STORAGE DEVELOPER CONFERENCE
SDC 22
Fremont, CA
September 12-15, 2022
BY Developers FOR Developers

A SNIA Event

The Challenges in Creating a Clustered Software Defined Fileserver From **"Scratch"** on HCI

5 Years of Trials, Tribulations, and Triumph

Presented by Will Strickland, Staff Engineer, Nutanix, Inc.
Dan Chilton, Staff Engineer, Nutanix, Inc.

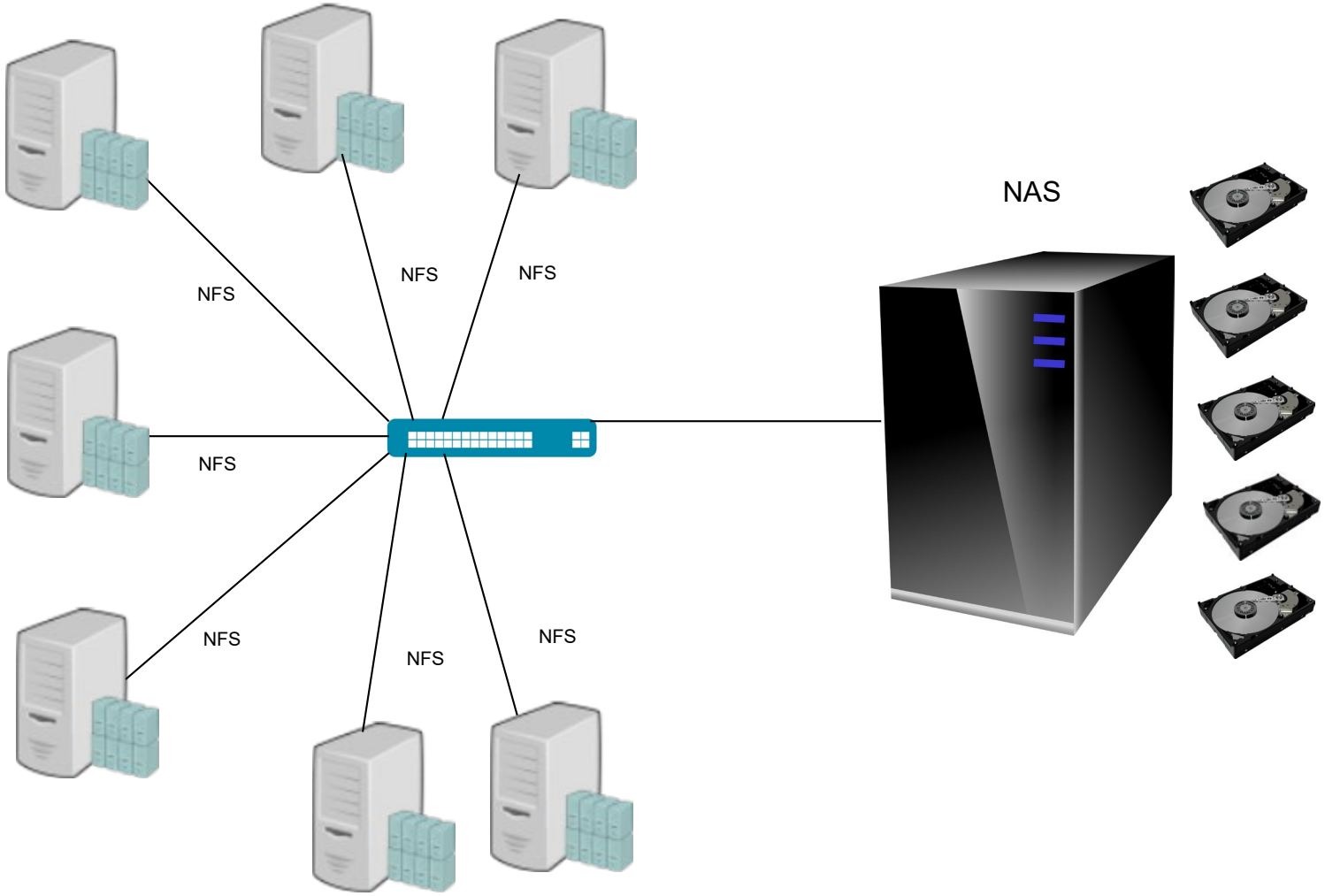
Agenda

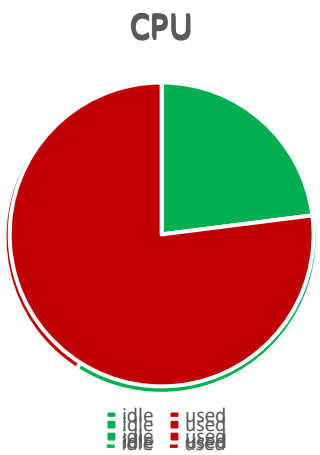
- Overview of a traditional NFS environment
- Nutanix architecture from the ground up
- So how do you build a fileserver from "scratch" on HCI?
- A few things we found to be true
- Performance
  - SMB
  - NFS
  - MinervaFS
- Q&A

Traditional NFS Architecture

Hypervisors connecting to a central NAS

Traditional Hypervisor storage

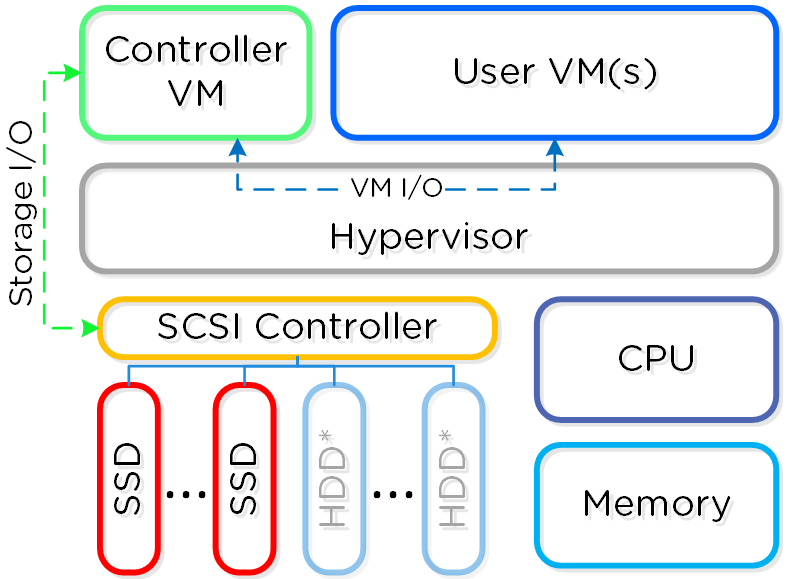




What is a "hyper-converged" solution?

A hyper-converged platform must:

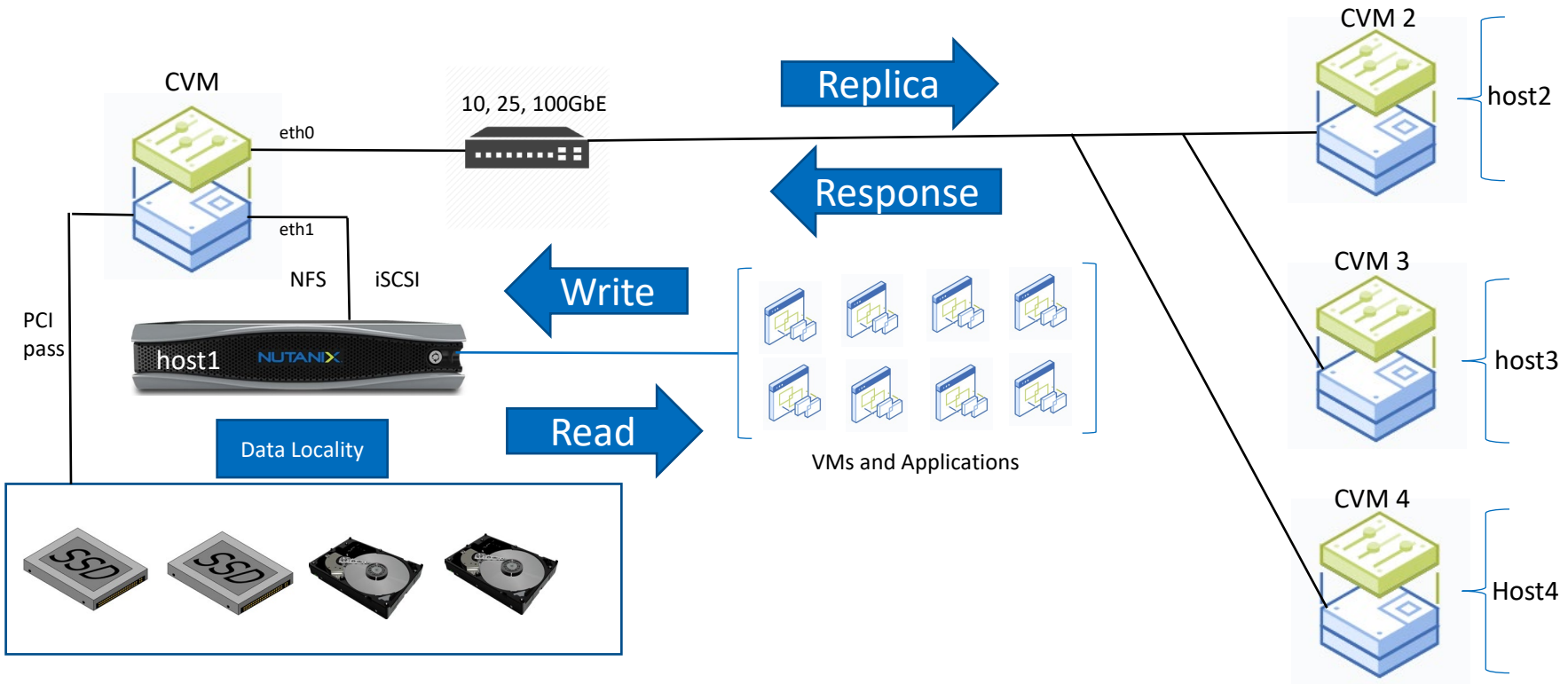
- Converge and collapse the entire stack (compute + storage + network)
- Distribute data and services across nodes in the system
- Appear and provide the same capabilities as centralized storage



*All flash nodes will only have SSD devices

As shown in the Nutanix Bible: https://www.nutanixbible.com/

HCI Cloud Architecture

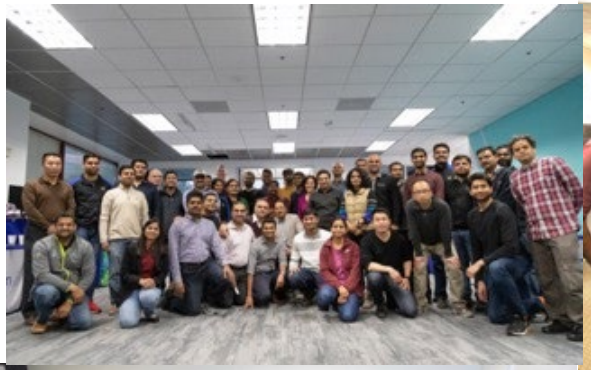
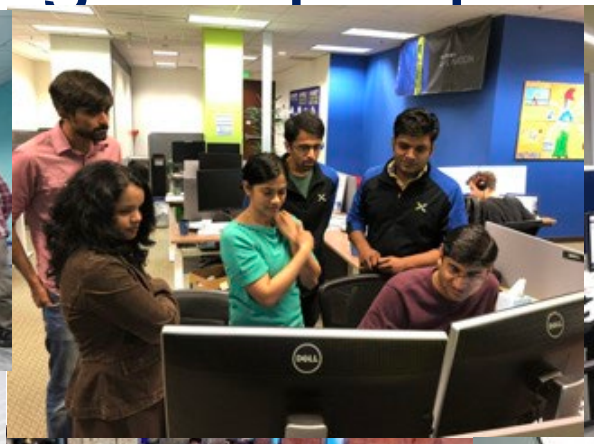


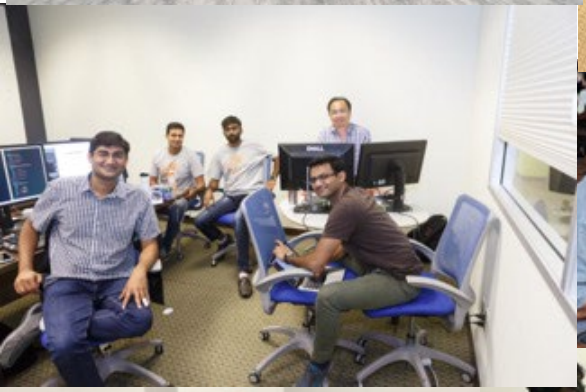
Why was Nutanix Files needed?

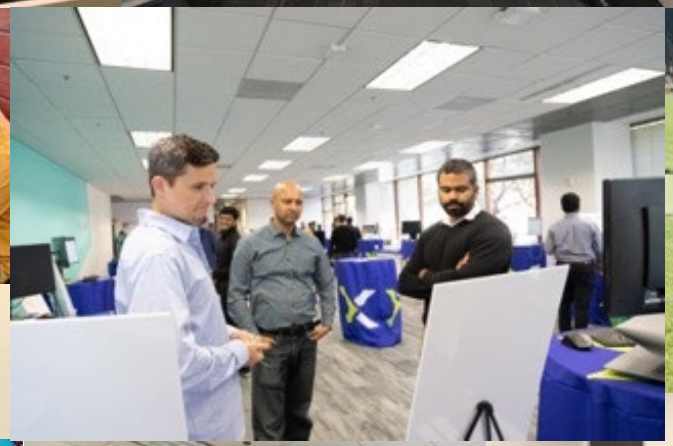
- Hypervisors tend to access very large files and are mostly performing NFS reads or writes.
- Our original storage layer was streamlined to handle hypervisor workloads efficiently
- Files allows us to converge the stack even further. Now shared storage, users, and applications can be housed on the same private cloud.

So how do you build a fileserver from "scratch" on an HCI cloud platform?

