Bright Cluster Manager 7.0

OpenStack Deployment Manual

Revision: c78c0f7

Date: Fri Sep 13 2024

Trademarks

Linux is a registered trademark of Linus Torvalds. PathScale is a registered trademark of Cray, Inc. Red Hat and all Red Hat-based trademarks are trademarks or registered trademarks of Red Hat, Inc. SUSE is a registered trademark of Novell, Inc. PGI is a registered trademark of The Portland Group Compiler Technology, STMicroelectronics, Inc. SGE is a trademark of Sun Microsystems, Inc. FLEXlm is a registered trademark of Globetrotter Software, Inc. Maui Cluster Scheduler is a trademark of Adaptive Computing, Inc. ScaleMP is a registered trademark of ScaleMP, Inc. All other trademarks are the property of their respective owners.

Rights and Restrictions

All statements, specifications, recommendations, and technical information contained herein are current or planned as of the date of publication of this document. They are reliable as of the time of this writing and are presented without warranty of any kind, expressed or implied. Bright Computing, Inc. shall not be liable for technical or editorial errors or omissions which may occur in this document. Bright Computing, Inc. shall not be liable for any damages resulting from the use of this document.

Limitation of Liability and Damages Pertaining to Bright Computing, Inc.

The Bright Cluster Manager product principally consists of free software that is licensed by the Linux authors free of charge. Bright Computing, Inc. shall have no liability nor will Bright Computing, Inc. provide any warranty for the Bright Cluster Manager to the extent that is permitted by law. Unless confirmed in writing, the Linux authors and/or third parties provide the program as is without any warranty, either expressed or implied, including, but not limited to, marketability or suitability for a specific purpose. The user of the Bright Cluster Manager product shall accept the full risk for the quality or performance of the product. Should the product malfunction, the costs for repair, service, or correction will be borne by the user of the Bright Cluster Manager product. No copyright owner or third party who has modified or distributed the program as permitted in this license shall be held liable for damages, including general or specific damages, damages caused by side effects or consequential damages, resulting from the use of the program or the un-usability of the program (including, but not limited to, loss of data, incorrect processing of data, losses that must be borne by you or others, or the inability of the program to work together with any other program), even if a copyright owner or third party had been advised about the possibility of such damages unless such copyright owner or third party has signed a writing to the contrary.

Table of Contents

Preface

Welcome to the *OpenStack Deployment Manual* for Bright Cluster Manager 7.0.

0.1 About This Manual

This manual is aimed at helping cluster administrators install, understand, configure, and manage basic OpenStack capabilities easily using Bright Cluster Manager. The administrator is expected to be reasonably familiar with the *Administrator Manual*.

0.2 About The Manuals In General

Regularly updated versions of the Bright Cluster Manager 7.0 manuals are available on updated clusters by default at `/cm/shared/docs/cm`. The latest updates are always online at `http://support.brightcomputing.com/manuals`.

- The *Installation Manual* describes installation procedures for a basic cluster.
- The *Administrator Manual* describes the general management of the cluster.
- The *User Manual* describes the user environment and how to submit jobs for the end user.
- The *Cloudbursting Manual* describes how to deploy the cloud capabilities of the cluster.
- The *Developer Manual* has useful information for developers who would like to program with Bright Cluster Manager.
- The *OpenStack Deployment Manual* describes how to deploy OpenStack with Bright Cluster Manager.
- The *Hadoop Deployment Manual* describes how to deploy Hadoop with Bright Cluster Manager.
- The *UCS Deployment Manual* describes how to deploy the Cisco UCS server with Bright Cluster Manager.

If the manuals are downloaded and kept in one local directory, then in most pdf viewers, clicking on a cross-reference in one manual that refers to a section in another manual opens and displays that section in the second manual. Navigating back and forth between documents is usually possible with keystrokes or mouse clicks.

For example: `<Alt>-<Backarrow>` in Acrobat Reader, or clicking on the bottom leftmost navigation button of xpdf, both navigate back to the previous document.

The manuals constantly evolve to keep up with the development of the Bright Cluster Manager environment and the addition of new hardware and/or applications. The manuals also regularly incorporate customer feedback. Administrator and user input is greatly valued at Bright Computing. So any comments, suggestions or corrections will be very gratefully accepted at `manuals@brightcomputing.com`.

0.3 Getting Administrator-Level Support

Unless the Bright Cluster Manager reseller offers support, support is provided by Bright Computing over e-mail via `support@brightcomputing.com`. Section 10.2 of the *Administrator Manual* has more details on working with support.

1

Introduction

OpenStack is an open source implementation of cloud services. It is currently (2015) undergoing rapid development, and its roadmap is promising.

A relatively stable implementation of OpenStack, based on the OpenStack Icehouse release (`https://www.openstack.org/software/icehouse/`) is integrated into the Bright Cluster Manager 7.0 for OpenStack edition.

By relatively stable it is meant that OpenStack itself is usable and stable for regular use in common configurations, but not quite production-ready when carrying out some less common configuration changes. In a complex and rapidly-evolving product such as OpenStack, the number of possible unusual configuration changes is vast. As a result, the experience of Bright Computing is that Bright Cluster Manager can sometimes run into OpenStack issues while implementing the less common OpenStack configurations.

As one of the supporting organizations of OpenStack, Bright Computing is committed towards working together with OpenStack developers to help Bright customers resolve any such issue. The end result after resolving the issue means that there is a selection pressure that helps evolve that aspect of OpenStack to become convenient and stable for regular use. This process benefits all participants in the OpenStack software ecosystem.

OpenStack consists of subsystems, developed as software projects[1]. A software project provides capabilities to OpenStack via the implementation of a backend service, and thereby provides an OpenStack service. The OpenStack service can thus be implemented by interchangeable backends, which projects can provide.

For example, the OpenStack Cinder project provides block storage capabilities to OpenStack via the implementation of, for example, NFS or Ceph block storage. The OpenStack's block storage service can therefore be implemented by interchangable backends. As far as the user is concerned the result is the same.

An analogy to OpenStack is operating system packaging, as provided by distributions:

[1]The term projects must not be confused with the term used in OpenStack elsewhere, where projects, or sometimes tenants, are used to refer to a group of users

An operating system distribution consists of subsystems, maintained as packages and their dependencies. Some subsystems provide capabilities to the operating system via the implementation of a backend service. The service can often be implemented by interchangeable backends for the subsystem.

A specific example for an operating system distribution would be the mailserver subsystem that provides mail delivery capabilities to the operating system via the implementation of, for example, Postfix or Sendmail. The mailserver package and dependencies can therefore be implemented by interchangeable backends. As far as the e-mail user is concerned, the end result is the same.

The project that implements the backend can also change, if the external functionality of the project remains the same.

Some of the more common OpenStack projects are listed in the following table:

Service	OpenStack Project	Managed By Bright
Compute	Nova	✓
Object Storage	Swift	depends*
Block Storage	Cinder	✓
Networking	Neutron	✓
Dashboard	Horizon	✓
Identity Service	Keystone	✓
Orchestration	Heat	✓
Telemetry	Ceilometer	×
Database Service	Trove	×
Image Service	Glance	✓

* Bright Cluster Manager does not manage the OpenStack reference implementation for Swift object storage, but does manage a replacement, the API-compatible Ceph RADOS Gateway implementation.

Not all of these projects are integrated, or needed by Bright Cluster Manager for a working OpenStack system. For example, Bright Cluster Manager already has an extensive monitoring system and therefore does not for now implement `Ceilometer`, while `Trove` is ignored for now because it is not yet production-ready.

Projects that are not yet integrated can in principle be added by administrators on top of what is deployed by Bright Cluster Manager, even though this is not currently supported or tested by Bright Computing. Integration of the more popular of such projects, and greater integration in general, is planned in future versions of Bright Cluster Manager.

This manual explains the installation, configuration, and some basic use examples of the OpenStack projects that have so far been integrated with Bright Cluster Manager.

2

OpenStack Installation

To Use Ceph, It Must Be Installed Before Deploying OpenStack
If OpenStack is to access Ceph for storage purposes, for any combination of block storage (Cinder), image storage (Glance), ephemeral storage (Nova), or object storage (RADOS Gateway), then the Ceph components must first be installed with `cm-ceph-setup` (Chapter 3) before starting the OpenStack installation procedure covered here.

Ways Of Installing OpenStack
The version of OpenStack that is integrated with Bright Cluster Manager can be installed in the following two ways:

- Using the GUI-based `Setup Wizard` button from within `cmgui` (section 2.1). This is the recommended installation method.
- Using the text-based `cm-openstack-setup` utility (section 2.2). The utility is a part of the standard `cluster-tools` package.

The priorities that the package manager uses are expected to be at their default settings, in order for the installation to work.

By default, deploying OpenStack installs the following projects: Keystone, Nova, Cinder, Glance, Neutron, Heat and Horizon (the dashboard).

If Ceph is used, then Bright also can also optionally deploy RADOS Gateway to be used as a Swift-API-compatible object storage system. Using RADOS Gateway instead of the reference Swift object storage is regarded in the OpenStack community as good practice, and is indeed the only object storage system that Bright Cluster Manager manages for OpenStack. Alternative backend storage is possible at the same time as object storage, which means, for example, that block and image storage are options that can be used in a cluster at the same time as object storage.

2.1 Installation Of OpenStack From `cmgui`

The `cmgui` OpenStack `Setup Wizard` is the preferred way to install OpenStack. A prerequisite for running it is that the head node should be connected to the distribution repositories.

Some suggestions and background notes These are given here to help the administrator understand what the setup configuration does, and to

help simplify deployment. Looking at these notes after a dry-run with the wizard will probably be helpful.

- A VXLAN (Virtual Extensible LAN) network is similar to a VLAN network in function, but has features that make it more suited to cloud computing.
  - If VXLANs are to be used, then the wizard is able to help create a VXLAN *overlay network* for OpenStack *tenant networks*.

    An OpenStack tenant network is a network used by a group of users allocated to a particular virtual cluster.

    A VXLAN overlay network is a Layer 2 network "overlaid" on top of a Layer 3 network. The VXLAN overlay network is a virtual LAN that runs its frames encapsulated within UDP packets over the regular TCP/IP network infrastructure. It is very similar to VLAN technology, but with some design features that make it more useful for cloud computing needs. One major improvement is the around 16 million VXLANs that can be made to run over the underlying Layer 3 network. This is in contrast to the 4,000 or so VLANs that can be made to run over their underlying Layer 2 network, if the switch port supports that level of simultaneous capability.

    By default, if the VXLAN network and VXLAN network object do not exist, then the wizard helps the administrator create a `vxlanhostnet` network and network object (section 2.1.13). The network is attached to, and the object is associated with, all non-head nodes taking part in the OpenStack deployment. If a `vxlanhostnet` network is pre-created beforehand, then the wizard can guide the administrator to associate a network object with it, and ensure that all the non-head nodes participating in the OpenStack deployment are attached and associated accordingly.
  - The VXLAN network runs over an IP network. It should therefore have its own IP range, and each node on that network should have an IP address. By default, a network range of 10.161.0.0/16 is suggested in the VXLAN configuration screen (section 2.1.13, figure 2.14).
  - The VXLAN network can run over a dedicated physical network, but it can also run over an alias interface on top of an existing internal network interface. The choice is up to the administrator.
  - It is possible to deploy OpenStack without VXLAN overlay networks if user instances are given access to the internal network. Care must then be taken to avoid IP addressing conflicts.
- Changing the hostname of a node after OpenStack has been deployed is a best practice, but does involve some manual steps. So it is recommended to change the hostnames before running the wizard. For example, to set up a network node:
  - a single regular node which is to be the network node of the deployment should be chosen and renamed to, say, `networknode`

  - it should then be restarted
  - the Setup Wizard is then run, from the head node. When the wizard reaches the network node selection screen (section 2.1.5, figure 2.6) networknode can be selected as the network node.

- When allowing for Floating IPs and/or enabling outbound connectivity from the virtual machines (VMs) to the external network via the network node, the network node can be pre-configured manually according to how it is connected to the internal and external networks. Otherwise, if the node is not pre-configured manually, the wizard then carries out a basic configuration on the network node that
  - configures one physical interface of the network node to be connected to the internal network, so that the network node can route packets for nodes on the internal network.
  - configures the other physical interface of the network node to be connected to the external network so that the network node can route packets from external nodes.

The wizard asks the user several questions on the details of how OpenStack is to be deployed. From the answers, it generates an XML document with the intended configuration. Then, in the back-end, largely hidden from the user, it runs the text-based cm-openstack-setup script with this configuration on the active head node. In other words, the wizard can be regarded as a GUI front end to the cm-openstack-setup utility.

The practicalities of executing the wizard: The explanations given by the wizard during its execution steps are intended to be verbose enough so that the administrator can follow what is happening.

The wizard is accessed via the OpenStack resource in the left pane of cmgui (figure 2.1). Launching the wizard is only allowed if the Bright Cluster Manager license (Chapter 4 of the *Installation Manual*) entitles the license holder to use OpenStack.

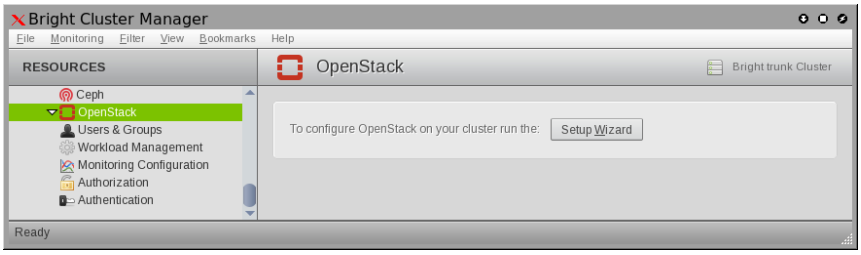

Figure 2.1: The Setup Wizard Button In cmgui's OpenStack Resource

The wizard runs through the screens in sections 2.1.1-2.1.21, described next.

2.1.1 OpenStack Setup Wizard Overview

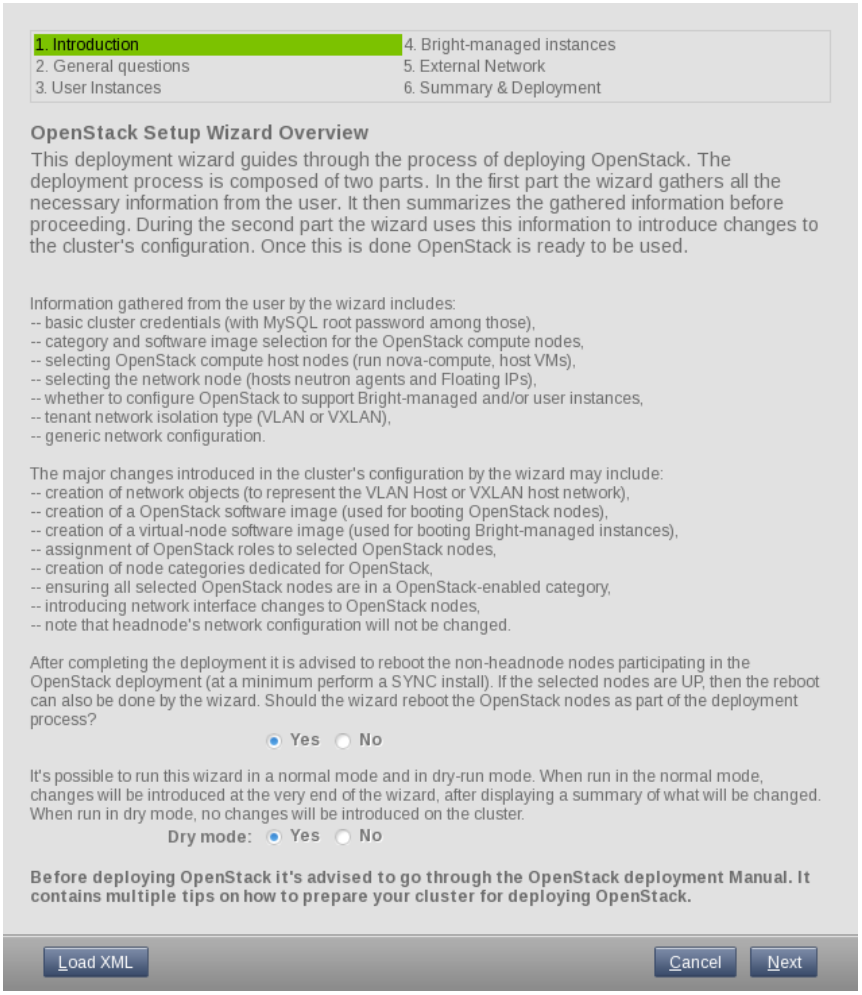

Figure 2.2: OpenStack Setup Wizard Overview Screen

The overview screen (figure 2.2) explains what information is gathered, and what the wizard intends to do with the information.

The screen also asks for input on the following:

- Should the regular nodes that become part of the OpenStack cluster be rebooted? A reboot installs a new image onto the node, and is recommended if interface objects need to be created in CMDaemon for OpenStack use. Creating the objects is typically the case during the first run ever for a particular configuration. Subsequent runs of the wizard do not normally create new interfaces, and for small changes do not normally require a node reboot. If in doubt, the reboot option can be set to enabled. Reboot is enabled as the default.
- Should a dry-run be done? In a dry-run, the wizard pretends to carry out the installation, but the changes are not really implemented. This is useful for getting familiar with options and their possible consequences. A dry run is enabled as the default.

