Comparing Parallel File Systems

- Block Management
- How Metadata is stored
- What is cached, and where
- Fault tolerance mechanisms
- Management/Administration

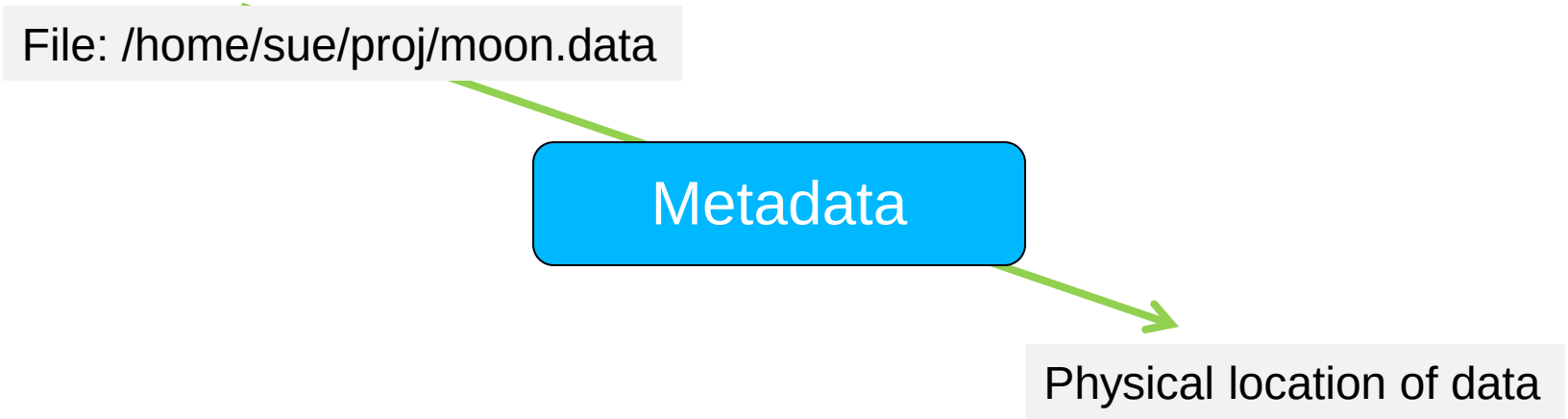
Designer cares about

- Performance
- Reliability
- Manageability
- Cost

Customer cares about

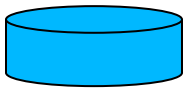
Meta Data Review

- Metadata names files and describes where they are located in the distributed system
  - Inodes hold attributes and point to data blocks
  - Directories map names to inodes
- Metadata updates can create performance problems
- Different approaches to metadata are illustrated via the File Create operation

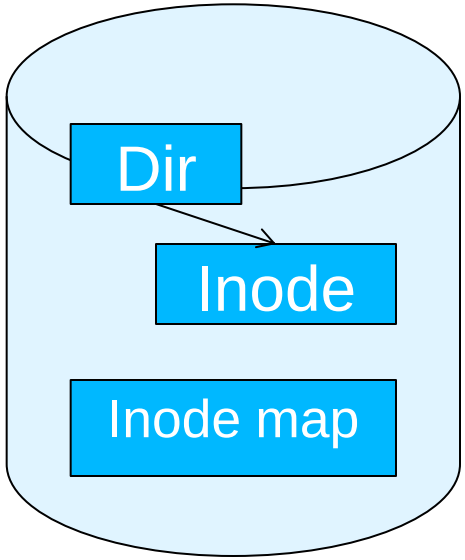


File Create on Local File System

- 4 logical I/Os
  - Journal update
  - Inode allocation
  - Directory insert
  - Inode update
- Performance determined by journal updates
  - Or lack there of – Journal protects integrity after a crash
  - Details vary among systems



Fast Journal device (SSD)

