Managing Scalability in Object Storage Systems for HPC Linux Clusters

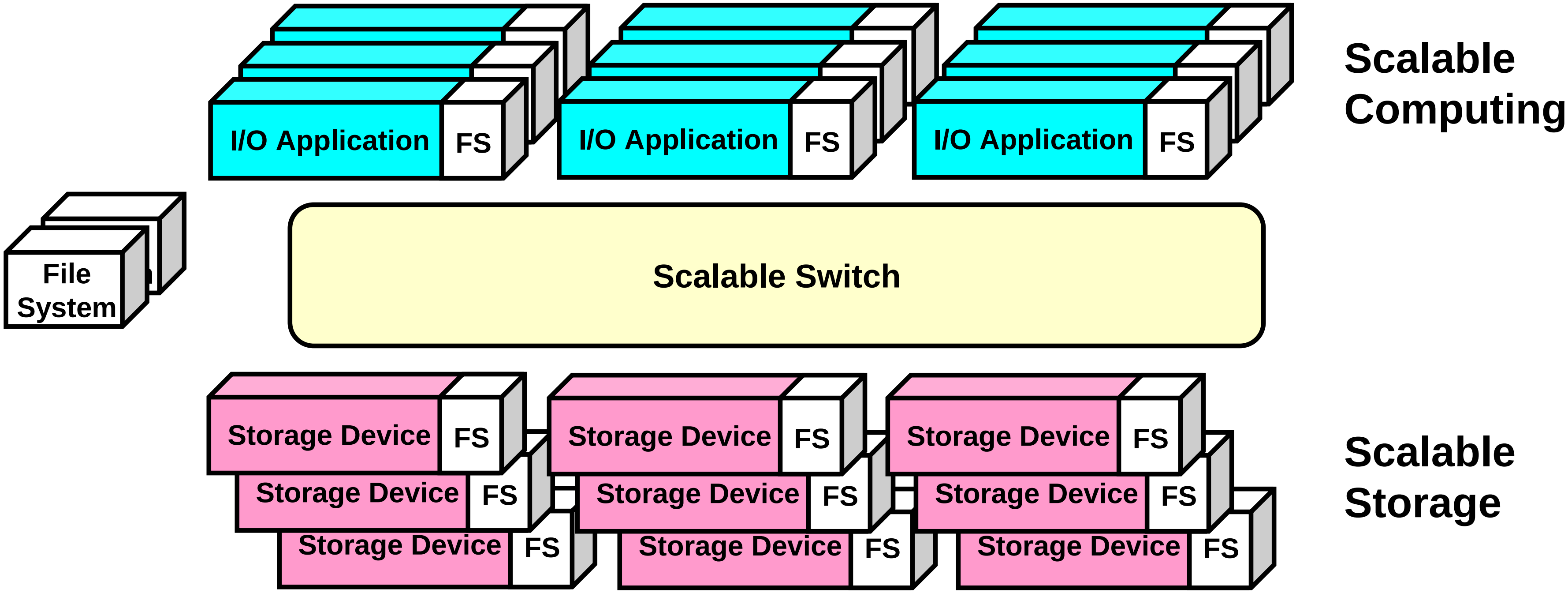
Brent Welch

Welch@panasas.com

NASA/IEEE MSST 2004
12th NASA Goddard/21st IEEE Conference on
Mass Storage Systems & Technologies
The Inn and Conference Center
University of Maryland University College
Adelphi MD USA
April 13-16, 2004

Ideal: Shared Storage Cluster

- **Separate storage for optimized reliability, centralized management**
  - Build storage as clusters of "sweet spot" HW + mostly independent SW
- **Direct I/O access to storage cluster**
  - Out-of-band meta data server



Scalable shared storage w/ NAS management

Blocks, Files and Objects

- **Block-base architecture:** ***fast but private***
  - Traditional SCSI and FC approaches
  - Expensive fabric, difficult to share between hosts
  - SAN Filesystems provide sharing, but have complex block manager
- **File-based architecture:** ***sharable, but bottlenecked performance***
  - NAS storage (NFS, CIFS, AFS and DFS)
  - Filer CPU and memory system between clients and disks
- **Object-based architecture:** ***fast and sharable***
  - Storage nodes directly accessible by clients via GbE
  - Out-of-band metadata servers make policy decisions for a file system
  - Storage nodes enforce access control to allow safe sharing

Objects vs. Blocks

- **Object Storage Device (OSD) is a Secure Shared Device**
  - Multiple hosts (clients) can access OSD simultaneously
  - OSD enforces access rights with capability check
- **Object Interface is higher level, so fewer round trips**
  - Create, Delete, Read, Write, GetAttributes, SetAttributes
  - Compact location information. (objID vs. list of blocks)
- **Object (File) operations map to several I/Os on data and metadata**
  - Private disks: hosts manage blocks and do local I/O
  - Shared disks: block manager is a bottleneck; hosts do more remote ops
  - Object disks: block management is hidden

Distributed File System

- **Object architecture supports many different models**
  - Panasas has implemented a shared, distributed file system
  - Files and Directories are stored in objects
- **Single system image**
  - Distribution across blades in Panasas cluster is transparent
  - One “mount point” (/panfs) connects client to all the storage
- **Global file system**
  - /panfs/*realmname*/*volume*/dir2/dir3/file
  - Potential to share a global namespace across organizations

Storage Hierarchy

NAS network
cut point

- **Directories**
  - Stored as files with special type
- **Files**
  - Stored in a collection of objects for fault tolerance

Object network
cut point

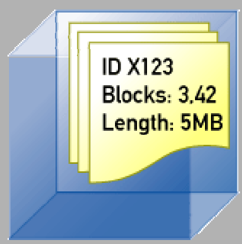
- **Object Groups**
  - Allow multiple managers to control objects in the OSD
- **Objects**
  - Collection of data blocks, plus attributes