2.1.2 MySQL Credentials & OpenStack admin User

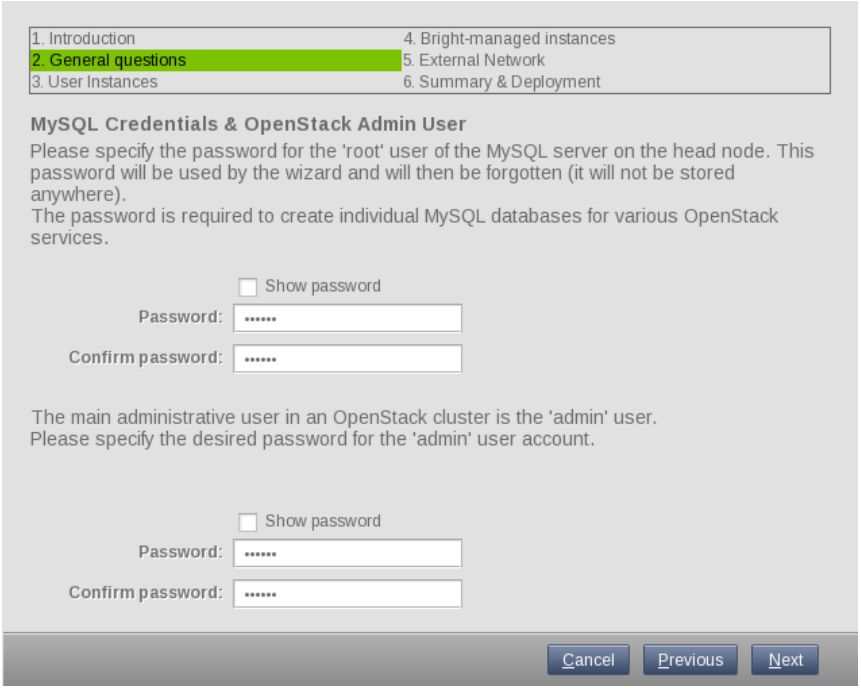

Figure 2.3: MySQL Credentials & OpenStack admin User Screen

The MySQL and OpenStack credentials screen (figure 2.3) allows the administrator to set passwords for the MySQL root user and the OpenStack admin user.

2.1.3 OpenStack Category Configuration

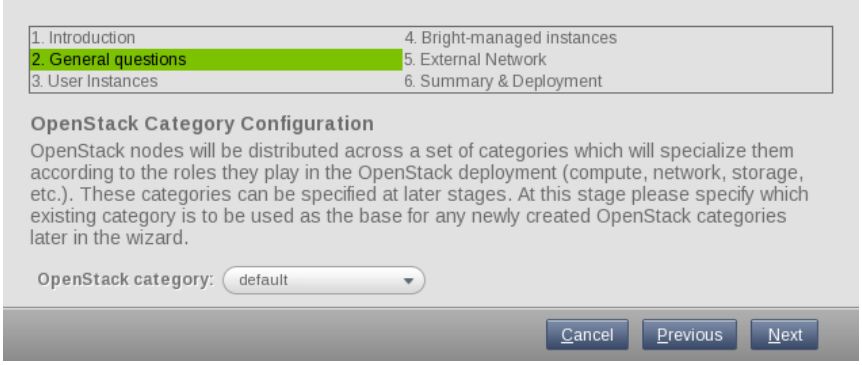

Figure 2.4: OpenStack Category Configuration Screen

The category configuration screen (figure 2.4) sets the node category that will be used as the template for several OpenStack categories that are going to be created by the wizard.

2.1.4 OpenStack Compute Hosts

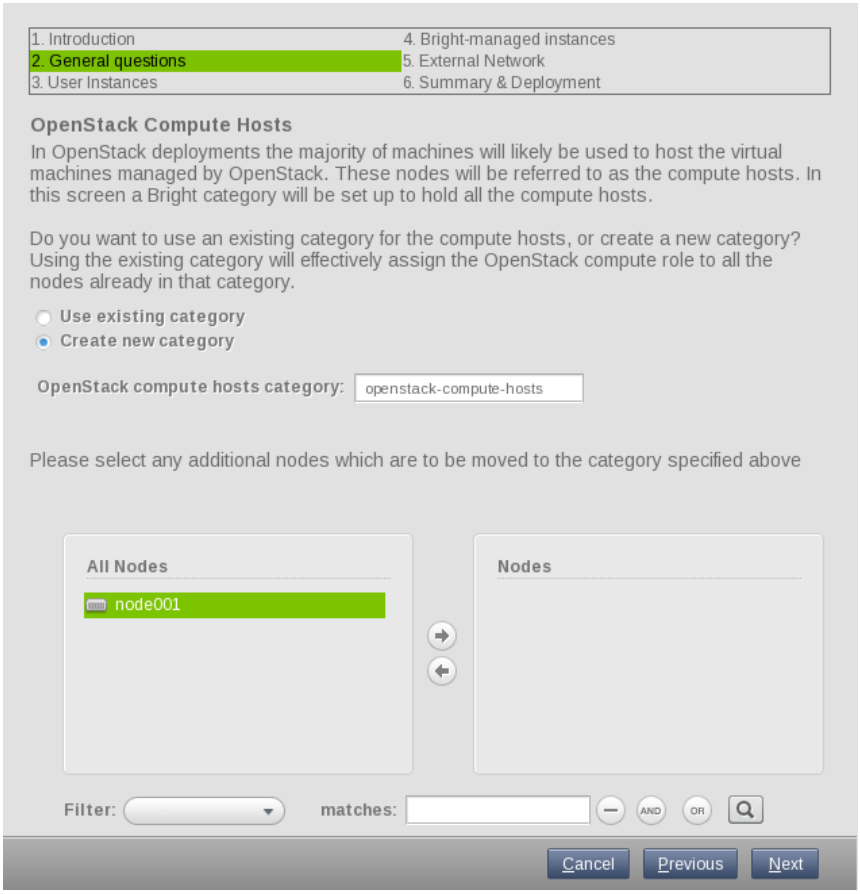

Figure 2.5: OpenStack Compute Hosts Screen

The compute hosts configuration screen (figure 2.5) allows the administrator to take nodes which are still available and put them into a category that will have a compute role.

The category can be set either to be an existing category, or a new category can be created. If an existing category is used, then `default` can be chosen. If Ceph has been integrated with Bright Cluster Manager, then the `ceph` category is another available option.

Creating a new category is recommended, and is the default option. The suggested default category name is `openstack-compute-hosts`.

2.1.5 OpenStack Network Node

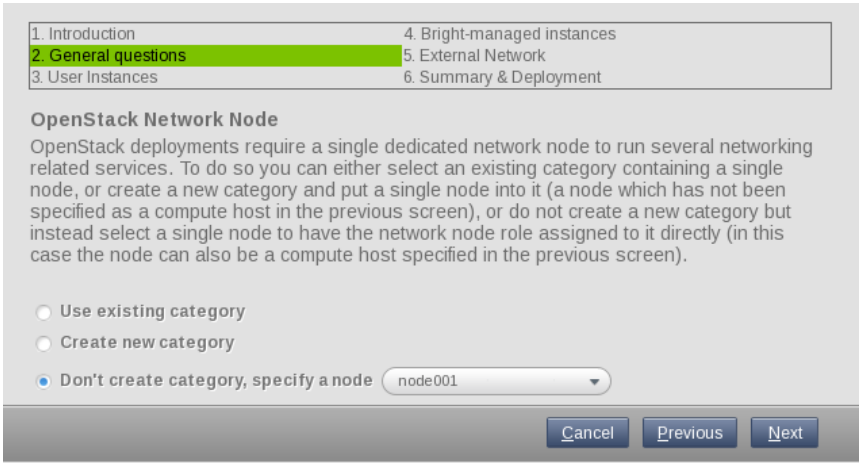

Figure 2.6: OpenStack Network Node Screen

The network node screen (figure 2.6) makes a node the network node, or makes a category of nodes the network nodes. A network node is a dedicated node that handles OpenStack networking services.

If a category is used to set up network nodes, then either an existing category can be used, or a new category can be created. If a new category is to be created, then `openstack-network-hosts` is its suggested name.

The option to specify a node as a network node is most convenient in the typical case when all of the non-head nodes have been set as belonging to the compute node category in the preceding screen (figure 2.5). Indeed, in the case that all non-head nodes have been set to be in the compute node category, the category options displayed in figure 2.6 are then not displayed, leaving only the option to specify a particular node.

A network node inherits many of the OpenStack-related compute node settings, but will have some exceptions to the properties of a compute node. Many of the exceptions are taken care of by assigning the `openstacknetwork` role to any network nodes or network node categories, as is done in this screen.

2.1.6 Ceph Configuration

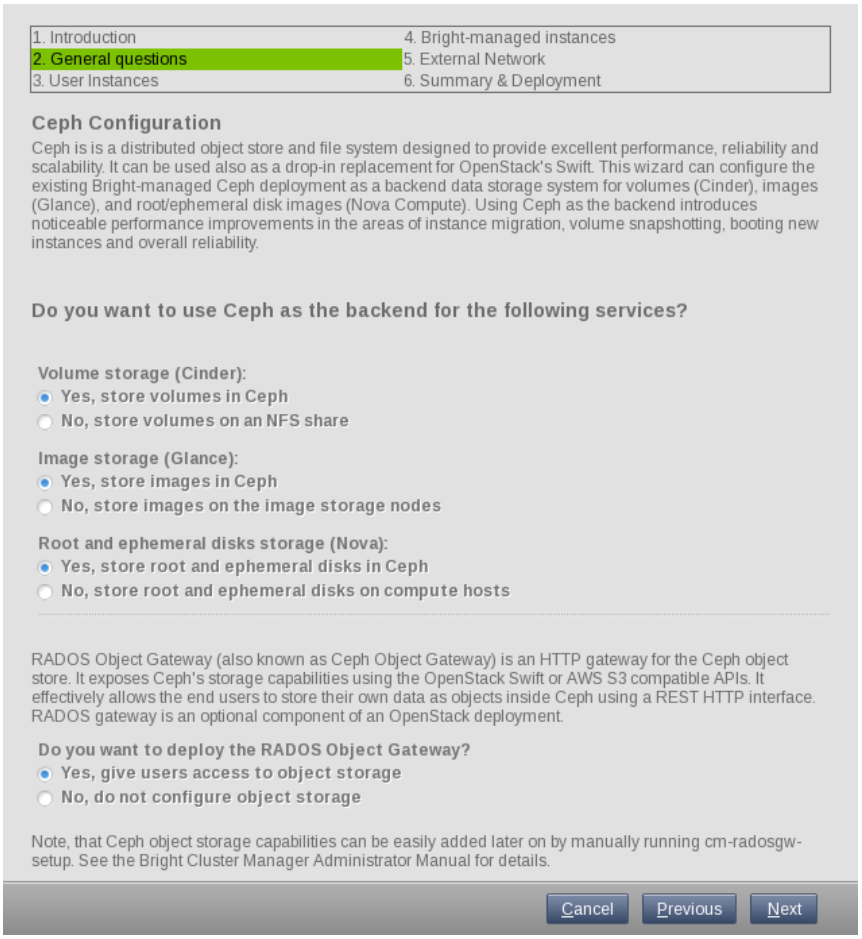

Figure 2.7: Ceph Configuration Screen

If Ceph has been configured with Bright Cluster Manager before the wizard is run, then the Ceph configuration screen (figure 2.7) is displayed. Choosing any of the Ceph options requires that Ceph be pre-installed. This is normally done with the `cm-ceph-setup` script (section 3.2).

Ceph is an object-based distributed parallel filesystem with self-managing and self-healing features. Object-based means it handles each item natively as an object, along with meta-data for that item. Ceph is a drop-in replacement for Swift storage, which is the reference OpenStack object storage software project.

The administrator can decide on:

- Using Ceph for volume storage, instead of on NFS shares. This is then instead of using the OpenStack Cinder reference project for the implementation of volume storage.[1]

[1] An advantage of Ceph, and one of the reasons for its popularity in OpenStack implementations, is that it supports volume snapshots in OpenStack. Snapshotting is the ability to take a copy of a chosen storage, and is normally taken to mean using copy-on-write (COW) technology. More generally, assuming enough storage is available, non-COW technology can also be used to make a snapshot, despite its relative wastefulness.

In contrast to Ceph, the reference Cinder implementation displays an error if attempting to use the snapshot feature, due to its NFS driver.

The administrator should understand that root or ephemeral storage are concepts that

- Using Ceph for image storage, instead of on image storage nodes. This is instead of using the OpenStack Glance reference project for the implementation of virtual machine image storage.
- Using Ceph for root and ephemeral disks storage, instead of on the filesystem of the compute hosts. This is instead of using the OpenStack Nova reference project implementation for the implementation for disk filesystem storage.[2]

Object Storage (Swift Project)

In addition, the screen also asks if the RADOS Object Gateway should be deployed. RADOS Object Gateway is a high-level interface that allows Ceph to be accessed in the same way as Amazon S3 or OpenStack Swift, using their HTTP APIs. The higher level RADOS Object Gateway should not be confused with RADOS, a lower level interface that the gateway relies on.

If RADOS Object Gateway is enabled, it means that users have access to object storage using Ceph, instead of using the OpenStack Swift reference project implementation for object storage.

The RADOS Object Gateway can be added separately from this OpenStack installation wizard, after it has completed, by using the `cm-radosgw-setup` utility (section 3.4).

---

are valid only inside the Nova project, and are completely separate storages from Cinder. These have nothing to do with the Cinder reference implementation, so that carrying out a snapshot for these storages does not display such an error.

[2] Compute hosts need to have the root and ephemeral device data belonging to their hosted virtual machines stored somewhere. The default directory location for these images and related data is under `/var/lib/nova`. If Ceph is not enabled, then a local mount point of every compute host is used for data storage by default. If Ceph is enabled, then Ceph storage is used instead according to its mount configuration.

In either case, compute hosts that provide storage for virtual machines, by default store the virtual machine images under the `/var` partition. However, the default partition size for `/var` is usually only large enough for basic testing purposes. For production use, with or without Ceph, the administrator is advised to either adjust the size of the existing partition, using the `disksetup` command (section 3.9.3 of the *Administrator Manual*), or to use sufficient storage mounted from elsewhere.

To change the default paths used by OpenStack images, the following two path variables in the Nova configuration file `/etc/nova/nova.conf` should be changed:

1. `state_path=/var/lib/nova`
2. `lock_path=/var/lib/nova/tmp`

If `$state_path` has been hardcoded by the administrator elsewhere in the file, the location defined there should also be changed accordingly.

2.1.7 OpenStack Internal Network Selection

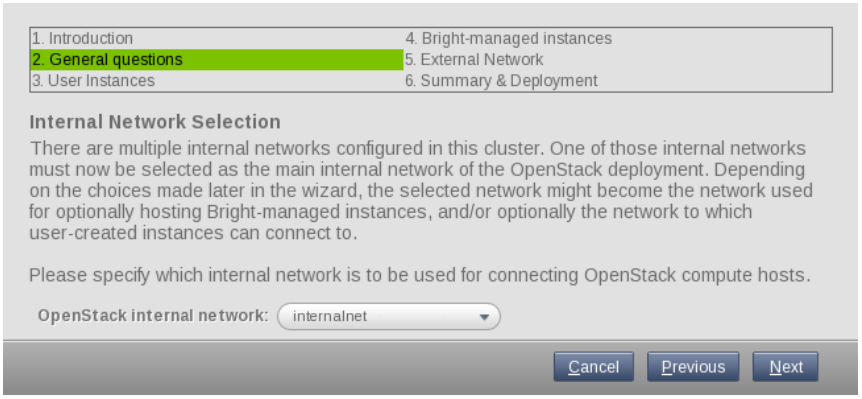

Figure 2.8: OpenStack Internal Network Selection Screen

The OpenStack internal network selection screen (figure 2.8) decides the network to be used on the nodes that run OpenStack.

2.1.8 OpenStack Software Image Selection

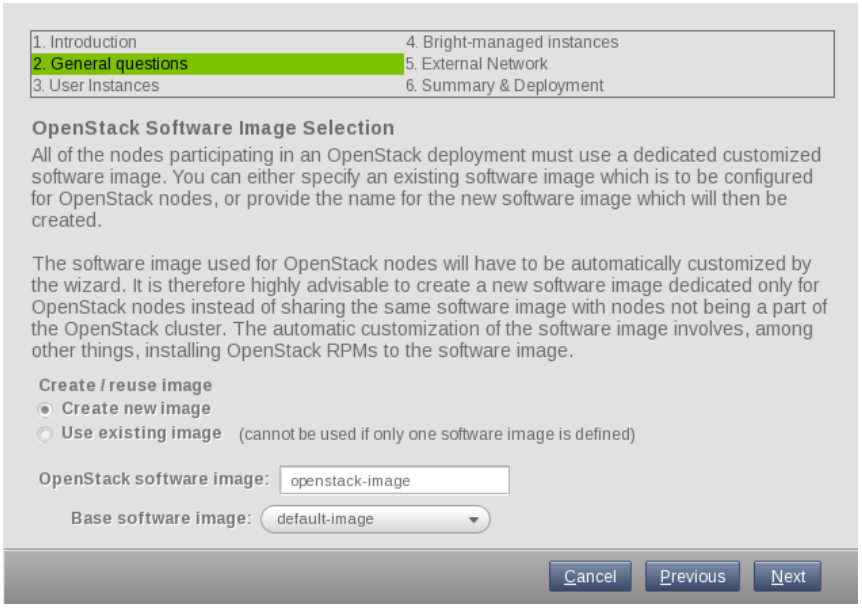

Figure 2.9: OpenStack Software Image Selection Screen

The OpenStack software image selection screen (figure 2.9) decides the software image name to be used on the nodes that run OpenStack.

