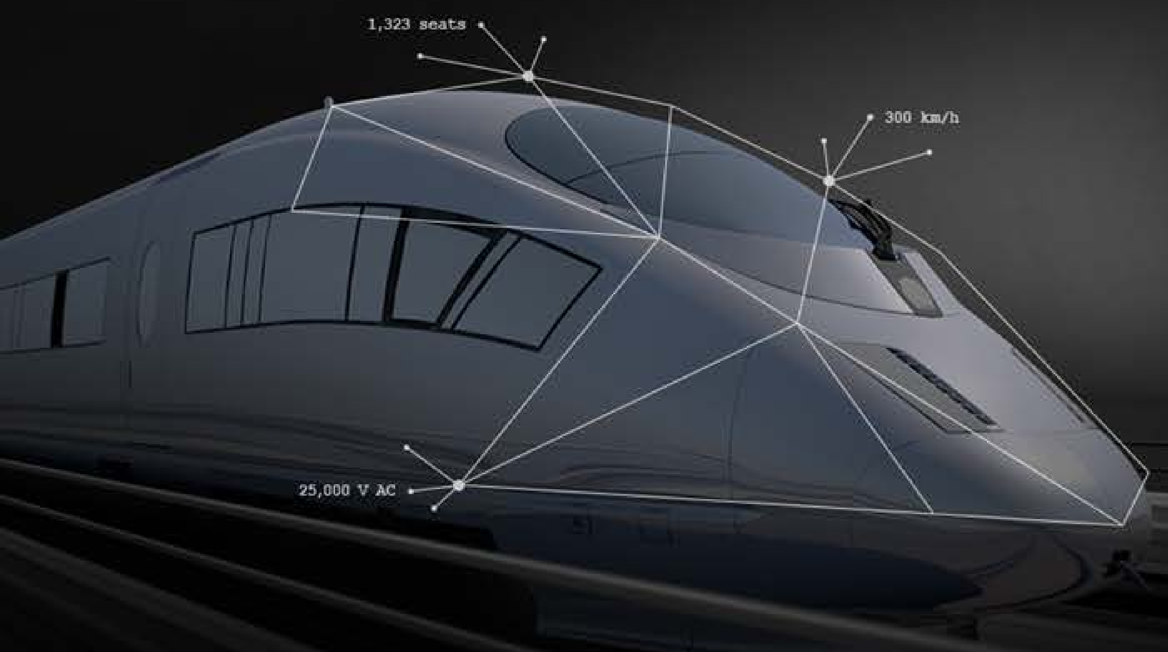
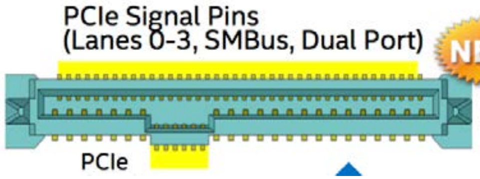


The Future of Software-Defined Storage - What Does It Look Like in 3 Years Time?

Richard McDougall, VMware, Inc

This Talk: The Future of SDS

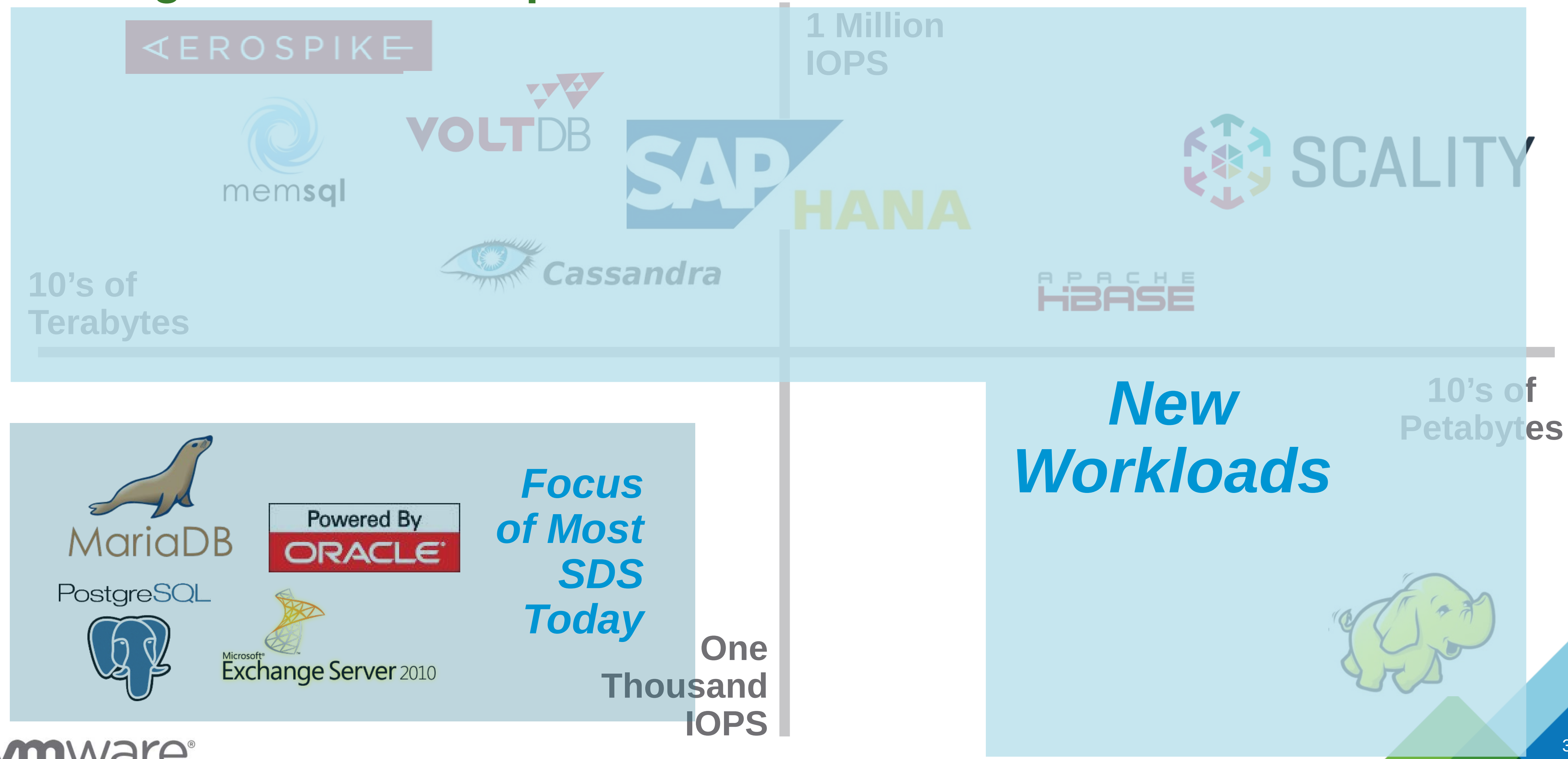


Hardware
Futures & Trends

Emerging Cloud Native Platforms

New Workloads

Storage Workload Map



What Are Cloud-Native Applications?

Developer access via APIs

Continuous integration and deployment

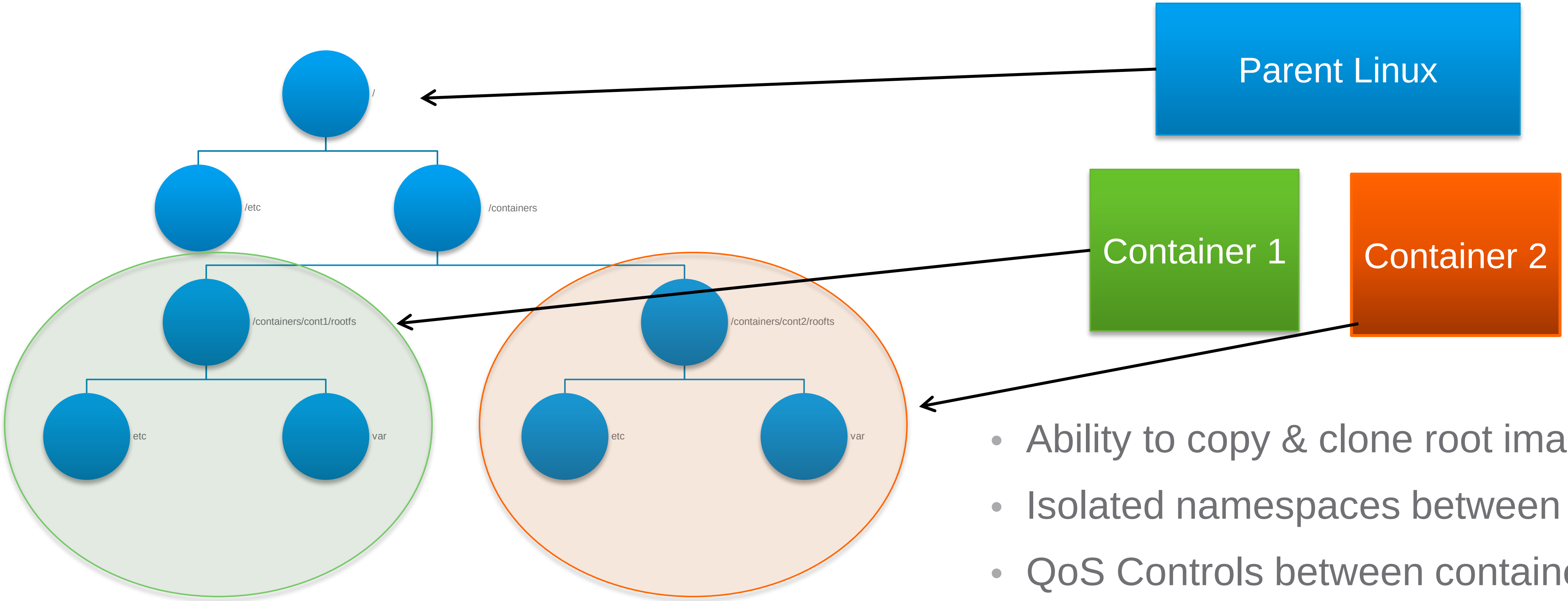
Built for scale

Availability architected in the app, not hypervisor

Microservices, not monolithic stacks

Decoupled from infrastructure

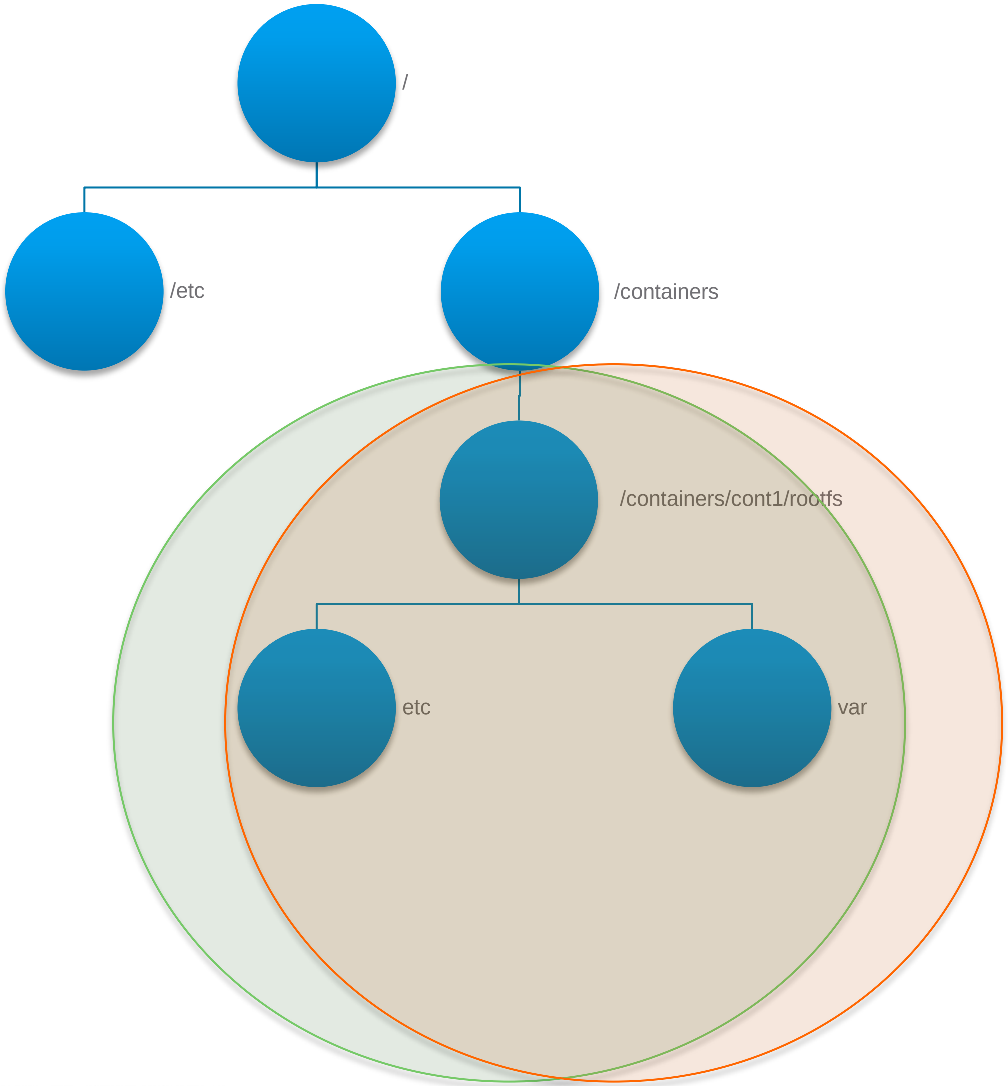
What do Linux Containers Need from Storage?



