Universität Zürich[UZH]

S3IT: Service and Support for Science IT

# Cinder

Or: Openstack's Block Storage Service

Tyanko Aleksiev <tyanko.aleksiev@s3it.uzh.ch>

31/08/2014

# Cinder's role inside OpenStack

Cinder is a service used to provide Virtualized Block Storage Devices to the end users of OpenStack without requiring them to have any knowledge of where that storage is actually deployed or on what type of device.

The hypervisor simply provides a standard SCSI volume to the VM, very likely when a user plug-in a USB stick in his laptop.

# Cinder components: cinder-api

The role of *cinder-api* is to accept and manage API calls to:

- create, delete and list volumes and snapshots,
- attach and detach volumes (called by nova).

# Cinder components: cinder-scheduler

The role of *cinder-scheduler* is to:

- manage the cooperation between cinder-api and cinder-volume through the message queue and the DB,
- examine and choose the back-end which is going to provide the storage capacity requested by the user.

Different filters are available as plug-ins.

# Cinder components: cinder-volume

The role of *cinder-volume* is to:

- handle and execute the requests coming from cinder-scheduler,
- interact directly with variuos types of storage back-ends.

Multiple cinder-volumes instances can be run either on the same node providing access to different type of storage back-ends or on multiple nodes to increase scalability.

# Cinder components: Message queues

*Messaging queues* are used for communication between the services. RabbitMQ is a common choice and is widely used inside OpenStack.

# Cinder components: Database

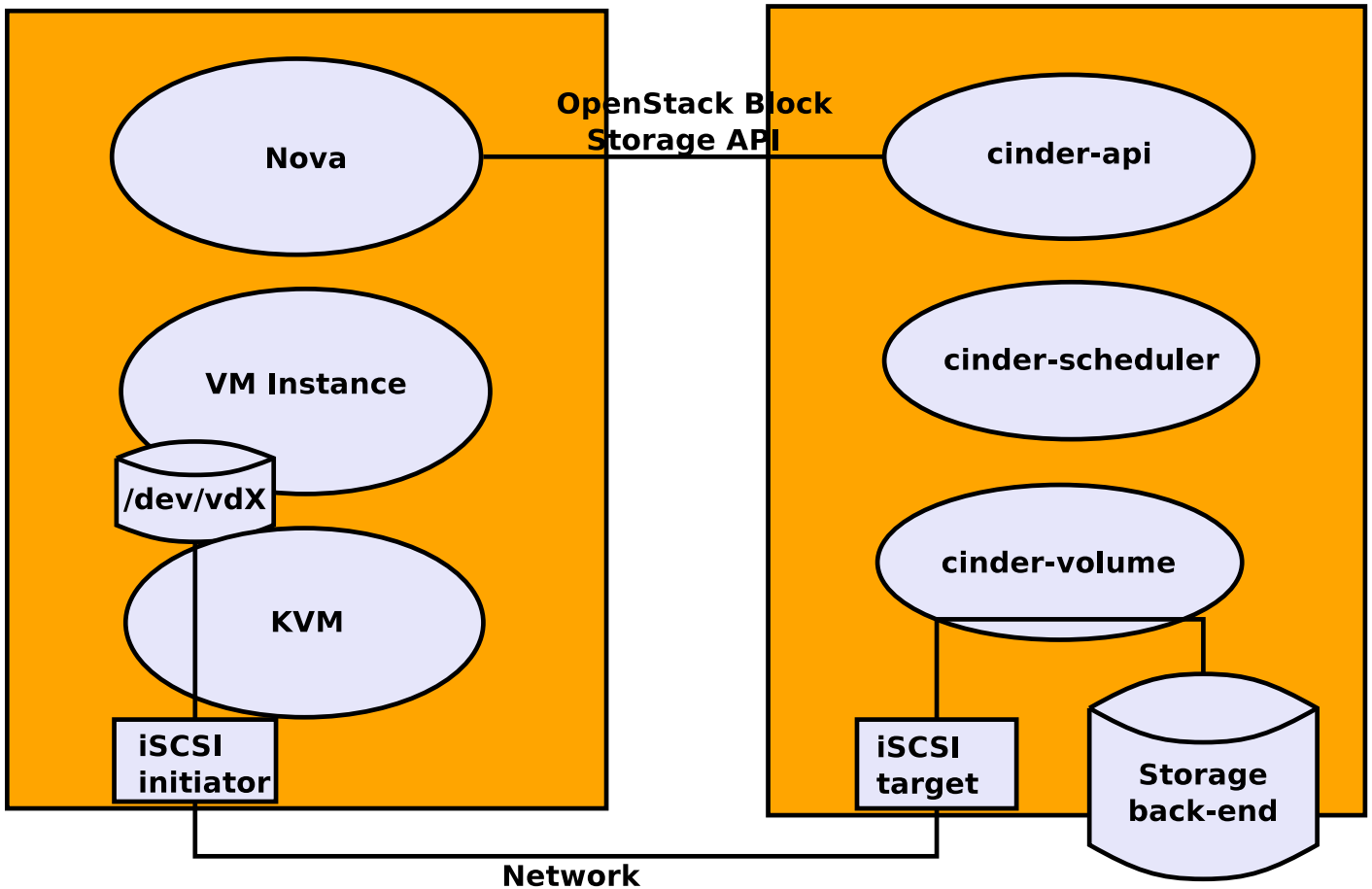
A *Database* is used for storing volume metadata. A common implementation choice here is MySQL.

# Cinder components: Storage

*Storage* is the back-end from where the volumes are actually served. Different back-end types are supported:

- LVM
- Ceph,
- Gluster,
- NFS,
- ZFS,
- Sheepdog,
- Some commercial (IBM, NetApp...)

# Cinder: components interaction 1/3



# Cinder: components interaction 2/3

Interaction example:

- nova calles cinder-api with connection information (hostname, iSCSI initiator name, etc),
- cinder-api writes the information in the DB and passes the message to cinder-scheduler,
- cinder-scheduler chooses the back-end to be used for the volume provisioning and calles cinder-volume,
- cinder-volumes provides the parameters to the storage back-end driver,

# Cinder: components interaction 3/3

Interaction example:

- the storage back-end allows the connection and allocates the storage space,
- cinder-volume returns then the connection information to nova,
- nova creates the connections using the given parameteres and passes the volume device/file to the hypervisor.

# Notes and Remarks

- Logs directory is: `/var/log/cinder`
- Files you are going to edit often:
  - cinder-api conf. file is:
    `/etc/cinder/cinder-api.conf`
  - cinder-volume conf. file is:
    `/etc/cinder/cinder-volume.conf`
- We will see everything in more detail during the tutorial.

# Useful Links

- Configure Cinder
- Cinder Project Documentation