An existing image name can be used if there is more than one available image name.

Creating a new image, with the default name of `openstack-image`, is recommended. This image, `openstack-image`, is to be the base OpenStack image, and it is cloned and modified from an original, pre-OpenStack-deployment image.

By default, the name `openstack-image` is chosen. This is recommended because the image that is to be used by the OpenStack nodes has many modifications from the `default` image, and it is useful to keep the

default image around for comparison purposes.

2.1.9 OpenStack User Settings: Default Tenant Creation Choice

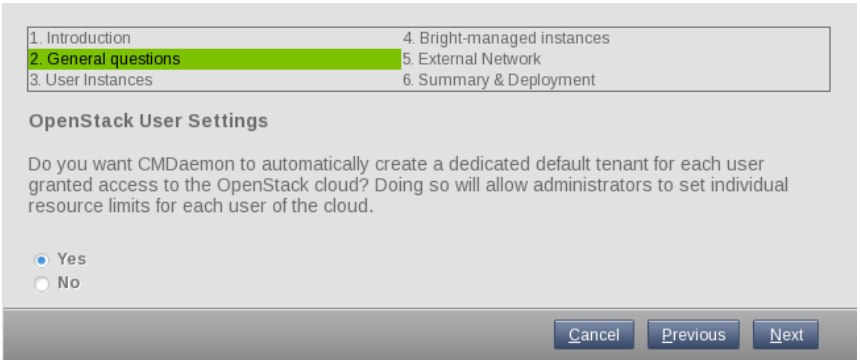

Figure 2.10: User Settings Screen

The User Settings screen (figure 2.10) sets whether OpenStack is to be configured so that with Bright Cluster Manager each user that is added to OpenStack gets a new tenant, with default quotas (resource limits) set for the tenant.

If set, this allows administrators to manage tenant quotas for each user. If not set, then each user will have to be assigned to a atenant before the user can log into OpenStack.

2.1.10 User Instances

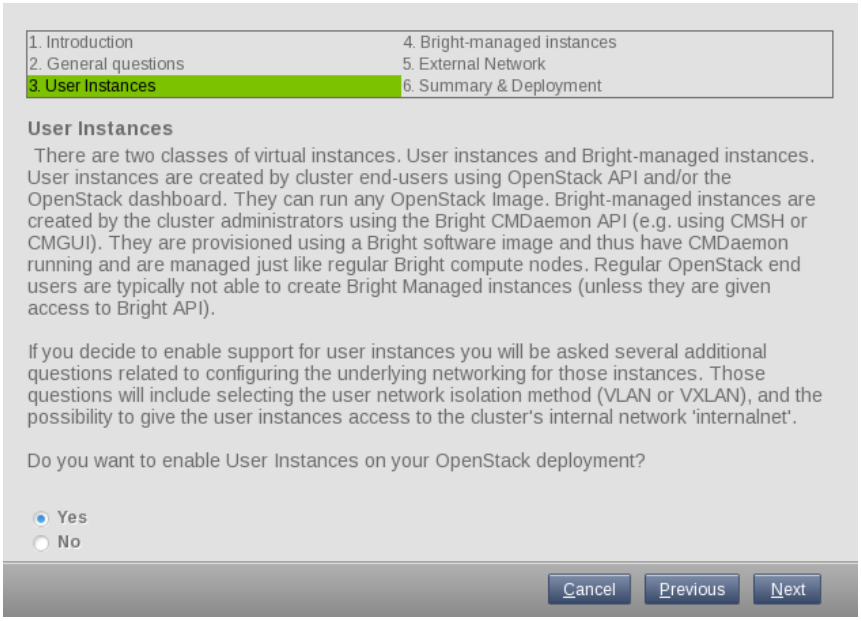

Figure 2.11: User Instances Screen

The User Instances screen (figure 2.11) allows the administrator to allow the Bright-end-user to create user instances.

The following overview may help get a perspective on this part of the wizard configuration procedure:

The main function of OpenStack is to manage virtual machines. From

the administrator's point of view there are two classes of virtual machines, or instances:

- User instances, configured in this section
- Bright-managed instances, configured with the help of figure 2.16

The wizard allows OpenStack to be configured to support both types of instances, or only one of them.

Deploying OpenStack without configuring it for either type of instance is also possible, but such an OpenStack cluster is very limited in its functionality and typically has to be customized further by the administrator.

Both types of instances are virtual machines hosted within a hypervisor managed by the OpenStack compute project, Nova. The main difference between these two types instances include the following:

- **User instances** are typically created and managed by the end-users of the deployment, either directly via the OpenStack API, or via OpenStack Dashboard, outside of direct influence from Bright Cluster Manager. User instances are provisioned using any OpenStack-compatible software image provided by the user, and thus have no CMDaemon running on them. User instances are attached to user-created virtual networks. Optionally, they can be allowed to connect directly to the cluster's internal network (section 2.1.11). The number of user instances that can be run is not restricted in any way by the Bright Cluster Manager license.
- **Bright-managed instances**, sometimes also called `virtual nodes`, are typically created and managed by the cluster/cloud administrators using CMDaemon, via `cmsh` or `cmgui` or `pythoncm`. They are provisioned using a Bright software image, and therefore have CMDaemon running on them. Because of CMDaemon, the administrator can manage Bright-managed instances just like regular nodes under Bright Cluster Manager. Bright-managed instances are always connected to the cluster's internal network, but can also be attached to user-created networks.

To allow user instances to be created, the `Yes` radio-button should be ticked in this screen. This will lead to the wizard asking about user-instance network isolation (VLAN/VXLAN).

Whether or not Bright-managed instances are to be allowed is set later in the Bright-managed instances screen (figure 2.16).

2.1.11 User Instance Isolation from Internal Cluster Network

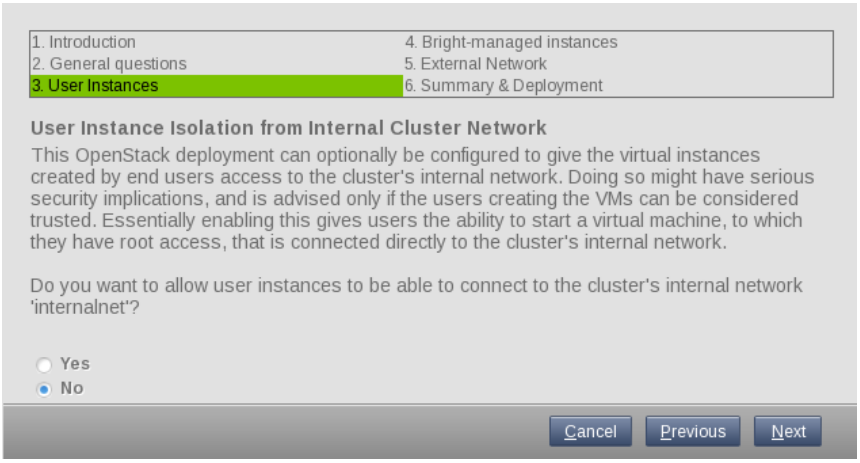

Figure 2.12: User Instance Isolation from Internal Cluster Network Screen

If the creation of user instances has been enabled (figure 2.11), the user instance internal cluster network isolation screen (figure 2.12) allows the administrator to allow OpenStack-end users to create user instances which have direct network connectivity to the cluster's internal network.

If the "network isolation" restriction is removed, so that there is "network promiscuity" between user instances on the internal network, then this allows user instances (figure 2.11) to connect to other user instances on the internal network of the cluster. End users can then manage other user instances. Allowing this should only be acceptable if all users that can create instances are trusted.

2.1.12 Network Isolation

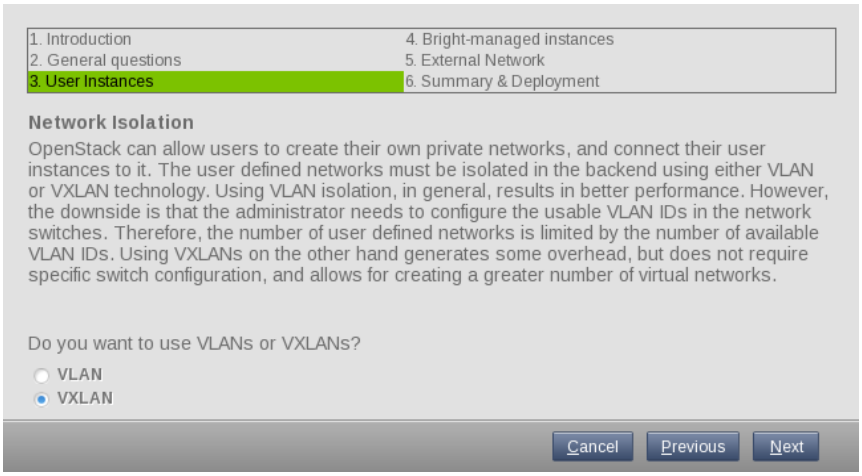

Figure 2.13: Network Isolation Screen

The network isolation screen (figure 2.13) allows the administrator to set the virtual LAN technology that user instances can use for their user-defined private networks. Using virtual LANs isolates the IP networks used by the instances from each other. This means that the instances attached to one private network will always avoid network conflicts with

the instances of another other network, even if using the same IP address ranges.

Bright Cluster Manager supports two virtual LAN technologies:

- `VLAN`: VLAN technology tags Ethernet frames as belonging to a particular VLAN. However it requires manual configuration of the VLAN IDs in the switches, and also the number of IDs available is limited to 4094.
- `VXLAN`: VXLAN technology has more overhead per packet than VLANs, because it adds a larger ID tag, and also because it encapsulates layer 2 frames within layer 3 IP packets. However, unlike with VLANs, configuration of the VXLAN IDs happens automatically, and the number of IDs available is about 16 million.

By default, VXLAN technology is chosen. This is because for VXLAN, the number of network IDs available, along with the automatic configuration of these IDs, means that the cluster can scale further and more easily than for VLAN.

Selecting a network isolation type is mandatory, unless user instances are configured to allow access to the internal network of the cluster by the administrator (figure 2.11).

Presently, only one type of network isolation is supported at a time.

2.1.13 VXLAN Configuration

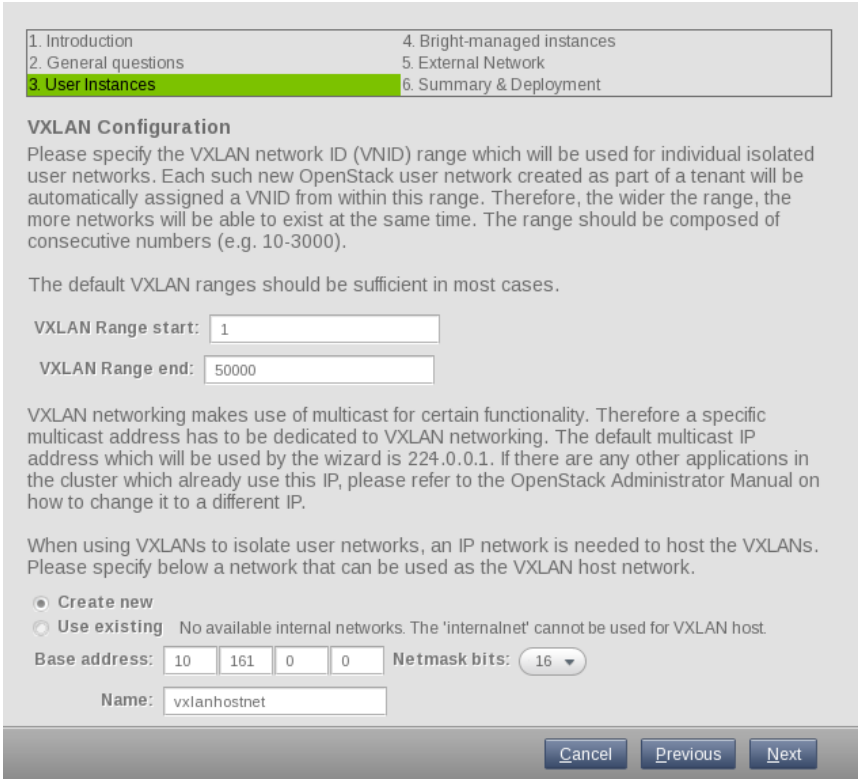

Figure 2.14: VXLAN Configuration Screen

The VXLAN screen (figure 2.14) shows configuration options for the VXLAN network if VXLAN has been chosen as the network isolation technology

in the preceding screen. If the network isolation technology chosen was VLAN, then a closely similar screen is shown instead.

For the VXLAN screen, the following options are suggested, with overrideable defaults as listed:

- VXLAN Range start: default: 1
- VXLAN Range end: default: 50000

The VXLAN range defines the number of user IP networks that can exist at the same time. While the range can be set to be 16 million, it is best to keep it to a more reasonable size, such as 50,000, since a larger range slows down Neutron significantly.

An IP network is needed to host the VXLANs and allow the tunneling of traffic between VXLAN endpoints. This requires

- either choosing an existing network that has already been configured in Bright Cluster Manager, but not internalnet
- or it requires specifying the following, in order to create the network:
  - Base address: default: 10.161.0.0
  - Netmask bits: default: 16
  - A new network Name: default: vxlanhostnet

VXLAN networking uses a multicast address to handle broadcast traffic in a virtual network. The default multicast IP address that is set, 224.0.0.1, is unlikely to be used by another application. However, if there is a conflict, then the address can be changed using the CMDaemon OpenStackVXLANGroup directive (Appendix C, page 527 of the *Administrator Manual*).

2.1.14 Dedicated Physical Networks

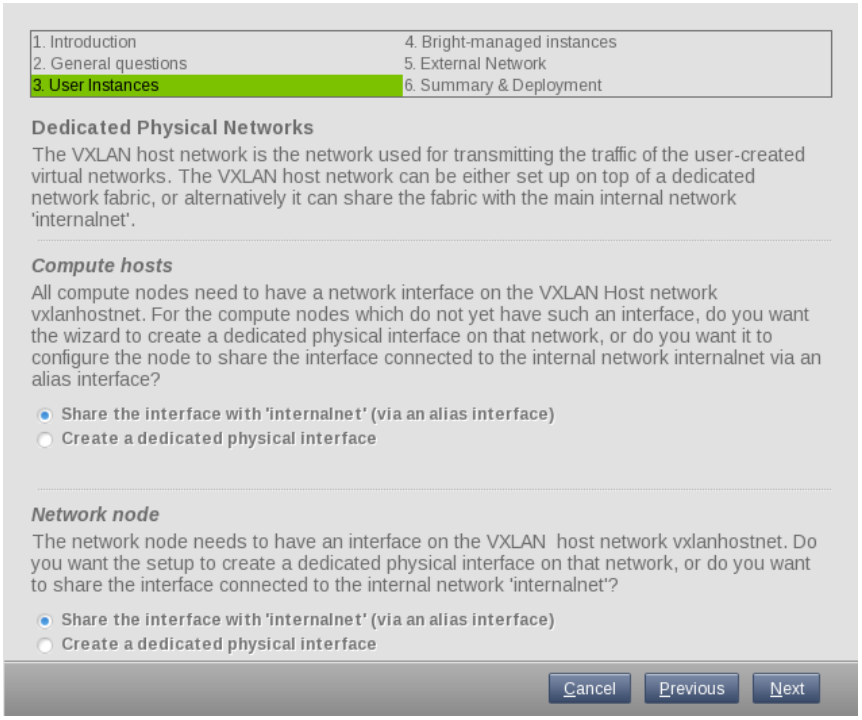

Figure 2.15: Dedicated Physical Networks Screen

The dedicated physical networks screen (figure 2.15): allows the following to be configured:

- the compute nodes that host the OpenStack instances
- the network nodes that provides networking services for OpenStack.

For each of these types of node (compute node or network node), the interface can:

- either be shared with `internalnet` using an alias interface
- or created separately on a physical network interface. The interface must then be given a name. The name can be arbitrary.

2.1.15 Bright-Managed Instances

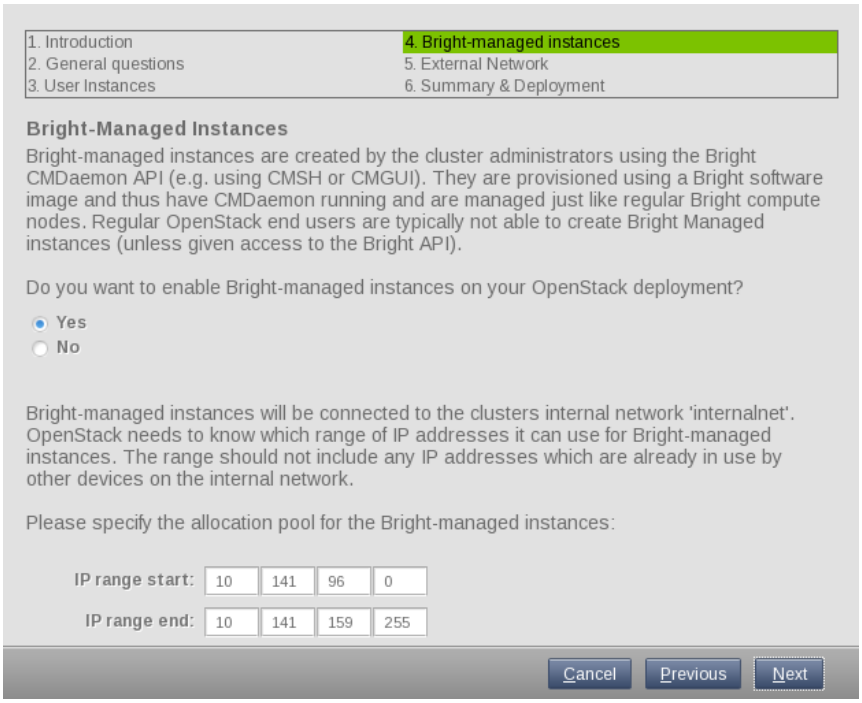

Figure 2.16: Bright-Managed Instances Screen

The Bright-managed instances screen (figure 2.16) allows administrators to enable Bright-managed instances. These instances are also known as virtual nodes. Administrators can then run OpenStack instances using Bright Cluster Manager.

