



Red Hat Reference Architecture Series

Deploying a Highly Available Red Hat Satellite Server 6 Environment

Satellite Server version 6.2.1

Balaji Jayavelu
Principal Software Engineer
RHCE

Version 2.0

July 2016





100 East Davie Street
Raleigh NC 27601 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park NC 27709 USA

Linux is a registered trademark of Linus Torvalds. Red Hat, Red Hat Enterprise Linux and the Red Hat "Shadowman" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.

UNIX is a registered trademark of The Open Group.

Intel, the Intel logo and Xeon are registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

All other trademarks referenced herein are the property of their respective owners.

© 2016 by Red Hat, Inc. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at <http://www.opencontent.org/openpub/>).

The information contained herein is subject to change without notice. Red Hat, Inc. shall not be liable for technical or editorial errors or omissions contained herein.

Distribution of modified versions of this document is prohibited without the explicit permission of Red Hat Inc.

Distribution of this work or derivative of this work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from Red Hat Inc.

The GPG fingerprint of the security@redhat.com key is:
CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

Send feedback to refarch-feedback@redhat.com

Comments and Feedback

In the spirit of open source, we invite anyone to provide feedback and comments on any reference architectures. Although we review our papers internally, sometimes issues or typographical errors are encountered. Feedback allows us to not only improve the quality of the papers we produce, but allows the reader to present thoughts on potential improvements and topic expansion to the papers.

Feedback on the papers can be provided by emailing refarch-feedback@redhat.com. Please refer to the title within the email.

Staying In Touch

Join us on some of the popular social media sites where we keep our audience informed on new reference architectures as well as offer related information on things we find interesting.

Like us on Facebook:

<https://www.facebook.com/rhrefarch>

Follow us on Twitter:

<https://twitter.com/RedHatRefArch>

Plus us on Google+:

<https://plus.google.com/u/0/b/114152126783830728030/>

Table of Contents

1 Executive Summary.....	1
2 Components Overview.....	2
2.1 Red Hat Satellite 6 Architecture.....	2
2.2 Red Hat Satellite 6 Overview.....	3
2.2.1 Provisioning.....	3
2.2.2 Configuration Management.....	3
2.2.3 Software Management.....	4
2.2.4 Subscription Management.....	4
2.2.5 Foreman.....	4
2.2.6 Katello.....	4
2.2.7 Candlepin.....	5
2.2.8 Pulp.....	5
2.2.9 Hammer.....	5
2.2.10 REST API.....	5
2.2.11 Capsule.....	5
3 Reference Architecture Configuration Details.....	6
3.1 Environment.....	6
3.1.1 Required Repositories (repos).....	7
3.1.1.1 Repositories for Satellite 6.2.1 server.....	7
3.1.1.2 Channels for Satellite Capsules.....	7
3.1.2 Software Versions.....	7
3.1.2.1 Operating Systems Required To Run Satellite.....	7
3.1.2.2 Satellite Server.....	8
3.1.2.3 Satellite Capsule.....	8
3.1.3 Security Reference.....	9
3.1.3.1 Satellite Server.....	9
3.1.3.2 Satellite Capsule Server.....	9
3.1.3.3 Pacemaker Ports.....	9
3.1.4 Server Hardware Configuration.....	10
3.1.4.1 Server Resource Specifications.....	10
3.1.5 Storage Configuration.....	11
3.1.6 Network Configuration.....	11
4 Installation and Configuration.....	13
4.1 Installing Highly Available Satellite Server.....	13

4.1.1	Satellite Server Installation Prerequisites.....	14
4.1.2	Install Operating System And Configure Servers.....	15
4.1.2.1	Operating System Setup.....	15
4.1.2.2	Subscription Settings.....	15
4.1.2.3	Shared SSH Keys.....	16
4.1.2.4	Chrony Setting.....	17
4.1.2.5	Firewall Setting.....	18
4.1.2.6	ISCSI Settings.....	18
4.1.2.7	Create Shared Filesystems For HA LVM.....	19
4.1.3	Install Satellite Server.....	22
4.1.3.1	Installing Satellite On The First Node - <i>satnode1</i>	22
4.1.3.2	Install Integrated Capsule On First Node - <i>satnode1</i>	23
4.1.3.3	Verify The Satellite Functionality.....	24
4.1.3.4	Backup Of Satellite Environment.....	25
4.1.3.5	Cleanup On <i>satnode1</i>	26
4.1.3.6	Install Satellite On Second Node – <i>satnode2</i>	27
4.1.3.7	Restore backed up content from <i>satnode1</i> on <i>satnode2</i>	28
4.1.3.8	Verify Satellite Functionality.....	28
4.1.3.9	Installing Satellite on third node – <i>satnode3</i>	29
4.1.3.10	Add Trusted Hosts.....	29
4.1.4	Pacemaker Install.....	30
4.1.4.1	Install Pacemaker Packages.....	30
4.1.4.2	Create The Cluster.....	30
4.1.4.3	Create Fencing.....	31
4.1.4.4	Exclusive Activation Of Shared Volume Group.....	32
4.1.4.5	Creating Pacemaker Resources.....	32
4.1.4.6	Resource Details.....	37
4.1.4.7	Pacemaker Cluster Status.....	38
4.1.4.8	Pacemaker Resource Stickiness.....	39
4.1.5	Verify Cluster Fail Over.....	40
4.1.6	Backing Up And Restoring A Pacemaker Cluster Configuration.....	41
4.2	Configuring Satellite 6 Environment.....	42
4.2.1	Satellite 6 Server Configuration Work Flow.....	42
4.2.2	Administer.....	43
4.2.2.1	Change Satellite User Password.....	43
4.2.2.2	Create Organization.....	43
4.2.2.3	Define Location.....	43
4.2.3	Content.....	44
4.2.3.1	Red Hat Subscriptions.....	44

4.2.3.2 Red Hat Repositories.....	46
4.2.3.3 Synchronize Content.....	47
4.2.3.4 Create Content View And Publish.....	48
4.2.3.5 Create Activation Keys.....	50
4.2.4 Hosts.....	51
4.2.4.1 Operating System.....	51
4.2.4.2 Provisioning Templates.....	51
4.2.4.3 Installation Media.....	53
4.2.5 Infrastructure.....	53
4.2.5.1 Create Domain.....	53
4.2.5.2 Create Subnet.....	54
4.2.5.3 Create Compute Resources.....	55
4.2.6 Configure.....	55
4.2.6.1 Host Groups.....	55
5 Configuring Load Balanced External Capsules.....	56
5.1 Capsule Configuration Detail.....	56
5.2 Installing Capsule Server For Foreman-Proxy Service.....	59
5.2.1 Installing cap1a capsule server.....	59
5.2.1.1 Hostname and Network.....	59
5.2.1.2 Subscriptions.....	59
5.2.1.3 Firewall Settings.....	60
5.2.1.4 Shared Filesystems.....	60
5.2.1.5 Capsule Server Certificate For Capsule Server “cap1”.....	61
5.2.1.6 Capsule Server Install.....	61
5.2.1.7 Multi-hosts Certificate for Virtual Capsule Server.....	63
5.2.1.8 Certificate Update.....	63
5.2.1.9 Update Template_URL.....	64
5.2.1.10 Configuration Backup.....	64
5.2.2 Installing cap1b capsule server.....	65
5.2.3 Installing cap1c capsule server.....	66
5.2.4 Pacemaker Install.....	66
5.2.4.1 Exclusive Activation Of Shared Volume Group.....	67
5.2.4.2 Creating Pacemaker Resources.....	68
5.2.4.3 Pacemaker Cluster Status.....	70
5.3 Installing Capsule Servers For Content Services.....	72
5.3.1 Installing Capsule Server cap2.....	72
5.3.1.1 Subscriptions.....	72
5.3.1.2 Firewall Settings.....	73

5.3.1.3 Capsule Server Certificate For Capsule Server “cap2”	73
5.3.1.4 Capsule Server Install.....	74
5.3.1.5 Certificate update.....	74
5.3.1.6 Assign Capsule and Synchronize Content.....	74
5.3.1.7 Installing Capsule Server cap3 and cap4.....	74
5.4 HAProxy installation.....	75
5.4.1 Subscription.....	75
5.4.2 Firewall Settings.....	75
5.4.3 Shared Filesystems.....	75
5.4.4 Install and Configure HAProxy.....	76
5.4.4.1 HAProxy configuration:.....	76
5.4.4.2 HAProxy logging.....	77
5.4.4.3 Manually Adding Virtual IP.....	78
5.4.4.4 Start HAProxy Services.....	78
5.4.5 Pacemaker Install.....	78
5.4.5.1 Exclusive Activation Of Shared Volume Group.....	79
5.4.5.2 Creating Pacemaker Resources.....	79
5.4.5.3 Pacemaker Cluster Activate Nodes.....	80
5.4.5.4 Creating Virtual Capsule Server.....	81
6 Provisioning.....	82
6.1 Provisioning A Host Using PXE.....	82
6.2 Test For Capsule Failure While Provisioning.....	88
6.2.1 Failure Of Capsule Server With Content.....	88
6.3 Satellite Fail Over During Provisioning.....	92
7 Conclusion.....	95
Appendix A: Troubleshooting.....	96
A.1 HAProxy – SELinux Issue.....	96
A.2 Registration Issues.....	96
A.2.1 Capsule Installation Failure.....	96
A.2.2 Time Synchronization Issues.....	97
A.3 Certificate update failure:.....	97
Appendix B: Configuration.....	98
B.1 Configuration Files.....	98
B.2 Provisioning Output Yaml File.....	99
B.3 Pacemaker Configuration Output.....	100

Appendix C: Scripts.....	106
C.1 Create Shared Filesystems Script.....	106
C.2 Cleanup Filesystem Script.....	107
C.3 Move Shared Filesystem Script.....	108
C.4 Manual Satellite Start And Stop Scripts.....	108
C.5 Mount And Unmount Filesystems Scripts.....	109
C.6 Satellite Backup Script.....	109
C.7 Satellite Restore Script.....	111
C.8 Create Pacemaker Resources Script.....	113
C.9 Capsule Backup Script:.....	115
C.10 Mult-host Capsule Certificate Generation Script.....	116
Appendix D: Contributors.....	120
Appendix E: Revision History.....	121

1 Executive Summary

Information Technology has been constantly evolving where the volume of change is exponentially rising, while the time interval of change is shrinking. To keep up with the pace, the infrastructure has to be on par meeting the scaling and diversity challenges. It is key to efficiently and effectively deploy and manage the IT infrastructure.

Red Hat Satellite is a complete system management product that allows [system administrators](#) to manage the full life cycle of Red Hat deployments across physical, virtual, and private clouds. Red Hat Satellite delivers system provisioning, configuration management, software management, and subscription management- all while maintaining high scalability and security..

Red Hat Satellite 6.2.1 is the new release of a new generation of Systems Management for Red Hat Enterprise Linux (RHEL) that demonstrates measurable improvements over Red Hat Satellite 5 environment on the following topics:

- Content management
- Container management
- Automation provisioning
- Discovery
- Configuration management
- Federated services
- Overall performance
- Scalability
- Common vulnerabilities and exposure management

The purpose of this reference architecture is to demonstrate how a Highly Available Red Hat Satellite 6.2 environment is deployed on servers running Red Hat Enterprise Linux 7.2 operating system through a detailed use case:

- Executing Satellite Installer.
- Highly available (HA) environment using Pacemaker.
- Installing and configuring external load balanced Satellite Capsule Server environment
- Demonstrate a succesful host provisioning during a Satellite Capsule server failure.

Every step of this use case has been tested in Red Hat's engineering lab with production code on bare-metal hardware for the Satellite server nodes and Red Hat Enterprise Virtualization for Capsule servers.

For further details on Red Hat Satellite Server, please refer to documentation¹

¹ <https://access.redhat.com/documentation/en/red-hat-satellite/?version=6.2/>

2 Components Overview

2.1 Red Hat Satellite 6 Architecture

Red Hat Satellite 6 is based upon several open source projects as mentioned in the Illustration 1: Satellite 6 Architecture.

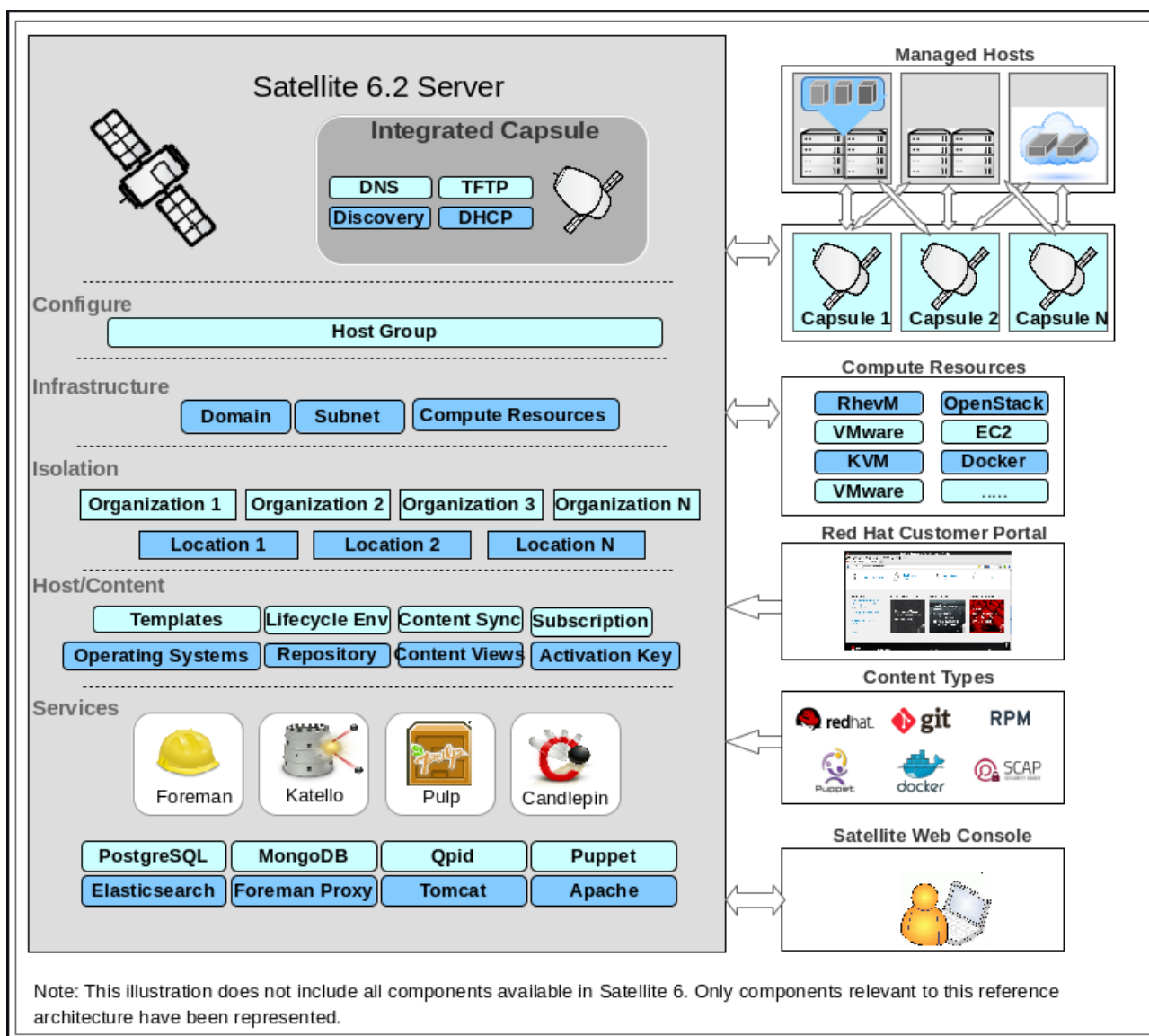


Illustration 1: Satellite 6 Architecture

2.2 Red Hat Satellite 6 Overview

Red Hat Satellite is a system management solution that makes Red Hat infrastructure easier to deploy, scale, and manage across physical, virtual, and cloud environments. Satellite helps users provision, configure, and update systems to ensure they run efficiently, securely, and in compliance with various standards. By automating most tasks related to maintaining systems, Satellite helps organizations increase efficiency, reduce operational costs, and enable IT to better respond to strategic business needs.

Red Hat Satellite automates many tasks related to system management and easily integrates into existing workflow frameworks. The centralized console provides administrators one place for accessing reports and for provisioning, configuring, and updating systems.

2.2.1 Provisioning

Provision on bare metal, virtualized infrastructure, and on public or private clouds—all from one centralized console and with one simple process.

- Quickly provision and update your entire bare-metal infrastructure.
- Easily create and manage instances across virtualized infrastructure or private and public clouds.
- Create complex Kickstart and PXE scenarios with powerful variables and snippets.
- Discover and search across non-provisioned hosts for rapid deployment.

2.2.2 Configuration Management

Analyze and automatically correct configuration drift and control, and enforce the desired host end-state, all from the Red Hat Satellite user interface (UI). This lets you configure Red Hat Enterprise Linux systems more efficiently for more agility.

- Integrate synchronizing Puppet modules. This integration provides the ability to manage, promote, and distribute configuration easily across your environment.
- Automatically correct system state with complete reporting, auditing, and history of changes.
- Integrate synchronizing Puppet modules. This integration provides the ability to manage, promote, and distribute configuration easily across your environment.
- Automatically correct system state with complete reporting, auditing, and history of changes.

2.2.3 Software Management

Red Hat Satellite helps ensure a systematic process is used to apply content (including patches) to deployed systems— whether they are deployed on physical, virtual, or cloud infrastructure—in all stages from dev to production. This ensures better consistency and availability of systems, freeing IT to quickly respond to business needs and vulnerabilities.

- Content views are collections of RPMs, container content, or Puppet modules refined with filters and rules. Content views are published and promoted throughout lifecycle environments, enabling end-to-end system management. While Satellite 5 used channels and cloning, content views in Satellite 6 contain both software and configuration content in one place, greatly simplifying managing the lifecycles of systems.
- Integrated with the Red Hat CDN to let users control synchronization of Red Hat content straight from the UI.
- Distribution and federation of provisioning, configuration, and content delivery via Red Hat Satellite Capsule Server.

