IBM Deep Computing

How to Build a Petabyte Sized Storage System

Invited Talk for LISA'09

Ray Paden
raypaden@us.ibm.com

Version 2.0 (alternate)
4 Nov 09

A Familiar Story When Building PB Sized Storage Systems

- Center manager is negotiating with vendor for updated system
- Focused attention given to
  - CPU architecture
  - Memory architecture
  - Bus architecture
  - Network topology and technology
  - Linpack performance
  - Qualifying for Top 500
  - Power and cooling
- Oh, almost forget storage…
  - “Give me what I had, only more of it.”
- System performance is compromised by inadequate storage I/O bandwidth

Storage Capacity, Performance Increases over Time

- 1965
  - Capacity < 205 MB
  - Streaming data rate < 2 MB/s (26 platters laterally mounted)
  - Rotational speed = 1200 RPM
- 1987
  - Capacity < 1.2 GB
  - Streaming data rate < 3 MB/s (2 spindles)
  - Rotational speed = 3600 RPM
  - Average seek time = 12 ms
- 1996
  - Capacity < 9 GB
  - Streaming data rate < 21 MB/s
  - Rotational speed = 10 Krpm
  - Average seek time = 7.7 ms

- 2008
  - SATA
    - Capacity < 1000 GB
    - Streaming data rate < 105 MB/s
    - Rotational speed = 7200 RPM
    - Average seek time = 9 ms
  - Fibre Channel
    - Capacity < 450 GB
    - Streaming data rate < 425 MB/s
    - Rotational speed = 15 Krpm
    - Average seek time = 3.6 ms

Planning for the System Upgrade

- System administrators are generally responsible for “operationalizing” system upgrades.
- The following pages provide some common and some not so common cases of processing centers scaling to the PB range.

Common Scenario #1

- Juan currently manages a small cluster
  - 64 Linux nodes with SAN attached storage
  - Storage = 25 TB (64 x 146 GB FC disks + 64 x 300 GB FC disks)
- Juan’s new cluster will be much larger
  - 256 Linux nodes with future upgrades up to 512 Linux nodes
  - Raw capacity starting at 200 TB increasing up to 0.5 PB

Common Scenario #2

- Soo Jin’s company has a variety of computer systems that are independently managed
  - Modest cluster of 128 Linux nodes with a clustered file system
  - Several smaller clusters consisting of 16 to 64 Linux or Windows nodes accessing storage via NFS or CIFS
  - Several SMP systems with SAN attached storage
  - 2 types of storage
    - FC and SAS disk: 100 TB
    - SATA: 150 TB
- Soo Jin has been asked to consolidate and expand the company’s computer resources into a new system configured as a cluster
  - 512 Linux nodes with future upgrades up to 1024 Linux nodes
    - No more SMP systems
  - Raw disk capacity starting at 0.5 TB increasing up to 1 PB
  - Must provide tape archive

Common Scenario #3

- Lynn manages a small cluster with a large storage capacity
  - Small cluster of 32 nodes (mixture of Linux and Windows)
  - All storage is SAN attached
  - 3 classes of storage
    - FC disk ~= 75 TB (256 disks behind 4 controllers)
    - SATA disk ~= 360 TB (720 disks behind 3 controllers)
    - Tape archive approaching 1 PB
- Lynn’s new system will double every 18 months for the next 5 years with similar usage patterns
- With the next upgrade, Lynn’s storage must be more easily accessible to other departments and *vice-verse*; currently files are exchanged using ftp, scp or exchanging tape cartridges. One department has a cluster consisting of 256 Linux nodes.

Not as Common Scenario #4

- Abdul currently manages a moderate sized university cluster
  - 256 Linux nodes
  - Storage
    - 20 TB of FC disk under a clustered file system for fast access
    - 50 TB of SATA disks accessible via a NFS system
- Abdul new cluster will be *much* larger
  - 2000 Linux nodes
  - 2 large SMP systems (e.g., 64 cores) using a proprietary OS
  - Storage capacity = 5 PB
  - Mixed I/O profile:
    - Small file, transaction access
    - Large file, streaming access

Lots of Questions

- What is my I/O profile?
- How can I control cost?
- How do I configure my system?
- Should I use a LAN or SAN approach?
- What kind of networks do I need?
- Can I extend my current solution, or do I need to start with a whole new design?
- Given the rate of growth in storage systems, how should I plan for future upgrades?
- What is the trade-off between capacity and performance?
- Can I use NFS or CIFS, or do I need a specialized file system?
- What are the performance issues imposed by a PB sized file system?
  - streaming rates, IOP rates, metadata management

Understanding Your User Profile

- Cache Locality
  - Working set: a subset of the data that is actively being used
  - Spatial locality: successive accesses are clustered in space
  - Temporal locality: successive accesses are clustered in time
- Optimum Size of the Working Set
  - Good spatial locality generally requires a smaller working set
    - Only need to cache the next 2 blocks for each LUN (*e.g.*, 256 MB)
  - Good temporal locality often requires a larger working set
    - The longer a block stays in cache, the more likely it can be accessed multiple times without swapping
- Generic file systems generally use virtual memory system for cache
  - Favor temporal locality
    - Can be tuned to accommodate spatial locality (*n.b.*, vmtune)
    - Virtual memory caches can be as large as all unused memory
    - Examples: ext3, JFS, Reiser, XFS

Understanding Your User Profile

- Common Storage Access Patterns
  - Streaming
    - Large files (*e.g.*, GB or more) with spatial locality
    - Performance is measured by bandwidth (*e.g.*, MB/s, GB/s)
    - Common in HPC, scientific/technical applications, digital media
  - IOP Processing
    - Small transactions with poor temporal and poorer spatial locality
      - small files or irregular small records in large files
    - Performance is measured in operation counts (*e.g.*, IOP/s)
    - Common in bio-informatics, rendering, EDA, home directories
  - Transaction Processing
    - Small transactions with varying degrees of temporal locality
      - Databases are good at finding locality
    - Performance is measured in operation counts (*e.g.*, IOP/s)
    - Common in commercial applications

Understanding Your User Profile

- Most environments have mixed access patterns
  - If possible, segregate data with different access patterns
    - Best Practice: do not place home directories on storage systems used for scratch space
- Best practice: before purchasing a storage system
  - Develop “use cases” and/or representative benchmarks
  - Develop file size histogram
  - Establish mean and standard deviation data rates
    - Rule of thumb: “Design a storage system to handle data rates 3 or 4 standard deviations above the mean.”
      - John Watts, Solution Architect, IBM

Understanding Your User Profile

- Use Cases
  - Benchmarks based on real applications
    - Provide the best assessment of actual usage
      - Carefully select representative workload
    - Can be difficult to use
      - Requires more time to evaluate then with synthetic benchmarks.
      - Can you give the data/code to vendor to use?
      - Is vendor willing to provide “loaner” system to customer?
  - Synthetic benchmarks
    - Easier to use and results are often published in white papers
      - Vendor published performance is usually based on synthetic benchmarks
      - But do they use a real file system configured for production environment?
    - Select benchmark codes that correlate to actual usage patterns
      - If a storage system meets a stated performance objective using a given benchmark, then it will be adequate for my application environment
    - Common examples
      - Bonnie++, IOR, iozone, xdd, SpecFS

Cost *vs.* Capacity *vs.* Performance *vs.* Reliability

- Do you want to optimize
  - Streaming performance
  - IOP performance
  - Capacity
  - Cost
  - Reliability
- How much can you spend to get what you need?
  - Gripe: Accountants should not dictate technical policy!

Cost *vs.* Capacity *vs.* Performance *vs.* Reliability

- Enterprise Class Disk
  - Fibre Channel (FC) Disk
  - Serial Attached SCSI (SAS)
  - Common Sizes: 146, 300, 450 GB
  - MTBF = 1.4 MHour
  - Rotational speed = 15 Krpm
  - Single drive IOP rate, 4K transactions (no caching): 380 IOP/s
  - Single drive streaming rate* via RAID controller
    - Controller cache disabled: write = 50.8 MB/s, read = 95.4 MB/s
    - Controller cache enabled: write = 154.6 MB/s, 123.6 MB/s
  - Best practice: Configure using RAID 3 or RAID 5
    - 4+P or 8+P is common

Optimizes reliability as well as streaming and IOP performance.

*Based on DS4800 benchmark accessing the “raw disk” via dd.
dd buffer size = 1024K, cache block size = 16K, segment size = 256K

Cost *vs.* Capacity *vs.* Performance *vs.* Reliability

Optimizes capacity.
Streaming performance and reliability are often good enough.

- Cost Optimized Disk
  - Serial ATA (SATA) Disk
  - Common Sizes: 750, 1000 GB
    - Larger sizes net generally in many current generation controllers
  - MTBF = 0.7 MHour
    - The MTBF rating is being replaced by annualized failure rate (AFR) which is 0.34% on representative SATA disks
  - Rotational speed = 7200 RPM
  - Single drive IOP rate, 4K transactions (no caching): 70 IOP/s
    - Command tag queuing (NCQ) can increase this rate to 120 IOP/s
  - Single drive streaming rate* via RAID controller
    - Controller cache disabled: write = 18.5 MB/s, read = 59.2 MB/s
    - Controller cache enabled: write = 30.3 MB/s, 74.9 MB/s
  - Best practice: Configure using RAID 6, especially in larger storage systems
    - 8+P+Q is common

*Based on DS4700 benchmark accessing the “raw disk” via dd.
dd buffer size = 1024K, cache block size = 16K, segment size = 64K

Cost *vs*. Capacity *vs*. Performance *vs*. Reliability

- For PB sized file systems, SATA may be good enough!
  - Depends in part on how the storage controller manages RAID
  - 240 SATA disks yield similar *streaming* performance to 128 FC disks*
    - SATA IOP rates are much less the FC IOP rates given poor locality
  - SATA using RAID 6 “levels the playing field” compared with FC using RAID 5
    - RAID 6 significantly lowers the risk of data loss due to “dual disk failures”
    - RAID capacity overhead is similar for 8+2P RAID 6 and 4+P RAID 5
    - RAID rebuild times with SATA/RAID 6 are longer than FC/RAID 5; this may be exacerbated by more frequent RAID rebuilds for SATA
      - Some storage controllers can in part compensate for this
  - Usable Capacity for SATA is much greater than FC disks
    - SATA with 8+2P RAID 6: 240 x 1 TB < 192 TB
    - FC with 4+P RAID 5: 128 * 450 GB < 46 TB

*Based on DS5300 benchmarks using the EXP5000 trays with 15Krpm FC and EXP5060 trays with 7200 RPM SATA. The trade-off point is different for different storage controllers.

Cost *vs.* Capacity *vs.* Performance *vs.* Reliability

- Reduce Cost Using Storage Hierarchy
  - Multiple storage tiers
    - Tier 1: Enterprise class
      - FC, SAS)
    - Tier 2: Cost optimized storage
      - SATA
    - Tier 3: Tape stored in libraries
    - Tier 4: Tape stored off-site
  - Backup vs. Archive
    - Archive – single copy of data
    - Backup – multiple copies of data
    - Best practice: integrate disk and tape layer

Tier-1

- Fast disk
  - e.g., FC disk
- Scratch Space

Tier-2

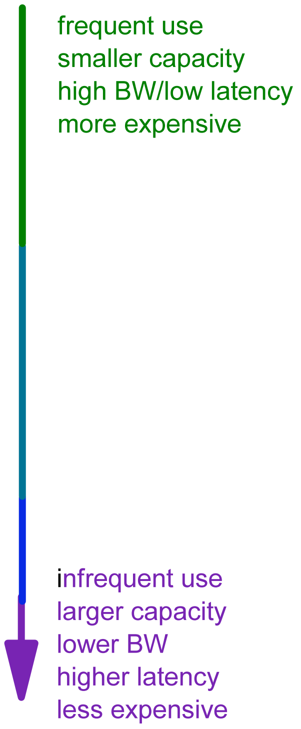
- High capacity disk
  - e.g., SATA
- Infrequently used files

Tier-3

- Local tape libraries

Tier-4

- Remote tape libraries



frequent use
smaller capacity
high BW/low latency
more expensive

infrequent use
larger capacity
lower BW
higher latency
less expensive

Cost *vs*. Capacity *vs*. Performance *vs*. Reliability

- Realistically Assess Uptime and Availability Requirements
  - Is a quality of service (QOS) guarantee necessary
    - Example: guaranteeing full performance in spite of component failures
  - Percentage of uptime requirements
    - 99.999% uptime ~= 5 min of down time per year
    - 99.99% uptime ~= 1 hour of down time per year
    - 99.9% uptime ~= 9 hours of down time per year
  - Guaranteed access to data
    - If this is a requirement…
      - Is access to all data in your data store necessary?
      - Is immediate access to the data necessary?
    - Design disaster recovery procedures
- Setting artificially high standards requires redundant systems and unnecessary cost.