End-users are allowed to run OpenStack instances managed by OpenStack only if explicit permission has been given. This permission is the default that is set earlier on in the user instances screen (figure 2.11).

If Bright-managed instances are enabled, then an IP allocation scheme must be set. The values used to define the pool are:

- `IP range start`: By default, this is: `10.141.96.0`
- `IP range end`: By default this is: `10.141.159.255`

The screens shown in figures 2.17 to 2.21 are displayed next if Bright-managed instances are enabled.

2.1.16 Virtual Node Configuration

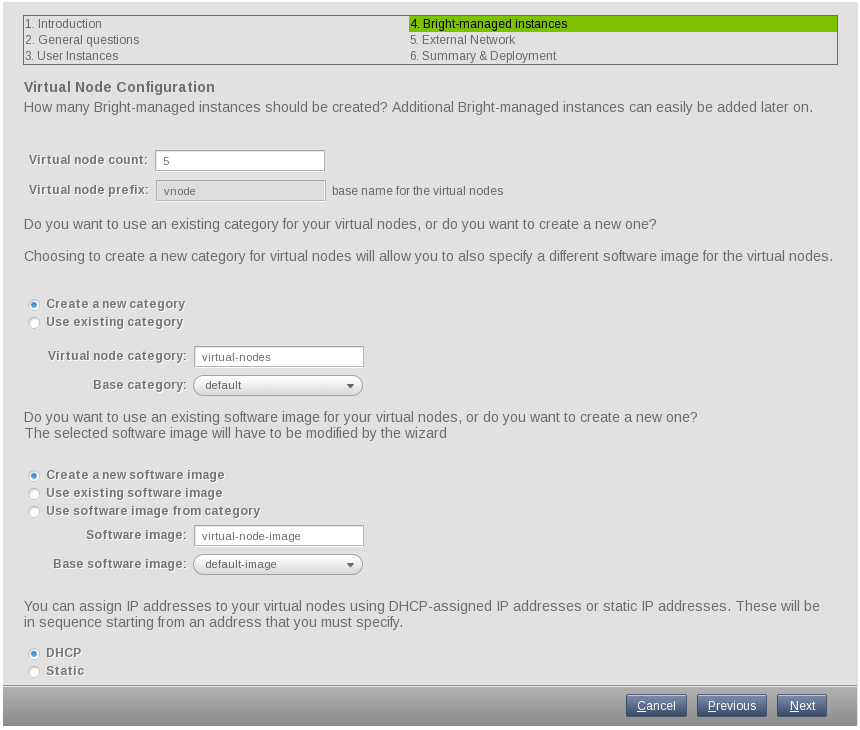

Figure 2.17: Virtual Node Configuration Screen

The virtual node configuration screen (figure 2.17) allows the administrator to set the number, category, and image for the virtual nodes. The suggestions presented in this screen can be deployed in a test cluster.

- `Virtual node count`: 5
- `Virtual node prefix`: vnode

A `Virtual node category` can be set for virtual nodes further down in the screen.

During a new deployment, virtual nodes can be placed in categories, either by creating a new category, or by using an existing category:

- If the `Create a new category` radio button is selected (recommended), then:
  - The `Virtual node category` is given a default value of `virtual-nodes`. This is a sensible setting for a new deployment.
  - The `Base category` can be selected. This is the category from which the new virtual node category is derived. Category settings are copied over from the base category to the virtual node category. The only category choice for the `Base category` in a newly-installed cluster is `default`. Some changes are then made to the category settings in order to make virtual nodes in that category run as virtual instances.

One of the changes that needs to be made to the category settings for a virtual node is that a software image must be set. The following options are offered:

- Create a new software image: This option is recommended for a new installation. Choosing this option presents the following suboptions:
  - The Software image is given a default value of virtual-node-image. This is a sensible setting for a new deployment.
  - The Base software image can be selected. This is the software image from which the new virtual node software image is derived. In a newly-installed cluster, the only base software image choice is default-image.
- Use existing software image: An existing software image can be set. The only value for the existing image in a newly-configured cluster is default-image.
- Use software image from category: The software image that the virtual node category inherits from its base category is set as the software image.

Setting the software image means that the wizard will copy over the properties of the associated base image to the new software image, and configure the new software image with the required virtualization modules. The instance that uses this category then uses a modified image with virtualization modules that enable it to run on a virtual node.

- If the Use existing category button is selected, then:
  - The Virtual node category can be selected from the existing categories. In a newly-installed cluster, the only possible value is default

Setting the category means that the wizard will copy over the properties of the existing category to the new virtual category, and configure the new software image with the required virtualization modules. The instance that uses the configured image is then able to run it on a virtual node.

The virtual nodes can be configured to be assigned one of the following types of addresses:

- DHCP
- Static

The addresses in either case go in a sequence, and begin with an address that the administrator sets in the wizard.

2.1.17 Inbound External Traffic

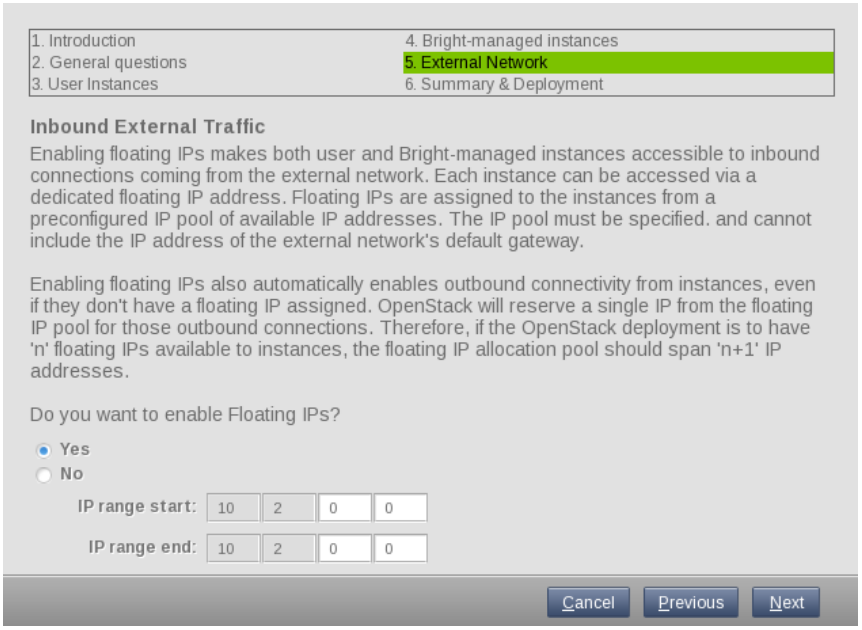

Figure 2.18: Inbound External Traffic Screen

All OpenStack-hosted virtual machines are typically attached to one or more virtual networks. However, unless they also connected to the internal network of the cluster, there is no simple way to connect to them from outside their virtual network. To solve this, *Floating IPs* have been introduced by OpenStack. Floating IPs are a range of IP addresses that the administrator specifies on the external network, and they are made available to the users (tenants) for assignment to their user instances. The number of Floating IPs available to users is limited by the Floating IP quotas set for the tenants.

Administrators can also assign Floating IP addresses to Bright-managed instances. Currently, however, this has to be done via the OpenStack API or Dashboard.

The inbound external traffic screen (figure 2.18) reserves a range of IP addresses within the used external network. In this example it happens to fall within the 10.2.0.0/16 network range.

If specifying a Floating IP range, a single IP address from this range is always reserved by OpenStack for outbound traffic. This is implemented via sNAT. The address is reserved for instances which have not been assigned a Floating IP. Therefore, the IP address range specified in this screen normally expects a minimum of two IP addresses—one for reserved for outbound traffic, and one Floating IP.

If the administrator would like to allow OpenStack instances to have outbound connectivity, but at the same time not have floating IPs, then this can be done by:

- not configuring Floating IPs by selecting the `No` option, and
- in the next screen, figure 2.19, specifying a single IP address.

Alternatively, a user instance can be configured to access the external network via the head node, however this is a bit more complicated to set

up.

Since outgoing connections use one IP address for all instances, the remaining number of IP addresses is what is then available for Floating IPs. The possible connection options are therefore as indicated by the following table:

Floating IP Address	Outbound Allowed?	Inbound Allowed?
Not enabled	no	no
One set, disabled	yes	no
Two or more enabled	yes	yes

2.1.18 Allow Outbound Traffic

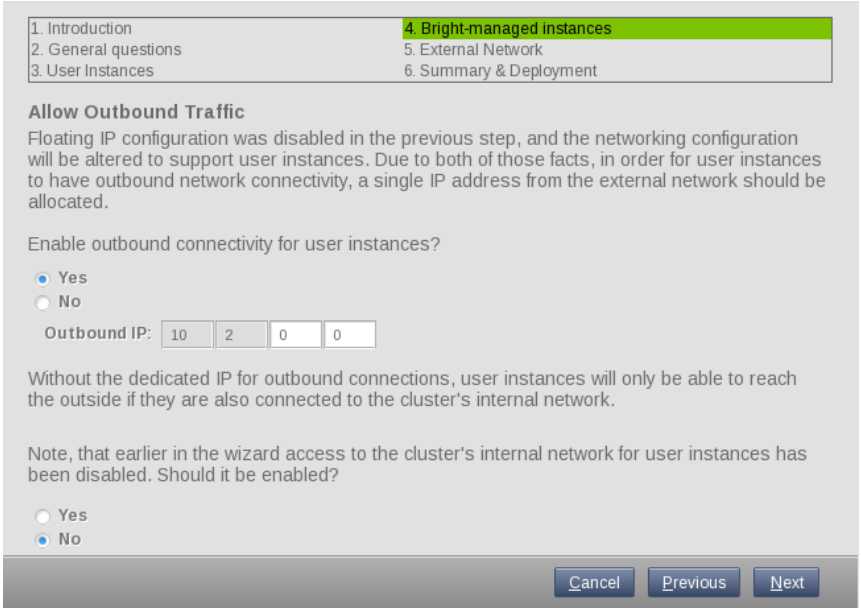

Figure 2.19: Allow Outbound Traffic Screen

The `Allow Outbound Traffic` screen appears if in the previous screen, figure 2.18, Floating IPs have not been configured, and if the `No` option has been selected, and if there are user instances being configured.

The `Allow Outbound Traffic` screen does not appear if the cluster has only Bright-managed instances, and no user-managed instances. This is because Bright-managed instances can simply route their traffic through the head node, without needing the network node to be adjusted with the configuration option of this screen.

If the `Allow Outbound Traffic` screen appears, then specifying a single IP address for the `Outbound IP` value in the current screen, figure 2.19, sets up the configuration to allow outbound connections only.

A decision was made earlier about allowing user instances to access the internal network of the cluster (section 2.1.11). In the dialog of figure 2.19, if user instances are not enabled, then the administrator is offered the option once more to allow access to the internal network of the cluster by user instances.

2.1.19 External Network Interface for Network Node

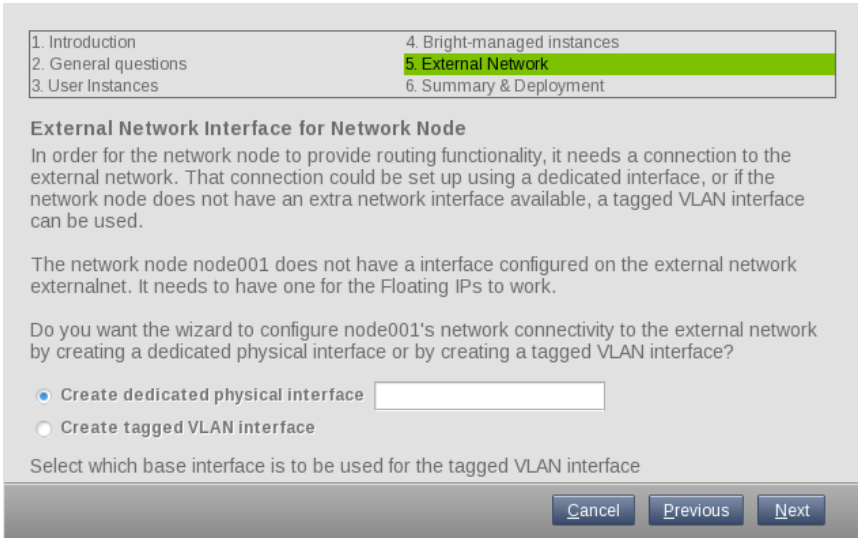

Figure 2.20: External Network Interface for Network Node Screen

The external network interface for network node screen (figure 2.20) allows the administrator to configure either a dedicated physical interface or a tagged VLAN interface for the network node. The interface is to the external network and is used to provide routing functions for OpenStack.

The network node must have a connectivity with the external network when Floating IPs or/and outbound traffic for instances is being configured.

If the node already has a connection to the external network configured in Bright, the wizard will skip this step.

The options are:

- `Create dedicated physical interface`: If this option is chosen, then a dedicated physical interface is used for connection from the network node to the external network.
  - `Interface name`: The name of the physical node should be set. For example: `eth0`.
- `Create tagged VLAN interface`: If this option is chosen, then a tagged VLAN interface is used for the connection from the network node to the external network.
  - `Base interface`: The base interface is selected. Typically the interface selected is `BOOTIF`
  - `Tagged VLAN ID`: The VLAN ID for the interface is set.

2.1.20 VNC Proxy Hostname

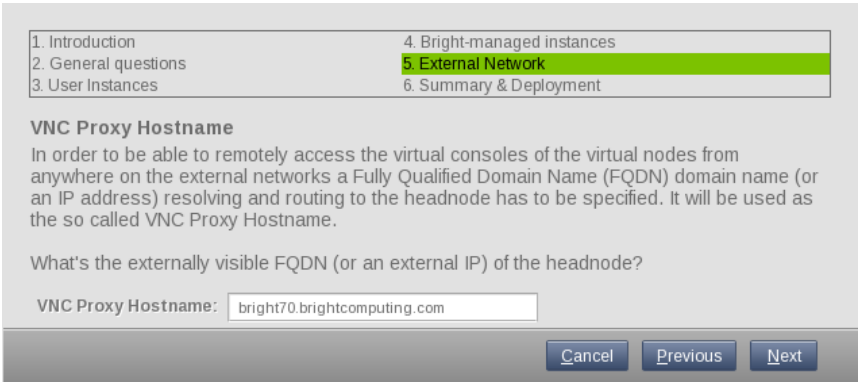

Figure 2.21: VNC Proxy Hostname Screen

The VNC proxy hostname screen (figure 2.21) sets the FQDN or external IP address of the head node of the cluster, as seen by a user that would like to access the consoles of the virtual nodes from the external network.

For a cluster with a hostname `bright70`, and resolving within the `brightcomputing.com` network domain, the value of `VNC Proxy Hostname` would be `bright70.brightcomputing.com`.

2.1.21 Summary

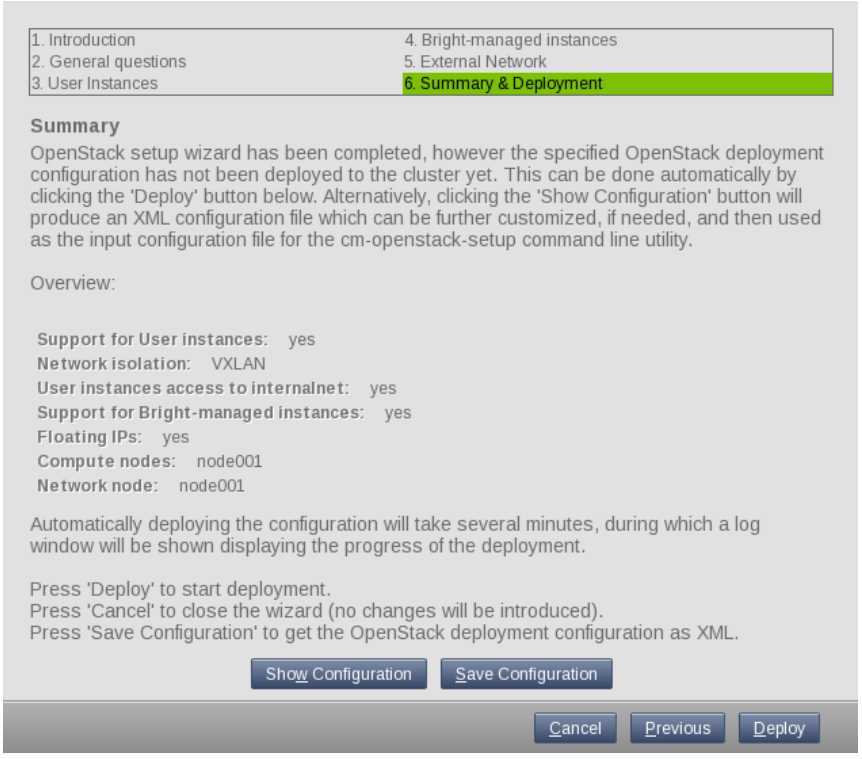

Figure 2.22: Summary Screen

Viewing And Saving The Configuration

The summary screen (figure 2.22) gives a summary of the configuration. The configuration can be changed in `cmgui` if the administrator goes back

through the screens to adjust settings.