2.2.4 Subscription Management

Easily report and map your Red Hat products to registered systems for end-to-end subscription consumption visibility.

Easily import and manage the distribution of your Red Hat software subscriptions.

- Report and map your purchased products to registered systems within Red Hat
- Satellite for end-to-end subscription usage visibility.

2.2.5 Foreman

Foreman is an open source application used for provisioning and lifecycle management of physical and virtual systems. Foreman automatically configures these systems using various methods, including kickstart and Puppet modules. Foreman also provides historical data for reporting, auditing, and troubleshooting.

2.2.6 Katello

Katello is a subscription and repository management application. It provides a means to subscribe to Red Hat repositories and download content. You can create and manage



different versions of this content and apply them to specific systems within user-defined stages of the application lifecycle.

2.2.7 Candlepin

Candlepin is a service within Katello that handles subscription management.

2.2.8 Pulp

Pulp is a service within Katello that handles repository and content management.

2.2.9 Hammer

Hammer is a CLI tool that provides command line and shell equivalents of most Web UI functions.

2.2.10 REST API

Red Hat Satellite 6 includes a RESTful API service that allows system administrators and developers to write custom scripts and third-party applications that interface with Red Hat Satellite.

2.2.11 Capsule

Red Hat Satellite Capsule Server acts as a proxy for some of the main Satellite functions including repository storage, DNS, DHCP, and Puppet Master configuration. Each Satellite Server also contains integrated Capsule Server services.

For further details on Satellite Server components, refer to [documentation](#) ²

² <https://access.redhat.com/products/red-hat-satellite/overview>

3 Reference Architecture Configuration Details

This section describes the hardware, software, and prerequisites used to configure the reference architecture.

3.1 Environment

The reference architecture environment consists of the components required to build a highly available Red Hat Satellite 6 infrastructure.

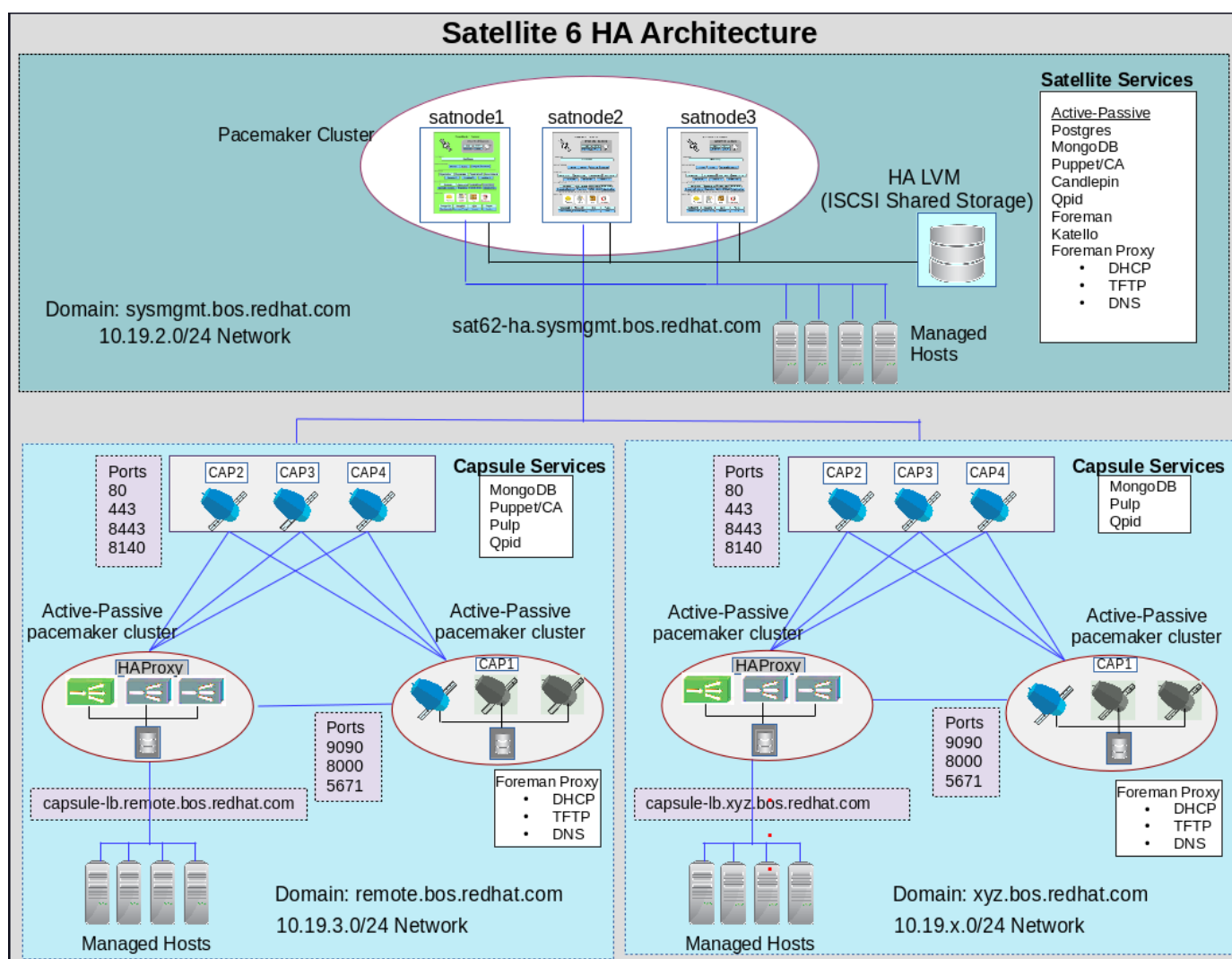


Illustration 2: Satellite 6 HA Architecture

3.1.1 Required Repositories (repos)

3.1.1.1 Repositories for Satellite 6.2.1 server

Repository	repo Name
Red Hat Enterprise Linux 7 Server	rhel-7-server-rpms
Red Hat Satellite 6.2	rhel-7-server-satellite-6.2-rpms
Red Hat Enterprise Linux RHSCl	rhel-server-rhsc1-7-rpms
Red Hat Enterprise Linux High Availability	rhel-ha-for-rhel-7-server-rpms

Table 3.1.1.1: Required Channels -Satellite

3.1.1.2 Channels for Satellite Capsules

Repository	repo Name
Red Hat Enterprise Linux 7 Server	rhel-7-server-rpms
Red Hat Satellite 6.2	rhel-7-server-satellite-6.2-rpms
Red Hat Enterprise Linux RHSCl	rhel-server-rhsc1-7-rpms

Table 3.1.1.2: Required Channels -Satellite Capsules

3.1.2 Software Versions

3.1.2.1 Operating Systems Required To Run Satellite

Satellite 6.0 runs on RHEL 6.5+ or 7.0+ operating system and Satellite 6.1 requires RHEL 6.6+ or 7.1+. This reference architecture is based on Satellite 6.2.1 and RHEL 7.2 operating system. Only RHEL 6.1+ or 7.0+ clients can be provisioned with Satellite 6.2.

For details on supported client systems, please refer to article <https://access.redhat.com/solutions/1156723>.

3.1.2.2 Satellite Server

Software	Version
postgreSQL	9.2.15-1.el7_2
mongoDB	2.6.11-2.el7sat
puppet	3.8.6
qpid	0.30-11.el7sat
Pulp	2.8.3.3-1.el7sat
dhcp	12:4.2.5-42.el7
tftp	5.2-11.el7
tomcat	7.0.54-2.el7_1
foreman	1.11.0.48-1.el7sat
foreman-proxy	1.11.0.4-1.el7sat
foreman-tasks	0.7.14.6-3.el7sat
apache	2.4.6-40.el7_2.1
pacemaker	1.1.12-22.el7
pcs	0.9.137-13.el7
iscsi	6.2.0.873-33.el7_2.1

Table 3.1.2.1: Satellite Software Versions

3.1.2.3 Satellite Capsule

Software	Version
puppet	3.8.6-2.el7sat
foreman	1.11.0.2-1.el7sat
qpid	0.30-11.el7sat
Pulp	2.8.3.3-1.el7sat
dhcp	12:4.2.5-42.el7
tftp	5.2-11.el7
foreman-proxy	1.11.0.4-1.el7sat
apache	2.4.6-40.el7_2.1

Table 3.1.2.2: Satellite Capsule Software Versions

3.1.3 Security Reference

All the managed hosts being deployed and the installers have SELinux set to enforcing by default. Some of the important ports handled in each host are mentioned below:

3.1.3.1 Satellite Server

Service	Protocol	Port
apache	tcp	80,443,8443
foreman proxy/smart proxy	tcp	8000,9090
qpid	tcp	5671
qrouterd	tcp	5646,5647
puppet	tcp	8140
dns	udp	53
dhcp	udp	67,68
tftp	udp	69
PostgresDB (if external)	tcp	5432
MongoDB (if external)	tcp	27017,28017

Table 3.1.3.1: Satellite Server Network ports

3.1.3.2 Satellite Capsule Server

Service	Protocol	Port
apache	tcp	80,443,8443
foreman proxy/smart proxy	tcp	8000,9090
qpid	tcp	5671
qrouterd	tcp	5647
puppet	tcp	8140
dns	udp	53
dhcp	udp	67,68
tftp	udp	69

Table 3.1.3.2: Capsule Server Network ports

3.1.3.3 Pacemaker Ports

Service	Protocol	Port
pacemaker	tcp	2224,3121
corosync	udp	5404,5405
distributed lock manager	tcp	21064

Table 3.1.3.3: Pacemaker Ports

3.1.4 Server Hardware Configuration

Red Hat Satellite 6 Server is only supported on x86_64 architecture.

The Satellite environment setup for this reference architecture uses three physical servers in an active-passive mode for high availability. Besides an integrated capsule, an external capsule has been setup on a virtual machine that has a different network and domain.

3.1.4.1 Server Resource Specifications

The Table 3.1.4.1: Satellite Server Hardware Specifications lists the hardware specifications for the servers used in this reference architecture.

Hardware	Specifications
Satellite server Dell PowerEdge FX2s FC430 Count – 3 nodes	<u>Processor</u> 2 x Intel(R) Xeon(R) CPU E5-2670 v3 @ 2.30GHz
	<u>Network</u> Fabric A1 - BRCM 10GbE 2P 57810s LOM (4 port aggregated) Fabric A2 - BRCM 10GbE 2P 57810s LOM (4 port aggregated)
	<u>Memory</u> 64 GB
	<u>Disk</u> 2 x 200GB SAS internal disk drives
	<u>Storage</u> (shared, external) 900GB Equallogic iSCSI

Table 3.1.4.1: Satellite Server Hardware Specifications

Hardware	Specifications
RHEV Manager (Virtual environment for capsule and client hosts)	<u>Processor</u> 1 vCPU
	<u>Network</u> 1 vNIC
	<u>Memory</u> 4 GB
	<u>Disk</u> 16GB
RHEV Hypervisors Dell R810 blade server Count – 2 nodes	<u>Processor</u> 2 x Intel(R) Xeon(R) CPU X7560 @ 2.27GHz
	<u>Network</u> 4 x Broadcom Gigabit Ethernet BCM5720 2 x Broadcom NetXtreme II 10 Gb Ethernet BCM57810
	<u>Memory</u> 128GB
	<u>Disk</u> 2 x 146GB SAS internal disk drives
	<u>Storage</u> (external) 3TB Equallogic iSCSI

Table 3.1.4.2: Compute Resource Specifications

Capsule Servers - VM Purpose – Foreman-proxy Count – 3 nodes (cap1a/cap1b/cap1c)	Processor <u>2</u> vCPU
	Network <u>2</u> vNIC
	Memory <u>4</u> GB
	Disk <u>16</u> GB
	Storage (shared, external) <u>100</u> GB Equallogic iSCSI
Capsule Servers - VM Purpose – Content Count – 3 nodes (cap2/cap3/cap4)	Processor <u>2</u> vCPU
	Network <u>1</u> vNIC
	Memory <u>8</u> GB
	Disk <u>300</u> GB
HAProxy Servers – VM Purpose – Load Balancer Count – 3 nodes (hapcap2/cap3/cap4)	Processor <u>1</u> vCPU
	Network <u>2</u> vNIC
	Memory <u>4</u> GB
	Disk <u>16</u> GB

Table 3.1.4.3: Virtual Machines Specifications

3.1.5 Storage Configuration

For this reference architecture, shared storage is provided by iSCSI storage pool. It can be complimented by SAN, or other means of shared storage according to criticality or proximity of the environment. The amount of content stored in the Satellite Server is deterministic of the shared storage size. A shared storage size of 1TB is recommended for a Satellite Server environment inclusive of Satellite Capsule server.

3.1.6 Network Configuration

The following is the list of network addresses used in this reference architecture:

- Satellite servers

Host	Role	Network	Interface	Network Address
sat62-ha.sysmgmt.bos.redhat.com	Virtual Name	Public	p2p2	10.19.2.10
satnode1.sysmgmt.bos.redhat.com	Satellite server 1 (bare metal)	Public	p2p2	10.19.2.11
		Storage (10GB)	em2	172.31.136.11
satnode2.sysmgmt.bos.redhat.com	Satellite server 2 (bare metal)	Public	p2p2	10.19.2.12
		Storage (10GB)	em2	172.31.136.12
satnode3.sysmgmt.bos.redhat.com	Satellite server 3 (bare metal)	Public	p2p2	10.19.2.13
		Storage (10GB)	em2	172.31.136.12

Table 3.1.6.1: Network Information- Satellite

- Satellite Capsule

Host	Role	Network	Interface	Network Address
sat62-ha.sysmgmt.bos.redhat.com	Capsule (integrated)	Public	p2p2	10.19.2.10
		storage	em2	172.31.136.*
cap1.remote.bos.redhat.com	Virtual	Public/virtual	eth0	10.19.3.51
cap1a.remote.bos.redhat.com	Capsule (foreman-proxy)	Public	eth0	10.19.3.48
		Storage	eth1	172.31.3.48
cap1b.remote.bos.redhat.com	Capsule (foreman-proxy)	Public	eth0	10.19.3.49
		Storage	eth1	172.31.3.49
cap1.remote.bos.redhat.com	Capsule (foreman-proxy)	Public/virtual	eth0	10.19.3.51
cap2.remote.bos.redhat.com	Capsule (Content)	Public/virtual	eth0	10.19.3.52
cap3.remote.bos.redhat.com	Capsule (Content)	Public/virtual	eth0	10.19.3.53
cap3.remote.bos.redhat.com	Capsule (Content)	Public/virtual	eth0	10.19.3.53

Table 3.1.6.2: Network Information- Capsules

Note: It is recommended to dedicate a separate network for cluster interconnect. This configuration uses the public network for cluster communications.

4 Installation and Configuration

4.1 Installing Highly Available Satellite Server

The following lists the steps and their sequence followed in this reference architecture to render a functional highly available integrated Satellite Server:

1. Verify Satellite Server installation prerequisites
2. Install operating system and configure servers - **satnode1**, **satnode2** and **satnode3**
 - Network , subscription, time synchronization and connectivity setup
 - Firewall settings
 - iSCSI storage setup
 - Create and mount shared filesystems (exclusive on one node)
3. Install Satellite Server on **satnode1**
4. Install integrated capsule on **satnode1**
5. Verify the Satellite functionality
6. Make a backup of the setup
7. Unmount shared filesystems on **satnode1** and mount on **satnode2**
8. Repeat steps 2 and 3 on **satnode2**
9. Restore the backup content taken from **satnode1** on **satnode2**
10. Verify the Satellite functionality on **satnode2**
11. Unmount shared filesystems on **satnode2** and mount it on **satnode3**
12. Repeat steps 2, 3, 8 and 9 on **satnode3**
13. Stop services and unmount the filesystems
14. Install pacemaker, create the cluster and configure it
15. Verify satellite functionality on each node by failing over satellite services from one node to the other

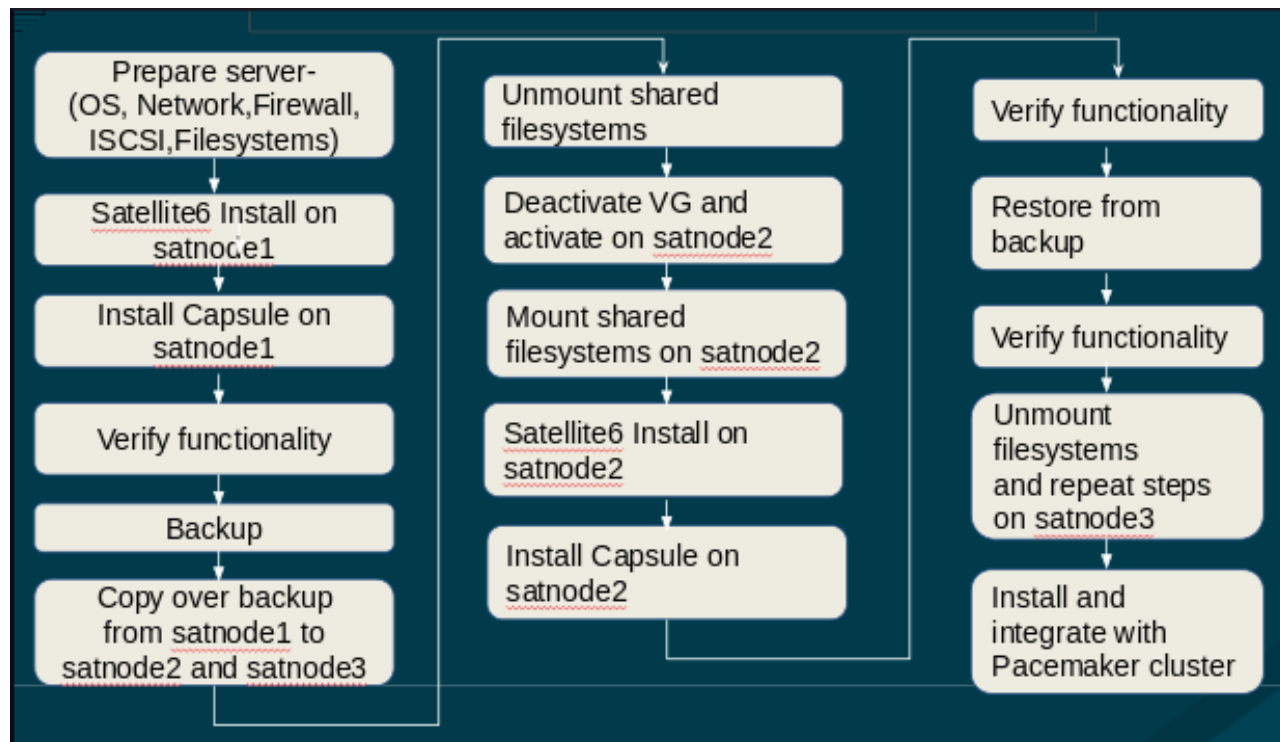


Illustration 3: Satellite Installation Workflow

4.1.1 Satellite Server Installation Prerequisites

Red Hat Satellite requires a networked base system with the following minimum specifications:

- 64-bit architecture.
- The latest version of RHEL 6 Server or 7 Server.
- A minimum of two CPU cores, but four CPU cores are recommended.
- A minimum of 12GB memory but ideally 16GB of memory for each instance of Satellite. Use 4GB of swap space where possible.
- A unique hostname.
- No Java virtual machine installed on the system, remove any if they exist.
- No Puppet RPM files installed on the system.
- No third-party unsupported yum repositories enabled.
- A current Red Hat Network subscription.
- Administrative user (root) access.
- Full forward and reverse DNS resolution using a fully qualified domain name, ensuring that *hostname* and *localhost* resolve correctly.