SAN FS network
cut point

- **Blocks**
  - Fixed sized containers

Objects as Building Blocks

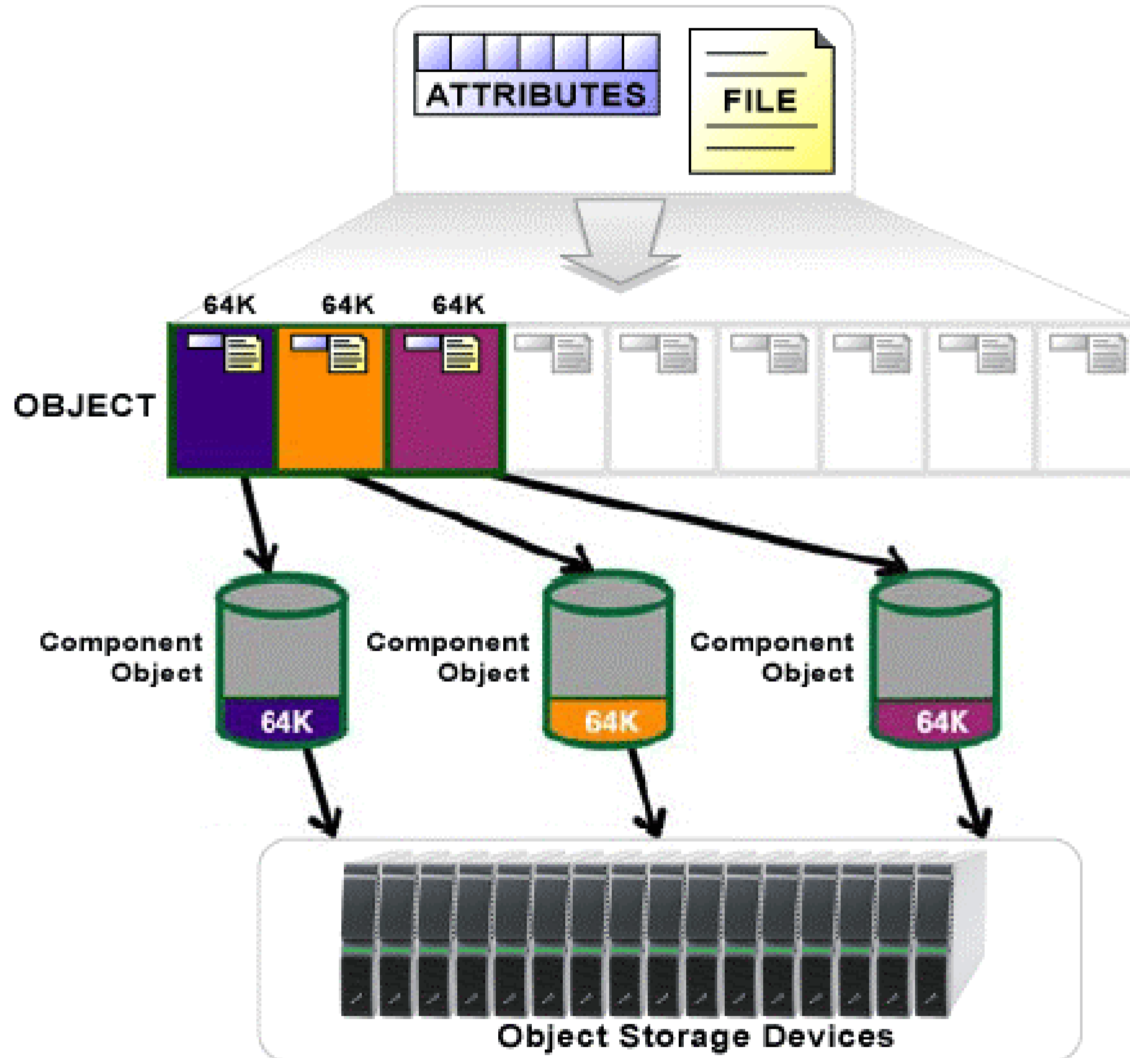
Object



Comprised of:
- User Data
- Attributes
- Layout

Interface:
- ID <dev,grp,obj>
- Read/Write
- Create/Delete
- Getattr/Setattr
- Capability-based

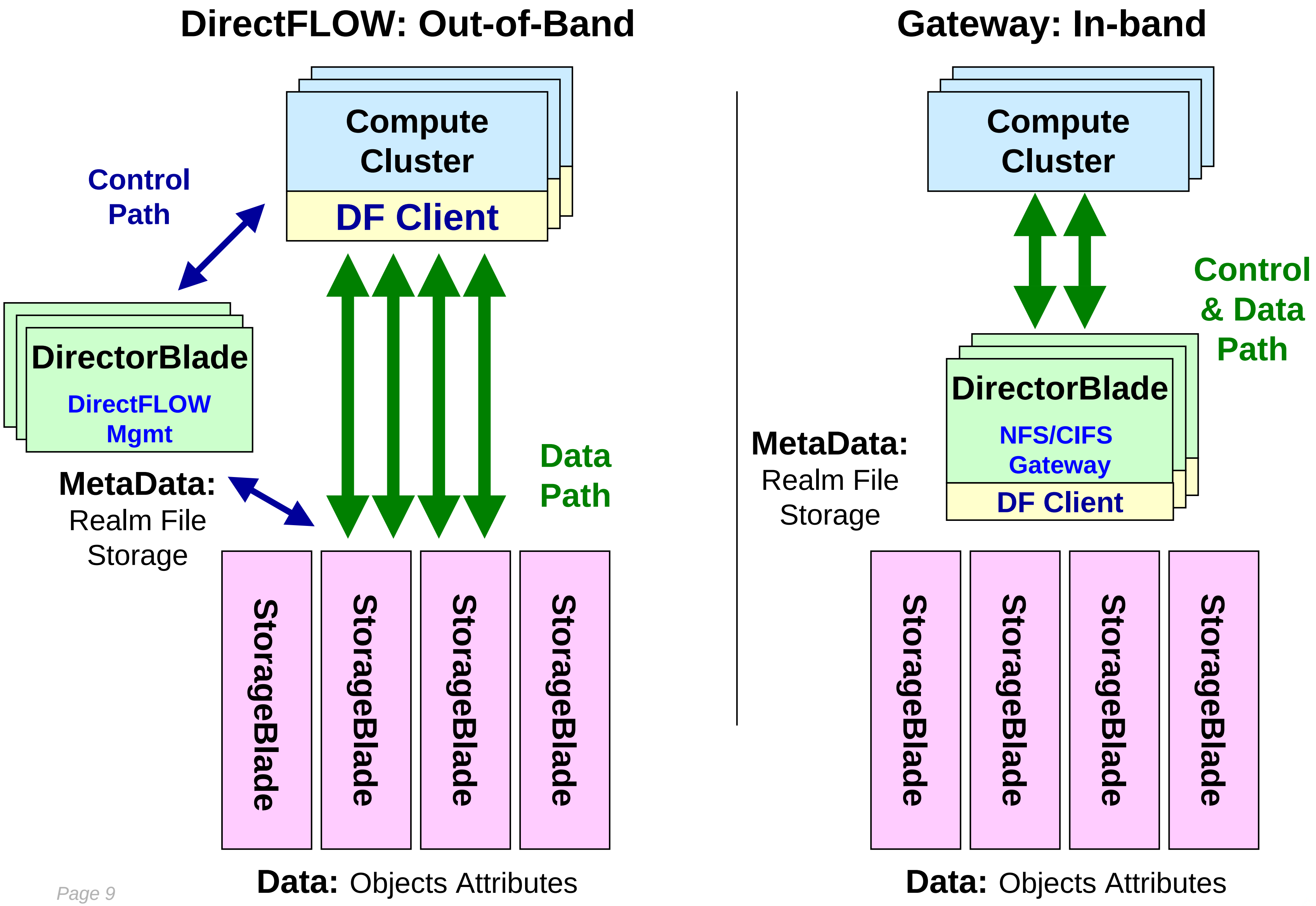
File Component:
- Stripe files across storage nodes