The full configuration is kept in an XML file, which can be viewed by clicking on the `Show Configuration` button. The resulting read-only view is shown in figure 2.23.

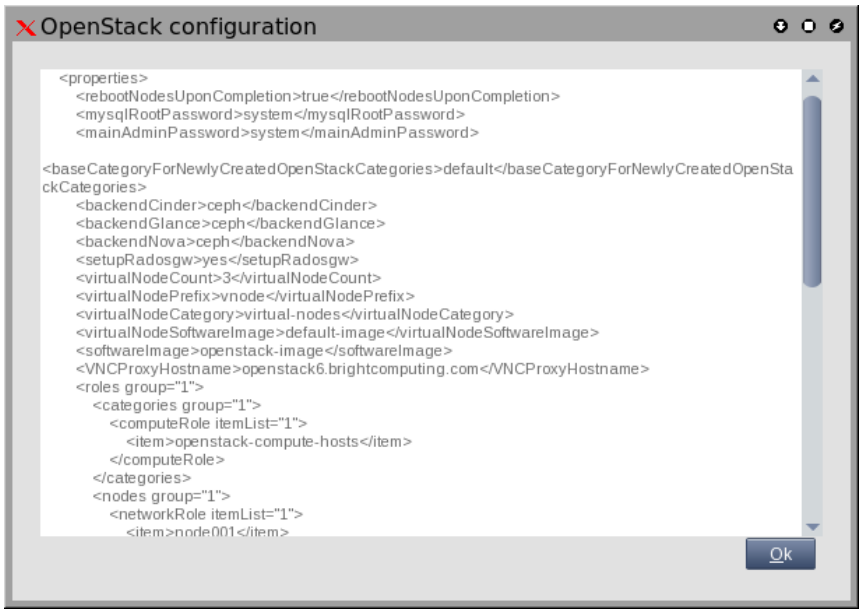

Figure 2.23: OpenStack Configuration Screen

The configuration can be saved with the `Save Configuration` option of figure 2.22.

After exiting the wizard, the XML file can be directly modified if needed in a separate text-based editor.

Using A Saved Configuration And Deploying The Configuration

Using a saved XML file is possible.

- The XML file can be used as the configuration starting point for the text-based `cm-openstack-setup` utility (section 2.2), if run as:

  ```
  [root@bright70~]# cm-openstack-setup -c <XML file>
  ```

- Alternatively, the XML file can be deployed as the configuration by launching the `cmgui` wizard, and then clicking on the `Load XML` button of first screen (figure 2.2). After loading the configuration, a `Deploy` button appears.

Clicking the `Deploy` button that appears in figure 2.2 after loading the XML file, or clicking the `Deploy` button of figure 2.22, sets up OpenStack in the background. The direct background progress is hidden from the administrator, and relies on the text-based `cm-openstack-setup` script (section 2.2). Some log excerpts from the script are displayed within a `Deployment Progress` window (figure 2.24).

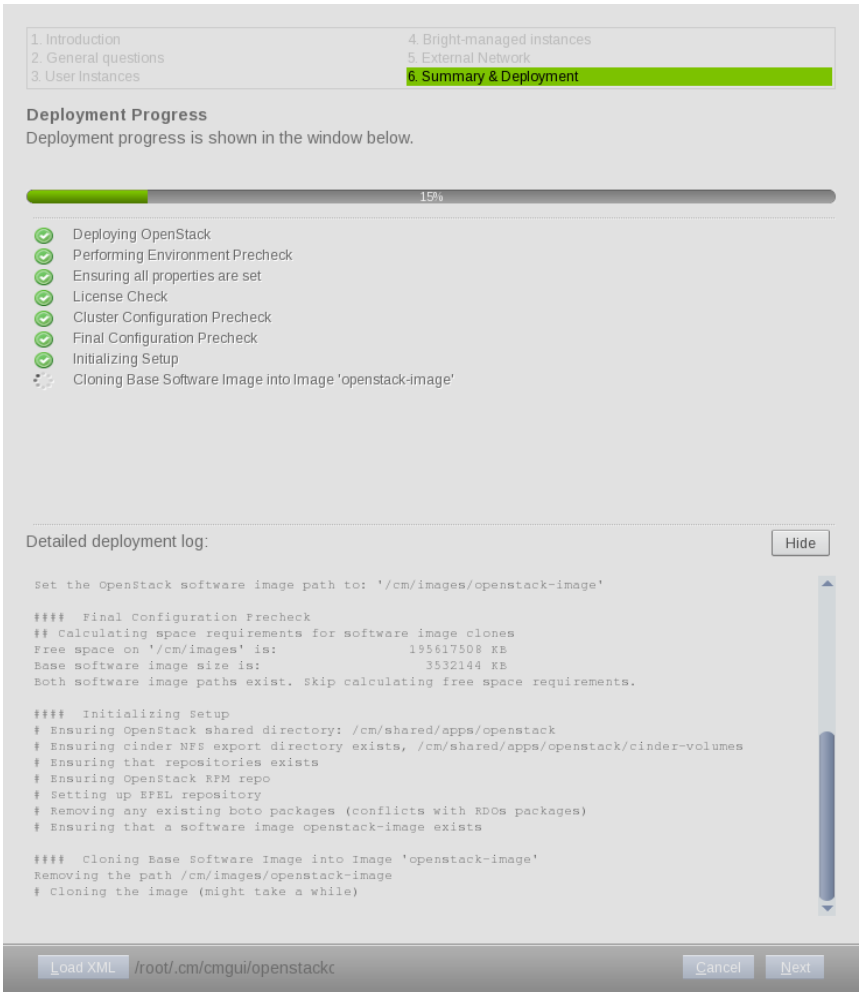

Figure 2.24: OpenStack Deployment Progress Screen

At the end of its run, the cluster has OpenStack set up and running in an integrated manner with Bright Cluster Manager.

The administrator can now configure the cluster to suit the particular site requirements.

2.2 Installation Of OpenStack From The Shell

The `cmgui` OpenStack installation (section 2.1) uses the `cm-openstack-setup` utility during deployment, hidden from normal view. The installation can also be done directly with `cm-openstack-setup`. The `cm-openstack-setup` utility is a less-preferred alternative to the installation of OpenStack from `cmgui`.

The `cm-openstack-setup` utility is a part of the standard `cluster-tools` package. Details on its use are given in its manual page (`man (8) cm-openstack-setup`). When run, the regular nodes that are to run OpenStack instances are rebooted by default at the end of the dialogs, in order to deploy them.

A prerequisite for running `cm-openstack-setup` is that the head node should be connected to the distribution repositories.

A sample `cm-openstack-setup` wizard session is described next,

starting from section 2.2.1. The session runs on a cluster consisting of one head node and one regular node. The wizard can be interrupted gracefully with a <ctrl-c>.

2.2.1 Start Screen

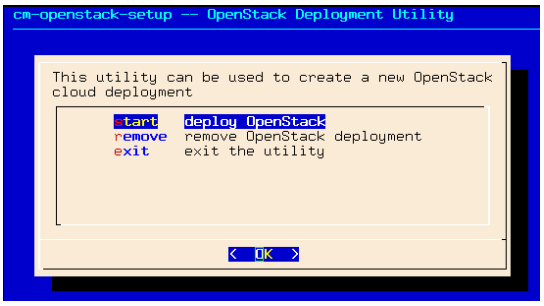

Figure 2.25: Start Screen

The start screen (figure 2.25) lets the administrator:

- deploy Bright Cluster Manager OpenStack.
- remove Bright Cluster Manager's OpenStack if it is already on the cluster.
- exit the installation.

Removal removes OpenStack-related database entries, roles, networks, virtual nodes, and interfaces. Images and categories related to OpenStack are however not removed.

2.2.2 Informative Text Prior To Deployment

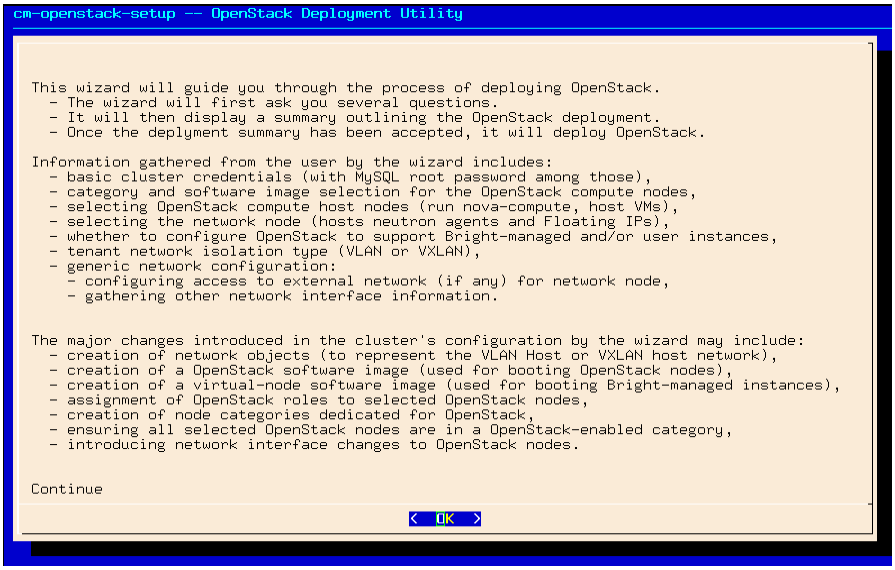

Figure 2.26: Informative Text Prior To Deployment

If deployment is selected in the preceding screen, an informative text screen (figure 2.26) gives a summary of what the script does.

2.2.3 Pre-Setup Suggestions

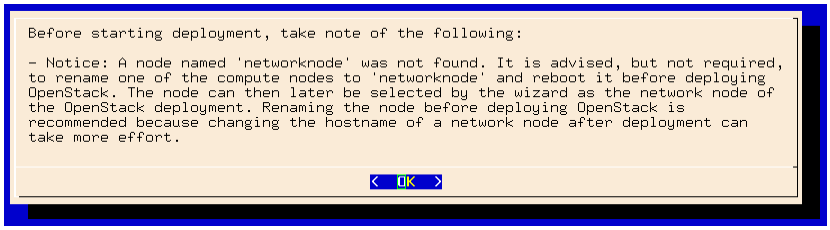

Figure 2.27: Pre-Setup Suggestions

The pre-setup suggestions screen (figure 2.27) suggests changes to be done before going on.

2.2.4 MySQL `root` And OpenStack `admin` Passwords

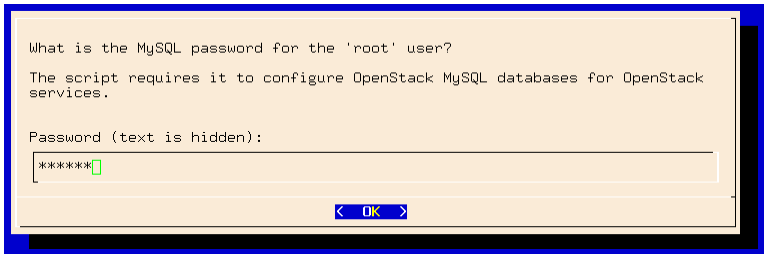

Figure 2.28: MySQL `root` Password Screen

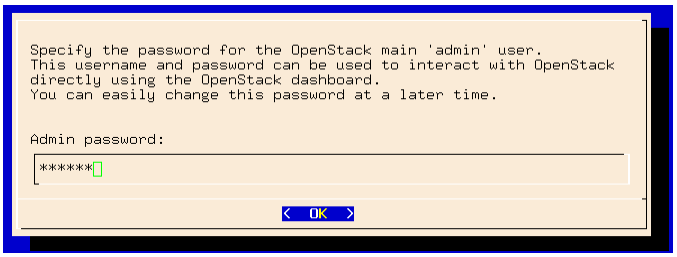

Figure 2.29: OpenStack `admin` Password Screen

The MySQL `root` password screen (figure 2.28) prompts for the existing root password to `MySQL` to be entered, while the OpenStack `admin` password screen (figure 2.29) prompts for a password to be entered, and then re-entered, for the soon-to-be-created `admin` user in OpenStack.

2.2.5 Reboot After Configuration

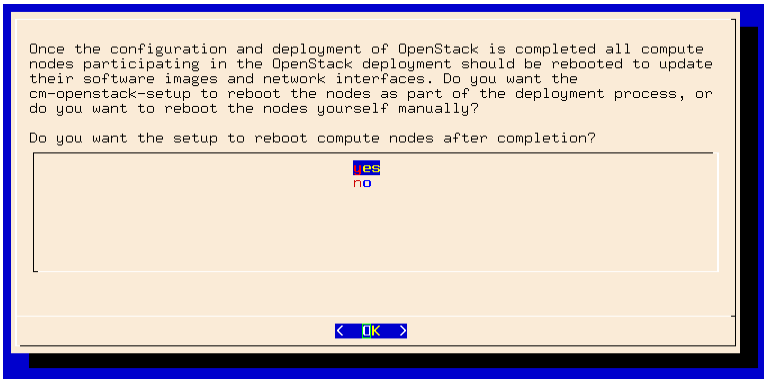

Figure 2.30: Reboot After Configuration Screen

A screen is shown asking if the compute host nodes, that is, the nodes used to host the virtual nodes, should be re-installed after configuration (figure 2.30). A re-install is usually best, for reasons discussed on page 6 for the `cmgui` installation wizard equivalent of this screen option.

2.2.6 Ceph Options

Background notes on the Ceph options can be read on page 10, in the section on the `cmgui` wizard configuration of Ceph options. This section (2.2.6) covers the Ncurses `cm-openstack-setup` wizard configuration of Ceph options.

Glance Image Storage With Ceph

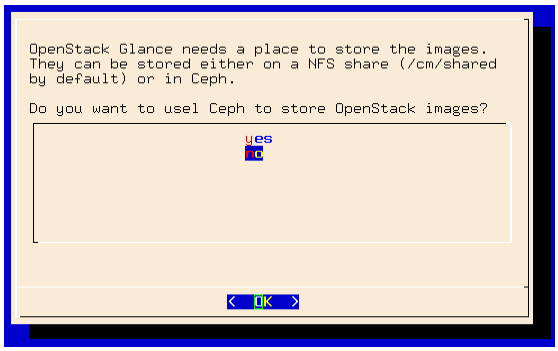

Figure 2.31: Image Storage With Ceph

Ceph can be set for storing virtual machine images, instead of the OpenStack reference Glance, using the Ceph image storage screen (figure 2.31).

Block Storage With Ceph

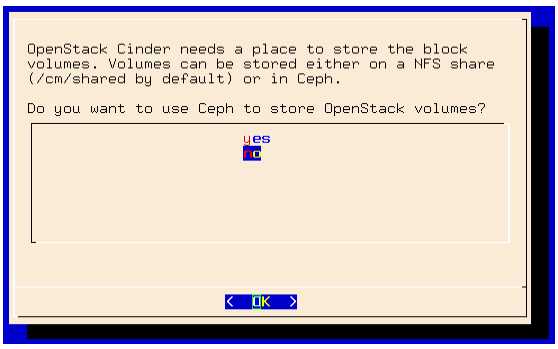

Figure 2.32: Block Storage With Ceph

Ceph can be set for handling block volume storage read and writes, instead of the OpenStack reference Cinder, by using the Ceph for OpenStack volumes screen (figure 2.32).

Root And Ephemeral Device Storage With Ceph

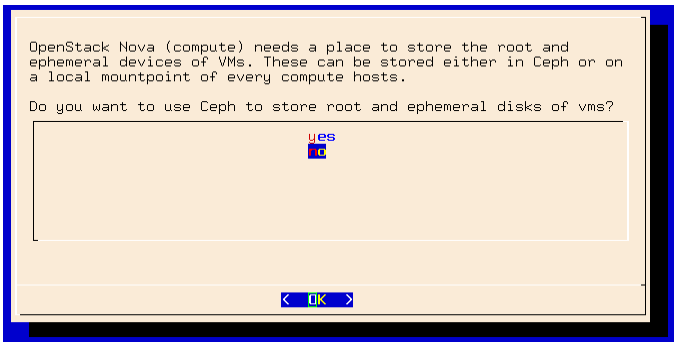

Figure 2.33: Root And Ephemeral Device Storage With Ceph

Data storage with Ceph can be enabled by the administrator by using the Ceph for OpenStack root and ephemeral device storage screen (figure 2.33).

Ceph Object Gateway (Ceph RADOS Gateway)

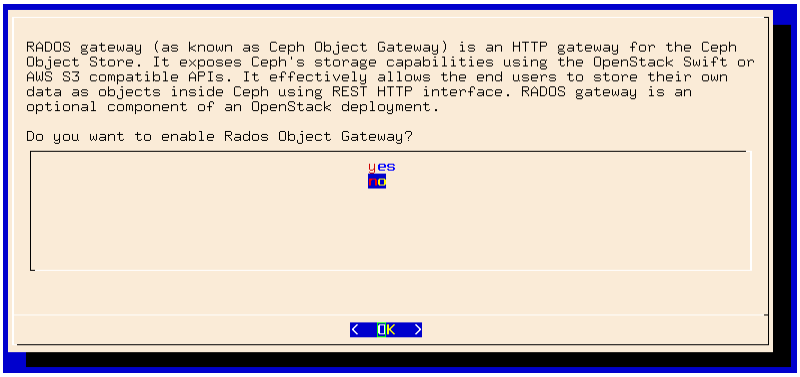

Figure 2.34: Root And Ephemeral Device Storage With Ceph

The Ceph RADOS gateway screen (figure 2.34) lets the administrator set the Ceph RADOS gateway service to run when deployment completes.