Update

For a detailed description, please refer to Red Hat Documentation Satellite Server 6.1



Installation Guide

https://access.redhat.com/documentation/en-US/Red_Hat_Satellite/6.1/html/Installation_Guide/sect-Red_Hat_Satellite-Installation_Guide-Prerequisites.html

4.1.2 Install Operating System And Configure Servers

Perform the following steps on each Satellite node.

4.1.2.1 Operating System Setup

Install RHEL 7.2 on three physical servers and configure both public and storage networks.

```
[root@satnode1 /]# cat /etc/sysconfig/network-scripts/ifcfg-em1
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=yes
PEERDNS=no
NAME=em1
DEVICE=em1
ONBOOT=yes
IPADDR=10.19.2.11
PREFIX=24
GATEWAY=10.19.2.254

[root@satnode1 /]# cat /etc/sysconfig/network-scripts/ifcfg-em2
TYPE=Ethernet
DEVICE=em2
ONBOOT=yes
BOOTPROTO=static
IPADDR=172.31.136.11
NETMASK=255.255.0.0
```

Note: Configure unique IP addresses for each node.

Update */etc/hosts* files with IP address of peer nodes:

```
[root@satnode1 config_files]# cat /etc/hosts
127.0.0.1 localhost localhost.localdomain localhost4
localhost4.localdomain4
localhost6.localdomain6
10.19.2.10 sat62-ha.sysmgmt.bos.redhat.com sat62-ha
10.19.2.11 satnode1.sysmgmt.bos.redhat.com satnode1
10.19.2.12 satnode2.sysmgmt.bos.redhat.com satnode2
10.19.2.13 satnode3.sysmgmt.bos.redhat.com satnode3
```

4.1.2.2 Subscription Settings

Register and attach RHEL 7 server and Satellite subscription pools using the customer delivery network (CDN)-customer portal user name/password. Please refer to Subscription management documentation for details

https://access.redhat.com/documentation/en-US/Red_Hat_Subscription_Management/

Register

```
# subscription-manager register
```

List all available subscription pools and identify required pool IDs

```
# subscription-manager list --all --available
```

Attach the pools

```
# subscription-manager attach --pool=RHEL server pool ID
```

```
# subscription-manager attach --pool=Satellite server pool ID
```

Enable required repositories

```
#subscription-manager repos --enable rhel-7-server-rpms --enable
```

```
rhel-7-server-satellite-6.2-rpms --enable rhel-server-rhsc1-7-rpms --enable
```

```
rhel-ha-for-rhel-7-server-rpms
```

4.1.2.3 Shared SSH Keys

Generate an RSA key on **satnode1** and share the public key to **satnode2** and **satnode3**.

Repeat this step on other two nodes as well, to enable password less ssh connectivity among the cluster nodes.

```
[root@satnode1 /]# ssh-keygen -t rsa
[root@satnode1 /]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
7f:a5:18:dd:bf:94:15:79:5e:fe:55:90:58:10:f0:e7
root@satnode1.sysmgmt.bos.redhat.com
The key's randomart image is:
+--[ RSA 2048 ]-----+
|          ..0=0. |
|          .. ... |
|          . .0+ |
|          . + 0= |
|         S . . E = |
|          . 0 0 .= |
|          0 0 00 |
|          . . . |
|          . |
+-----+

```

Copy over the ssh keys to other nodes

```
[root@satnode1 ~]# for HOST in satnode2 satnode3
> do
>   ssh-copy-id $HOST ;
> done
```

```
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to
filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are
prompted now it is to install the new keys
```




```
root@satnode2's password:
Number of key(s) added: 1
Now try logging into the machine, with: "ssh 'satnode2'"
and check to make sure that only the key(s) you wanted were added.
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to
filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are
prompted now it is to install the new keys
root@satnode3's password:
Number of key(s) added: 1
Now try logging into the machine, with: "ssh 'satnode3'"
and check to make sure that only the key(s) you wanted were added.
```

4.1.2.4 Chrony Setting

Install **chrony** to ensure all the three HA nodes have their time synchronized.

```
[root@satnode3 ~]# yum install -y chrony
```

Configure the **/etc/chronyd.conf** file with the appropriate NTP server settings.

```
[root@satnode1 ~]# cat /etc/chronyd.conf
...
# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
#server 0.rhel.pool.ntp.org iburst
#server 1.rhel.pool.ntp.org iburst
#server 2.rhel.pool.ntp.org iburst
#server 3.rhel.pool.ntp.org iburst
server 10.16.xxx.2
server 10.16.xxx.3
[root@satnode1 ~]# systemctl enable chronyd.service; systemctl start
chronyd.service
```

Verify the service

```
[root@satnode1 ~]# chronyc tracking
Reference ID      : 10.16.255.2 (ns1.bos.redhat.com)
Stratum          : 3
Ref time (UTC)   : Tue Jun 21 18:20:33 2016
System time      : 0.000002895 seconds fast of NTP time
Last offset      : +0.000002361 seconds
RMS offset       : 0.000012099 seconds
Frequency        : 1.220 ppm slow
Residual freq    : +0.000 ppm
Skew             : 0.020 ppm
Root delay       : 0.077825 seconds
Root dispersion  : 0.025917 seconds
Update interval  : 1037.2 seconds
Leap status      : Normal
```

Verify nodes are in sync

```
[root@satnode1 ~]# date; ssh satnode2 date ; ssh satnode3 date ; date
Tue Jun 21 14:26:17 EDT 2016
Tue Jun 21 14:26:18 EDT 2016
Tue Jun 21 14:26:18 EDT 2016
Tue Jun 21 14:26:18 EDT 2016
```

4.1.2.5 Firewall Setting

Configure firewalld with the following settings across all nodes:

```
[root@satnode1 ~]# firewall-cmd --permanent \  
--add-port="53/udp" \  
--add-port="53/tcp" \  
--add-port="67/udp" \  
--add-port="68/udp" \  
--add-port="69/udp" \  
--add-port="80/tcp" \  
--add-port="443/tcp" \  
--add-port="5647/tcp" \  
--add-port="8140/tcp" \  
--add-port="2224/tcp" \  
--add-port="3121/tcp" \  
--add-port="5404/udp" \  
--add-port="5405/udp" \  
--add-port="21064/tcp" && firewall-cmd --reload
```

Firewall settings for Pacemaker cluster

```
[root@satnode1 ~]# firewall-cmd --permanent --add-service=high-availability \  
&& firewall-cmd --reload
```

```
[root@satnode1 ~]# iptables -nvl
```

```
...
```

```
Chain IN_public_allow (1 references)
```

target	prot	opt	source	destination	
ACCEPT	tcp	--	0.0.0.0/0	0.0.0.0/0	tcp dpt:22 ctstate NEW
ACCEPT	tcp	--	0.0.0.0/0	0.0.0.0/0	tcp dpt:443 ctstate NEW
ACCEPT	tcp	--	0.0.0.0/0	0.0.0.0/0	tcp dpt:80 ctstate NEW
ACCEPT	tcp	--	0.0.0.0/0	0.0.0.0/0	tcp dpt:8140 ctstate NEW
ACCEPT	tcp	--	0.0.0.0/0	0.0.0.0/0	tcp dpt:5647 ctstate NEW
ACCEPT	udp	--	0.0.0.0/0	0.0.0.0/0	udp dpt:67 ctstate NEW
ACCEPT	udp	--	0.0.0.0/0	0.0.0.0/0	udp dpt:68 ctstate NEW
ACCEPT	tcp	--	0.0.0.0/0	0.0.0.0/0	tcp dpt:53 ctstate NEW
ACCEPT	udp	--	0.0.0.0/0	0.0.0.0/0	udp dpt:69 ctstate NEW
ACCEPT	udp	--	0.0.0.0/0	0.0.0.0/0	udp dpt:53 ctstate NEW
ACCEPT	tcp	--	0.0.0.0/0	0.0.0.0/0	tcp dpt:2224 ctstate NEW
ACCEPT	tcp	--	0.0.0.0/0	0.0.0.0/0	tcp dpt:3121 ctstate NEW
ACCEPT	udp	--	0.0.0.0/0	0.0.0.0/0	udp dpt:5404 ctstate NEW
ACCEPT	udp	--	0.0.0.0/0	0.0.0.0/0	udp dpt:5405 ctstate NEW
ACCEPT	tcp	--	0.0.0.0/0	0.0.0.0/0	tcp dpt:21064 ctstate NEW

4.1.2.6 ISCSI Settings

Install ISCSI packages

```
[root@satnode1 ~]# yum install -y iscsi-initiator-utils
```

Update the initiator details

```
[root@satnode1 ~]# cat /etc/iscsi/initiatorname.iscsi  
InitiatorName=iqn.1994-05.com.redhat:satHA  
Enable and start ISCSI services
```



```
[root@satnode3 ~]# systemctl enable iscsid.service ; systemctl start iscsid.service
```

Discover the iSCSI LUN

```
[root@satnode3 ~]# iscsiadm -m discovery -t st -p 172.31.143.200:3260
172.31.143.200:3260,1
iqn.2001-05.com.equallogic:0-1cb196-13d8a3c08-9597895aa3e559ed-satvol
172.31.143.200:3260,1
```

Add the discovered LUN to the server

```
[root@satnode3 ~]# iscsiadm -m node -T \
iqn.2001-05.com.equallogic:0-1cb196-13d8a3c08-9597895aa3e559ed-satvol \
-p 172.31.143.200:3260 -l
Logging in to [iface: default, target:
iqn.2001-05.com.equallogic:0-1cb196-13d8a3c08-9597895aa3e559ed-satvol,
portal: 172.31.143.200,3260] (multiple)
Login to [iface: default, target:
iqn.2001-05.com.equallogic:0-1cb196-13d8a3c08-9597895aa3e559ed-satvol,
portal: 172.31.143.200,3260] successful.
```

Verify the newly added 900GB lun

```
[root@satnode1 ~]# lsblk
NAME                MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda                  8:0    0 186.3G  0 disk
├─sda1                8:1    0   500M  0 part /boot
├─sda2                8:2    0 185.8G  0 part
│   └─rhel-swap       253:0    0   18.6G  0 lvm  [SWAP]
│   └─rhel-root       253:1    0    50G   0 lvm  /
│   └─rhel-home       253:2    0 117.1G  0 lvm  /home
sdb                  8:16   0 186.3G  0 disk
└─sdb1                8:17   0 186.3G  0 part
sdc                  8:32   0   900G  0 disk
```

4.1.2.7 Create Shared Filesystems For HA LVM

Create physical volume, Volume group and logical volumes:

Note: The steps in Create Shared Filesystems For HA LVM were created using create_filesystem.sh script. Please refer to 106. Please size the filesystems according to the Satellite environment

```
[root@satnode1 ~]# pvcreate -f /dev/sdc
Physical volume "/dev/sdc" successfully created
```

```
[root@satnode1 ~]# vgcreate sat_vg /dev/sdc
Volume group "sat_vg" successfully created
```

```
[root@satnode1 ~]# lvcreate -L 500G -n lv_pulp sat_vg
Logical volume "lv_pulp" created.
```

```
[root@satnode1 ~]# lvcreate -L 10G -n lv_foreman sat_vg
Logical volume "lv_foreman" created.
```

```
[root@satnode1 ~]# lvcreate -L 10G -n lv_puppet sat_vg
Logical volume "lv_puppet" created.
```

```
[root@satnode1 ~]# lvcreate -L 10G -n lv_wwwpulp sat_vg
Logical volume "lv_wwwpulp" created.

[root@satnode1 ~]# lvcreate -L 10G -n lv_candlepin sat_vg
Logical volume "lv_candlepin" created.

[root@satnode1 ~]# lvcreate -L 100G -n lv_mongodb sat_vg
Logical volume "lv_mongodb" created.

[root@satnode1 ~]# lvcreate -L 10G -n lv_psqldata sat_vg
Logical volume "lv_psqldata" created.

[root@satnode1 ~]# lvcreate -L 10G -n lv_puppetenv sat_vg
Logical volume "lv_puppetenv" created.

[root@satnode1 ~]# lvcreate -L 10G -n lv_tftpboot sat_vg
Logical volume "lv_tftpboot" created.

[root@satnode1 ~]# lvcreate -L 10G -n lv_dhcp sat_vg
Logical volume "lv_dhcp" created.

[root@satnode1 ~]# lvcreate -L 10G -n lv_named sat_vg
Logical volume "lv_named" created.
```

Note: Size the filesystems according to the size of the Satellite environment.

Verify the logical volumes:

```
[root@satnode1 ~]# lsblk /dev/sdc
NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sdc                                  8:32   0  900G  0 disk
├─sat_vg-lv_pulp                      253:3   0   500G  0 lvm
├─sat_vg-lv_foreman                   253:4   0    10G  0 lvm
├─sat_vg-lv_puppet                     253:5   0    10G  0 lvm
├─sat_vg-lv_wwwpulp                   253:6   0    10G  0 lvm
├─sat_vg-lv_candlepin                 253:8   0    10G  0 lvm
├─sat_vg-lv_mongodb                   253:9   0   100G  0 lvm
├─sat_vg-lv_psqldata                  253:10  0    10G  0 lvm
├─sat_vg-lv_puppetenv                 253:11  0    10G  0 lvm
├─sat_vg-lv_tftpboot                  253:12  0    10G  0 lvm
├─sat_vg-lv_dhcp                      253:13  0    10G  0 lvm
└─sat_vg-lv_named                     253:14  0    10G  0 lvm
```

Create the filesystems:

```
[root@satnode1 ~]# for i in $(lvs | grep sat_vg | awk '{print $1}'); do
mkfs.xfs /dev/sat_vg/$i; done
```

Note: "XFS" is the default filesystem in RHEL 7 and is the preferred choice. However it can be complimented with "ext4" if there is a requirement.

Stop and disable "named" service on all the nodes.

```
[root@satnode1 ~]# for HOST in satnode1 satnode2 satnode3; do systemctl
stop named.service; sysmtemctl disable named.service; done
```



Move the “/var/named” contents

```
[root@satnode1 ~]# for HOST in satnode1 satnode2 satnode3; do ssh $HOST  
mv /var/named /var/named.orig; done
```

Create mount point directories:

```
[root@satnode1 ~]# mkdir -p /var/lib/pulp  
[root@satnode1 ~]# mkdir -p /var/lib/candlepin  
[root@satnode1 ~]# mkdir -p /var/lib/foreman  
[root@satnode1 ~]# mkdir -p /var/lib/puppet  
[root@satnode1 ~]# mkdir -p /var/www/pulp  
[root@satnode1 ~]# mkdir -p /var/lib/mongodb  
[root@satnode1 ~]# mkdir -p /var/lib/pgsql  
[root@satnode1 ~]# mkdir -p /etc/puppet/environments  
[root@satnode1 ~]# mkdir -p /var/lib/tftpboot  
[root@satnode1 ~]# mkdir -p /var/lib/dhcpd  
[root@satnode1 ~]# mkdir -p /var/lib/named
```

Mount filesystems

```
[root@satnode1 /]# mount /dev/mapper/sat_vg-lv_pulp /var/lib/pulp  
[root@satnode1 /]# mount /dev/mapper/sat_vg-lv_candlepin /var/lib/candlepin  
[root@satnode1 /]# mount /dev/mapper/sat_vg-lv_foreman /var/lib/foreman  
[root@satnode1 /]# mount /dev/mapper/sat_vg-lv_puppet /var/lib/puppet  
[root@satnode1 /]# mount /dev/mapper/sat_vg-lv_wwwpulp /var/www/pulp  
[root@satnode1 /]# mount /dev/mapper/sat_vg-lv_mongodb /var/lib/mongodb  
[root@satnode1 /]# mount /dev/mapper/sat_vg-lv_psqldata /var/lib/pgsql  
[root@satnode1 /]# mount /dev/mapper/sat_vg-lv_puppetenv \  
/etc/puppet/environments  
[root@satnode1 /]# mount /dev/mapper/sat_vg-lv_tftpboot /var/lib/tftpboot  
[root@satnode1 /]# mount /dev/mapper/sat_vg-lv_dhcp /var/lib/dhcpd  
[root@satnode1 /]# mount /dev/mapper/sat_vg-lv_named /var/lib/named
```

4.1.3 Install Satellite Server

Perform the following steps to install Satellite Server across each node.

