An Introduction to OpenStack Cinder

Sean McGinnis
Dell

Cinder Mission Statement

To implement services and libraries to provide on demand, self-service access to Block Storage resources. Provide Software Defined Block Storage via abstraction and automation on top of various traditional backend block storage devices.

What is Cinder?

- Created in the OpenStack Folsom release (2012)
  - Spun off from Nova volume
- Cinder manages block storage
  - Different than shared file storage – that's Manila
  - Different than object storage – that's Swift
  - Focuses on:
    - Attaching volumes to VM instances
    - Booting from volumes
    - Providing management abstraction over a variety of backends
  - Volumes have lifecycles independent of VMs

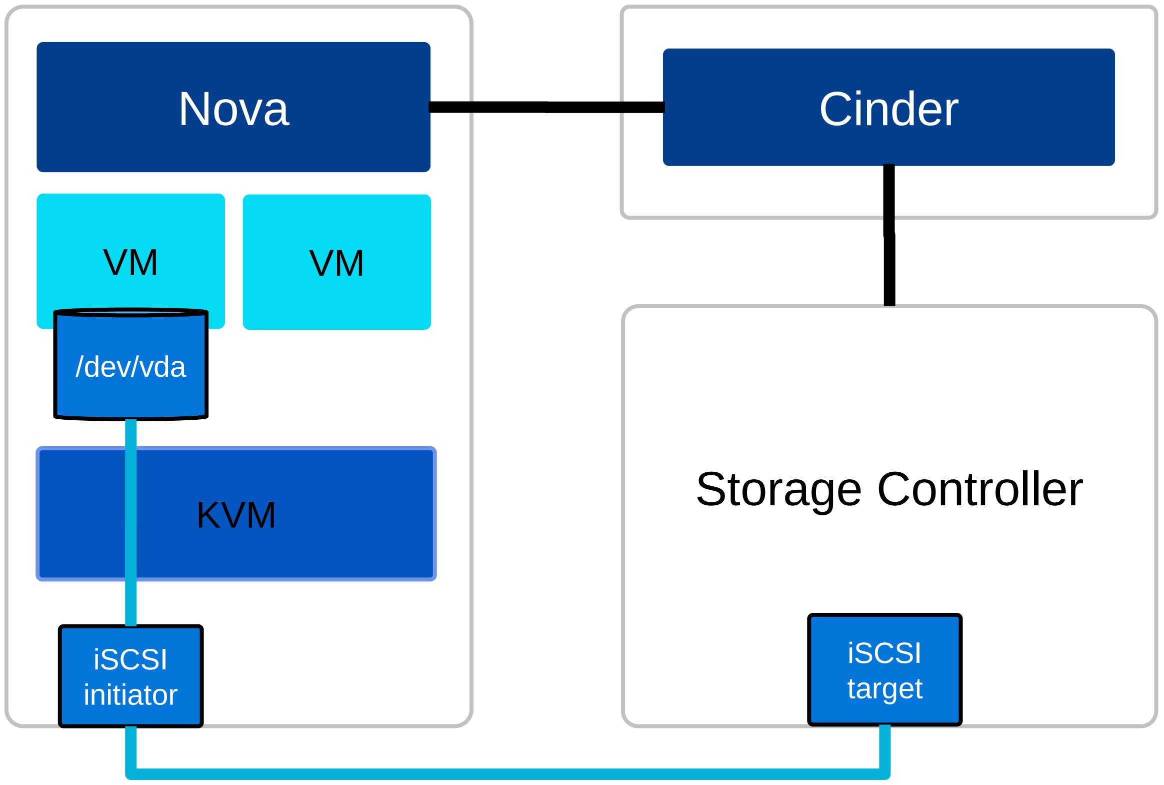
How Does Cinder Fit?

- Cinder provides API’s to interact with vendors’ storage backends
- Exposes vendors storage hardware to the cloud
- Provides persistent storage to VMs, containers, bare metal
- Enables end users to manage their storage without knowing where that storage is coming from
  - Create/delete
  - Attach/detach
  - Snapshot
  - Backup

How Does Cinder Fit?

