Concepts on Moving From SAS connected JBOD to an Ethernet Connected JBOD (EBOD)

Jim Pinkerton
Microsoft Corporation
9/22/2015
Re-Thinking Software Defined Storage
Conceptual Model Definition

- Three “entities”
  - Compute Node
  - Storage Node
  - Flaky Storage Devices
- Front end fabric: Ethernet, IB, FC
- Back end fabric: Direct Attached or Shared Storage

Compute/ Application Node (mem, CPU) → Reliable, Available Storage → Storage Processing Node (mem, CPU) → Flaky Non-volatile Device

Fabrics: Ethernet, InfiniBand, Fibre Channel

Direct Attach: SATA, NVME, CPU Memory Interface

Shared Storage: SAS, Fibre Channel, Ethernet

File or Block Access

Disk Flash Tape Other
Yesterday’s Storage Architecture: Still highly profitable

- **Compute Farm**
- **Compute/Application Node** (mem, CPU)

**Storage Processing Node** (mem, CPU)
- Reliable, Available Storage
- **Fabrics**:
  - Ethernet
  - InfiniBand
  - Fibre Channel

**Non-volatile storage device**
- **Direct Attach**:
  - SATA
  - NVME
  - CPU Memory Interface

**Shared Storage**
- SAS
- Fibre Channel
- Ethernet

- **Moved to general purpose CPU & OS**
- **“Appliance” Vendor**
  - Tests it as a unit

- **Disk Flash Tape Other**
Today: Software Defined Storage (SDS) “Converged” Deployments

- The rise of componentization of storage – but an interop nightmare for the user

Software ships independent of hardware

Customer is the first to test as a system

Compute Farm

- Compute/ Application Node (mem, CPU)

Storage Service

- Reliable, Available Storage
- Storage Processing Node (mem, CPU)

JBODs

- Non-volatile storage device

Fabrics:
- Ethernet
- InfiniBand
- Fibre Channel

Direct Attach:
- SATA
- NVME
- CPU Memory Interface

Shared Storage:
- SAS
- Fibre Channel
- Ethernet

Flakey Non-volatile Device

Customer is the first to test as a system
Today: Software Defined Storage (SDS) “Hyper-Converged” (H-C) Deployments

- H-C appliances are a dream for the customer
- H-C $/GB storage is expensive

Typically sold as an appliance (OEM owns E2E testing)
An Example: Microsoft’s Cloud Platform System (CPS)

“Azure aligned innovation”
“Appliance like experience”
“Single Throat to Choke”

Package Deal

Shipped as of Oct/2014
**SDS with DAS**

With DAS, rebuild & replication traffic moves to Ethernet.
Software Defined Storage Topologies

- Physical host boundary, blue is workload on physical node, arrows can go between physical nodes

Hyper-Converged

Converged

Future? EBOD

EBOD Future? = physical host boundary, blue is workload on physical node, arrows can go between physical nodes

File or Block Access

Device Access

Data Mover Workload

SDC 2015 Storage Developer Conference. © Microsoft Corporation. All Rights Reserved.
EBOD Works for a Variety of Access Protocols & Topologies

- SMB3 “block”
- Lustre – object store
- Ceph – object store
- NVME Fabric?…
- T10 Objects
I have a problem... shared SAS interop

My Nightmare Experience
- Disk multi-path interop
- Expander multi-path interop
- HBA distributed failures

SAS Shared Fabric
- Multi-Initiator
- Multi-Path

There is a good reason why customers prefer appliances!

Example Storage Cluster (Microsoft’s CPS)
To Share or Not to Share?

- **Shared SAS:**
  - Customer deployment can have serious bugs
  - Failure of a FE node: JBOD fails over to another node
  - Failure of a JBOD: all data is replicated

- **Non-Shared SAS (or SATA or NVME or SCM):**
  - Customer deployment more straightforward
  - Failure of a FE node: EBOD fails over to another node
  - Failure of a EBOD: all data is replicated
  - New Ethernet traffic
    - Triple replica (3x increase bandwidth on Ethernet)
    - Rebuild traffic
Hyper-Scale Cloud Tension – Fault Domain Rebuild Time

- Rebuild time is a function of
  - Number of disks/size of disk behind a node
  - Speed of network and how much of it you want to use

- Storage cost reduction is driving higher drive counts behind a node (>30 drives)
  - Causes higher network costs because rebuild time must occur in constant time

<table>
<thead>
<tr>
<th>TB behind one node</th>
<th>% BW utilization</th>
<th>Net speed (gb/s)</th>
<th># nodes in rebuild</th>
<th>Min for rebuild</th>
</tr>
</thead>
<tbody>
<tr>
<td>180</td>
<td>25</td>
<td>40</td>
<td>120</td>
<td>20</td>
</tr>
<tr>
<td>1080</td>
<td>25</td>
<td>40</td>
<td>120</td>
<td>120</td>
</tr>
<tr>
<td>1080</td>
<td>50</td>
<td>100</td>
<td>120</td>
<td>24</td>
</tr>
</tbody>
</table>

