



Ironic

Project Update, OpenStack Summit Sydney

Julia Kreger - TheJulia - juliaashleykreger@gmail.com

Hironori Shiina - hshiina - shiina.hironori@jp.fujitsu.com

What is Ironic?

A project to provide an API service and tooling to facilitate the lifecycle management of a hardware in a cloud.

In essence, Bare-Metal as a Service.

Ironic started as “Nova Baremetal”, and now provides a virt driver for Nova, which has resulted in 13% of the OpenStack deployments utilizing bare metal for instances.

145 unique contributors contributed to Ironic during the Pike cycle, from 33 different organisations.

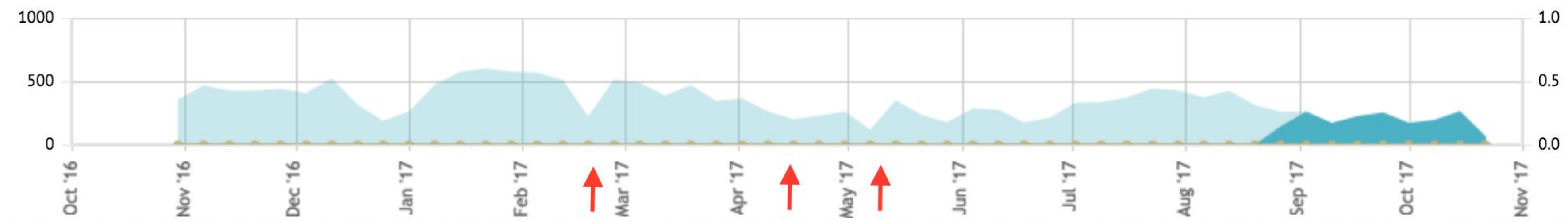


IRONIC

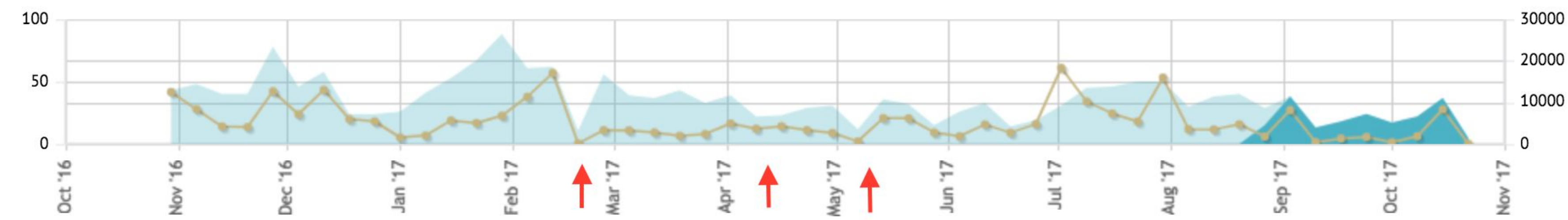
an OpenStack Community Project

Irony Contributor Velocity

Community Review Velocity



Community Commit Velocity



Highlighted events (Left to Right): Pike PTG, OSIC Impact, Boston Summit

OpenStack Pike Ironi Features

Rolling upgrades!

- ➡ Enables downtime-less N-1 upgrades

Initial Boot from Volume support!

- ➡ iPXE based PXE boot configurations
- ➡ iRMC virtual media boot

Hardware Types

- ➡ Easier to now change driver behavior!
- ➡ For all previous “classic” drivers
- ➡ Including a redfish hardware type!

OpenStack Pike IroniC Features

Networking

- ➡ Physical network information storage
- ➡ Port group information now sent to Neutron
- ➡ VIF attachment/detachment support

Migration to scheduling based on a custom resource class

Drivers that do not have Third Party CI have been removed

Many bug fixes!

OpenStack Queens IroniC

Client/Server API version defaults and negotiation updates

Reworking service authentication to use keystoneauth

Reference architecture documentation \o/

Neutron event Processing

Routed networks support

RESCUE mode \o/

OpenStack Queens IroniC

BIOS configuration framework

Ansible deployment interface

Deprecations

- ➡ Classic Drivers
- ➡ “ironic” CLI in favor of OpenStack Client

Beyond Queens

Support for Traits!

