Creating Differentiated Storage Offerings Using Cinder Volume Types

Bob Callaway, PhD
Cloud Solutions Group, NetApp
@rdcallaw
Agenda

- Why Offer Differentiated Storage Service Levels?
- Quick Overview of OpenStack Block Storage (Cinder)
- Cinder Volume Types
  - Definition
  - Extra specs (with examples)
  - QoS specs (with examples)
- How to differentiate using Cinder Volume Types
- Brief Demo
- Q & A
Why you should consider multiple service levels for block storage

- All workloads are not created equal
- Differentiation can lead to greater provider efficiencies, revenue
- Cinder provides an consistent API to access different types of storage through a common object model
Example of Differentiation Strategies

- **Performance**
  - Media format, IOPS limits, thin/thick provisioning

- **Data protection**
  - Basic, replicated, snapshot/backup intervals

- **Environment-specific**
  - Storage protocol, vendor, driver
Amazon EBS Service Levels
Rackspace Cloud Block Storage Example
OpenStack Block Storage Service (Cinder)

- Self-service management of block storage entities (volumes)
- Core OpenStack project; forked from Nova (was nova-volume)
- Typically deployed as part of a larger cloud; can be used alone
- Cinder is the “control plane”; not in the data path of storage traffic
OpenStack Block Storage Service (Cinder)
Volume Type

- An abstract collection of criteria used to describe a particular service level
- *Defined by cloud administrators* as a list of key/value pairs
- **Utilized by end users** when volumes are created
  - *Volumes can be retyped after creation*
Volume Types: Names are arbitrary

Cinder Volume Types

- Gold
- Silver
- Bronze

- Thin provisioned
- Deduplication
- Hourly Backups
- Replicated
- Isolation
- Highly Available

Cinder Volume Types

- Archival
- Analytics
- Streaming
- Temporal
- Database
Volume Types: Names are arbitrary
I’d like a 520GB Silver volume

<table>
<thead>
<tr>
<th>Volume Type</th>
<th>Definition</th>
<th>Cinder backend A</th>
<th>Cinder backend B</th>
</tr>
</thead>
<tbody>
<tr>
<td>netapp_dedup</td>
<td>{netapp_dedup=True}</td>
<td>Driver capabilities</td>
<td>Driver capabilities</td>
</tr>
</tbody>
</table>

Available capacity, capabilities

Provision in backend B
Volume Type: Default Capabilities

- Cinder backends periodically advertise capabilities (key/value pairs) to the Cinder scheduler
  - `volume_backend_name`
    - The name of the backend as defined in `cinder.conf`
  - `vendor_name`
    - The name of the vendor who has implemented the driver (e.g. NetApp, ‘Open Source’)
  - `driver_version`
    - The version of the driver (e.g. 1.0)
  - `storage_protocol`
    - The protocol used by the backend to export storage to clients (e.g. iSCSI, NFS, FC)
Volume Type: Extra Specs

- Driver/vendor specific capabilities

<table>
<thead>
<tr>
<th>Extra Spec</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>netapp:raid_type</td>
<td>String</td>
<td>Limit the candidate volume list based on one of the following raid types: raid4, raid_dp.</td>
</tr>
<tr>
<td>netapp:disk_type</td>
<td>String</td>
<td>Limit the candidate volume list based on one of the following disk types: ATA, BSAS, EATA, FCAL, FSAS, LUN, MSATA, SAS, SATA, SCSI, XATA, XSAS, or SSD.</td>
</tr>
<tr>
<td>netapp_thin_provisioned</td>
<td>Boolean</td>
<td>Limit the candidate volume list to only the ones that support thin provisioning on the storage controller.</td>
</tr>
<tr>
<td>netapp_dedup</td>
<td>Boolean</td>
<td>Limit the candidate volume list to only the ones that have deduplication enabled on the storage controller.</td>
</tr>
</tbody>
</table>

- This is not an exclusive list – and each vendor has their own…
- Refer to the Config Reference guide @ docs.openstack.org
Volume Type: QoS Specs

- Define Quality-of-Service (QoS) targets for volumes
- Can be enforced either at
  - the hypervisor “frontend”
  - the storage subsystem “backend”
  - both
Volume Type: QoS Specs - Limits

- **Frontend**: Limit by throughput
  - Total bytes/sec, read bytes/sec, write bytes/sec

- **Frontend**: Limit by IOPS
  - Total IOPS/sec, read IOPS/sec, write IOPS/sec

- **Backend**: Vendor specific fields
  - HP 3PAR (IOPS, tput: min, max; latency, priority)
  - Solidfire (IOPS: min, max, burst)
  - NetApp* (QoS Policy Group)
  - Huawei* (priority)

* defined through extra specs
# Using Volume Types to Create Differentiated Storage Offerings

<table>
<thead>
<tr>
<th>Volume Type</th>
<th>Extra Specs</th>
<th>QoS Specs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gold</td>
<td><code>{netapp:disk_type=SSD, netapp_thick_provisioned=True}</code></td>
<td><code>{}</code></td>
</tr>
<tr>
<td>Silver</td>
<td><code>{}</code></td>
<td><code>{total_iops_sec=500}</code></td>
</tr>
<tr>
<td>Bronze</td>
<td><code>{volume_backend_name=lvm}</code></td>
<td><code>{total_iops_sec=100}</code></td>
</tr>
</tbody>
</table>
# Using Volume Types to Create Differentiated Storage Offerings

<table>
<thead>
<tr>
<th>Volume Type</th>
<th>Extra Specs</th>
<th>QoS Specs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Archival</td>
<td><code>{netapp_mirror=\textbf{True}, netapp_compression=\textbf{True}}</code></td>
<td><code>{\text{read\_bytes\_sec}=5000}</code></td>
</tr>
<tr>
<td>Analytics</td>
<td><code>{netapp\_thick\_provisioned=\textbf{True}}</code></td>
<td><code>{}</code></td>
</tr>
<tr>
<td>Streaming</td>
<td><code>{netapp\_disk\_type=SSD}</code></td>
<td><code>{}</code></td>
</tr>
<tr>
<td>Temporal</td>
<td><code>{netapp\_thin\_provisioned=\textbf{True}}</code></td>
<td><code>{\text{total\_iops\_sec}=100}</code></td>
</tr>
<tr>
<td>Database</td>
<td><code>{storage\_protocol=\textbf{iscsi}}</code></td>
<td><code>{}</code></td>
</tr>
</tbody>
</table>
Demo
Questions?
Thank you