2.2.7 Internal Network To Be Used For OpenStack

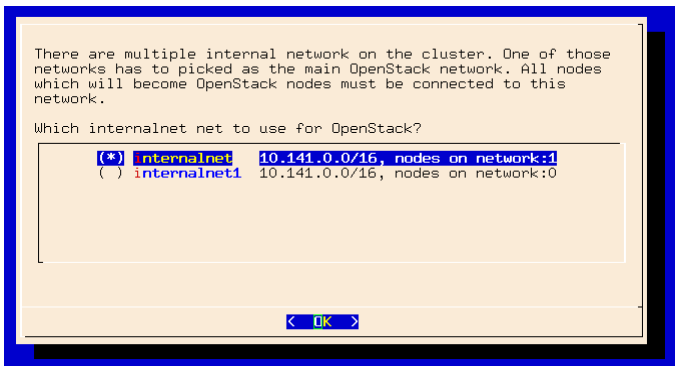

Figure 2.35: Internal Network To Be Used For OpenStack

If there are multiple internal networks, then the internal network selection screen (figure 2.35) lets the administrator choose which of them is to be used as the internal network to which the OpenStack nodes are to be connected.

2.2.8 User Instances

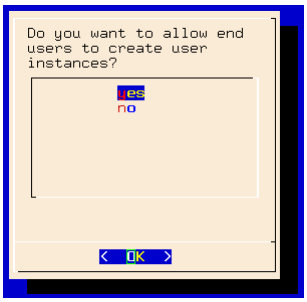

Figure 2.36: User Instances

The user instances screen (figure 2.36) lets the administrator decide if end users are to be allowed to create user instances.

2.2.9 Virtual Instance Access To Internal Network

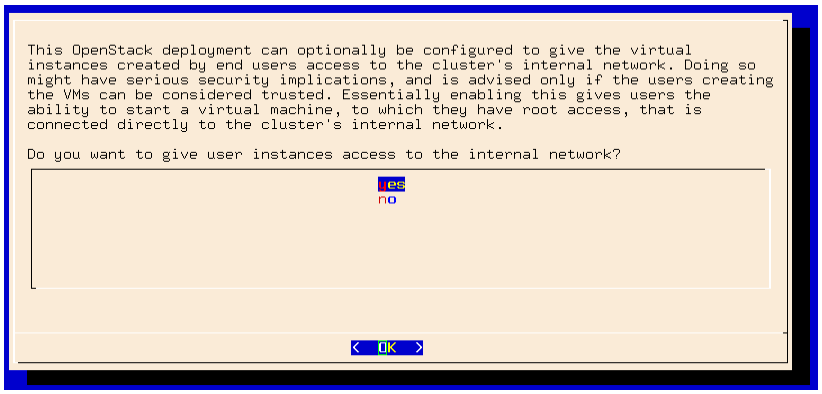

Figure 2.37: Virtual Instance Access To Internal Network

The screen in figure 2.37 lets the administrator allow virtual instances access to the internal network. This should only be allowed if the users creating the instances are trusted. This is because the creator of the instance has root access to the instance, which is in turn connected directly to the internal network of the cluster, which means all the packets in that network can be read by the user.

2.2.10 Network Isolation Type

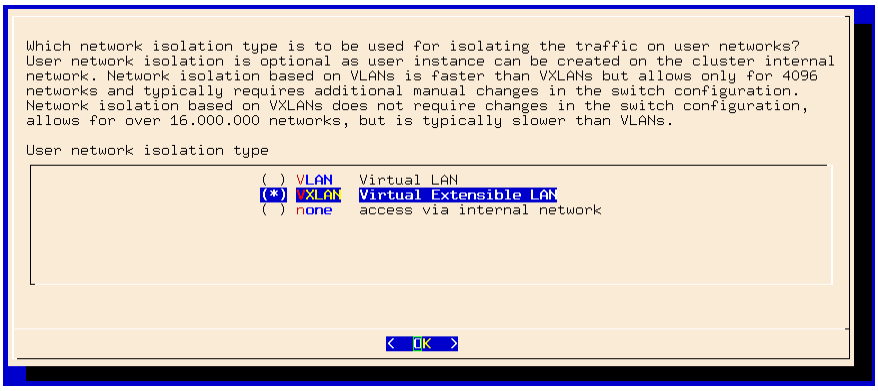

Figure 2.38: Network Isolation Type

The network isolation type screen (figure 2.38) allows the administrator to choose what kind of network isolation type, if any, should be set for the user networks.

2.2.11 Choosing The Network That Hosts The User Networks

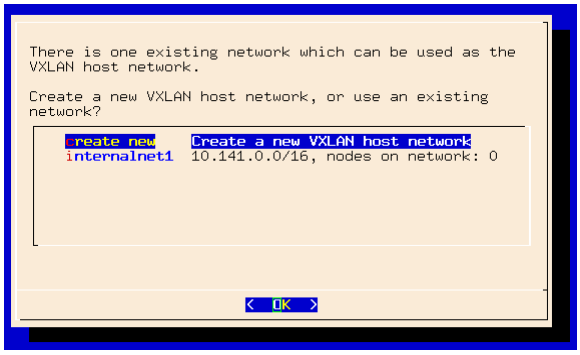

Figure 2.39: User Networks Hosting

If the user networks has their type (VXLAN, VLAN, or no virtual LAN) chosen in section 2.2.10, then a screen similar to figure 2.39 is displayed. This allows one network to be set as the host for the user networks.

If there are one or more possible networks already available for hosting the user networks, then one of them can be selected. Alternatively, a completely new network can be created to host them.

2.2.12 Setting The Name Of The Hosting Network For User Networks

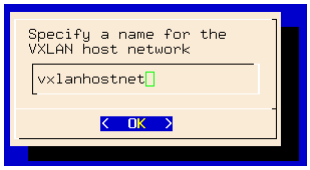

Figure 2.40: Setting The Name Of The Network For User Networks

If a network to host the user networks is chosen in section 2.2.11, then a screen similar to figure 2.40 is displayed. This lets the administrator set the name of the hosting network for user networks.

2.2.13 Setting The Base Address Of The Hosting Network For User Networks

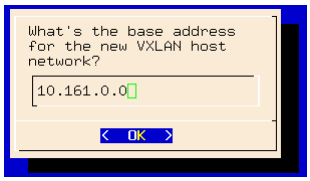

Figure 2.41: Setting The Base Address Of The Network For User Networks

If the network name for the network that hosts the user networks is chosen in section 2.2.12, then a screen similar to figure 2.41 is displayed. This lets the administrator set the base address of the hosting network for user networks.

2.2.14 Setting The Number Of Netmask Bits Of The Hosting Network For User Networks

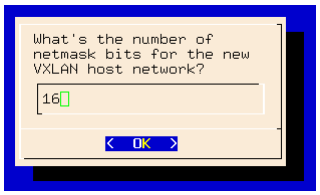

Figure 2.42: Setting The Number Of Netmask Bits Of The Network For User Networks

If the base address for the network that hosts the user networks is set in section 2.2.13, then a screen, similar to figure 2.42 is displayed. This lets the administrator set the number of netmask bits of the hosting network for user networks.

2.2.15 Enabling Support For Bright-managed Instances

Figure 2.43: Enabling Support For OpenStack Instances Under Bright Cluster Manager

There are two kinds of OpenStack instances, also known as virtual nodes, that can run on the cluster. These are called user instances and Bright-managed instances. The screen in figure 2.43 decides if Bright-managed instances to run. Bright-managed instances are actually a special case of user instances, just managed much more closely by Bright Cluster Manager.

Only if permission is set in the screen of section 2.2.9, can an end user access Bright-managed instances.

The screens from figure 2.44 to figure 2.46 are only shown if support for Bright-managed instances is enabled.

2.2.16 Starting IP Address For Bright-managed Instances

Figure 2.44: Starting IP Address For Bright-managed Instances

A starting IP address must be set for the Bright-managed instances (figure 2.44)

2.2.17 Ending IP Address For Bright-managed Instances

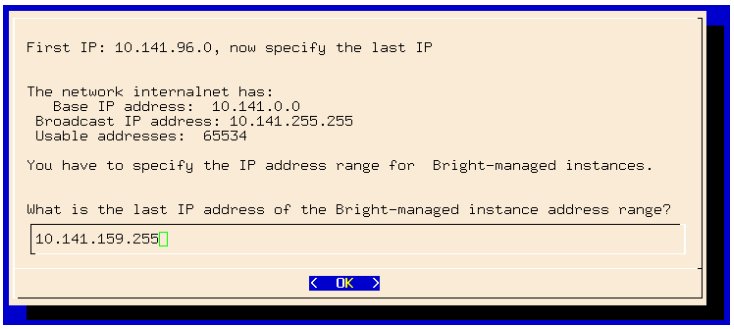

Figure 2.45: Ending IP Address For Bright-managed Instances

An ending IP address must be set for the Bright-managed instances (figure 2.45).

2.2.18 Number Of Virtual Nodes For Bright-managed Instances

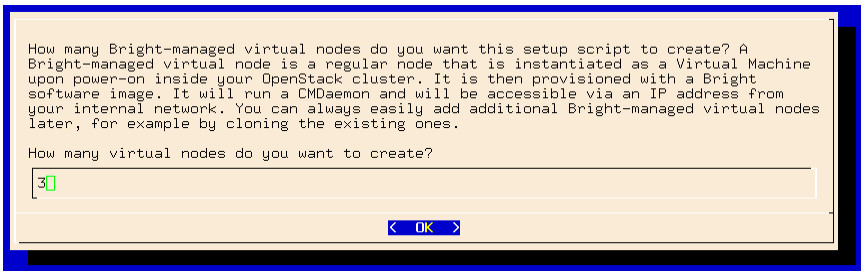

Figure 2.46: Number Of Virtual Nodes For Bright-managed Instances

The number of Bright-managed virtual machines must be set (figure 2.46). The suggested number of instances in the wizard conforms to the defaults that OpenStack sets. These defaults are based on an overcommit ratio of virtual CPU:real CPU of 16:1, and virtual RAM:real RAM of 1.5:1. The instance flavor chosen then determines the suggested number of instances.

2.2.19 DHCP And Static IP Addresses

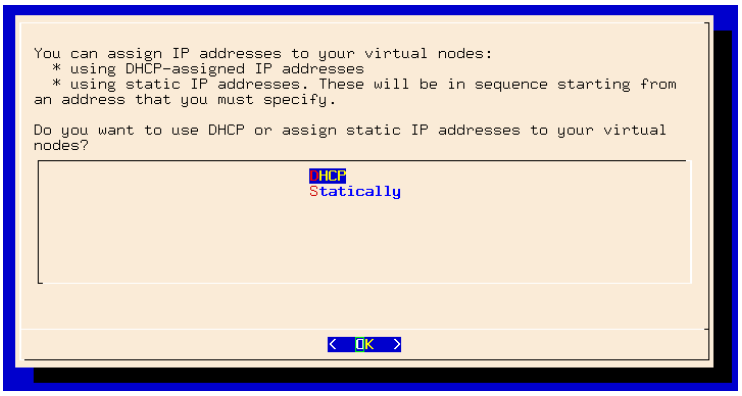

Figure 2.47: DHCP Or Static IP Address Selection

The instances can be configured to obtain their IP addresses either via DHCP, or via static address assignment (figure 2.48).

2.2.20 Floating IPs

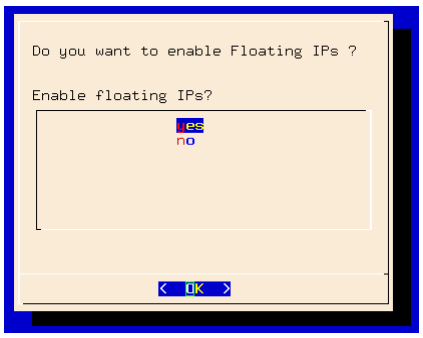

Figure 2.48: Floating IPs

The Floating IPs screen (figure 2.48) lets the administrator enable Floating IPs on the external network, so that instances can be accessed using these.

2.2.21 External Network Starting Floating IP

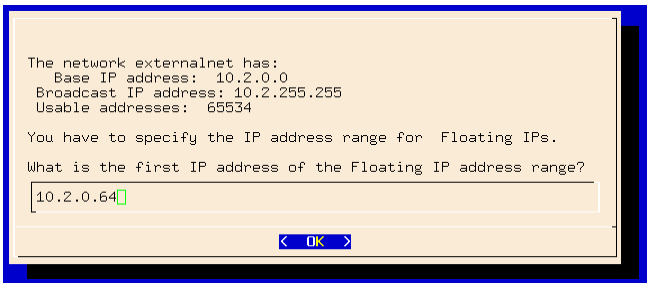

Figure 2.49: External Network: Starting Floating IP

A screen similar to figure 2.49 allows the administrator to specify the starting floating IP address on the external network.

2.2.22 External Network Ending Floating IP

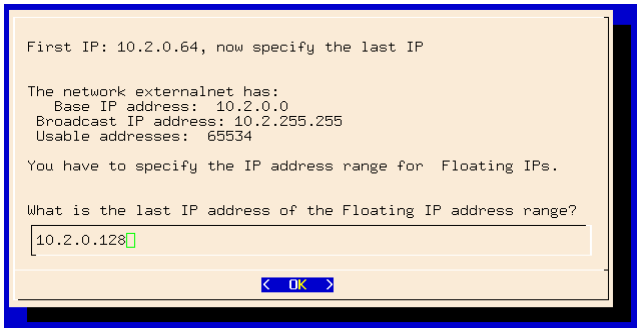

Figure 2.50: External Network: Ending Floating IP

A screen similar to figure 2.50 allows the administrator to specify the ending floating IP address on the external network.

2.2.23 VNC Proxy Hostname

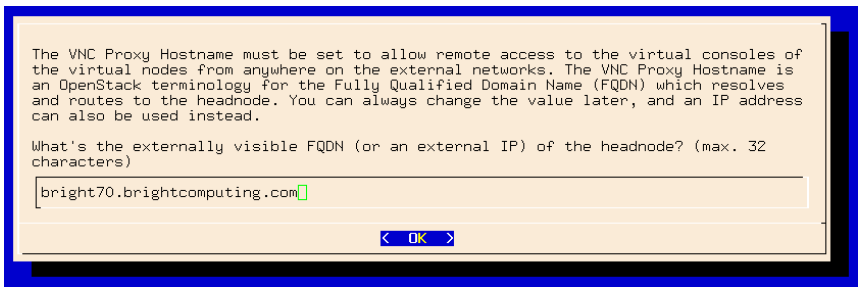

Figure 2.51: VNC Proxy Hostname

The VNC Proxy Hostname screen (figure 2.51) lets the administrator set the FQDN as seen from the external network. An IP address can be used instead of the FQDN.

2.2.24 Dedicated Default Tenant

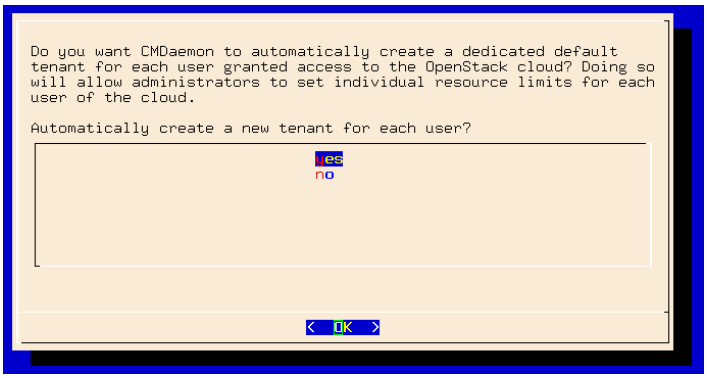

Figure 2.52: Setting A Dedicated Default Tenant

The dedicated default tenant setting screen (figure 2.52) allows a new tenant to be set automatically, with default quotas, for each user that is granted access the OpenStack cloud. This allows resource quotas to be

enforced.

2.2.25 Nova Compute Hosts

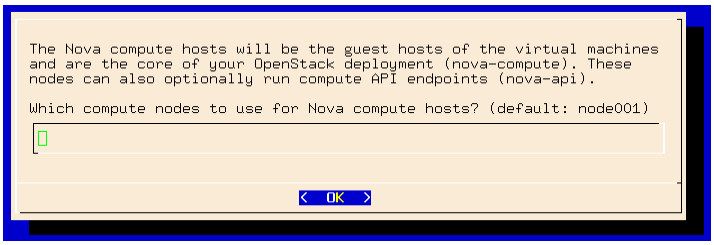

Figure 2.53: Nova Compute Hosts

The Nova compute hosts screen (figure 2.53) prompts the administrator to set the nodes to use as the hosts for the virtual machines.

