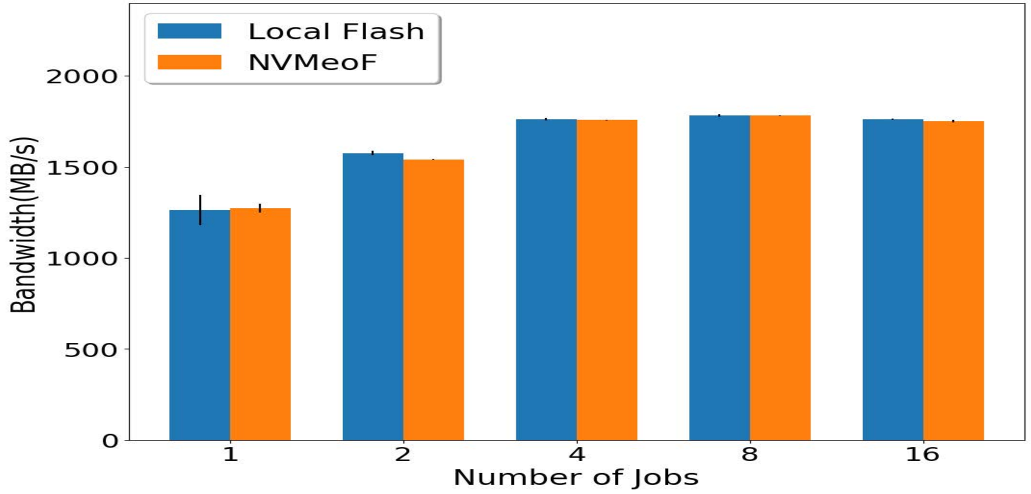


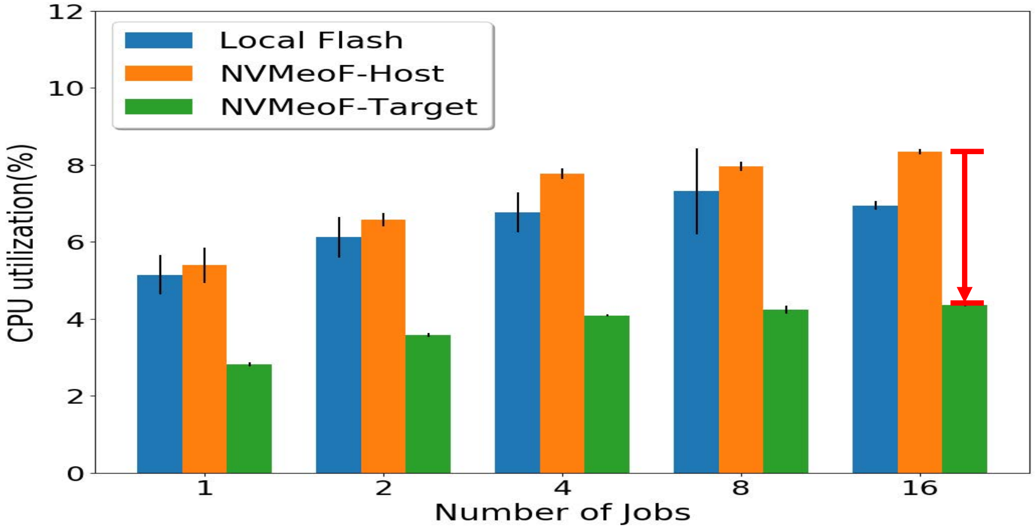
1. Latency increases as the number of jobs increases
2. NVMeoF has a close or shorter tail latency for seq read
3. BW reaches plateau when job number reaches 4
4. CPU utilization on target side is much lower
5. Arm is powerful enough to be storage server

*Sequential Read, 4KB, 128 IODepth, 1-16 Concurrent Jobs

Finding #1: Effect of Parallelism



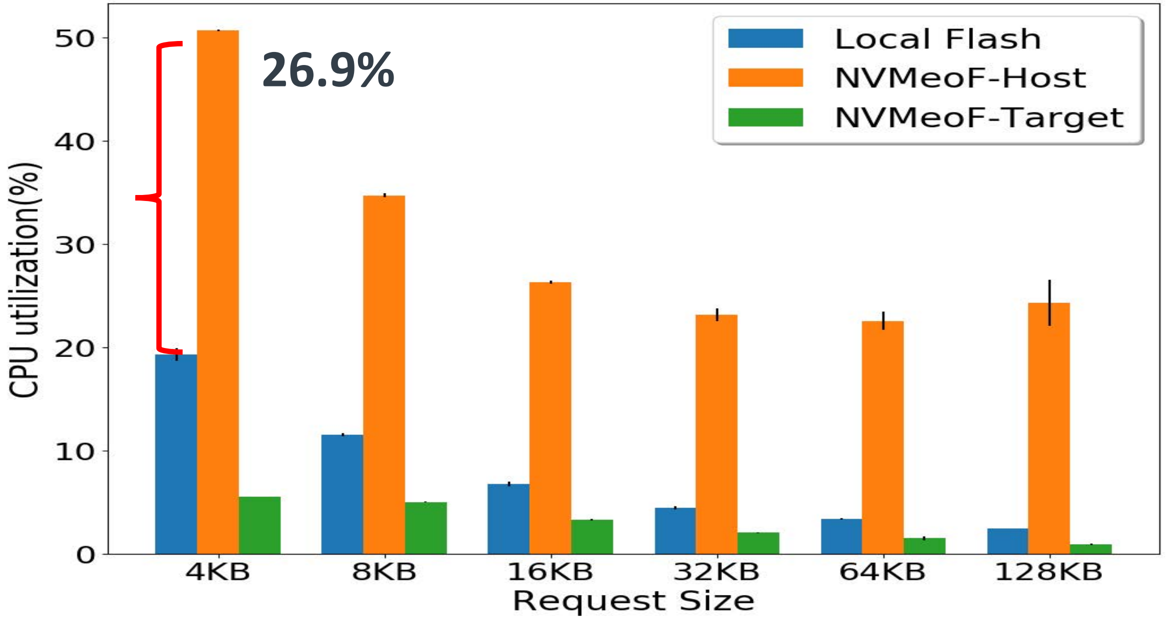




1. Latency increases as the number of jobs increases
2. NVMeoF has a close or shorter tail latency for seq read
3. BW reaches plateau when job number reaches 4
4. CPU utilization on target side is much lower
5. Arm is powerful enough to be storage server

*Sequential Read, 4KB, 128 IODepth, 1-16 Concurrent Jobs

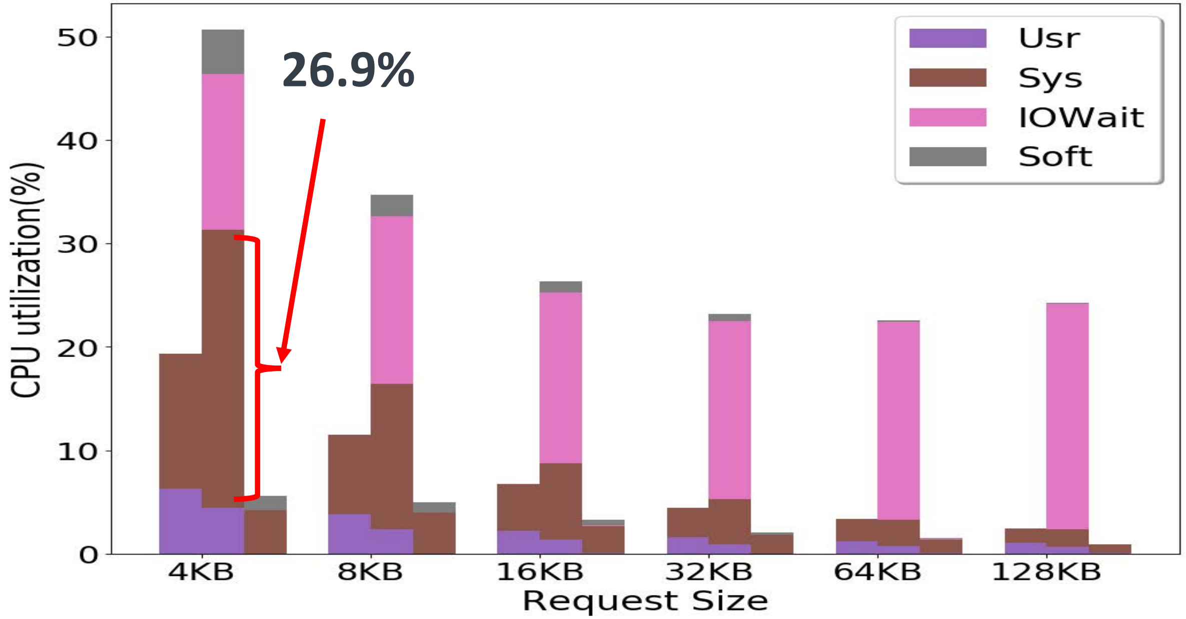
Finding #2 : Computational Cost



1. NVMeoF consumes 31.5% more CPU on host side than local NVMe
2. Kernel level overhead is dominant(26.9%) when request size is 4KB
3. Kernel level overhead are amortized as request size increases

