Optimizing Hyperconverged Infrastructure for NVMe-Based Flash Storage

Richard Elling

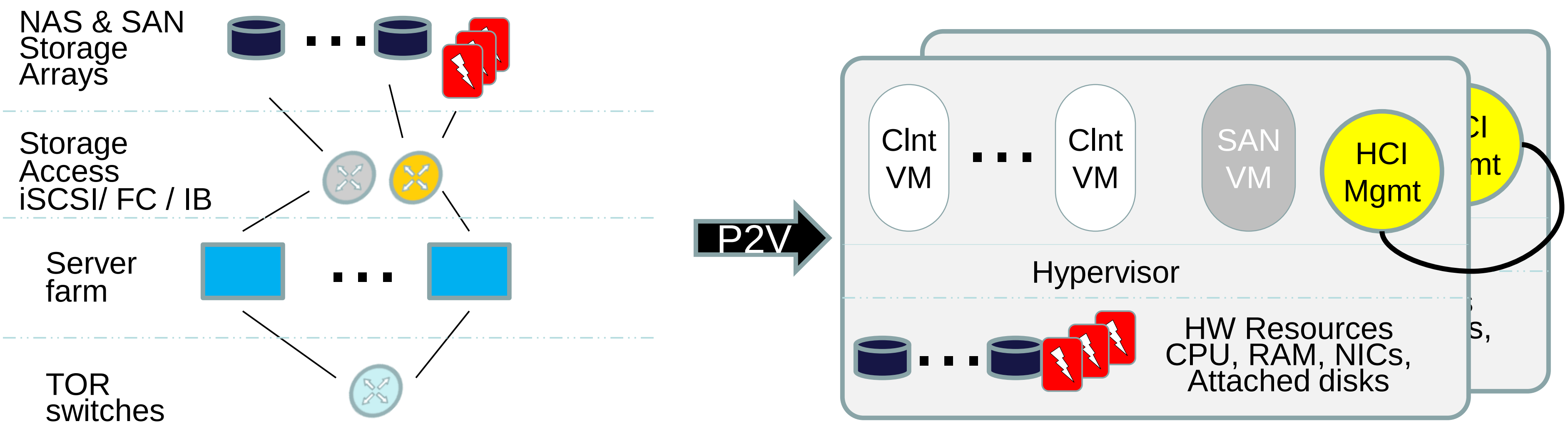
Kais Belgaied

Infrastructure Evolution to HCI

	Pre-history **Independent Resources**	**History** **Converged Infrastructure**	**Present** **Hyper-Converged Infrastructure**
Level of Integration (*)	Independent vendors "meet at the customers"	Single point of contact for procurement & support	Modular appliance with HCI SW abstracts away the physical resources
Advantages	Maximum choice - Best of breed - Optimal configs for cost or perf.	Pre-validated config combination Unified admin model Simpler	Very simple: little expertise required to operate. Better uptime
Challenges	Customer responsible for integration. No single support	Less choice Still requires skilled sys/storage/net admins	1-size fit all Sub-optimal resource utilization under large scale

(*) Integration refers to the 3 resources: Compute (servers), networking (switches & routers) and Storage (NAS, SAN, HBAs, etc.)

HCI 1st Gen: SAN-in-a-VM Model



Traditional IT:

- Each tier is a separate set of managed entities
  - → different lifecycles
  - →Different skills
- Costly & complex storage NAS & SAN for data protection and reduction

Hyper Converged Infrastructure:

- All hardware tiers owned by and managed through hypervisors using single HCI software
- NAS & SAN virtualized:
  - → Storage volumes served by SAN VMs
  - →Easier to offer client and application-aware data protection and reduction
- Scales horizontally by adding appliances

Container Scale & Shifting Bottlenecks

	Virtual Machines	Containers
Density	~20 per server	100s per server
Granularity	Coarse, server size. Fixed.	Fine, single application. Variable.
Diversity	A few OS images	Thousands of images
Scale	A few instances	"Herds" of containers
Lifecycle	Long lived Server model: provision/install/update/patch	Automatic Service Process model: Load/add instances/kill
Footprint	Uniform, through hypervisor	Wide range
Isolation	Enforced by hypervisor	Containers share the same bare-metal
Resource assignment	Permanent to Machines with general purpose OS	Ephemeral to Workload – collection of running containers
Storage Access	Penalized by crossing the hypervisor-VM spaces	Possible at very low latency, with the right architecture