4.1.3.1 Installing Satellite On The First Node - *satnode1*

Note: Perform the following steps on the first node - *satnode1*

Update the server to latest versions:

```
[root@satnode1 /]# yum update -y
```

Download katello packages

```
[root@satnode1 /]# yum install satellite -y
```

Hostname and IP address change:

For the satellite to be associated with a virtual hostname and IP address that can switch over among the HA nodes, the following steps must be followed:

1. Change the hostname to `sat62-HA.sysmgmt.bos.redhat.com`

```
[root@satnode1 /]# cat /etc/hostname
sat62-ha.sysmgmt.bos.redhat.com
```

2. Change the ip address in `/etc/sysconfig/network-scripts/ifcfg-em1` file and restart the network

```
[root@satnode1 /]# cat /etc/sysconfig/network-scripts/ifcfg-em1
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=yes
PEERDNS=no
NAME=em1
DEVICE=em1
ONBOOT=yes
IPADDR=10.19.2.10
PREFIX=24
GATEWAY=10.19.2.254
```

Install katello:

```
[root@satnode1 /]# satellite-installer --scenario satellite
```

```
Installing           Done [100%]
[.....]
Success!
* Katello is running at https://sat62-ha.sysmgmt.bos.redhat.com
  Initial credentials are admin / xxxxxxxxxx
* Capsule is running at https://sat62-ha.sysmgmt.bos.redhat.com:9090
* To install additional capsule on separate machine continue by running:
  capsule-certs-generate --capsule-fqdn "$CAPSULE" --certs-tar "~/
$CAPSULE-certs.tar"

The full log is at /var/log/satellite-installer/katello.log
```

4.1.3.2 Install Integrated Capsule On First Node - *satnode1*

```
[root@satnode1 /]# satellite-installer --scenario capsule \
--foreman-proxy-dns-interface "eth0" \
--foreman-proxy-dns "true" \
--foreman-proxy-dns-zone "sysmgmt.bos.redhat.com" \
--foreman-proxy-dns-forwarders "10.19.2.10" \
--foreman-proxy-dns-reverse "2.19.10.in-addr.arpa" \
--foreman-proxy-dhcp "true" \
--foreman-proxy-dhcp-interface "p2p2" \
--foreman-proxy-dhcp-range "10.19.2.101 10.19.2.130" \
--foreman-proxy-dhcp-gateway "10.19.2.254" \
--foreman-proxy-dhcp-nameservers "10.19.2.10" \
--foreman-proxy-tftp "true" \
--foreman-proxy-tftp-servername "10.19.2.10"
--capsule-puppet "true" \
--capsule-puppetca "true"

Installing          Done          [100%]
[.....]
Success!
* Katello is running at https://sat62-ha.sysmgmt.bos.redhat.com
  Initial credentials are admin / xxxxxxxxxx
* Capsule is running at https://sat62-ha.sysmgmt.bos.redhat.com:9090
* To install additional capsule on separate machine continue by running:"
  capsule-certs-generate --capsule-fqdn "$CAPSULE" --certs-tar "~/
$CAPSULE-certs.tar"
The full log is at /var/log/satellite-installer/katello.log
```

The table Table 4.1.3.1: Capsule Parameters describes the “satellite-installer” command options for Capsule Server configuration and the values used in the reference architecture:

Option	Description	Va
--foreman-proxy-dns	Enable DNS proxy capability	true
--foreman-proxy-dns-interface	Which interface named should listen on	em1
--foreman-proxy-dns-zone	The Forward DNS zone that the Satellite will host	sysmgmt.bos.redhat.com
--foreman-proxy-dns-forwarders	The DNS server that unknown queries are forwarded to	10.19.2.10
--foreman-proxy-dns-reverse	The Reverse DNS zone the Satellite hosts. This is usually the first three octets of the IP address (172.17.13) reversed , and appended with ".in-addr.arpa".	2.19.10.in-addr.arpa
--foreman-proxy-dhcp	Enable DHCP proxy capability	true
--foreman-proxy-dhcp-interface	The interface that DHCP listens on	em1
--foreman-proxy-dhcp-range	The range of IP addresses to issue to clients.	10.19.2.101 – 10.19.2.10
--foreman-proxy-dhcp-gateway	The default gateway IP to issue to clients.	10.19.2.254
--foreman-proxy-dhcp-nameservers	The host that the clients should use for name resolution. This should be configured with the Satellite's IP in this deployment model.	10.19.2.10
--foreman-proxy-tftp	Enable TFTP proxy capability. This is needed to PXE boot the clients.	true
--foreman-proxy-tftp-servername	Sets the TFTP host name. Set this to match the server's host name	\$(hostname)
--foreman-proxy-puppet	Enable the Puppet Master.	true
--foreman-proxy-puppetca	Enable the Puppet CA.	true

Table 4.1.3.1: Capsule Parameters

4.1.3.3 Verify The Satellite Functionality

Invoke a browser, point the URL and login with information as mentioned in the output of **katello-installer** script.

The password for the user (admin by default) can also be retrieved from the installer answer file generated during install:

```
[root@satnode1 katello-installer]# grep admin_password \
/etc/katello-installer/answers.katello-installer.yaml | grep -v pulp
admin_password: sybh5aCkwJCxKHGp
```

Verify the presence for the integrated capsule for **sat62-ha.sysmgmt.bos.redhat.com** by navigating to **Installer-► Capsules** within the Satellite web console.

Capsule added to the Satellite Server environment.

Capsules

Filter ...

Name	Locations	Organizations	Features	Status	Actions
sat62-ha.sysmgmt.bos.redhat.com	Default Location	Default Organization and refarch	Pulp, Puppet, Puppet CA, Dynflow, and SSH		<input type="button" value="Edit"/>

Illustration 4: Sat62-ha.sysmgmt.bos.redhat.com Capsule Server

Create a test user (this will help confirm if the backup of this environment gets successfully restored to secondary nodes):

Administrator-► Users-► New User

Create a new user test1

Users

Filter ...

Username	First name	Surname	Email address	Administrator	Last login time	Authorized by	
admin	Admin	User	root@sysmgmt.bos.redhat.com		13 minutes ago	INTERNAL	<input type="button" value="Delete"/>
test1	Test user	satnode1	test@redhat.com			INTERNAL	<input type="button" value="Delete"/>

Illustration 5: New User- test1

4.1.3.4 Backup Of Satellite Environment

Make a backup of the Satellite environment. While the services are running, run the backup script - **satellite_backup.sh**

```
[root@satnode1 /]# /root/scripts/satellite-backup.sh
```

Note: Please refer to script 109

This will create a directory:

```
[root@satnode1 /]# ll /var/tmp/rhs6backup/2016-05-09-150342
total 176964
-rw-r--r--. 1 postgres postgres 289352 May 9 15:09 candlepin.dump
-rw-r-----. 1 root root 77639620 May 9 15:03 config_files.tar.gz
-rw-r--r--. 1 postgres postgres 1452229 May 9 15:09 foreman.dump
-rw-r-----. 1 root root 74353151 May 9 15:07 mongo_data.tar.gz
drwxr-x---. 4 root root 37 May 9 15:09 mongo_dump
-rw-r-----. 1 root root 27466185 May 9 15:08 pgsqldata.tar.gz
drwxr-xr-x. 10 apache apache 4096 May 6 16:25 pulp
```

Note: It is safe to ignore the following error: 20151107140516 Return from command '/bin/tar --selinux -czvf /var/tmp/rhs6backup/2016-05-07-140237/pgsqldata.tar.gz /var/lib/pgsqldata >> /var/log/satellite6-backup.log 2>&' '1' != '0' . Aborting script : 'exit 1' ' as long as /var/log/satellite6-backup.log shows success of the backup.

Copy over the backup to *satnode2*

```
[root@satnode1 /]# rsync --partial --progress --delete -avz1 \  
/var/tmp/rhs6backup/2015-05-09-150342 \  
root@satnode2.sysmgmt.bos.redhat.com:/var/tmp/rhs6backup
```

Repeat for node *satnode3*

4.1.3.5 Cleanup On *satnode1*

Note: The steps in 26 were performed using the script 107

Stop Satellite services on *satnode1*

```
[root@satnode1 /]#systemctl stop dhcpd.service  
[root@satnode1 /]#systemctl stop puppet.service  
[root@satnode1 /]#systemctl stop postgresql.service  
[root@satnode1 /]#katello-service stop
```

Unmount the filesystems on *satnode1*

```
[root@satnode1 /]# umount /dev/mapper/sat_vg-lv_pulp /var/lib/pulp  
[root@satnode1 /]# umount /dev/mapper/sat_vg-lv_candlepin /var/lib/candlepin  
[root@satnode1 /]# umount /dev/mapper/sat_vg-lv_foreman /var/lib/foreman  
[root@satnode1 /]# umount /dev/mapper/sat_vg-lv_puppet /var/lib/puppet  
[root@satnode1 /]# umount /dev/mapper/sat_vg-lv_wwwpulp /var/www/pulp  
[root@satnode1 /]# umount /dev/mapper/sat_vg-lv_mongodb /var/lib/mongodb  
[root@satnode1 /]# umount /dev/mapper/sat_vg-lv_psqldata /var/lib/pgsql  
[root@satnode1 /]# umount /dev/mapper/sat_vg-lv_puppetenv \  
/etc/puppet/environments  
[root@satnode1 /]# umount /dev/mapper/sat_vg-lv_tftpboot /var/lib/tftpboot  
[root@satnode1 /]# umount /dev/mapper/sat_vg-lv_dhcp /var/lib/dhcpd
```

Deactivate the volume group on *satnode1*

```
[root@satnode1 /]# vgchange -a n sat_vg
```

Restore the hostname back to *satnode1.bos.redhat.com* and IP address to 10.19.2.11



4.1.3.6 Install Satellite On Second Node – *satnode2*

Change the hostname of *satnode2* to *sat62-ha.sysmgmt.bos.redhat.com* and IP address to 10.19.2.10 as mentioned in step 4.1.3.1

Verify the hostname:

```
[root@satnode2 ~]# hostname
sat62-ha.sysmgmt.bos.redhat.com
```

Verify the IP address:

```
[root@satnode2 ~]# ip a ddr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: p2p2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen
1000
    link/ether 44:a8:42:73:a1:af brd ff:ff:ff:ff:ff:ff
    inet 10.19.2.10/24 brd 10.19.2.255 scope global em1
        valid_lft forever preferred_lft forever
    inet6 2620:52:0:1302:46a8:42ff:fe73:a1af/64 scope global dynamic
        valid_lft 2591850sec preferred_lft 604650sec
    inet6 fe80::46a8:42ff:fe73:a1af/64 scope link
        valid_lft forever preferred_lft forever
3: em2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen
1000
    link/ether 44:a8:42:73:a1:b2 brd ff:ff:ff:ff:ff:ff
    inet 172.31.136.12/16 brd 172.31.255.255 scope global em2
        valid_lft forever preferred_lft forever
    inet6 fe80::46a8:42ff:fe73:a1b2/64 scope link
```

The steps in 27 were performed using script 108

Activate the volume group on *satnode2*

```
[root@satnode2 /]# vgchange -a y sat_vg
```

Mount the filesystems on *satnode2*

```
[root@satnode2 /]# mount /dev/mapper/sat_vg-lv_pulp /var/lib/pulp
[root@satnode2 /]# mount /dev/mapper/sat_vg-lv_candlepin /var/lib/candlepin
[root@satnode2 /]# mount /dev/mapper/sat_vg-lv_foreman /var/lib/foreman
[root@satnode2 /]# mount /dev/mapper/sat_vg-lv_puppet /var/lib/puppet
[root@satnode2 /]# mount /dev/mapper/sat_vg-lv_wwwpulp /var/www/pulp
[root@satnode2 /]# mount /dev/mapper/sat_vg-lv_mongodb /var/lib/mongodb
[root@satnode2 /]# mount /dev/mapper/sat_vg-lv_psqldata /var/lib/pgsql
[root@satnode2 /]# mount /dev/mapper/sat_vg-lv_puppetenv \
```

```
/etc/puppet/environments
```

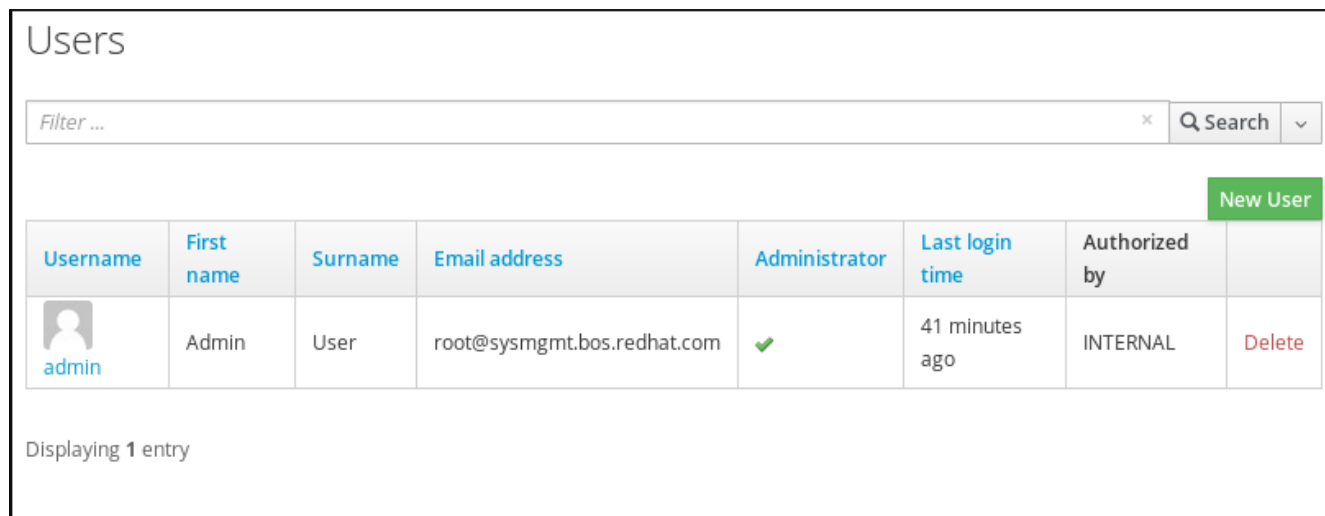
```
[root@satnode2 /]# mount /dev/mapper/sat_vg-lv_tftpboot /var/lib/tftpboot
```


```
[root@satnode2 /]# mount /dev/mapper/sat_vg-lv_dhcp /var/lib/dhcpd
```

Install Satellite and Capsule on **satnode2**. Refer to steps 22,23 and Verify The Satellite Functionality.

Verify the list of users:

Administrator-► Users



Username	First name	Surname	Email address	Administrator	Last login time	Authorized by	
 admin	Admin	User	root@sysmgmt.bos.redhat.com	✓	41 minutes ago	INTERNAL	Delete

Displaying 1 entry

Illustration 6: Users List

4.1.3.7 Restore backed up content from **satnode1** on **satnode2**

Restore the contents while the satellite is running on **satnode2**

```
[root@satnode2 /]# /root/scripts/satellite-restore.sh \  
/var/tmp/rhs6backup/2015-05-09-150342
```

Note: Please refer to script 111

4.1.3.8 Verify Satellite Functionality

Invoke a browser to **satnode2**, and login with information mentioned in the output of **katello-installer** script. Verify the presence of capsule **sat62-ha.sysmgmt.bos.redhat.com** by selecting **Installer-► Capsules**

Verify if the user test1 exists (carried over from **satnode1** through backup)

Administrator-► Users

Users

Filter ... Q Search



Username	First name	Surname	Email address	Administrator	Last login time	Authorized by	
 admin	Admin	User	root@sysmgmt.bos.redhat.com	✓	13 minutes ago	INTERNAL	Delete
 test1	Test user	satnode1	test@redhat.com			INTERNAL	Delete

Illustration 7: Users List

4.1.3.9 Installing Satellite on third node – *satnode3*

Repeat steps on *satnode3* as mentioned in step Cleanup On satnode1 to 28

4.1.3.10 Add Trusted Hosts

Add all the hostnames as trusted hosts in the foreman-proxy *settings.yml* file:

```
[root@satnode1 /]# cat /etc/foreman-proxy/settings.yml
### File managed with puppet ###
## Module:          'foreman_proxy'

:settings_directory: /etc/foreman-proxy/settings.d

# SSL Setup

# if enabled, all communication would be verified via SSL
# NOTE that both certificates need to be signed by the same CA in order for
this to work
# see http://theforeman.org/projects/smart-proxy/wiki/SSL for more
information
:ssl_ca_file: /etc/foreman-proxy/ssl_ca.pem
:ssl_certificate: /etc/foreman-proxy/ssl_cert.pem
:ssl_private_key: /etc/foreman-proxy/ssl_key.pem

# the hosts which the proxy accepts connections from
# commenting the following lines would mean every verified SSL connection
allowed
:trusted_hosts:
- sat62-ha.sysmgmt.bos.redhat.com
- satnode1.sysmgmt.bos.redhat.com
- satnode2.sysmgmt.bos.redhat.com
- satnode3.sysmgmt.bos.redhat.com
```

4.1.4 Pacemaker Install

4.1.4.1 Install Pacemaker Packages

Install the packages on all three nodes:

```
[root@satnode1/2/3 ~]# yum install -y fence-agents resource-agents pcs ccs pacemaker
```

Enable and start *pcsd* services:

```
[root@satnode1/2/3 ~]# systemctl enable pcsd.service
```

```
[root@satnode1/2/3 ~]# systemctl start pcsd.service
```

Set password for cluster:

```
[root@satnode1/2/3 ~]# passwd hacluster
New password: cluster_passwd
Retype new password: cluster_passwd
```

Perform the following steps on *satnode1*:

Verify authentication among the nodes for cluster communication:

```
[root@satnode1 ~]# pcs cluster auth satnode1
```

```
[root@satnode1 ~]# pcs cluster auth satnode1
```

```
Username: hacluster
Password: cluster_passwd
satnode1: Authorized
```

```
[root@satnode1 ~]# pcs cluster auth satnode2
```

```
Username: hacluster
Password: cluster_passwd
satnode2: Authorized
```

```
[root@satnode1 ~]# pcs cluster auth satnode3
```

```
Username: hacluster
Password: cluster_passwd
satnode3: Authorized
```