*Random Write, 4 KB-128KB, 8 Concurrent jobs, 128 IODepth

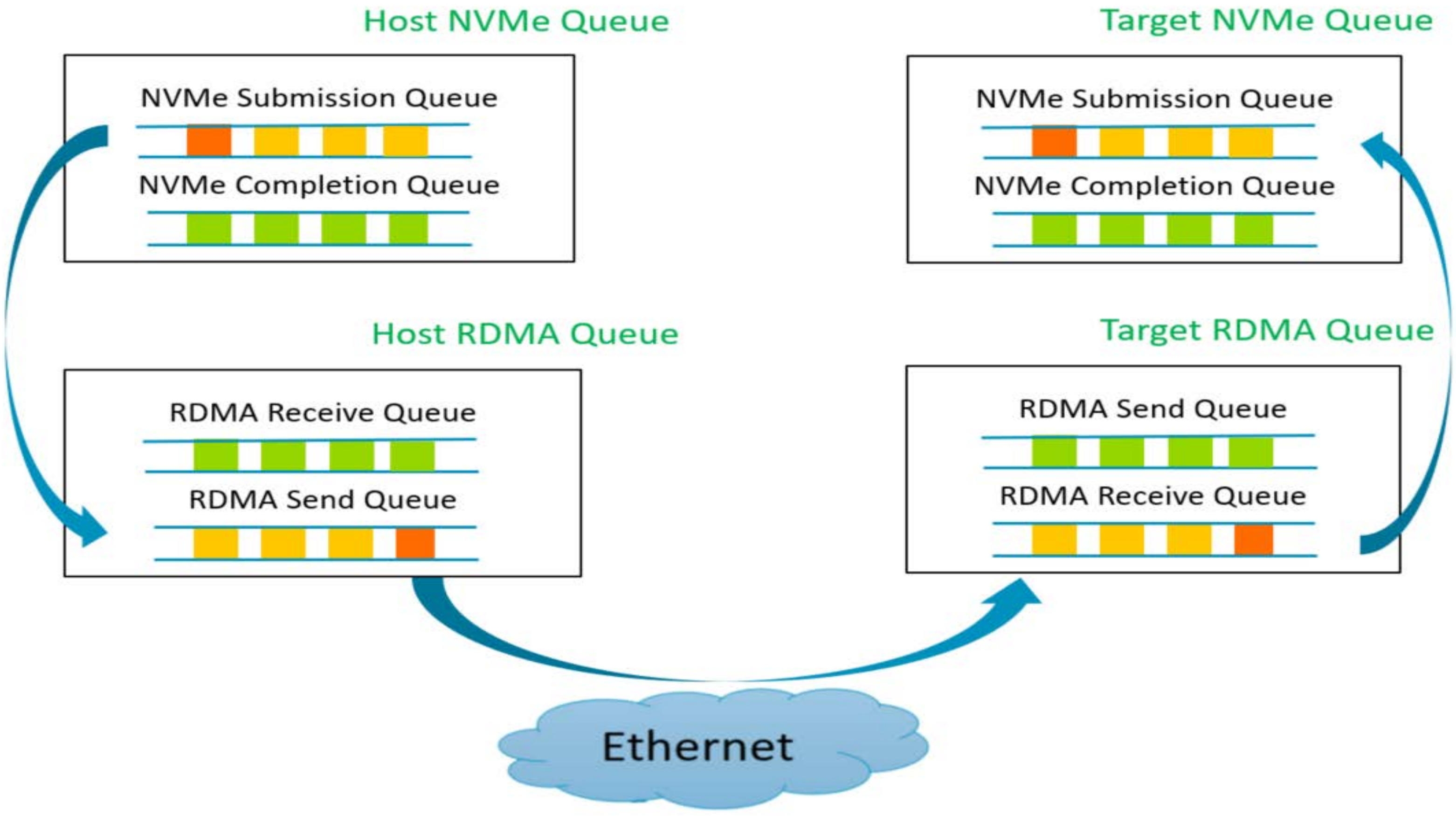
Finding #2 : Computational Cost



1. NVMeoF consumes 31.5% more CPU on host side than local NVMe
2. Kernel level overhead is dominant(26.9%) when request size is 4KB
3. Kernel level overhead are amortized as request size increases

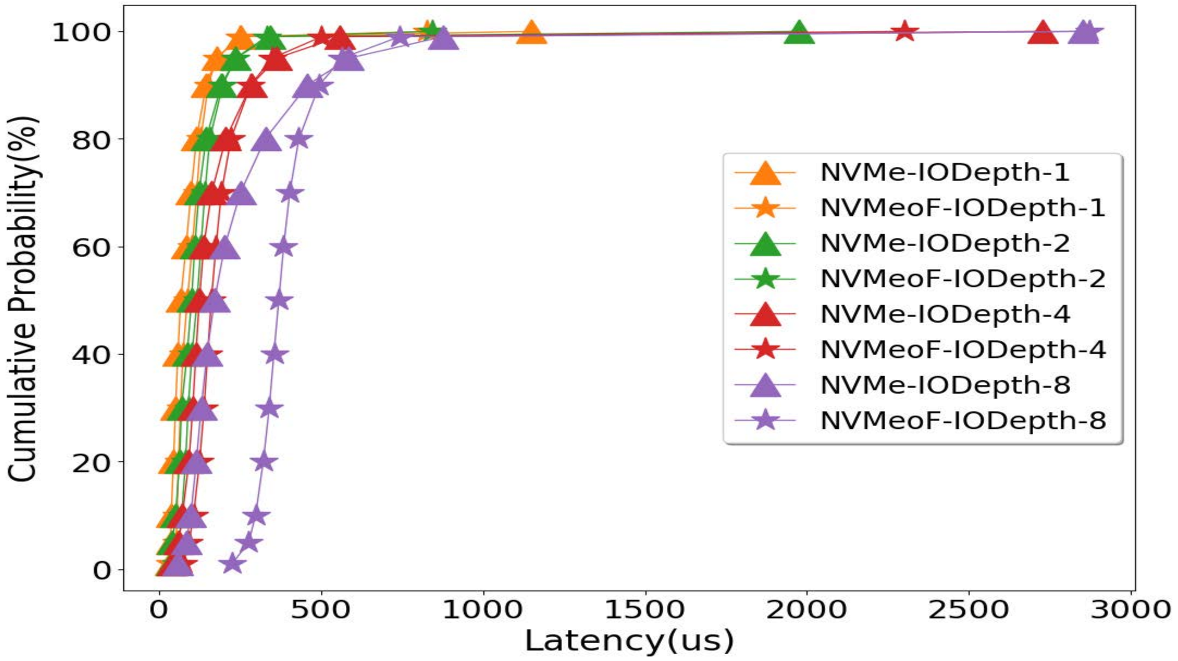
*Random Write, 4 KB-128KB, 8 Concurrent jobs, 128 IODepth