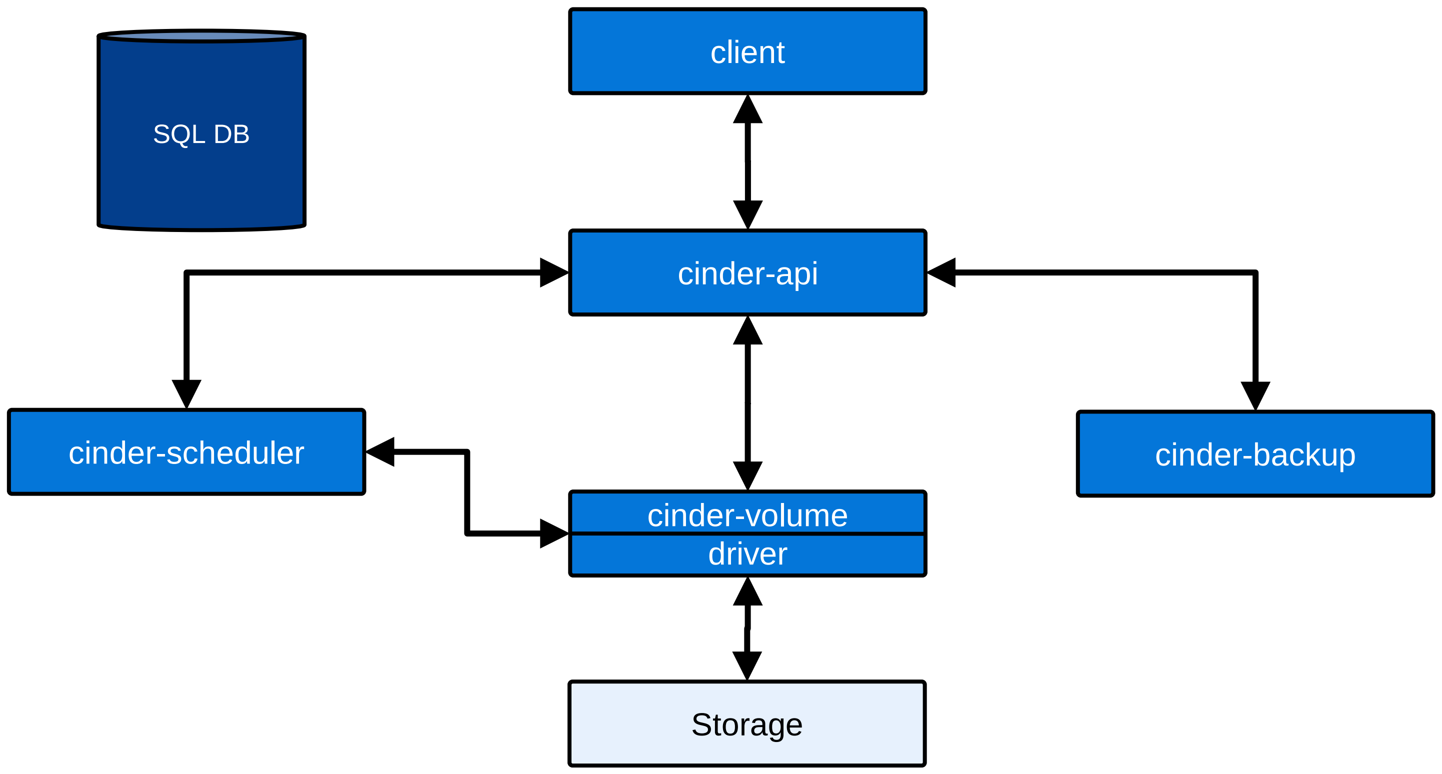
<u>Legend</u>

■ Control Path
■ Data Path



Note that iSCSI is just an example – several additional protocols are supported
(e.g., FC, NFS)

Cinder Architecture



Cinder Services

- API
  - REST interface to Cinder
  - Generally runs on control node
- Scheduler
  - Takes requests from the API service
  - Works with the volume services to satisfy requests
  - Generally runs on control node