Use of traits to influence deployments

Removal of Classic Drivers

Removal of “ironic” CLI

Ironi sub-projects

ironic-inspector

“Discovery of hardware properties for a node managed by ironic.”

ironic-python-agent

“Agent to facilitate the deployment and undeployment of bare metal.”

ironic-ui

“A horizon panel to manage resources in ironic.”

bifrost

“An Ansible based toolkit for standalone Ironi usage.”

Ironic sub-projects

networking-baremetal

“Additional networking integration for bare metal.”

molteniron

“Tooling to assist with pure bare metal cloud management.”

sushy

“Library to facilitate communication to, and emulation of redfish.”

virtualbmc

“An emulated IPMI management controller for testing.”

Ironi-inspector Pike Features

Dependencies for introspection hooks

LLDP Processing Enhancements

API Usability Enhancements

Option to disable port creation upon (re)introspection

DHCP/PXE filter driver framework

Ironic-inspector Queens

Implementation of firewall and dnsmasq filter drivers

High availability support for active/active inspector deployments

Virtual Media boot integration with Ironic

UX Enhancements (inspect-wait state)

Ironic-inspector Queens

Some impact anticipated from merger of inspector functionality into Ironic.

Deprecation:

➡ No deprecations expected

Delayed:

➡ uWSGI support

Ironic-inspector beyond Queens

Possible merger of inspection capability into Ironic

Introspection rules processing as a service

Inspector may solely focus on node discovery

Cross-Project Work

Python 3 Compatibility:

- ironic - Ready - Gate change remains

- ironic-inspector - Blocked on Swift usage

- ironic-python-agent - Ready minus TinyIPA, this is being worked.

Policy in Code:

- ironic - Mostly completed in the past, minor items finished

- ironic-inspector - Completed in Queens

uWSGI:

- ironic - Mostly done - Reviews needed

- ironic-inspector - Blocked

Tempest Plugin Split:

- Blocked until zuulv3 jobs are in-tree.

Cross-Project Work

Scheduling/Traits Support

- ➡ Automatic resource class identification
- ➡ Forward direction on Traits
- ➡ Automatic discovery of Traits
- ➡ Scheduling for Traits