Enter *hacluster* passwd. Repeat the steps on *satnode2* and *satnode3*. Once all the nodes are communicating, the output must be as follows:

```
[root@satnode1 ~]# pcs cluster auth satnode1 satnode2 satnode3
satnode1: Already authorized
satnode2: Already authorized
satnode3: Already authorized
```

4.1.4.2 Create The Cluster

```
[root@satnode1 ~]# pcs cluster setup --start --name satnode1 satnode2 \
satnode3
```

```
Shutting down pacemaker/corosync services...
Redirecting to /bin/systemctl stop pacemaker.service
Redirecting to /bin/systemctl stop corosync.service
```



```
Killing any remaining services...
Removing all cluster configuration files...
satnode2: Succeeded
satnode3: Succeeded
Starting cluster on nodes: satnode2, satnode3...
```

Enable cluster to be started up in persistent mode upon reboot:

```
[root@satnode1 ~]# pcs cluster enable -all
```

Verify cluster status

```
[root@satnode1 /]# pcs status
Cluster name: SAT62-HA
Last updated: Sat Nov 14 14:18:18 2015      Last change: Thu Nov 12
10:18:16 2015 by root via crm_attribute on satnode1
Stack: corosync
Current DC: satnode2 (version 1.1.13-a14efad) - partition with quorum
3 nodes configured
0 Resources configured
Online: [ satnode1 satnode2 satnode3 ]
Full list of resources:
PCSD Status:
  satnode1: Online
  satnode2: Online
  satnode3: Online
Daemon Status:
  corosync: active/enabled
  pacemaker: active/enabled
  pcsd: active/enabled
```

```
[root@satnode1 scripts]# pcs property
Cluster Properties:
  cluster-infrastructure: corosync
  cluster-name: SAT62-HA
  dc-version: 1.1.13-a14efad
  have-watchdog: false
```

4.1.4.3 Create Fencing

```
[root@satnode1 /]## pcs stonith create satnode1-fence fence_ipmilan params \
login="root" passwd="ipmi_passwd-" ipaddr="10.19.143.24" \ lanplus="1"
verbose="" pcmk_host_list=satnode1
```

```
[root@satnode1 /]## pcs stonith create satnode2-fence fence_ipmilan params \
login="root" passwd="ipmi_passwd-" ipaddr="10.19.143.23" \ lanplus="1"
verbose="" pcmk_host_list=satnode2
```

```
[root@satnode1 /]# pcs stonith create satnode3-fence fence_ipmilan params \
login="root" passwd="ipmi_passwd-" ipaddr="10.19.143.22" \ lanplus="1"
verbose="" pcmk_host_list=satnode3
```

```
[root@satnode1 /]# pcs stonith show --all
satnode1-fence (stonith:fence_ipmilan): Started
satnode2-fence (stonith:fence_ipmilan): Started
satnode3-fence (stonith:fence_ipmilan): Started
[root@satnode1 ~]# pcs stonith show satnode1-fence
```

```
Resource: satnode1-fence (class=stonith type=fence_ipmilan)
Attributes: login=root passwd=ipmi_passwd action=reboot
ipaddr=10.19.143.24 lanplus=1 verbose= pcmk_host_list=satnode1
Operations: monitor interval=60s (satnode1-fence-monitor-interval-60s)
```

4.1.4.4 Exclusive Activation Of Shared Volume Group

After ensuring the Satellite services are down, and the filesystems have been unmounted on all the nodes, perform the following on all the nodes:

```
[root@satnode1/2/3 ~]# vgchange -a n sat_vg
```

```
[root@satnode1/2/3 ~]# lvchange -aey sat_vg
```

Ensure that the shared volume group does not get activated at boot time and is activated only by pacemaker. This is achieved by making an entry of all volume groups that must be auto-activated at boot time in the `/etc/lvm/lvm.conf` file.

In this environment, only “**rhel**” volume group exists and must be auto-activated.

```
[root@satnode1/2/3 ~]# vgs | grep -v sat_vg
```

```
VG #PV #LV #SN Attr VSize VFree
rhel 1 3 0 wz--n- 185.82g 60.00m
```

Edit the file to add “**rhel**” in the “`volume_list`” variable.

```
[root@satnode1/2/3]# cat /etc/lvm/lvm.conf | grep "volume_list =" | grep -v "#"
volume_list = [ "rhel" ]
```

Ensure this change is reflected in the kernel and reboot the server:

```
[root@satnode1/2/3]# dracut -H -f /boot/initramfs-$(uname -r).img $(uname -r)
```

```
[root@satnode1/2/3]# reboot
```

4.1.4.5 Creating Pacemaker Resources

Prework:

The resources must be created from any one node. **satnode1** in this case.

It is ideal to disable both second and third node before creating resources. This step helps manually ensure that all the services are started up on the first node, instead of being scattered across different nodes. Resource constraints created in subsequent steps, will handle this automatically.

```
[root@satnode1 scripts]# pcs cluster standby satnode2
```

```
[root@satnode1 scripts]# pcs cluster standby satnode3
```

```
[root@satnode1 scripts]# pcs status
```

```
Cluster name: SAT62-HA
```

```
Stack: corosync
```

```
Current DC: satnode2 (version 1.1.13-a14efad) - partition with quorum
```

```
3 nodes and 30 resources configured
```

```
Node satnode2: standby
```

```
Node satnode3: standby
```


Online: [satnode1]

Create Resources

1. Virtual IP resource:

```
[root@satnode1 ~]# pcs resource create VirtualIP IPAddr2 ip=10.19.2.10  
cidr_netmask=24
```

2. Volume group:

```
[root@satnode1 /]# pcs resource create sat_vg LVM volgrpname=sat_vg  
exclusive=true
```

3. Filesystems:

```
[root@satnode1 /]# pcs resource create fs_pulp Filesystem \  
device="/dev/sat_vg/lv_pulp" directory="/var/lib/pulp" fstype="xfs"
```

```
[root@satnode1 /]# pcs resource create fs_candlepin Filesystem \  
device="/dev/sat_vg/lv_candlepin" directory="/var/lib/candlepin" \  
fstype="xfs"
```

```
[root@satnode1 /]# pcs resource create fs_foreman Filesystem \  
device="/dev/sat_vg/lv_foreman" directory="/var/lib/foreman" \  
fstype="xfs"
```

```
[root@satnode1 /]# pcs resource create fs_puppet Filesystem \  
device="/dev/sat_vg/lv_puppet" directory="/var/lib/puppet" \  
fstype="xfs"
```

```
[root@satnode1 /]# pcs resource create fs_wwwpulp Filesystem \  
device="/dev/sat_vg/lv_wwwpulp" directory="/var/www/pulp" \  
fstype="xfs"
```

```
[root@satnode1 /]# pcs resource create fs_mongodb Filesystem \  
device="/dev/sat_vg/lv_mongodb" directory="/var/lib/mongodb" \  
fstype="xfs" --group sat-group
```

```
[root@satnode1 /]# pcs resource create fs_pgsql Filesystem \  
device="/dev/sat_vg/lv_psqldata" directory="/var/lib/pgsql" \  
fstype="xfs"
```

```
[root@satnode1 /]# pcs resource create fs_puppetenv Filesystem \  
device="/dev/sat_vg/lv_puppetenv" directory="/etc/puppet/environments" \  
fstype="xfs"
```

```
[root@satnode1 /]# pcs resource create fs_tftpboot Filesystem \  
device="/dev/sat_vg/lv_tftpboot" directory="/var/lib/tftpboot" \  
fstype="xfs"
```

```
[root@satnode1 /]# pcs resource create fs_dhcp Filesystem \  
device="/dev/sat_vg/lv_dhcp" directory="/var/lib/dhcpd" fstype="xfs"
```

```
[root@satnode1 /]# pcs resource create fs_named Filesystem  
device="/dev/sat_vg/lv_named" directory="/var/named" fstype="xfs"
```

4. Services:

```
[root@satnode1 /]# pcs resource create rs_dhcp systemd:dhcpd op monitor\  
interval=10s op start interval=0s timeout=100s op stop interval=0s \  

```

```
timeout=100s
[root@satnode1 /]# pcs resource create rs_named systemd:named op monitor
interval=10s op start interval=0s timeout=100s op stop interval=0s
timeout=100s

[root@satnode1 /]# pcs resource create rs_puppet systemd:puppet op\
monitor interval=10s op start interval=0s timeout=100s op stop \
interval=0s timeout=100s

[root@satnode1 /]# pcs resource create rs_pgsql systemd:postgresql op\
monitor interval=10s op start interval=0s timeout=100s op stop \
interval=0s timeout=100s

[root@satnode1 /]# pcs resource create rs_mongodb systemd:mongod op\
monitor interval=10s op start interval=0s timeout=100s op stop \
interval=0s timeout=100s

[root@satnode1 /]# pcs resource create rs_qpidd systemd:qpidd op \
monitor interval=10s op start interval=0s timeout=100s op stop \
interval=0s timeout=100s

[root@satnode1 /]# pcs resource create rs_qdrouterd systemd:qdrouterd \
op monitor interval=10s op start interval=0s timeout=100s op stop \
interval=0s timeout=100s

[root@satnode1 /]# pcs resource create rs_tomcat systemd:tomcat op \
monitor interval=10s op start interval=0s timeout=100s op stop \
interval=0s timeout=100s

[root@satnode1 /]# pcs resource create rs_pulp_workers \
systemd:pulp_workers op monitor interval=10s op start interval=0s \
timeout=100s op stop interval=0s timeout=100s

[root@satnode1 /]# pcs resource create rs_foreman-proxy \
systemd:foreman-proxy op monitor interval=10s

[root@satnode1 /]# pcs resource create rs_pulp_resource_manager \
systemd:pulp_resource_manager op monitor interval=10s op start \
interval=0s timeout=100s op stop interval=0s timeout=100s

[root@satnode1 /]# pcs resource create rs_pulp_celerybeat \
systemd:pulp_celerybeat op monitor interval=10s op start interval=0s \
timeout=100s op stop interval=0s timeout=100s

[root@satnode1 /]# pcs resource create rs_httpd systemd:httpd op \
monitor interval=10s

[root@satnode1 /]# pcs resource create rs_foreman-tasks \
systemd:foreman-tasks op monitor interval=10s timeout=30s op start \
interval=0s timeout=100s op stop interval=0s timeout=100s
```

Note: systemd has a timeout of 90 seconds for starting a service. Hence resources of type systemd, have been set to 100 seconds timeout, where if systemd fails. Pacemaker will handle it beyond that.

For additional details on creating pacemaker resources, please refer to Pacemaker documentation ¹.

1 https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html-single/Configuring_t

5. Resource Constraints:

Resource constraints help decide on which node the resources start on and in what order. This helps control collocation of like resources to be started in the same node. This reference environment requires that all resources other than fencing resources, must be located on a single node. This is achieved by the following setting:

Service resource creation :

- Colocation Constraints

```
[root@satnode1/# pcs constraint colocation add sat_vg with \
VirtualIP

[root@satnode1/# pcs constraint colocation set fs_dhcp fs_named \
fs_puppet fs_puppetenv fs_pgsql fs_mongodb fs_pulp fs_wwwpulp \
fs_foreman fs_candlepin fs_tftpboot \ sequential=false set sat_vg

[root@satnode1/# pcs constraint colocation add rs_dhcp with \
fs_dhcp

[root@satnode1/# pcs constraint colocation add rs_named with
fs_named

[root@satnode1/# pcs constraint colocation add rs_pgsql with \
fs_pgsql

[root@satnode1/# pcs constraint colocation add rs_mongodb with \
fs_mongodb

[root@satnode1/# pcs constraint colocation add rs_foreman-proxy \
with fs_foreman

[root@satnode1/# pcs constraint colocation set fs_puppet \
fs_puppetenv rs_puppet

[root@satnode1/# pcs constraint colocation set fs_pulp \
fs_wwwpulp rs_pulp_workers

[root@satnode1/# pcs constraint colocation set VirtualIP \
rs_qpidd rs_qdrouterd

[root@satnode1/# pcs constraint colocation add rs_tomcat with \
VirtualIP
[root@satnode1/# pcs constraint colocation add rs_httpd with \
VirtualIP

[root@satnode1/# pcs constraint colocation set fs_pulp \
fs_wwwpulp \ rs_pulp_resource_manager

[root@satnode1/# pcs constraint colocation set fs_pulp \
fs_wwwpulp \ rs_pulp_celerybeat

[root@satnode1/# pcs constraint colocation add rs_foreman-tasks \
with \ fs_foreman
```

- Order constraints to start services after the filesystems

```
[root@satnode1/# pcs constraint order set sat_vg set fs_dhcp \
fs_named fs_puppet fs_puppetenv fs_pgsql fs_mongodb fs_pulp
fs_wwwpulp fs_foreman fs_candlepin fs_tftpboot \
sequential=false

[root@satnode1/# pcs constraint order VirtualIP then sat_vg
[root@satnode1/# pcs constraint order set sat_vg fs_dhcp rs_dhcp
[root@satnode1/# pcs constraint order set sat_vg fs_puppet \
fs_puppetenv rs_puppet
[root@satnode1/# pcs constraint order set sat_vg fs_pgsql \
rs_pgsql
[root@satnode1/# pcs constraint order set sat_vg fs_mongodb \
rs_mongodb
[root@satnode1/# pcs constraint order set sat_vg fs_pulp \
fs_wwwpulp rs_pulp_workers rs_pulp_celerybeat
[root@satnode1/# pcs constraint order set sat_vg fs_foreman \
rs_foreman-proxy
[root@satnode1/# pcs constraint order start fs_wwwpulp then \
rs_httpd
```
- Order constraints to start services after parent service

```
[root@satnode1/# pcs constraint order set VirtualIP rs_qpidd \
rs_qdrouterd

[root@satnode1/# pcs constraint order VirtualIP then rs_dhcp
[root@satnode1/# pcs constraint order VirtualIP then rs_named
[root@satnode1/# pcs constraint order set rs_puppet rs_dhcp \
rs_named rs_pgsql sequential=false set rs_mongodb set rs_qpidd \
rs_qdrouterd sequential=false set rs_tomcat rs_pulp_workers \
rs_pulp_celerybeat rs_pulp_resource_manager \ rs_foreman-proxy
sequential=false set rs_httpd \ rs_foreman-tasks
sequential=false
```

For more details on creating pacemaker resource constraints, please refer to Pacemaker resource constraint documentation ³

4.1.4.6 Resource Details

1. Resources configured:

```
[root@satnode1 scripts]# pcs resource
VirtualIP (ocf::heartbeat:IPaddr2):           Started
sat_vg    (ocf::heartbeat:LVM):              Started
fs_pulp   (ocf::heartbeat:Filesystem):       Started
fs_candlepin (ocf::heartbeat:Filesystem):    Started
fs_foreman(ocf::heartbeat:Filesystem):       Started
fs_puppet (ocf::heartbeat:Filesystem):       Started
fs_wwwpulp(ocf::heartbeat:Filesystem):       Started
fs_mongodb(ocf::heartbeat:Filesystem):       Started
fs_pgsql  (ocf::heartbeat:Filesystem):       Started
fs_puppetenv (ocf::heartbeat:Filesystem):    Started
fs_tftpboot (ocf::heartbeat:Filesystem):     Started
fs_dhcp   (ocf::heartbeat:Filesystem):       Started
fs_named  (ocf::heartbeat:Filesystem):       Started
rs_dhcp   (systemd:dhcpd):                   Started
rs_named  (systemd:named):                   Started
rs_puppet (systemd:puppet):                  Started
rs_pgsql  (systemd:postgresql):              Started
rs_mongodb(systemd:mongod):                  Started
rs_qpidd  (systemd:qpidd):                   Started
rs_qdrouterd (systemd:qdrouterd):            Started
rs_tomcat (systemd:tomcat):                  Started
rs_pulp_workers (systemd:pulp_workers):      Started
rs_foreman-proxy (systemd:foreman-proxy):    Started
rs_pulp_resource_manager(systemd:pulp_resource_manager): Started
rs_pulp_celerybeat (systemd:pulp_celerybeat): Started
rs_httpd  (systemd:httpd):                   Started
rs_foreman-tasks (systemd:foreman-tasks):    Started
```

2. Resource constraints:

```
[root@satnode1 scripts]# pcs constraint
Location Constraints:
Ordering Constraints:
  start VirtualIP then start sat_vg (kind:Mandatory)
  start fs_wwwpulp then start rs_httpd (kind:Mandatory)
  start VirtualIP then start rs_dhcp (kind:Mandatory)
  start VirtualIP then start rs_named (kind:Mandatory)
Resource Sets:
  set sat_vg set fs_dhcp fs_named fs_puppet fs_puppetenv
  fs_pgsql fs_mongodb fs_pulp fs_wwwpulp fs_foreman fs_candlepin
  fs_tftpboot sequential=false
  set sat_vg fs_dhcp rs_dhcp
  set sat_vg fs_named rs_named
  set sat_vg fs_puppet fs_puppetenv rs_puppet
  set sat_vg fs_pgsql rs_pgsql
  set sat_vg fs_mongodb rs_mongodb
  set sat_vg fs_pulp fs_wwwpulp rs_pulp_workers
rs_pulp_celerybeat
  set sat_vg fs_foreman rs_foreman-proxy
  set VirtualIP rs_qpidd rs_qdrouterd
```

```
set rs_puppet rs_dhcp rs_named rs_pgsql sequential=false set
rs_mongodb set rs_qpidd rs_qdrouterd sequential=false set
rs_tomcat rs_pulp_workers rs_pulp_celerybeat
rs_pulp_resource_manager rs_foreman-proxy sequential=false set
rs_httpd rs_foreman-tasks sequential=false
Colocation Constraints:
  sat_vg with VirtualIP (score:INFINITY)
  rs_dhcp with fs_dhcp (score:INFINITY)
  rs_named with fs_named (score:INFINITY)
  rs_pgsql with fs_pgsql (score:INFINITY)
  rs_mongodb with fs_mongodb (score:INFINITY)
  rs_foreman-proxy with fs_foreman (score:INFINITY)
  rs_tomcat with VirtualIP (score:INFINITY)
  rs_httpd with VirtualIP (score:INFINITY)
  rs_foreman-tasks with fs_foreman (score:INFINITY)
Resource Sets:
  set fs_dhcp fs_named fs_puppet fs_puppetenv fs_pgsql
fs_mongodb fs_pulp fs_wwwpulp fs_foreman fs_candlepin fs_tftpboot
sequential=false set sat_vg setoptions score=INFINITY
  set fs_puppet fs_puppetenv rs_puppet setoptions score=INFINITY
  set fs_pulp fs_wwwpulp rs_pulp_workers setoptions
score=INFINITY
  set VirtualIP rs_qpidd rs_qdrouterd setoptions score=INFINITY
  set fs_pulp fs_wwwpulp rs_pulp_resource_manager setoptions
score=INFINITY
  set fs_pulp fs_wwwpulp rs_pulp_celerybeat setoptions
score=INFINITY
```