- Ability to copy & clone root images
- Isolated namespaces between containers
- QoS Controls between containers

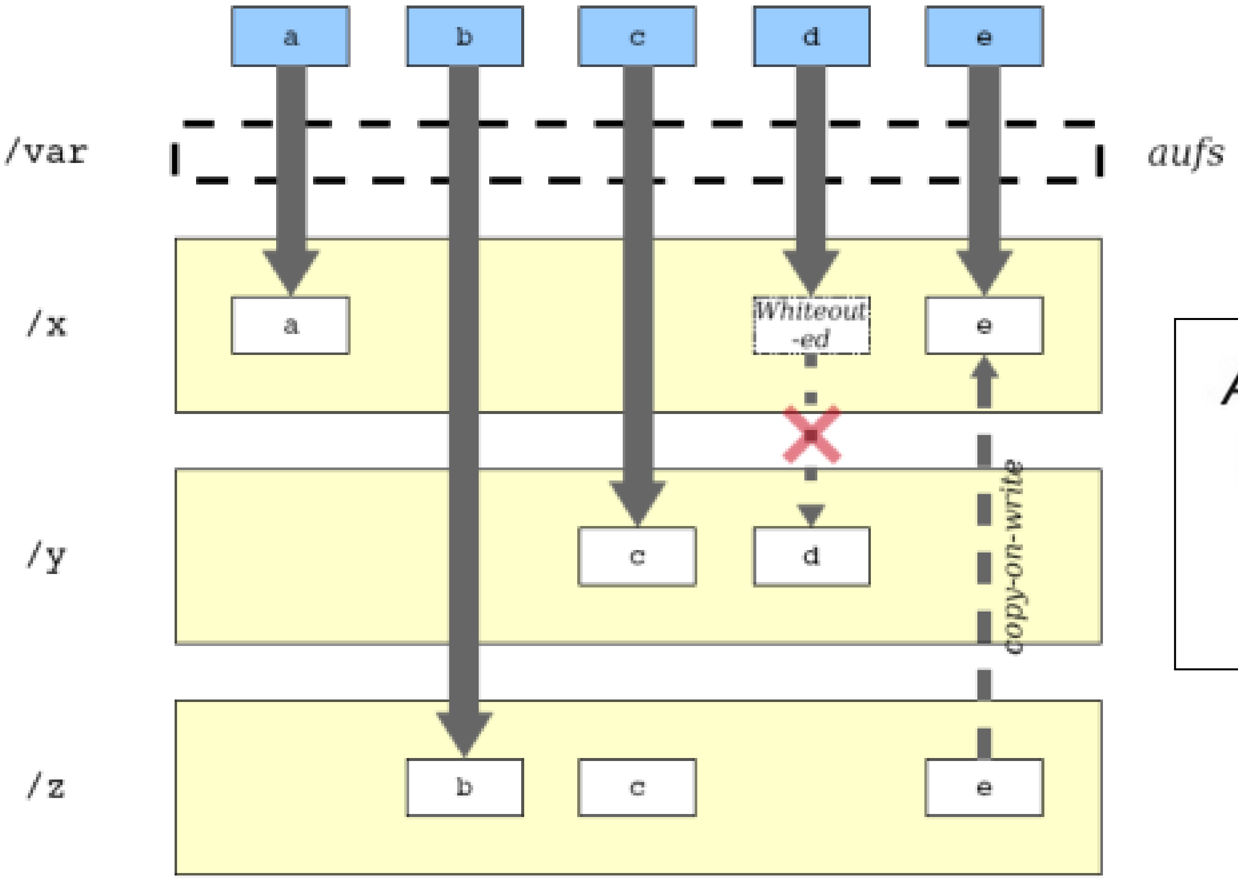
Option A: copy whole root tree

Containers and Fast Clone Using Shared Read-Only Images



- Actions
  - Quick update of child to use same boot image
- Downsides:
  - Container root fs not writable

Docker Solution: Clone via “Another Union File System” (aufs)

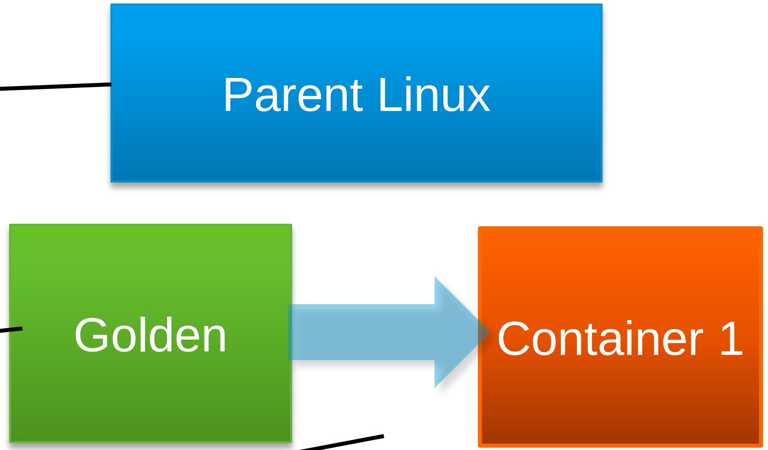


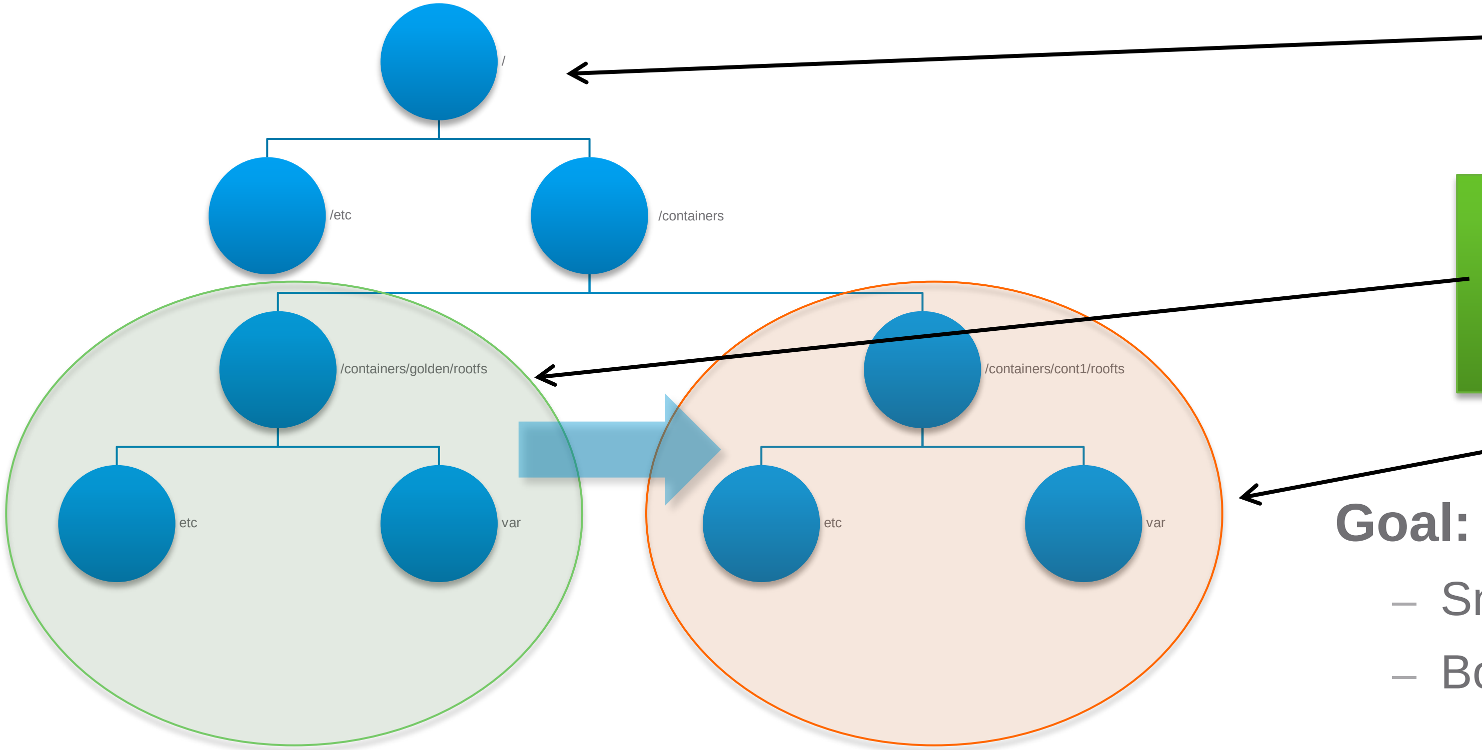
Except:

AUFS is the worst union filesystems out there; except for all the others that have been tried.

Not Churchill

Docker's Direction: Leverage native copy-on-write filesystem



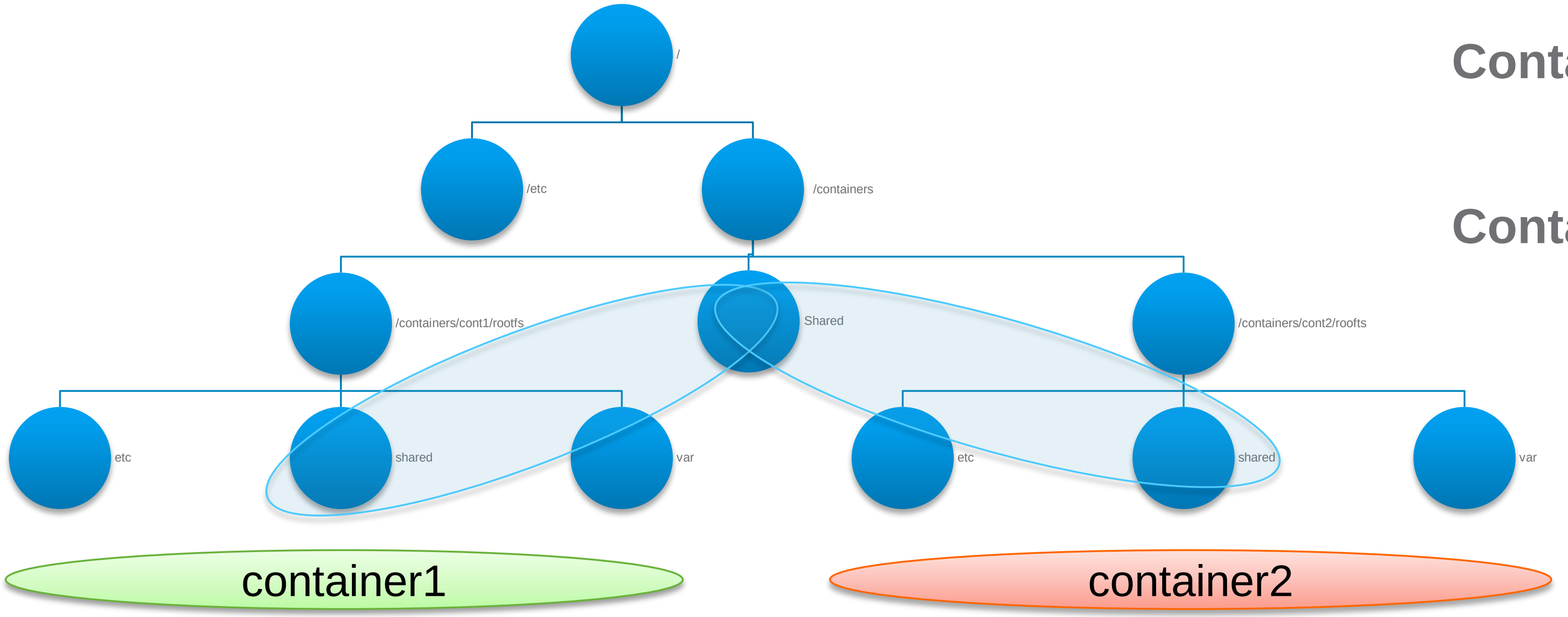


Goal:

- Snap single image into writable clones
- Boot clones as isolated linux instances

 # zfs snapshot cpool/golden@base
 # zfs clone cpool/root@base cpool/cont1

Shared Data: Containers can Share Filesystem Within Host



Containers Today:

Share within the host

Containers Future:

Share across hosts

Clustered File Systems

Docker Storage Abstractions for Containers

Docker Container

Persistent Data

/data

"VOLUME /var/lib/mysql"

Local Disks

Non-persistent boot environment

/

/etc

/var

Docker Storage Abstractions for Containers

Docker
Container

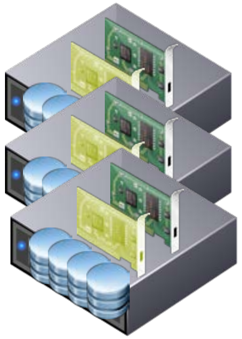
Persistent Data

/data

“VOLUME /var/lib/mysql”

Block Volumes

VSAN
CEPH RBD
ScaleIO
Flocker
Etc,…



Non-persistent boot environment

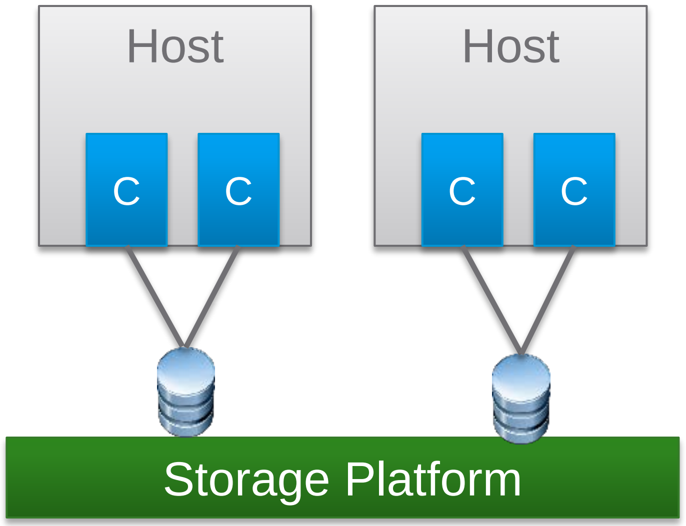
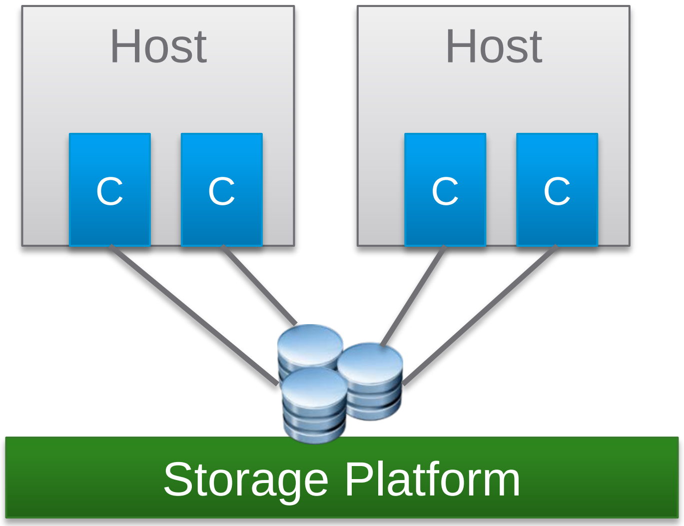
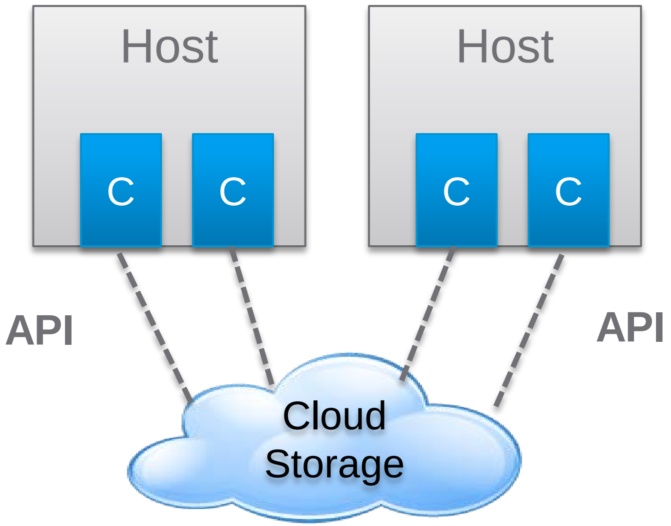
/