Object-Based Storage Clusters

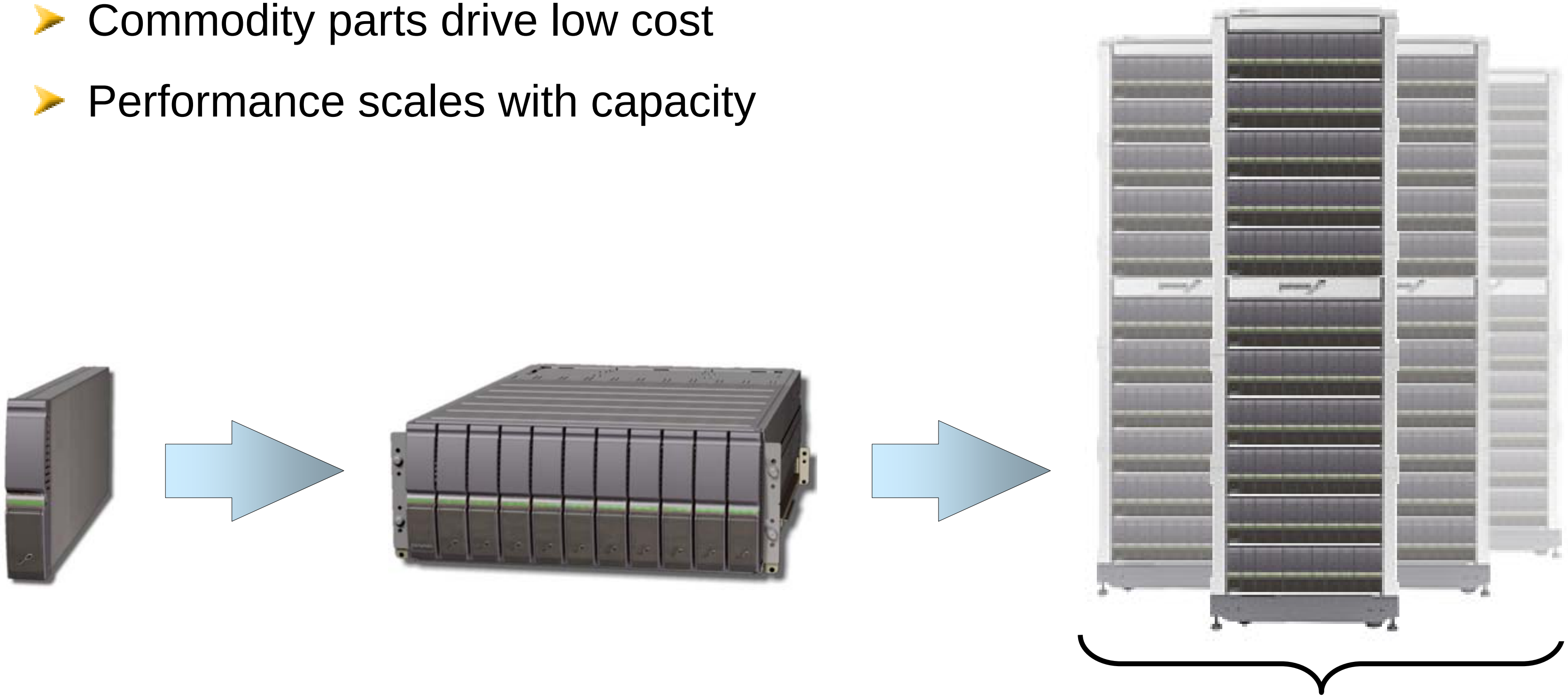
- **Consist of two primary components**
  - Object Storage Devices (OSD): **StorageBlades**
  - MetaData Manager: **DirectorBlades**
- **Directors implement file system semantics**
  - Access control, cache consistency, user identity, etc.
- **Directors have rights to perform these object operations**
  - Create, delete, create group, delete group
  - Get attributes and set attributes
  - Clone group, copy-on-right support for snapshots
- **Clients perform direct I/O with these object operations**
  - Read, write
  - Get attributes, set (some) attributes

Panasas Realm



Panasas StorageBlade (OSD)

- **Balanced storage device**
  - CPU, SDRAM, GE NIC and 2 spindles
    - 1.2 GHz, 512 MB, 2x250GB SATA
  - Commodity parts drive low cost
  - Performance scales with capacity



Single Seamless Namespace!

DirectFLOW Client

- **DirectFLOW client is kernel loadable FS module**
  - Implements standard Vnode interface
  - Uses native Panasas network protocols (RPC and iSCSI)
- **Caches data, directories, attributes, capabilities**
- **Responds to callbacks for cache consistency**
- **Does RAID I/O directly to StorageBlades w/ iSCSI/OSD**