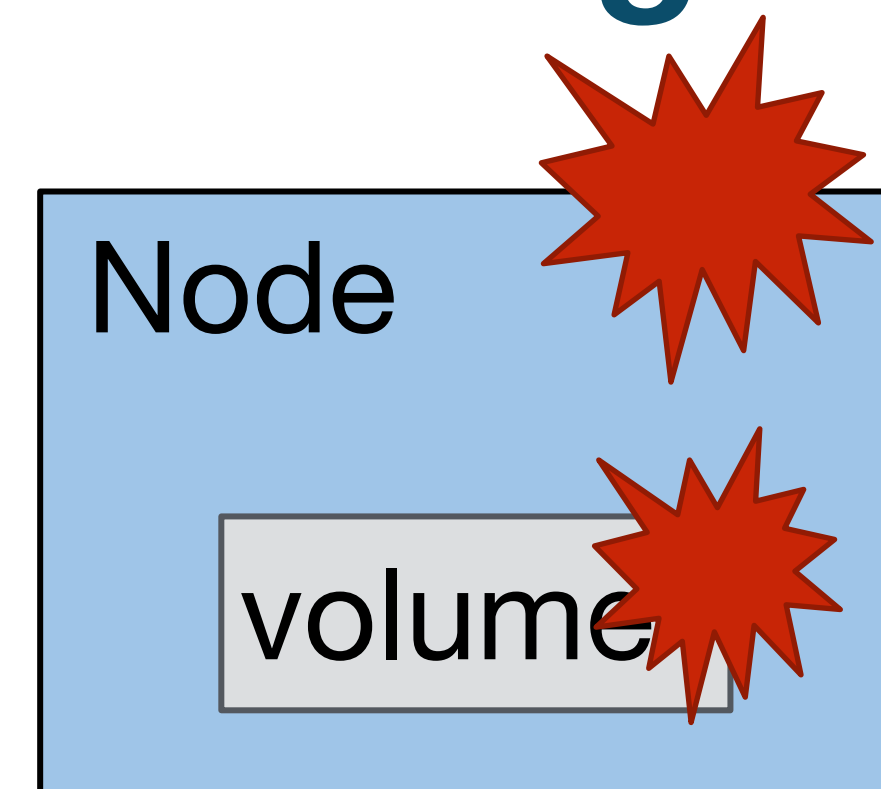
Dive into Boot from Volume

- ➡ Overview of Boot from Volume
- ➡ Making bare metal servers more reliable with BFV