It starts with great people



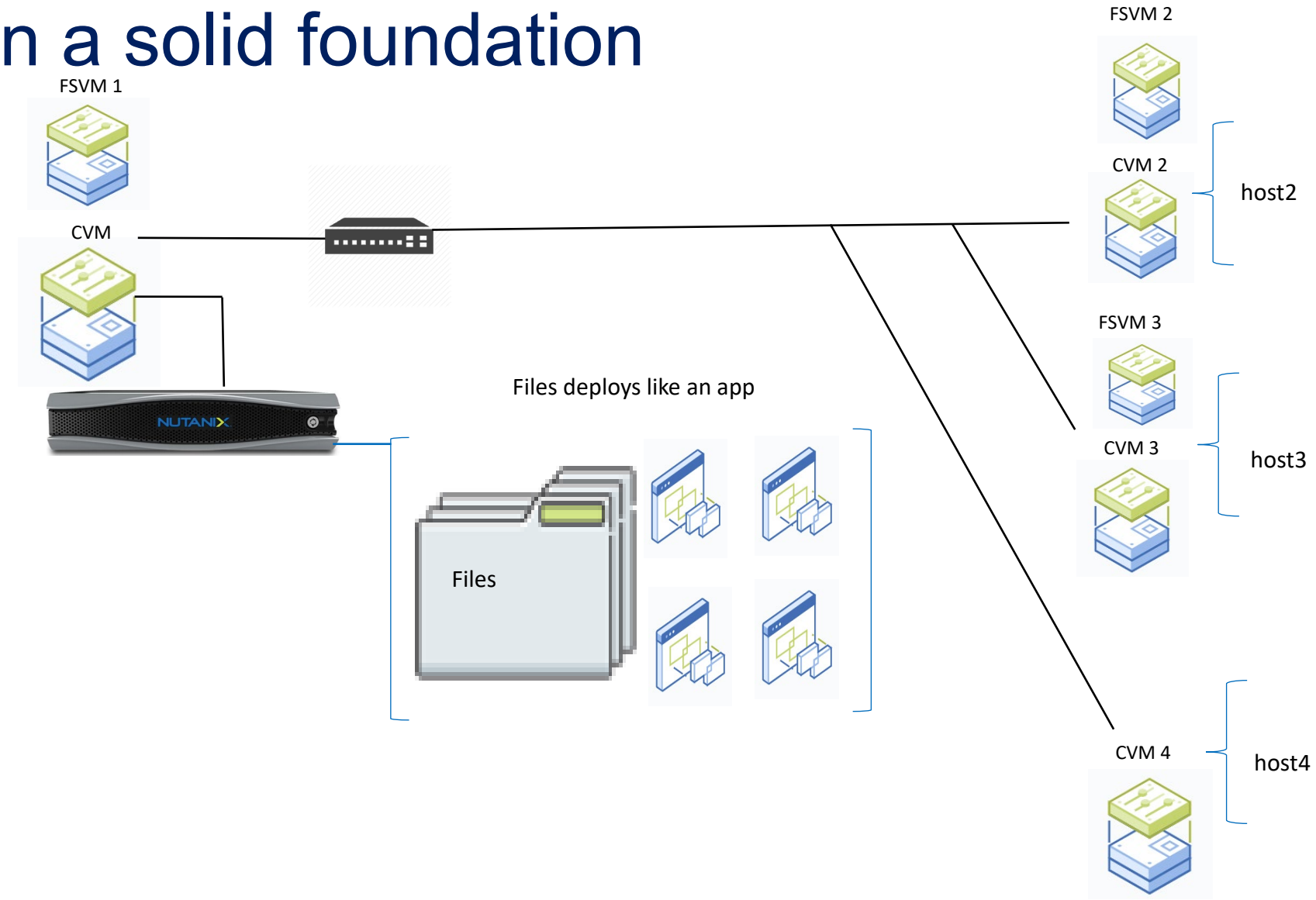




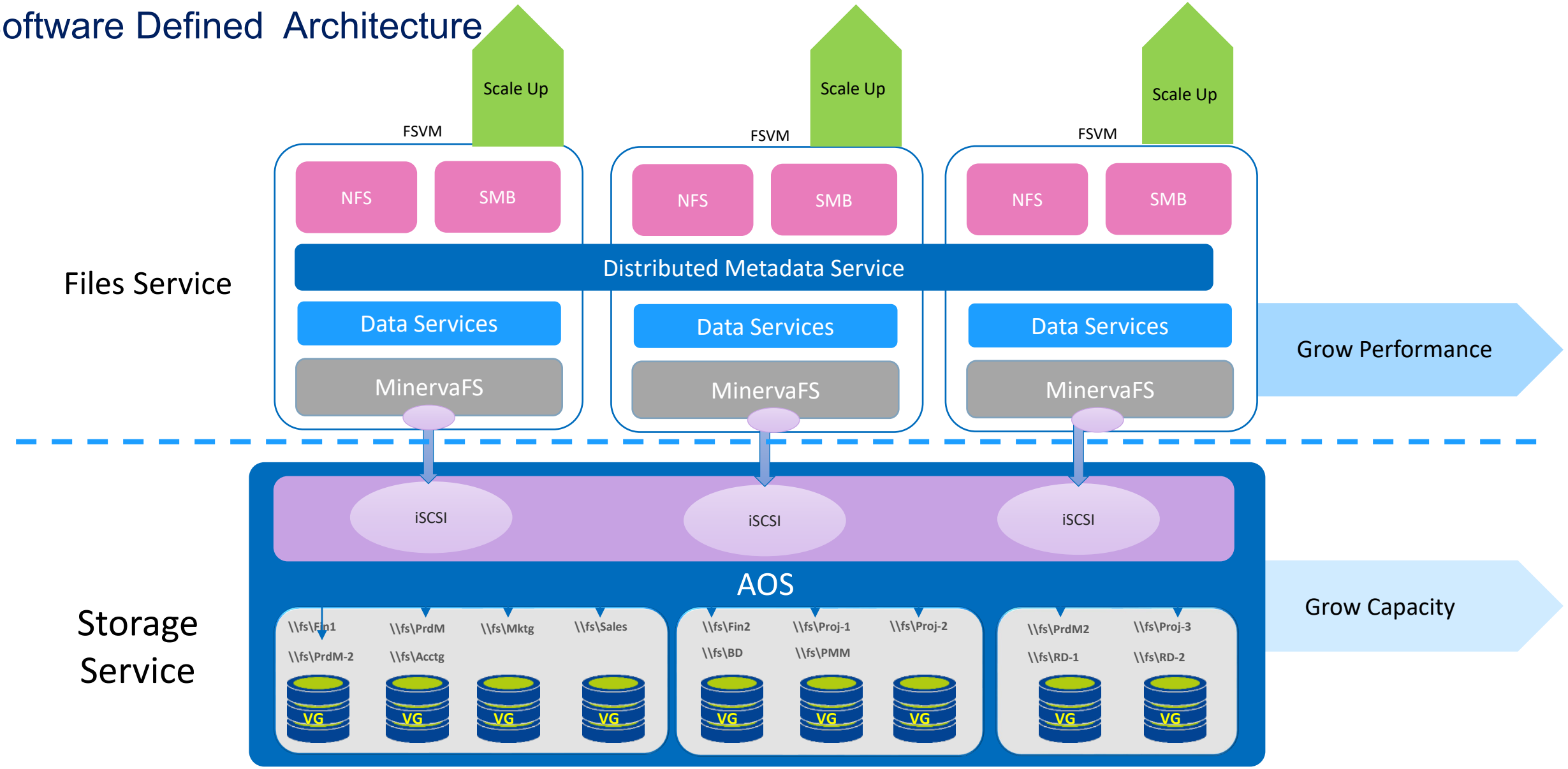



It leverages open source components

It's built on a solid foundation



Software Defined Architecture



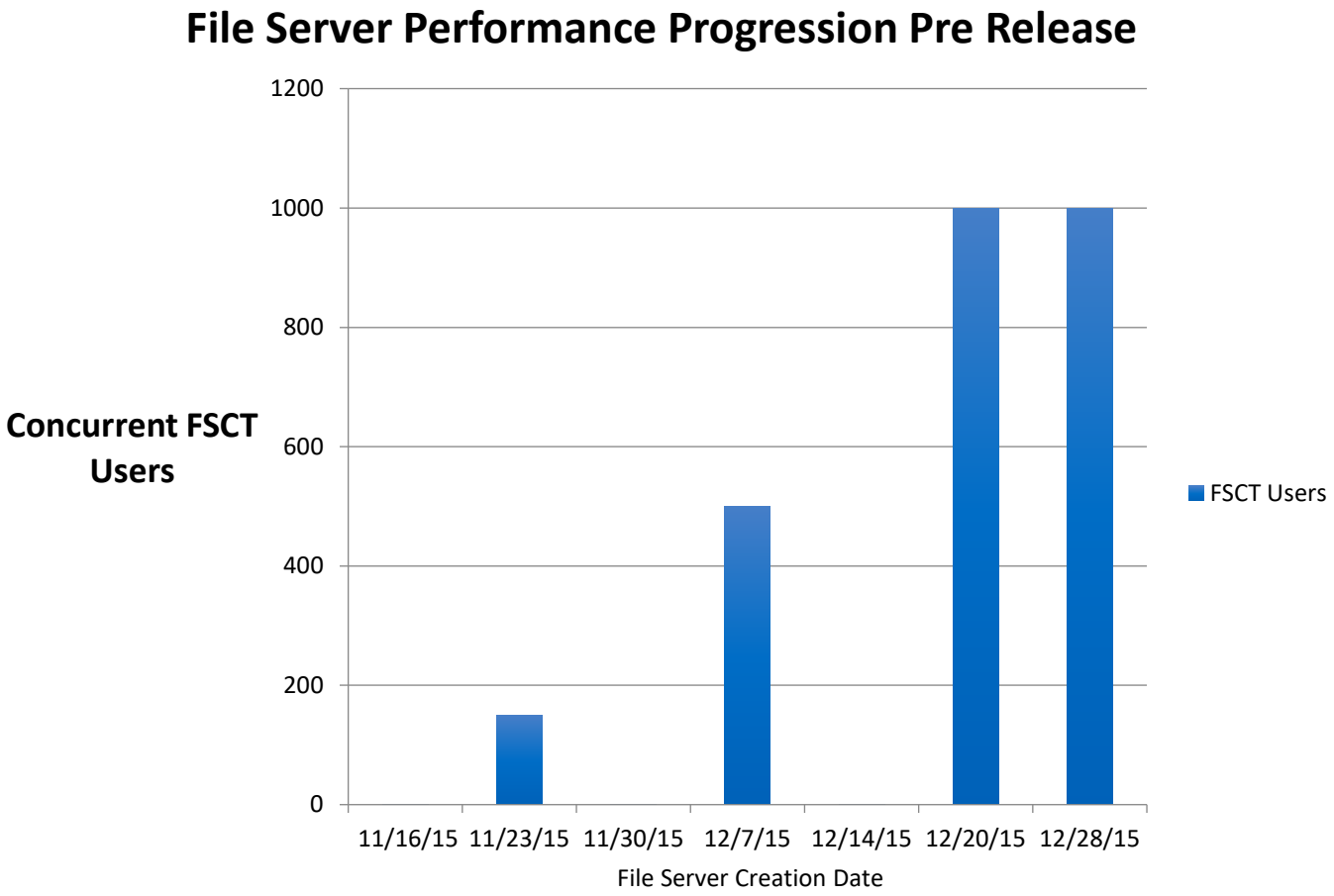
Performance

Performance Foundations

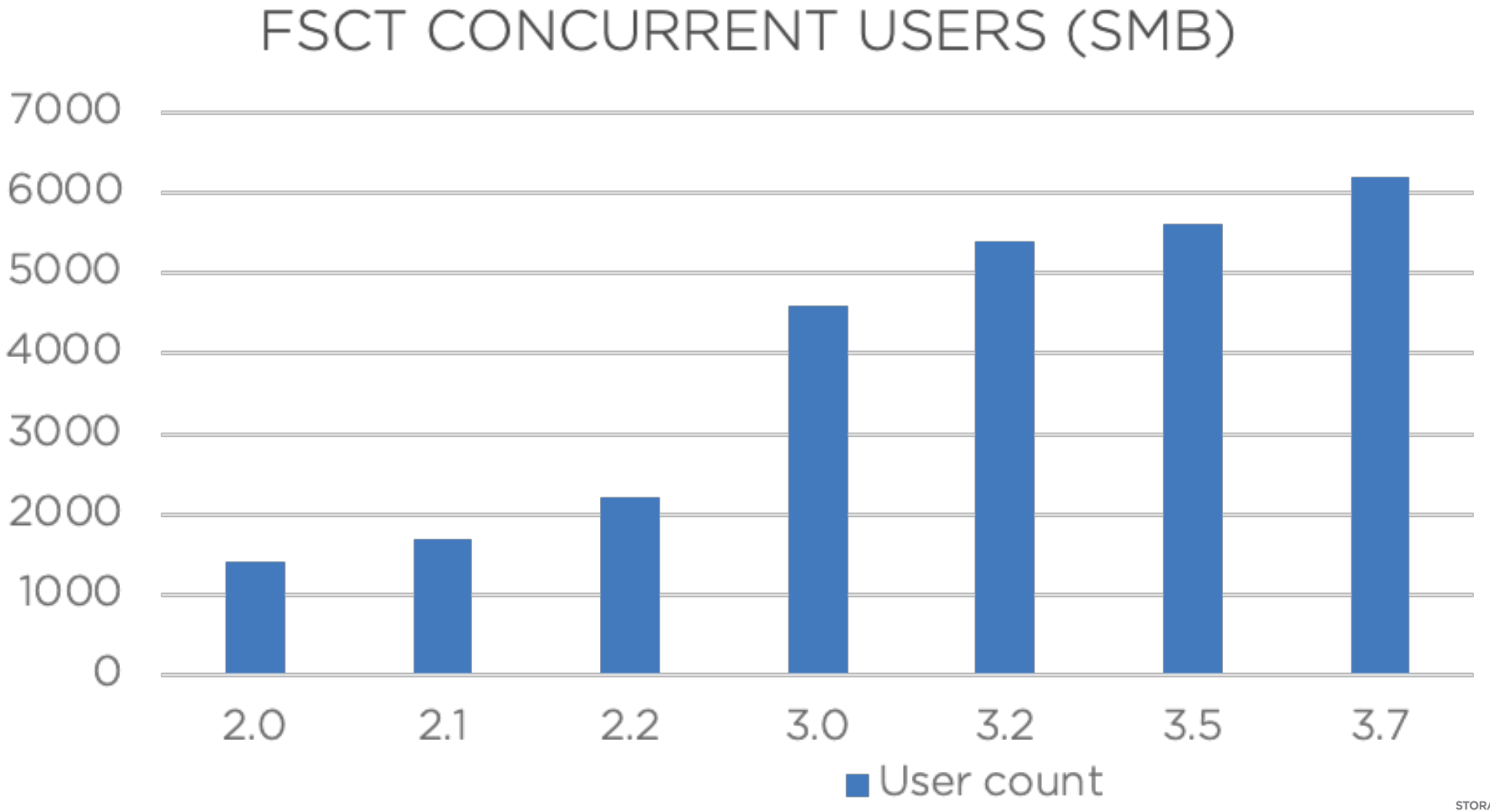
Crawl, Walk, Run

Crawl: The first step is always the hardest

- First target use cases for file server
  - Home directories
  - Departmental Shares
  - VDI User Profiles
- Small CPU and memory configuration
- Test performance to closely simulate first use case –home directories
- Test suite - Microsoft FSCT (File Services Capacity Tool)