Cost *vs.* Capacity *vs.* Performance *vs.* Reliability

- Considerations for Re-provisioning Legacy Storage
  - Can I preserve my investment?
  - Can I save money doing it? Does the cost of re-provisioning storage exceed its value?
  - Does it lock me into older technology that is no longer optimum for my application environment?
  - Is it feasible to segregate legacy storage and new storage?
    - If this is true, this is generally the easiest way to do it.
  - If not, is there an appropriate software product for my environment that can integrate them?
- Re-provisioning storage hardware is a common requirement.
  - Many file systems can accommodate this requirement to varying degrees.
  - There are also specialized software products that can also do this.
  - When other strategies are not feasible, NFS is often “good enough”.

Building Block Strategy

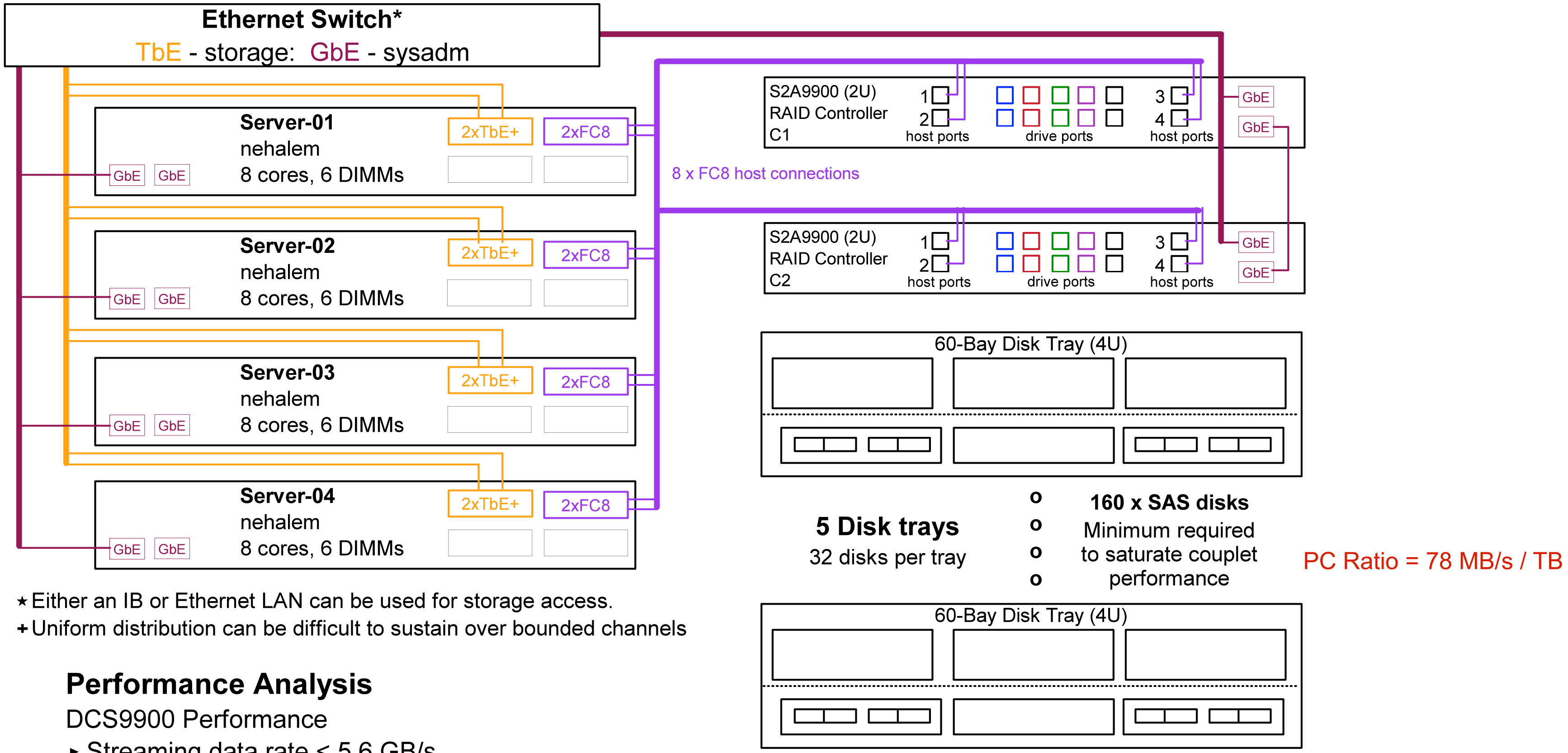
- Building Block Concept
  - Define a smallest common storage unit consisting of servers, controllers and disks
  - Replicate it multiple times until capacity and performance requirements are satisfied
  - Leads to a “build out as you grow” strategy
- Issues
  - Building blocks work best with LAN based file systems
  - Today’s storage technology is well suited for large building blocks which is appropriate for PB sized storage systems!
    - Controller cost/architecture make small building blocks less feasible
    - Small building blocks are not as effective in PB sized file systems
      - Small building blocks increase component counts which increases the risk of failure, yet they can have excellent price/performance curves
  - Building block design is often dictated by the choice of file system

Building Block Strategy

- Balance
  - Ideally, an I/O subsystem should be balanced
    - Do not make one part of storage system fast and another slow
      - Overtaxing some components of the I/O subsystem may disproportionately degrade performance
  - **Warning**: customer requirements may make this goal unachievable
    - "Performance is often inversely proportional to capacity."
      - Todd Virnoche, Business Partner Enablement, IBM
    - Number of disks needed to meet capacity exceeds performance
    - Number of disks needed to meet capacity yields greater performance than needed
      - Common example: data warehouses
    - Number of disks needed to meet performance exceeds capacity
      - Common example: national labs, university computing centers

Building Block Example

Example #1A – Large Building Block, Performance Optimized



⋆Either an IB or Ethernet LAN can be used for storage access.

+Uniform distribution can be difficult to sustain over bounded channels

Performance Analysis

DCS9900 Performance

- Streaming data rate < 5.6 GB/s
- Noncached IOP rate < 40,000 IOP/s

LAN: 4xDDR IB HCA (RDMA)

- Potential peak data rate per HCA < 1500 MB/s
- Required peak data rate per HCA < 1200 MB/s

SAN: 2xFC8 (dual port 8 Gbit/s Fibre Channel)

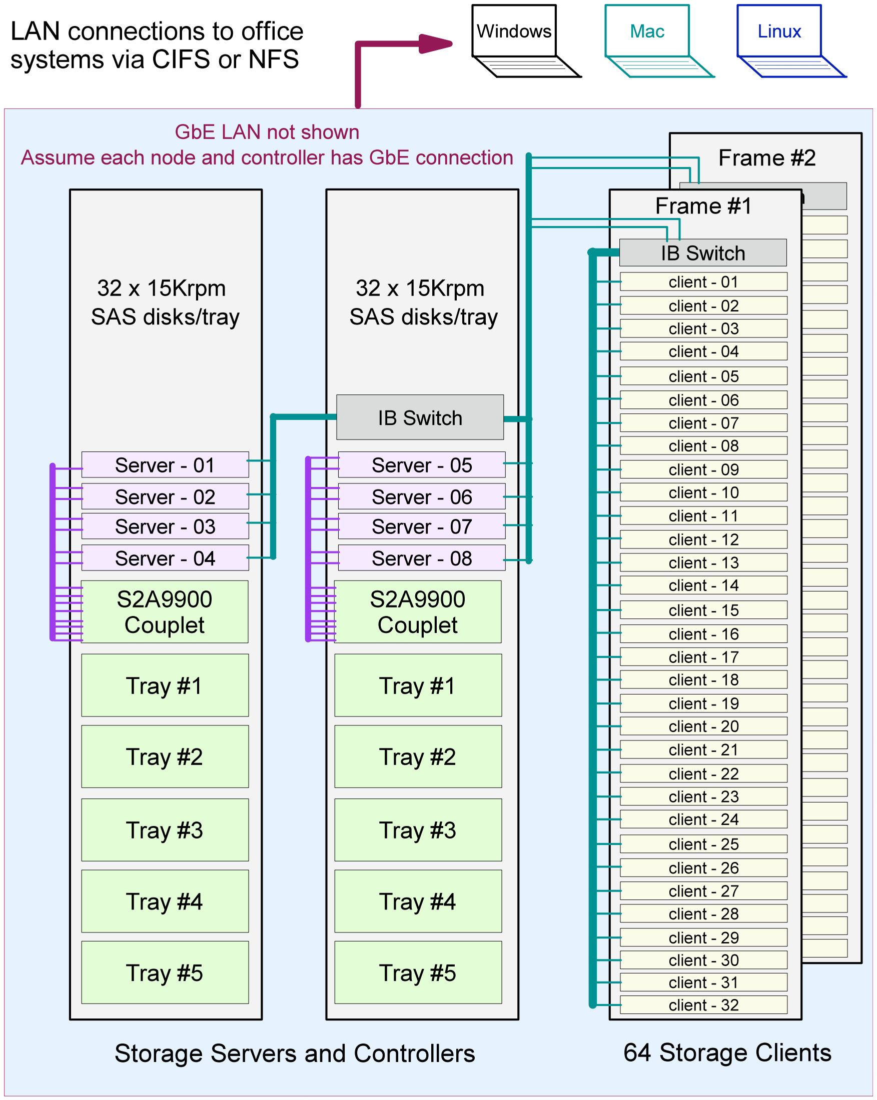
- Potential peak data rate per 2xFC8 < 1500 MB/s
- Required peak data rate per 2xFC8 < 1200 MB/s

Capacity Analysis

- SAS @ 15Krpm
  - 160 disks @ 450 GB/disk
  - 16 x 8+2P RAID 6 tiers
  - Capacity < 72 TB

Building Block Example

Example #1A (IB) – 2 Building Blocks, Performance Optimized



GbE —— IB —— FC8 ——

Aggregate Statistics

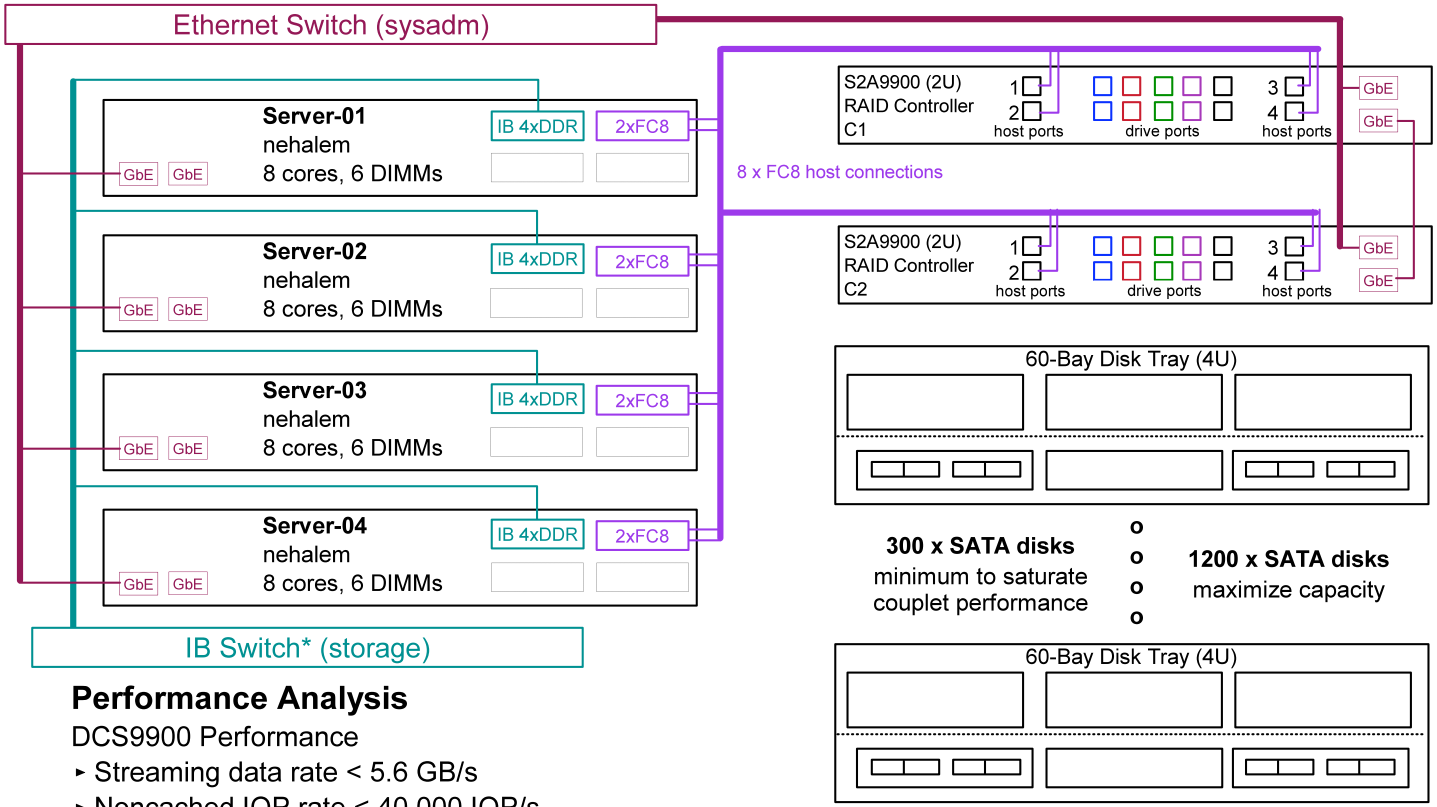
- 64 client nodes
- Streaming < 11 GB/s
  - Avg ~= 180 MB/s per node
  - Requires IB to be BW effective
- Capacity < 144 TB
  - 320 disks * 450 GB/disk