2.2.26 Neutron Network Node

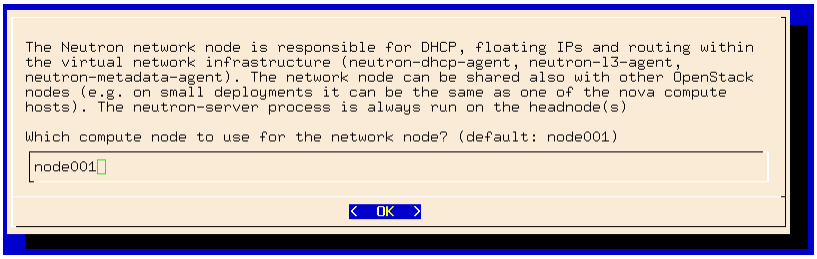

Figure 2.54: Neutron Network Node

The Neutron network node screen (figure 2.54) prompts the administrator to set the node to use for Neutron network node.

2.2.27 Pre-deployment Summary

Figure 2.55: Pre-deployment-summary

The pre-deployment summary screen (figure 2.55) displays a summary of the settings that have been entered using the wizard, and prompts the administrator to deploy or abort the installation with the chosen settings.

The options can also be saved as an XML configuration, by default `cm-openstack-setup.conf` in the directory under which the wizard is running. This can then be used as the input configuration file for the `cm-openstack-setup` utility using the `-c` option.

2.2.28 The State After Running `cm-openstack-setup`

At this point, the head node has OpenStack installed on it.

A regular node that has been configured with the OpenStack compute host role, ends up with OpenStack installed on it after the operating system running on the node is updated, and the newly-configured interfaces are set up. The update can be carried using `imageupdate` (section 5.6 of the *Administrator Manual*), or by rebooting the regular node. A reboot is however required if the interfaces have been changed, which is normally the case, unless the script is being run after a first run has already set up the changes. A reboot of the regular node is therefore normally a recommended action, because it ensures the updates and the interface changes have been carried out on the node. The reboot action is carried about by default (as shown in the preceding output), while the option is set by the `cm-openstack-setup` script (page 30).

Next, the administrator can further configure the cluster to suit requirements.

3

Ceph Installation

3.1 Ceph Introduction

Ceph, at the time of writing, is the recommended storage software for OpenStack for serious use. The Ceph RADOS Gateway is a drop-in replacement for Swift, with a compatible API. Ceph is the recommended backend driver for Cinder, Glance and Nova.

The current chapter discusses

- The concepts and required hardware for Ceph (section 3.1)
- Ceph installation and management (section 3.2)
- RADOS GW installation and management (section 3.4)

3.1.1 Ceph Object And Block Storage

Ceph is a distributed storage software. It is based on an object store layer called RADOS (Reliable Autonomic Distributed Object Store), which consists of Ceph components called OSDs (Object Storage Devices) and MONs (Monitoring Servers). These components feature heavily in Ceph. OSDs deal with storing the objects to a device, while MONs deal with mapping the cluster. OSDs and MONs, together carry out object storage and block storage within the object store layer. The stack diagram of figure 3.1 illustrates these concepts.

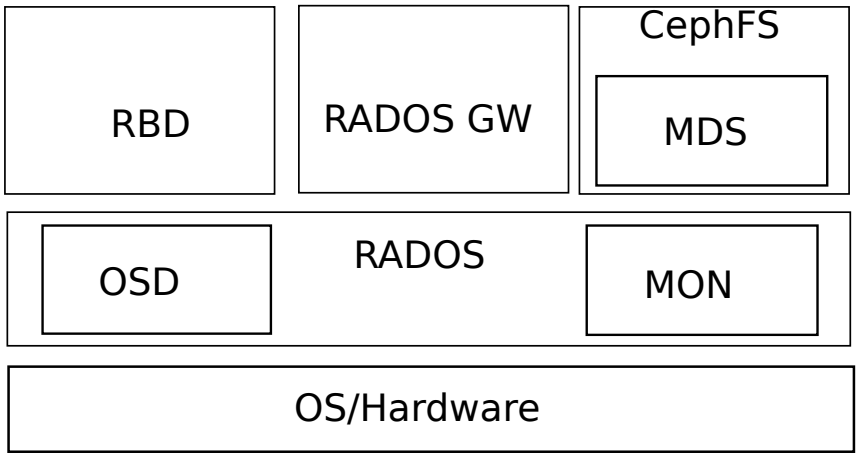

Figure 3.1: Ceph Concepts

On top of the object store layer are 3 kinds of access layers:

1. **Block device access:** RADOS Block Device (RBD) access can be carried out in two slightly different ways:

   (i) via a Linux kernel module based interface to RADOS. The module presents itself as a block device to the machine running that kernel. The machine can then use the RADOS storage, that is typically provided elsewhere.

   (ii) via the `librbd` library, used by virtual machines based on `qemu` or `KVM`. A block device that uses the library on the virtual machine then accesses the RADOS storage, which is typically located elsewhere.

2. **Gateway API access:** RADOS Gateway (RADOS GW) access provides an HTTP REST gateway to RADOS. Applications can talk to RADOS GW to access object storage in a high level manner, instead of talking to RADOS directly at a lower level. The RADOS GW API is compatible with the APIs of Swift and Amazon S3.

3. **Ceph Filesystem access:** CephFS provides a filesystem access layer. A component called MDS (Metadata Server) is used to manage the filesystem with RADOS. MDS is used in addition to the OSD and MON components used by the block and object storage forms when CephFS talks to RADOS. The Ceph filesystem is not regarded as production-ready by the Ceph project at the time of writing (July 2014), and is therefore not yet supported by Bright Cluster Manager.

3.1.2 Recommended Filesystem For Ceph Use

The storage forms of Ceph (object, block, or filesystem) can use a filesystem for storage. For production use of Ceph, XFS is currently the recommended filesystem option due to its stability, ability to handle extreme storage sizes, and its intrinsic ability to deal with the significant sizes of the extended attributes required by Ceph.

The nodes that run OSDs are typically regular nodes. Within the nodes, the storage devices used by the OSDs automatically have their filesystems configured to be of the XFS type during the installation of Ceph with Bright Cluster Manager.

Use Of `datanode` For Protection Of OSD Data

Typically, a filesystem used for an OSD is not on the same device as that of the regular node filesystem. Instead, typically, OSD storage consists of several devices that contain an XFS filesytem, with the devices attached to the node. These devices need protection from being wiped during the reprovisioning that takes place during a reboot of regular nodes .

The recommended way to protect storage devices from being wiped is to set the `datanode` property of their node to `yes` (page 186 of the *Administrator Manual*).

3.1.3 Hardware For Ceph Use

An absolute minimum installation: can be carried out on two nodes, where:

- 1 node, the head node, runs one Ceph Monitor and the first OSD.
- 1 node, the regular node, runs the second OSD.

This is however not currently recommended, because the first OSD on the head node requires its own Ceph-compatible filesystem. If that filesystem is not provided, then Ceph on the cluster will run, but in a degraded state. Using such a system to try to get familiar with how Ceph behaves in a production environment with Bright Cluster Manager is unlikely to be worthwhile.

A more useful minimum: if there is a node to spare, installing Ceph over 3 nodes is suggested, where:

- 1 node, the head node, runs one Ceph Monitor.
- 1 node, the regular node, runs the first OSD.
- 1 more node, also a regular node, runs the second OSD.

For production use: a redundant number of Ceph Monitor servers is recommended. Since the number of Ceph Monitoring servers must be odd, then at least 3 Ceph Monitor servers, with each on a separate node, are recommended for production purposes. The recommended minimum of nodes for production purposes is then 5:

- 2 regular nodes running OSDs.
- 2 regular nodes running Ceph Monitors.
- 1 head node running a Ceph Monitor.

Drives usable by Ceph: Ceph OSDs can use any type of disk that presents itself as a block device in Linux. This means that a variety of drives can be used.

3.2 Ceph Installation With cm-ceph-setup

3.2.1 cm-ceph-setup

Ceph installation for Bright Cluster Manager can be carried out with the ncurses-based cm-ceph-setup utility. It is part of the cluster-tools package that comes with Bright Cluster Manager. If the Ceph packages are not already installed, then the utility is able to install them for the head and regular nodes, assuming the repositories are accessible, and that the package manager priorities are at their defaults.

3.2.2 Starting With Ceph Installation, Removing Previous Ceph Installation

The cm-ceph-setup utility can be run as root from the head node.

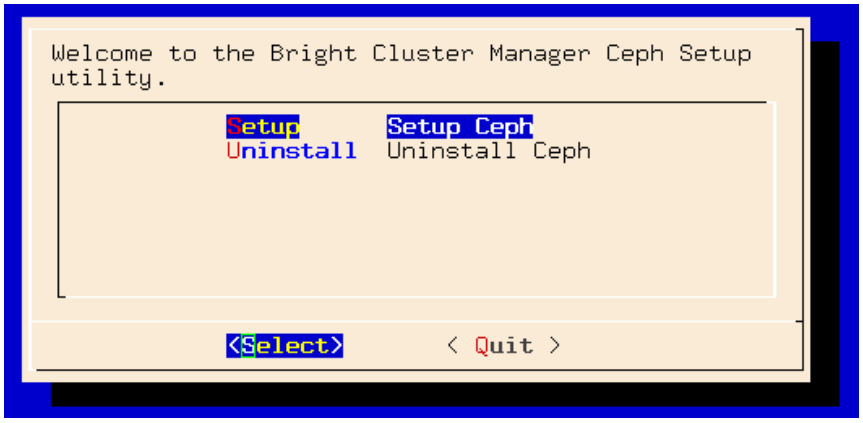

Figure 3.2: Ceph Installation Welcome

At the welcome screen (figure 3.2), the administrator may choose to

- Set up Ceph
- Remove Ceph if it is already installed.

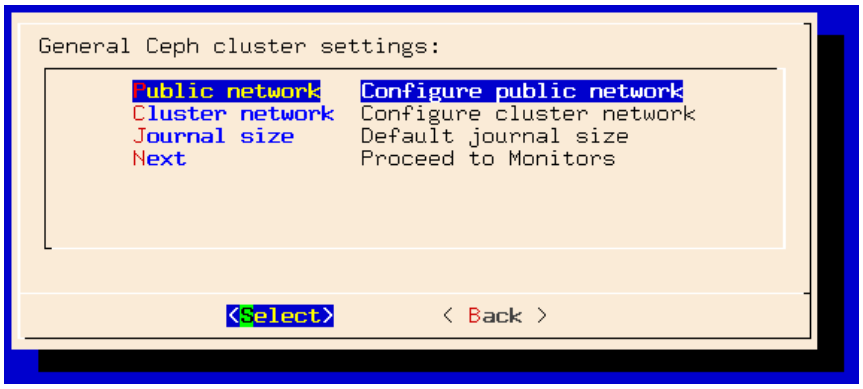

Figure 3.3: Ceph Installation General Cluster Settings

If the setup option is chosen, then a screen for the general Ceph cluster settings (figure 3.3) is displayed. The general settings can be adjusted via subscreens that open up when selected. The possible general settings are:

- `Public network`: This is the network used by Ceph Monitoring to communicate with OSDs. For a standard default Type 1 network this is `internalnet`.
- `Private network`: This is the network used by OSDs to communicate with each other. For a standard default Type 1 network this is `internalnet`.
- `Journal size`: The default OSD journal size, in MiBs, used by an OSD. The actual size must always be greater than zero. This is a general setting, and can be overridden by a category or node setting later on.

  Defining a value of 0 MiB here means that the default that the Ceph software itself provides is set. At the time of writing (March 2015), Ceph software provides a default of 5GiB.

Network Types are discussed in section 3.3.6 of the *Installation Manual*.

Selecting the Next option in figure 3.3 continues on with the next major screen of the setup procedure, and displays a screen for Ceph Monitors configuration (figure 3.4).

3.2.3 Ceph Monitors Configuration

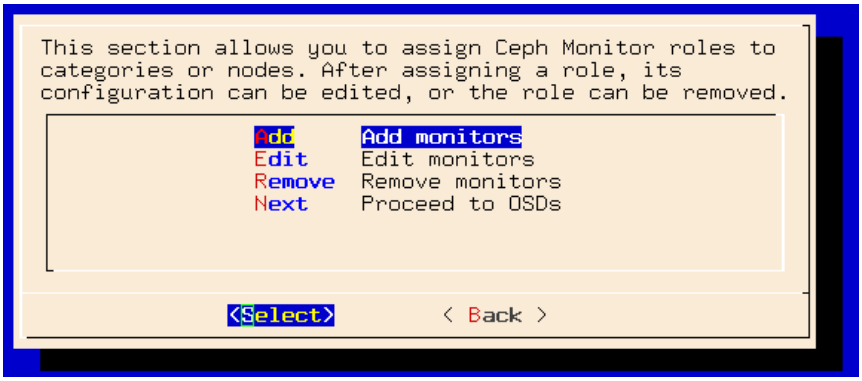

Figure 3.4: Ceph Installation Monitors Configuration

In this screen:

- Ceph Monitors can be added to nodes or categories.
- Existing Ceph Monitors can be edited or removed (figure 3.5), from nodes or categories.
- The OSD configuration screen can be reached after making changes, if any, to the Ceph Monitor configuration.

Typically in a first run, the head node has a Ceph Monitor added to it.

Editing Ceph Monitors

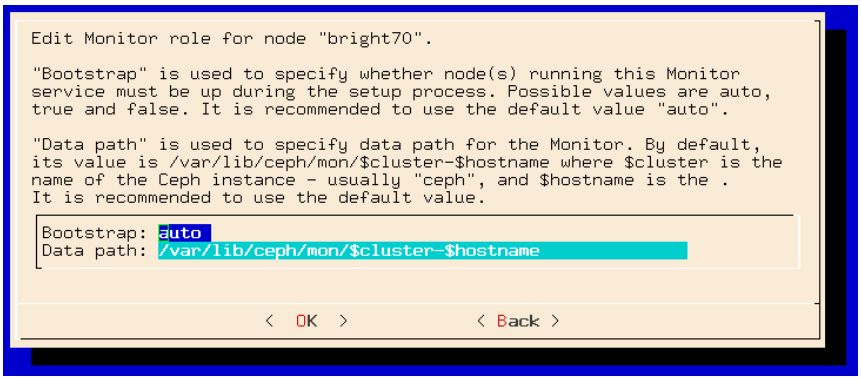

Figure 3.5: Ceph Installation Monitors Editing: Bootstrap And Data Path

The Edit option in figure 3.4 opens up a screen, figure 3.5, that allows the editing of existing or newly-added Ceph Monitors for a node or category:

- The bootstrap option can be set. The option configures initialization of the maps on the Ceph Monitors services, prior to the actual setup process. The bootstrap option can take the following values:

  - `auto`: This is the default and recommended option. If the majority of nodes are tagged with `auto` during the current configuration stage, and configured to run Ceph Monitors, then
    * If they are up according to Bright Cluster Manager at the time of deployment of the setup process, then the Monitor Map is initialized for those Ceph Monitors on those nodes.
    * If they are down at the time of deployment of the setup process, then the maps are not initialized.
  - `true`: If nodes are tagged `true` and configured to run Ceph Monitors, then they will be initialized at the time of deployment of the setup process, even if they are detected as being down during the current configuration stage.
  - `false`: If nodes are tagged `false` and configured to run Ceph Monitors, then they will not be initialized at the time of deployment of the setup process, even if they are detected as being up during the current configuration stage.

- The data path is set by default to:

  ```
  /var/lib/ceph/mon/$cluster-$hostname
  ```

  where:

  - `$cluster` is the name of the Ceph instance. This is `ceph` by default.
  - `$hostname` is the name of the node being mapped.

- The `Back` option can be used after accessing the editing screen, to return to the Ceph Monitors configuration screen (figure 3.4).

3.2.4 Ceph OSDs Configuration

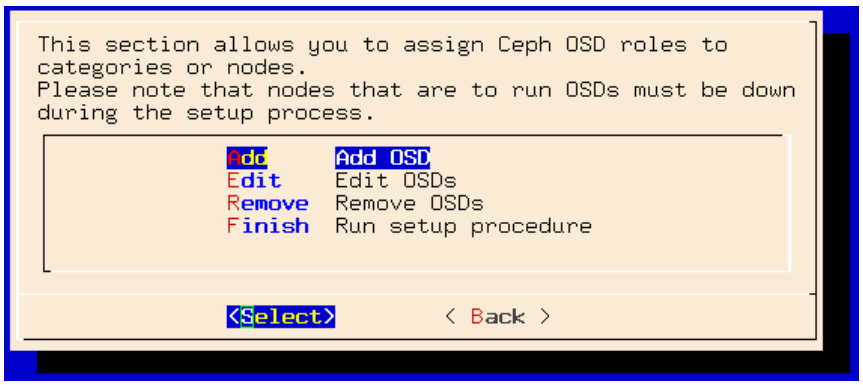

Figure 3.6: Ceph OSDs Configuration

If `Proceed to OSDs` is chosen from the Ceph Monitors configuration screen in figure 3.4, then a screen for Ceph OSDs configuration (figure 3.6) is displayed, where:

- OSDs can be added to nodes or categories. On adding, the OSDs must be edited with the edit menu.

- Existing OSDs can be edited or removed (figure 3.7), from nodes or categories.
- To finish up on the installation, after any changes to the OSD configuration have been made, the Finish option runs the Ceph setup procedure itself.