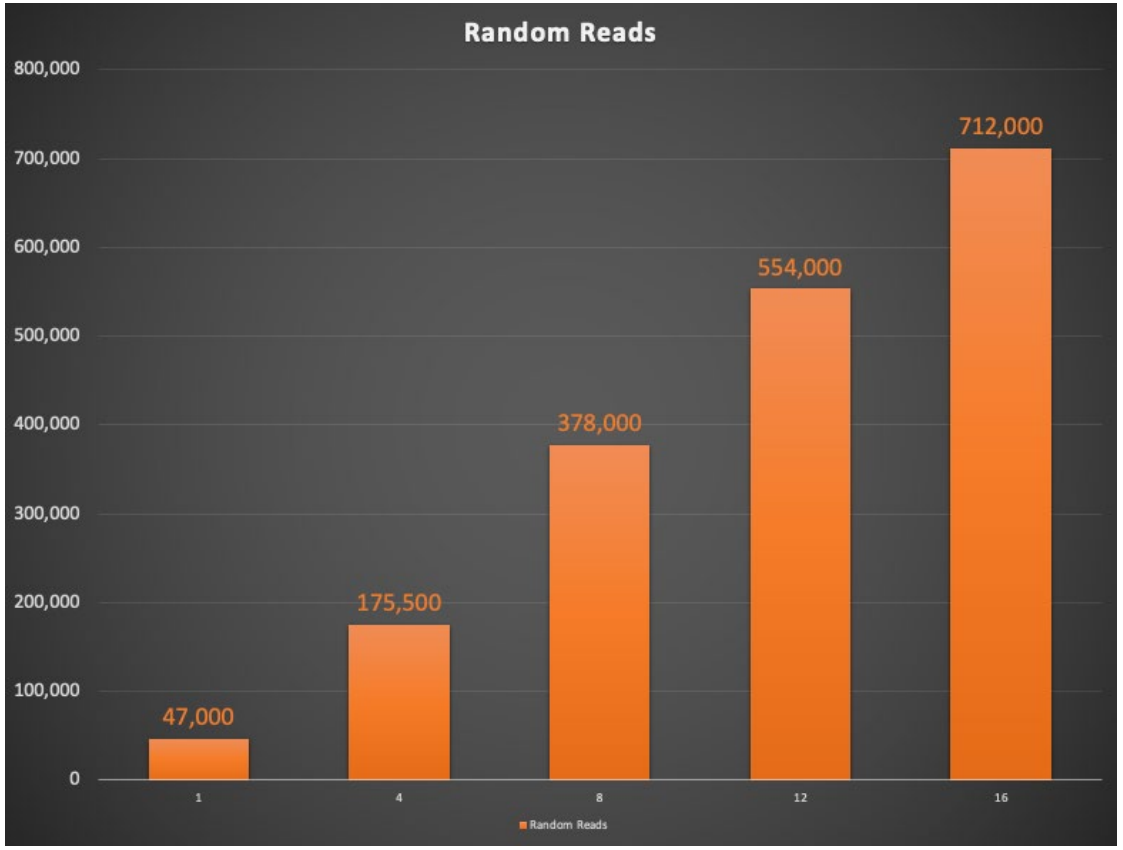


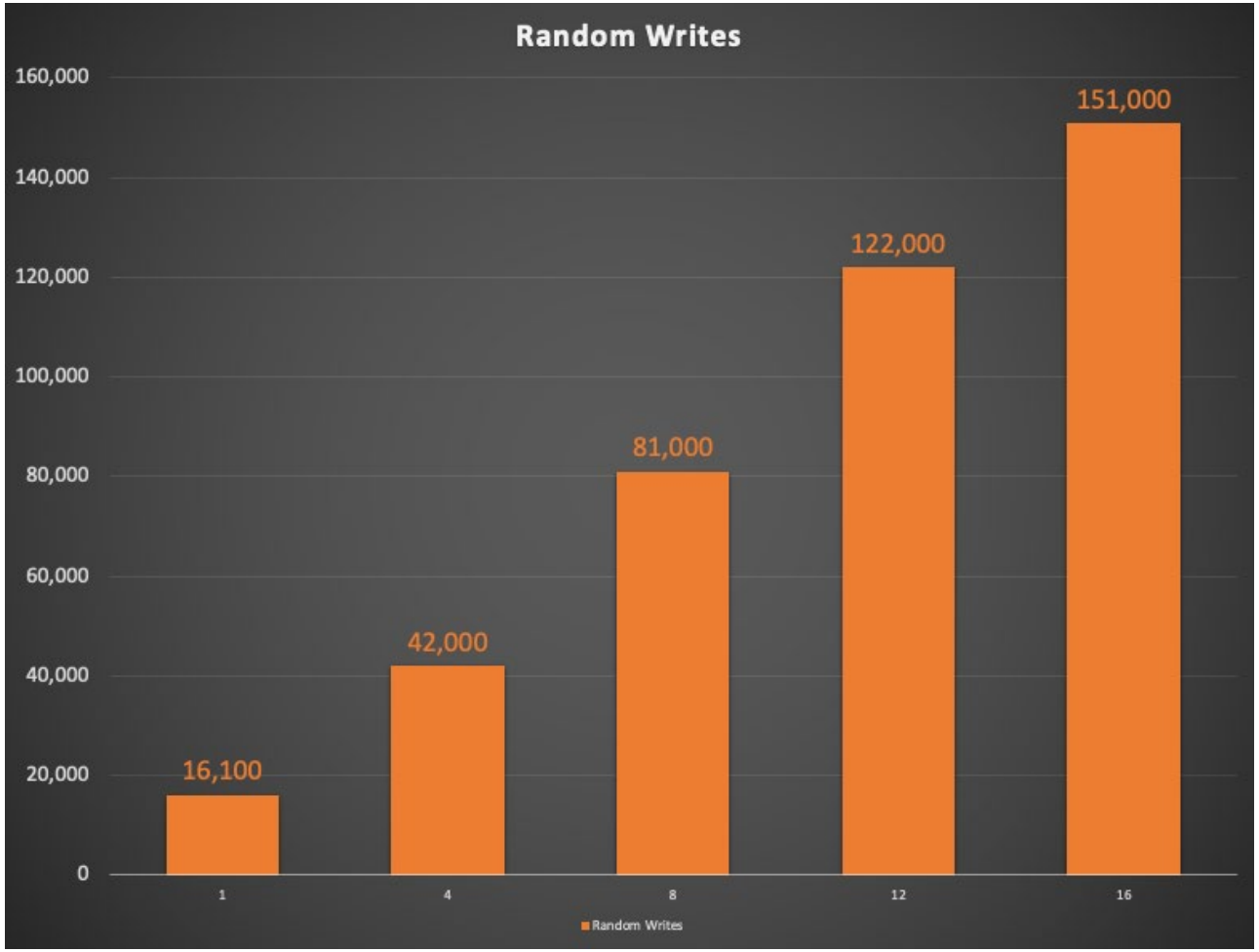
Walk



Run:

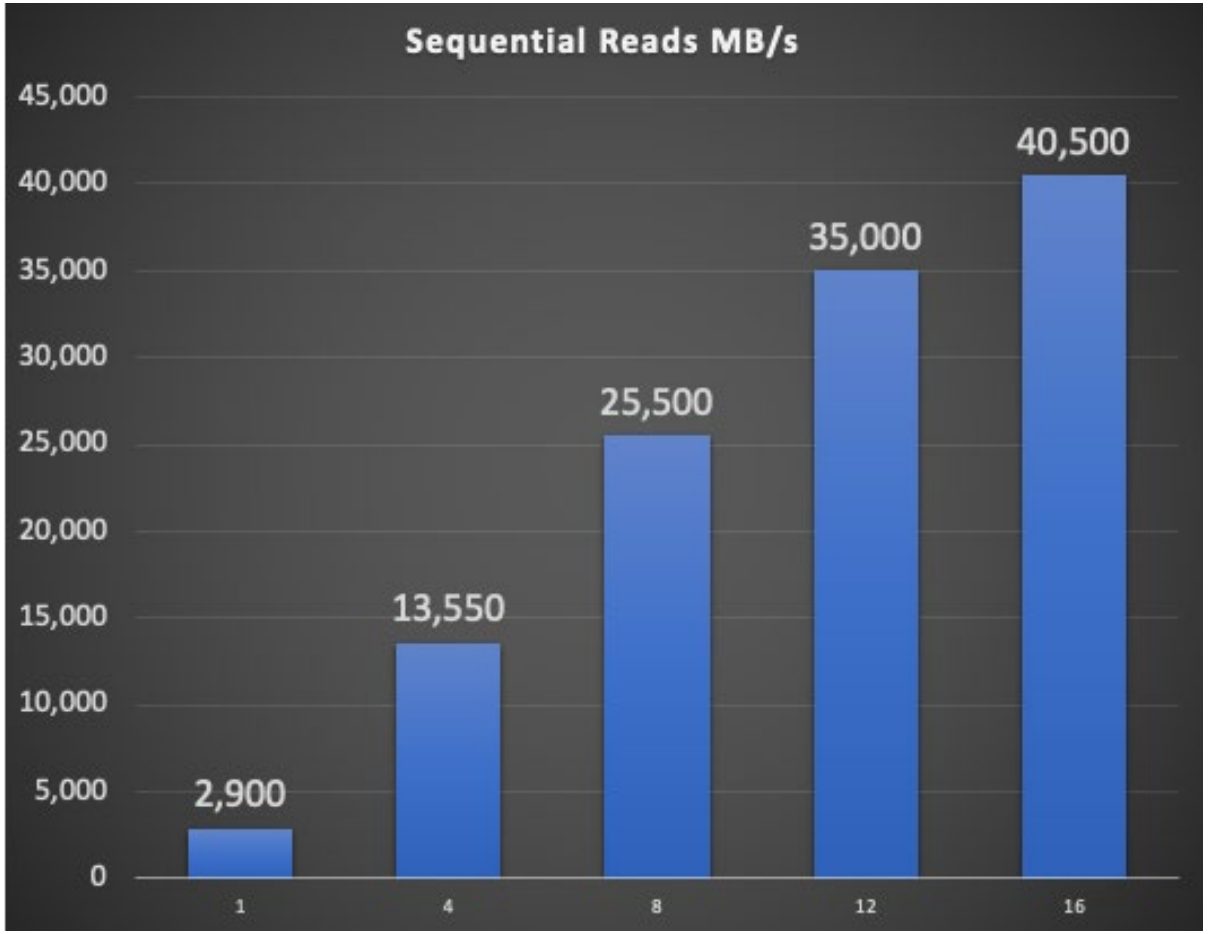
8K Random IOPS scales from 1 to 16 FSVMs