Scaling to PB Range

- Requires 14 bldg blocks
- Streaming < 78 GB/s

Building Block Example

Example #1B – Large Building Block, Capacity Optimized



300 x SATA disks
PC Ratio = 18 MB/s / TB
1200 x SATA disks
PC Ratio = 4.6 MB/s / TB

Performance Analysis

DCS9900 Performance

- Streaming data rate < 5.6 GB/s
- Noncached IOP rate < 40,000 IOP/s

LAN: 4xDDR IB HCA (RDMA)

- Potential peak data rate per HCA < 1500 MB/s
- Required peak data rate per HCA < 1200 MB/s

SAN: 2xFC8 (dual port 8 Gbit/s Fibre Channel)

- Potential peak data rate per 2xFC8 < 1500 MB/s
- Required peak data rate per 2xFC8 < 1200 MB/s

*Either an IB or Ethernet LAN can be used for storage access.

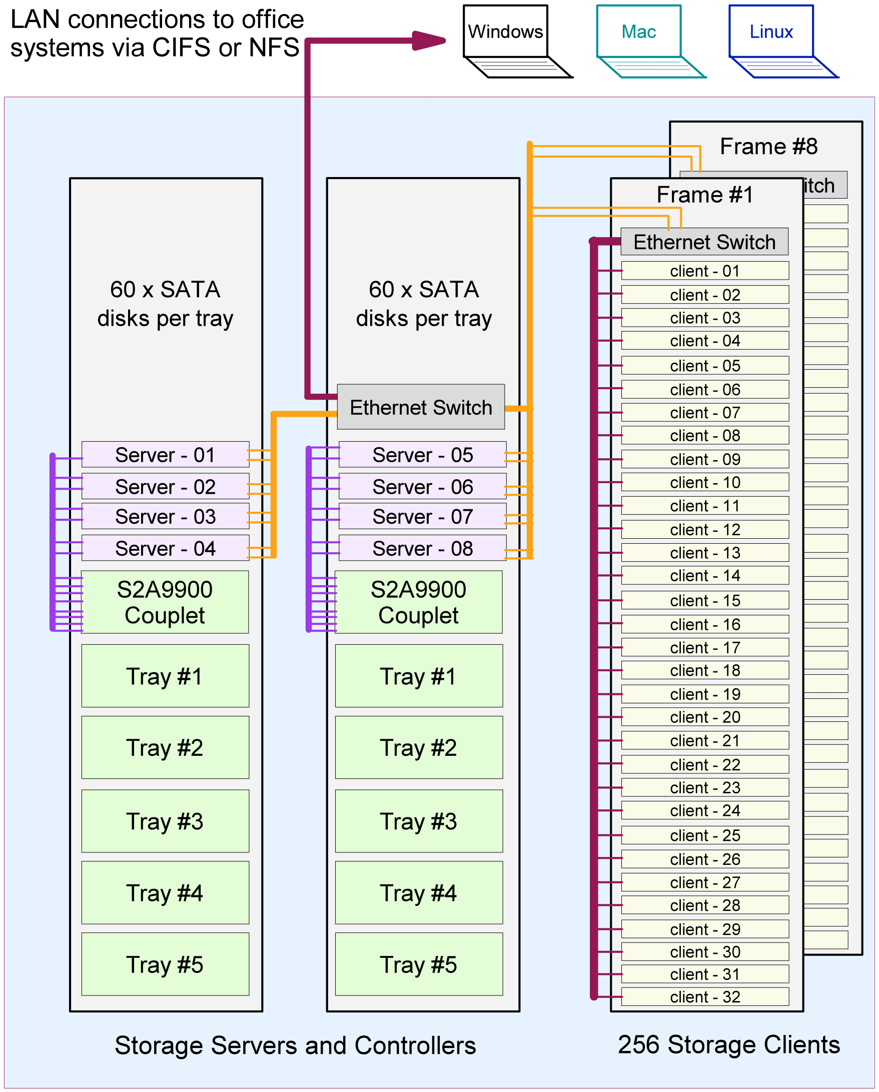
Capacity Analysis

- Balanced capacity/performance
- 300 x SATA disks
  - 5 disk trays
  - 30 x 8+2P RAID 6 tiers
  - Capacity < 300 TB

- Purely capacity optimized
- 1200 x SATA disks
  - 20 disk trays
  - 120 x 8+2P RAID 6 tiers
  - Capacity < 1.2 PB

Building Block Example

Example #1B (TbE) – 2 Building Blocks, Balanced Performance/Capacity



GbE —— TbE —— FC8 ——

Aggregate Statistics

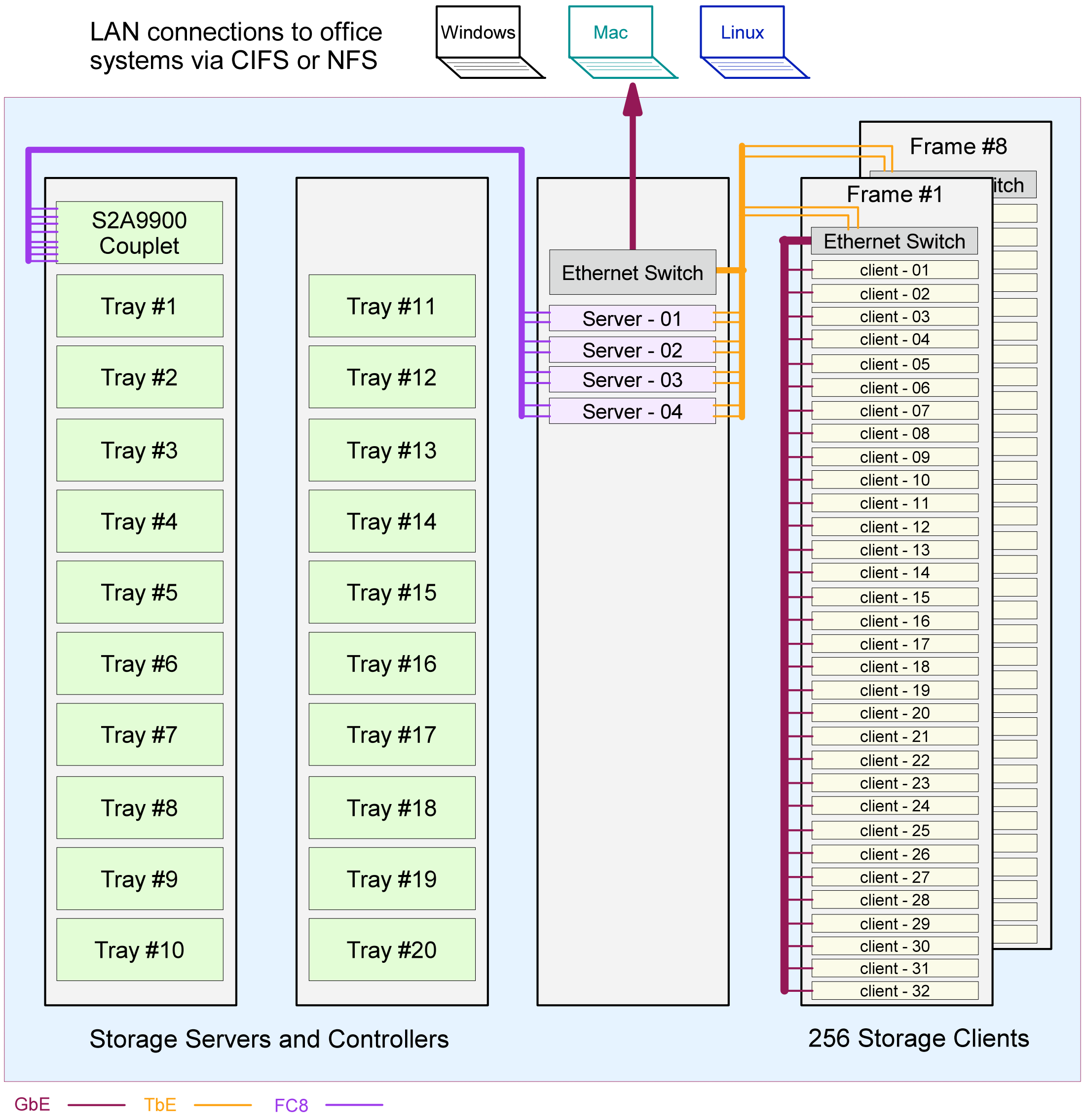
- 256 client nodes
- Streaming < 11 GB/s
  - Avg ~= 45 MB/s per node
  - IB is overkill for this storage system
    GbE is adequate for 256 nodes unless there is a large variance in the workload requiring short bursts of high bandwidth.
- Capacity < 600 TB
  - 600 disks * 1 TB/disk

Scaling to PB Range

- Requires 4 bldg blocks
- Streaming < 22 GB/s

Building Block Example

Example #1B (TbE) – 1 Building Block, Capacity Optimized



LAN connections to office systems via CIFS or NFS

Windows

Mac

Linux

S2A9900 Couplet

Tray #1

Tray #2

Tray #3

Tray #4

Tray #5

Tray #6

Tray #7

Tray #8

Tray #9

Tray #10

Tray #11

Tray #12

Tray #13

Tray #14

Tray #15

Tray #16

Tray #17

Tray #18

Tray #19

Tray #20

Ethernet Switch

Server - 01

Server - 02

Server - 03

Server - 04

Frame #8

itch

Frame #1

Ethernet Switch

client - 01

client - 02

client - 03

client - 04

client - 05

client - 06

client - 07

client - 08

client - 09

client - 10

client - 11

client - 12

client - 13

client - 14

client - 15

client - 16

client - 17

client - 18

client - 19

client - 20

client - 21

client - 22

client - 23

client - 24

client - 25

client - 26

client - 27

client - 28

client - 29

client - 30

client - 31

client - 32

Storage Servers and Controllers

256 Storage Clients

GbE —— TbE —— FC8 ——

Aggregate Statistics

- 256 client nodes
- Streaming < 5.6 GB/s
  - Avg ~= 22 MB/s per node
  - IB is overkill for this case

    GbE is adequate for 256 nodes unless there is a large variance in the workload requiring short bursts of high bandwidth.
- Capacity < 1.2 PB