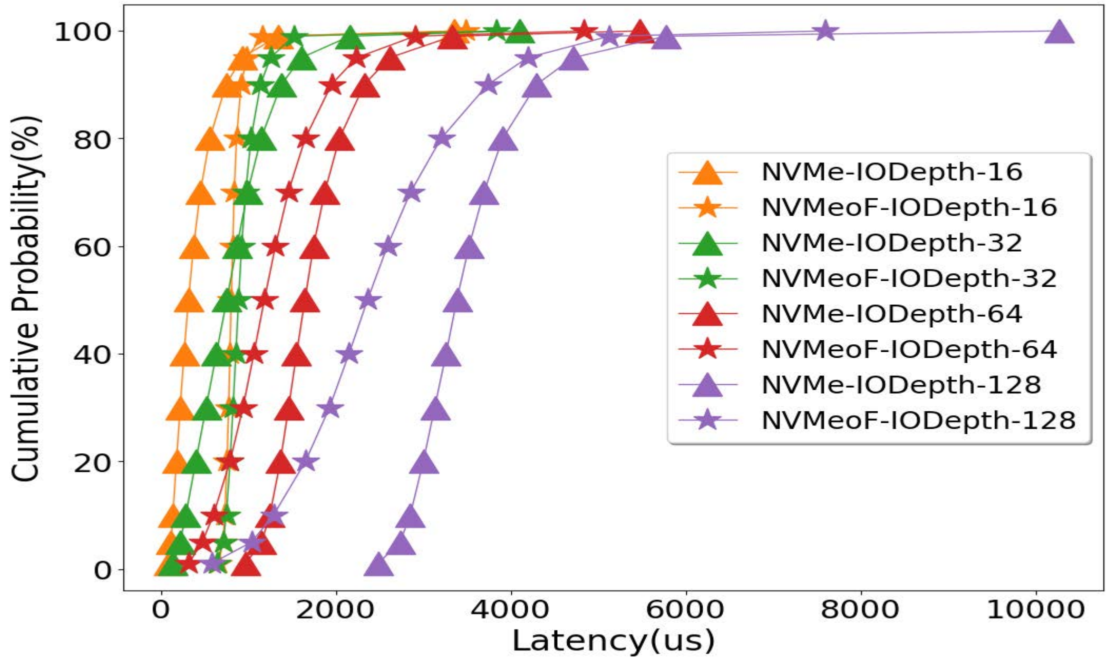
IODepth is important for NVMeoF



NVMe and RDMA Queues

Finding #3: Effect of IODepth

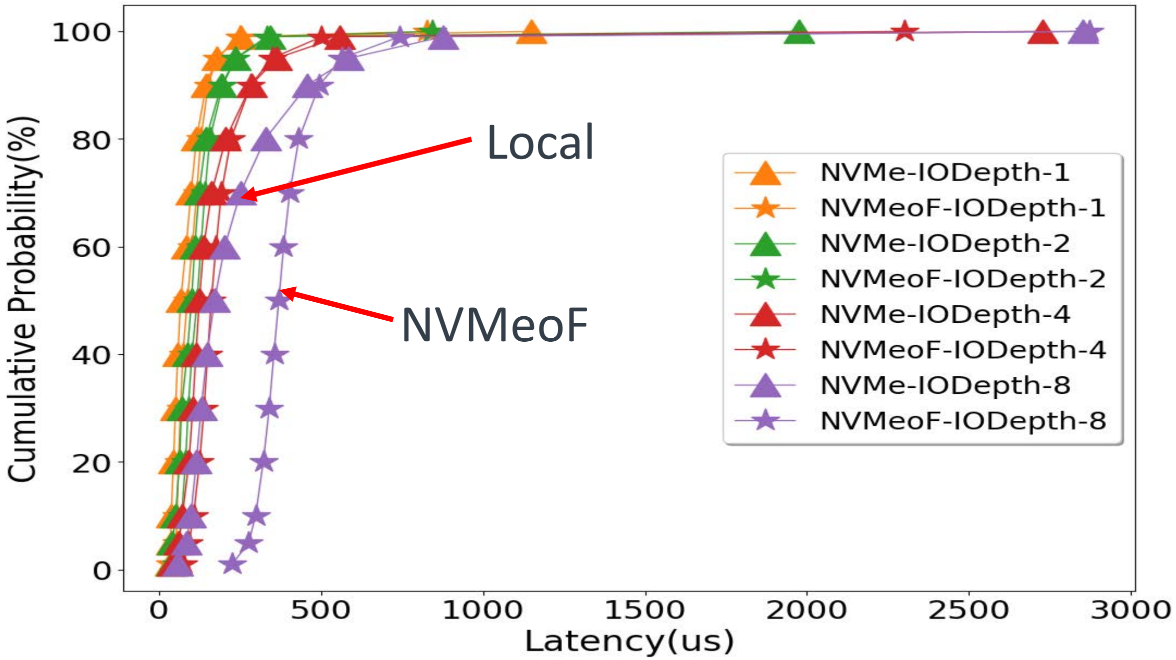


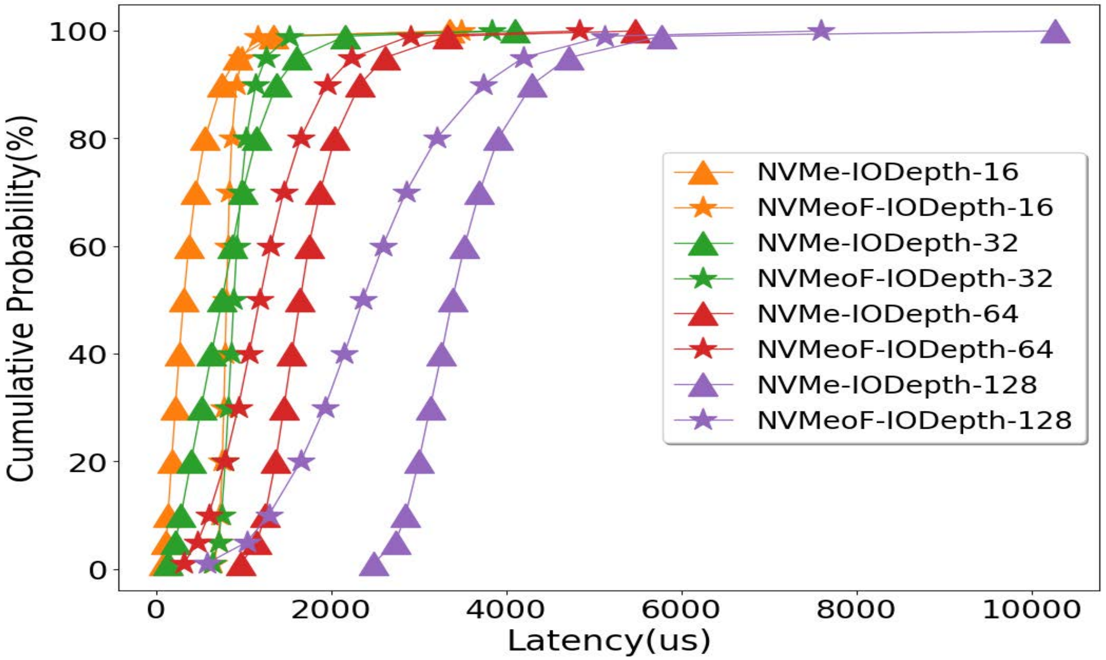


- When IODepth is small, local access has a short tail latency than remote access
- When IODepth is large, remote access has a short tail latency than local access

* Sequential Read, 4KB, 8 Concurrent Jobs, 1-128 IODepth, one Arm core

Finding #3: Effect of IODepth

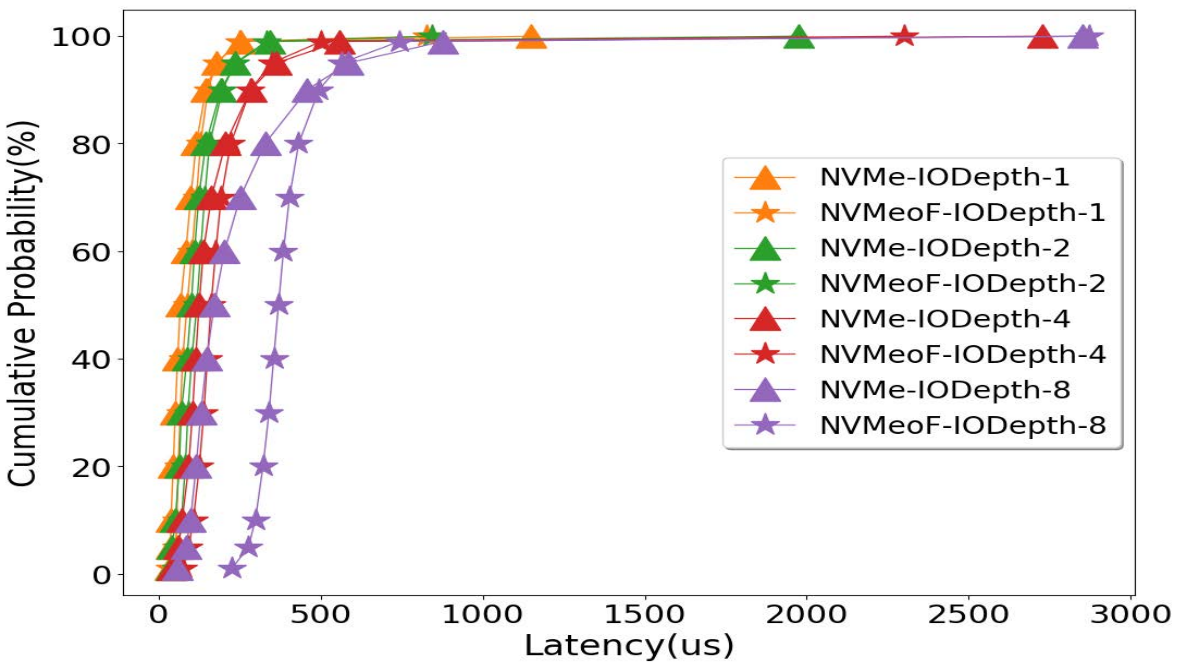


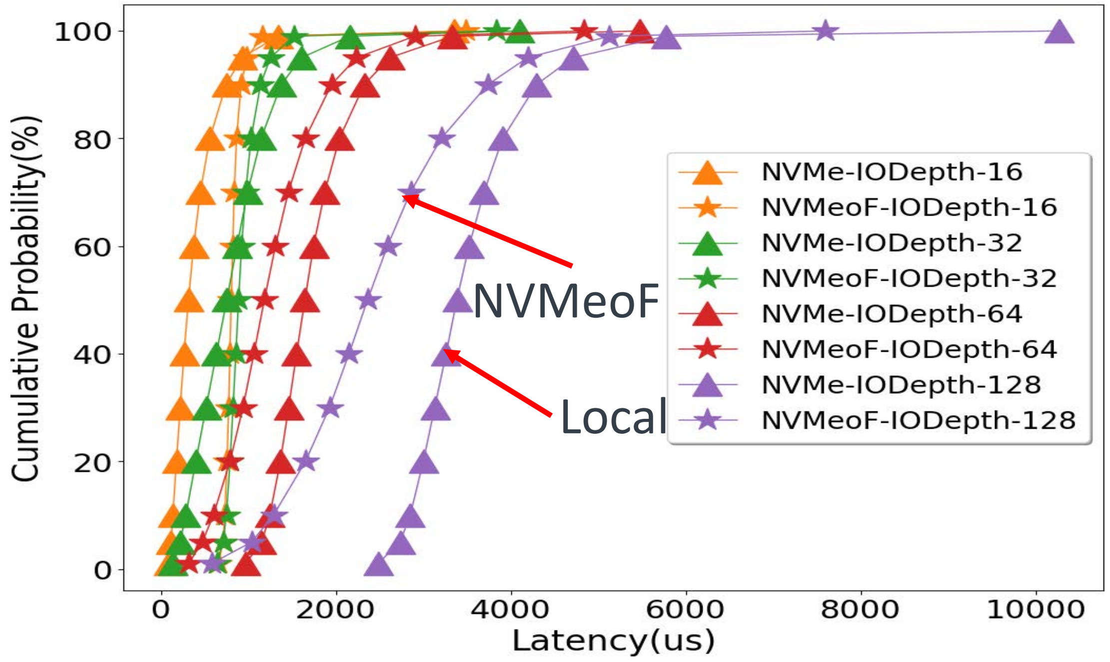


- When IODepth is small, local access has a short tail latency than remote access
- When IODepth is large, remote access has a short tail latency than local access