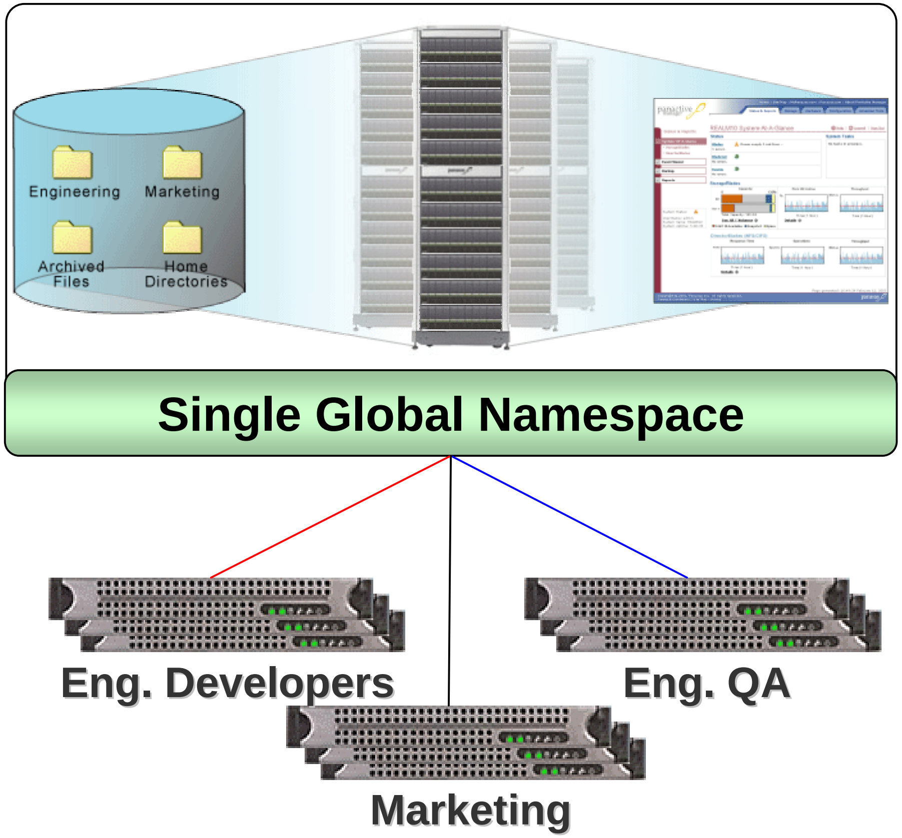
DirectorBlades

- **Metadata manager**
  - Realm Control – admit blades, start/stop services, failover
  - File Manager – access control, cache consistency, file system semantics
  - Storage Manager – file virtualization (maps), recovery, reconstruction
- **Management console**
  - Web-based GUI or Command Line Interface (CLI)
  - Status, charts, reporting
  - Storage management
- **Gateway function (NFS/CIFS) collocated on DirectorBlade**
  - Fast processor and large main memory
- **Multiple DirectorBlades allow service replication for fault tolerance**

Manageability

- **Single filesystem namespace**
  - Removes physical & logical boundaries
  - Dynamic load-balancing
- **Interoperability**
  - Gateway for NFS/CIFS
  - “Free” clustered NAS
- **Internal cluster management**
  - IP address block (panDHCP)
  - Fault tolerance
  - Environmental/thermal monitoring
  - Software upgrades
- **Service and Support**
  - Personalized extranet for bugs, SRs, orders

Panasas ActiveScale Architecture



Environment

- **AC Power**
  - Each shelf has dual power supplies and battery
  - Automatic graceful shutdown if you lose AC power
  - Masks brownouts and short (5-sec) power glitches

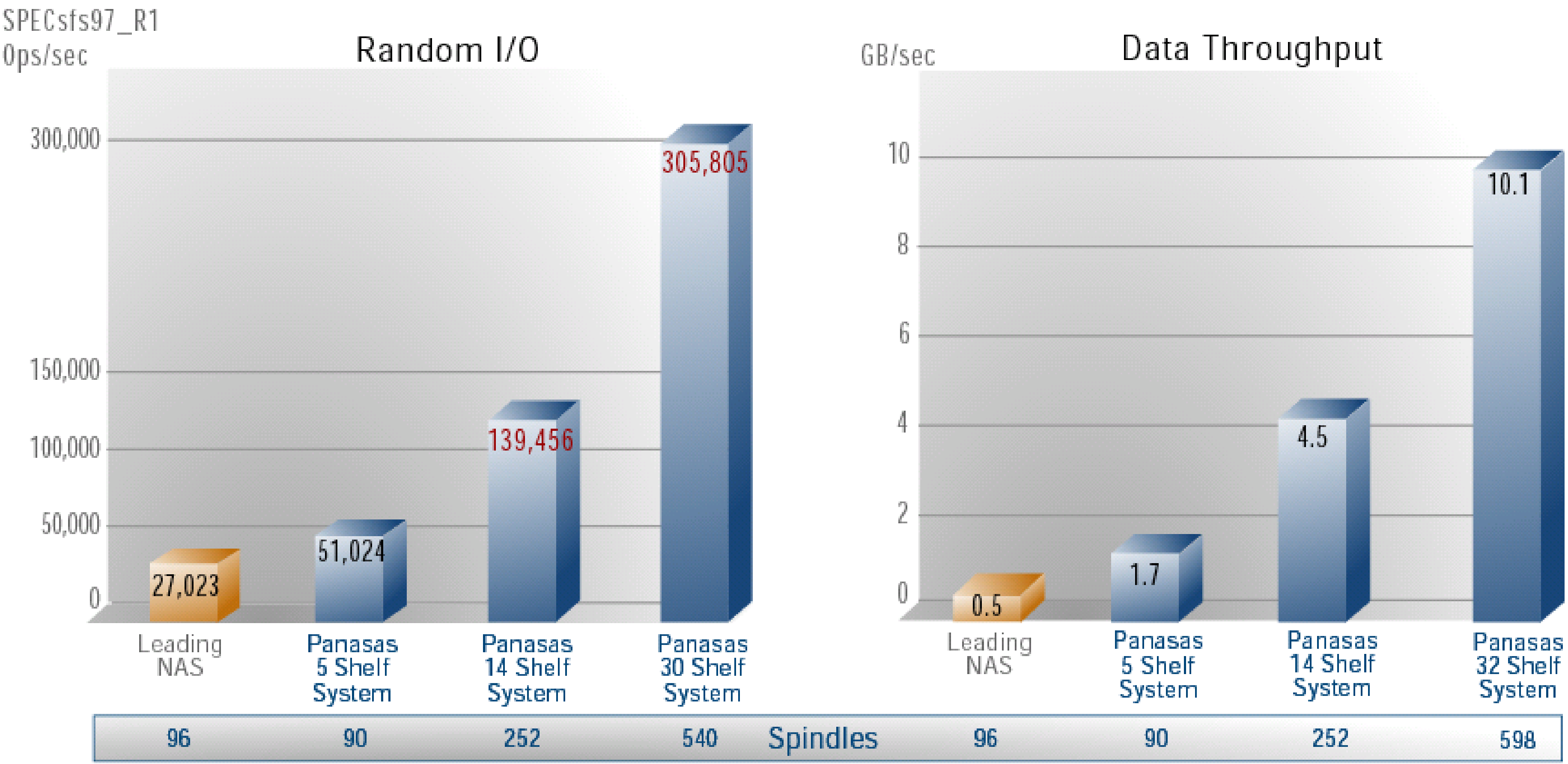
- **Thermal**
  - 800 Watts in 4u!
  - Power supplies and batteries have fans that cool the shelf
  - Blades, power supplies, batteries, network cards all monitor tempurature
  - Warnings generated near tempurature limit
  - Unilateral blade shutdown if a blade gets very hot
  - Graceful shutdown of a whole shelf if multiple blades are hot

Bladesets and Volumes

- **Bladeset is a storage (OSD) failure domain**
  - Single OSD failure results in degraded operation and reconstruction
  - Two OSD failures results in data unavailability
  - Bladesets can be expanded or merged (but not unmerged) for growth
  - Capacity balancing occurs within a bladeset
- **Volume is a file hierarchy with a quota**
  - One or more volumes compete for space within a bladeset
  - No physical boundaries between volumes, except quota limits
  - Volume is unit of DirectFlow metadata work
  - Each director blade manages one or more volumes
- **NFS/CIFS gateway workload is orthogonal to DirectFlow metadata**
  - All director blades provide uniform/symmetric NFS/CIFS access