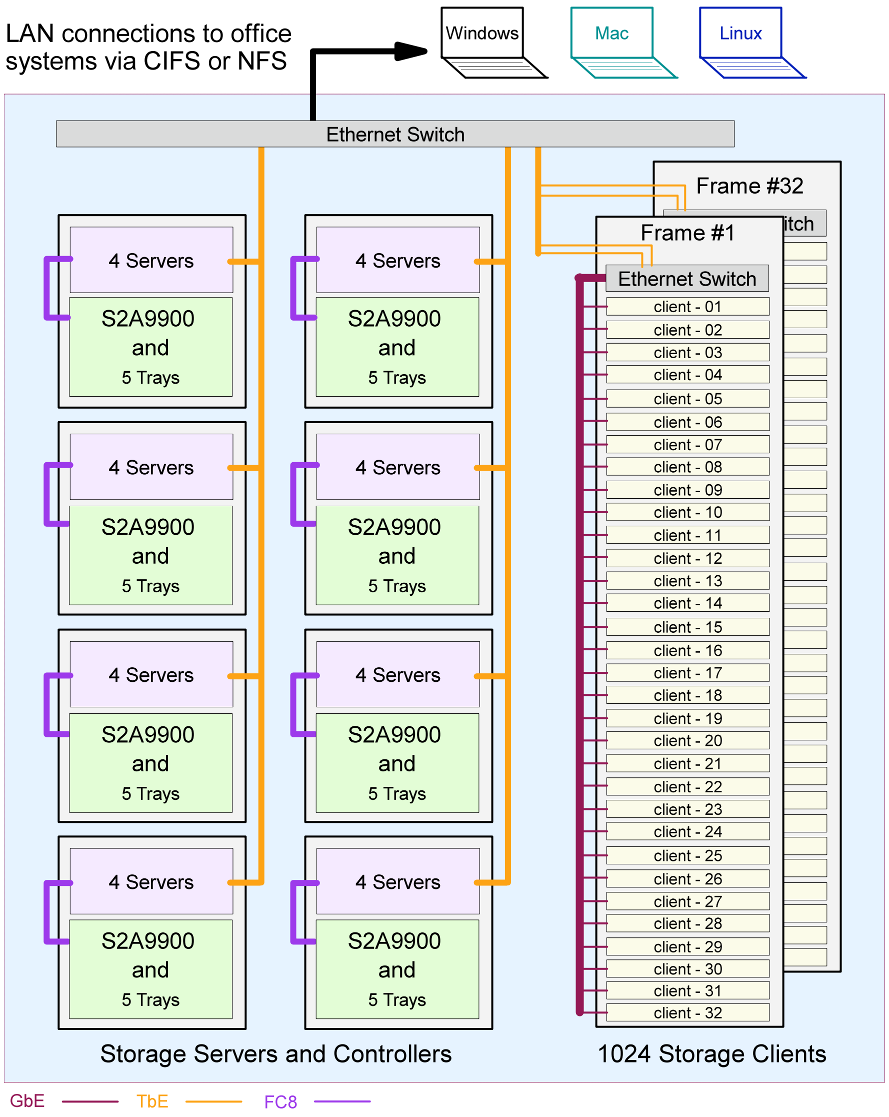
Scaling to PB Range

- Not necessary… this is a PB!

  Caution: If the client cluster is large (*e.g.*, 1024 nodes), the data rate per node will be very small (*e.g.*, 5 MB/s per node). If the variance is large, this may then be less of an issue.

Building Block Example

Summary Example – Capacity *vs.* Performance, IB *vs.* Ethernet



Aggregate Statistics

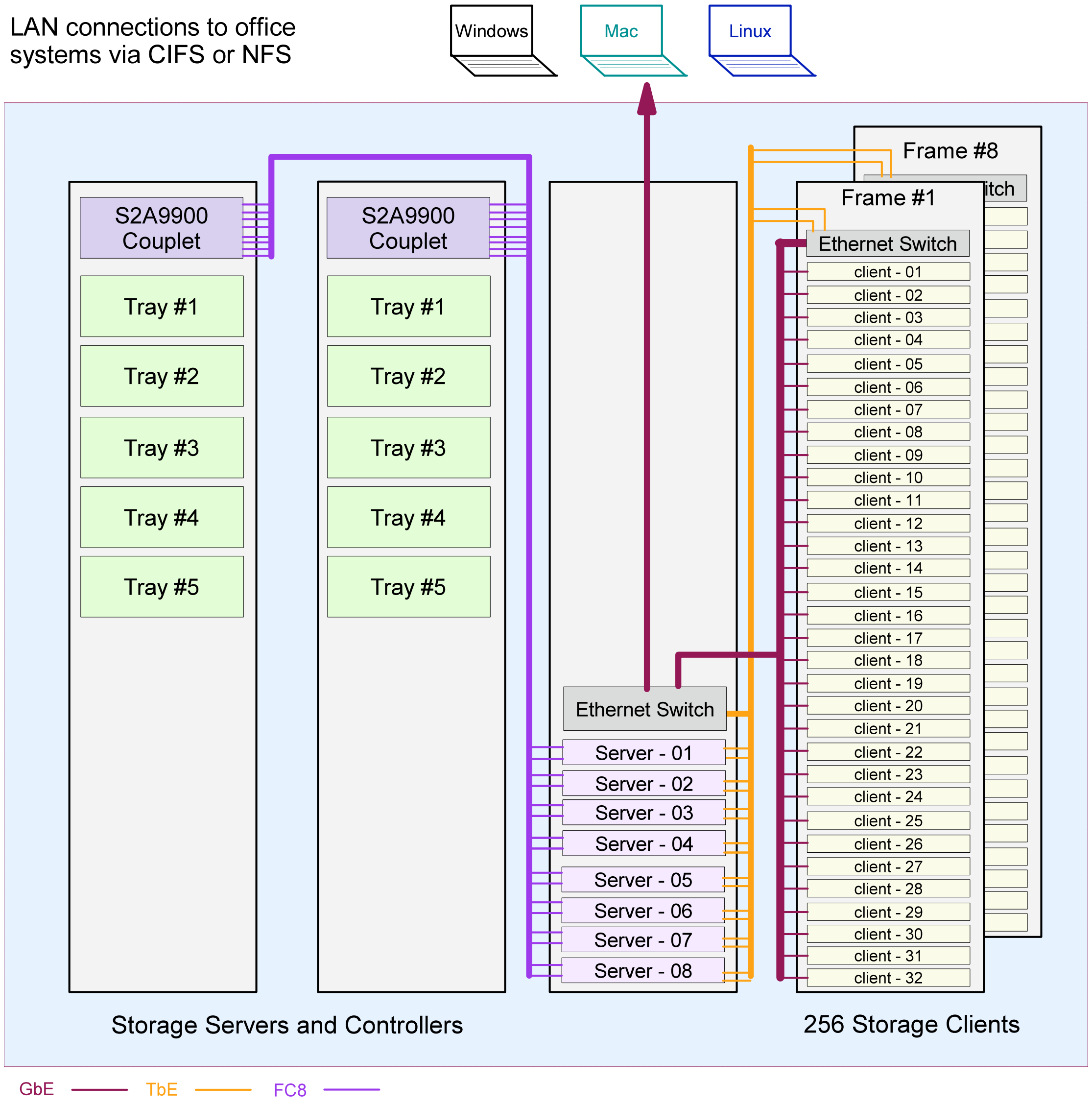
- 8 building blocks
- 1024 client nodes
- Using building block #1A
  - Streaming < 45 GB/s
  - Avg ~= 45 MB/s per node
  - Capacity < 576 TB
  - PC Ratio = 80 MB/s per TB
- Using building block #1B (balanced)
  - Streaming < 45 GB/s
  - Avg ~= 45 MB/s per node
  - Capacity < 2.4 PB
  - PC Ratio = 19 MB/s per TB

IB vs. Ethernet

- Ethernet is adequate for storage access
  - Avg ~ = 45 MB/s < GbE ~= 80 MB/s
  - assumes peak bandwidth per node < 80 MB/s
- Assume one or both of the following
  - Peak client storage rate > 80 MB/s
  - Avg message passing rate > 35 MB/s
- Two possible solutions
  - Create dedicated GbE LAN for message passing
  - Use IB LAN instead

Building Block Example – Common Mistake

Example #1B (TbE) – 2 Building Blocks



Aggregate Statistics

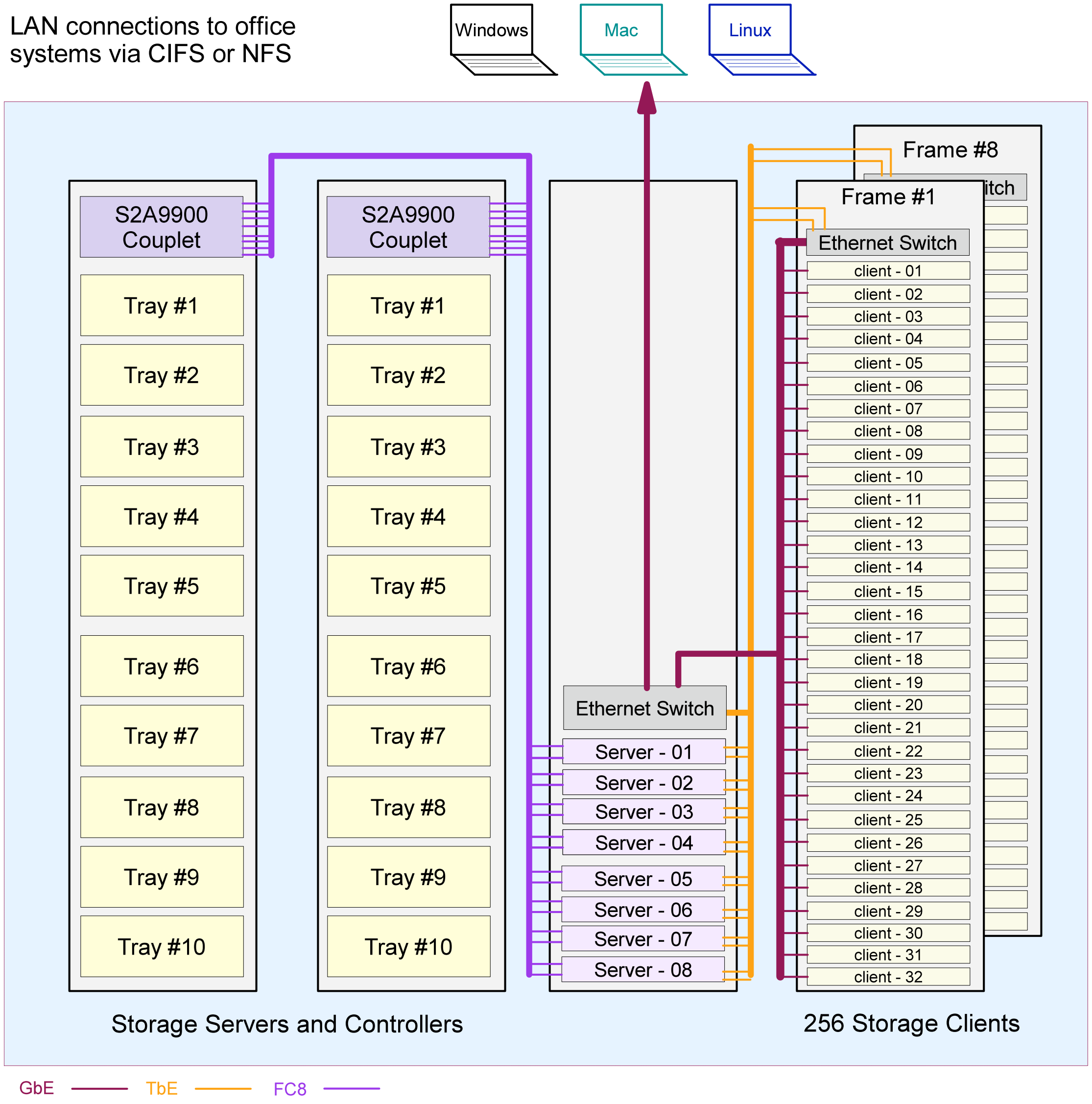
- 256 client nodes
- Streaming < 11 GB/s
  - Avg ~= 45 MB/s per node