* Sequential Read, 4KB, 8 Concurrent Jobs, 1-128 IODepth, one Arm core

Finding #3: Effect of IODepth

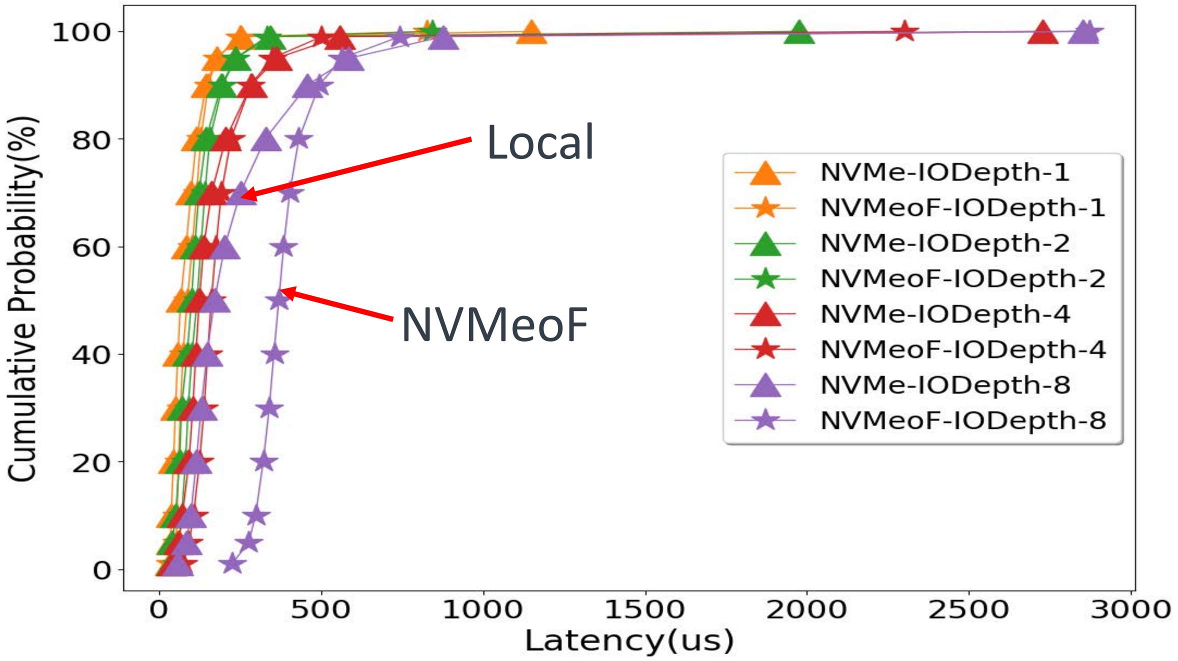


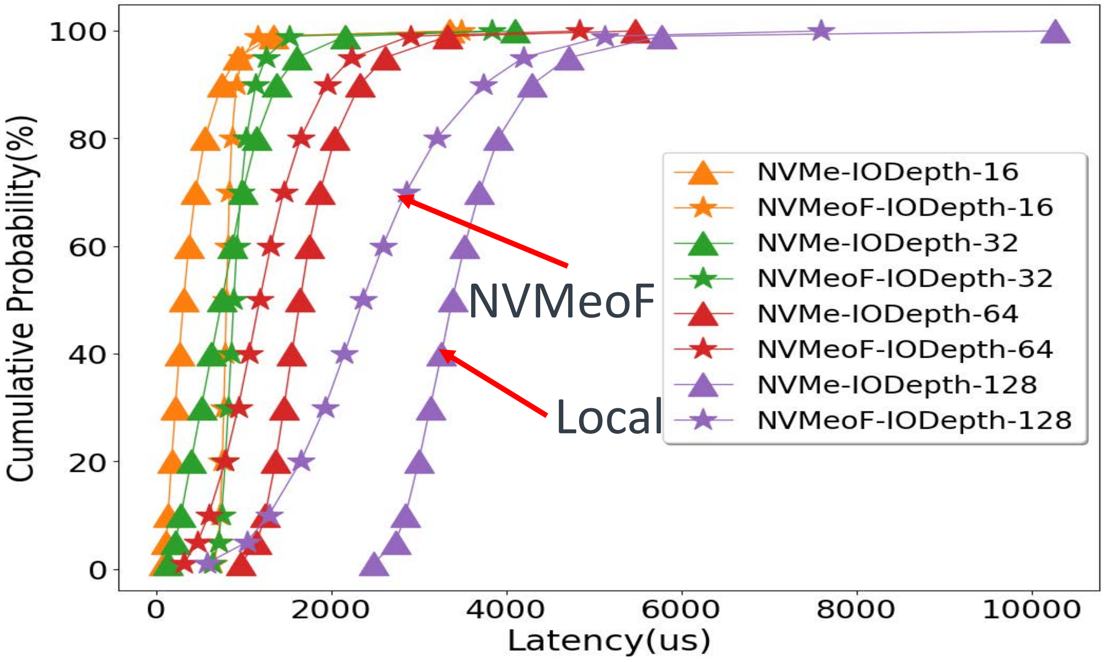


- When IODepth is small, local access has a short tail latency than remote access
- When IODepth is large, remote access has a short tail latency than local access

* Sequential Read, 4KB, 8 Concurrent Jobs, 1-128 IODepth, one Arm core

Finding #3: Effect of IODepth

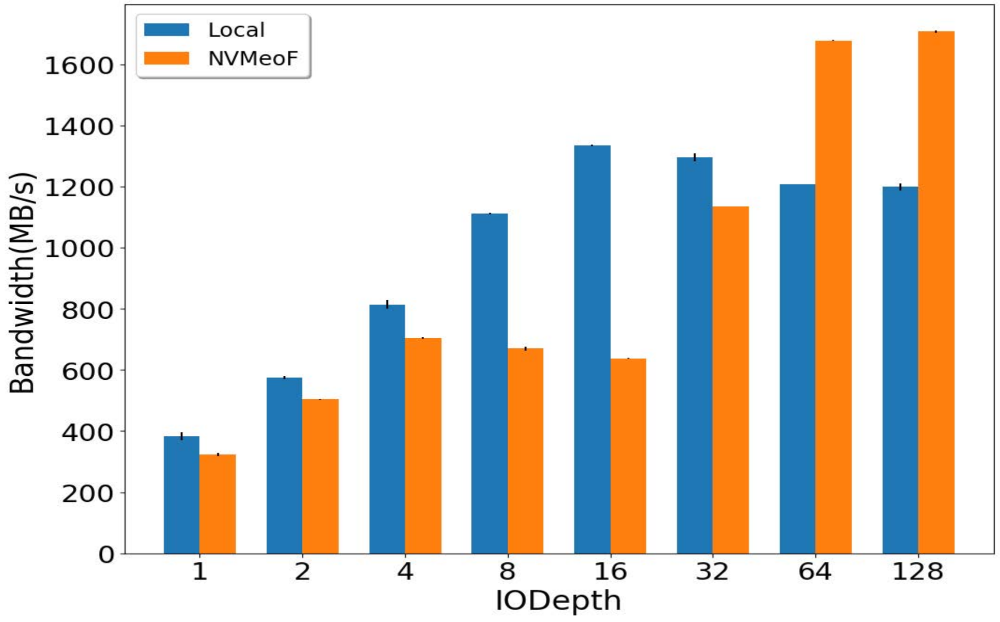


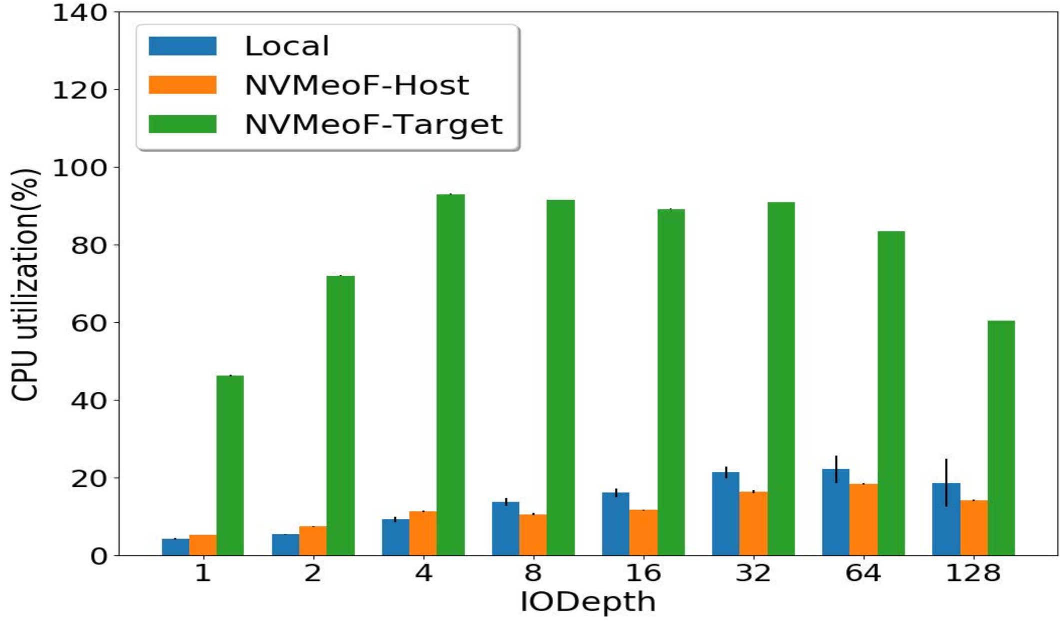


- When IODepth is small, local access has a short tail latency than remote access
- When IODepth is large, remote access has a short tail latency than local access