Editing Ceph OSDs

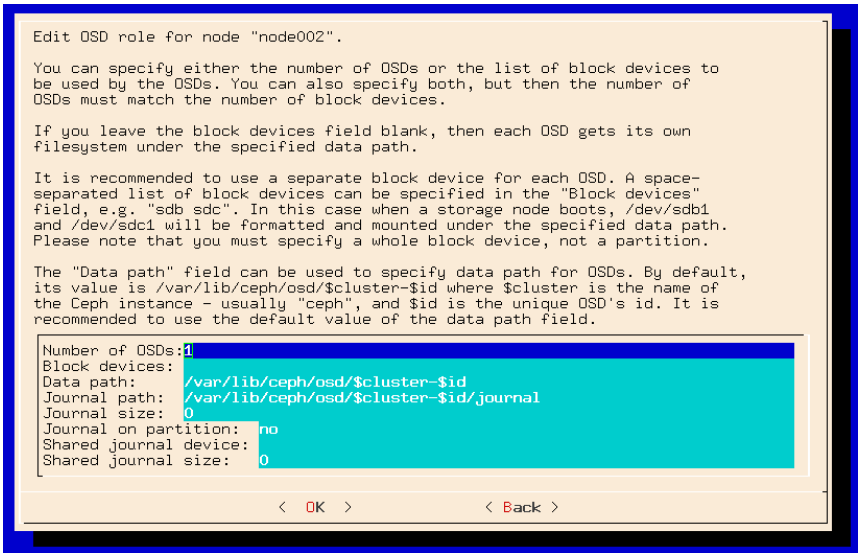

Figure 3.7: Ceph Installation OSDs Editing: Block Device Path, OSD Path, Journals For Categories Or Nodes

The Edit option in figure 3.6 opens up a screen, figure 3.7, that allows the editing of the properties of existing or newly-added Ceph OSDs for a node or category. In this screen:

- When considering the Number of OSDs and the Block devices, then it is best to set either
  - the Number of OSDs

  or
  - the Block devices

  Setting *both* the number of OSDs and block devices is also possible, but then the number of OSDs must match the number of block devices.
- If only a number of OSDs is set, and the block devices field is left blank, then each OSD is given its own filesystem under the data-path specified.
- Block devices can be set as a comma- or a space-separated list, with no difference in meaning.

Example

`/dev/sda,/dev/sdb,/dev/sdc`

and

`/dev/sda /dev/sdb /dev/sdc`

are equivalent.

- For the OSD `Data path`, the recommended, and default value is:

  `/var/lib/ceph/osd/$cluster-$id`

  Here:

  - `$cluster` is the name of the head node of the cluster.
  - `$id` is a number starting from `0`.

- For the `Journal path`, the recommended, and default value is:

  `/var/lib/ceph/osd/$cluster-$id/journal`

- The `Journal size`, in MiB, can be set for the category or node. A value set here overrides the default global journal size setting (figure 3.3). This is just the usual convention where a node setting can override a category setting, and a node or category setting can both override a global setting.

  Also, just like in the case of the global journal size setting, a journal size for categories or nodes must always be greater than zero. Defining a value of 0 MiB means that the default that the Ceph software itself provides is set. At the time of writing (March 2015), Ceph software provides a default of 5GiB.

  The `Journal size` for a category or node is unset by default, which means that the value set for `Journal size` in this screen is determined by whatever the global journal size setting is, by default.

- Setting `Journal on partition` to `yes` means that the OSD uses a dedicated partition. In this case:

  - The disk setup used is modified so that the first partition, with a size of `Journal size` is used
  - A value of `0` for the `Journal size` is invalid, and does not cause a Ceph default size to be used.

  The default value of `Journal on partition` is `no`.

- The `Shared journal device` path must be set if a shared device is used for all the OSD journals in the category or node for which this screen applies. The path is unset by default, which means it is not used by default..

- The `Shared journal size` in MiB can be set. For n OSDs each of size x MiB, the value of `Shared journal size` is $n \times x$. That is, its value is the sum of the sizes of all the individual OSD journals that are kept on the shared journal device. If it is used, then:

– The value of Shared journal size is used to automatically generate the disk layout setup of the individual OSD journals.
– A value of 0 for the Journal size is invalid, and does not cause a Ceph default size to be used.

The Shared journal size value is unset by default.

The Back option can be used after accessing the editing screen, to return to the Ceph OSDs configuration screen (figure 3.6).

Successful Completion Of The Ceph Installation

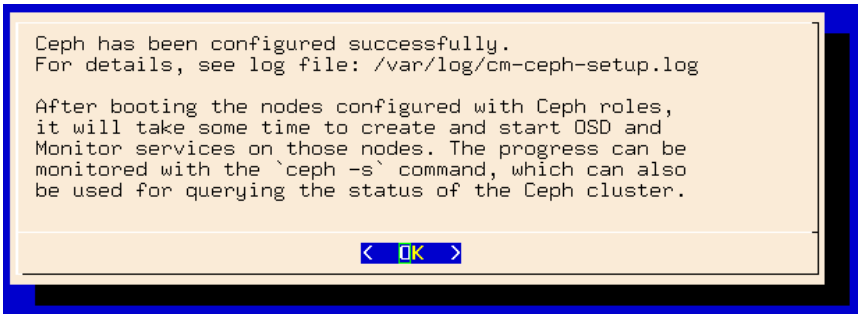

Figure 3.8: Ceph Installation Completion

After selecting the Finish option of figure 3.6, the Ceph setup proceeds. On successful completion, a screen as in figure 3.8 is displayed.

3.3 Checking And Getting Familiar With Ceph Items After cm-ceph-setup

3.3.1 Checking On Ceph And Ceph-related Files From The Shell

The status of Ceph can be seen from the command line by running:

Example

```
[root@bright70 ~]# ceph -s
   cluster d9422c23-321e-4fa0-b510-ca8e09a0a1fc
     health HEALTH_OK
     monmap e1: 1 mons at bright70=10.141.255.254:6789/0, election ep\
och 2, quorum 0 bright70
     osdmap e6: 2 osds: 2 up, 2 in
      pgmap v9: 192 pgs, 3 pools, 0 bytes data, 0 objects
            2115 MB used, 18340 MB / 20456 MB avail
                 192 active+clean
```

The -h option to ceph lists many options. Users of Bright Cluster Manager should usually not need to use these, and should find it more convenient to use the cmgui or cmsh front ends instead.

Generated XML Configuration File

By default, an XML configuration file is generated by the cm-ceph-setup utility, and stored after a run in the current directory as:

```
./cm-ceph-setup-config.xml
```

The name of the Ceph instance is by default ceph. If a new instance is to be configured with the cm-ceph-setup utility, then a new name must be set in the configuration file, and the new configuration file must be used.

Using An XML Configuration File

The -c option to cm-ceph-setup allows an existing XML configuration file to be used.

Example

```
[root@bright70 ~]# cm-ceph-setup -c /root/myconfig.xml
```

A Sample XML Configuration File

A sample schema for configuration can be seen at

```
/cm/local/apps/cluster-tools/other/samplecephconf.xml
```

Installation Logs

Installation logs to Ceph are kept at:

```
/var/log/cm-ceph-setup.log
```

3.3.2 Ceph Management With cmgui And cmsh

Only one instance of Ceph is supported at a time. Its name is ceph.

Ceph Overview And General Properties

From within cmsh, ceph mode can be accessed:

Example

```
[root@bright70 ~]# cmsh
[bright70]% ceph
[bright70->ceph]%
```

From within ceph mode, the overview command lists an overview of Ceph OSDs, MONs, and placement groups for the ceph instance. Parts of the displayed output are elided in the example that follows for viewing convenience:

Example

```
[bright70->ceph]% overview ceph
Parameter                      Value
------------------------------ ----------------------------
Status                         HEALTH_OK
Number of OSDs                 2
Number of OSDs up              2
Number of OSDs in              2
Number of mons                 1
Number of placements groups    192
Placement groups data size     0B
Placement groups used size     10.07GB
Placement groups available size 9.91GB
Placement groups total size    19.98GB

Name                           Used       Objects     ...
```

```
-------------------------------- ---------- ----------  ...
bright70:.rgw                    1B         0           ...
bright70:data                    0B         0           ...
bright70:metadata                0B         0           ...
bright70:rbd                     0B         0           ...
...
```

The cmgui equivalent of the overview command is the Overview tab, accessed from within the Ceph resource.

Some of the major Ceph configuration parameters can be viewed and their values managed by CMDaemon from ceph mode. The show command shows parameters and their values for the ceph instance:

Example

```
[bright70->ceph]% show ceph
Parameter                      Value
------------------------------ ----------------------------------------
Admin keyring path             /etc/ceph/ceph.client.admin.keyring
Bootstrapped                   yes
Client admin key               AQDkUM5T4LhZFxAA/JQHvzvbyb9txH0bwvxUSQ==
Cluster networks
Config file path               /etc/ceph/ceph.conf
Creation time                  Thu, 25 Sep 2014 13:54:11 CEST
Extra config parameters
Monitor daemon port            6789
Monitor key                    AQDkUM5TwM2lEhAA0CcdH/UFhGJ902n3y/Avng==
Monitor keyring path           /etc/ceph/ceph.mon.keyring
Public networks
Revision
auth client required cephx     yes
auth cluster required cephx    yes
auth service required cephx    yes
filestore xattr use omap       no
fsid                           abf8e6af-71c0-4d75-badc-3b81bc2b74d8
mon max osd                    10000
mon osd full ratio             0.95
mon osd nearfull ratio         0.85
name                           ceph
osd pool default min size      0
osd pool default pg num        8
osd pool default pgp num       8
osd pool default size          2
version                        0.80.5
[bright70->ceph]%
```

The cmgui equivalent of these settings is the Settings tab, accessed from within the Ceph resource.

Ceph OSD Properties

From within ceph mode, the osdinfo command for the Ceph instance displays the nodes that are providing OSDs along with their OSD IDs:

Example

```
[bright70->ceph]% osdinfo ceph
```

```
OSD id       Node                  OSD name
------------ --------------------- ------------
0            node001               osd0
1            node002               osd0
```

Within a device or category mode, the `roles` submode allows parameters of an assigned `cephosd` role to be configured and managed.

Example

```
[bright70->category[default]->roles]% show cephosd
Parameter                  Value
-------------------------- ------------------------------
Name                       cephosd
OSD associations           <1 in submode>
Provisioning associations  <0 internally used>
Revision
Type                       CephOSDRole
```

Within the `cephosd` role the templates for OSD filesystem associations, `osdassociations`, can be set or modified:

Example

```
[bright70->category[default]->roles]% use cephosd
[bright70...[default]->roles[cephosd]]% osdassociations
[bright70...osd]->osdassociations]% list -f name:10,osddata:30
name (key) osddata
---------- ------------------------------
osd0       /var/lib/ceph/osd/$cluster-$id
[bright70...osd->osdassociations]% list -f journaldata:38,journalsize:11
name (key) journaldata                            journalsize
---------- -------------------------------------- -----------
osd0       /var/lib/ceph/osd/$cluster-$id/journal 0
```

The `-f` option is used here with the `list` command merely in order to format the output so that it stays within the margins of this manual.

The `cmgui` equivalent of the preceding `cmsh` settings is accessed from within a particular `Nodes` or `Categories` item in the resource tree, then accessing the `Ceph` tab, and then choosing the OSD checkbox. The `Advanced` button allows `cephosd` role parameters to be set for the node or category.

Ceph Monitoring Properties

Similarly, the parameters of the `cephmonitor` role can be configured and managed from within the node object that runs Ceph Monitoring.

Example

```
[bright70]% device use bright70
[bright70->device[bright70]]% roles ; use cephmonitor
[ceph->device[bright70]->roles[cephmonitor]]% show
Parameter                      Value
------------------------------ ----------------------------------
Extra config parameters
Monitor data                   /var/lib/ceph/mon/$cluster-$hostname
Name                           cephmonitor
```

```
Provisioning associations        <0 internally used>
Revision
Type                             CephMonitorRole
```

Monitors in cmgui are similarly accessible for nodes and categories, with an Advanced button in their Ceph tab allowing the parameters for the Monitor checkbox to be set.

Ceph bootstrap

For completeness, the bootstrap command within ceph mode can be used by the administrator to initialize Ceph Monitors on specified nodes if they are not already initialized. Administrators are however not expected to use it, because they are expected to use the cm-ceph-setup installer utility when installing Ceph in the first place. The installer utility carries out the bootstrap initialization as part of its tasks. The bootstrap command is therefore only intended for use in the unusual case where the administrator would like to set up Ceph storage without using the cm-ceph-setup utility.

3.4 RADOS GW Installation, Initialization, And Properties

3.4.1 RADOS GW Installation And Initialization With cm-radosgw-setup

The cm-radosgw-setup utility can be run on the head node after installing Ceph with cm-ceph-setup. The cm-rados-setup utility configures RADOS so that it provides a REST API, called the RADOS GW service.

If cm-radosgw-setup is run with the -o option, then RADOS GW is integrated with OpenStack by enabling Keystone authentication for it.

Example

```
[root@bright70 ~]# cm-radosgw-setup -o
[root@bright70 ~]#
```

If cm-radosgw-setup is run without the -o option, then RADOS GW is installed, but Keystone authentication is disabled, and the gateway is therefore then not available to OpenStack instances.

Command line installation with the -o option initializes RADOS GW for OpenStack instances the first time it is run in Bright Cluster Manager.

3.4.2 Setting RADOS GW Properties

RADOS GW Properties In cmsh

RADOS GW properties can be managed in cmsh by selecting the device, then dropping into the radosgateway role:

```
[bright70]% device use bright70
[bright70->device[bright70]]% roles
[bright70->device[bright70]->roles]% use radosgateway
[bright70->device[bright70]->roles[radosgateway]]% show
Parameter                        Value
-------------------------------- --------------------------------
Name                             radosgateway
Provisioning associations        <0 internally used>
```

```
Revision
Type                                RadosGatewayRole
Module                              mod_fastcgi.so
Server Port                         18888
Server Root                         /var/www
Server Script                       s3gw.fcgi
Server Socket                       /tmp/radosgw.sock
Enable Keystone Authentication      yes
Keystone Accepted Roles
Keystone Revocation Interval        600
Keystone Tokens Cache Size          500
NSS DB Path                         /var/lib/ceph/nss
```

For example, setting `enablekeystoneauthentication` to `yes` and committing it makes RADOS GW services available to OpenStack instances, if they have already been initialized (section 3.4.1) to work with Bright Cluster Manager.

RADOS GW Properties In `cmgui`

RADOS GW properties can be accessed in `cmgui` by selecting the device from the resource tree, then selecting the `Ceph` subtab for that device, then ticking the `Rados Gateway` panel, then clicking on the `Advanced` button (figure 3.9).

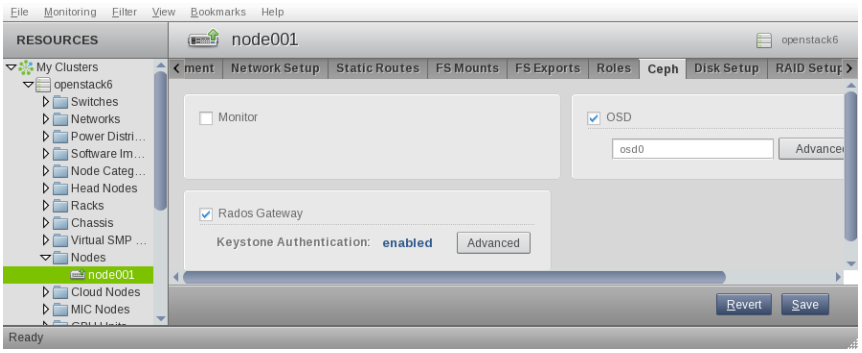

Figure 3.9: Accessing The RADOS GW Properties Button In `cmgui`

This brings up the RADOS GW `Advanced Role Settings` window (figure 3.10), which allows RADOS GW properties to be managed. For example, ticking the `Enable Keystone Authentication` checkbox and saving the setting makes RADOS GW services available to OpenStack instances, if they have already been initialized (section 3.4.1) to work with Bright Cluster Manager.

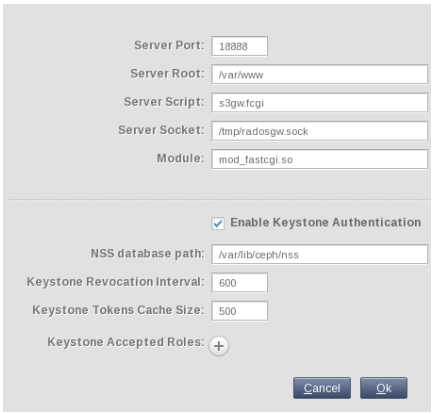

Figure 3.10: RADOS GW Advanced Role Settings In `cmgui`

3.4.3 Turning Keystone Authentication On And Off For RADOS GW

If command line installation is carried out for Bright Cluster Manager for the first time with `cm-radosgw-setup -o` (section 3.4.1), then RADOS GW is initialized for OpenStack instances.

This initialization for RADOS GW for OpenStack is not done by the `cmsh` commits or `cmgui` checkbox changes (section 3.4.2). So, toggling Ceph availability to OpenStack instances with `cmgui` and `cmsh` (section 3.4.3) is only possible if `cm-radosgw-setup -o` has been run once before to initialize the configuration (section 3.4.1).

If RADOS GW has been installed, then OpenStack instances can access it if Keystone authentication for RADOS GW is on. The following table summarizes how authentication is turned on and off.

Via:	RADOS GW Access On	RADOS GW Access Off
command line installation	`cm-radosgw-setup -o`	`cm-radosgw-setup`
`cmsh` device `radosgateway` role	`set enablekeystoneauthentication yes; commit`	`set enablekeystoneauthentication no; commit`
`cmgui` device, `Ceph` subtab	Ticked checkbox for `Enable Keystone Authentication`	Unticked checkbox for `Enable Keystone Authentication`

Table 3.4.3: Allowing OpenStack instances to access Ceph RADOS GW