Common mistake

- SATA
  - 2 x Couplets
  - 600 x 1 TB SATA < 600 TB

Building Block Example – Common Mistake

Example #1B (TbE) – 2 Building Blocks



Aggregate Statistics

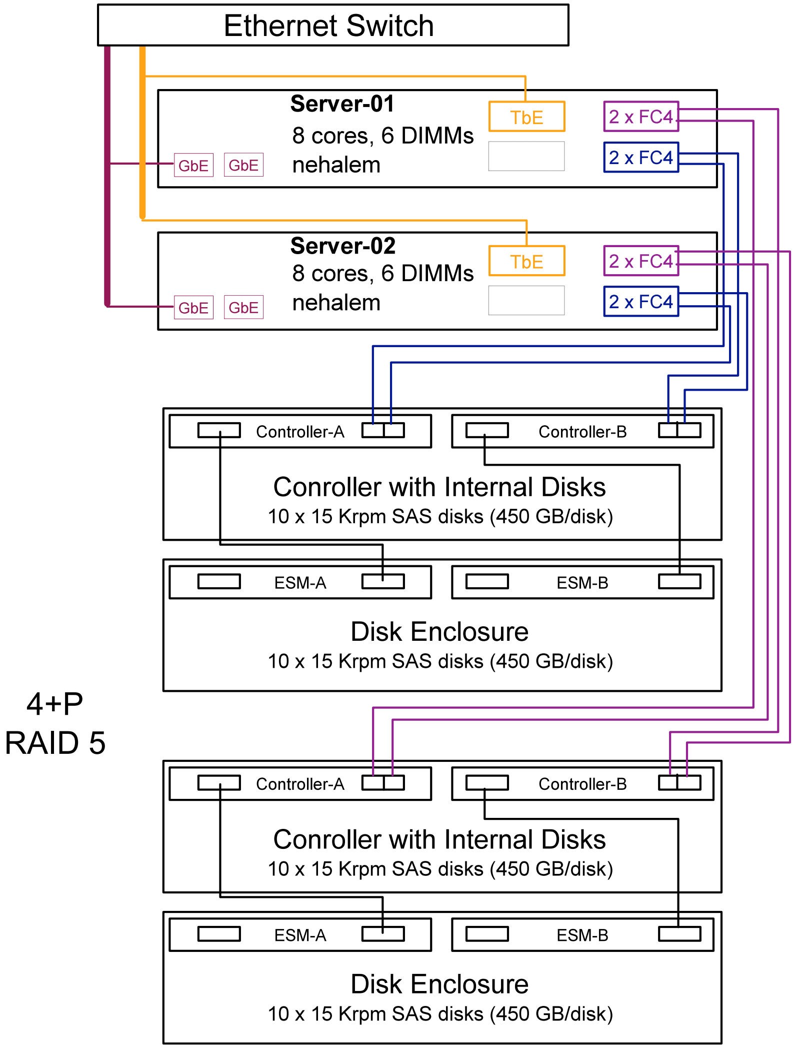
- 256 client nodes
- Streaming < 11 GB/s
  - Avg ~= 45 MB/s per node

Common mistake

- SATA *vs.* SAS
  - 2 x Couplets
  - 600 x 1 TB SATA < 600 TB
  - 1200 x 450 GB < 540 TB
  - Streaming performance is identical*

Building Block Example

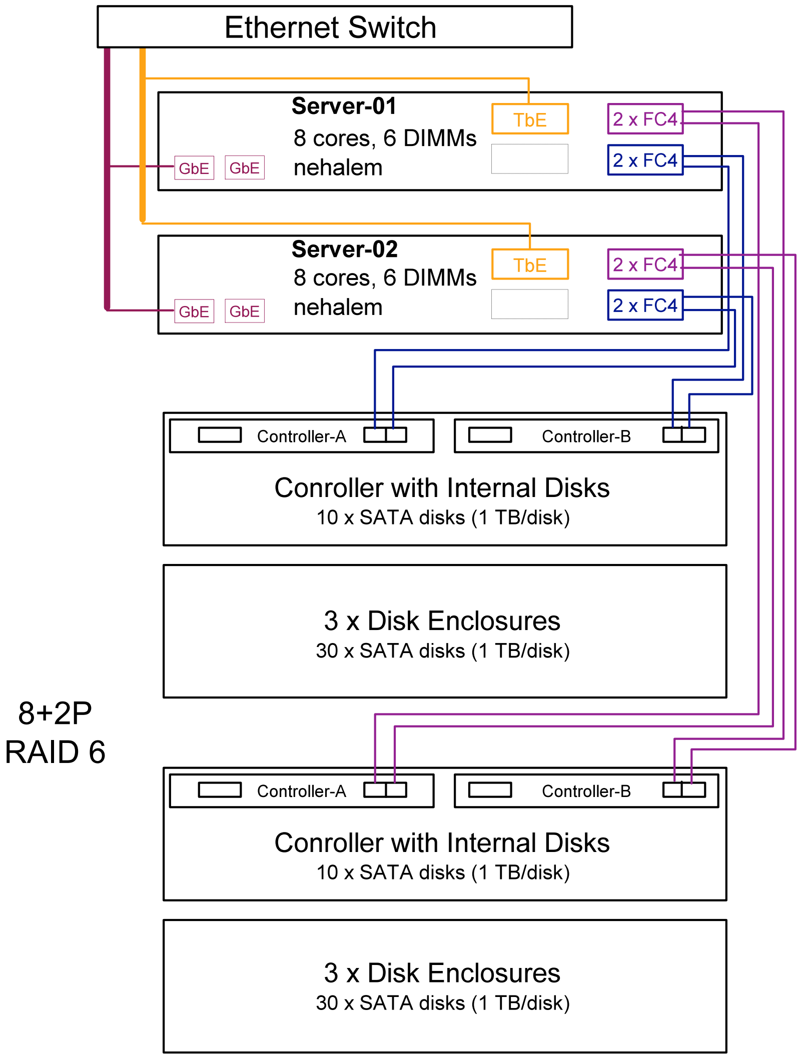
Example #2A – Small Building Block, Performance Optimized



- Storage Servers
  - 2xFC4 < 780 MB/s
  - TbE < 725 MB/s
- Storage Controller
  - “Twin tailed” disks
  - 20 disks per controller
  - 15Krpm FC disks
  - Write rate < 650 MB/s
  - Read rate < 800 MB/s
  - Capacity < 9 TB
- Aggregate Statistics
  - Data rate < 1450 MB/s
  - Capacity < 18 TB
  - PC Ratio = 80 MB/s / TB

Multiple servers, controllers and ports guarantee resilience.

Building Block Example
Example #2B – Small Building Block, Capacity Optimized



- Storage Servers
  - 2xFC4 < 780 MB/s
  - TbE < 725 MB/s
- Storage Controller
  - “Twin tailed” disks
  - 40 disks per controller
  - SATA disks
  - Write rate < 650 MB/s
  - Read rate < 800 MB/s
  - Capacity < 40 TB
- Aggregate Statistics
  - Data rate < 1450 MB/s
  - Capacity < 80 TB
  - PC Ratio = 18 MB/s / TB

Multiple servers, controllers and ports guarantee resilience.

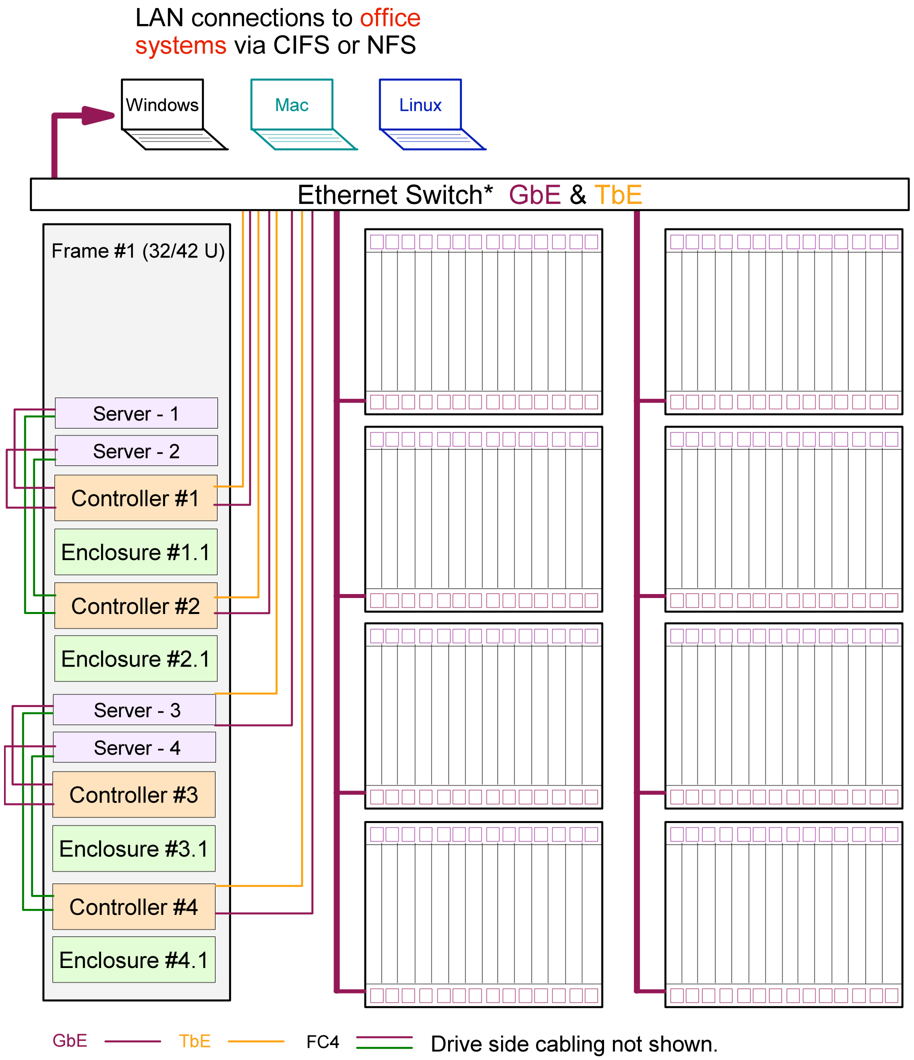
Building Block Example

Example #2 – Miscellaneous Comments

- Example #1A
  - There is room for 24 disks per disk controller, but 20 x 15Krpm disks in a 4+P RAID 5 configuration maximize the streaming performance of the controller.
    - In practice, 2 more disks are frequently included as “hot spares”.
  - To maximize IOP rate, the number of disks can be increased up to 48 per controller.
- Example #2A
  - There is room for 48 disks per disk controller, but 40 x SATA disks in a 4+2P RAID 6 configuration maximize the performance the controller.
- Caution
  - JBOD configuration increases the performance to capacity ratio, but the risk exposure of data loss in large configurations is unacceptably high.
  - While the streaming performance of these 2 solutions is similar, the IOP rate for the SATA solution is much less.

Building Block Example

Example #2A – 2 Building Blocks, Performance Optimized



* For greater redundancy, the Ethernet fabric can be deployed over 2 switches

Aggregate Statistics

- Streaming < 3 GB/s
- Capacity < 36 TB

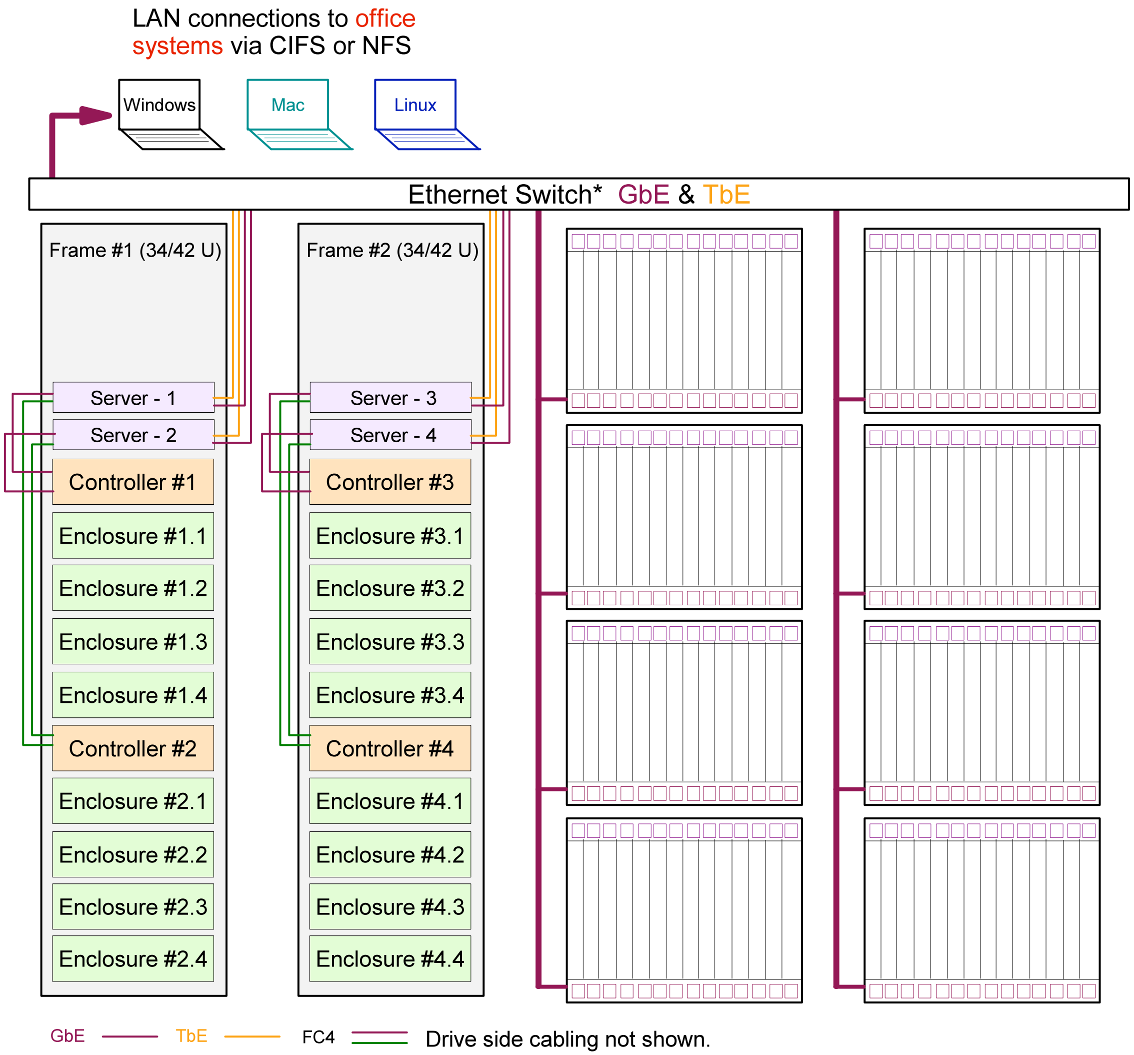
Scaling to PB Range

- Requires 28 bldg blocks+
- Streaming < 16 GB/s
  - Need > 500 GbE clients in order to fully utilize BW
- Small building block issues to be managed:
  - Complexity of managing 28 controllers
  - Controller failure (more controllers implies decreased MTBF)

+This is a good example of “give me the same thing, only bigger”. In practice, if this solution is scaled out to a PB, it will be difficult to administer and maintain.