BFV Separates Compute and Storage

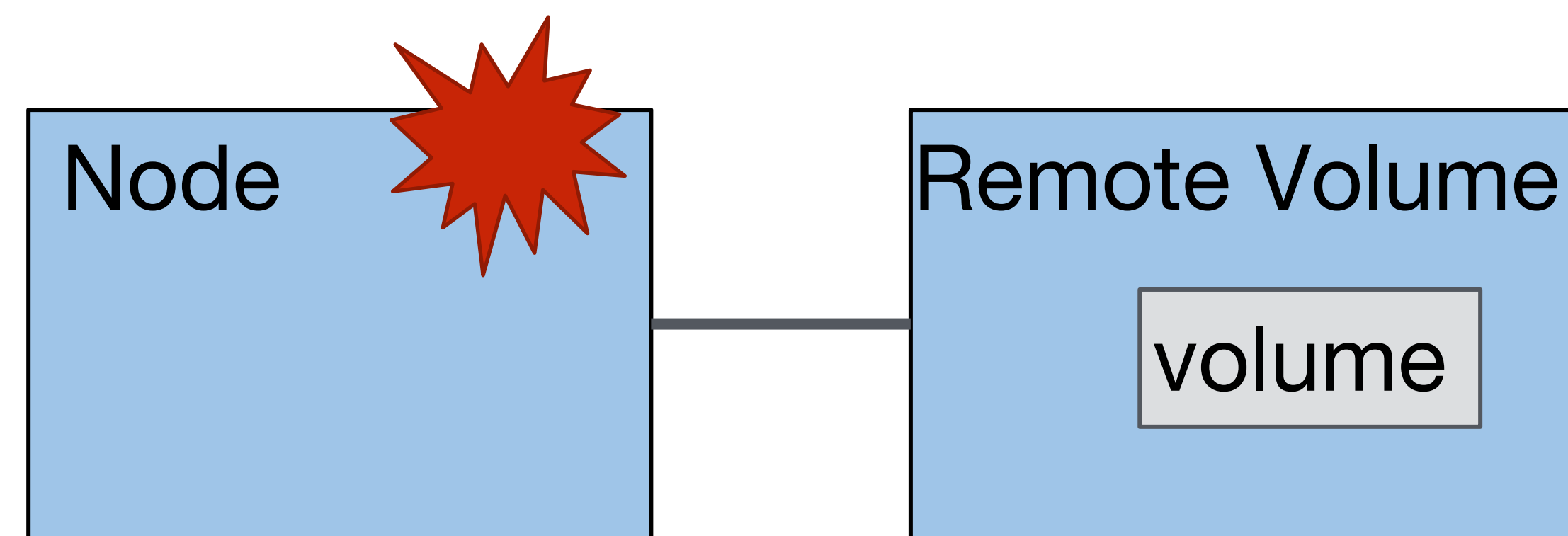
Boot from Local Disk

- ➡ Disk size is fixed
- ➡ Data cannot be accessed when a server has failed

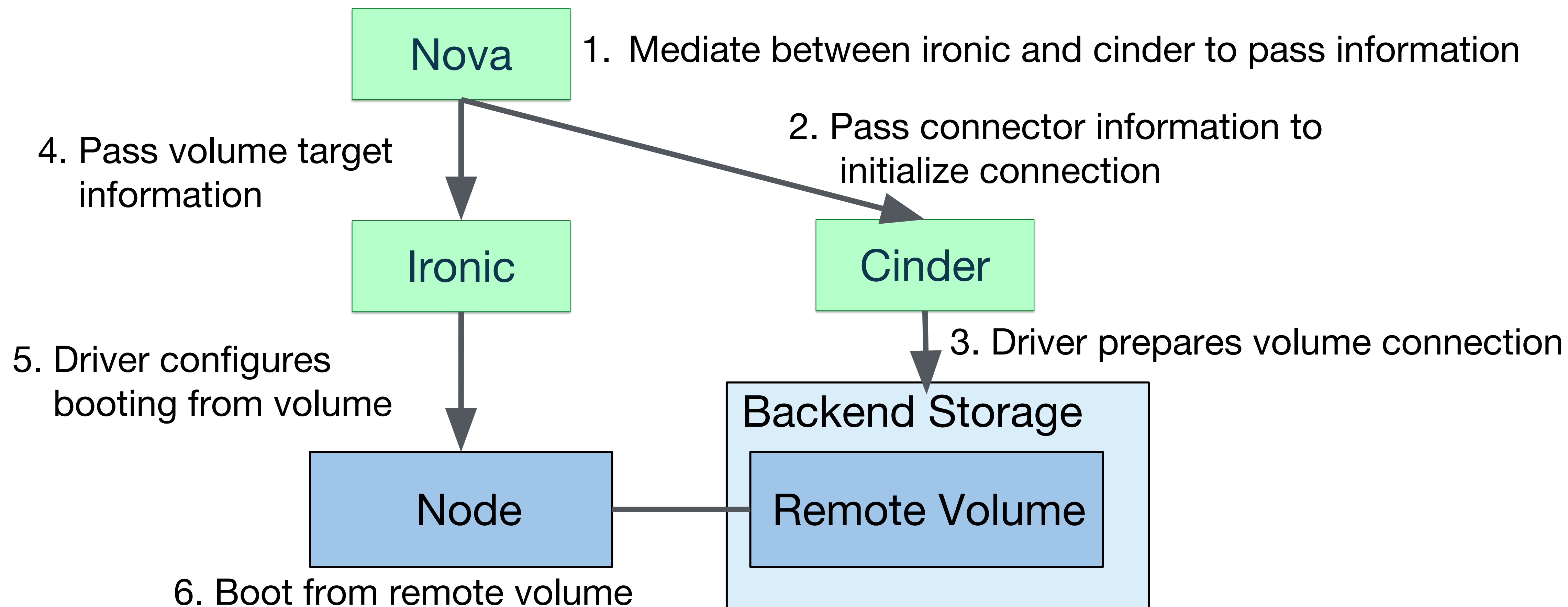


Boot from Remote Volume

- ➡ Disk size is flexible
- ➡ Data is available even if a server has failed



Overview of Booting from Volume