/etc

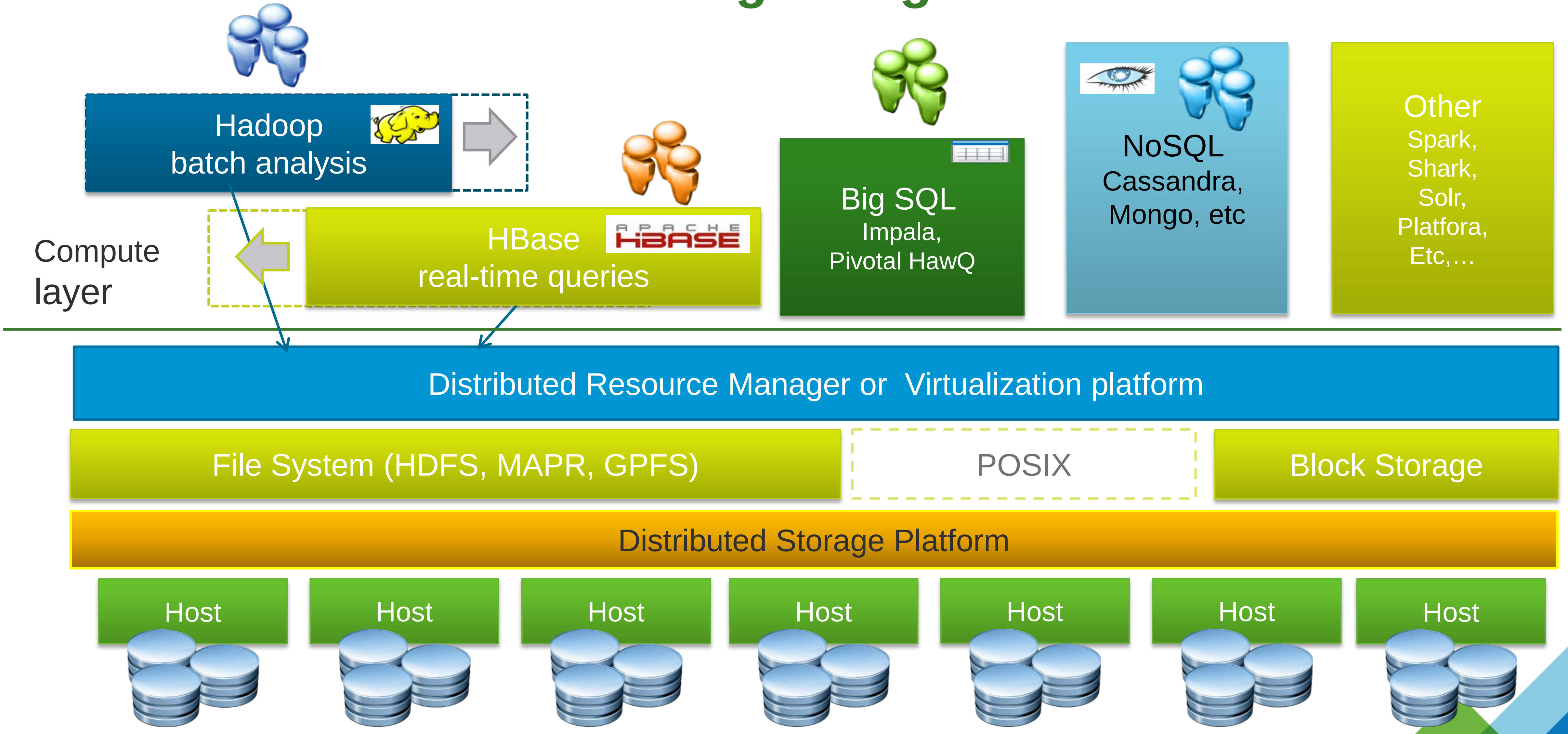
/var

vmware®

Container Storage Use Cases

Unshared Volumes	Shared Volumes	Persist to External Storage
		
Use Case: Running container-based SQL or noSQL DB	**Use Case:** Sharing a set of tools or content across app instances	**Use Case:** Object store for retention/archival, DBaaS for config/transactions

Eliminate the Silos: Converged Big Data Platform



Storage Platforms

Array Technologies

10 Million IOPS

10's of Terabytes

10's of Petabytes

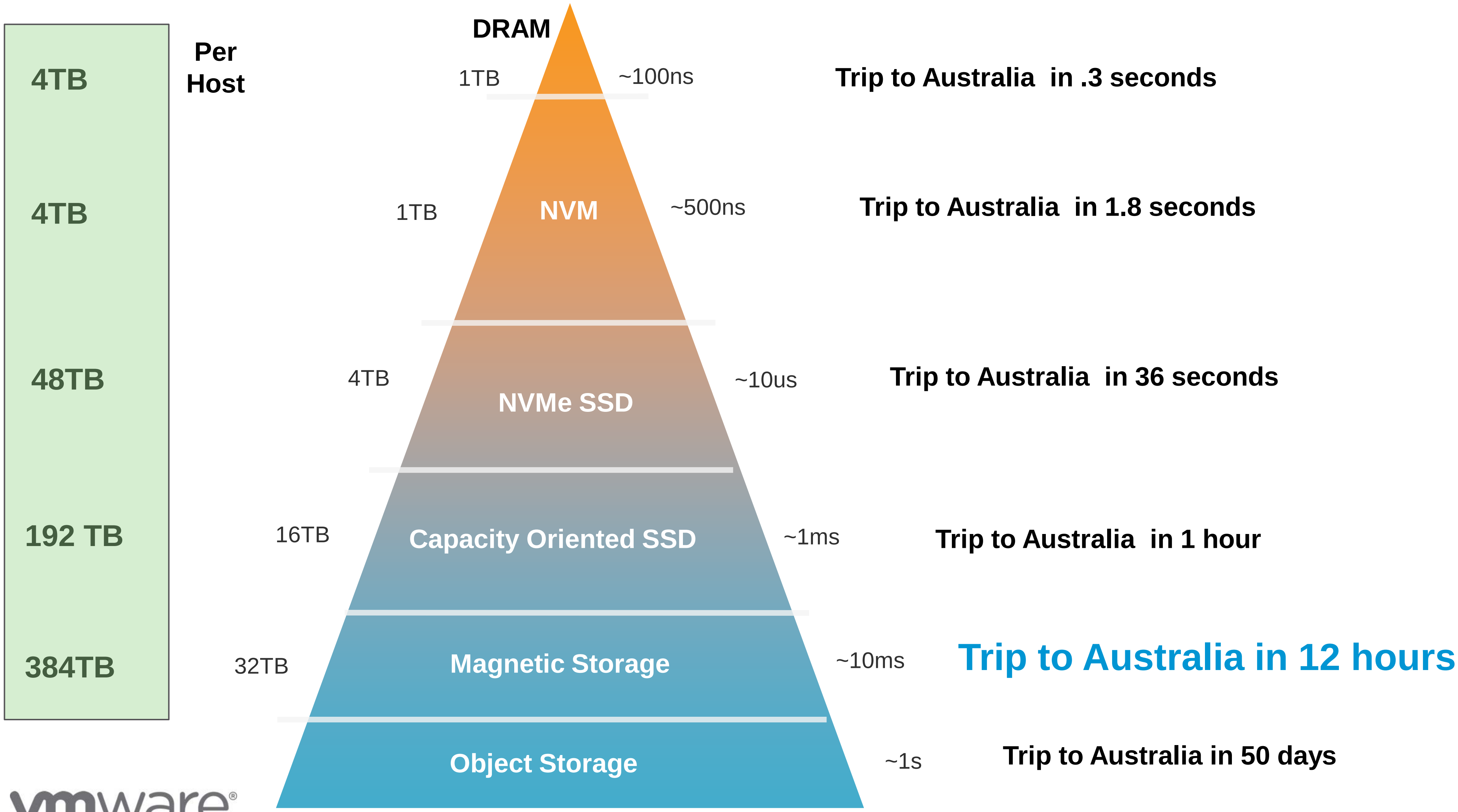
Consolidate

and

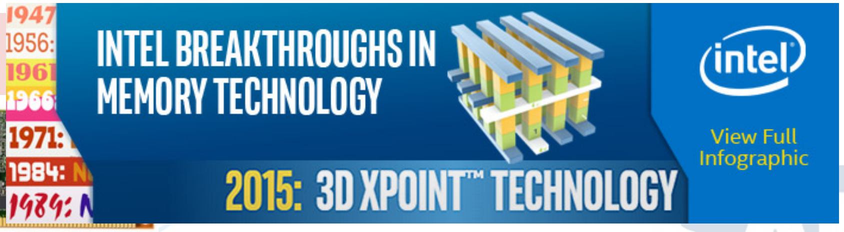
Increase Performance

One Thousand IOPS

Storage Media Technology: Capacities & Latencies – 2016



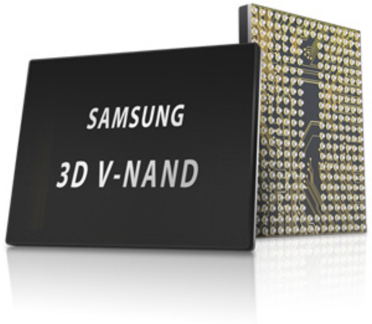
Storage Media Technology

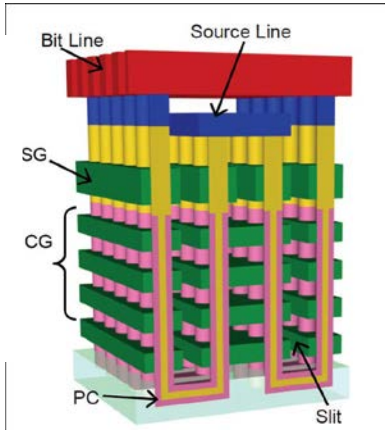


1 Million
IOPS

2D NAND
“IOPS Flash”

10’s of
Gigabytes



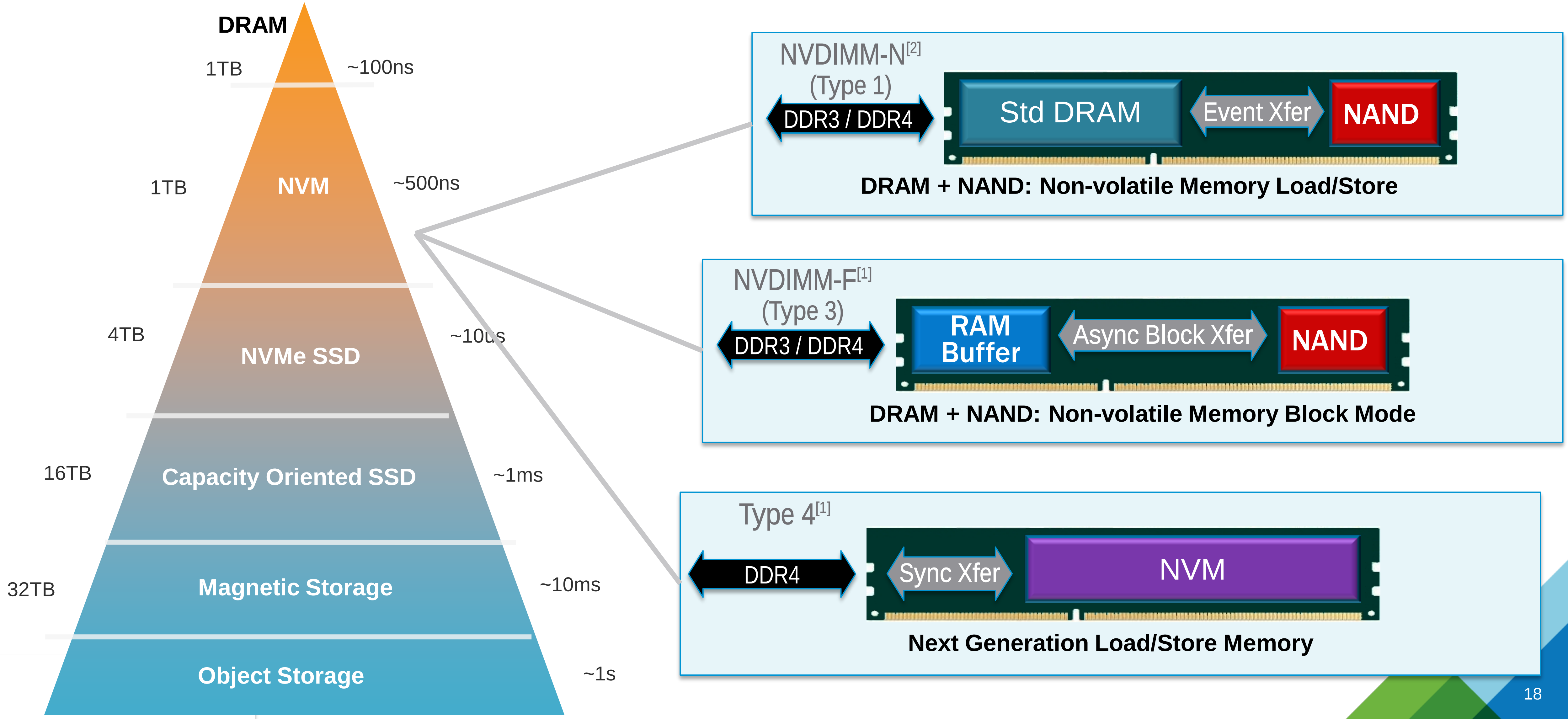


10’s of
Terabytes

3D NAND
“Capacity Flash”

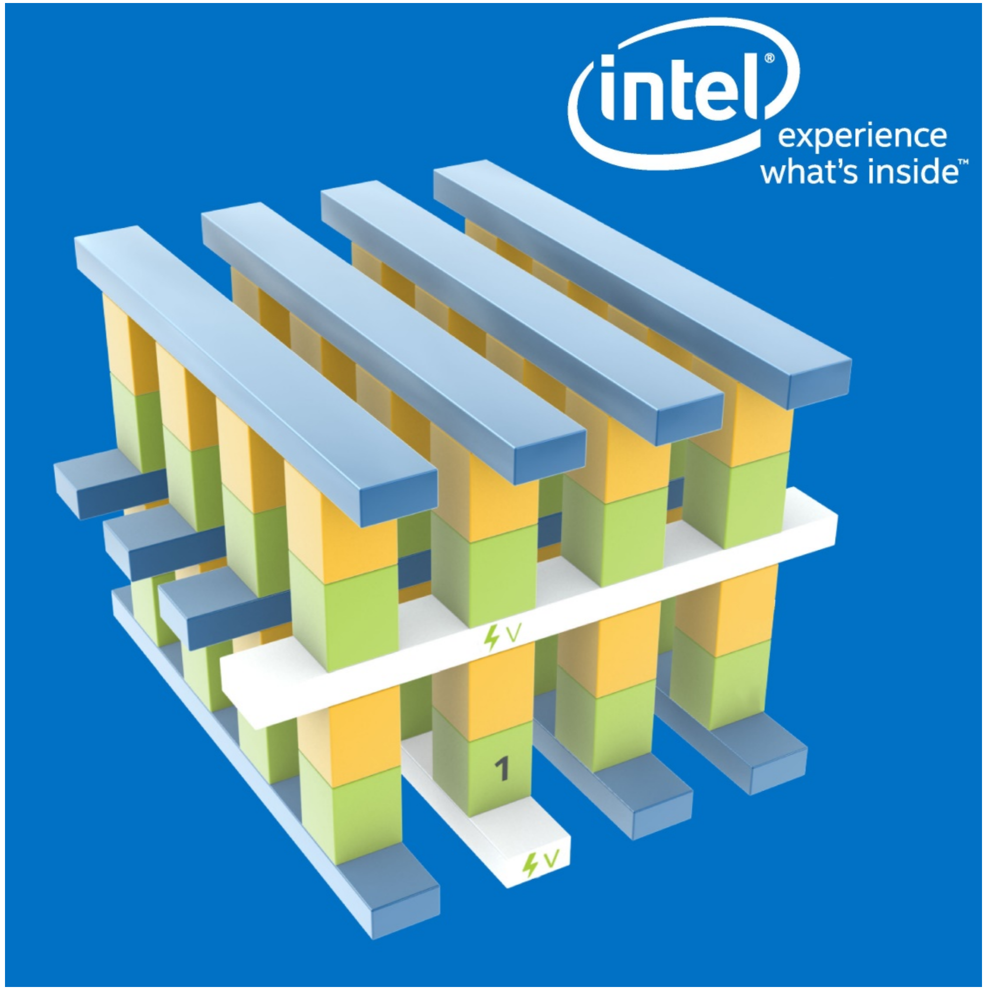
One
Thousand
IOPS

NVDIMM’s


DRAM
1TB
~100ns
1TB
NVM
~500ns
4TB
NVMe SSD
~10us
16TB
Capacity Oriented SSD
~1ms
32TB
Magnetic Storage
~10ms
Object Storage
~1s
NVDIMM-N[2]
(Type 1)
DDR3 / DDR4
Std DRAM
Event Xfer
NAND
DRAM + NAND: Non-volatile Memory Load/Store
NVDIMM-F[1]
(Type 3)
DDR3 / DDR4
RAM Buffer
Async Block Xfer
NAND
DRAM + NAND: Non-volatile Memory Block Mode
Type 4[1]
DDR4
Sync Xfer
NVM
Next Generation Load/Store Memory