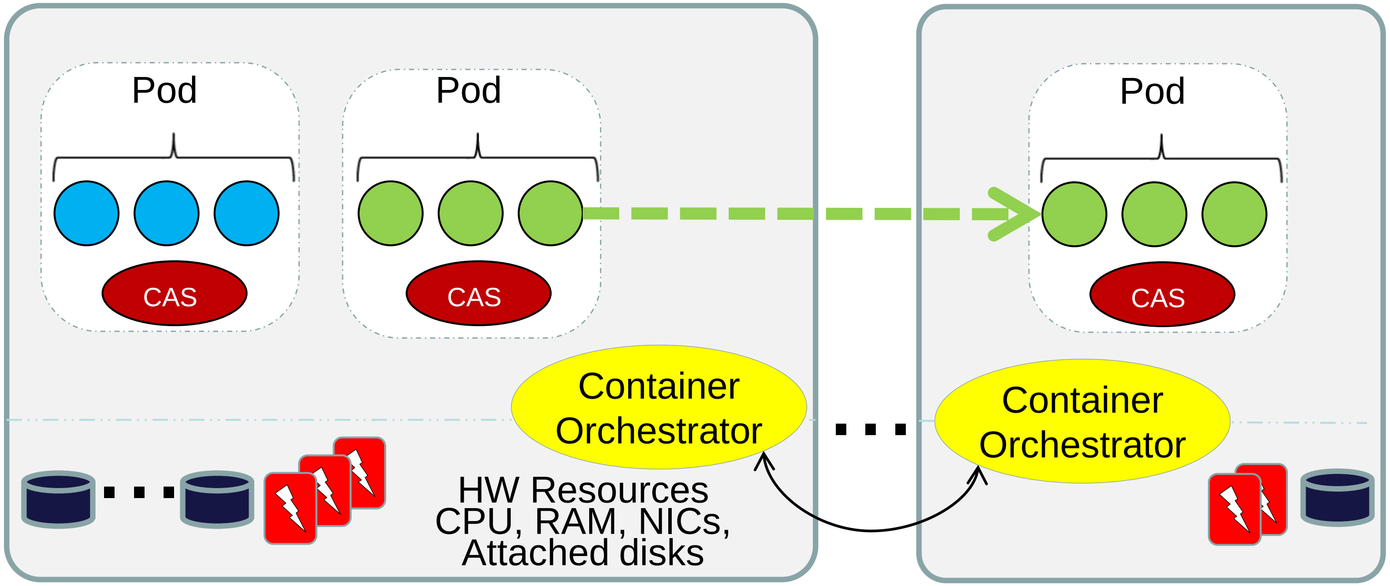
- The 1st generation HCI model is unable to meet the agility needs of Container based workloads

NVMe Storage: A Game Changer

	HDD	NVMe Disks
Density	14TB / 3.5” - plateau	32 TB / 2.5” - increasing
Latency	~8ms for 7200 rpm	8 ~ 100 µs.
Bandwidth	~200 MBps per disk	3.6 GBps per disk with PCI 3.0.
IOPs	A few 1000’s per disk. Lower with mixed and random	100s of 1000’s per disk. Multiple queues
Resilience	1.2 Million Hours MTBF (0.73% Annual Failure Rate)	Wide range. ~1.5 Million Hours MTBF

- Virtualization adds **~1.3ms** of latency to storage IOs
  - 1st Generation HCI was an acceptable small %age overhead over the HDD.
- Modern Containerized workloads using NVMe will suffer a **12 x** the latency overhead with virtualization compared to bare-metal

Cloud Native HCI with CAS Model



- Multiple instances of a containerized application run in a “Pod”
- Automatic provisioning, scheduling, resizing and migration using a container orchestrator (e.g. Docker Swarm, Kubernetes, Mesos, etc.)
- Stateful application pods include a software storage controller running as a special container CAS (Container Attached Storage)

CAS Features and Optimizations

- The CAS + Orchestrator ensure the application data “follows” the Pod as it moves to different hosts
- The CAS implements the data protection (replication, snapshots, clones, etc) and reduction (compression & deduplication) services
- Advanced CAS implementations:
  - Affinity and locality for bare-metal levels of performance, critical to latency sensitive applications requiring NVMe
  - Offer multiple classes of service
  - Ensure performance isolation, paramount for predictable behavior under diverse workloads