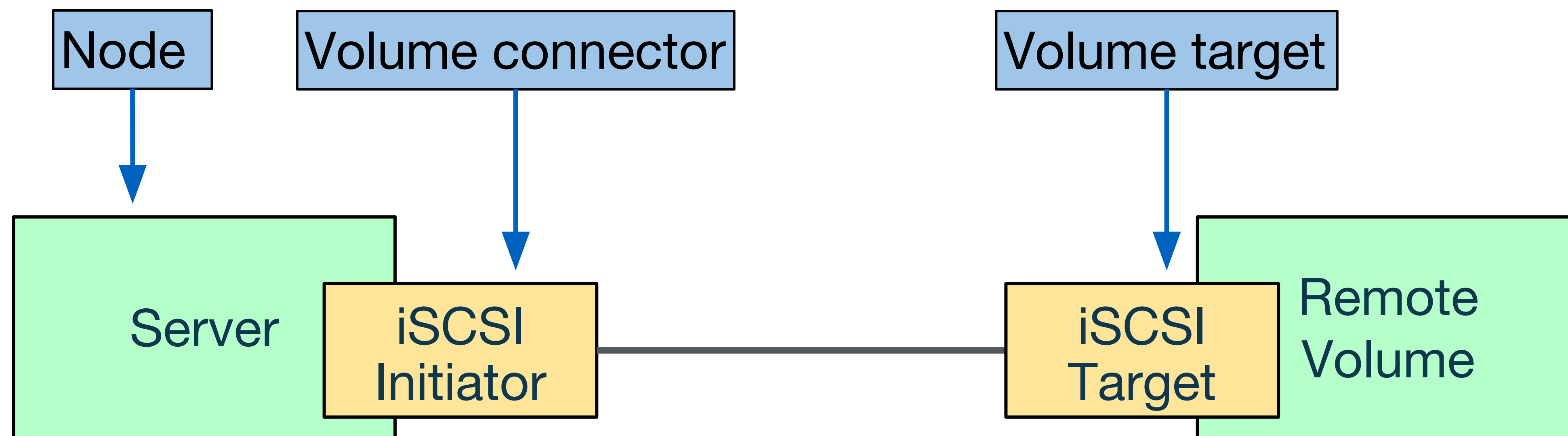
IroniC Resources for Boot from Volume

Volume connector:

contains connector information of a node

Volume Target:

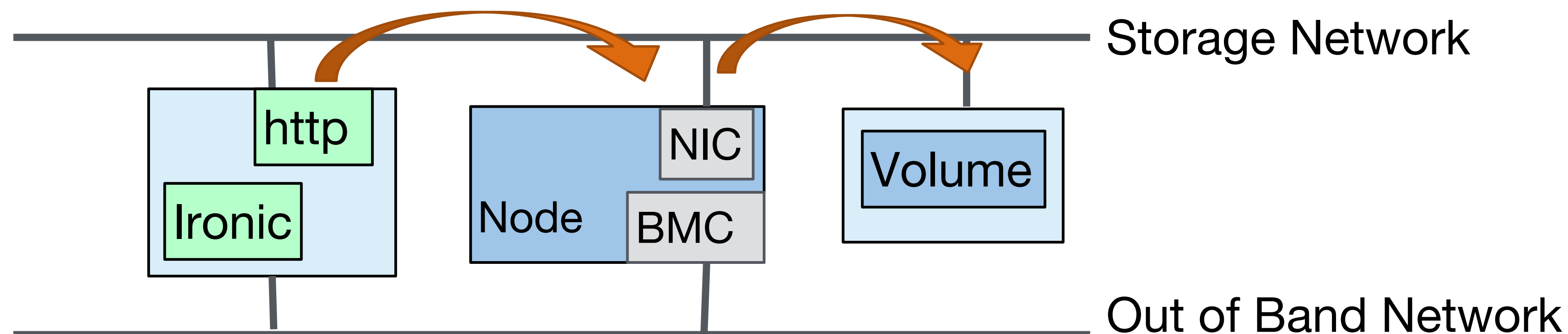
contains target information of a volume



Driver Implementation

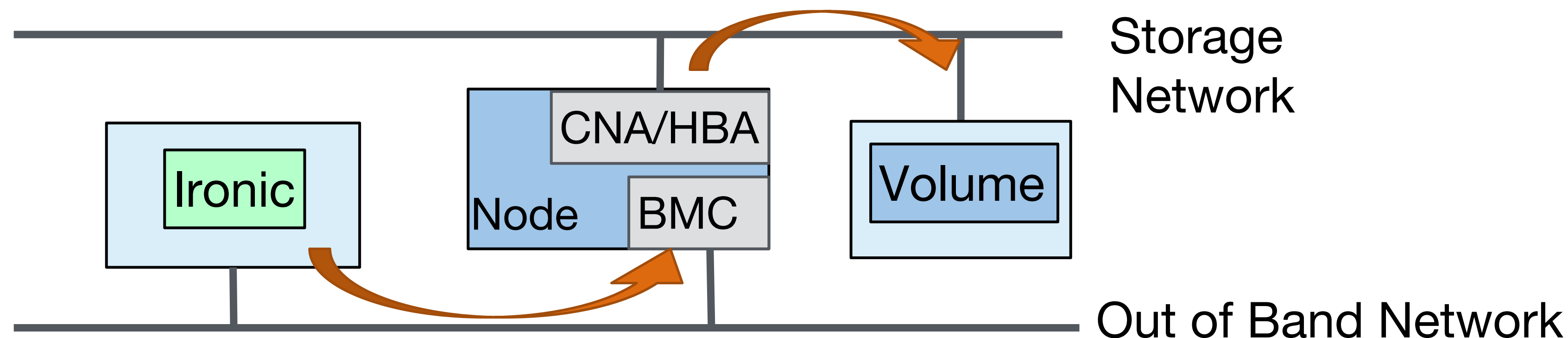
Generic implementation (iPXE based PXE boot configuration):