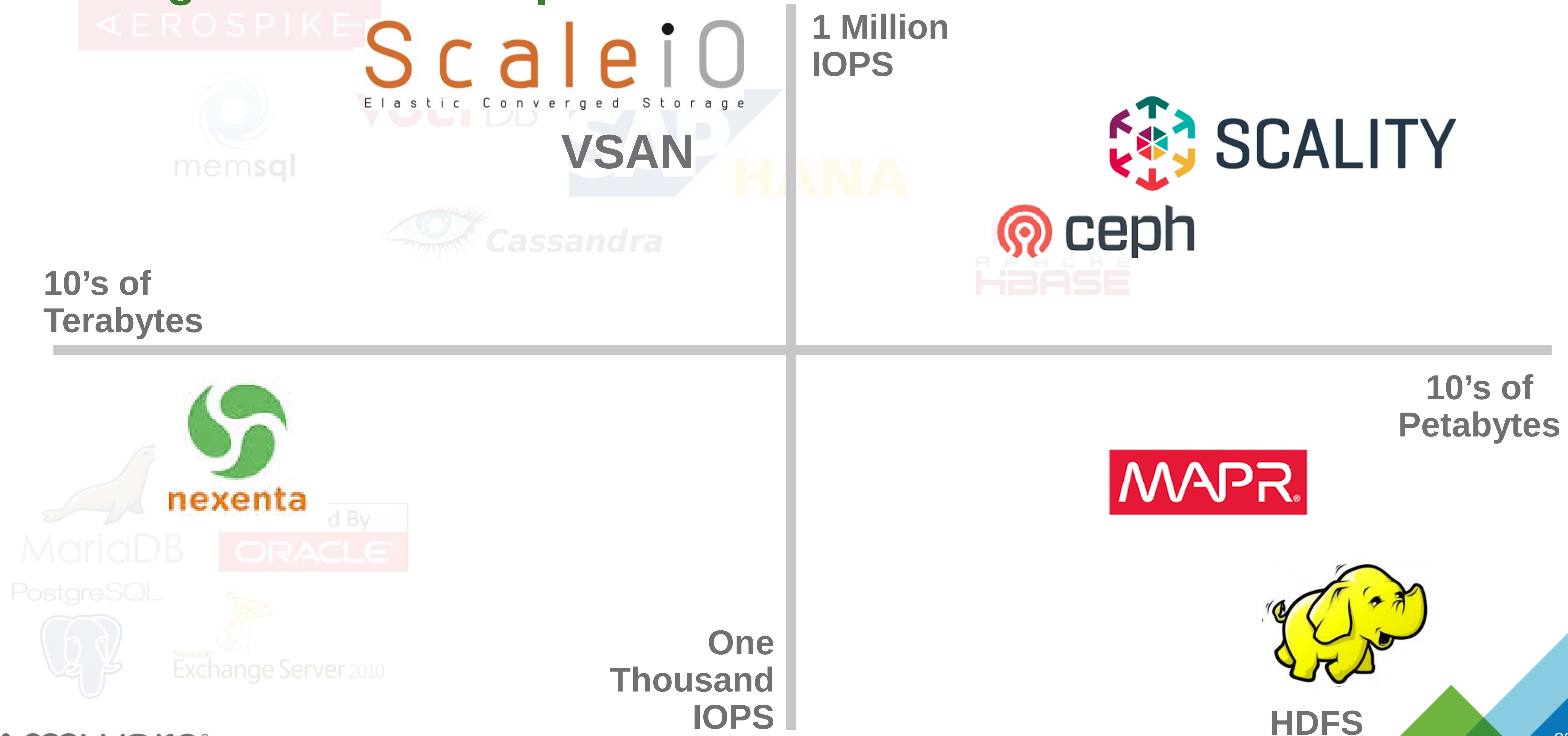
Intel 3D XPoint™ Technology

- Non-Volatile
- 1000x higher write endurance than NAND
- 1000x Faster than NAND
- 10x Denser than DRAM
- Can serve as system memory and storage
- Cross Point Structure: allows individual memory cell to be addressed
- Cacheline load/store should be possible.
- Stackable to boost density
- Memory cell can be read or written w/o requiring a transistor, reducing power & cost

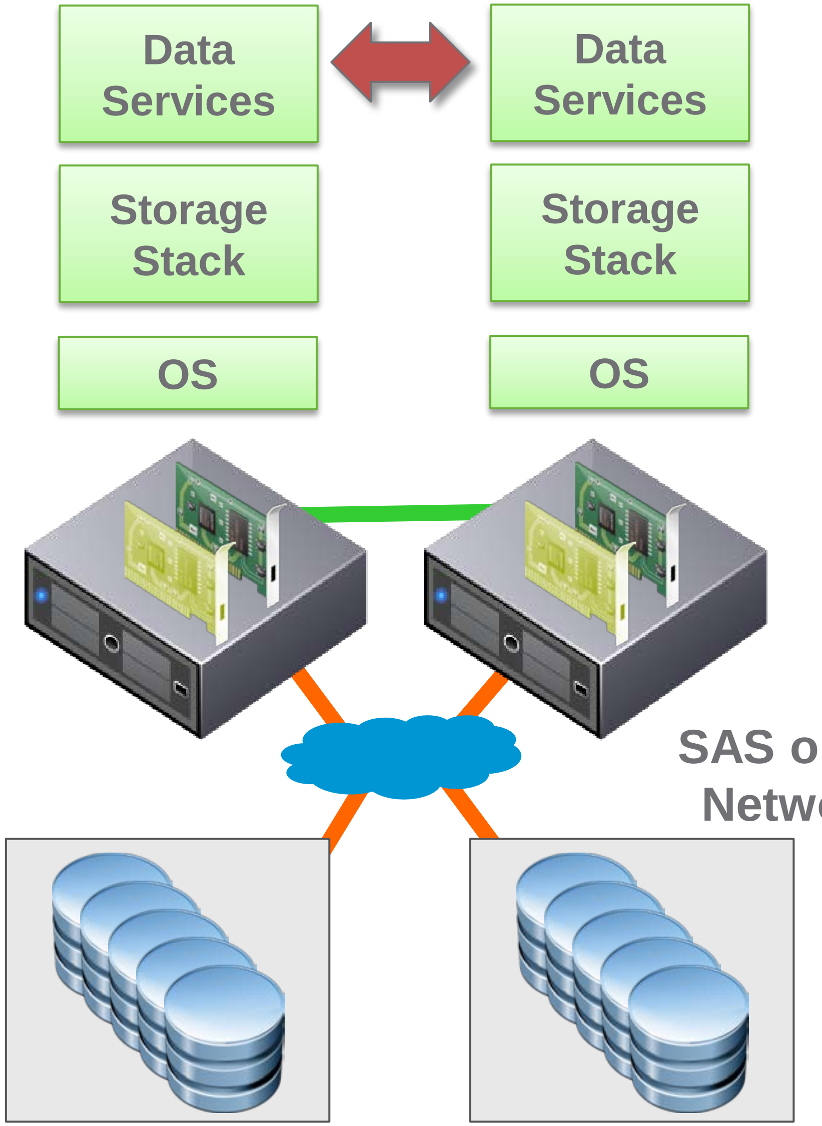


1. "Intel and Micron Produce Breakthrough Memory Technology", Intel

Storage Workload Map: SDS Solutions



What’s Really Behind a Storage Array?



Fail-over Services

iSCSI, FC, etc

De-dup
Volumes
Mirroring
Caching
Etc,…

Clustered OS

Dual Failover Processors

Industry std X86 Servers

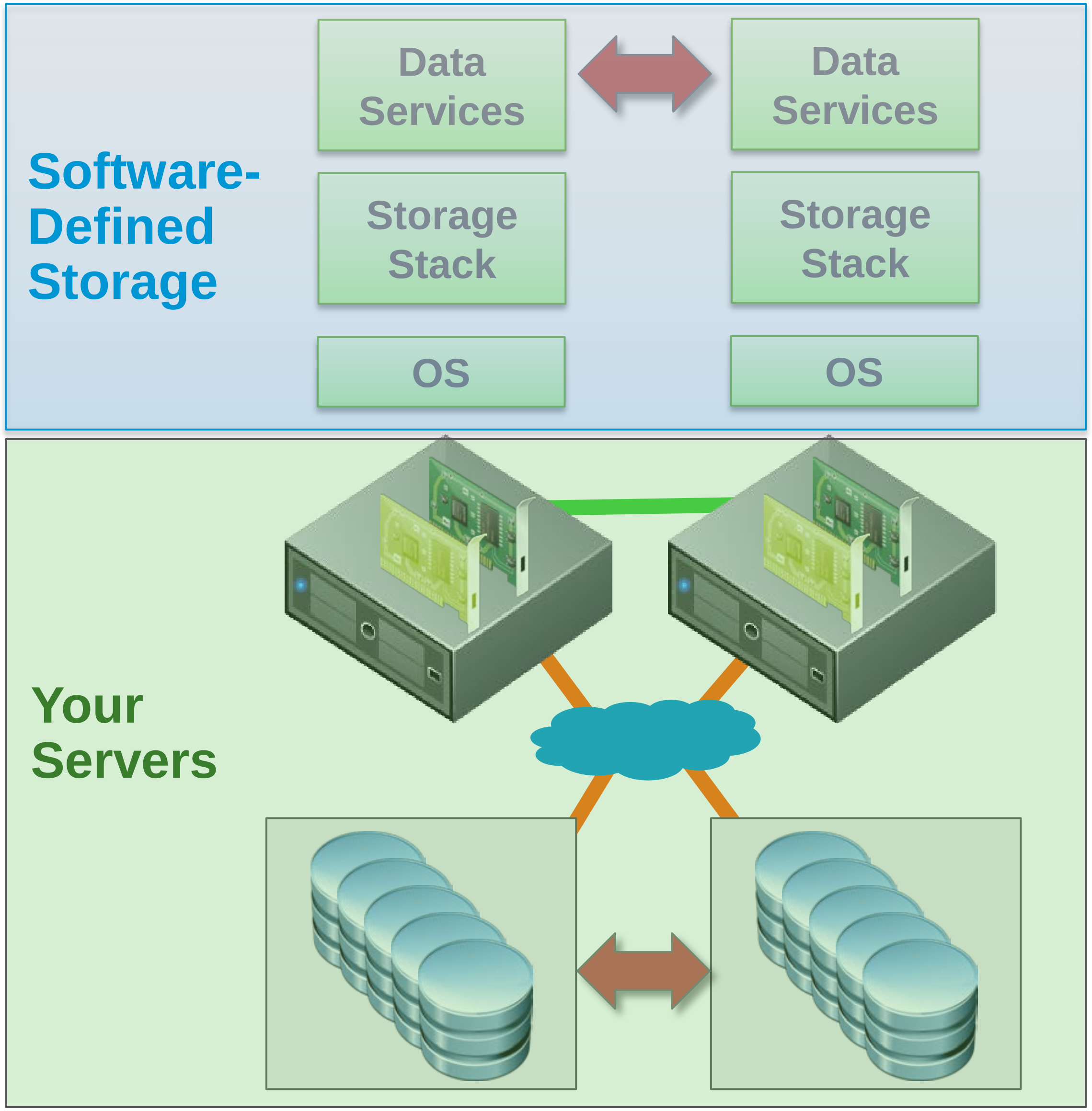
SAS or FC Network

Shared Disk SAS or FC

SDC 15

Different Types of SDS (1):