4.1.4.7 Pacemaker Cluster Status

Activate the all the nodes back in the cluster:

```
[root@satnode1 scripts]# pcs cluster unstandby satnode3
```

```
[root@satnode1 scripts]# pcs cluster unstandby satnode2
```

Confirm that all the nodes are active:

```
[root@satnode1 ~]# pcs status
```

```
Cluster name: SAT62-HA
Last updated: Wed May 18 10:03:11 2016      Last change: Wed Nov 18
10:03:06 2015 by root via crm_attribute on satnode1
Stack: corosync
Current DC: satnode1 (version 1.1.13-a14efad) - partition with quorum
3 nodes and 30 resources configured
```

```
Online: [ satnode1 satnode2 satnode3 ]
```

```
Full list of resources:
```

```
satnode1-fence (stonith:fence_ipmilan): Started satnode1
satnode2-fence (stonith:fence_ipmilan): Started satnode2
satnode3-fence (stonith:fence_ipmilan): Started satnode3
VirtualIP (ocf::heartbeat:IPaddr2): Started satnode1
sat_vg (ocf::heartbeat:LVM): Started satnode1
fs_pulp (ocf::heartbeat:Filesystem): Started satnode1
```



```
fs_candlepin      (ocf::heartbeat:Filesystem):Started satnode1
fs_foreman        (ocf::heartbeat:Filesystem):Started satnode1
fs_puppet         (ocf::heartbeat:Filesystem):Started satnode1
fs_wwwpulp        (ocf::heartbeat:Filesystem):Started satnode1
fs_mongodb        (ocf::heartbeat:Filesystem):Started satnode1
fs_pgsq1          (ocf::heartbeat:Filesystem):Started satnode1
fs_puppetenv      (ocf::heartbeat:Filesystem):Started satnode1
fs_tftpboot       (ocf::heartbeat:Filesystem):Started satnode1
fs_dhcp           (ocf::heartbeat:Filesystem):Started satnode1
fs_named          (ocf::heartbeat:Filesystem):Started satnode1
rs_dhcp           (systemd:dhcpd): Started satnode1
rs_named          (systemd:named): Started satnode1
rs_puppet         (systemd:puppet): Started satnode1
rs_pgsq1          (systemd:postgresql): Started satnode1
rs_mongodb        (systemd:mongod): Started satnode1
rs_qpidd          (systemd:qpidd): Started satnode1
rs_qdrouterd     (systemd:qdrouterd): Started satnode1
rs_tomcat         (systemd:tomcat): Started satnode1
rs_pulp_workers  (systemd:pulp_workers): Started satnode1
rs_foreman-proxy  (systemd:foreman-proxy): Started satnode1
rs_pulp_resource_manager (systemd:pulp_resource_manager): Started
satnode1
rs_pulp_celerybeat (systemd:pulp_celerybeat): Started satnode1
rs_httpd          (systemd:httpd): Stopped
rs_foreman-tasks  (systemd:foreman-tasks): Stopped

PCSD Status:
  satnode1: Online
  satnode2: Online
  satnode3: Online

Daemon Status:
  corosync: active/enabled
  pacemaker: active/enabled
  pcsd: active/enabled
```

4.1.4.8 Pacemaker Resource Stickiness

Pacemaker has the concept of resource stickiness which controls how much a service prefers to stay running where it is. This prevents resources from moving after cluster/resource recovery. It controls how much a service prefers to stay running where it is. By default, Pacemaker set it to zero. While it is possible to set the stickiness for each resource, a default setting applies to all resources.

```
[root@satnode1 scripts]# pcs property set \
default-resource-stickiness=INFINITY
```

Please refer to Pacemaker Configuration Output for configuration details.

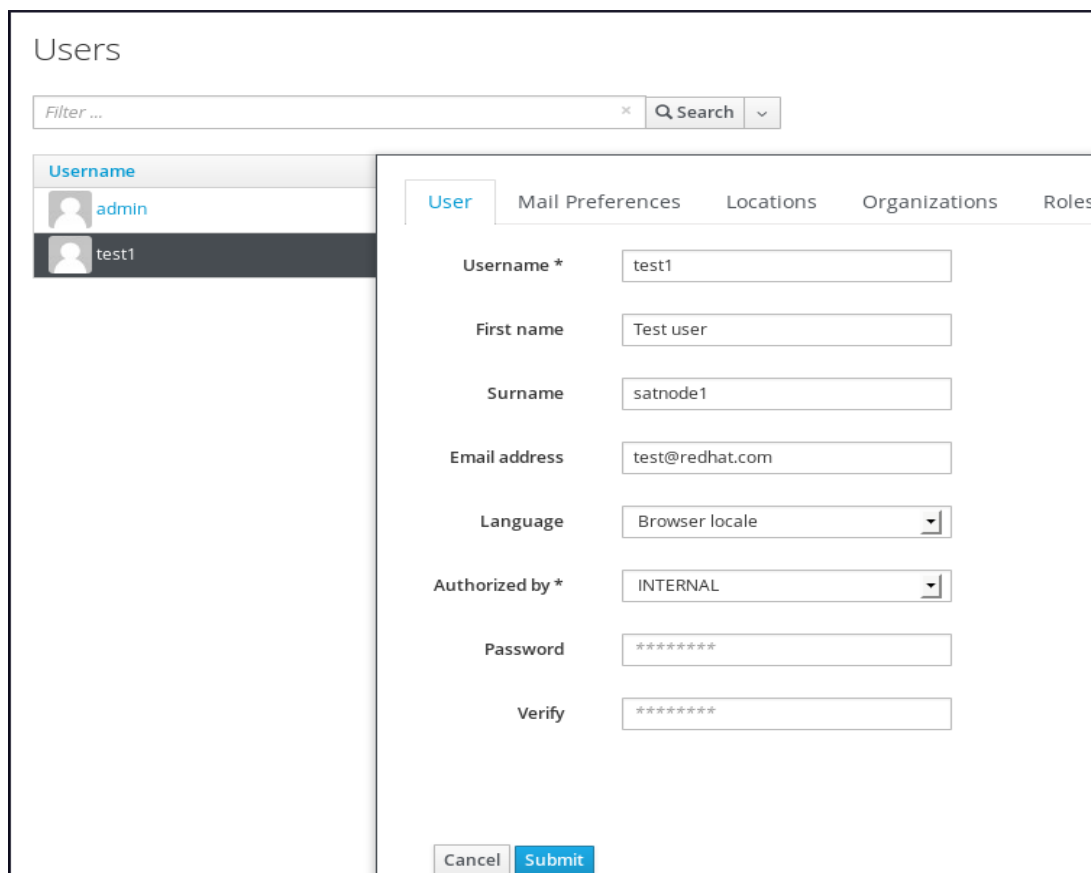
4.1.5 Verify Cluster Fail Over

While the Satellite is running on **satnode1**, verify Satellite functionality.

Add surname to user **test1** as 'satnode1' and save the change.

Initiate a fail over of the cluster to **satnode2**. After saving the change, fail over the cluster to **satnode2**.

```
[root@satnode1 scripts]# pcs cluster standby satnode1
```



The screenshot shows a web interface for managing users. On the left, a list of users includes 'admin' and 'test1'. The 'test1' user is selected, and a modal form is open to edit their details. The form fields are as follows:

Field	Value
Username *	test1
First name	Test user
Surname	satnode1
Email address	test@redhat.com
Language	Browser locale
Authorized by *	INTERNAL
Password	*****
Verify	*****

At the bottom of the form are 'Cancel' and 'Submit' buttons.

Illustration 8: Cluster Failover



After the failover is completed, verify via **pcs status** that Satellite is running on node **satnode2**.

```
[root@satnode1 scripts]# pcs status | grep foreman-tasks
rs_foreman-tasks (systemd:foreman-tasks): Started satnode2
```

Login to the Satellite user interface and identify if user “*test1*” has the surname updated as “*satnode1*”

4.1.6 Backing Up And Restoring A Pacemaker Cluster Configuration

As of RHEL 7.1, pacemaker configuration can be backed up and restored.

Create a backup of the configuration:

```
[root@satnode1 scripts]# pcs config backup sat62-ha_pacemaker_backup

[root@satnode1 scripts]# ls -l sat62-ha_pacemaker*
-rw----- . 1 root root 3190 Dec 10 10:22 sat62-ha_pacemaker_backup.tar.bz2
```

Command to use if there is requirement to restore from the backup

```
[root@satnode1 scripts]# pcs config restore [--local] \
sat62-ha_pacemaker_backup.tar.bz2
```

Note: For versions prior to RHEL 7.1, the pacemaker configuration can be backed up and restored using “*cibadmin* command”

Create a backup using *cibadmin*:

```
[root@satnode1 scripts]# cibadmin -Q > sat62-ha_backup.xml
```

Restore using *cibadmin*:

```
[root@satnode1 scripts]# cibadmin -C --xml-file sat62-ha_backup.xml
```

4.2 Configuring Satellite 6 Environment

The Satellite 6 environment has many configuration categories as displayed in the dashboard:

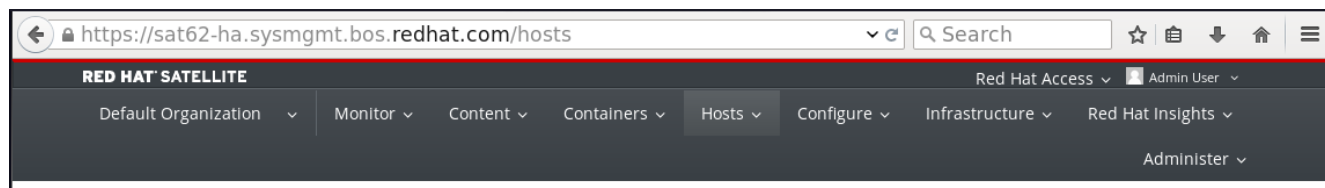


Illustration 9: Satellite 6.2 Dashboard

Some of these categories are required and the following are been used in the reference architecture environment:

- Administer
- Content
- Hosts
- Configure
- Infrastructure

4.2.1 Satellite 6 Server Configuration Work Flow

The following workflow describes the components that must be configured under each category and their sequence:

Administer

1. Change passwd
2. Define Organization
3. Define Location

Content

1. Life cycle environment
2. Red Hat Subscriptions
3. Add repositories
4. Synchronize content
5. Create content view and publish it
6. Create activation key

Hosts

1. Configure provisioning template
2. Define operating system and associate it with provisioning template and kickstart file

3. Define Installation media and associate it with Organization and location

Infrastructure

1. Create Domain
2. Create Subnet
3. Create Compute resources

Configure

1. Create a Host group

4.2.2 Administer

4.2.2.1 Change Satellite User Password

Click the user name at the top right corner

- Select **My Account** from the menu.
- Type in a new password in the **Password** field.
- Type in the new password again in the **Verify** field.
- Click the **Submit** button to save the new password

4.2.2.2 Create Organization

From the top right corner of the main page, select:

Administrator-► Organizations-► New Organization

Type in Organization name (“refarch” in this case) and description and click submit.

4.2.2.3 Define Location

This reference architecture uses “remote” location

Create “remote”location:

From the top right corner of the main page, select:**Administrator-► Locations-► New Location**

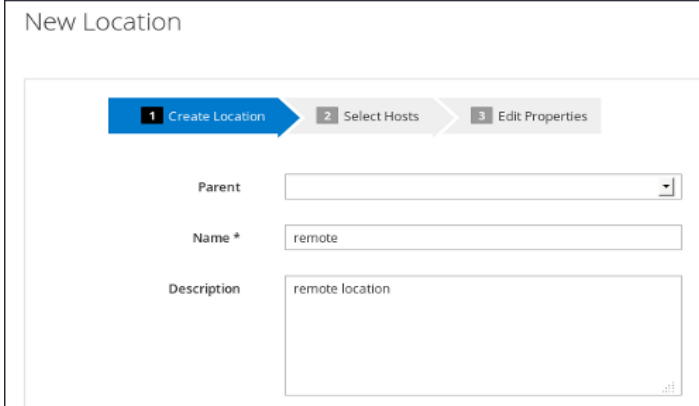


Illustration 10: Create Location

Type in location name (“remote” in this case) and description and click submit.

4.2.3 Content

This reference architecture uses “Library” as the default Environment.

4.2.3.1 Red Hat Subscriptions

Log in to access.redhat.com (available to subscribers with Satellite subscriptions on the account, when logged into the Customer Portal as an Organizational Administrator user)

Select "Subscriptions" from top left corner of access.redhat.com

Subscriptions-► Satellite-► Register a Satellite

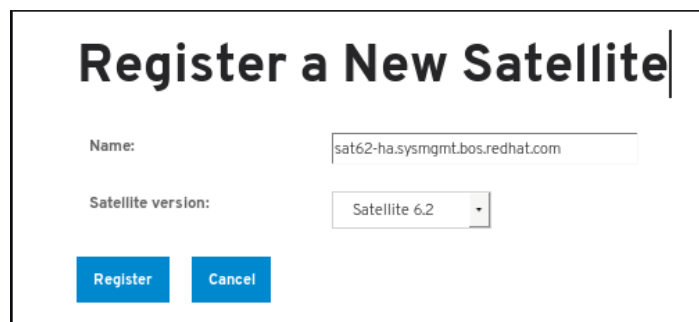
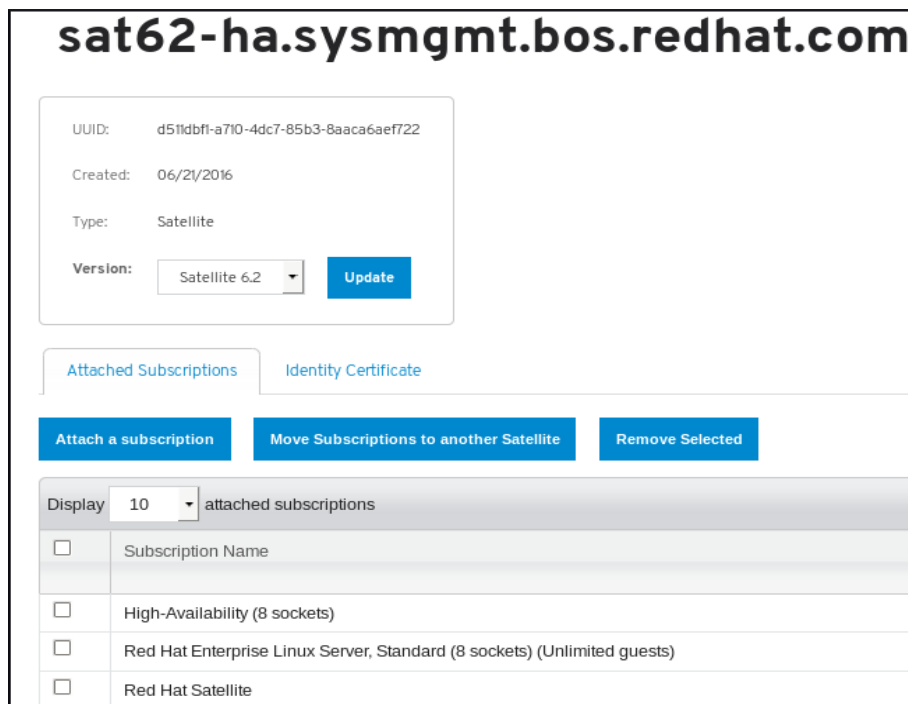


Illustration 11: Register Satellite

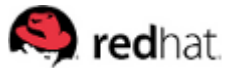
Attach required subscriptions. Once this has been submitted, the manifest can be downloaded as a zip file by clicking the “Download manifest” button.



<input type="checkbox"/>	Subscription Name
<input type="checkbox"/>	High-Availability (8 sockets)
<input type="checkbox"/>	Red Hat Enterprise Linux Server, Standard (8 sockets) (Unlimited guests)
<input type="checkbox"/>	Red Hat Satellite

Illustration 12: Subscription

The downloaded manifest is uploaded to the satellite server via user interface in the “refarch”



organization.

Navigate to:

Content-► Red Hat Subscriptions-► Manage Manifest

Browse the downloaded file and upload it. The subscription status will be displayed as described in the illustration.