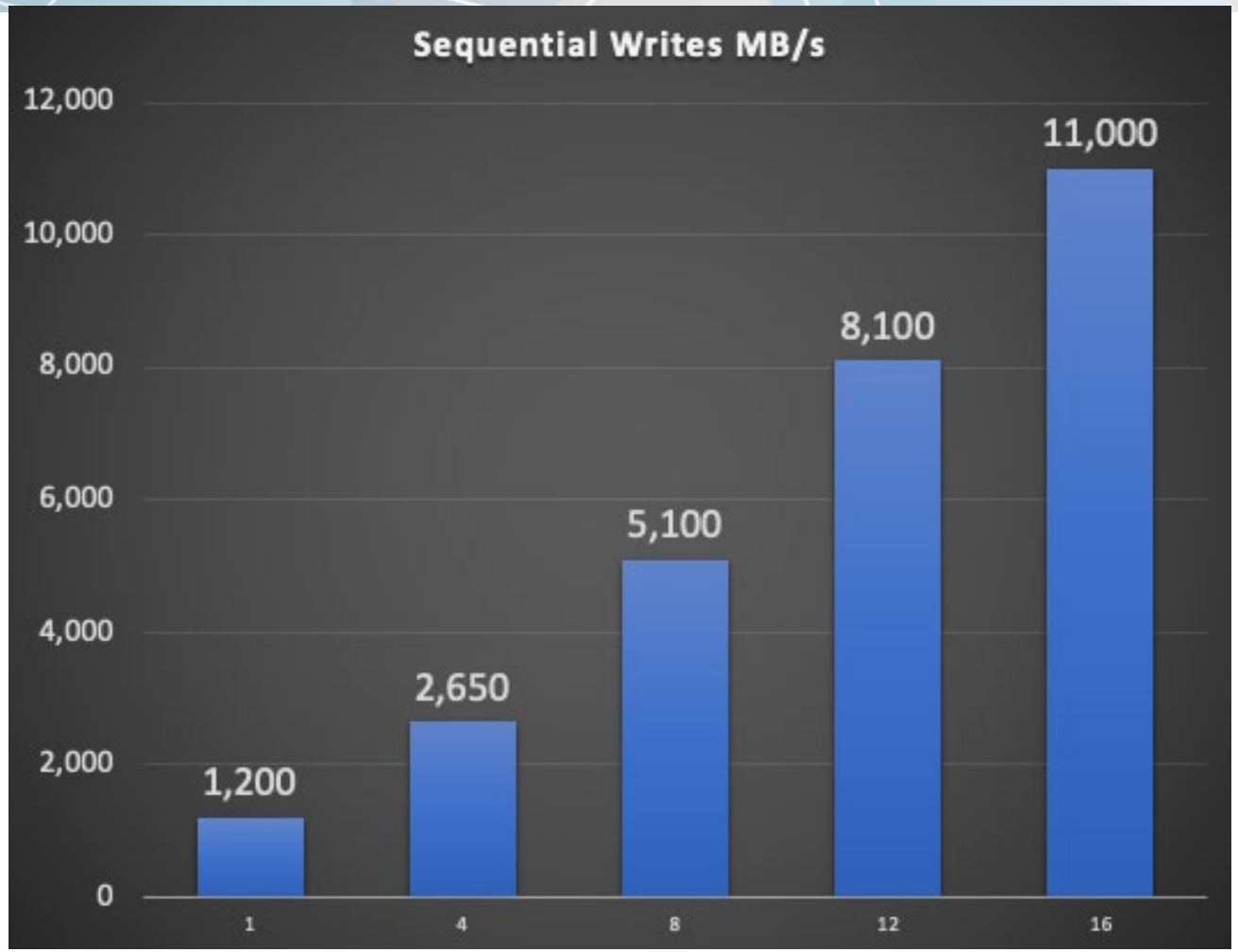




Run:

Throughput scales from 1 to 16 FSVMs





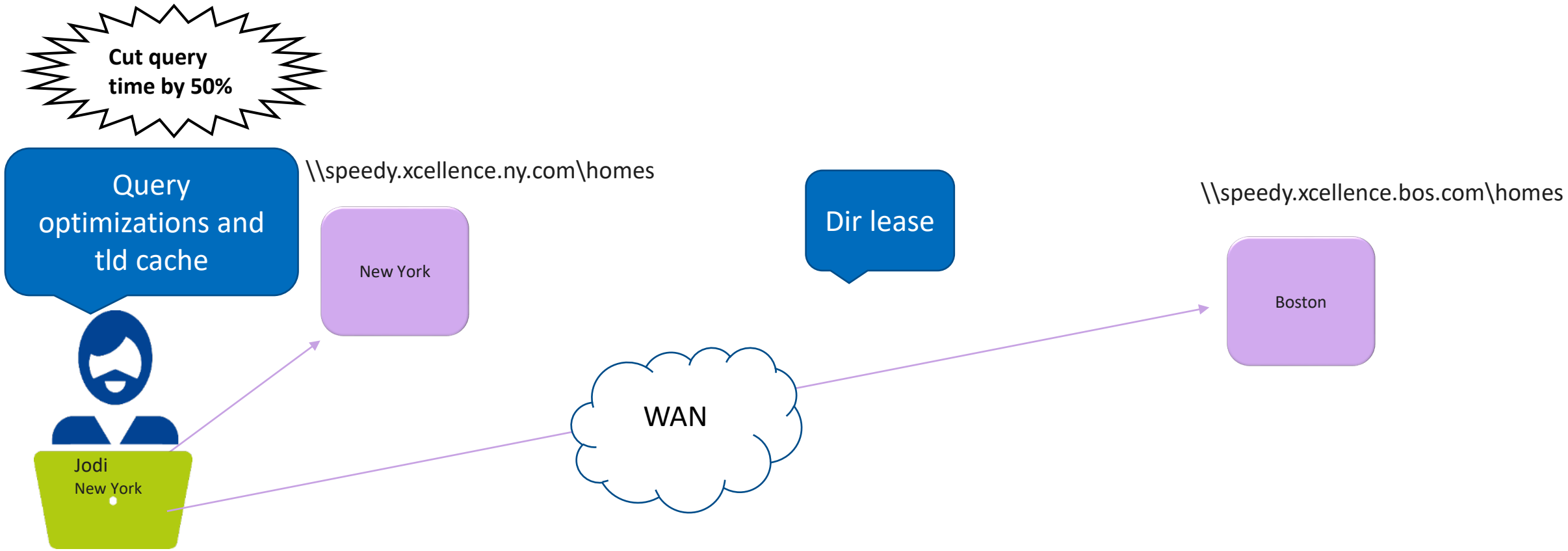
Truths

- Hardware matters
  - CPU count and speed
  - Hyperthreading
  - NUMA tuning
- Network matters (a lot)
  - 25 GbE, 40 GbE, 100 GbE
  - CVM RDMA
- Storage device (speeds) matter
  - Blockstore
  - SPDK