* Sequential Read, 4KB, 8 Concurrent Jobs, 1-128 IODepth, one Arm core

Finding #3: Effect of IODepth cont’d

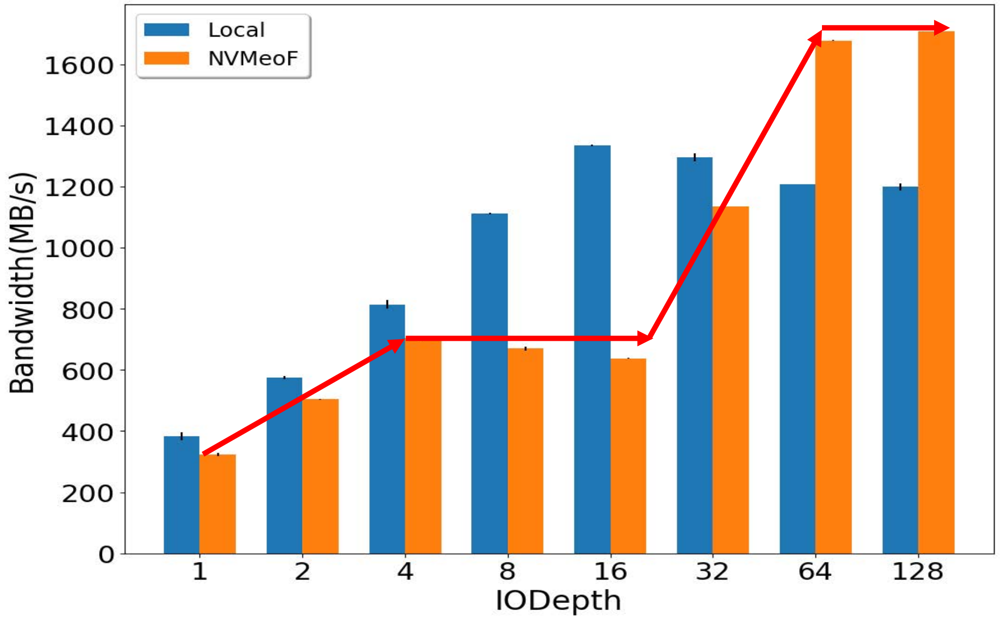


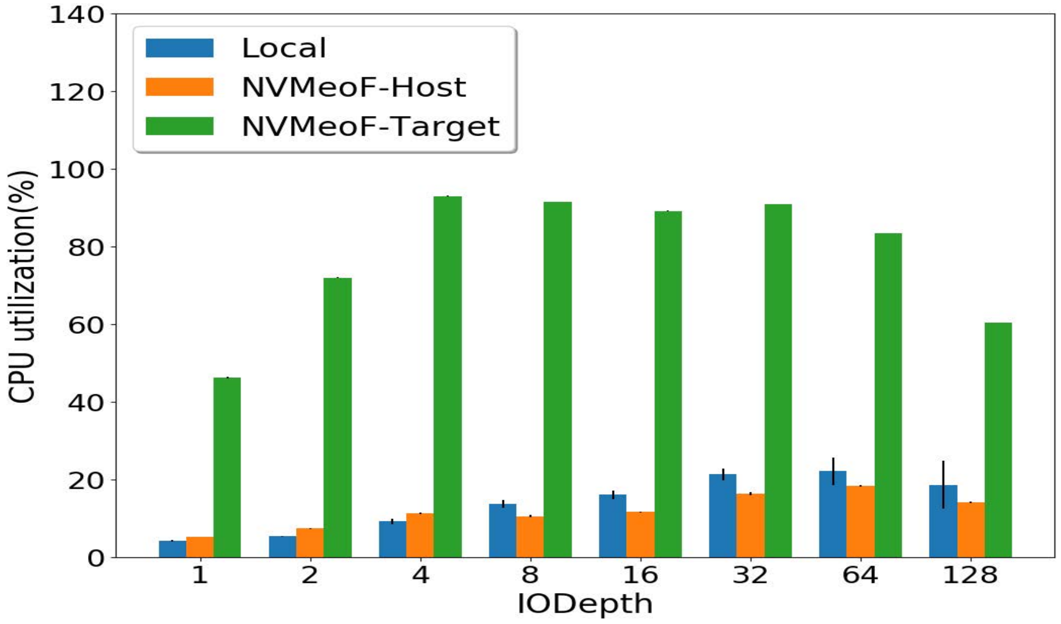


- Bandwidth will increase, and keep stable and increase again when IODepth is over 32.
- CPU utilization will increase, and keep stable and decrease when IODepth is larger than 32.

* Sequential Read, 4KB, 8 Concurrent Jobs, 1-128 IODepth, one Arm core

Finding #3: Effect of IODepth cont'd

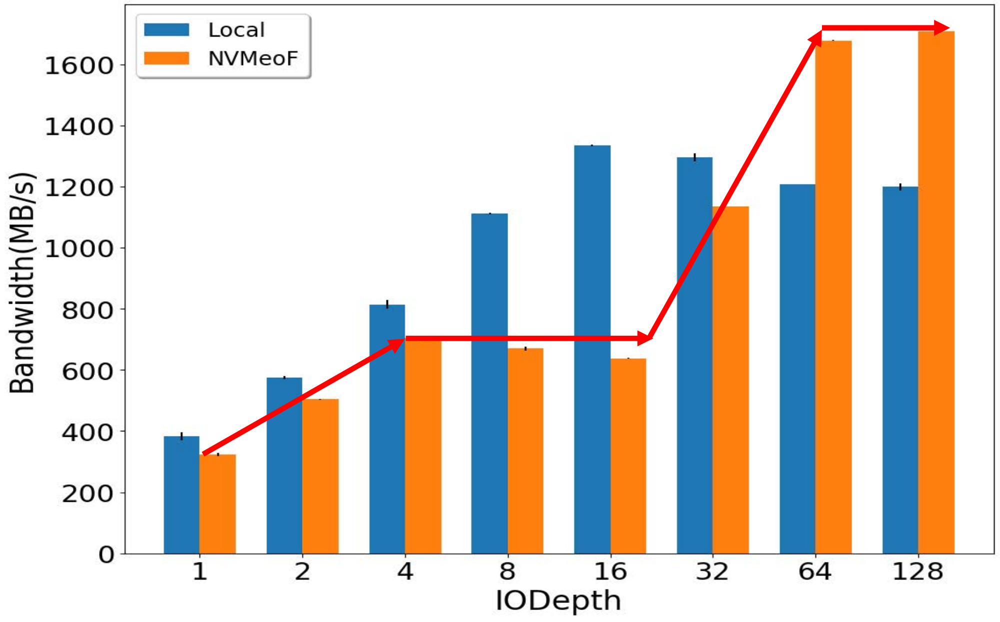


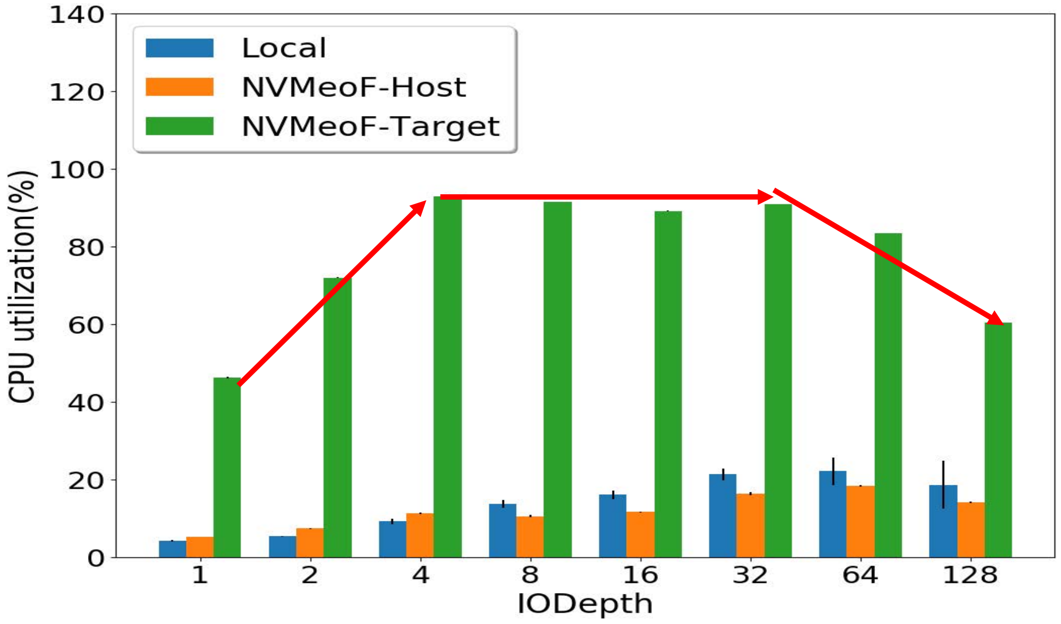


- Bandwidth will increase, and keep stable and increase again when IODepth is over 32.
- CPU utilization will increase, and keep stable and decrease when IODepth is larger than 32.