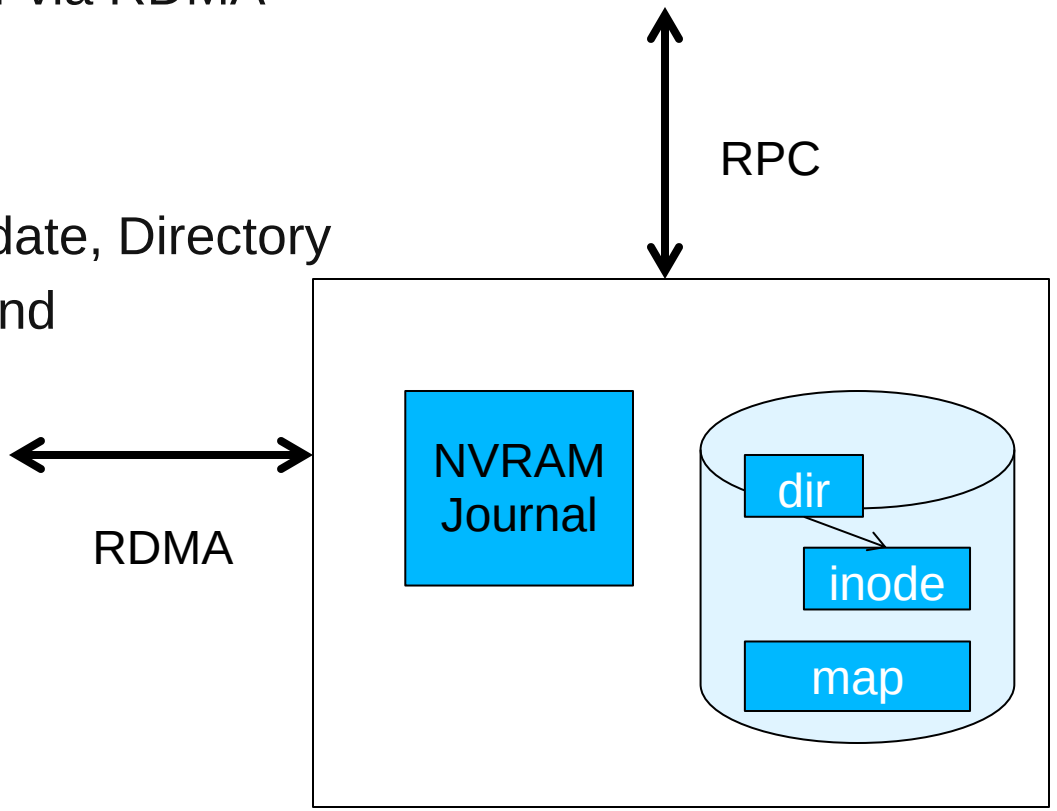


Inode	Name
0017	Fred
2981	Yoshi
7288	Racheta

Directory

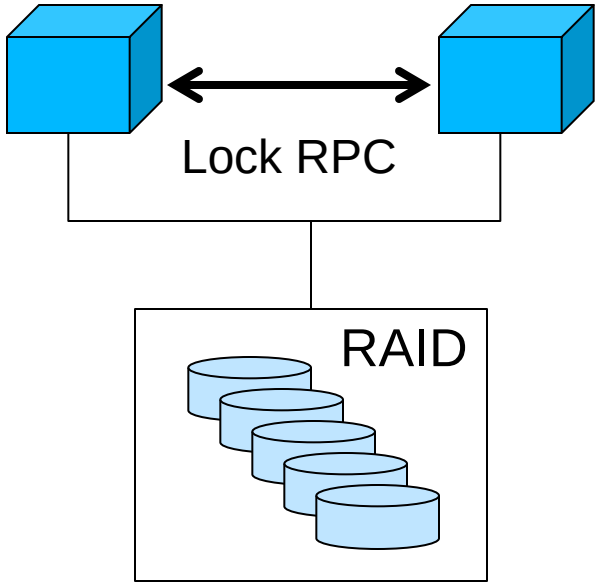
File Create on NFS Server

- RPC
  - Client to NFS server RPC
- NVRAM update
  - Mirrored copy on peer via RDMA
- Reply to client
- Local I/O
  - Inode alloc, Inode update, Directory
  - Done in the background
- Performance from
  - NVRAM+RDMA