Fail-Over Software
on Commodity Servers

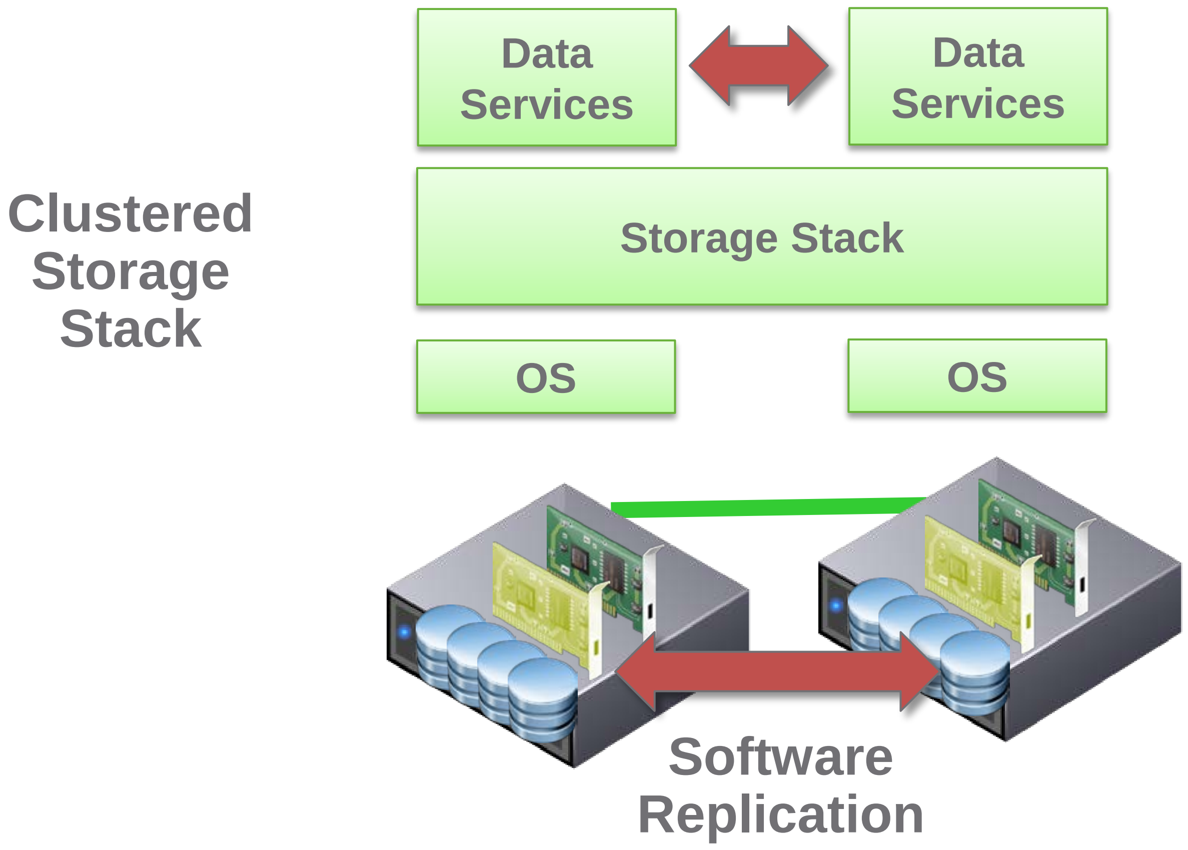


Nextenta
Data On-tap Virtual Appliance
EMC VNX VA
etc,…

HP
Dell
Intel
Super-Micro
etc,…

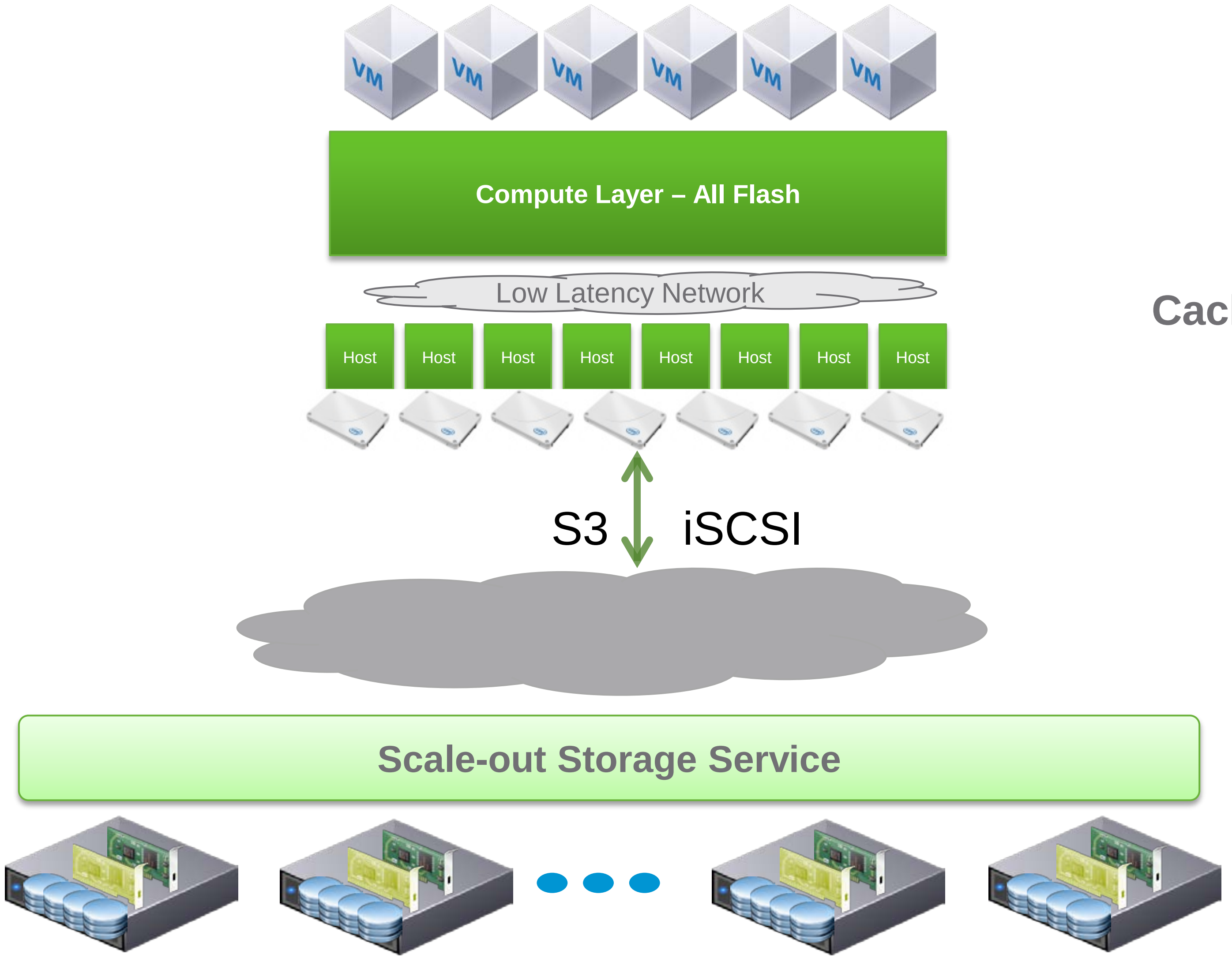
vmware

(1) Better: Software Replication Using Servers + Local Disks



- ✓ Simpler Server configuration
- ✓ Lower cost
- ✓ Less to go wrong
- Not very Scalable

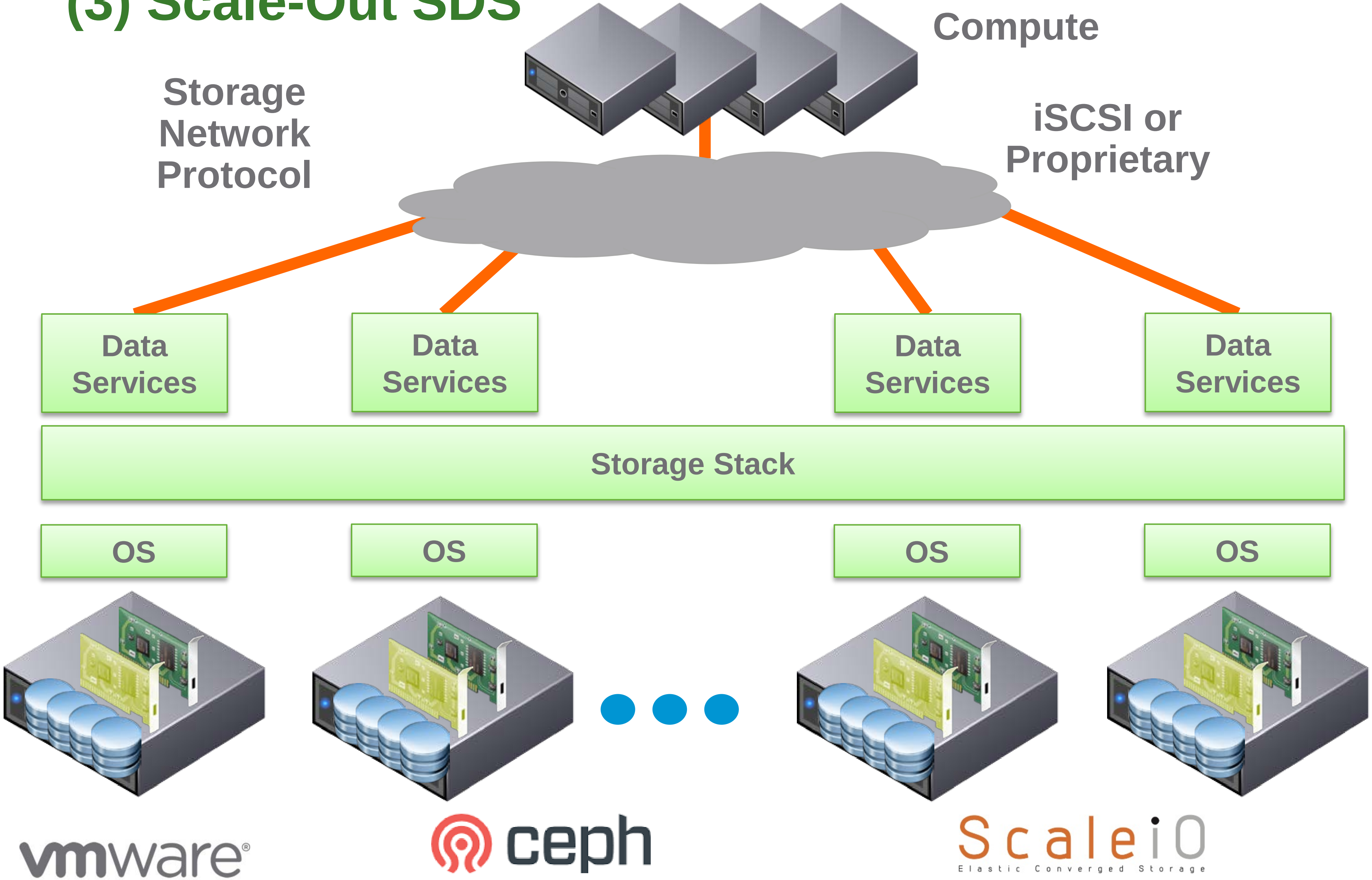
(2) Caching Hot Core/Cold Edge



Hot Tier
Caching of Reads/Writes

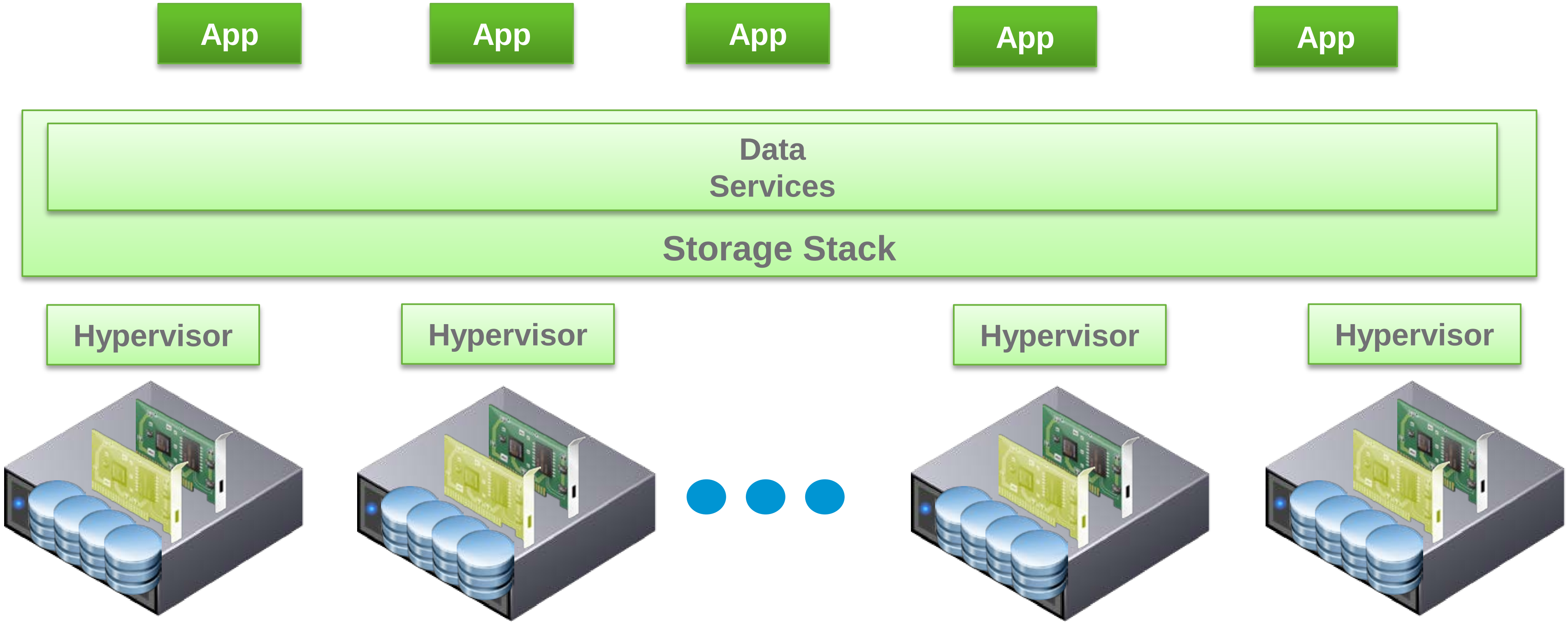
Scale-out
Storage Tier

(3) Scale-Out SDS



- ✓ Scalable
- ✓ More Fault Tolerance
- ✓ Rolling Upgrades
- More Management (Separate compute and storage silos)

(4) Hyperconverged SDS

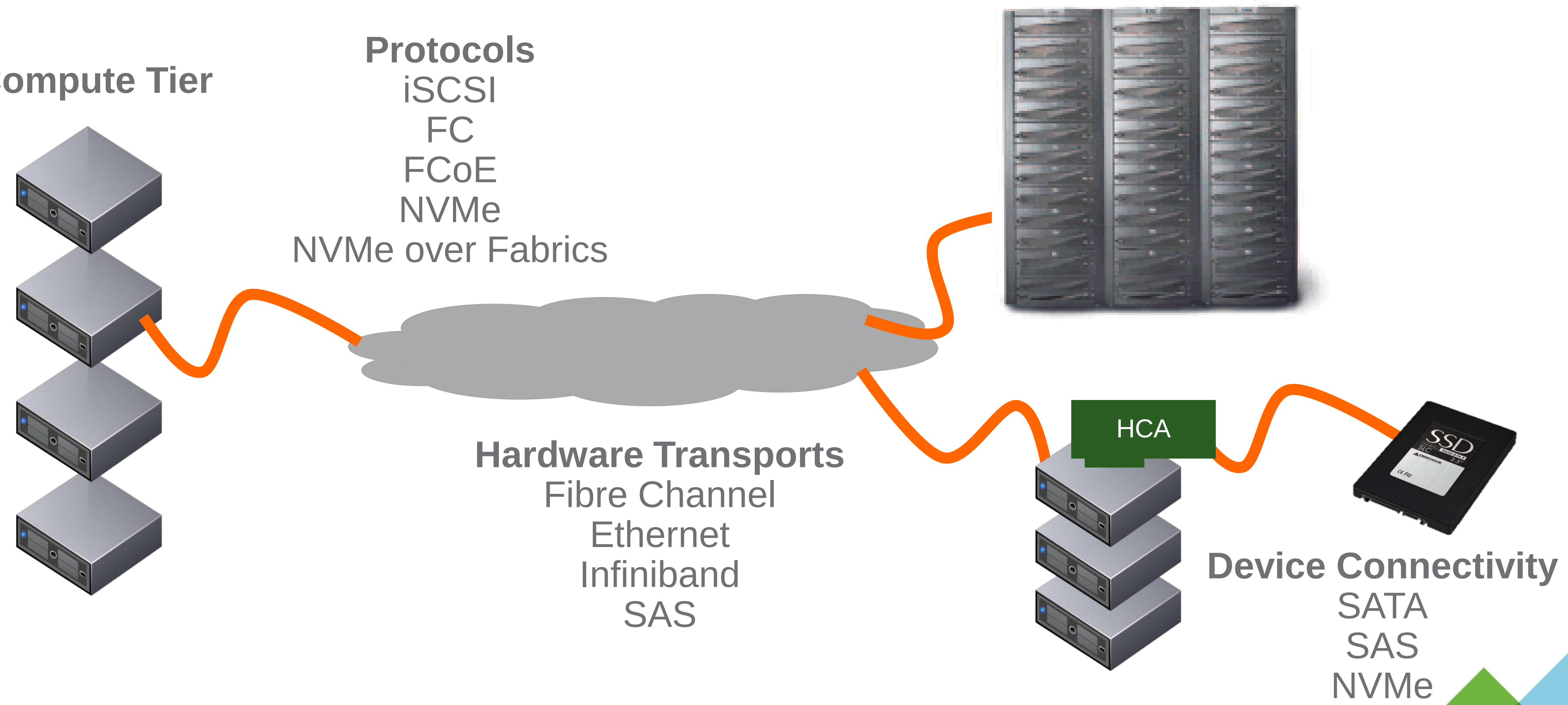


VSAN

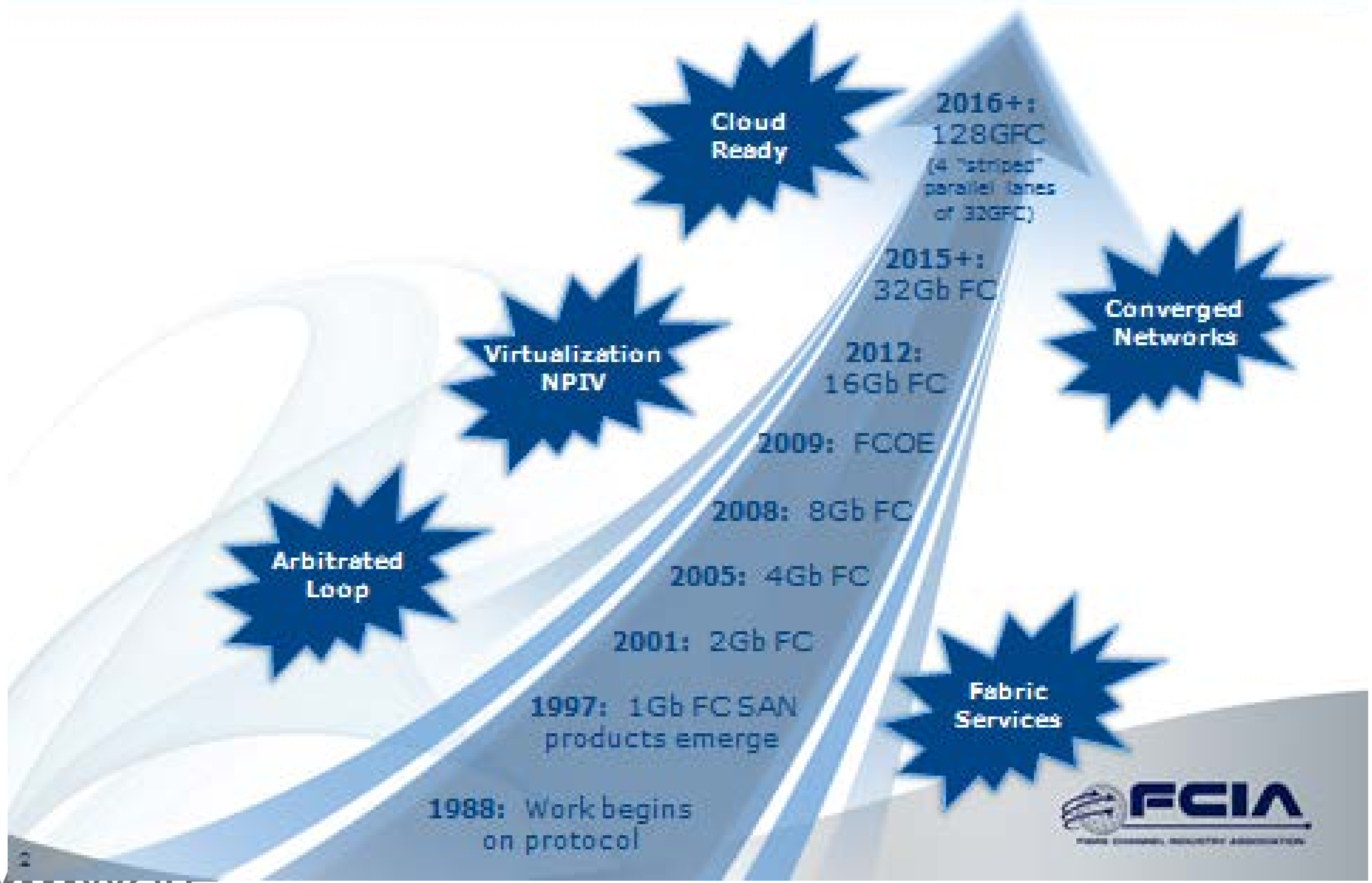
NUTANIX

- ✓ Easier Management: (Top-Down Provisioning)
- ✓ Scalable
- ✓ More Fault Tolerance
- ✓ Rolling Upgrades
- ✗ Fixed Compute and Storage Ratio