Building Block Example

Example #2B – 2 Building Blocks, Capacity Optimized



* For greater redundancy, the Ethernet fabric can be deployed over 2 switches

Aggregate Statistics

- Streaming < 3 GB/s
- Capacity < 160 TB

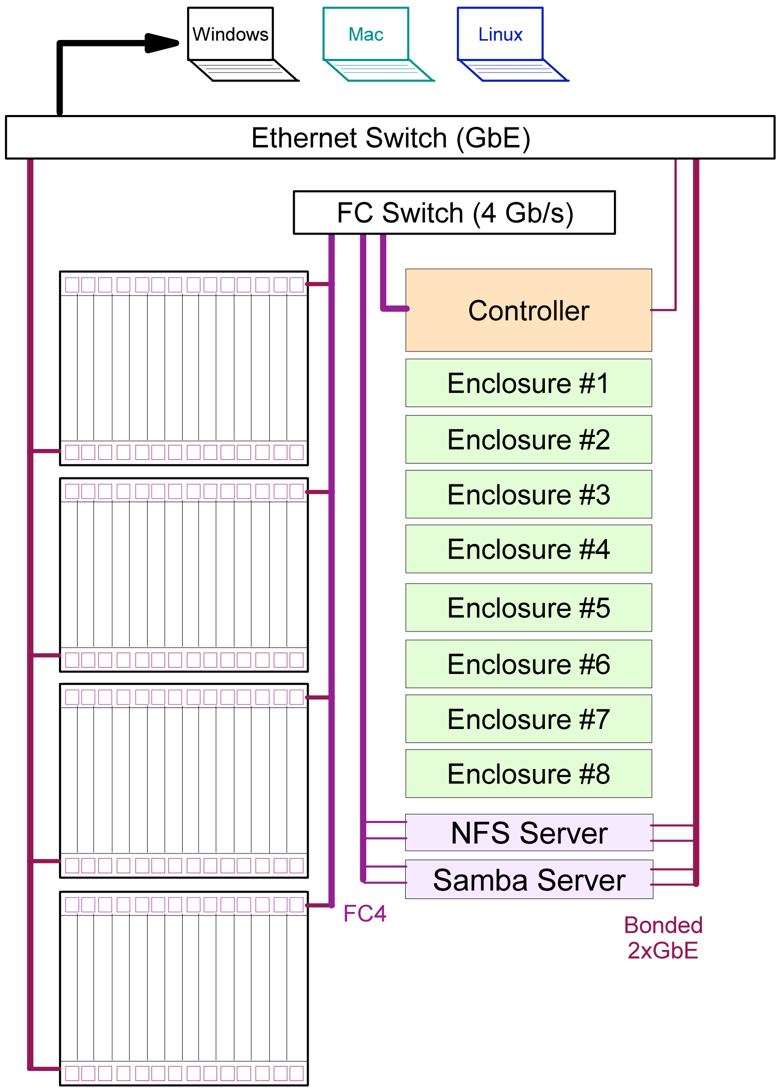
Scaling to PB Range

- Requires 12 bldg blocks
- Streaming < 16 GB/s
  - Need > 200 GbE clients in order to fully utilize BW
- Small building block issues to be managed:
  - RAID rebuild time
  - Controller failure (more controllers implies decreased MTBF)

Impact of 2 TB SATA Drives

- Lower PC ratio = 9 MB/s / TB
- Longer RAID rebuild times
- Requires only 6 building blocks lowering the component count to something manageable.

SAN Example
Current System



Current System

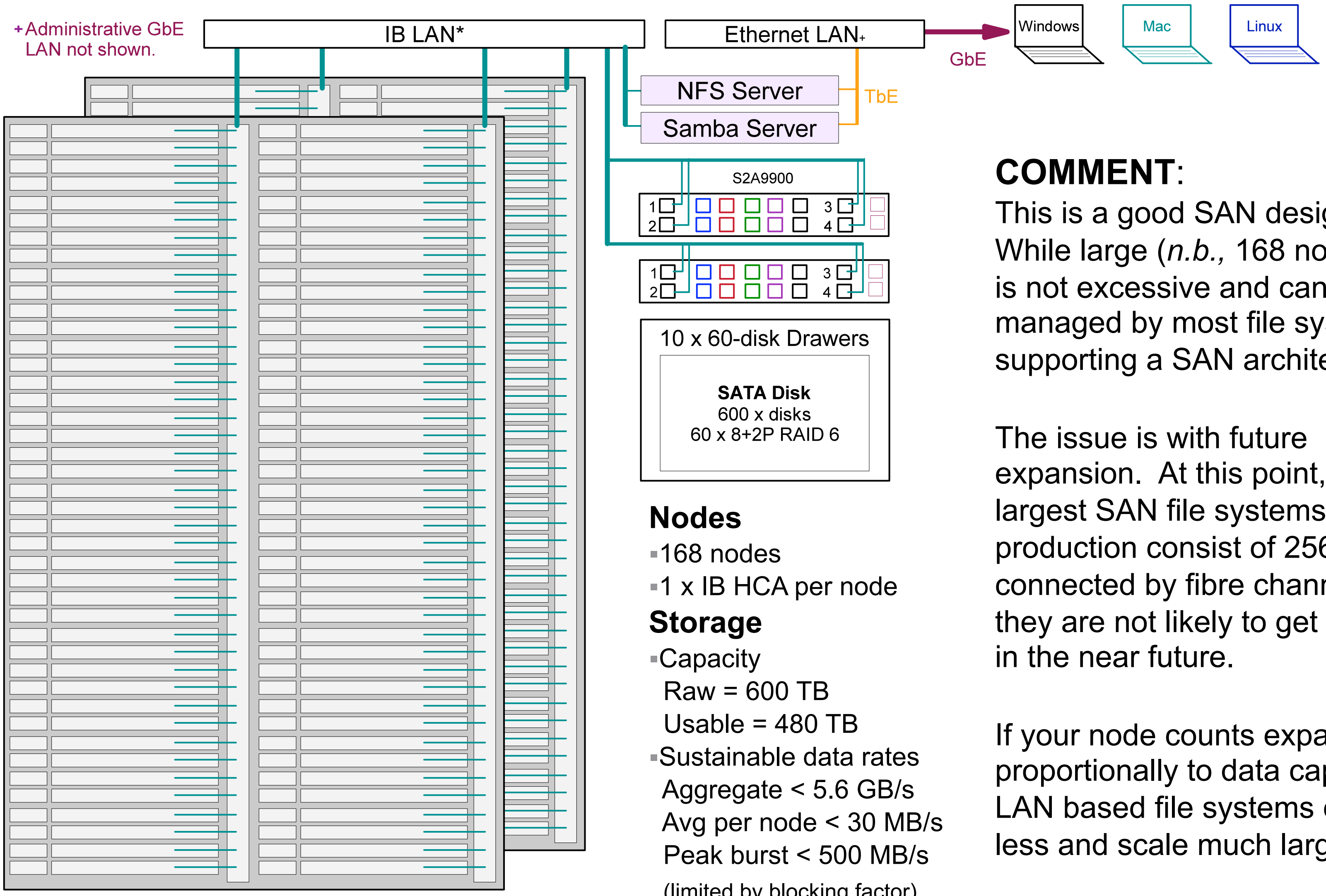
- 56 blades, each with GbE and FC4 port
- Desktop access
  - NFS Server with 2xFC4 and 2xGbE
  - Samba Server with 2xFC4 and 2xGbE
- Storage Controller Under a SAN File System
  - Capacity < 150 TB
  - Data rate
    - Aggregate rate: 1 to 2 GB/s
    - Average rate per node: 10 to 15 MB/s
    - Burst rate*: up to 200 MB/s

Requirements for New Cluster

- Phase 1
  - 160 nodes with IB network
  - Capacity = 500 TB
  - Data rate
    - Aggregate rate: 3 to 4 GB/s
    - Average rate per node: up to 20 MB/s
    - Burst rate*: up to 300 MB/s
- Phase 2: everything doubles in 18 months

*Short bursts of activity occurring on several blades at any given time.

SAN Example
New System: “Give Me the Same Thing, Only Bigger”



Nodes

- 168 nodes
- 1 x IB HCA per node

Storage

- Capacity
  Raw = 600 TB
  Usable = 480 TB
- Sustainable data rates
  Aggregate < 5.6 GB/s
  Avg per node < 30 MB/s
  Peak burst < 500 MB/s
  (limited by blocking factor)

COMMENT:
This is a good SAN design. While large (*n.b.,* 168 nodes), it is not excessive and can be managed by most file system supporting a SAN architecture.

The issue is with future expansion. At this point, the largest SAN file systems in production consist of 256 nodes connected by fibre channel; they are not likely to get larger in the near future.

If your node counts expand proportionally to data capacity, LAN based file systems cost less and scale much larger.