Cinder Services

- Volume
  - Interacts with vendor storage backends
    - Create
    - Manage
    - Export
  - Can run on control node, or a different host
- Backup
  - Interface to backup volumes to storage like Swift, TSM, Google Cloud Storage, etc.

cinder.conf

- Used by all of Cinder's services
- Usually located at /etc/cinder/cinder.conf
- Provides settings such as database connection string, message queue settings, services options, etc.
- Sections for defining backend configuration
  - Driver to load
  - Driver specific settings

cinder.conf

- Set debug=True and verbose=True to get additional logging output
  - By default logs go to /var/log/cinder
  - Devstack defaults to /opt/stack/logs
- Any changes made to cinder.conf require the Cinder services to be restarted before changed settings take effect

Cinder Drivers

- Block Device Driver (local)
- Blockbridge (iSCSI)
- CloudByte (iSCSI)
- Coho (NFS)
- Datera (iSCSI)
- Dell Equallogic (iSCSI)
- Dell Storage Center (iSCSI/FC)
- Disco (disco)
- DotHill (iSCSI/FC)
- DRBD (DRBD/iSCSI)
- EMC VMAX (iSCSI/FC)
- EMC VNX (iSCSI/FC)
- EMC XtremIO (iSCSI/FC)
- EMC ScaleIO (scaleio)
- Fujitsu ETERNUS (iSCSI/FC)
- GlusterFS (GlusterFS)
- HGST (NFS)
- HPE 3PAR (iSCSI/FC)
- HPE LeftHand (iSCSI)
- HPE MSA (iSCSI/FC)
- HPE XP (FC)
- Hitachi HBSD (iSCSI/FC)
- Hitachi HNAS (iSCSI/NFS)
- Huawei (iSCSI/FC)
- IBM DS8000 (FC)
- IBM Flashsystem (iSCSI/FC)
- IBM GPFS (GPFS)
- IBM Storwize SVC (iSCSI/FC)
- IBM XIV (iSCSI/FC)
- Infortrend (iSCSI/FC)
- Lenovo (iSCSI/FC)
- **LVM (iSCSI) – *Reference****
- NetApp ONTAP (iSCSI/NFS/FC)
- NetApp E Series (iSCSI/FC)
- Nexenta (iSCSI/NFS)
- **NFS – *Reference***
- Nimble Storage (iSCSI)
- Oracle Zfssa (iSCSI/NFS)
- Pure Storage (iSCSI/FC)
- ProphetStor (iSCSI/FC)
- Quobyte (quobyte)
- **RBD (Ceph) - *Reference***
- Scality SOFS (scality)
- Sheepdog (sheepdog)
- SMBFS (SMB)
- SolidFire (iSCSI)
- Tegile (iSCSI/FC)
- Tintri (NFS)
- Violin (FC)
- VMware (VMDK)
- Virtuozzo Storage (NFS)
- Windows (SMB)
- X-IO (iSCSI/FC)

(Drivers in **bold** are the reference for the architecture)

Minimum Driver Features

Drivers must implement support for the core features:

- Volume Create/Delete
- Volume Attach/Detach
- Snapshot Create/Delete
- Create Volume from Snapshot
- Copy Image to Volume
- Copy Volume to Image
- Clone Volume
- Extend Volume

Fibre Channel Support

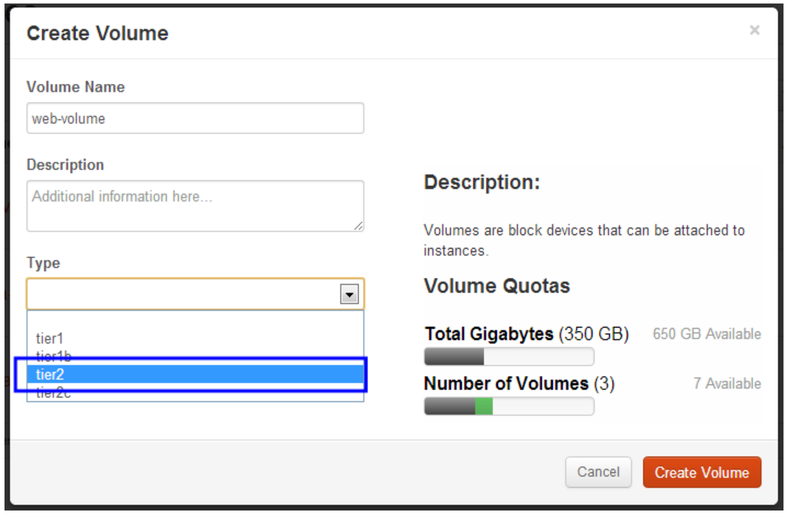
- Fibre Channel Zone Manager
- Dynamically create and delete zones
- Drivers to support fabric management
    - Brocade
    - Cisco