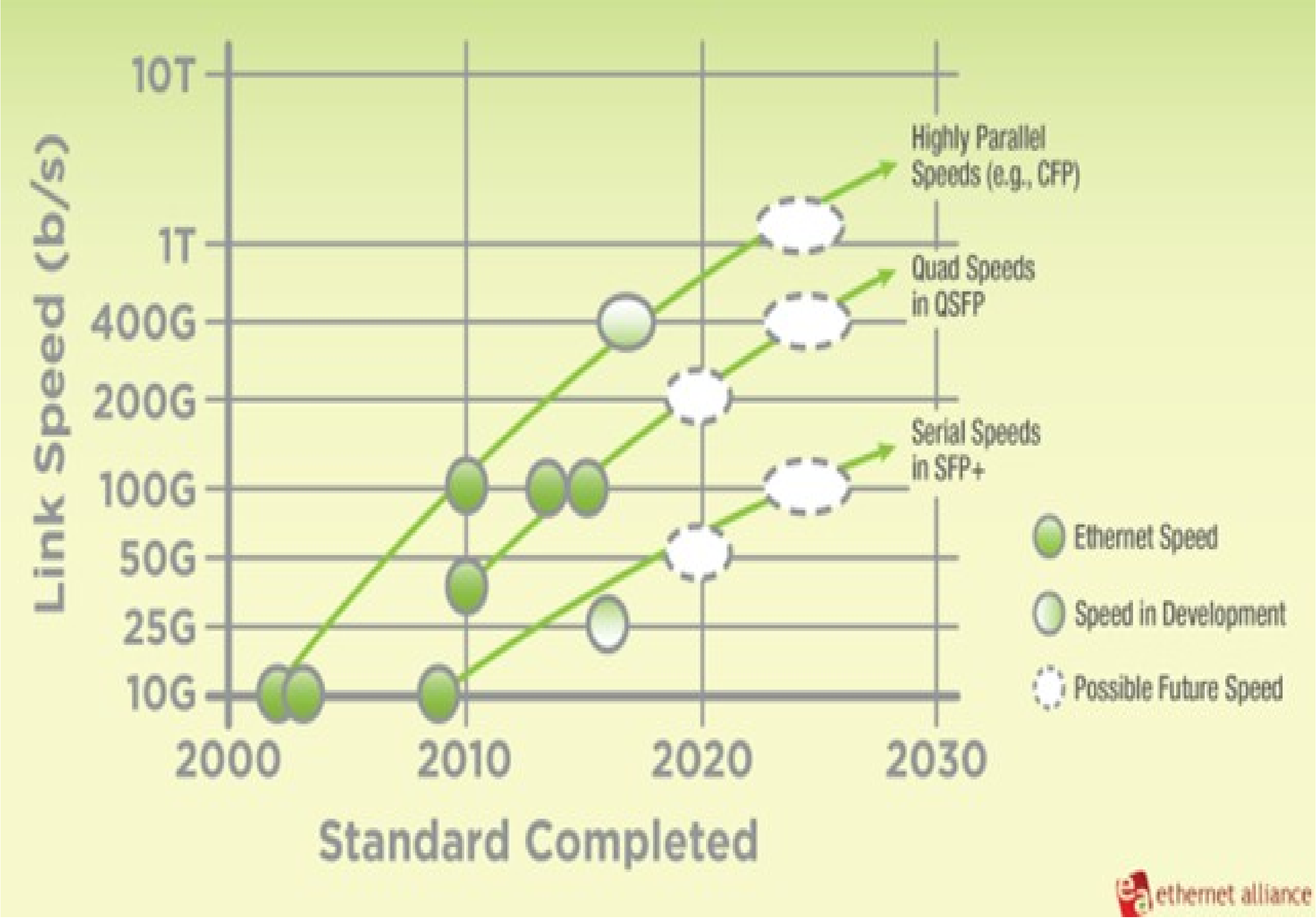
Storage Interconnects



FIBRE CHANNEL: TIMELINE



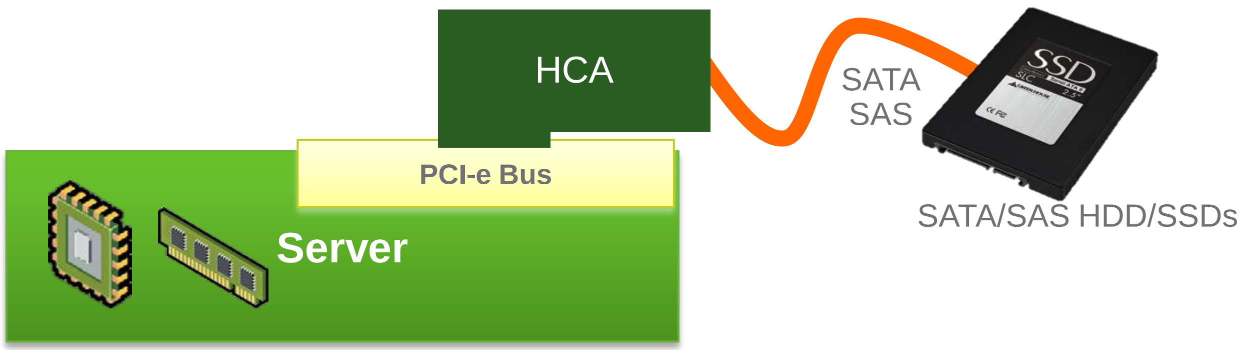
Network

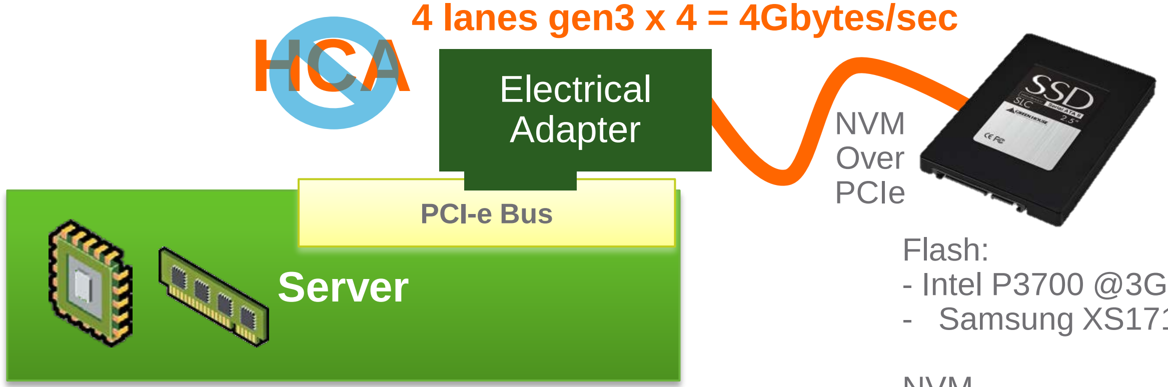


- Lossy
- Dominant Network Protocol
- Enables converged storage network:
  - iSCSI
  - iSER
  - FCoE
  - RDMA over Ethernet
  - NVMe Fabrics

Device Interconnects

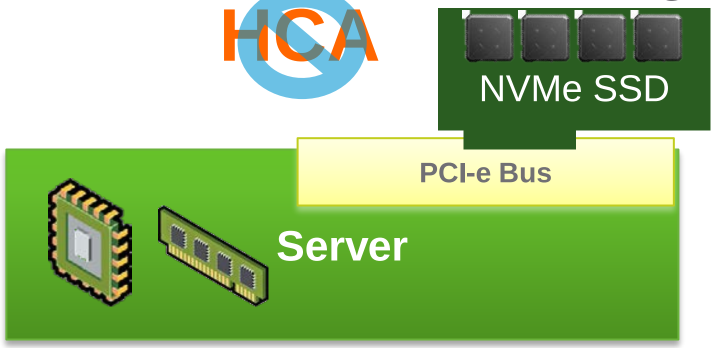
*There are now PCIe extensions for Storage





Flash:
- Intel P3700 @3GB/s
- Samsung XS1715

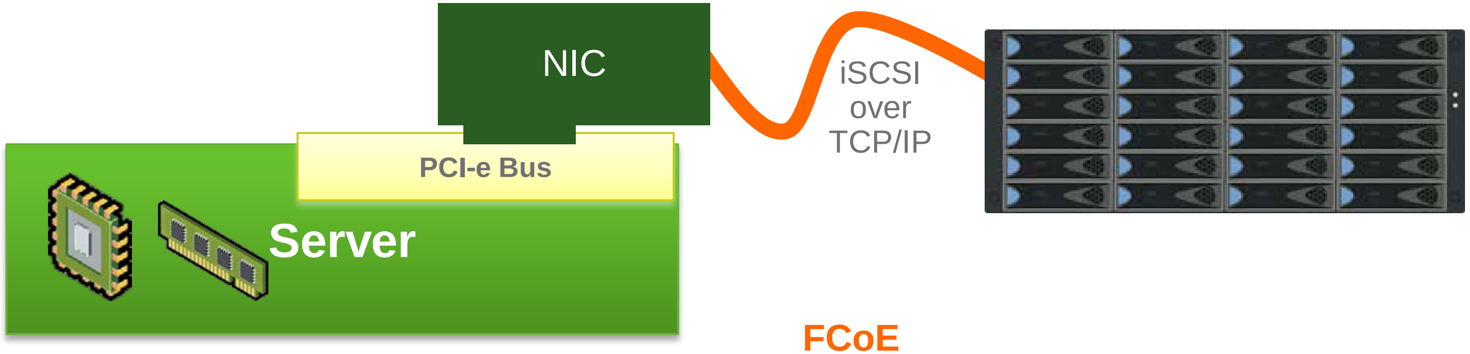
NVM
- Future devices



PCIe	Serial Speed	Bytes/sec
Gen 1	2gbits per lane	256MB/sec
Gen 2	4gbits per lane	512MB/sec
Gen 3	8gbits per lane	1GB/sec

Storage Fabrics

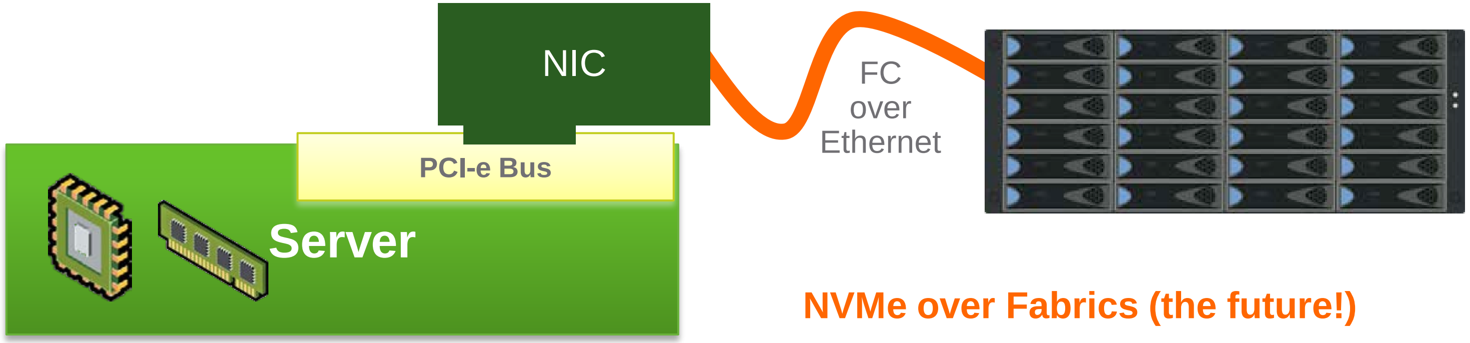
iSCSI



Pros:
- Popular: Windows, Linux, Hypervisor friendly
- Runs over many networks: routeable. TCP
- Optional RDMA acceleration (iSER)

Cons:
- Limited protocol (SCSI)
- Performance
- Setup, naming, endpoint management

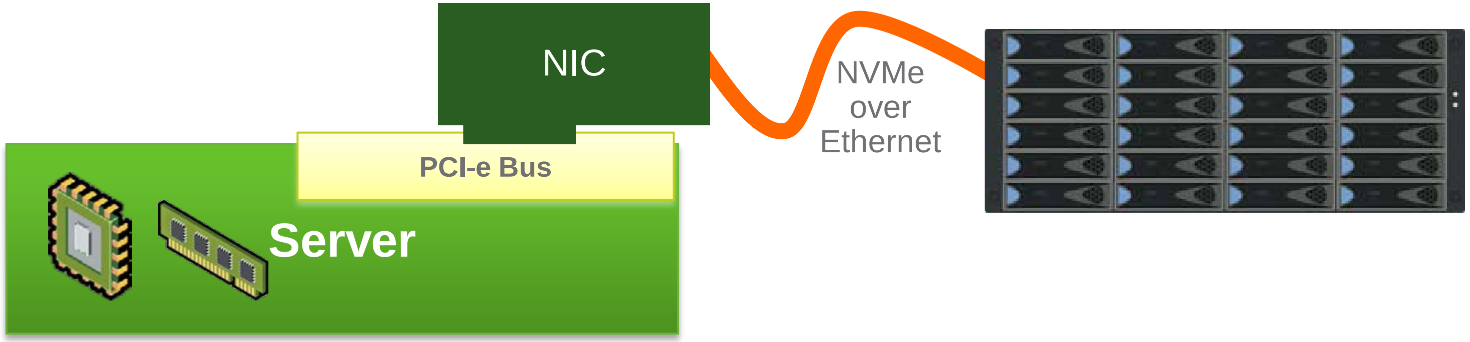
FCoE



Pros:
- Runs on ethernet
- Can easily gateway between FC and FCoE

Cons:
- Not routable
- Needs special switches (lossless ethernet)
- Not popular
- No RDMA option?