- Conclusions:
  - Required network speed offsets benefits of greater density
  - Fault domain for storage is too big

Large # drives require extreme network bandwidth
Private Cloud Tension – Not enough Disks

- Goal is entry point at 4 nodes (or less)
- If used same 30 disk JBOD
  - Loss of one node implies loss of 30 disks (180 TB)
  - To recover from node loss, must have 25% of capacity idle for single node failure, 50% idle for dual node fault tolerance
- Conclusion:
  - Fault domain is too large
Goals in Refactoring SDS

- Optimize workloads for class of CPU
  - Backend is “data mover” (EBOD)
    - Primarily movement of data and background tasks
      - Data Integrity, caching, …
    - Little processing power
  - Frontend is “general purpose CPU”
    - Still need accelerators for data movement, data integrity, encryption, etc.
EBOD Goals

- Reduce Storage Costs
  - Right size config for front end and back-end workloads
- Reduce size of fault domain (and rebuild traffic and network bandwidth requirements)
  - Small Private Clouds
  - Move storage fabric to Ethernet
- Build on more robust ecosystem of DAS
  - Keep topology for storage device simple
EBOD Design Points

- Ethernet Connected JBOD
  - High End Box (NVME, Storage Class Memory)
  - Volume Box (Some NVME/SSD, HDD)
  - Capacity Box (primarily HDD, some NVME/SSD)
- What If we used “small core” CPU?
- Fewer disks because cheaper CPU
Enable an Ethernet connected JBOD with low disk count at *very* low cost

- Critical to hit a price point similar to existing SAS JBODs that are integrated into the chassis
- Export just raw disks to keep CPU as simple as possible, and SDS as close to hardware as possible
  - Needed for Storage Class Memory
- Enable front end nodes (big core) to create reliable/available storage
Comparing Storage Node Design Points

Current Hyper-Scale

- NIC
- Large Core CPU
- HBA

HBA Replaced With:
- NIC on Storage Node
- Switch port to Storage Node
- Switch port to EBOD

EBOD (JBOD)

- NIC
- Large Core CPU
- Ethernet Switch

Switch in Chassis?

EBOD

- NIC
- SOC
- Expander

Expander Replaced With:
- SOC
- NIC
- Smaller expanders

JBOD

- Multiple Expanders
- 60 disks

- NIC
- SOC
- Expander

- 20 disks x 3

Ethernet Cable x3

2015 Storage Developer Conference. © Microsoft Corporation. All Rights Reserved.
EBOD Volume Concept

- CPU and Memory cost optimized for EBOD
- Dual attach >=10 GbE
- SOC with integrated
  - RDMA NIC
  - SATA/SAS/PCIE connectivity to ~20 devices
- Universal connector (SFF-8639)
- Management
  - Out-of-band management through BMC
  - In-band management with SCSI Enclosure Services
Volume EBOD Proof Point

- Intel Avaton Microserver
- PCIE Gen 2
- Chelsio 10 GbE NIC
- SAS HBA
- SAS SSD
Max remote performance
- ~159K IOPS w/ RDMA
- ~122K IOPS w/o RDMA

Higher Performance gain with RDMA at lower IOPs
At Higher queue depths, RDMA gains reduce to ~30%
- CPU% capped at 122K (28 outstanding IOPs/7 SSDs)
- CPU is bottleneck

Remote Access is bottlenecked on Network Speed

Configuration
- Intel Avaton Microserver
  - PCIE Gen 2
- Chelsio 10 GbE NIC
- SAS HBA
- SAS SSD
EBOD Avaton Microserver POC - 8K Random Reads, Latency

- With RDMA
  - At least 65% speedup in IO latency (~75% at higher IOPs)
- Remote Latency drops at 4 outstanding IO/disk (7 disks)
- Goes back up at 8 outstanding IO/disk
EBOD Performance Concept

- Goal is highest performant storage, thus big-CPU is small part of total cost
  - Hi-speed CPU reduces latency
- Dual attach >=40 GbE
- SOC with integrated
  - RDMA NIC
  - PCIE connectivity to ~20 devices
- Possibly all NVME attach or Storage Class Memory

See “SMB3.1.1 Update” SDC Talk for Dual 100 GbE early results
Summary

EBOD enables

- Shared storage on Ethernet using DAS storage
  - DAS storage has better interop within eco-system
- Price point of EBOD must be carefully managed
  - Microserver CPU is viable for broad spectrum of perf
  - Integrated SOC solution is preferred
- Low price point of EBOD CPU/Memory enables smaller fault domain (fewer disks can be behind the Microserver)
Q&A
Outstanding Technical Issues

- Enclosure level
  - How to manage storage enclosure? SCSI Enclosure Services (SES)?
  - Base Management? DMTF Redfish or IPMI?
  - How to provision raw storage?
  - Liveness of drives/enclosure
- How to fence individual drives?
- Security model
- Advanced features in EBOD?
  - Low latency, integrity check, caching, ...