Clients

- Cinder Client
  - python-cinderclient is the command line interface to Cinder
    - ‘cinder <command>’
  - Also client library for Python code
  - Uses REST to communicate with the cinder-api service
- OpenStack Client
  - All projects moving to OpenStack Client
    - ‘openstack volume <command>’

Volume Types

- Used to request properties of volumes during creation
- Can also control users' access to different storage
- Only admins can create volume types
- Users specify the volume type when they create a volume



Create Volume

Volume Name: web-volume

Description: Additional information here...

Type:

tier1
tier1b
tier2
tier2c

Description:

Volumes are block devices that can be attached to instances.

Volume Quotas

Total Gigabytes (350 GB) 650 GB Available

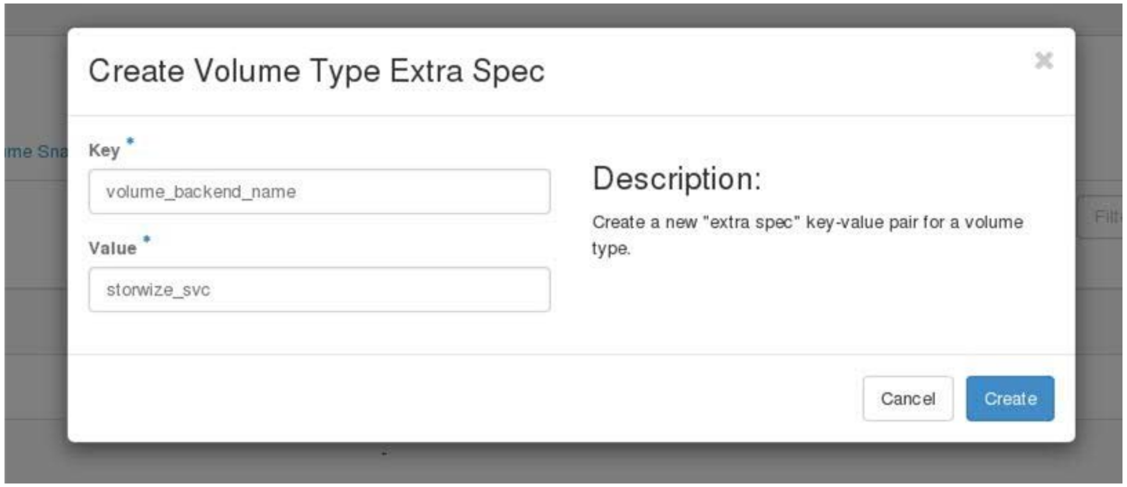
Number of Volumes (3) 7 Available

Volume Type Extra Specs

- Extra specs are used to set type properties
- Some standard, some vendor specific
  - volume_backend_name=lvm-1
  - sio:provisioning_type=thin
  - hp3par:persona=3
- Extra specs are only visible to the admin

Volume Type Extra Specs

- Extra specs can be modified via UI, CLI, or API



Create Volume Type Extra Spec

Key * volume_backend_name

Value * storwize_svc

Description:

Create a new "extra spec" key-value pair for a volume type.

```
$ cinder type-create GoldVolume
$ cinder type-key GoldVolume set storagetype:storageprofile=highpriority
$ cinder type-create BronzeVolume
$ cinder type-key BronzeVolume set storagetype:storageprofile=lowpriority
```