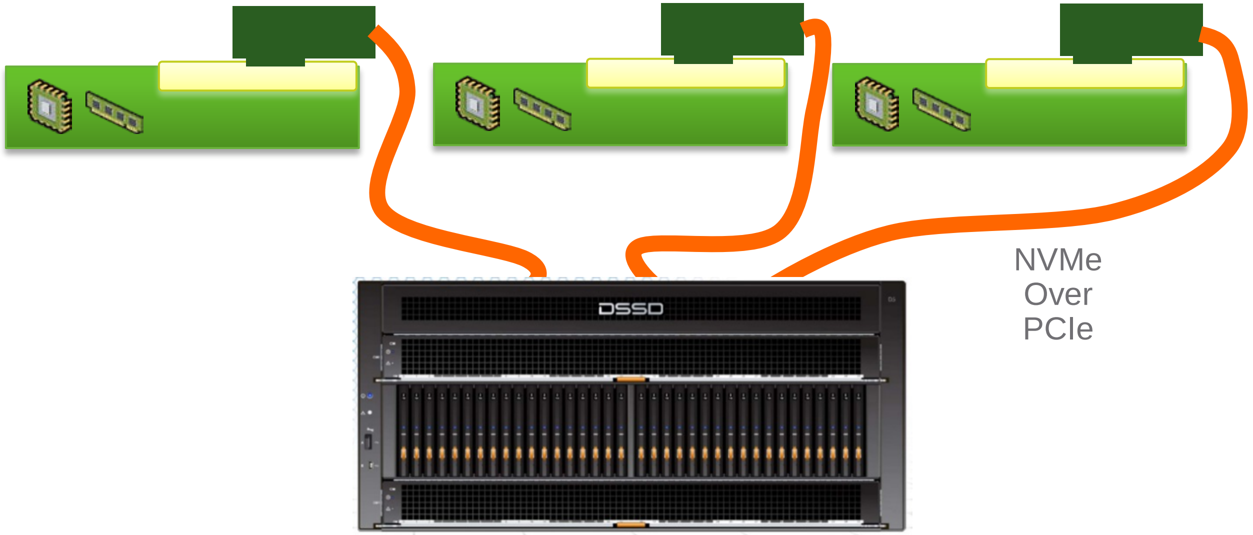
NVMe over Fabrics (the future!)



Pros:
- Runs on ethernet
- Same rich protocol as NVMe
- In-band administraton & data
- Leverages RDMA for low latency

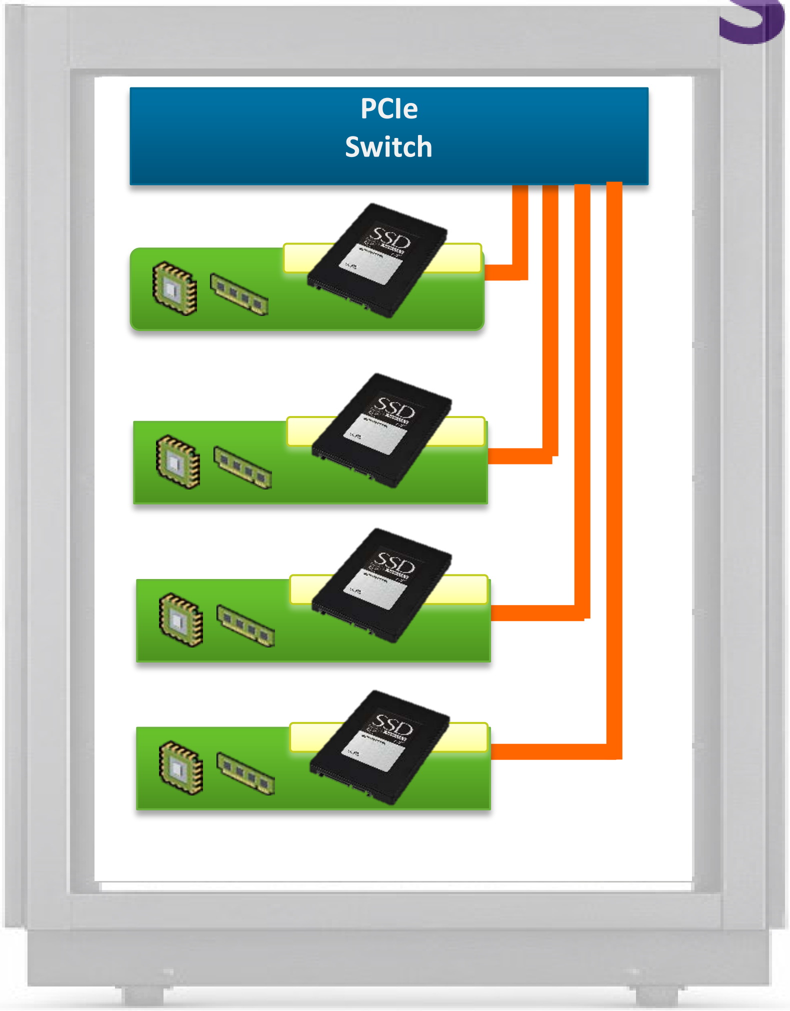
Cons:

Storage Fabrics (PCIe Rack Level)



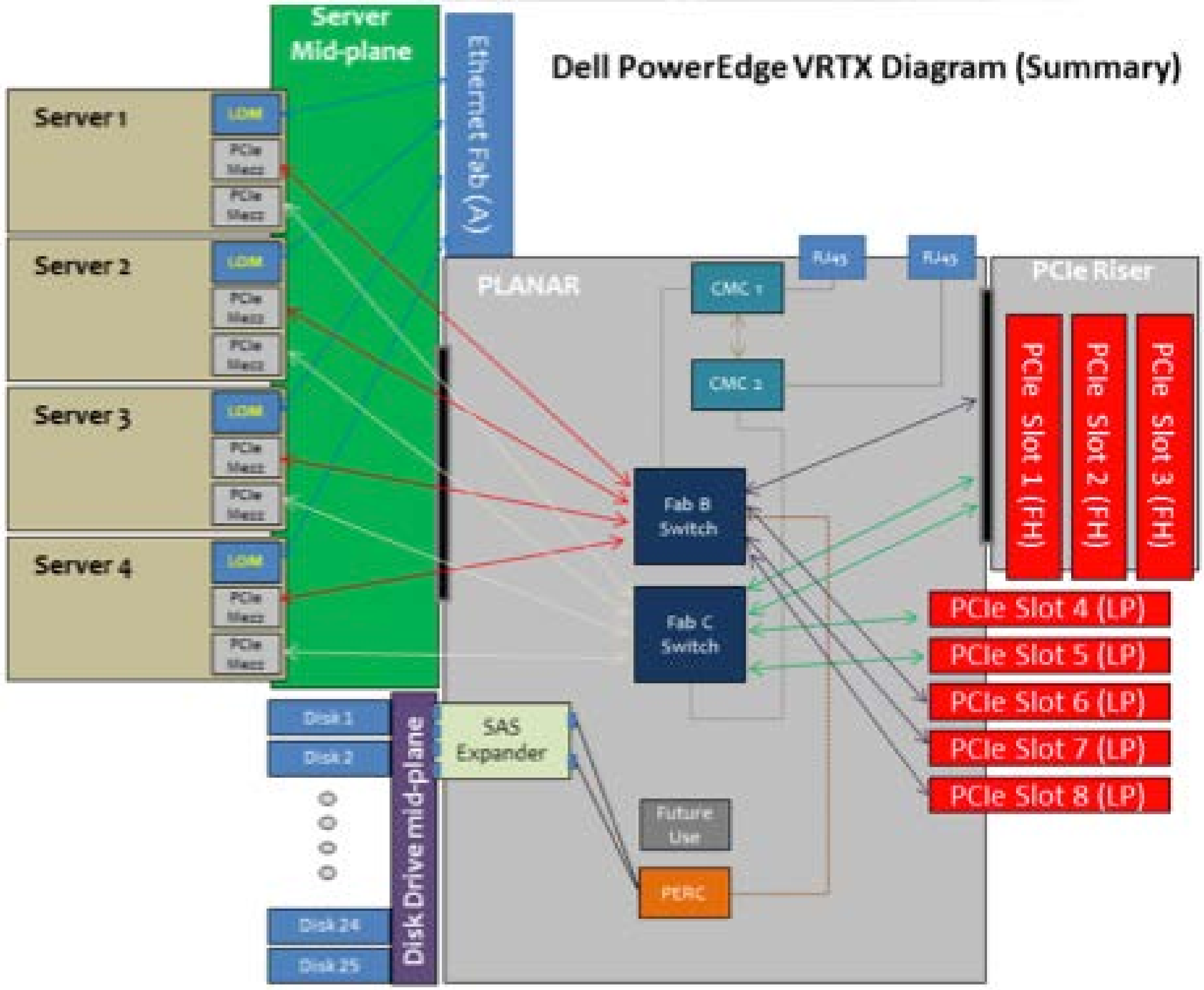
PCIe Shared Storage

RAID
Namespace
QoS
Autotiering

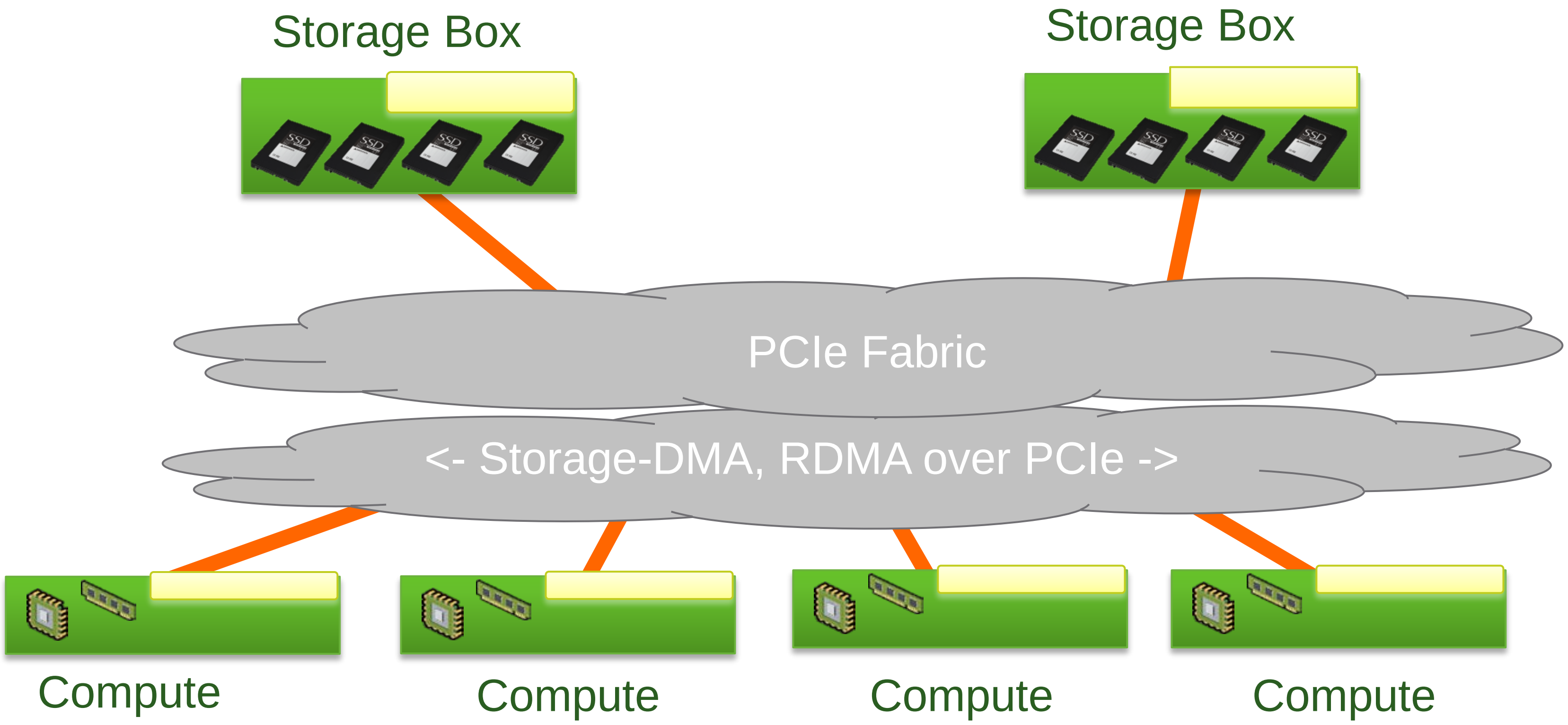


PCIe Rack Fabric
(Shared Drives between Hosts)

Integrated PCIe Fabric



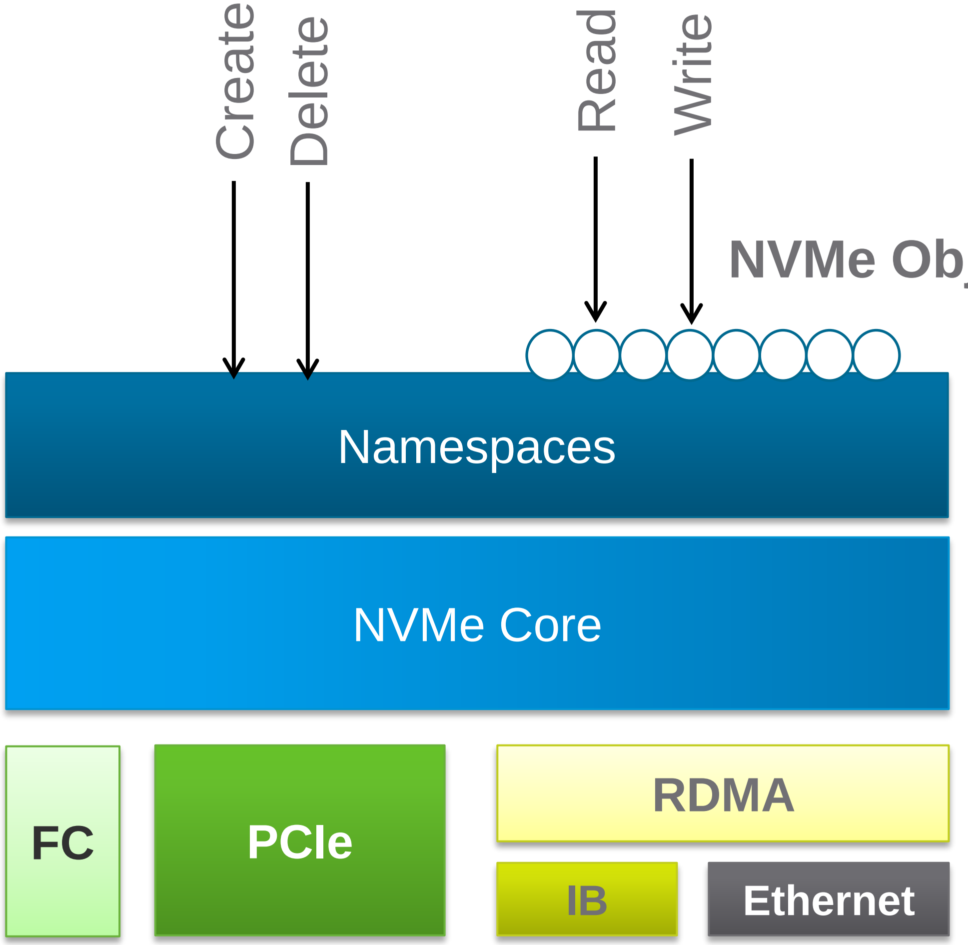
PCI-e Rack-Scale Compute+Storage



- PCIe Fabric:
  - Share disk boxes across hosts
- Storage-DMA:
  - Host-Host RDMA capability
  - Leverages the same PCI-e Fabric