* Sequential Read, 4KB, 8 Concurrent Jobs, 1-128 IODepth, one Arm core

Finding #3: Effect of IODepth cont'd



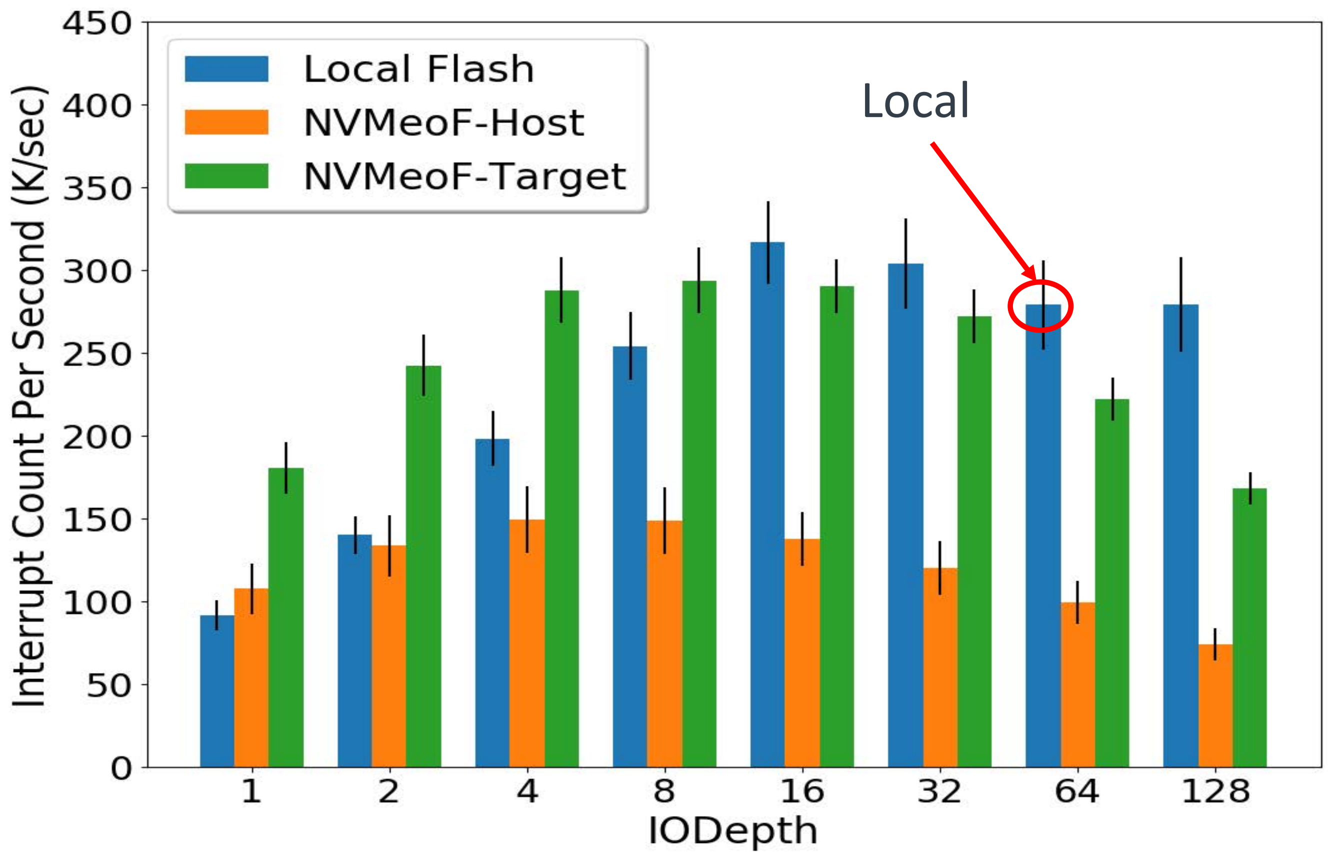


- Bandwidth will increase, and keep stable and increase again when IODepth is over 32.
- CPU utilization will increase, and keep stable and decrease when IODepth is larger than 32.

* Sequential Read, 4KB, 8 Concurrent Jobs, 1-128 IODepth, one Arm core

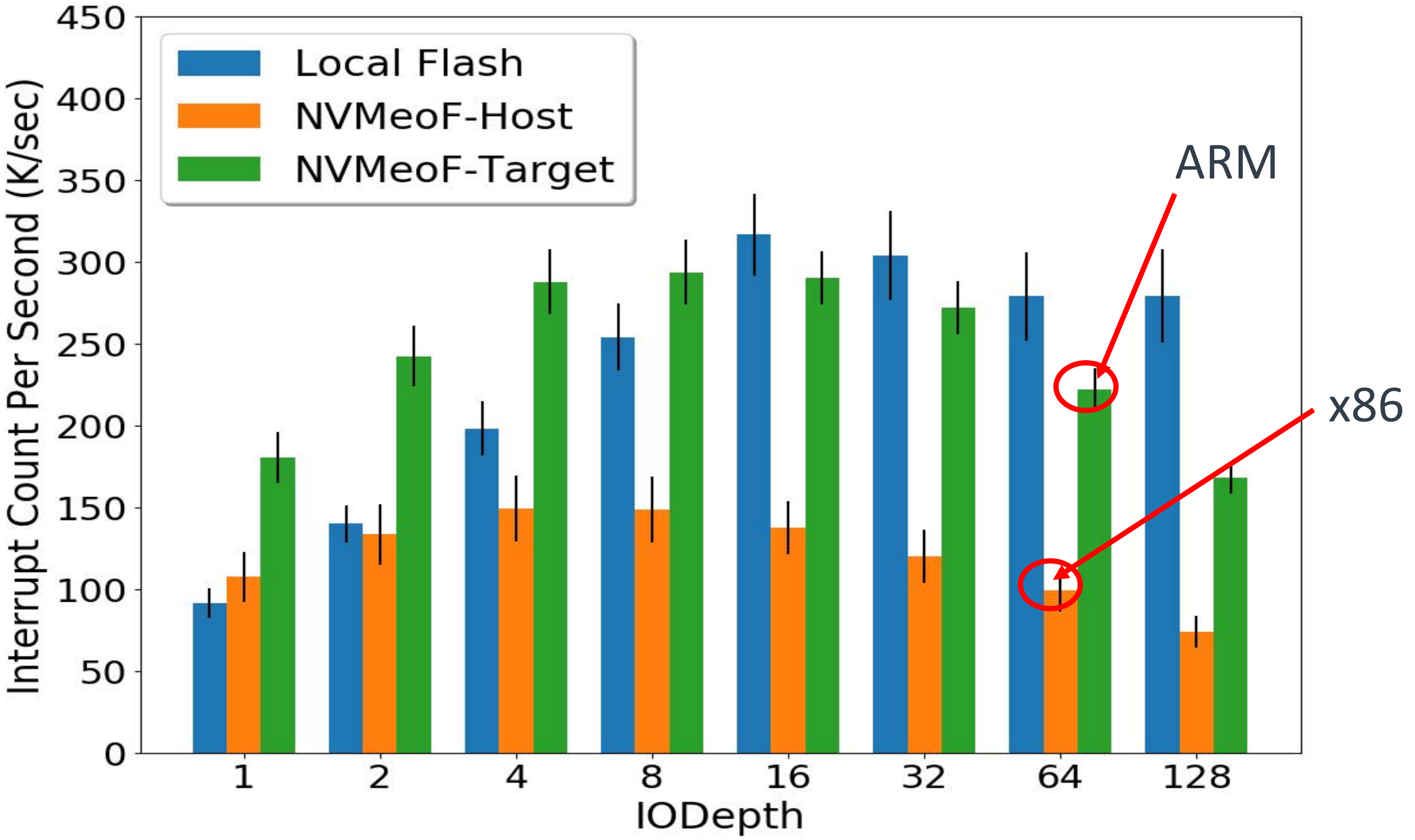
Interrupt Moderation

- Interrupt moderation means multiple packets are handled for each interrupt
- Overall interrupt-processing efficiency is improved and CPU utilization is decreased



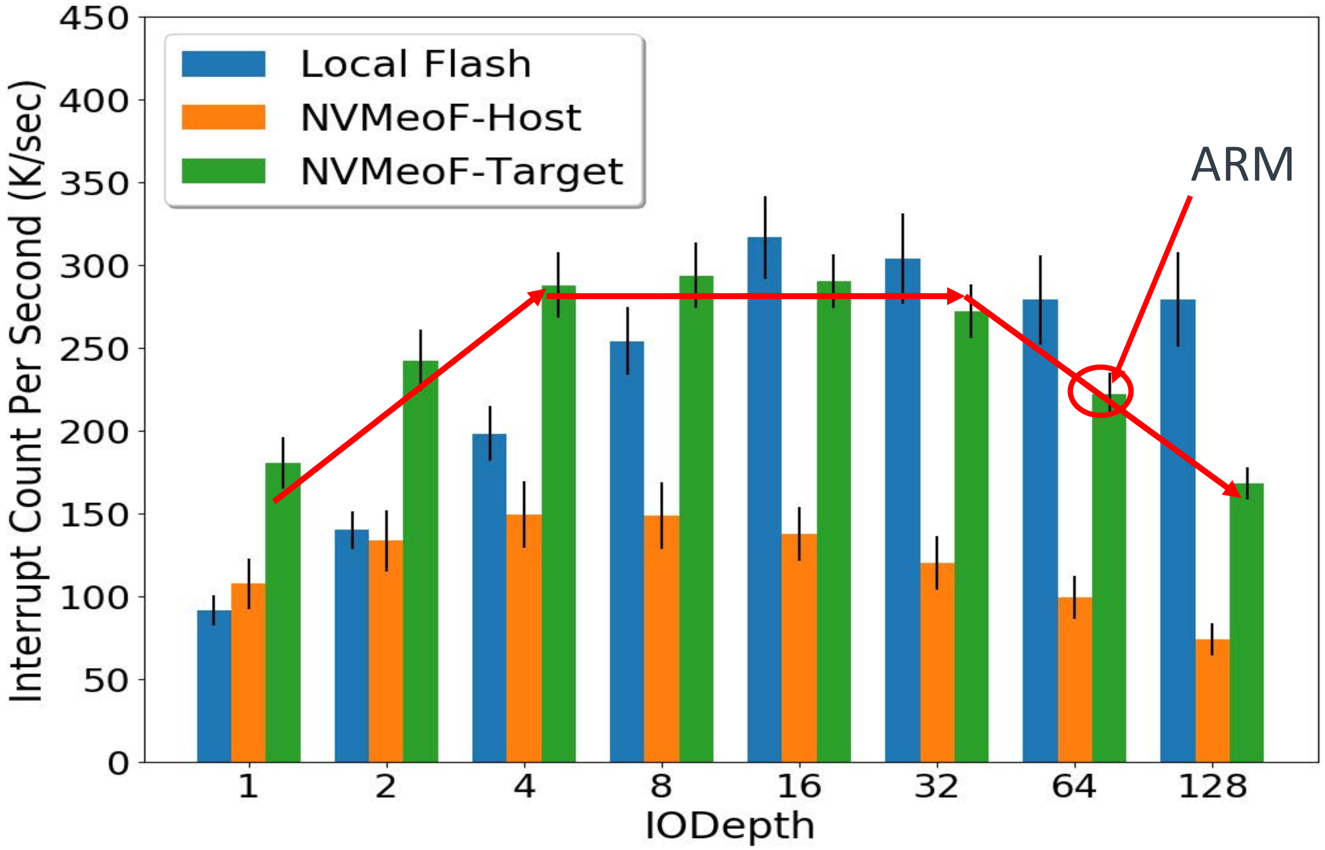
Interrupt Moderation

- Interrupt moderation means multiple packets are handled for each interrupt
- Overall interrupt-processing efficiency is improved and CPU utilization is decreased



Interrupt Moderation

- Interrupt moderation means multiple packets are handled for each interrupt
- Overall interrupt-processing efficiency is improved and CPU utilization is decreased



Summary and Implications

Observations

- NVMeoF can provide satisfactory and comparable performance to NVMe

Observations

- NVMeoF can provide satisfactory and comparable performance to NVMe
- Arm processor is powerful enough as the NVMeoF target

Observations

- NVMeoF can provide satisfactory and comparable performance to NVMe
- Arm processor is powerful enough as the NVMeoF target
- Request size, parallelism, and I/O queue depth are important for performance

Observations

- NVMeoF can provide satisfactory and comparable performance to NVMe
- Arm processor is powerful enough as the NVMeoF target
- Request size, parallelism, and I/O queue depth are important for performance
- Kernel level overhead can be significant on NVMeoF host

Observations

- NVMeoF can provide satisfactory and comparable performance to NVMe
- Arm processor is powerful enough as the NVMeoF target
- Request size, parallelism, and I/O queue depth are important for performance
- Kernel level overhead can be significant on NVMeoF host
- Interrupt moderation is important for overall performance improvement

Implications

- Application level
  - I/O Clustering. Merging small random operations into large sequential ones.
  - A proper configuration, such as parallelism, request size, etc.