Retype and Migration

- Retype is used to change settings of a volume
  - Some retypes can happen without moving data
  - Some require moving the volume to a different backend
- Migration is used to move a volume between two different backends
  - For example - from LVM to Ceph

Retype

Change volume types:

name:	dellsc1-nightly
extra_specs:	{volume_backend_name: sn12345, storagetype:replayprofile: nightly}

name:	dellsc1-hourly
extra_specs:	{volume_backend_name: sn12345, storagetype:replayprofile: hourly}

```
# cinder create 1 --name vol1 --volume-type dellsc1-nightly
# cinder retype vol1 dellsc1-hourly
```

Retype with Migration

Change volume types:

name:	dellsc1-nightly
extra_specs:	{volume_backend_name: sn12345, storagetype:replayprofile: nightly}

name:	dellsc2-hourly
extra_specs:	{volume_backend_name: sn54321, storagetype:replayprofile: hourly}

```
# cinder create 1 --name vol1 --volume-type dellsc1-nightly
# cinder retype vol1 dellsc2-hourly
```
FAILS!

Retype with Migration

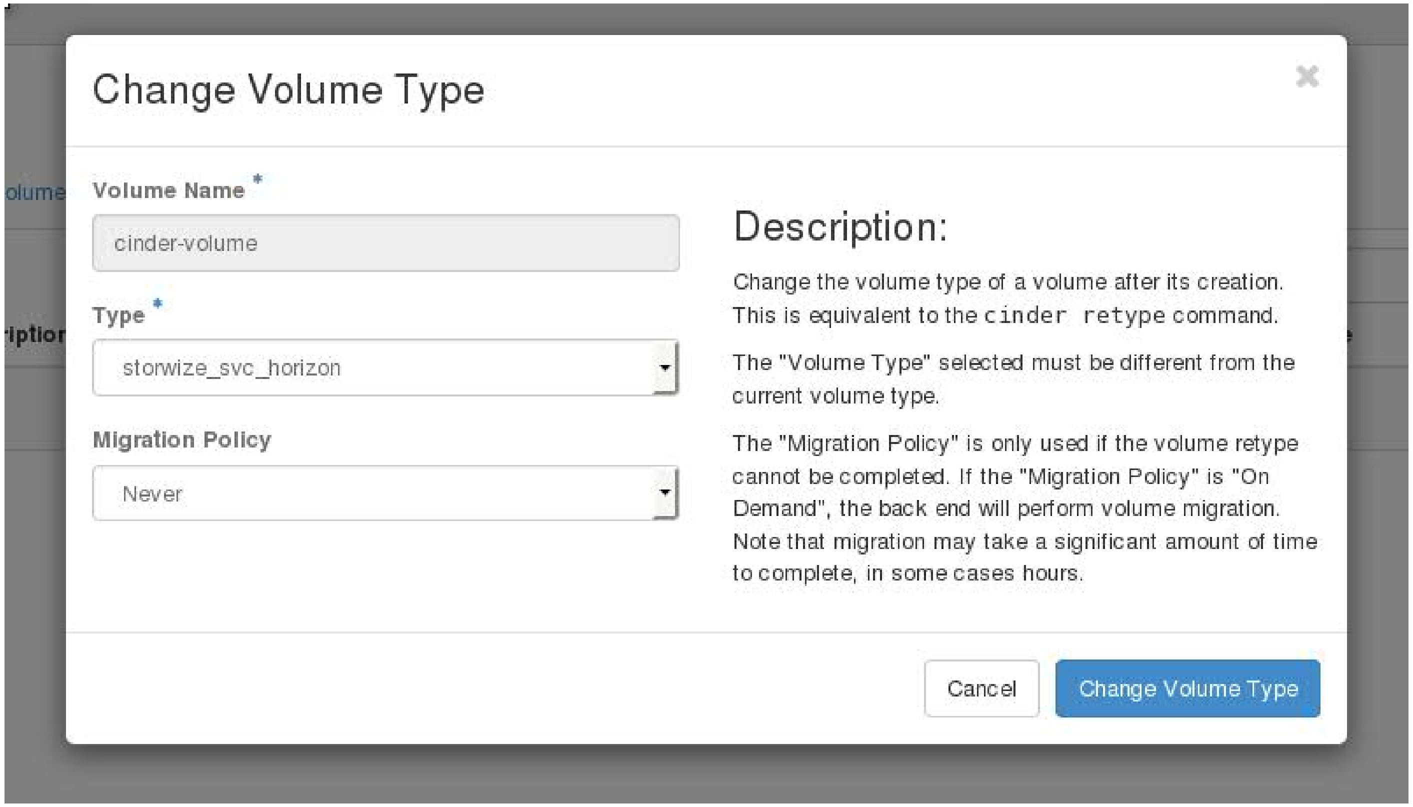
Change volume types:

name:	dellsc1-nightly
extra_specs:	{volume_backend_name: sn12345, storagetype:replayprofile: nightly}

name:	dellsc2-hourly
extra_specs:	{volume_backend_name: sn54321, storagetype:replayprofile: hourly}

```
# cinder create 1 --name vol1 --volume-type dellsc1-nightly
# cinder retype vol1 dellsc2-hourly --migration-policy on-demand
```

Retype via UI


Change Volume Type

Volume Name *

cinder-volume

Type *

storwize_svc_horizon

Migration Policy

Never

Description:

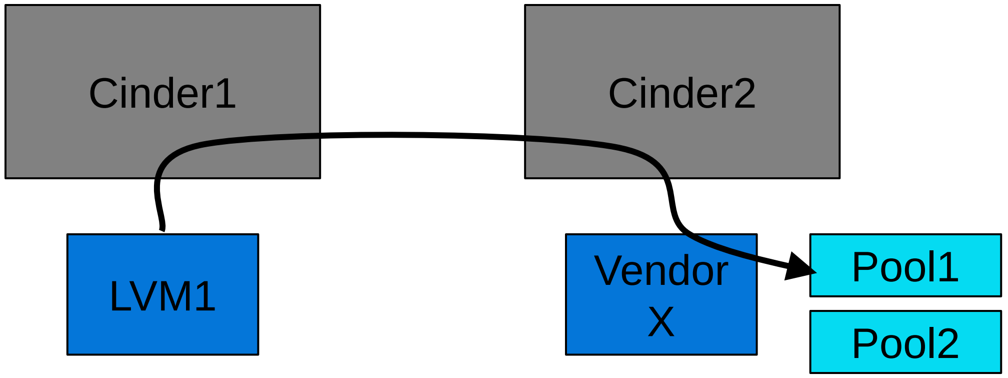
Change the volume type of a volume after its creation. This is equivalent to the `cinder retype` command.

The "Volume Type" selected must be different from the current volume type.

The "Migration Policy" is only used if the volume retype cannot be completed. If the "Migration Policy" is "On Demand", the back end will perform volume migration. Note that migration may take a significant amount of time to complete, in some cases hours.