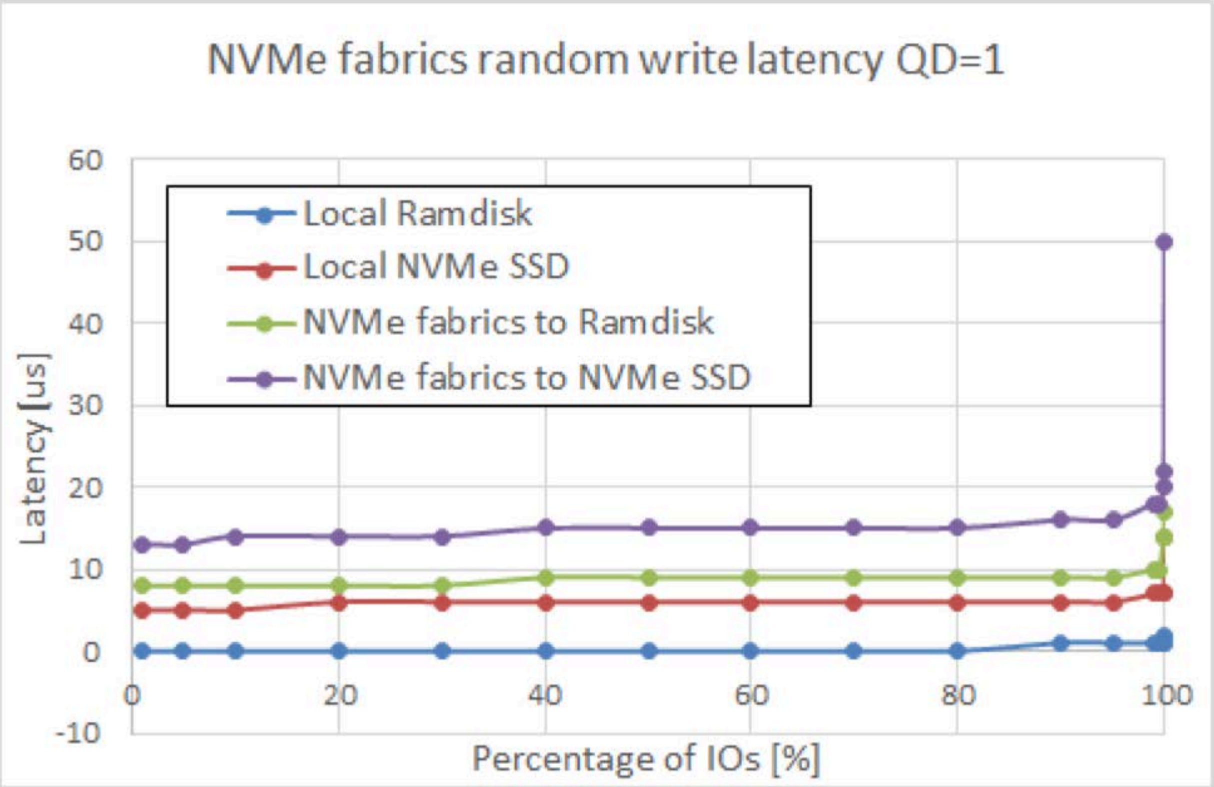
NVMe – The New Kid on the Block

SCSI-like: Block R/W Transport
NFS-like: Objects/Namespaces
vVol-like: Object + Metadata + Policy



NVMe Objects

NVMe over Fabrics



Now/Future

Interconnect	Now	Next	Future (Richard's Opinion)
SATA	6g	-	NVMe
SAS	12g	24g	NVMe
PCI-e Proprietary	4 x 8g		NVMe
SCSI	Over FC/Virtual		NVMe
FC	8g	32g	NVMe over Fabrics
Ethernet	10g	20/40g	100g

SDS Hardware Futures: Conclusions & Challenges

- Magnetic Storage is phasing out
  - Better IOPS per $$$ with Flash since this year
  - Better Space per $$$ after 2016
  - Devices approaching 1M IOPS and extremely low latencies
- Device Latencies push for new interconnects and networks
  - Storage (first cache) locality is important again
  - Durability demands low-latency commits across the wire (think RDMA)
- The network is critical
  - Flat 40G networks design point for SDS

Beyond Block:

Software-Defined Data Services Platforms

SDC 15

NoSQL K-V

Storage Protocols

AEROSPIKE

VOLTDB

memsql

SAP HANA

Cassandra

APACHE HBASE

S3/Swift

SCALITY

ceph

SCSI

MariaDB

Powered By ORACLE

PostgreSQL

Microsoft Exchange Server 2010

SQL Flavors

POSIX

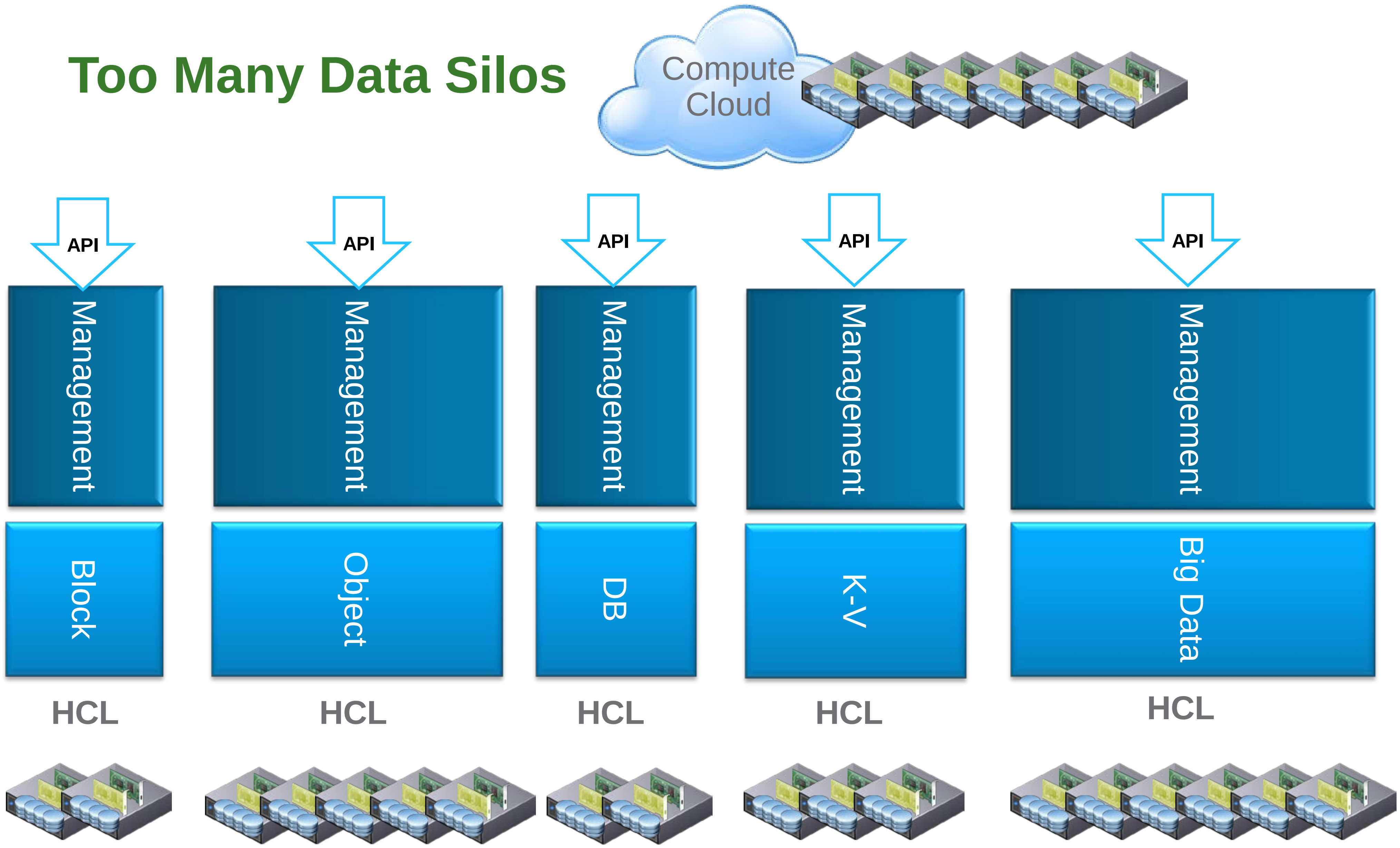
MAPR

HDFS

HDFS

vmware

Too Many Data Silos



Developers Are Happy

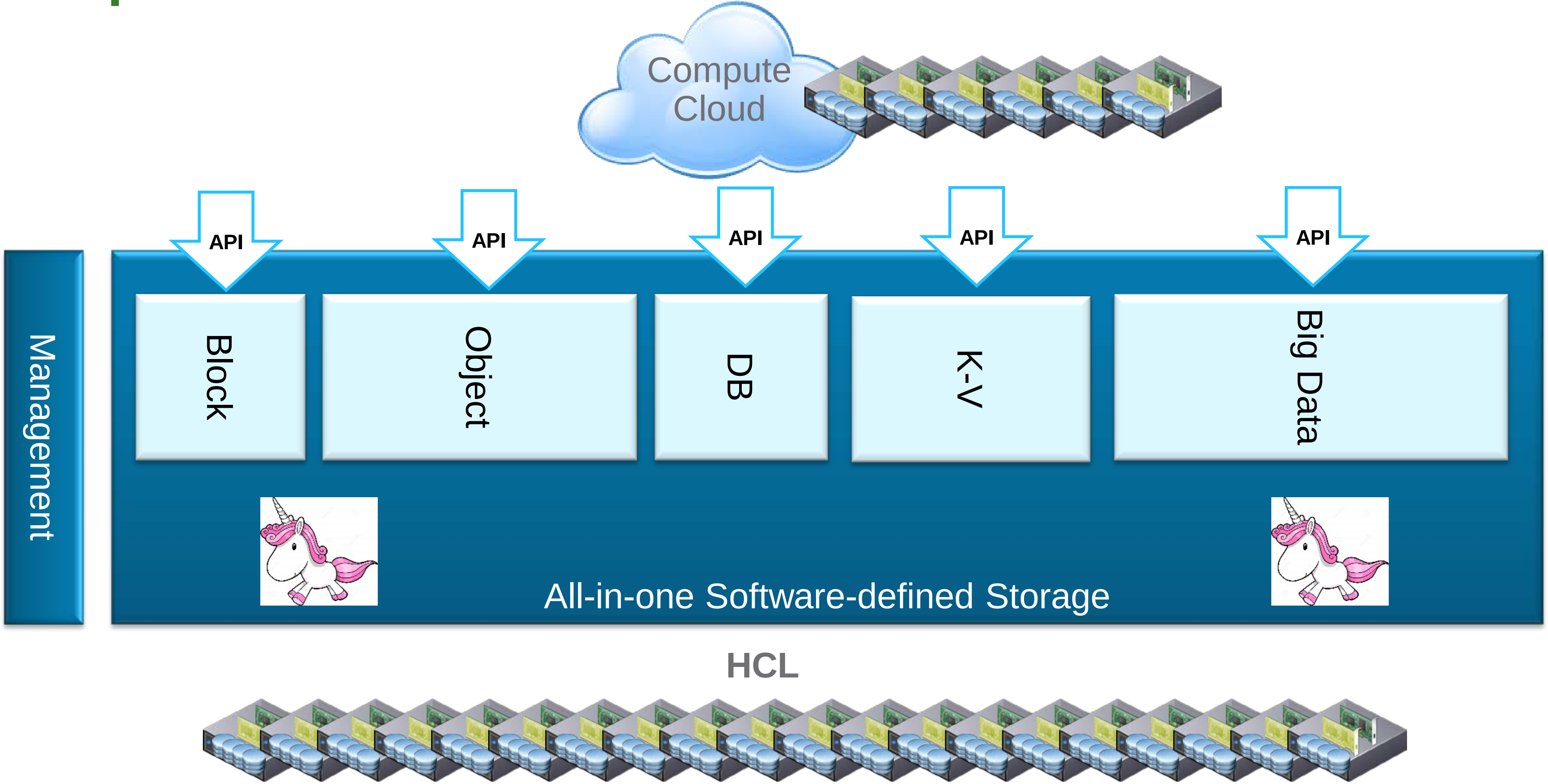
Lots of Management

Different SW Stacks

Different HCLs

Walled Resources

Option 1: Multi-Protocol Stack



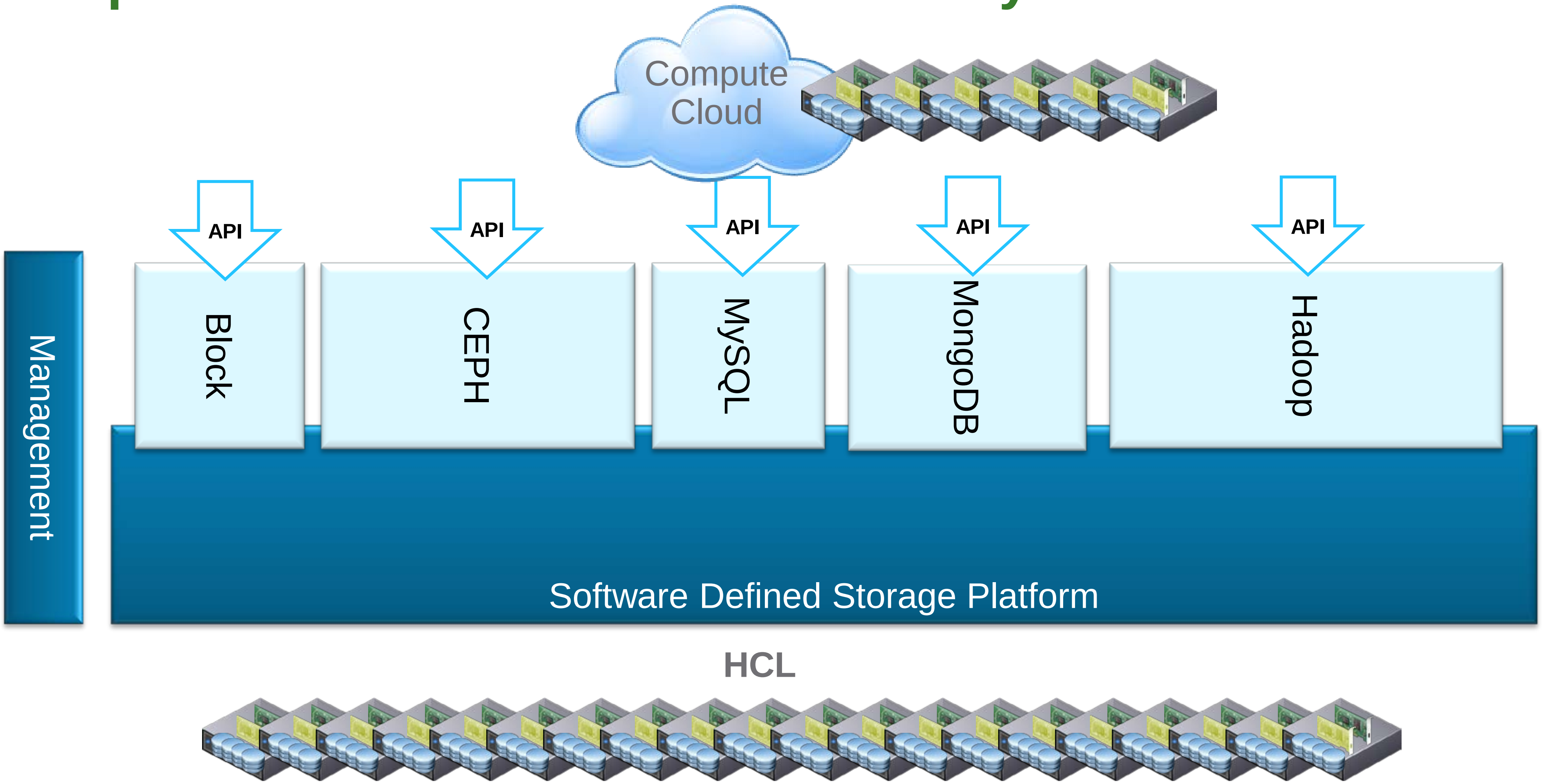
Developers Are Happy

One Stack to Manage

Least Capable Services

Shared Resources

Option 2: Common Platform + Ecosystem of Services



- Developers Are Happy
- One Platform To Manage
- Richest Services
- Shared Resources

Opportunities for SDS Platform

- Provide for needs of Cloud Native Data Services
  - E.g. MongoDB, Cassandra, Hadoop HDFS
- Advanced Primitives in support of Distributed Data Services
  - Policies for storage services: replication, raid, error handling
  - Fault-domain aware placement/control
  - Fine grained policies for performance: e.g. ultra-low latencies for database logs
  - Placement control for performance: locality, topology
  - Group snapshot, replication semantics

Thank You