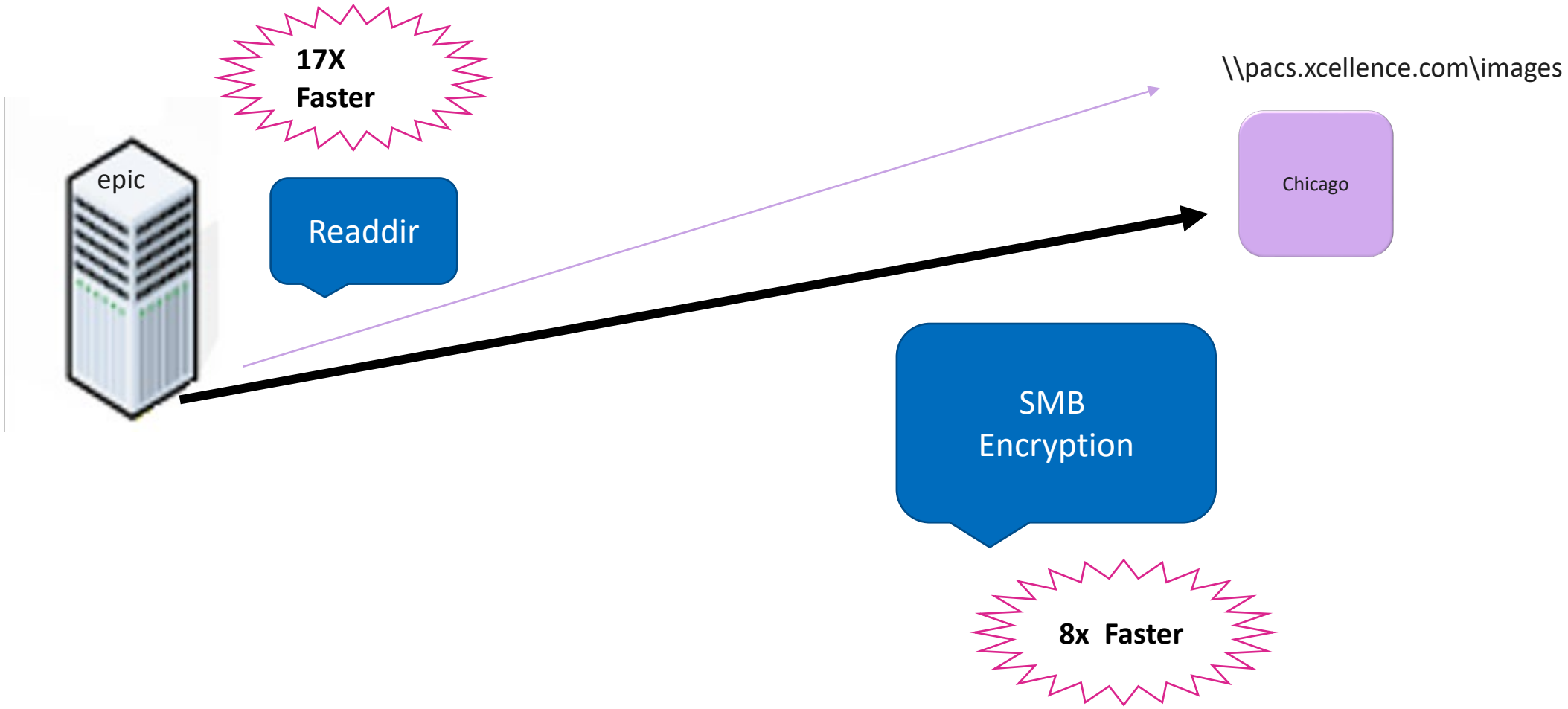
SMB Performance

Challenges and Opportunities

Challenge: Profile access speed local and remote



Challenge: Image file retrieval time and security



NFS Performance

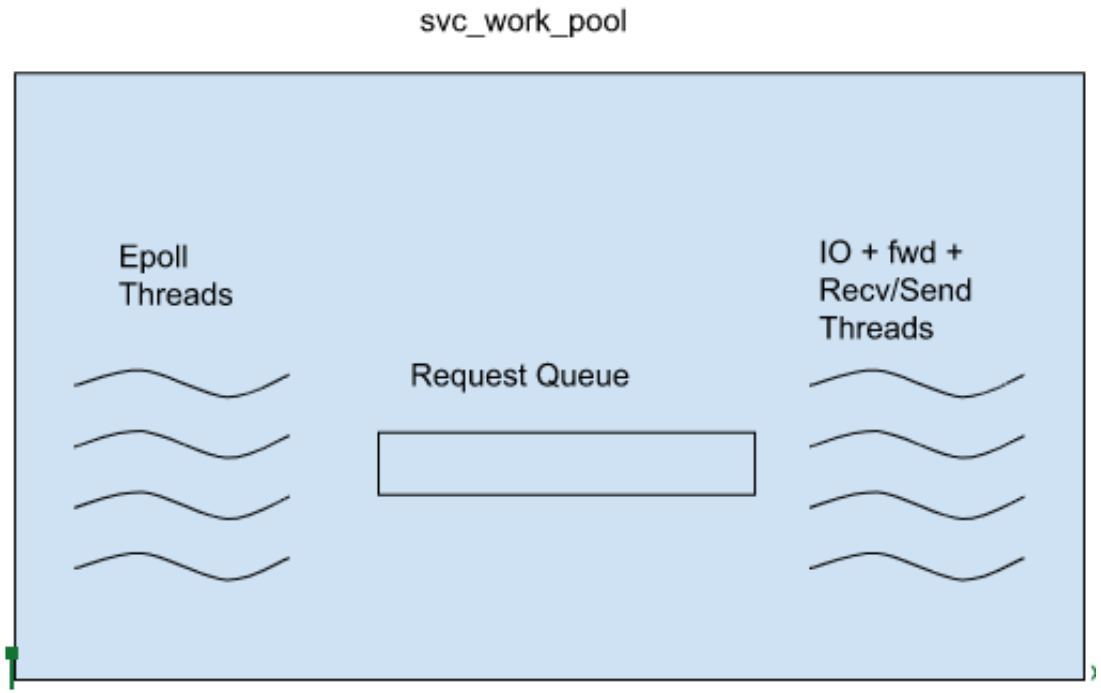
Challenges and Testing Best Practices

Challenge: NFS cached read performance

- Small cached random read IOPs seemed to be "stuck"
- No CPU bottleneck, no network or storage consumption issues.
- Bypassing the storage layer showed no change.

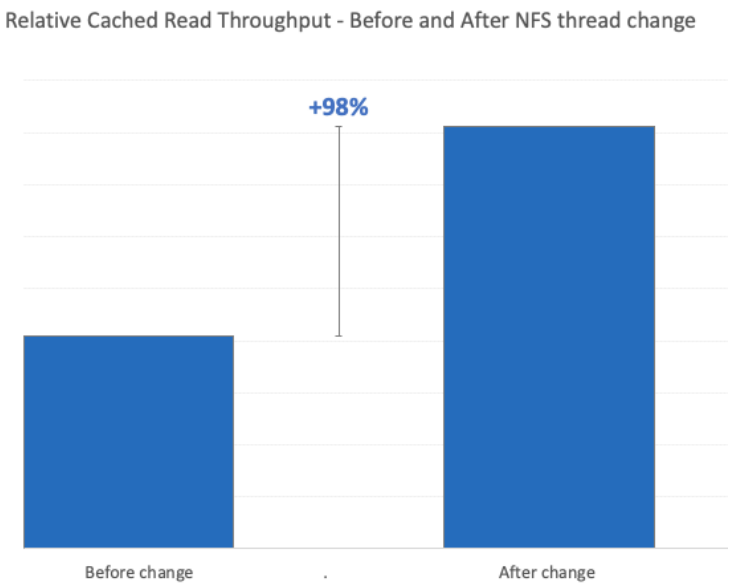
Challenge: NFS threading

- Original model: Single request queue and thread pool
- Required a lot of context switching and caused lock contention



Challenge: How we resolved it

- NFS I/O thread efficiency
  - Depending on the latency of the operation, dynamically set the max number of threads
  - Maintain different thread pools depending on the nature of the workload
    - Sequential vs. Random



Challenge: Inode and directory caching

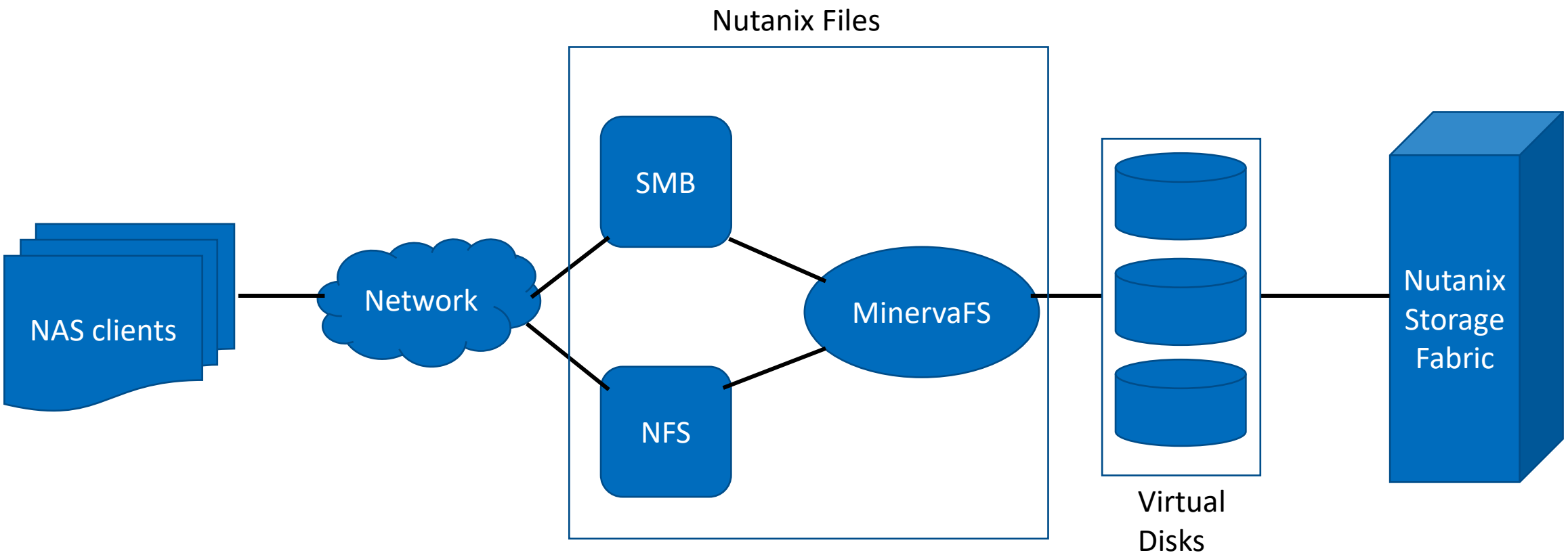
- Our NFS server implements inode caching for efficient access of files/dirs. This is in addition to data caching.

- The amount of cache allocated to inodes vs. data has been optimized
  - The initial number of inode cache entries was capped at 100k
  - Testing determined that as the RAM increases, inode cache should increase
  - The number of entries is now based on how much RAM the fileserver has.