A node gets iSCSI volume information with iPXE.



Vendor implementation:

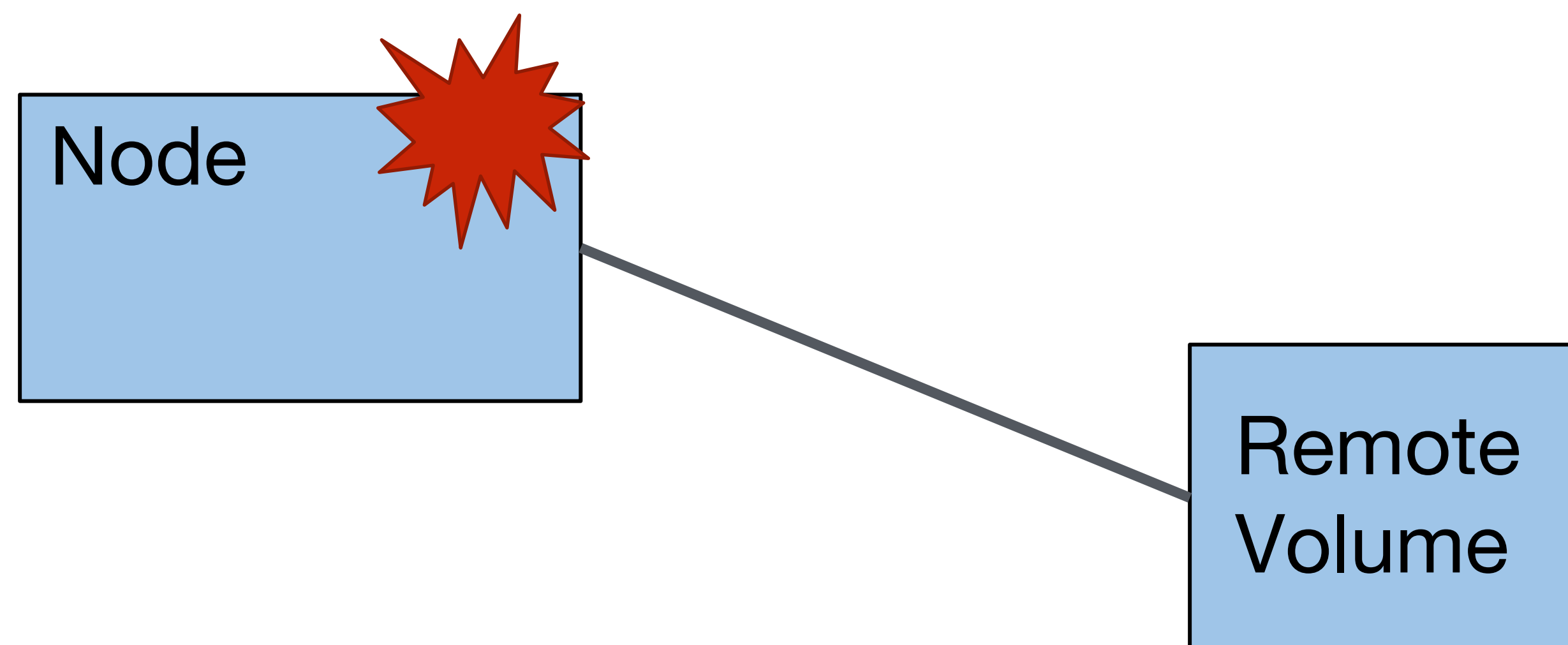
iRMC driver configures a node with BMC API for iSCSI and FibreChannel.