* IB Blocking factor ~= 3:1

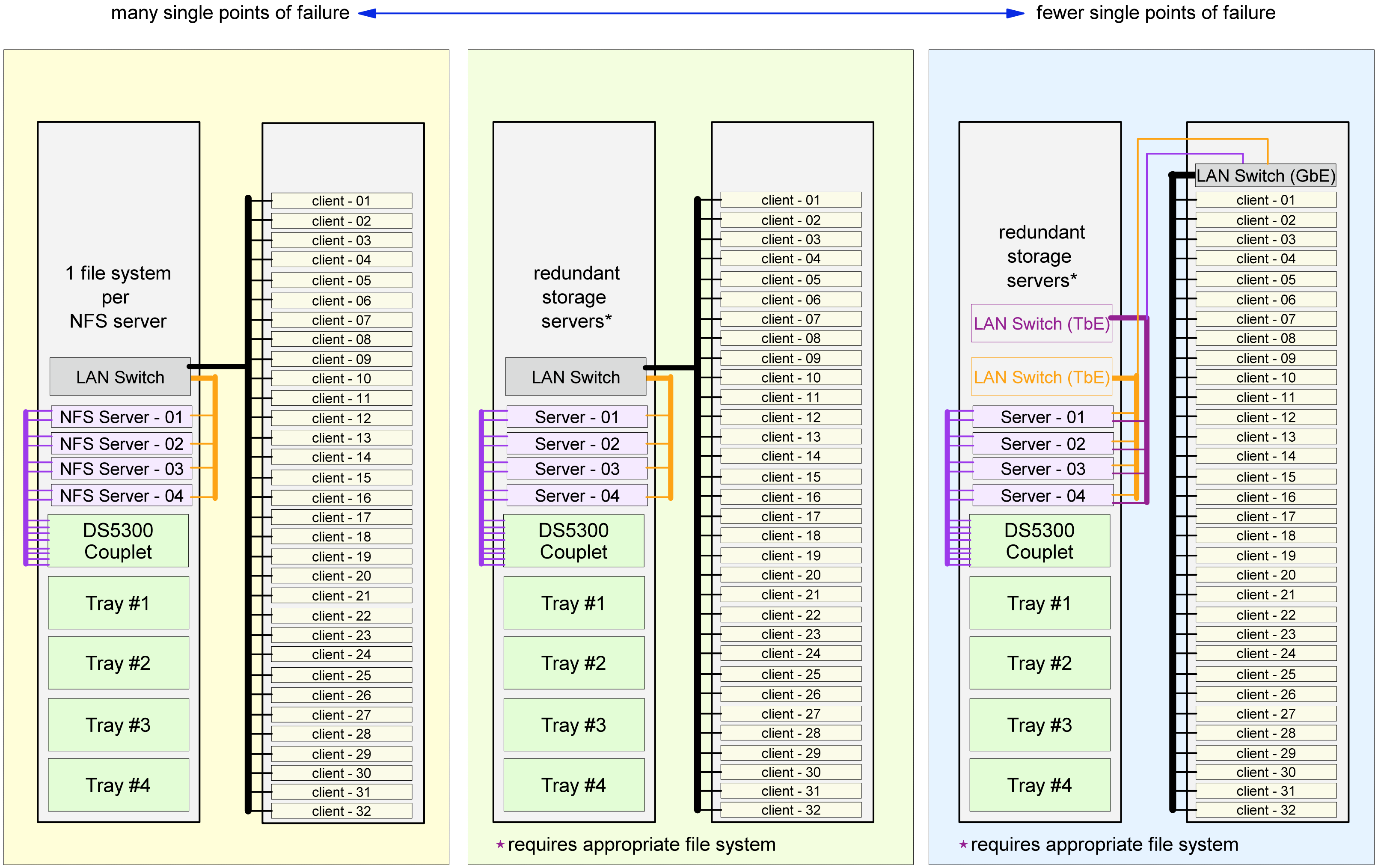
Performance Considerations: “Black Box Factor”

- Ease of Use (high black box factor)
  - Advantages
    - Are generally considered easy to use and administer
    - Performance is “good enough” for many environments
  - Principle limitation
    - Lack flexibility and tuning options to adapt to specialized applications
  - Example: NAS devices
- Flexibility
  - Advantages
    - Generally support a wide arrange of storage products
    - Provide wide range of tuning parameters making them adaptable to a wide range of applications
  - Limitations
    - More difficult to learn and use
  - Example: General purpose file systems

Performance Considerations: Seek Arm Mechanics

- Seek arm movement dominates disk performance
  - 15Krpm FC Disk: 3.6 ms
  - 7200 RPM SATA Disk: 9.0 ms
- Therefore, write applications to move as much data as possible per seek operation.
  - Small files (e.g., 4K) are generally accessed in a random order which forces 1 seek arm movement per file for a correspondingly small chunk of data.
  - Large records in a large file allows the disk to access a large volume of data per seek arm movement thereby improving efficiency.
  - But rewriting legacy codes is tedious and programming managers may not approve it.

Avoid Single Points of Failure in PB Sized Storage Systems



Increased redundancy can be achieved using 2xGbE per client and distributing the cluster over multiple sites. Carefully assess uptime requirements to avoid “gold plating” in this regard.

Principle Tools to Manage Storage

- Benchmarking Tools
  - Synthetic benchmarks *vs*. use cases
- System Monitoring Tools
  - Open source examples: ganglia, iostat, nmon, vmstat
- Storage Controllers
  - Provide disk management and monitoring
  - Example OEMs: DDN, EMC, IBM, LSI
- File Systems
  - The following pages take a closer look at file systems commonly used in clusters where PB sized file systems are common. Some of them are not as well suited for a PB scale as others.
  - Many file systems provide monitoring tools.

File System Taxonomy

The following pages examine a taxonomy of file systems commonly used with clusters. They may or may not be a clustered file system and they support varying degrees of parallelism. They do not represent mutually exclusive choices.

- Conventional I/O
- Asynchronous I/O
- Networked File Systems
- Network Attached Storage (NAS)
- Basic Clustered File Systems
- SAN File Systems
- Multi-component Clustered File Systems
- High Level Parallel I/O

Conventional I/O

- Used generally for "local file systems"
  - the basic, "no frills, out of the box" file system
- Supports POSIX I/O model
- Generally supports limited forms of parallelism
  - intra-node process parallelism
  - disk level parallelism possible via striping
  - not truly a parallel file system
- Journal, extent based semantics
  - *j ournaling* (AKA *logging):* to log information about operations performed on the file system meta-data as atomic transactions. In the event of a system failure, a file system is restored to a consistent state by replaying the log and applying log records for the appropriate transactions.
  - *extent:* a sequence of contiguous blocks allocated to a file as a unit and is described by a triple consisting of <logical offset, length, physical>
- If they are a native FS, they are integrated into the OS (*e.g.*, caching done via VMM)
- Examples: ext3, JFS, NTFS, ReiserFS, XFS

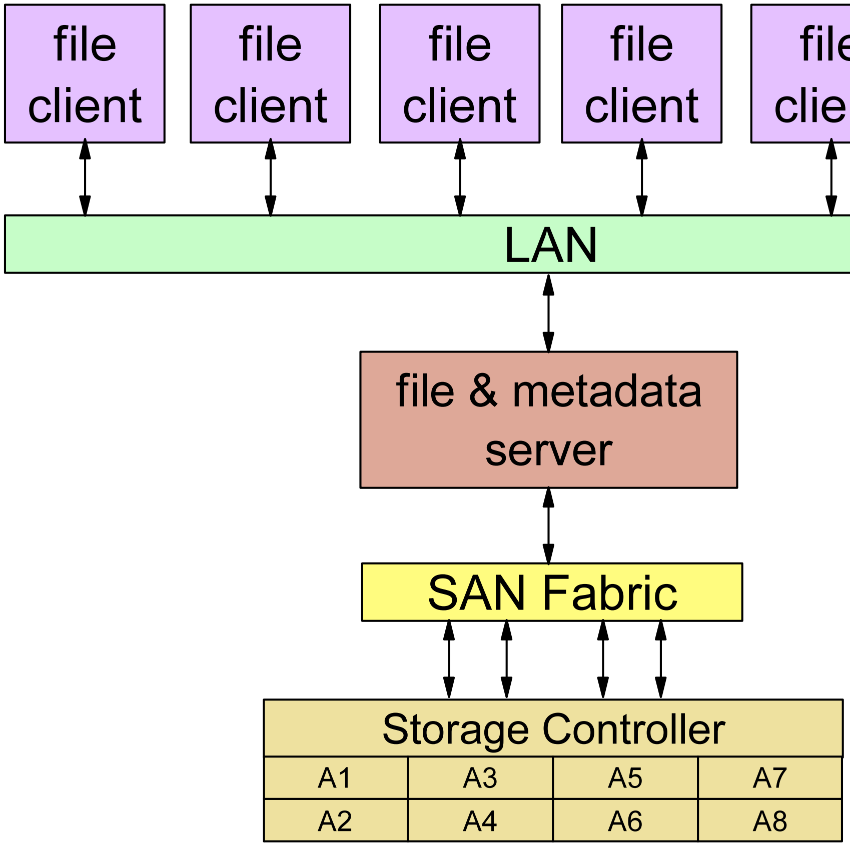
Asynchronous I/O

- Abstractions allowing multiple threads/tasks to *safely* and simultaneously access a common file
  - non-blocking I/O
  - built on top of a base file system
- Parallelism available if its supported in the base file system
- Included in the POSIX 4 standard
  - not *necessarily* supported on all Unix operating systems
- Examples:
  - commonly available under real time operating systems
  - Supported today on various "flavors" of standard Unix
    - AIX, Solaris, Linux (starting with 2.6)

Networked File Systems

- Disk access from remote nodes via network access
  - generally based on TCP/IP over Ethernet
  - Useful for on-line interactive access (*e.g.*, home directories)
- NFS is ubiquitous in Unix/Linux environments
  - does not provide a genuinely parallel model of I/O
    - it is not cache coherent (will future versions like pNFS correct this?)
    - parallel write requires O_SYNC and -noac options to be safe
  - poorer performance for HPC jobs, especially parallel I/O
    - write: only 90 MB/s on system capable of 400 MB/s (4 tasks)
    - read: only 381 MB/s on system capable of 740 MB/s (16 tasks)
  - uses POSIX I/O API, but not its semantics
  - traditional NFS configurations limited by "single server" bottleneck
  - while NFS is not designed parallel file access, by placing restrictions on an application's file access and/or doing non-parallel I/O, it may be possible to get "good enough" performance
  - NFS clients available for Windows, but POSIX to NTFS mapping is awkward
  - GPFS provides a high availability version of NFS called Clustered NFS
- CIFS is ubiquitous in Windows environments
  - Samba is a CIFS server available under Unix/Linux that maps a POSIX based file system to the Windows/NTFS model.

Networked File Systems



file client

file client

file client

file client

file client

file client

LAN

file & metadata server

SAN Fabric

Storage Controller

A1	A3	A5	A7
A2	A4	A6	A8

COMMENT:

Traditionally, a single NFS/CIFS file server manages both user data and metadata operations which "gates" performance/scaling and presents a single point of failure risk. Products (e.g., CNFS) are available that provide multiple server designs to avoid this issue.

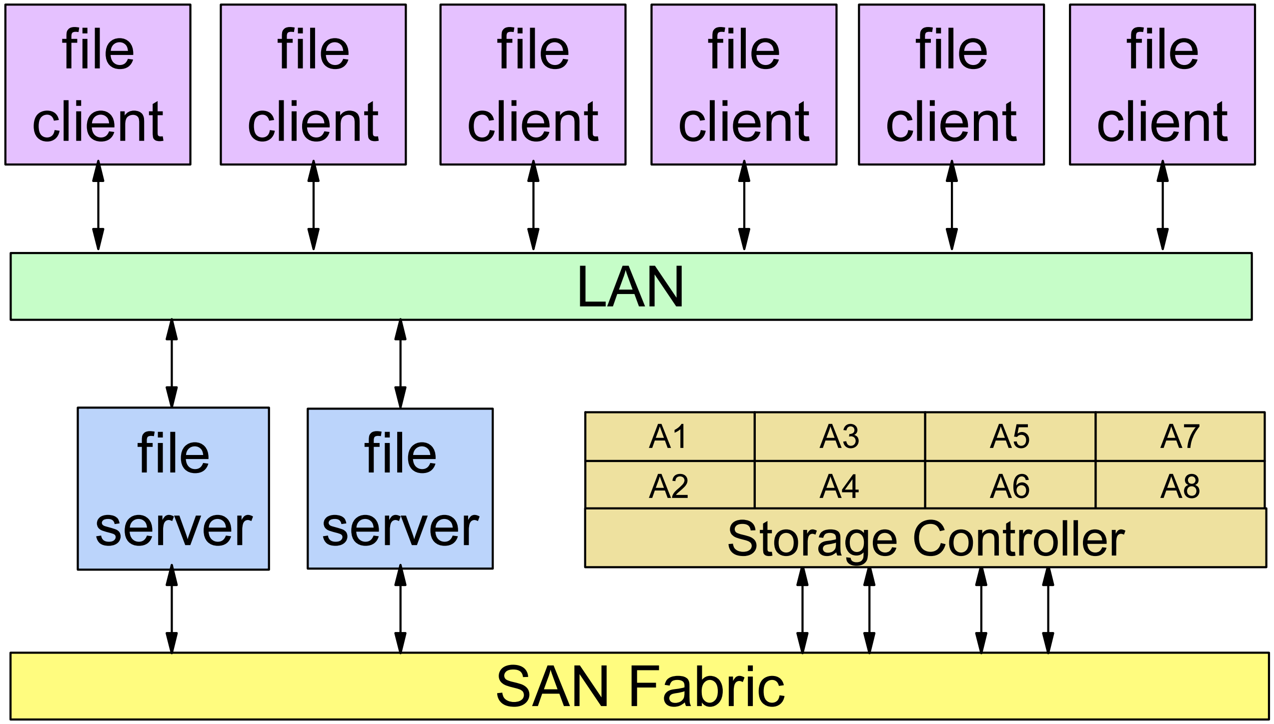
Network Attached Storage (AKA: Appliances)