Industry-Leading Performance

- **Breakthrough Random I/O AND Data Throughput**

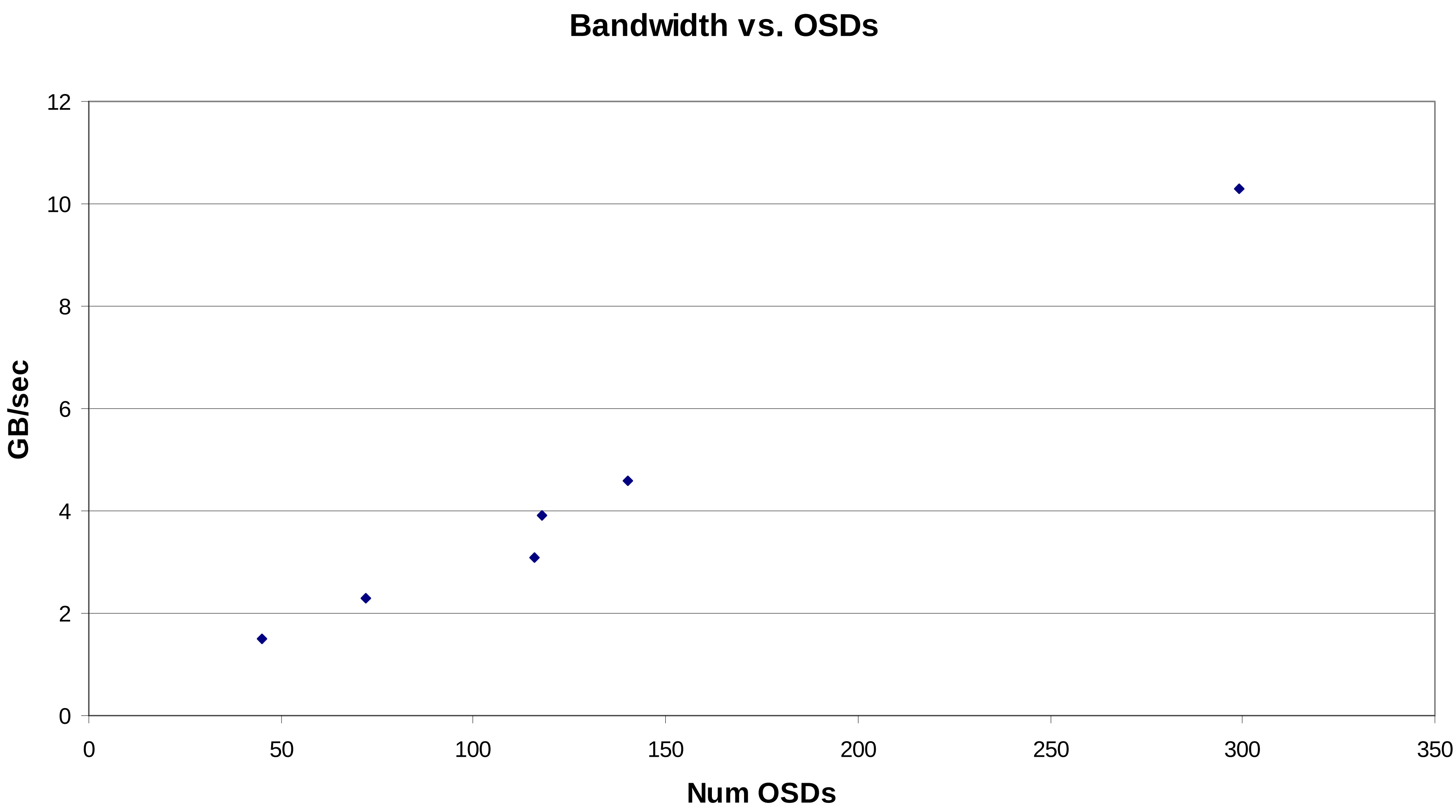


System performance scales linearly with capacity

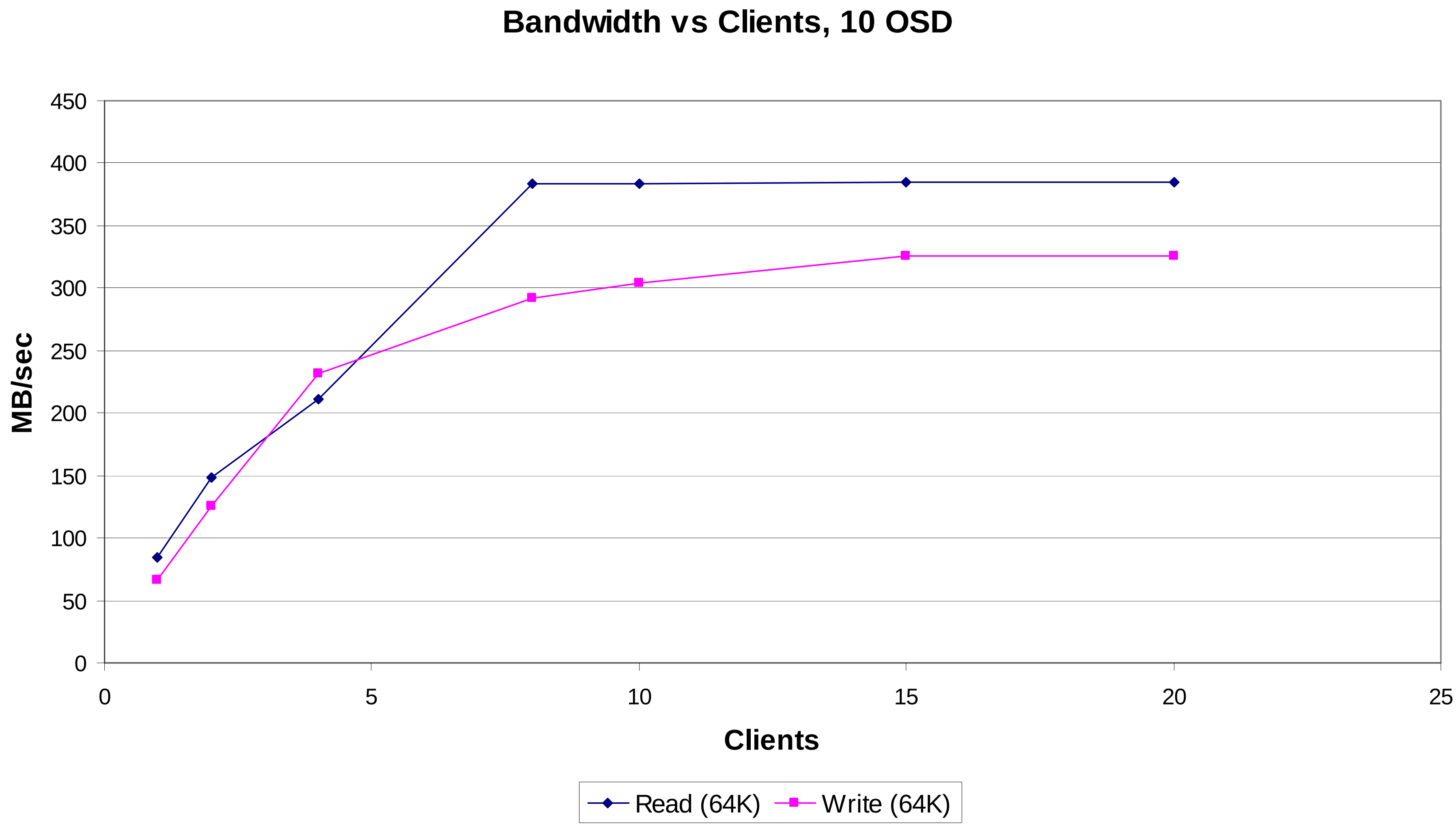
Bandwidth

- **Sustained Throughput 60 seconds, N clients to N files**
  - 1 Client, 10 OSDs: 95 MB/s read, 77 MB/s write
  - 10 Clients, 10 OSDs: 415 MB/s read, 335 MB/s write
  - 151 Clients, 299 OSDs: 10334 MB/s read
- **Barrier synchronized 1 TB move (MPI IO “min” time)**
  - 151 Clients, 299 OSDs: N to N, 7486 MB/s read, 6506 MB/s write
  - 151 Clients, 198 OSDs: 2775 MB/s concurrent write to one file
- **Clients are mostly 2.4 GHz uni-processors**
  - Large tests had a mix
  - Faster clients move data faster

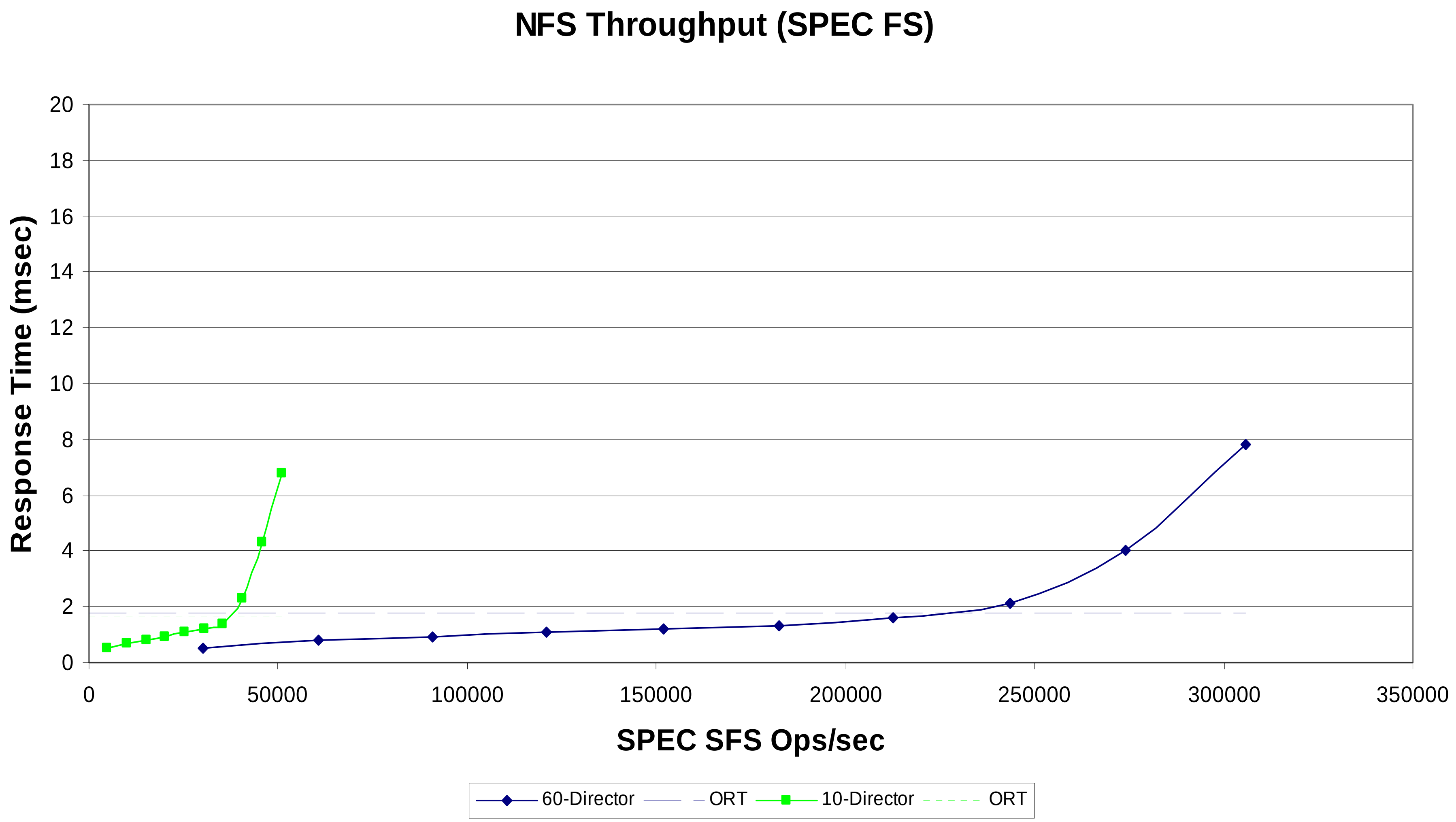
Scalable Bandwidth



Per Shelf Bandwidth



Scalable NFS



Strengths of Object Storage

- **Variable length data with layout metadata encapsulated at device**
- **With extensible attributes**
  - E.g. size, timestamps, ACLs, +
  - Some updated inline by device
  - Big enough to amortize object metadata
  - Small enough to share one access control decision
- **Metadata decisions are signed & cached at clients, enforced at device**
  - Rights and object map small relative to block allocation map
  - Clients can be untrusted (bugs & attacks expose only authorized object data)
  - Cache decisions & maps replaced transparently (dynamic remapping)
- **Command set works with SCSI architecture model (SAM)**
  - Encourages cost-effective implementation by storage device vendors

Object Architecture Momentum

- **Panasas helping lead industry adoption**
  - OSD Working group
    - 26 members including: Intel (**LEAD**), Panasas, HP, IBM, Veritas
  - NSIC NASD group chaired by Gibson
    - First Object-based storage standard draft
  - pNFS
    - Parallel I/O extension for NFSv4
  - Evangelizing through industry events
    - Technology benefits: functionality and architectural headroom
    - Business benefits: setting new price:performance metrics
- **Building on industry-wide acceptance**
  - EMC, IBM, Hitachi and Seagate endorsed during 2002