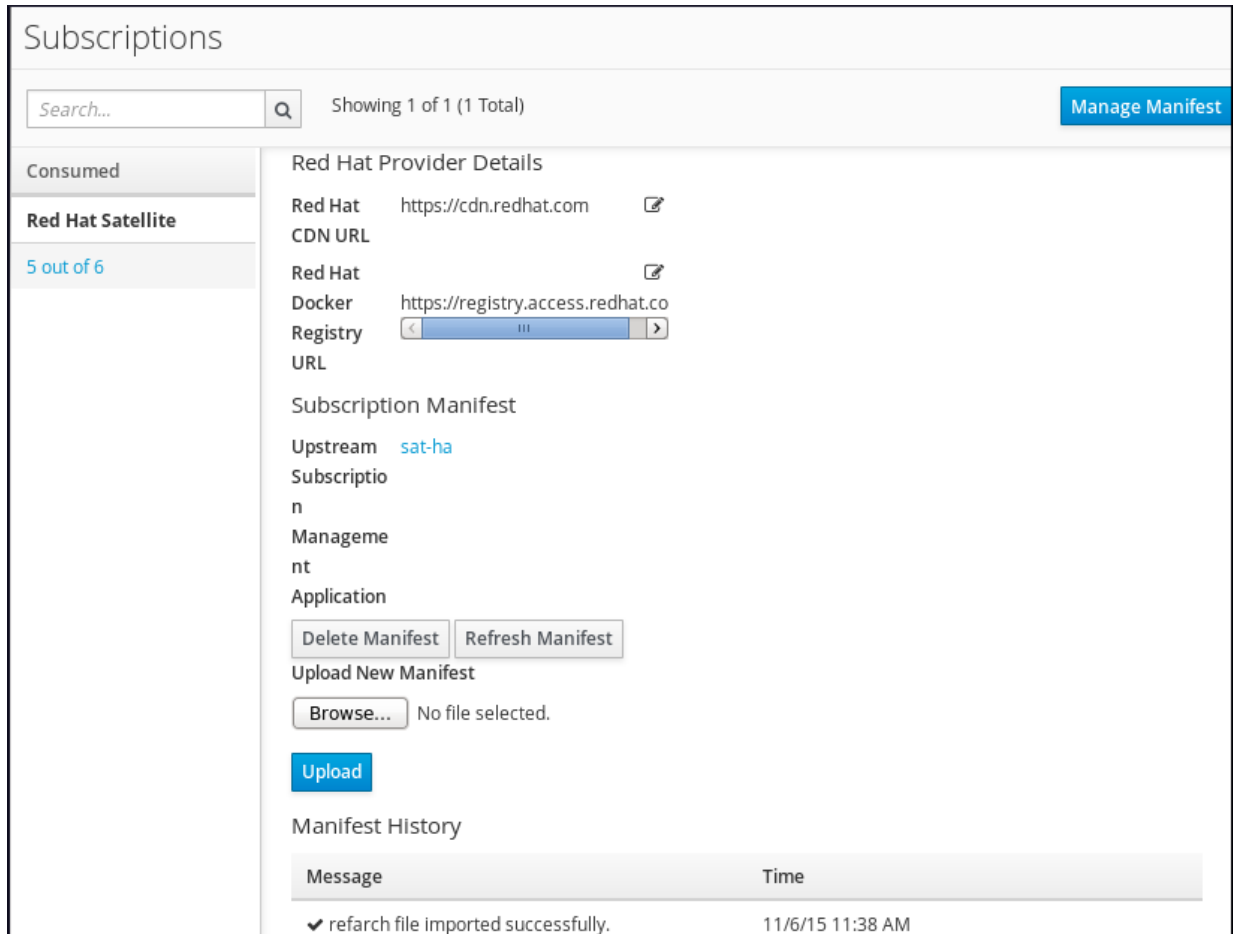


Illustration 13: Upload Manifest

4.2.3.2 Red Hat Repositories

Expand the available repositories and select the required ones:

Content-► Red Hat Repositories

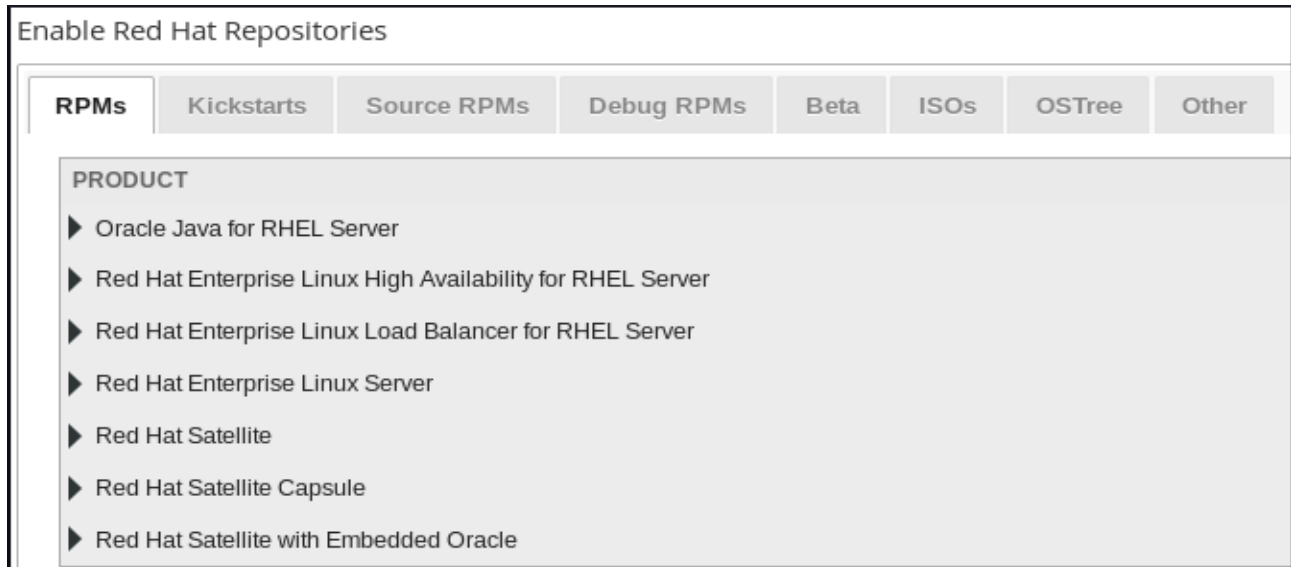


Illustration 14: Repository Selection

Select the kickstart file on the ***Kickstart*** tab

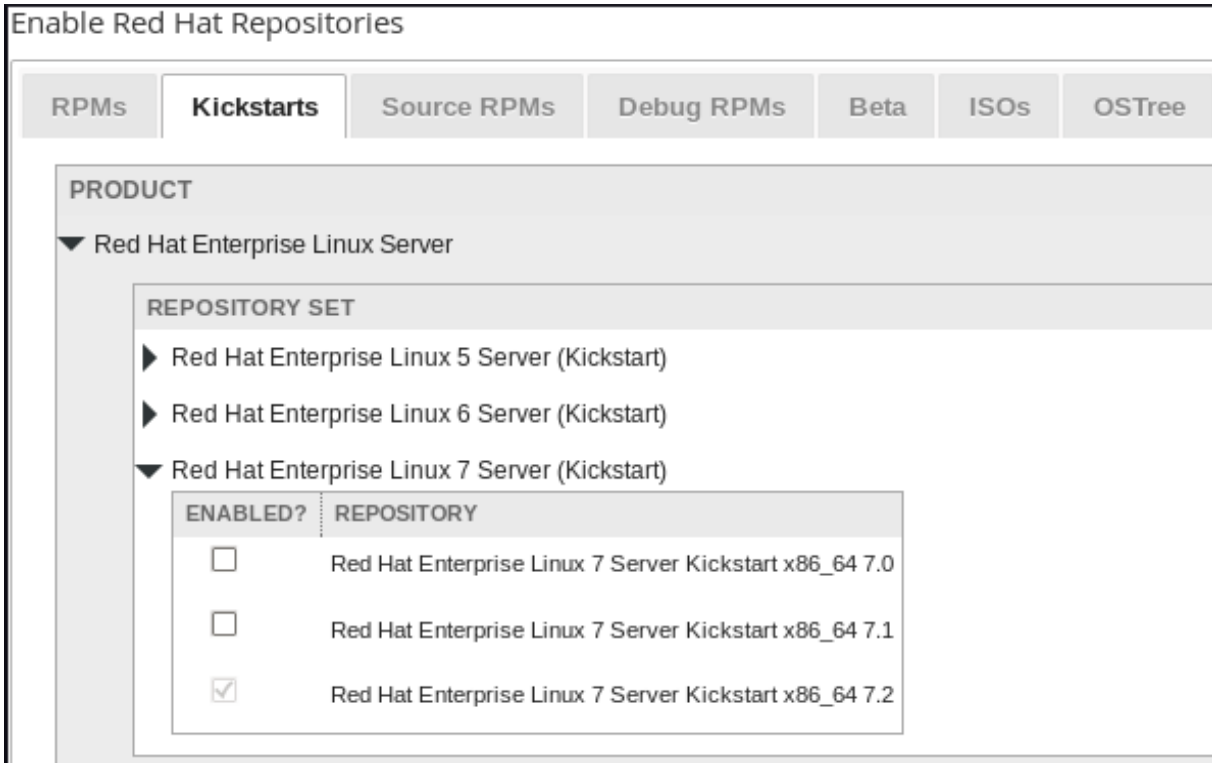


Illustration 15: Kickstart Selection

4.2.3.3 Synchronize Content

Once the repositories have been selected, the content must be synchronized:

Content-► Sync Status

Select all the relevant repositories that requires synchronizing and select **Synchronize now** button.

Sync Status

[Collapse All](#) [Expand All](#) [Select None](#) [Select All](#) Active only

PRODUCT	START TIME	DURATION	DETAILS	RESULT
▼ Red Hat Satellite Capsule <input checked="" type="checkbox"/> Red Hat Satellite Capsule 6.1 for RHEL 7 Server RPMs x86_64	10 days ago	1 minute	New packages: 176 (203 MB).	Syncing Complete.
▼ Red Hat Enterprise Linux High Availability for RHEL Server ▼ 7.1 ▼ x86_64 <input checked="" type="checkbox"/> Red Hat Enterprise Linux High Availability for RHEL 7 Server RPMs x86_64 7.1				Never Synced
▼ Red Hat Enterprise Linux Server ▼ 7.1 ▼ x86_64 <input checked="" type="checkbox"/> Red Hat Enterprise Linux 7 Server Kickstart x86_64 7.1 <input checked="" type="checkbox"/> Red Hat Enterprise Linux 7 Server RPMs x86_64 7.1	10 days ago	8 minutes	New packages: 4371 (3.21 GB).	Syncing Complete.
	10 days ago	20 minutes	New packages: 6950 (8.43 GB).	Syncing Complete.
▼ 7Server ▼ x86_64 <input checked="" type="checkbox"/> Red Hat Enterprise Linux 7 Server RPMs x86_64 7Server <input checked="" type="checkbox"/> Red Hat Satellite Tools 6.1 for RHEL 7 Server RPMs x86_64	10 days ago	20 minutes	New packages: 6952 (8.44 GB).	Syncing Complete.
	9 days ago	less than a minute	New packages: 20 (4.69 MB).	Syncing Complete.

[Synchronize Now](#)

Illustration 16: Content Synchronization

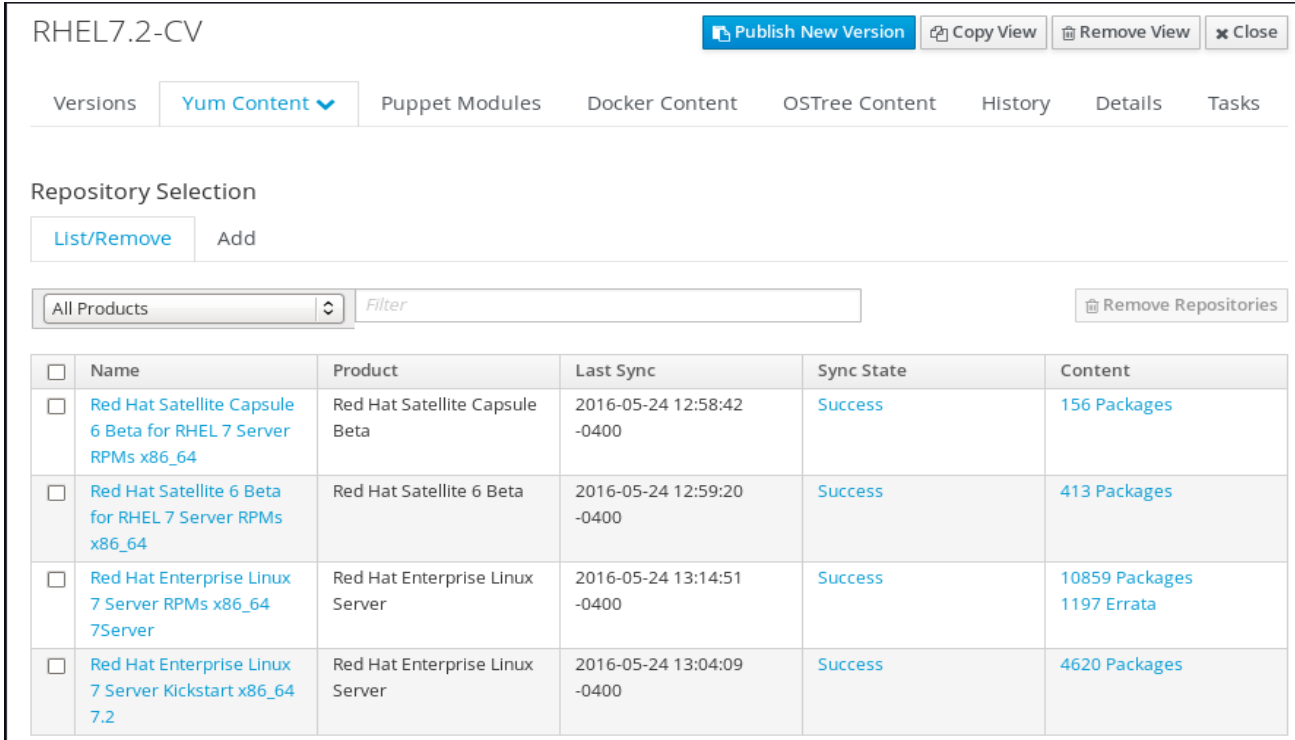
4.2.3.4 Create Content View And Publish

Content-► Content Views-► Create New View

Name RHEL7 . 2 - CV

Label RHEL7 . 2 - CV

Yum Content tab – Select the repositories required to be added to the content view

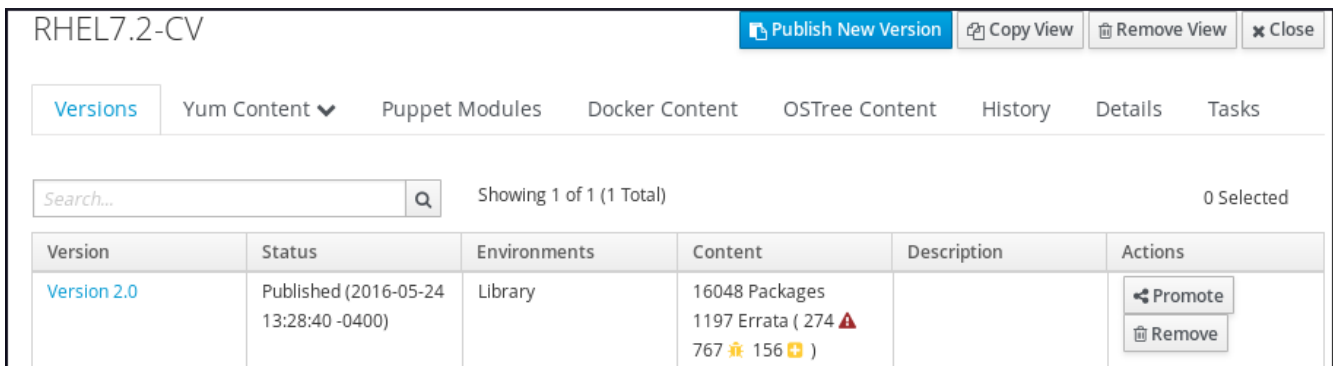


The screenshot shows the 'Yum Content' tab for 'RHEL7.2-CV'. It features a 'Repository Selection' section with a 'List/Remove' button and an 'Add' button. Below this is a search bar with 'All Products' selected and a 'Filter' input field. A table lists the selected repositories with columns for Name, Product, Last Sync, Sync State, and Content.

<input type="checkbox"/>	Name	Product	Last Sync	Sync State	Content
<input type="checkbox"/>	Red Hat Satellite Capsule 6 Beta for RHEL 7 Server RPMs x86_64	Red Hat Satellite Capsule Beta	2016-05-24 12:58:42 -0400	Success	156 Packages
<input type="checkbox"/>	Red Hat Satellite 6 Beta for RHEL 7 Server RPMs x86_64	Red Hat Satellite 6 Beta	2016-05-24 12:59:20 -0400	Success	413 Packages
<input type="checkbox"/>	Red Hat Enterprise Linux 7 Server RPMs x86_64 7Server	Red Hat Enterprise Linux Server	2016-05-24 13:14:51 -0400	Success	10859 Packages 1197 Errata
<input type="checkbox"/>	Red Hat Enterprise Linux 7 Server Kickstart x86_64 7.2	Red Hat Enterprise Linux Server	2016-05-24 13:04:09 -0400	Success	4620 Packages

Illustration 17: Create Content View

After a content view is created, it must be published to be usable by hosts. Click "**Publish New Version**" button.



The screenshot shows the 'Yum Content' tab for 'RHEL7.2-CV' after publishing. The 'Publish New Version' button is highlighted. Below the repository selection area, there is a search bar and a table showing the published version details.