Switch over at Server Failure

When a server is failed, another server can be booted from the same volume.

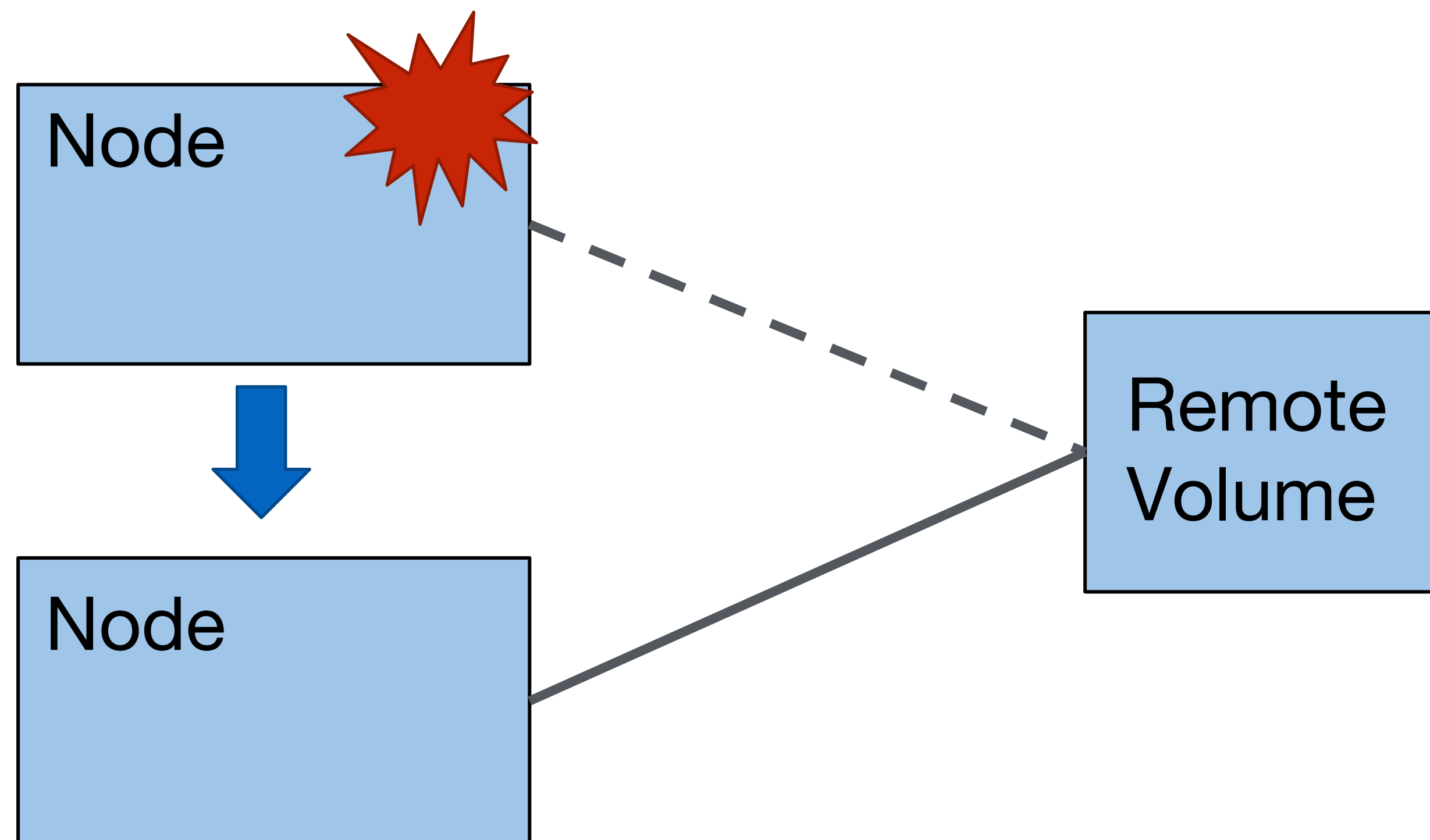
1. A server has failed.