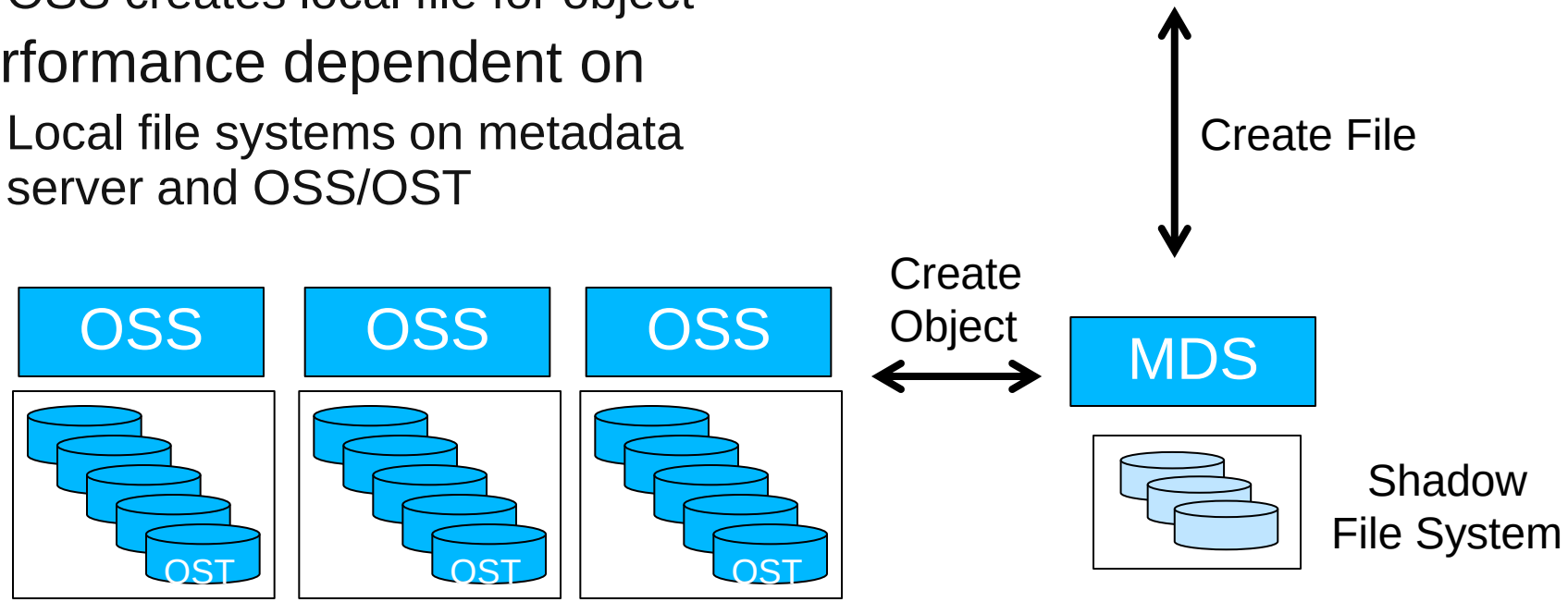
File Create on SAN FS

- Lock RPC for inode allocation
- Lock RPC for directory insert
- Journal update
- SAN I/O for inode and directory
  - Done in the background
- Performance dependent on
  - Journal updates
  - Lock manager updates
  - GPFS cache of lock ownership
  - SAN I/O