Backup

iSCSI: Storage over IP

- **iSCSI (internet Small Computer System Interface)**
  - SCSI sessions implemented over TCP/IP connections
  - Builds on stable and familiar standards (SCSI, Ethernet & TCP/IP)
  - Leverage Ethernet infrastructure to reduce TCO
- **OSD (Object Storage Devices)**
  - New SCSI command set for object storage
  - Working group to develop T10 standard

Opening Files & I/O

- **Mount**
  - At mount time, client learns the object ID of the root directory
  - Client asks DirectorBlade for capability to read directory and a callback to cache directory data
- **Pathname resolution**
  - Client iterates, checking its cache continually to see if it has capability to read directories, or the directory data itself
  - Client gets object ID for the file it wants and requests map and capability
  - Director checks ACL on the object, returns capability and RAID map
- **I/O**
  - Client does parallel I/O to all the storage nodes that store component objects
  - Storage nodes verify capability to enforce access control

Creating Files & Metadata

- **Creating a file**
  - Client asks metadata manager to create a file
  - Manager creates a pair of component objects
  - Manager updates file system directory, which is another object pair
  - Manager returns map and capabilities to client
  - Create file in 2.4 msec, delete a file in 1.9 msec, w/ distributed fault tolerance
- **Growing a file**
  - Small files (<64K) are mirrored on the first two component objects (RAID1)
  - Large files use additional component objects, up to a full stripe worth (RAID5)
  - Storage manager issues capabilities for data ranges, creating additional components and updating the file's map if necessary

Caching and Cache Consistency

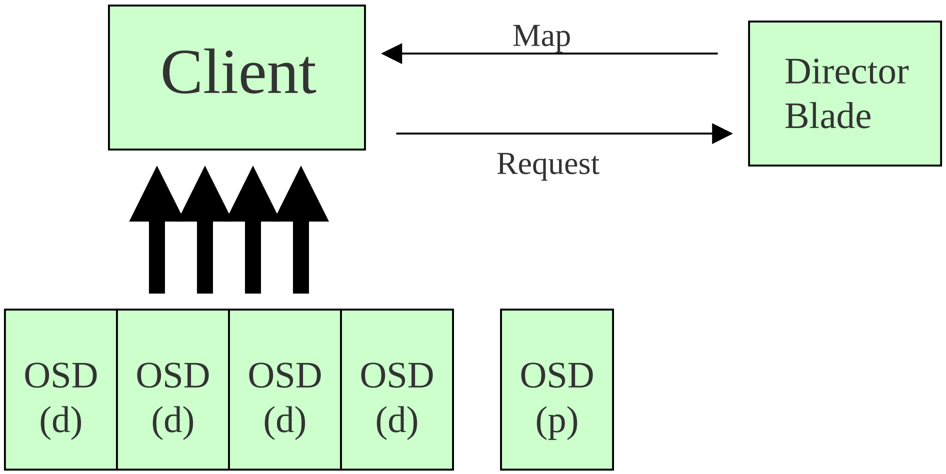
- **Caching information on client avoids interaction between DirectorBlades and StorageBlades**
  - Clients cache file data, directory data, attributes, and capabilities
- **DirectorBlades keep “callbacks”**
  - Promises to notify the client if cached data or attributes are invalid
- **Capabilities have a built-in expiration time**
  - Version number embedded in the cap that enables revocation

Scalability: Metadata

- Clustered servers (Director Blades) with active/active failover
- Block-level metadata controlled by Storage Blades (OSDs)
- Client caching with callbacks to reduce load for file-level metadata
- Metadata provides file system semantics over objects
- Chunk ownership over collections of files and directories
- For really large directories, hash into different collections
- Store metadata with the objects on storage nodes

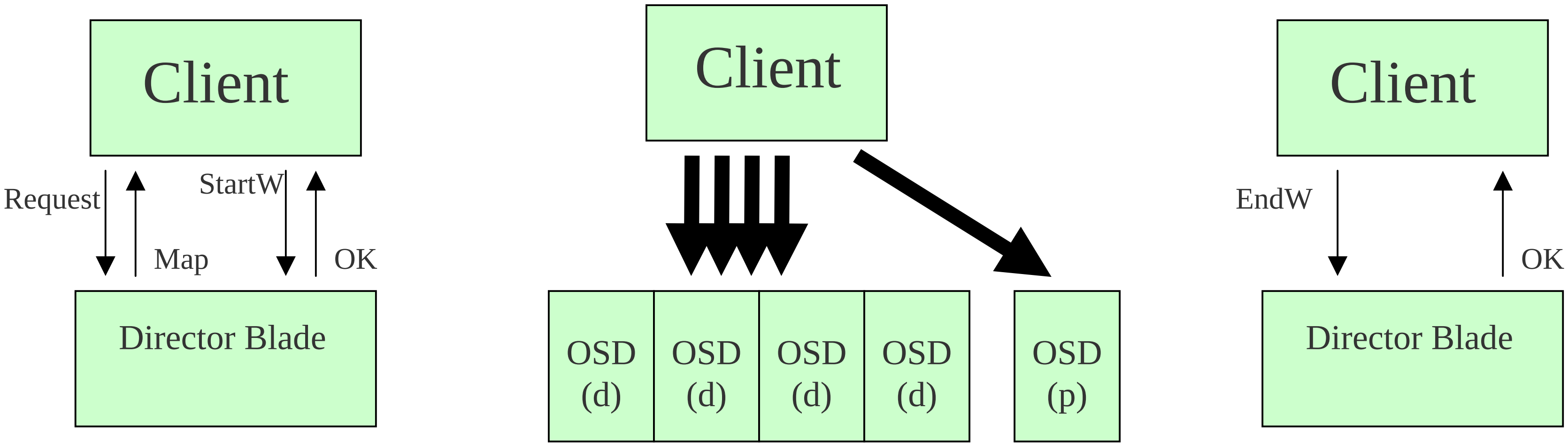
Client Read Operation

- **Steps in a client read operation**
  - Client determines file ID and responsible director from the directory entry
  - Client requests permission to read from the director
  - Director returns permission + file map identifying components
  - Client determines byte ranges within components and initiates a network transfer for each, all in parallel, ignoring the parity component
  - Client can re-use permission and map until told otherwise by director