Switch over at Server Failure

When a server is failed, another server can be booted from the same volume.

2. Power off the server.

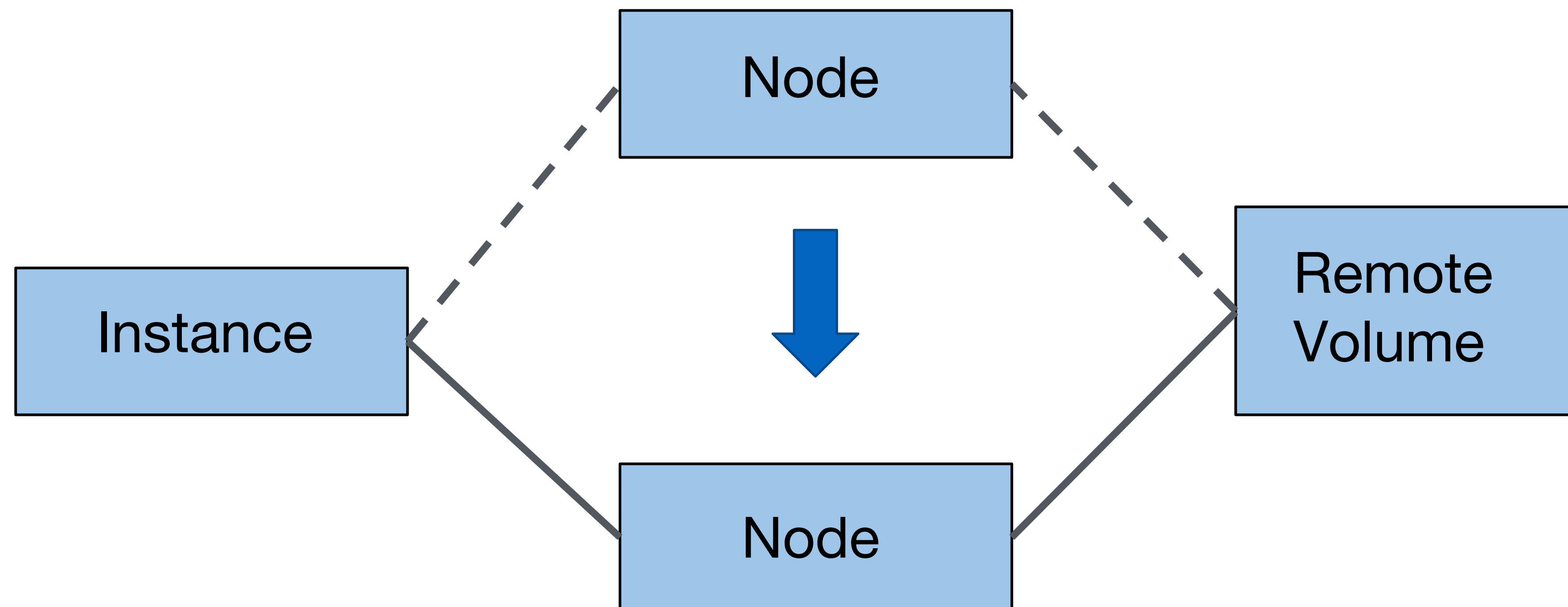


3. Deploy another server with the same volume.

Bare Metal Instance Switchover with nova (in progress)

With Compute service(nova), bare metal servers behind an instance can be switched by compute API such as cold-migration and resize.

Change for ironic driver in nova is working in progress.



How to give feedback

Come give the IroniC community feedback!
... Or just tell us we are crazy!

Tomorrow! 5:00 PM to 5:40 PM - Exhibition Centre - Level 4 - C4.10

How to contribute

Join us in #openstack-ironic

Contributor Guide can be found at <https://docs.openstack.org/ironic/>

Come ask questions at the Ironic project onboarding session!

Tomorrow: 11:40AM - 12:20 PM - Exhibition Centre - Level 4 - C4.6

Q&A

Thank you!



openstack



@OpenStack



openstack



OpenStackFoundation