Implications

- Application level
  - I/O Clustering. Merging small random operations into large sequential ones.
  - A proper configuration, such as parallelism, request size, etc.
- System level
  - Simplifying the I/O stack. Moving kernel level driver to user level.
  - Replacing interrupts with polling*. More tradeoff space when storage becomes faster.

*J.Yang, D.B.Minturn,and F.Hady. When Poll is Better Than Interrupt. FAST ’12

Implications

- Application level
  - I/O Clustering. Merging small random operations into large sequential ones.
  - A proper configuration, such as parallelism, request size, etc.
- System level
  - Simplifying the I/O stack. Moving kernel level driver to user level.
  - Replacing interrupts with polling*. More tradeoff space when storage becomes faster.
- Hardware level
  - Interrupt moderation. Important for performance improvement.
  - NIC configurations

*J.Yang, D.B.Minturn,and F.Hady. When Poll is Better Than Interrupt. FAST ’12

Conclusions

- We benchmark NVMe and NVMeoF on Arm based server
  - NVMe over Fabrics only incurs minimal overhead than (Local) NVMe
  - Arm servers are powerful enough to be target(storage) for NVMeoF
- NMVeoF shows better performance than NVMe for I/O intensive applications
  - We give explanations for this phenomena
- We discuss related system implications for performance optimization
  - I/O clustering, simplifying I/O stack, interrupt moderation, etc.

Acknowledgements

- We thank the anonymous reviewers for their constructive feedback and comments
- We also thank Haresh Sakariya from Broadcom Inc. for his technical support
- This paper was partially supported by National Science Foundation under Grants CCF-1453705 and CCF-1629291

Q&A

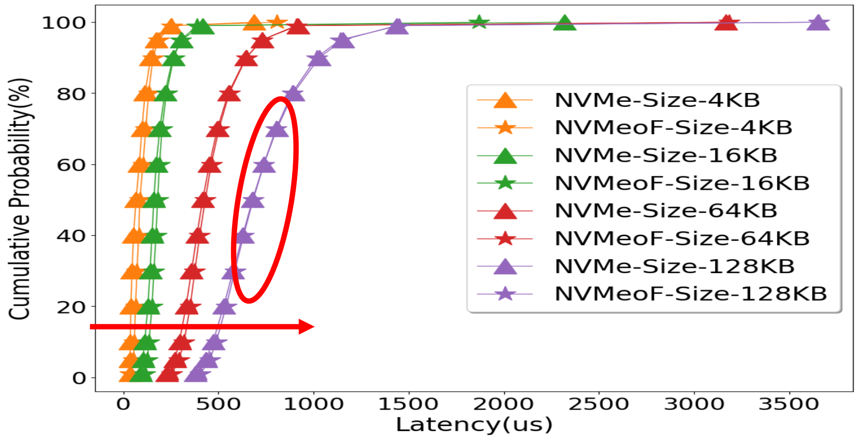
Thanks & Questions?

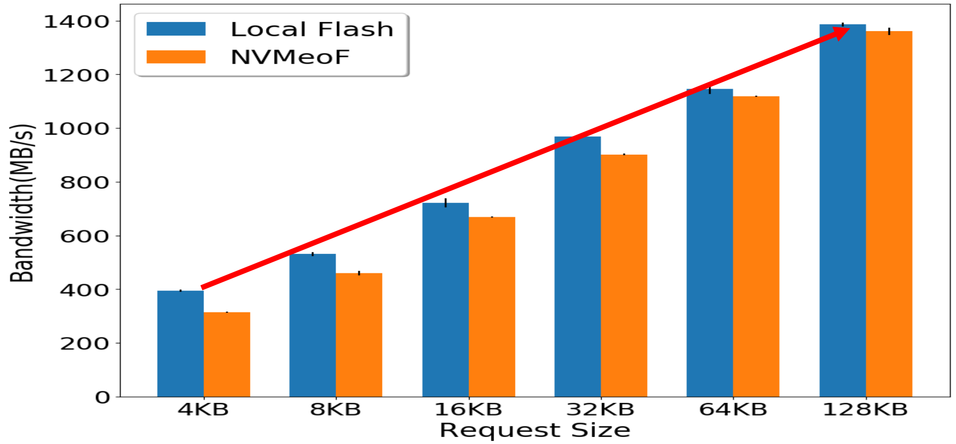
Yichen Jia

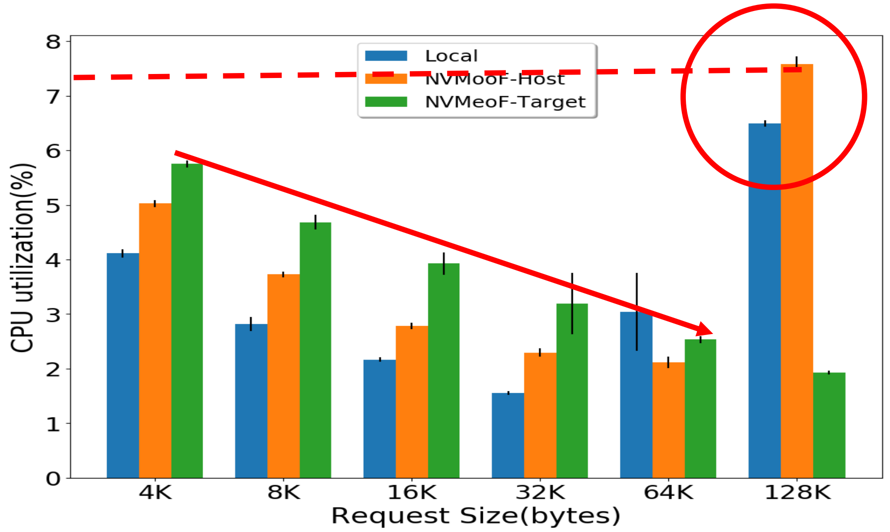
yjia@csc.lsu.edu

Backup Slides

Effect of Request Size



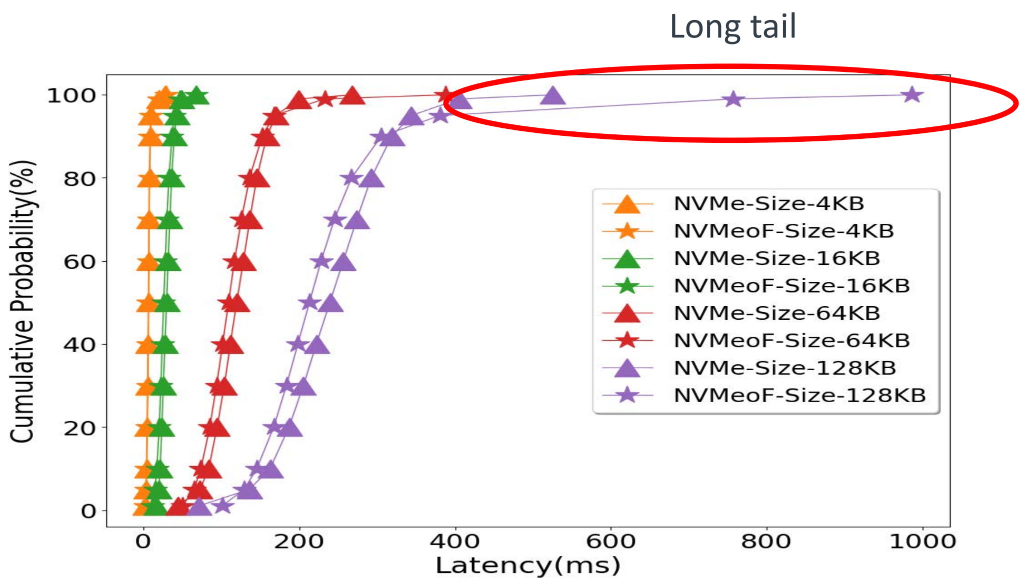


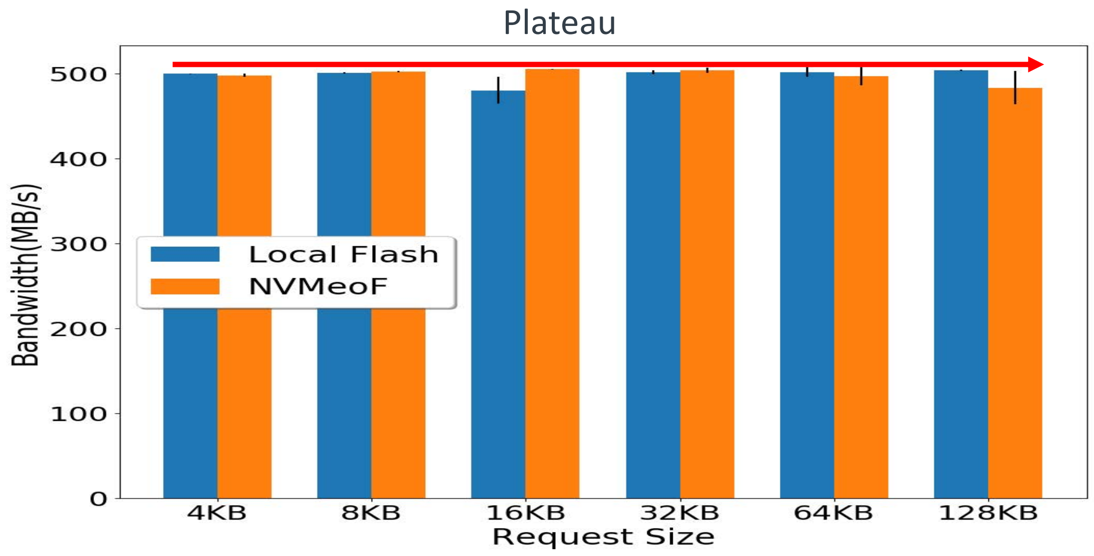


1. The latency and BW increase as req. size increases
2. Latency overhead is minimal (~2%)
3. BW overhead is at most 20%
4. CPU utilization decreases and keeps below 8% on both host and target side

*Sequential read, 4 KB -128 KB, 8 Concurrent jobs. 1 IODepth.