Filesystem Performance

(and some more challenges and improvements)

How does MinervaFS fit into this picture?



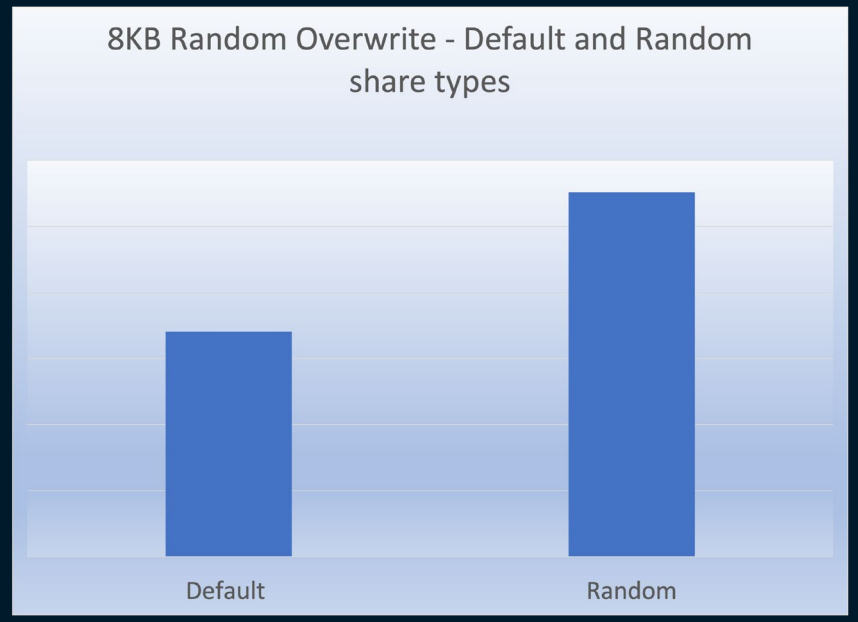
MinervaFS - Filesystem Optimizations

- Performance
  - Separation of metadata and data
  - Deduplication of ACLs
  - Record size – Share types
    - Default - 64KB
    - Random - 16KB
    - Sequential - 1MB

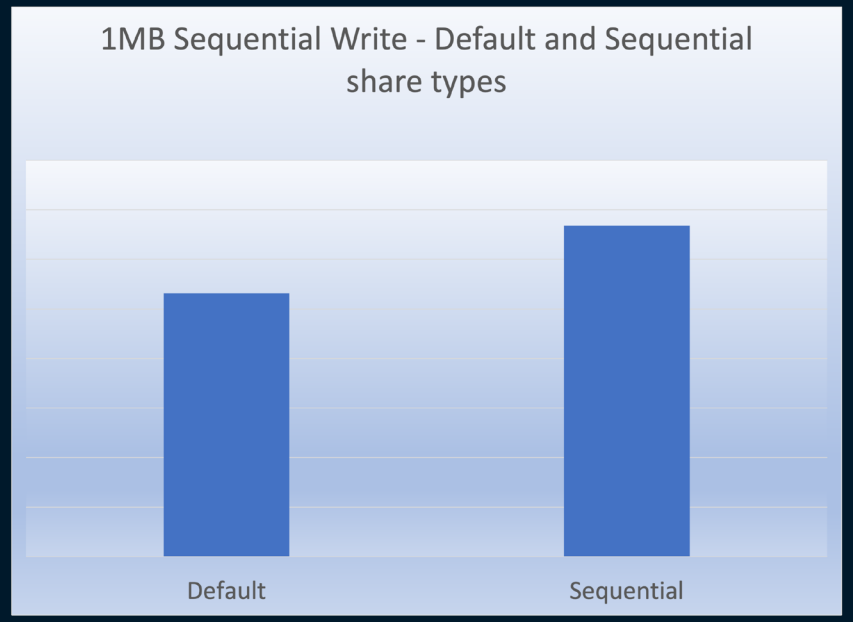
MinervaFS – Filesystem Optimizations

- Share Type performance increases:

+ 60%



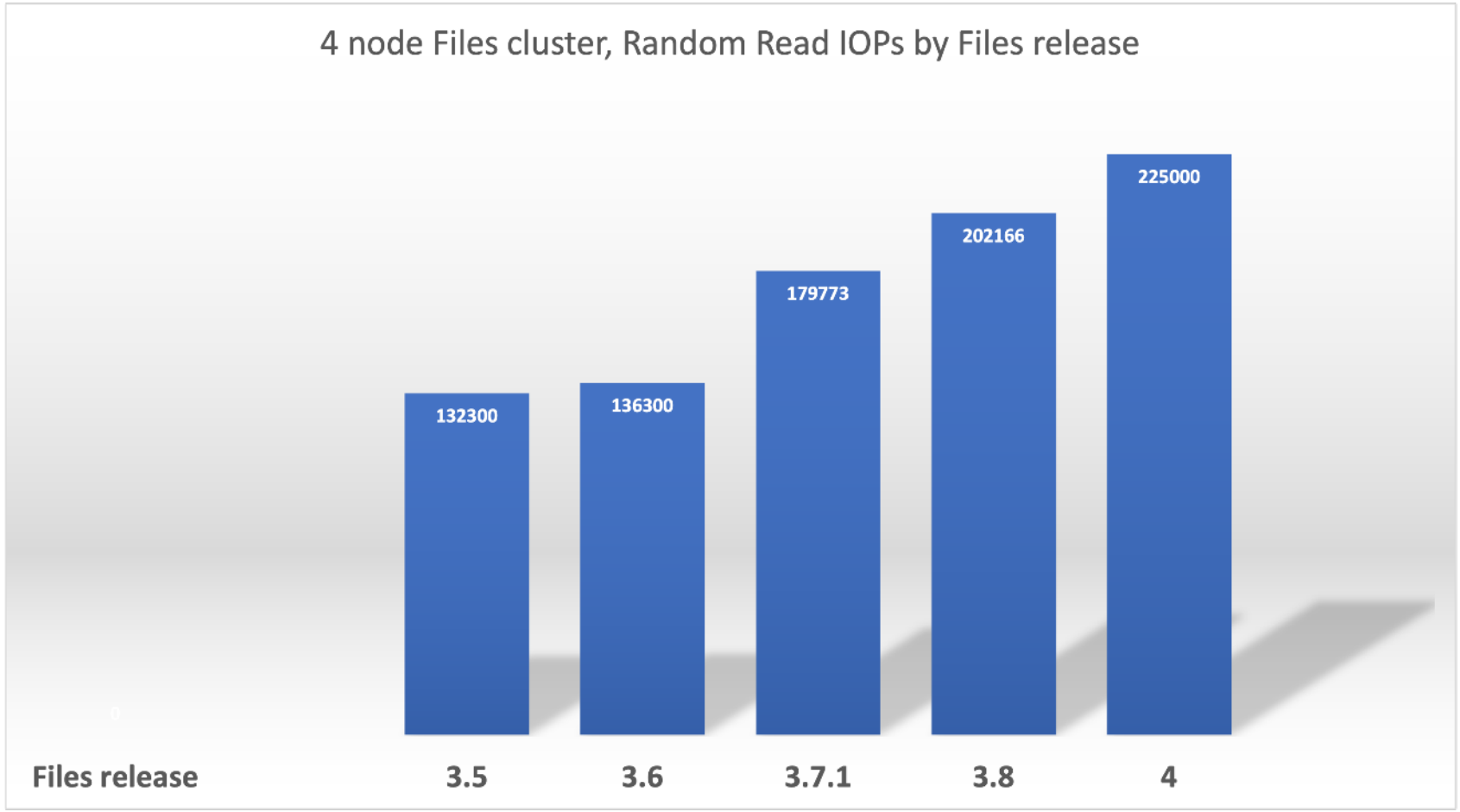
+ 20%



Filesystem – Challenges along the way

- Performance
  - vdev aggregation – improvement to sequential write performance
  - Sync Writes – SLOG device speeds up random writes
- Stability
  - Flow control – Needed for slower hardware, no regression on fast hardware

Where we've been, where we're going



So How do you build a fileserver from "scratch" on HCI?

- Great People
- Open Source components
- A solid Foundation (Nutanix Distributed Storage Fabric)
- Most Importantly: Our Customers

Please take a moment to rate this session.

Your feedback is important to us.

Reach us

- Dan Chilton Daniel.Chilton@Nutanix.com
- Will Strickland Will.Strickland@Nutanix.com