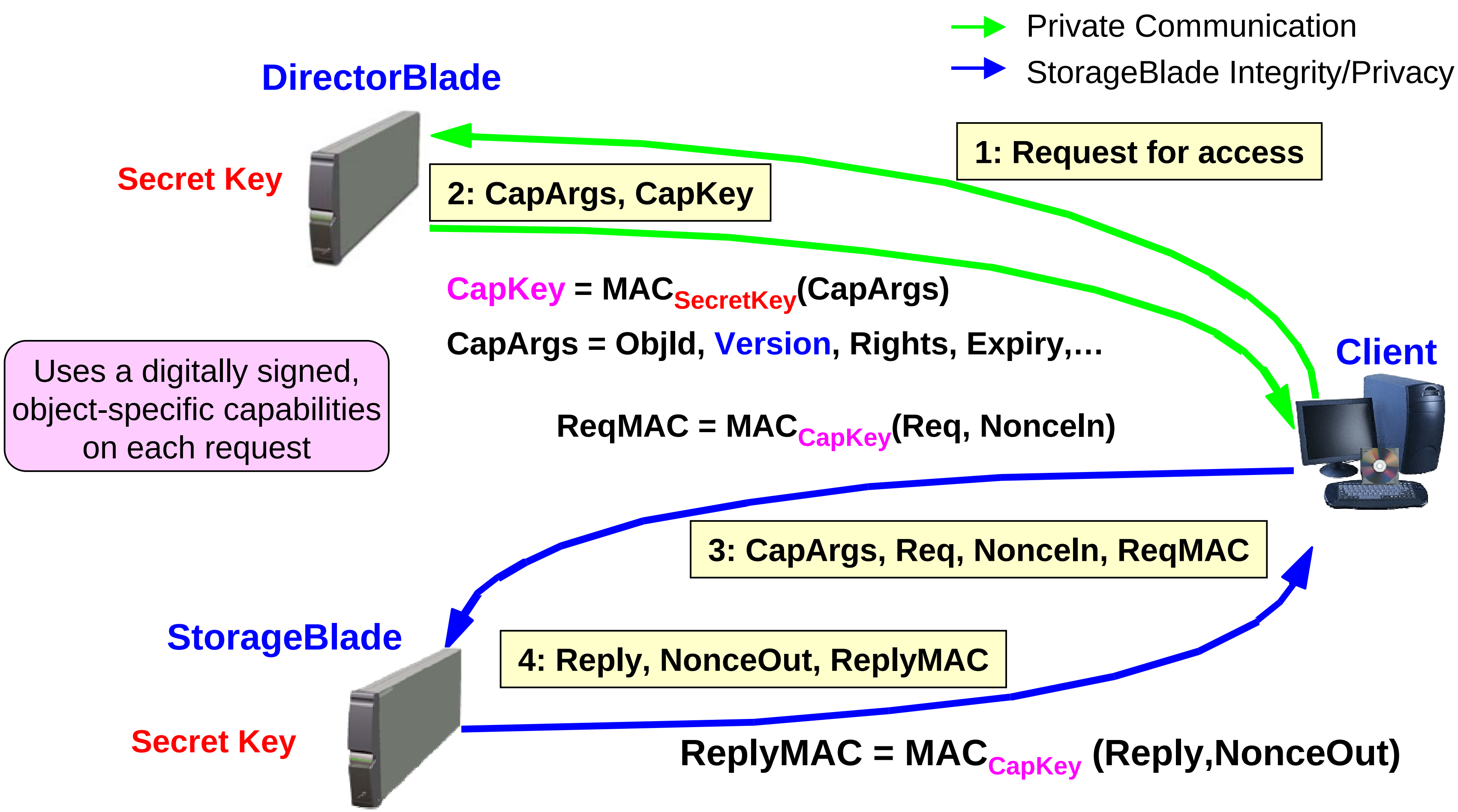
Client Write Operation

- Director must assure that concurrent writes do not corrupt parity, and that clients see coherent data in storage
- Therefore follow “start-write / end-write” policy
- Client accumulates “enough” dirty data to get high bandwidth transfer
- Client requests permission to write, writes all dirty data and updates parity, releases write permission back to the director
- Distinct from reads in that permission to read is long-lived



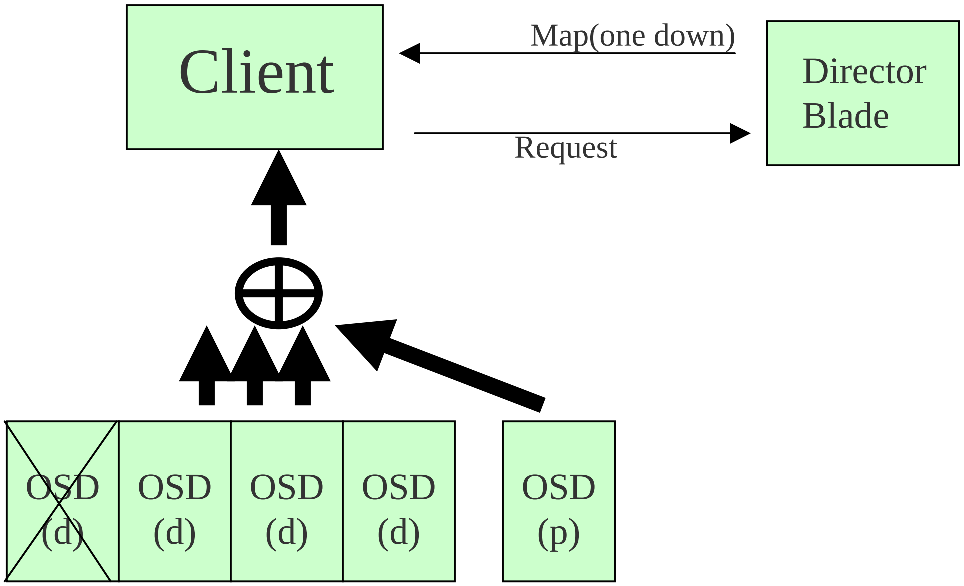
Access Enforcement

- **State of art is VPN of all out-of-band clients, all sharable data and metadata**
  - Accident prone & vulnerable to subverted client; analogy to single-address space computing



Reconstruction

- Failure of one OSD can be tolerated without data loss by using parity information
- Client reads all surviving data and computes missing information from parity
- Director can do the same thing and write all lost blocks to a new location in storage, thereby restoring the system to the fault-free state



Panasas and Lustre

- **Lustre OST**
  - Linux box plus stock RAID array (Fiber Channel)
  - Non-SCSI network protocol
  - Network -> Linux CPU -> RAID CPU(s) -> Many Drives
- **Panasas OSD**
  - Commodity parts, custom high density shelf w/ integrated UPS
  - T10 standards track iSCSI/OSD protocol, plus Panasas mgmt
  - Network -> Blade CPU -> 2 SATA Drives
- **Metadata**
  - Lustre – single (replicated) metadata server with database
  - Panasas – cluster of metadata servers, directories in objects
- **NFS/CIFS – Integrated w/ Panasas DirectorBlade**