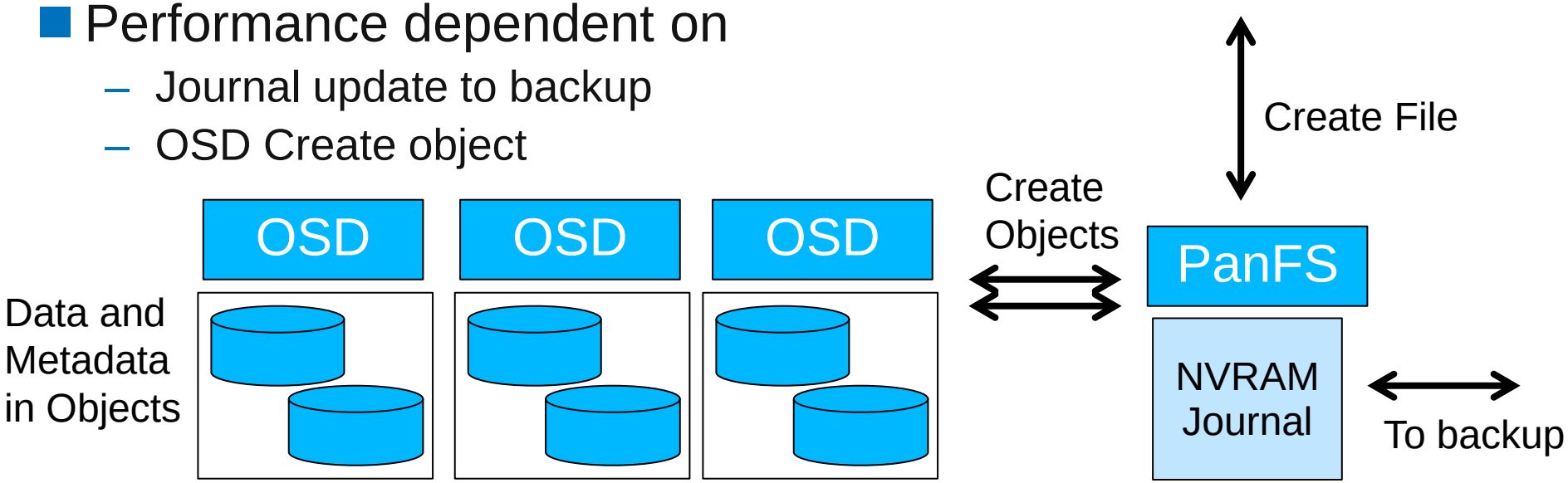
File Create on Lustre

- Client to Server RPC
- Server creates local file to store metadata
  - Journal update, local disk I/O
- Server creates container object(s)
  - Object create transaction with OSS
  - OSS creates local file for object
- Performance dependent on
  - Local file systems on metadata server and OSS/OST



File Create on PanFS

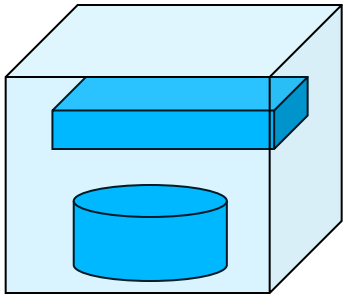
- Client to Server RPC
- MDS updates journal in NVRAM (locally and on backup)
- MDS creates 2 container objects (iSCSI/OSD Create Object)
  - OSDFS journals object create in NVRAM
  - MDS annotates objects with its own metadata (as attributes)
- Reply to client
- Update directory (mirrored OSD write) in background
- Performance dependent on
  - Journal update to backup
  - OSD Create object



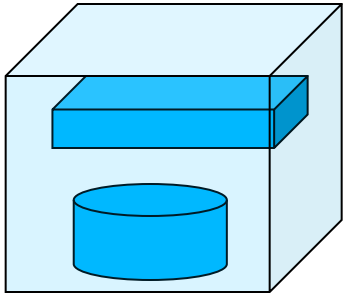
File Create on HDFS

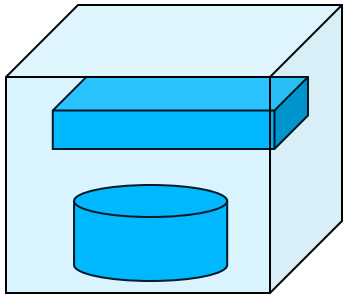
- RPC to Name Node
- Key-value store update
- Container Create
  - One on the client node
  - One replica “in rack”
  - One more replica “out of rack”
- Performance depends on
  - Metadata memory size
  - Local file system updates on Data Nodes
  - # Copies created before sending reply