- Appliance Concept
  - Traditionally focused on the CIFS and/or NFS protocols
  - Integrated HW/SW storage product
    - integrates servers, storage controllers, disks, networks, file system, protocol, etc. all into single product
    - main advantage: "black box" design (i.e., ease of use at the expense of flexibility)
    - not intended for high performance storage
  - Provides an NFS server and/or CIFS/Samba solution
    - these are server based products; they do not improve client access or operation
    - may support other protocols (*e.g.*, iSCSI, http)
  - Generally based on Ethernet LANs
  - Is this just a subclass of the networked file systems level?
- Examples
  - Netapp
    - Provides excellent performance for IOPS and transaction processing workloads with favorable temporal locality.
  - Scale-out File System (SoFS)
    - Supports CIFS (Samba), http, iSCSI, NFS, NSD (*i.e.*, GPFS) protocols

Basic Clustered File Systems

- Satisfies the definition of a clustered file system
- File access is parallel
  - supports POSIX API, but provides safe parallel file access semantics
    - guarantees portability to other POSIX based file systems
- File system overhead operations
  - file system overhead operations is distributed and done in parallel
  - there are no single server bottlenecks
    - *n.b.*, no metadata servers
- Common component architecture
  - commonly configured using seperate file clients and file servers
    - this is common for reasons of economy; for many storage systems, it costs too much to have a seperate storage controller for every node
  - some FS's allow a single component architecture where file clients and file servers are combined (*i.e.*, no distinction between client and server)
    - yields very good scaling for asynchronous applications
- file clients access file data through file servers via the LAN
- Example: GPFS, GFS, IBRIX Fusion

Basic Clustered File Systems

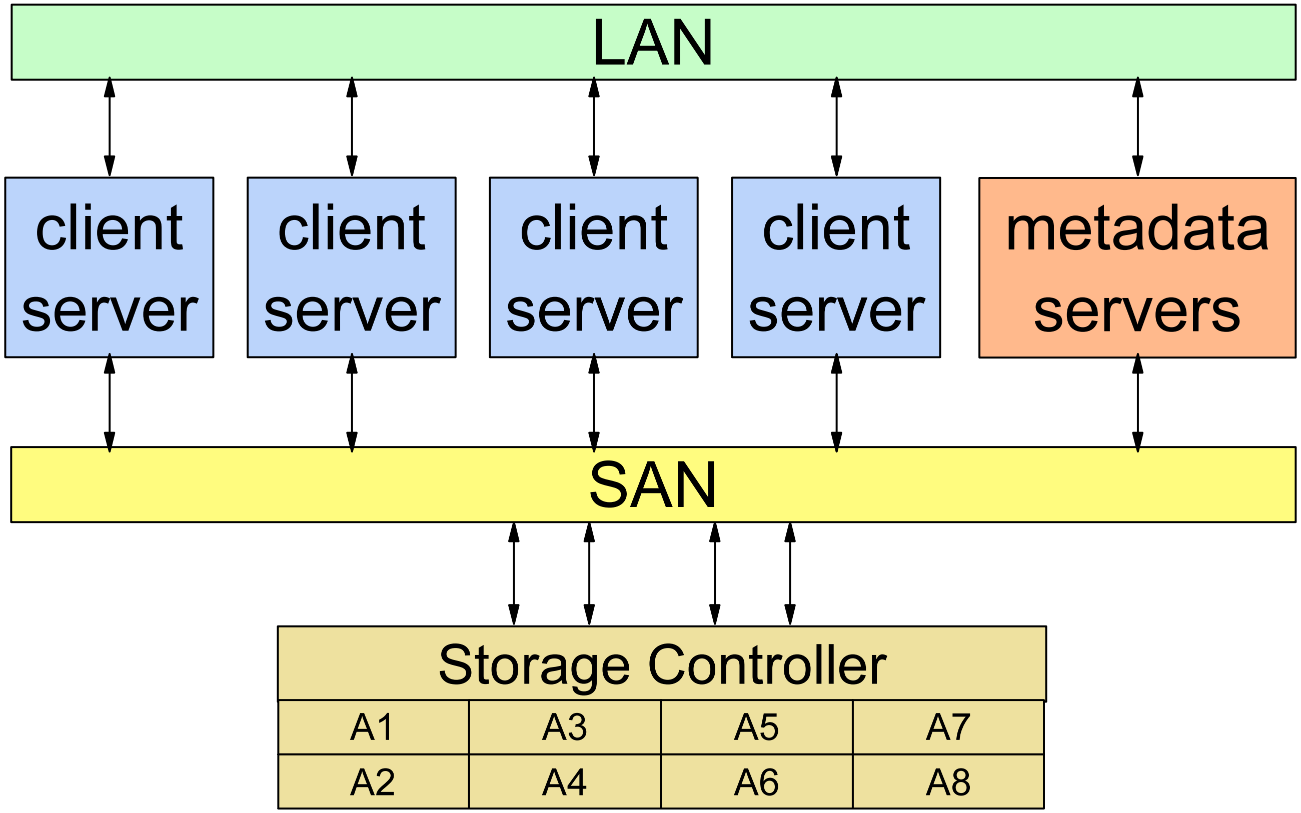


File system overhead operations are *distributed* across the entire cluster and is done in parallel; it is not concentrated in any given place. There is no single server bottleneck. User data and metadata flows betweem all nodes and all disks via the file servers.

SAN File Systems

- File access is parallel
  - supports POSIX API, but provides safe parallel file access semantics
    - guarantees portability to other POSIX based file systems
- File system overhead operations
  - it is NOT done in parallel
  - single metadata server with a backup metadata server
    - metadata server is accessed via the LAN
    - metadata server is a potential bottleneck, but it is not considered a limitation since these FS's are generally used for smaller clusters
- Dual component architecture
  - file client/server and metadata server
- All disks connected to all file client/server nodes via the SAN
  - file data accessed via the SAN, not the LAN
    - removes need for expensive LAN where high BW is required (*e.g.*, IB, Myrinet)
  - inhibits scaling due to cost of FC Switch Tree (*i.e.*, SAN)
- Example: CXFS (SGI), SNFS (Quantum, formerly ADIC), QFS (Sun)
  - ideal for smaller numbers of nodes
    - SNFS scales to 50+ nodes
    - CXFS scales up to 64+ nodes (appropriate for many-processor Altix systems)

SAN File Systems



A1	A3	A5	A7
A2	A4	A6	A8

File system protocol is *concentrated* in the metadata server and is not done in parallel; all file client/server nodes must coordinate file access via the metadata server. There are generally no client only nodes in this type of cluster, and hence the need for large scaling is not needed.

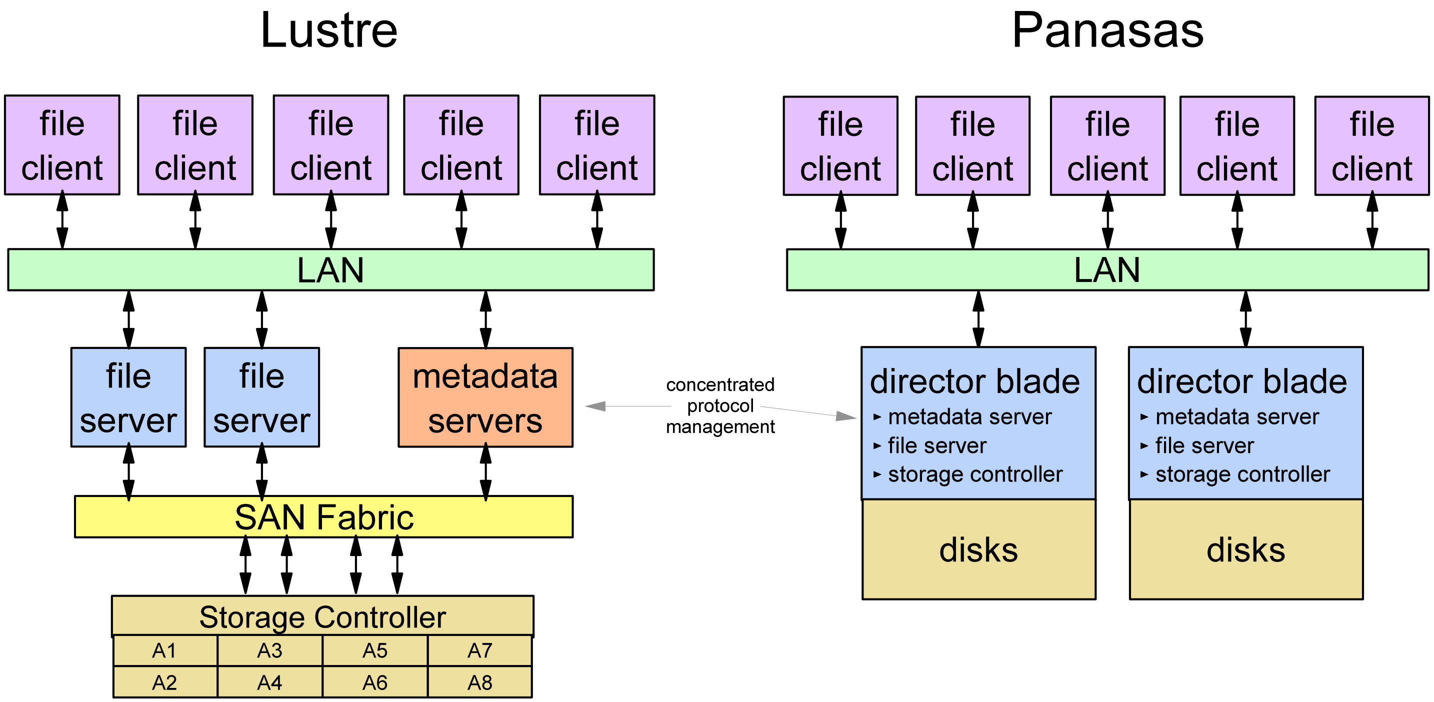
Multi-component Clustered File Systems

- Satisfies the definition of a clustered file system
- File access is parallel
  - supports POSIX API, but provides safe parallel file access semantics
    - guarantees portability to other POSIX based file systems
- File system overhead operations
  - Lustre: 1 metadata server per file system (with backup) accessed via LAN
    - potential bottleneck (deploy multiple file systems to avoid backup)
  - Panasas: the "director blades" manages protocol
    - each "shelf" contains a director blade and 10 disks accessible via Ethernet
    - this provides multiple metadata servers reducing contention
- Multi-component architecture
  - Lustre: file clients, file servers, metadata server
  - Panasas: file clients, director blade
    - director blade encapsulates file service, metadata service, storage controller operations
- file clients access file data through file servers or director blades via the LAN
- Examples: Lustre, Panasas
  - Lustre: Linux only, Panasas: Linux and Windows.
  - object oriented disks
    - Lustre emulates object oriented disks
    - Panasas uses actual OO disks; user can only use Panasas disks

Will this improve in the future?

Do OO disks really add value to the FS? Other FS's efficiently accomplish the same thing at a higher level.

Multi-component Clustered File Systems



While different in many ways, Lustre and Panasas are similar in that they both have concentrated file system overhead operations (*i.e.*, protocol management). The Panasas design, however, scales the number of protocol managers proportionally to the number of disks and is less of a bottleneck than for Lustre.

Higher Level Parallel I/O

- High level abstraction layer providing a parallel I/O model
- Built on top of a base file system (conventional or parallel)
- MPI-I/O is the ubiquitous model
  - parallel disk I/O extension to MPI in the MPI-2 standard
  - semantically richer API and semantics
    - can do things that POSIX I/O was never designed to do
  - applications using MPI-I/O are portable
- Requires significant source code modification for use in legacy codes, but it has the adavantages of being a standard (e.g., syntactic portability)
- Examples: MPICH, OpenMPI

Which File System is Best?

There is no concise answer to this question.

- It is application/customer specific.
- All of them serve specific needs.
- All of them work well *if properly deployed and used according to their design specs*.
- Issues to consider are
  - application requirements
    - often requires compromise between competing needs
  - how the product implements a specific architecture

Risk Is Inevitable… Manage It!

- If feasible, create multiple file systems localized to a subset of the disks to prevent collateral damage.
  - As an added benefit, this will allow you to have different file systems tuned for different access patterns.
- When using SATA disk, configure it using RAID 6
- Avoid single point of failure risk exposures
- Establishing disaster recovery procedures

Concluding Remarks

- PB sized file systems are not trivial
- Do not treat them as something peripheral to your environment
- Take time to analyze and understand your storage requirements
- Choose the proper storage tools (hardware and software) for your environment
- Storage is not the entire picture; improving I/O performance will uncover other bottle necks.
  - “A supercomputer is a device for turning compute-bound problems into I/O-bound problems.”
    - Ken Batcher, Professor of Computer Science, Kent State University