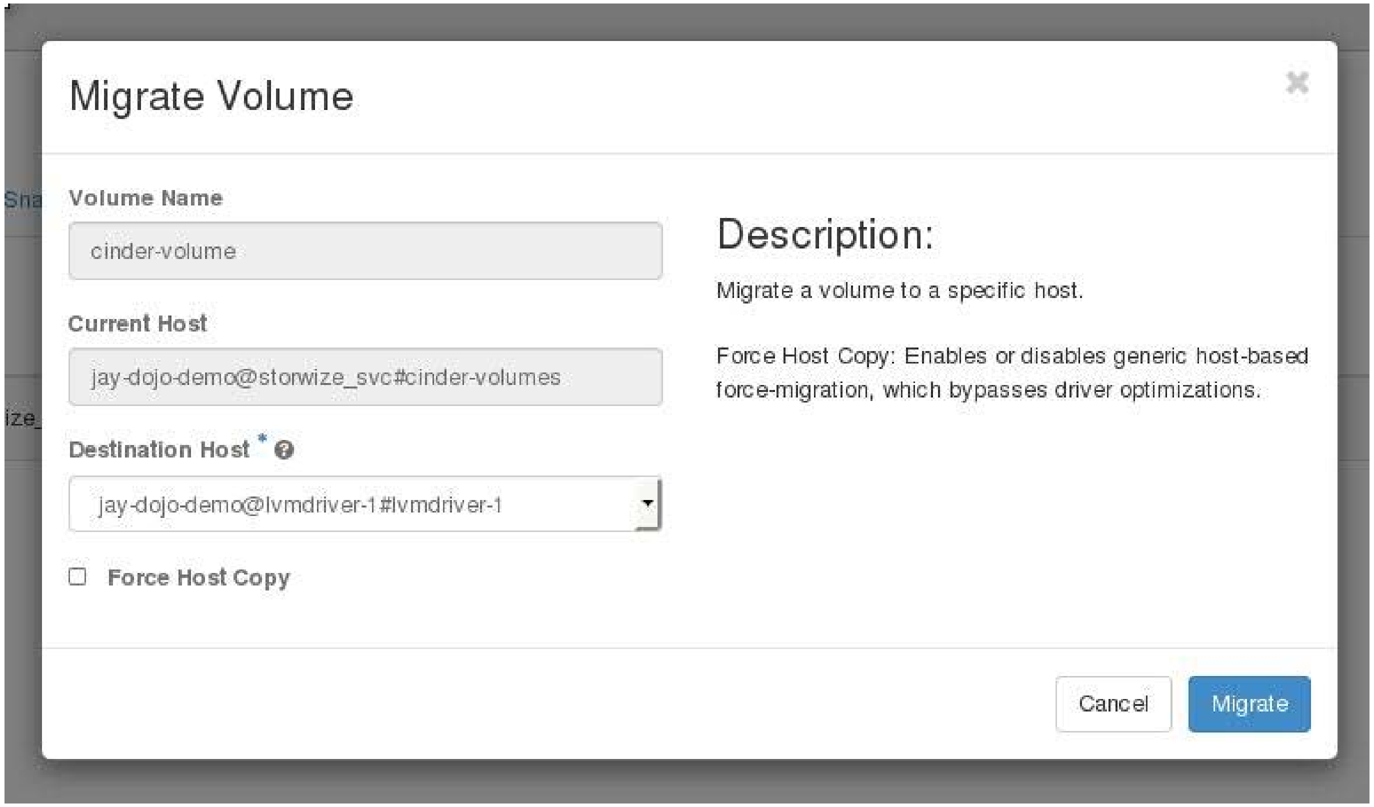
Migration

Migrating volume to new host:



```
# cinder create 1 --name vol1 --volume-type thin_provisioned
# cinder migrate vol1 Cinder2@VendorX#Pool1
```

Migration via UI



Migrate Volume

Volume Name: cinder-volume

Current Host: jay-dojo-demo@storwize_svc#cinder-volumes

Destination Host *: jay-dojo-demo@lvmdriver-1#lvmdriver-1

☐ Force Host Copy

Description:

Migrate a volume to a specific host.

Force Host Copy: Enables or disables generic host-based force-migration, which bypasses driver optimizations.

Cinder Backup

- Backup and restore volumes
- Must be either in Available state or able to create and mount snapshot
- Several backup drivers supported:
  - Ceph
  - Google Cloud Storage
  - NFS
  - Posix Filesystem
  - Swift
  - Tivoli Storage Manager

Cinder Backup

- Backup via CLI, UI, or API
- Needs to be enabled in Horizon
  - /etc/openstack-dashboard/local_settings.py
  - OPENSTACK_CINDER_FEATURES = {‘enable_backup’: True}
- No cron type scheduling in Cinder

```
# cinder backup-create --name MyBackup --description “pre patching” \
         --incremental vol1
# cinder backup-restore a006718b-b583-4d59-9ddb-d1109dc98ebf
```

Cinder Replication

- Basic support for replication
- Replicate Site A to Site B
- Site A is on fire, failover all volumes to Site B
- New in Mitaka – supported backends and functionality will continue to be expanded

Ongoing/Future Work

- Better support for replication
- Active/Active High Availability
- More backend storage support
- Better user error reporting

Thank You!