Client and Local Copy



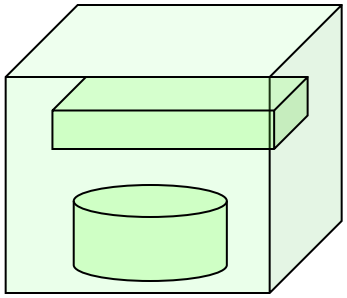
Remote Copy1



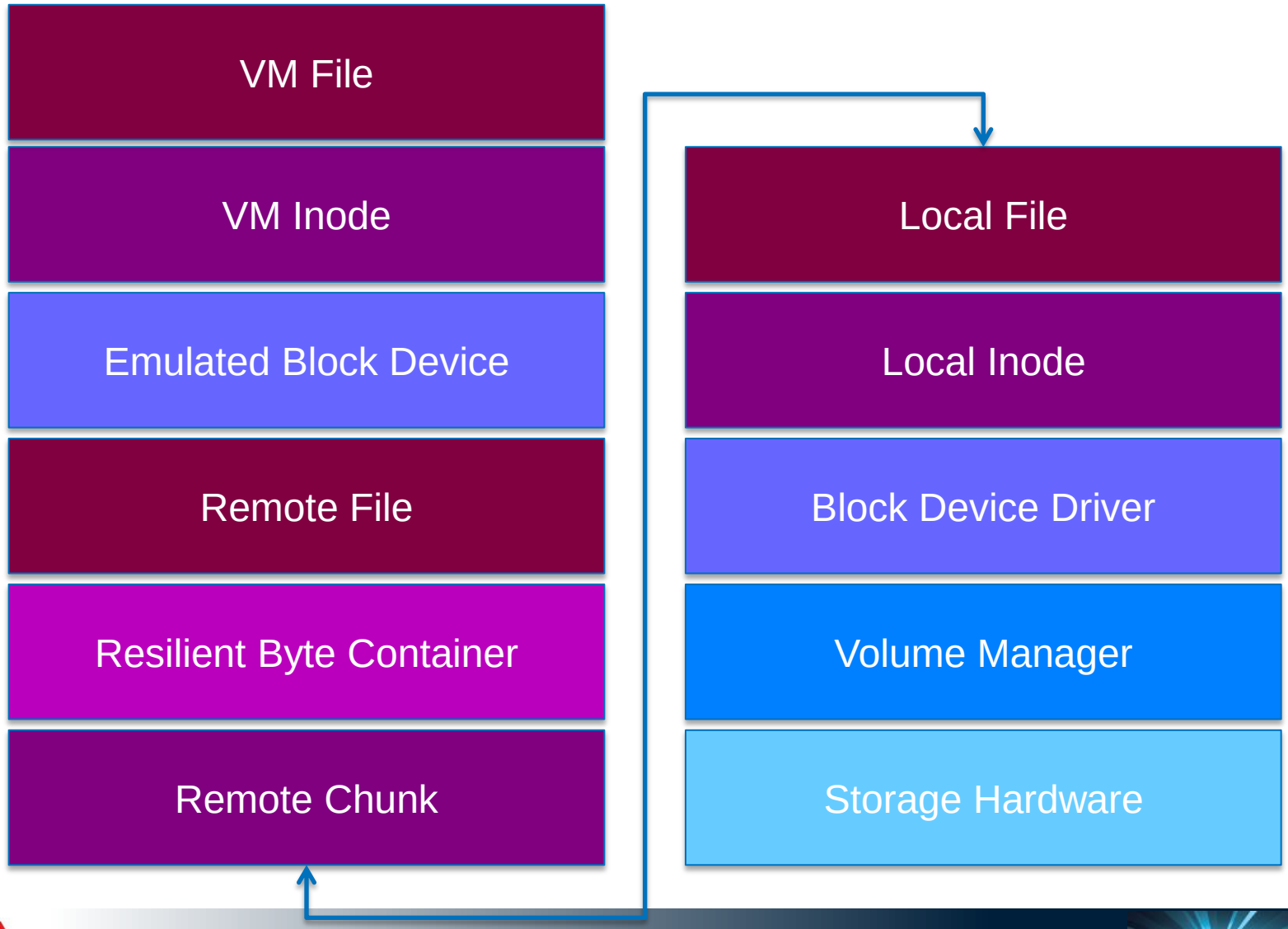


Remote Copy2

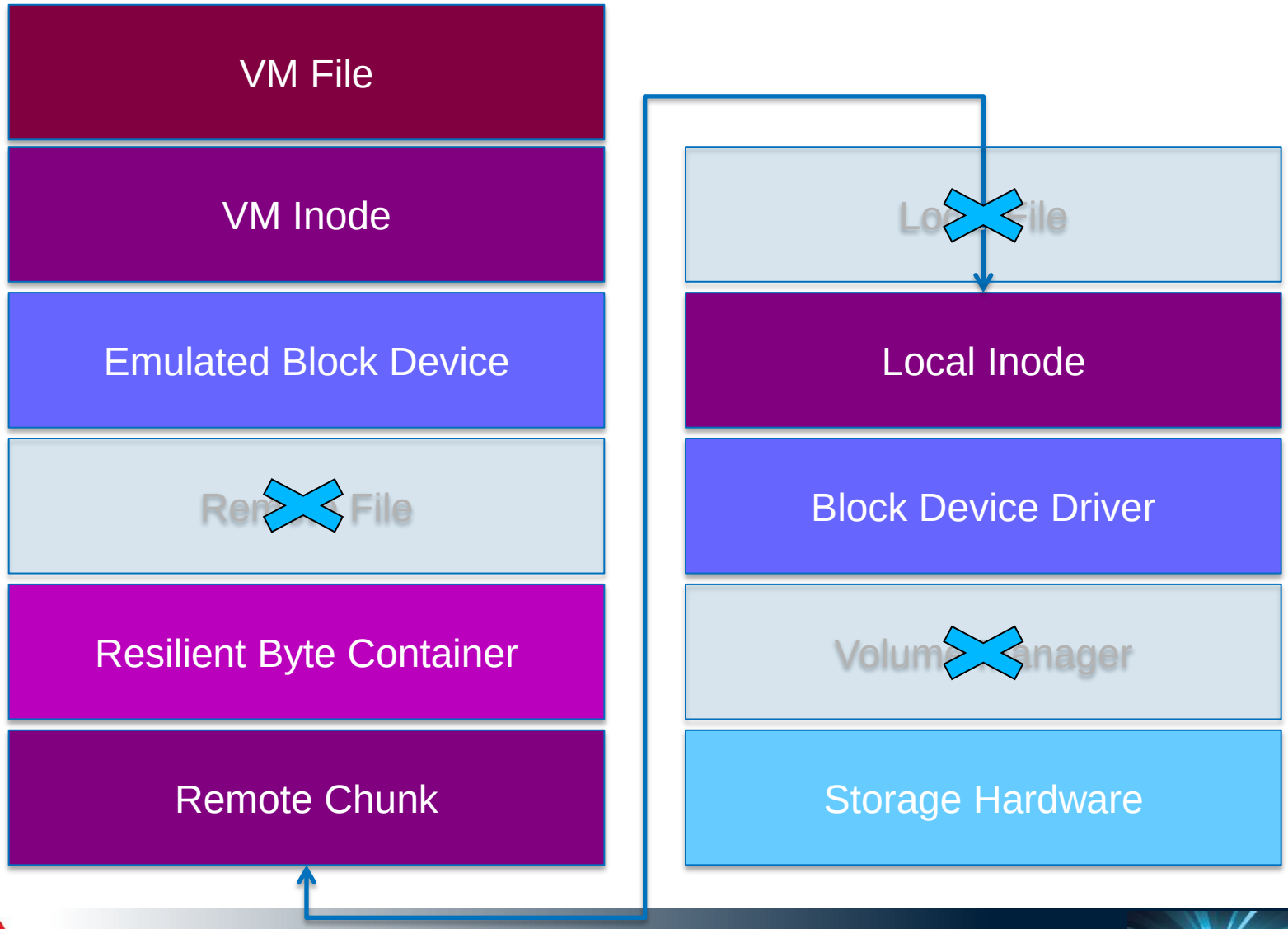
Name Node



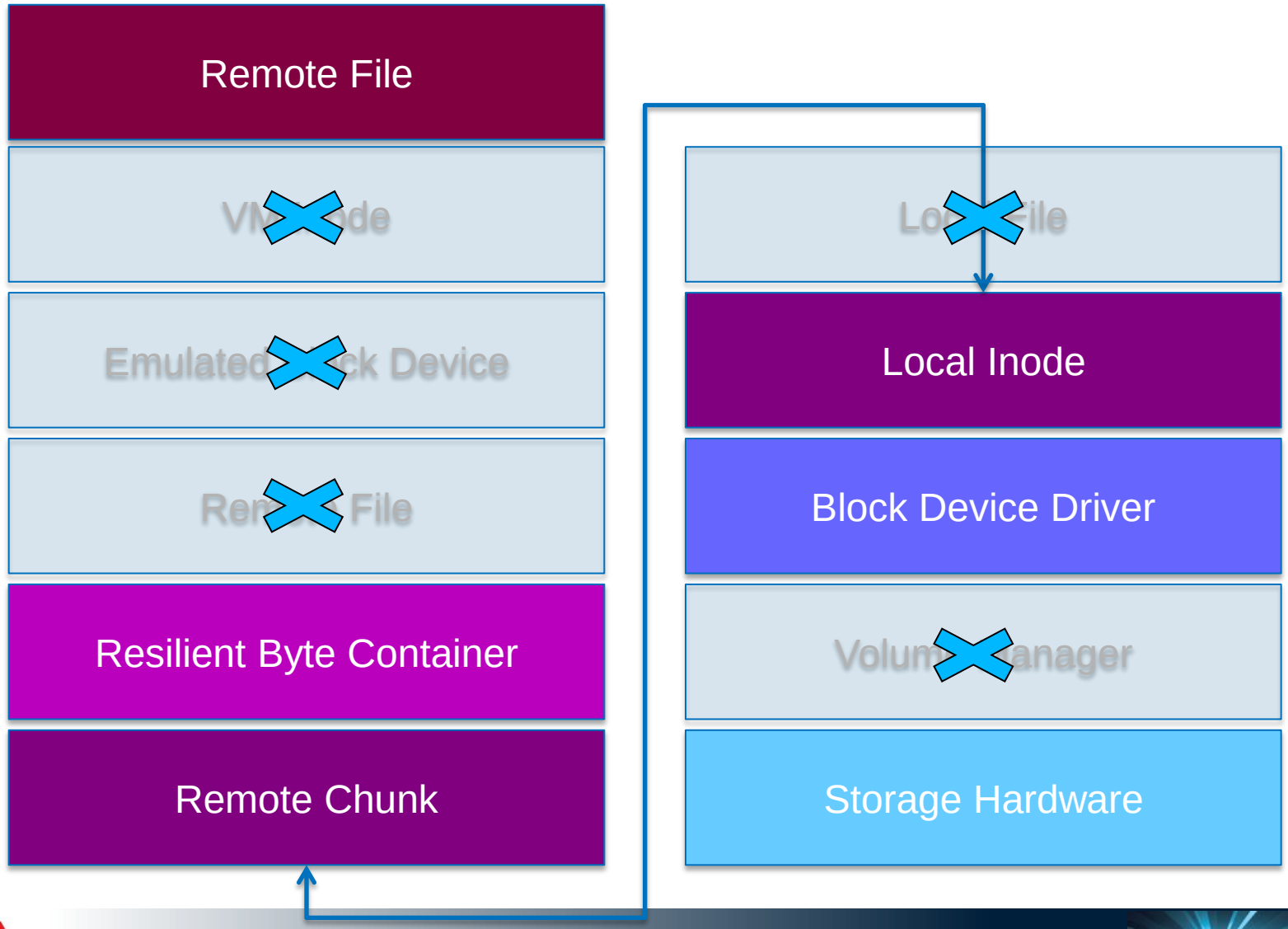
Layers upon Layers



Layers upon Layers (Optimized, e.g., VMFS)