Cloud Native HCI: Control Plane

- Higher Scale & Velocity of operation in a Cloud Native environment
  - → 1000's of containers spawned, moved, killed per day
  - → Automation is a must: nearly impossible to manually assign resources and schedule individual containers
  - → Control plane performance is a key differentiator among orchestrators, and CAS implementations.
  - → Resource decommissioning is equally important
- Resilience
  - → Relies on fine grain health and performance monitoring for fault detection
  - → Need sufficient redundancy for a fail-in-place model

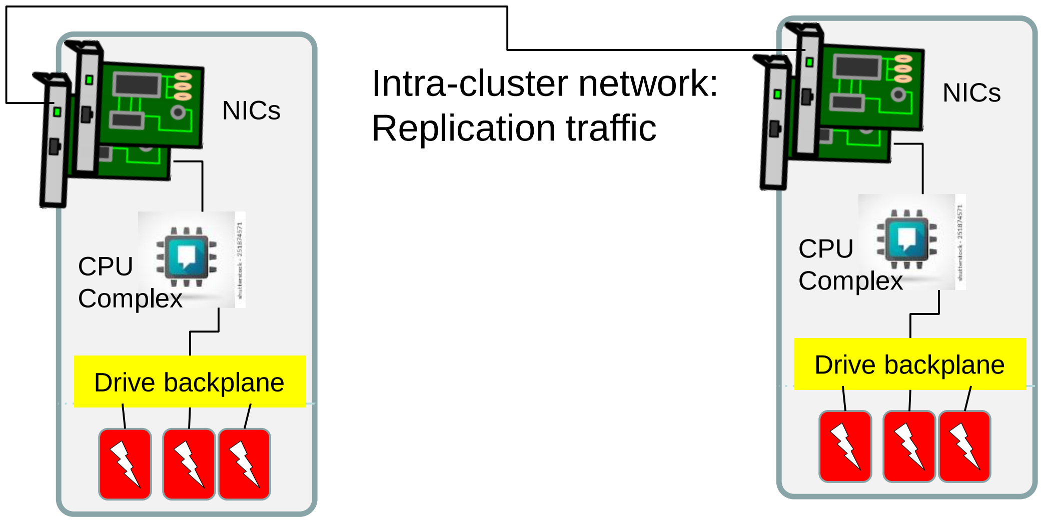
Hardware: Storage Considerations

- HCI pooling of disks based on price and performance profile of the underlying media
  - $/GB, and $/GBps
  - Latency : 3 levels commonly available 8us, 20us, 70us
  - Bandwidth – For NVMe, depends on
    - Media, cells, manufacturer, FTL
    - Queues (up to 64K. Typically only 1 offered per namespace
    - Number of PCI lanes
- Classes of Service associated with storage pools
- Workloads are awarded different classes of service

  → Independent scaling

  → Performance isolation

Hardware: The PCI Lanes Math



Desired features of a Cloud Native HCI Ready Platform

- Dense enough attached storage
  - ➔ Reads are served locally. The higher the density, the faster
- Balanced PCI lanes count:
  - ➔ At maximum write performance: Intra-cluster bandwidth = Bandwidth of Data generated within the Pods * Replication factor

    e.g. 4 disks @ 4 PCI lanes each, at 3 replicas, will require 32 PCI lanes on the NICs: 1 local copy + 2 remotes

Thank You

Demo of Cloud Native HCI platforms shown at Viking Enterprise Solutions booth

Questions?

Appendix: Abstract

The optimal choice for hyperconverged infrastructure has changed as NVMe and NVMe-oF become the most common flash storage connections. The traditional monolithic SAN-in-a-VM model is facing serious challenges as the bandwidth required for attached SSDs has increased greatly over what was typical for disk interfaces. The new generation of HCI software stacks, based on approaches such as replicated container-attached storage, has proven to be better suited for accommodating the unprecedented scale, density, and diversity of workloads. This however puts an increased strain on processor, interconnect, and network resources, which previously could readily handle large numbers of drives. Now infrastructure designers must consider new choices such as increasing the capabilities of central processors, using higher-bandwidth connections, or offloading functions to traditional hardware such as FPGAs or coprocessors. All these options add to the system cost and complexity. The correct choices depend on application areas, such as analytics, AI/ML, IoT, database searches, transactions processing or video and image processing.