Computational Cost(1)

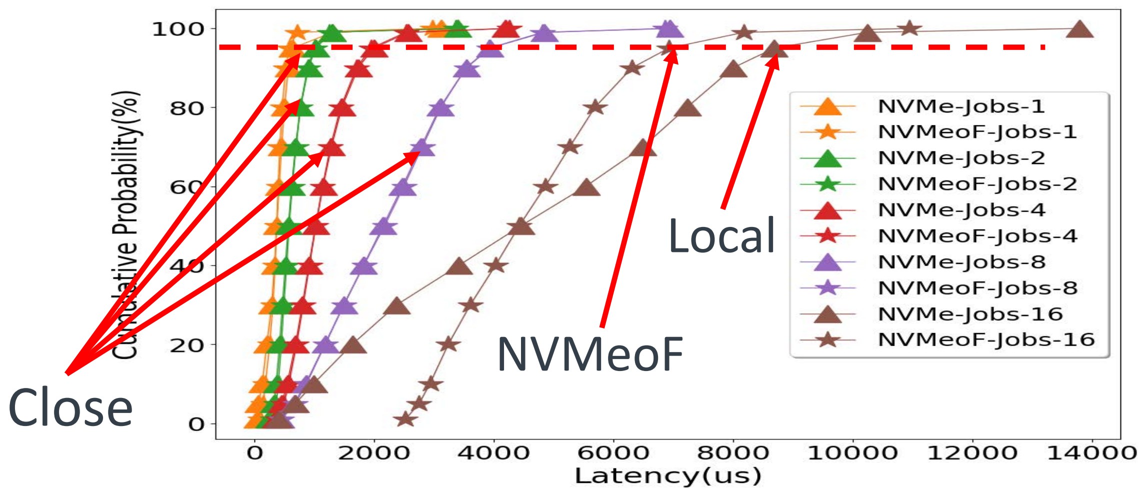


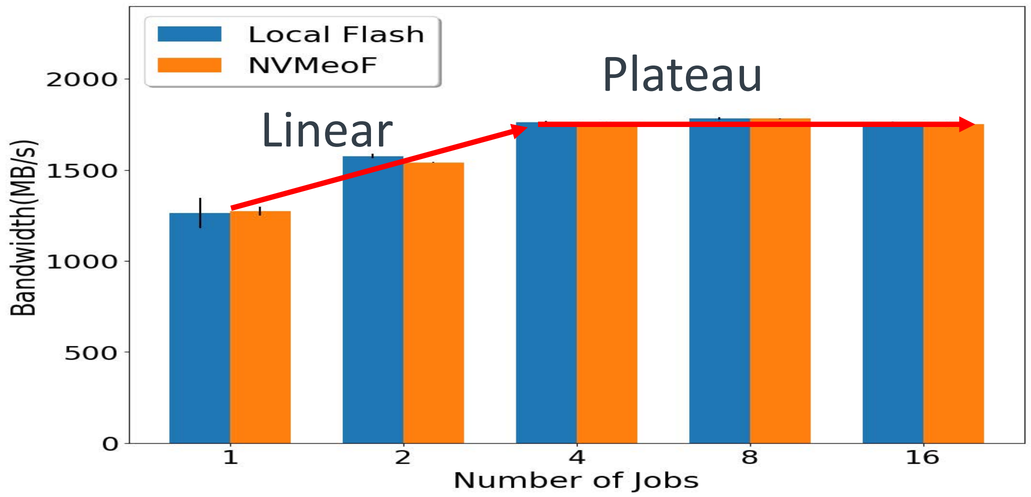


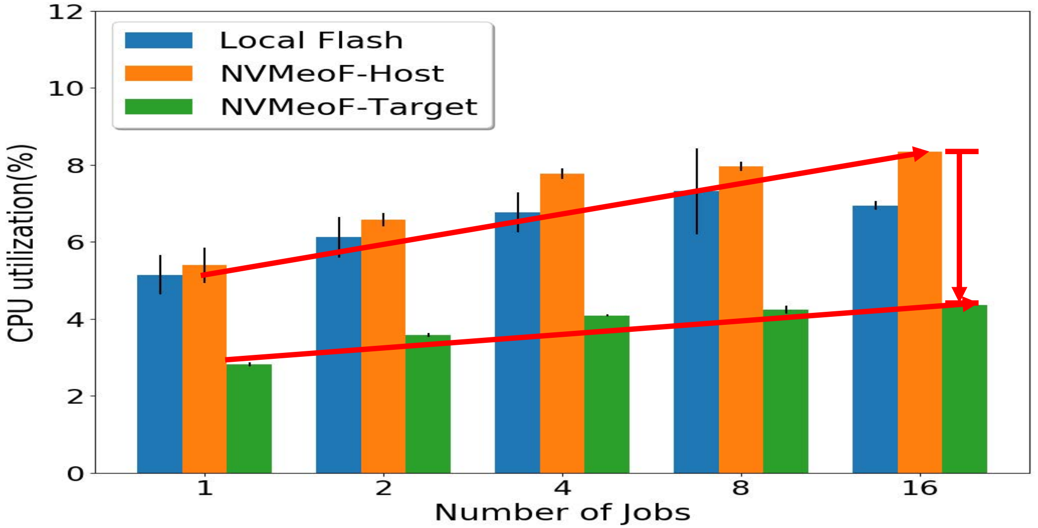
1. NVMeoF has a longer tail latency than NVMe for random writes
2. The bandwidth reaches the peak(about 500MB/s) for different request sizes

*Random write, 4 KB-128KB, 8 Concurrent Jobs, 128 IODepth

Finding #1: Effect of Parallelism



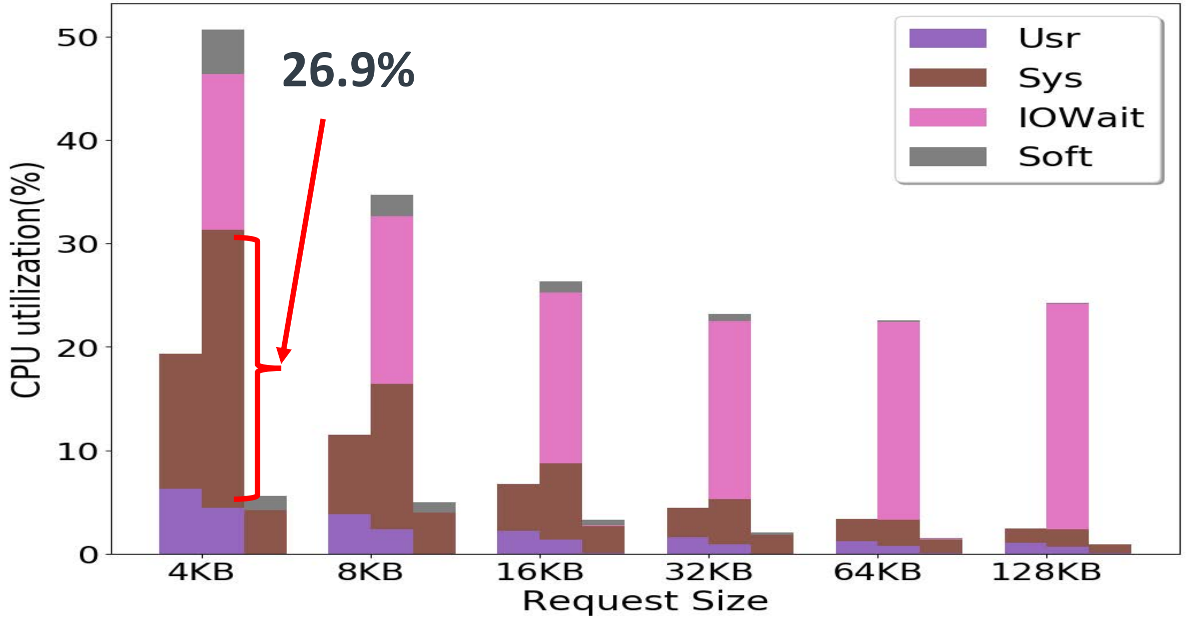




1. Latency increases as the number of jobs increases
2. NVMeoF has a close or shorter tail latency for seq read
3. BW reaches plateau when job number reaches 4
4. CPU utilization on target side is much lower
5. Arm is powerful enough to be storage server

*Sequential Read, 4KB, 128 IODepth, 1-16 Concurrent Jobs

Finding #2 : Computational Cost



1. NVMeoF consumes 31.5% more CPU on host side than local NVMe
2. Kernel level overhead is dominant(26.9%) when request size is 4KB
3. Kernel level overhead are amortized as request size increases

*Random Write, 4 KB-128KB, 8 Concurrent jobs, 128 IODepth

Interrupt Moderation

- Interrupt moderation means multiple packets are handled for each interrupt
- Overall interrupt-processing efficiency is improved and CPU utilization is decreased