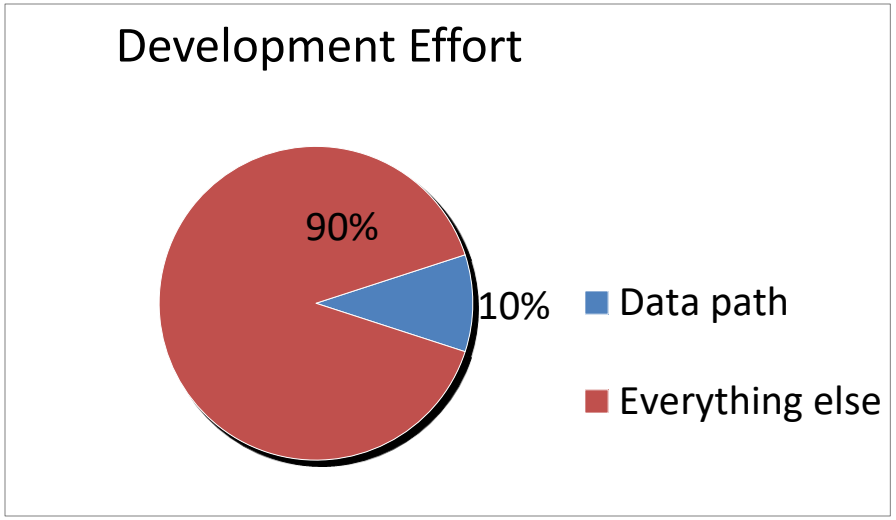
Layers upon Layers (Optimized, e.g. PanFS)



Other Issues

What about…

- Monitoring & troubleshooting?
- Backups?
- Snapshots?
- Disaster recovery & replication?
- Capacity management?
- System expansion?
- Retiring old equipment?
- Limitations of POSIX?



Things to remember

- Silicon bits cost 10x Magnetic bits
- Scale for performance demands reliability
- Parallel declustered data recovery
- Recurring pattern of layered storage abstractions
- Data vs. Metadata
- The power of smart clients and middleware