Version	Status	Environments	Content	Description	Actions
Version 2.0	Published (2016-05-24 13:28:40 -0400)	Library	16048 Packages 1197 Errata (274 ▲ 767 ★ 156 ☹)		<input type="button" value="Promote"/> <input type="button" value="Remove"/>

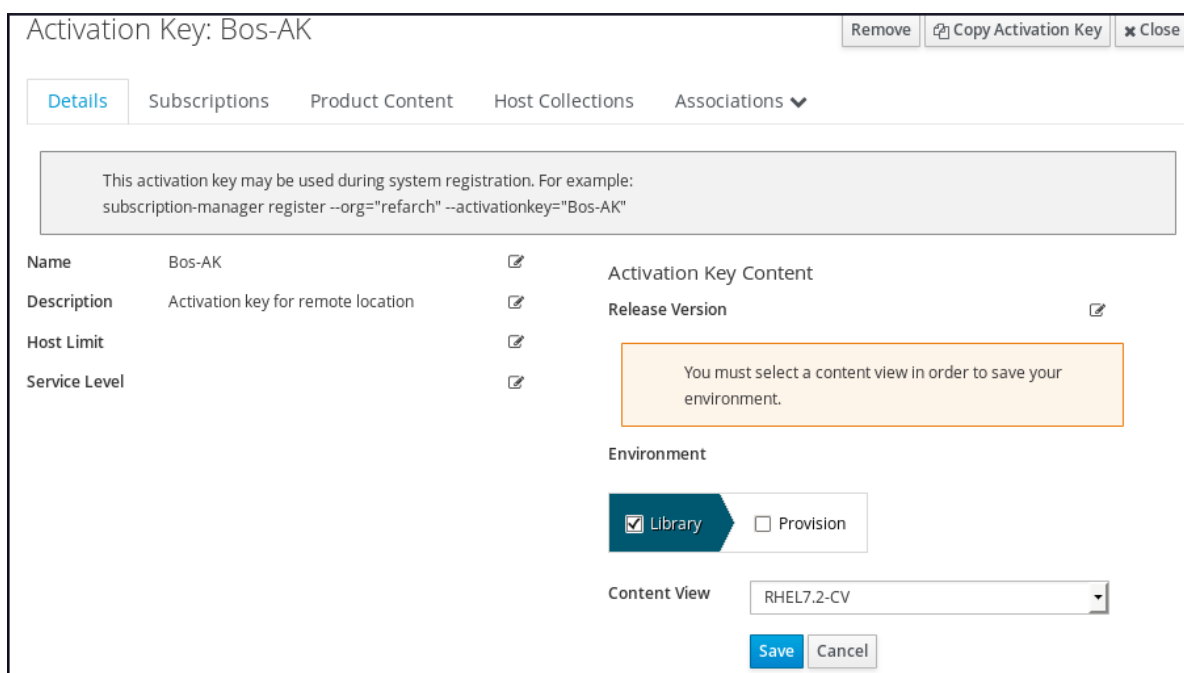
Illustration 18: Publish Content

Note: If there is a failure while publishing with the following message:
 "ERF12-4115 [ProxyAPI::ProxyException]: Klassen für KT_The_Demo_Org_Auto_Library_small_cv_22 konnten nicht von Puppet geladen werden ([RestClient::Forbidden]: 403 Forbidden) für Proxy <https://satellite.example.com:9090/puppet>", ensure that all the Satellite Server nodes (satnode1/2/3) have been added as trusted hosts as mentioned in step Add Trusted Hosts

4.2.3.5 Create Activation Keys

Create an activation key “Bos-AK”

Content-► Activation Keys-► New Activation Key



Activation Key: Bos-AK

Remove Copy Activation Key Close

Details Subscriptions Product Content Host Collections Associations ▼

This activation key may be used during system registration. For example:
`subscription-manager register --org="refarch" --activationkey="Bos-AK"`

Name	Bos-AK	<input checked="" type="checkbox"/>
Description	Activation key for remote location	<input checked="" type="checkbox"/>
Host Limit		<input checked="" type="checkbox"/>
Service Level		<input checked="" type="checkbox"/>

Activation Key Content

Release Version

You must select a content view in order to save your environment.

Environment

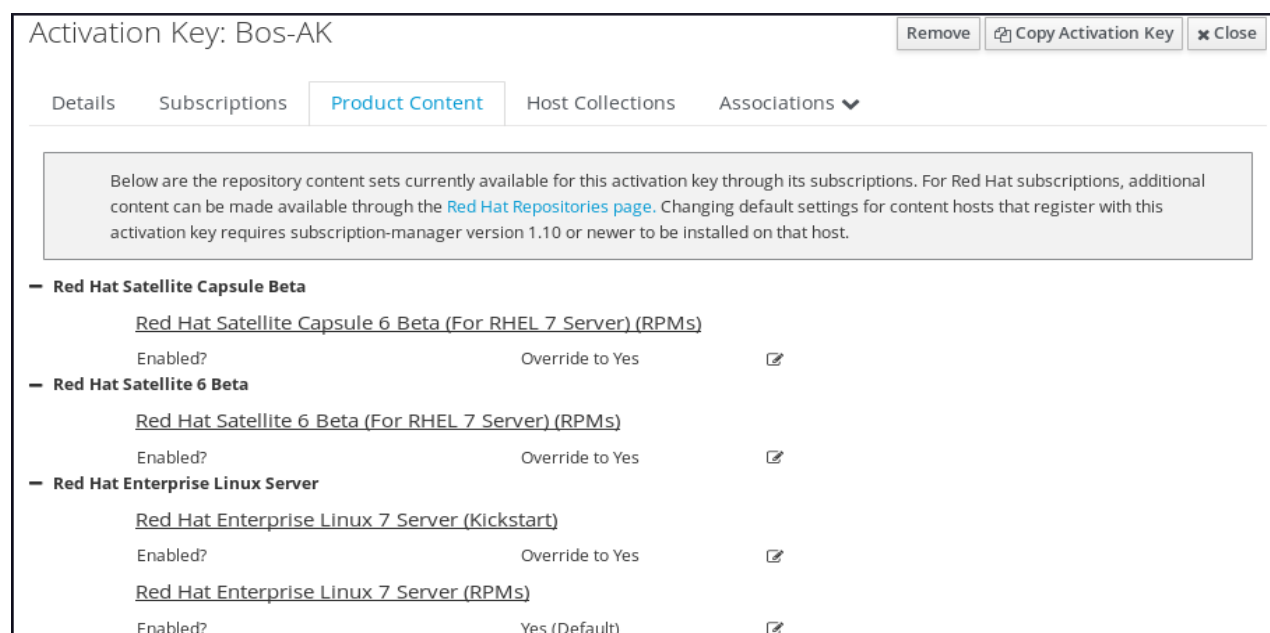
Library Provision

Content View RHEL7.2-CV

Save Cancel

Illustration 19: Activation Key

Select the **Subscription** tab and place a checkmark next to Auto Attach. This option attaches a system upon registration to all custom products and required Red Hat subscriptions.



Activation Key: Bos-AK

Remove Copy Activation Key Close

Details Subscriptions **Product Content** Host Collections Associations ▼

Below are the repository content sets currently available for this activation key through its subscriptions. For Red Hat subscriptions, additional content can be made available through the [Red Hat Repositories page](#). Changing default settings for content hosts that register with this activation key requires subscription-manager version 1.10 or newer to be installed on that host.

- Red Hat Satellite Capsule Beta**
 - Red Hat Satellite Capsule 6 Beta (For RHEL 7 Server) (RPMs)

Enabled?	Override to Yes	<input checked="" type="checkbox"/>
----------	-----------------	-------------------------------------
- Red Hat Satellite 6 Beta**
 - Red Hat Satellite 6 Beta (For RHEL 7 Server) (RPMs)

Enabled?	Override to Yes	<input checked="" type="checkbox"/>
----------	-----------------	-------------------------------------
- Red Hat Enterprise Linux Server**
 - Red Hat Enterprise Linux 7 Server (Kickstart)

Enabled?	Override to Yes	<input checked="" type="checkbox"/>
----------	-----------------	-------------------------------------
 - Red Hat Enterprise Linux 7 Server (RPMs)

Enabled?	Yes (Default)	<input checked="" type="checkbox"/>
----------	---------------	-------------------------------------

Illustration 20: Activation Key - Product Content



4.2.4 Hosts

1. Configure provisioning template
2. Define operating system and associate it with provisioning template and kickstart file
3. Define Installation media and associate it with Organization and location

4.2.4.1 Operating System

The Operating System gets created base on Repository selections as follows:

Hosts-► Operating systems-► RedHat 7.2

Operating System tab:

Name	RedHat7.2
Major version	7
Minor version	2
OS Family	RedHat
Root password hash	
Architecture	x86_64

Partition table tab:

tables	Kickstart default
--------	-------------------

Installation media tab: SHA256

Installation media	refarch/Library/Red_Hat_Server/Red_Hat_Enterprise_Linux_7_Server_Kickstart_x86_64_7_2
--------------------	---

Templates tab:

PXELinux*	Kickstart default PXELinux
iPXE*	Kickstart default iPXE
provision*	Satellite Kickstart Default
finish*	Satellite Kickstart Default Finish
user_data*	Satellite Kickstart Default User Data

4.2.4.2 Provisioning Templates

Existing templates were used in this reference architecture and were assigned to “refarch” Organization by default.

Following are the templates used:

- Kickstart default PXELinux
- Kickstart default iPXE
- Satellite Kickstart Default
- Satellite Kickstart Default Finish
- Satellite Kickstart Default User Data

These templates must be associated to host group, environment or Operating system. Also they must be assigned to locations and Organizations.

When a Host requests a template during provisioning, Foreman will select the best match from the available templates of that type, in the following order:

- Host group and Environment
- Host group only
- Environment only
- Operating system default

The illustration shows the template assigned to Operating system “RedHat7.2” :

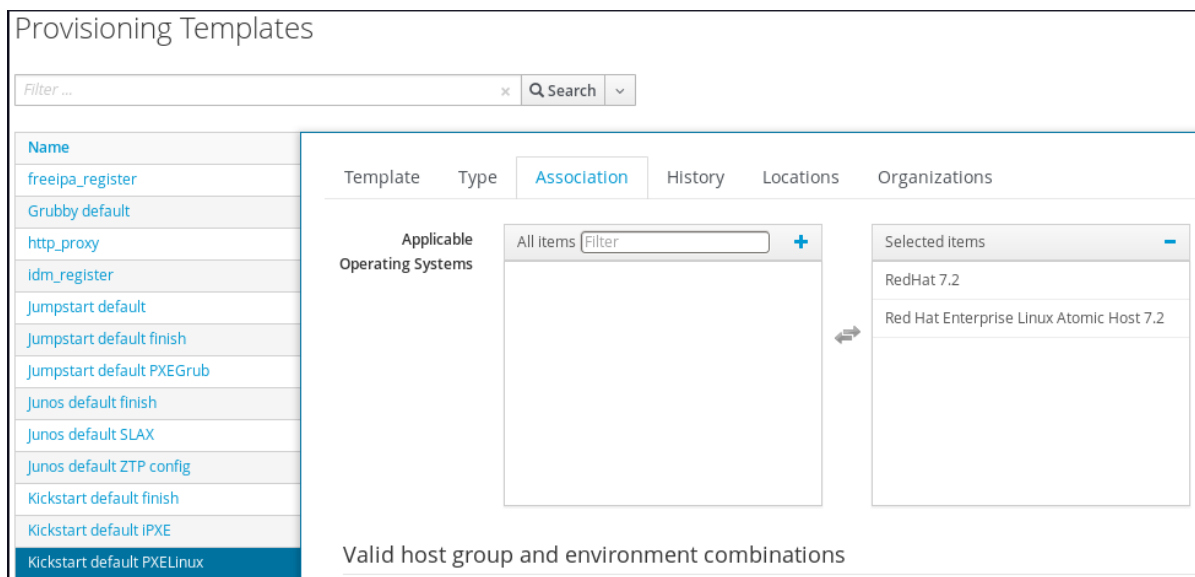


Illustration 21: Provisioning Template Association – Operating System

Associate location and Organization in the same manner:

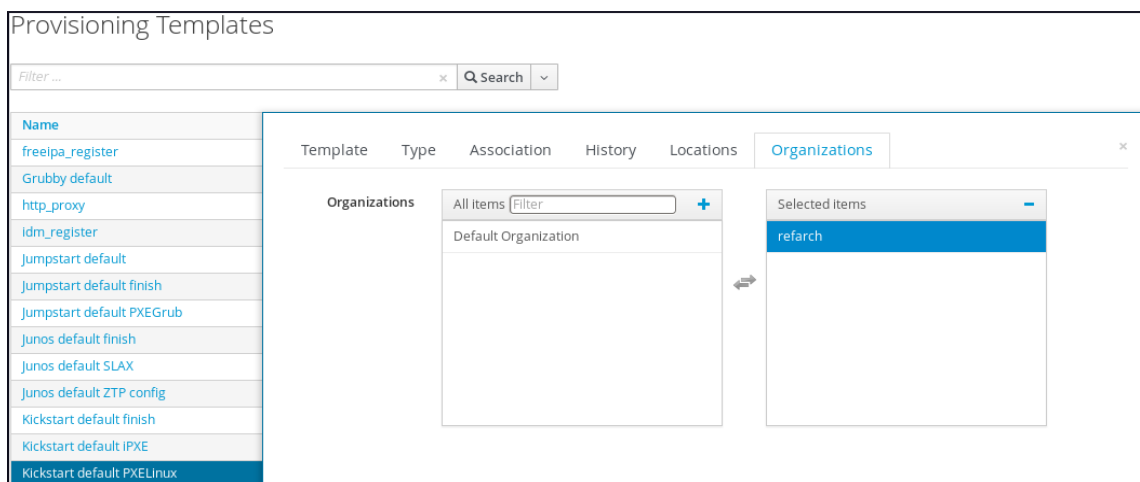


Illustration 22: Provisioning Template Association - Organization



To view the assigned templates:

refarch (top left corner of the dashboard)-► **Manage Organizations**-► **refarch**-► **Templates**

Notice the already assigned templates

4.2.4.3 Installation Media

No new medium was created in this reference architecture.

By default, the installation medium -

refarch/Library/Red_Hat_Server/Red_Hat_Enterprise_Linux_7_Server_Kickstart_x86_64_7_2

has been assigned to refarch and Library as part of the selection of Kickstart selection under step Red Hat Repositories.

4.2.5 Infrastructure

An integrated Satellite Capsule gets created during Satellite install by default. Additional Capsule Servers created using “*satellite-install scenario capsule*” command can be assigned to the appropriate Organization and Location as follows:

Manage Organizations -► **refarch**-► **Capsules**

Select the newly added Capsule Servers. The list of assigned Capsule Servers can be viewed at:

Infrastructure -► **Capsules**

Assign the Lifecycle Environment, Organization and Location if not assigned already:

Select the Capsule Server and select **Edit** and assign the appropriate values.

Content Synchronization:

Infrastructure -► **Capsules** Select the Capsule Server and select **Synchronize**.

For details on Capsule Server installation and configuration, refer to 56

4.2.5.1 Create Domain

There are two domains used in this reference architecture:

- `sysmgmt.bos.redhat.com` (created during Satellite Server Installation)
- `remote.bos.redhat.com` (for external capsule environment)

On the top left corner, select **Default Organization** -► **Default Organization** -► **refarch**

Create the domain:

Infrastructure -► **Domains**-► **New Domain**

Enter DNS Domain name “`remote.bos.redhat.com` ” and select DNS Capsule “`capsule-lb.remote.bos.redhat.com`”

Note: The Capsule Server “ `capsule-lb.remote.bos.redhat.com`” is a virtual capsule created under Creating Virtual Capsule Server

Description	Domain	Parameters	Locations	Organizations
remote.bos.redhat.com				
	DNS domain *	<input type="text" value="remote.bos.redhat.com"/>		The full DNS domain name
	Description	<input type="text"/>		Full name describing the domain
	DNS Capsule	<input type="text" value="capsule-lb.remote.bos.redhat.c..."/>		DNS Capsule to use within this do that PTR records are managed via

Illustration 23: Create Domain

Location tab select - remote

Organization tab select - refarch

4.2.5.2 Create Subnet

The subnet used in this reference architecture is “remote-net”

Infrastructure -> Subnets-> New Subnet

subnet tab specify -

Name	remote-net
Network address	10.19.3.0
Network mask	255.255.255.0
Gateway address	10.19.3. 254
Primary DNS	10.19.3.51 (IP address of cap1)
Secondary DNS	10.19.143.247
IPAM	DHCP
Start of IP range	10.19.3.101
End of IP range	10.19.3.150
Boot mode	DHCP

Domains tab select –

Domain	remote.bos.redhat.com
--------	-----------------------

Capsules tab select-

DHCP Capsule	capsule-lb.remote.bos.redhat.com
TFTP Capsule	capsule-lb.remote.bos.redhat.com
DNS Capsule	capsule-lb.remote.bos.redhat.com
Discovery Proxy	capsule-lb.remote.bos.redhat.com

Location tab select - remote

Organization tab select - refarch

Click submit to complete the subnet creation.



4.2.5.3 Create Compute Resources

This environment uses “Red Hat Enterprise Virtualization (RHEV)” for virtual machines. The RHEV environment used is managed by “se-rhev.cloud.lab.eng.bos.redhat.com”

Infrastructure - ► Compute Resources-► New Compute Resource

Compute Resource tab select -

Name	RHEVM
Provider	RHEV
URL	https://se-rhev.cloud.lab.eng.bos.redhat.com/api
Username	admin@internal
Password	*****
Datacenter	Westford

Locations tab select -

Default Location, remote

Organization tab select

refarch

4.2.6 Configure

4.2.6.1 Host Groups

HG-Remote-LB is the Host Group used in this reference architecture:

Configure - ► Host Groups-► New Host Group

Host Group tab specify -

Name	HG-Remote-LB
Lifecycle Environment	Library
Content View	RHEL7.2-CV
Puppet Environment	Production

Capsule Settings:

Content Source	capsule-lb.remote.bos.redhat.com
Puppet CA	capsule-lb.remote.bos.redhat.com
Puppet Master	capsule-lb.remote.bos.redhat.com

Network tab specify -

Domain	remote.bos.redhat.com
Subnet	remote-net(10.19.2.0/24)

Operating System tab specify -

Architecture	x86_64
Operating System*	RedHat 7.2
Media *	Red Hat Enterprise Linux 7 Server Kickstart x86_64 7.2

(http://sat62-ha.sysmgmt.bos.redhat.com/pulp/repos/refarch/Library/RHEL7_1-X86_64/content/dist/rhel/server/7/7.2/x86_64/kickstart)

Partition Table *	Kickstart default
Root password	*****

Locations tab specify -

Location remote

Organizations tab specify -

Organizations * refarch

Activation Keys tab specify -

Activation keys * Bos-AK

5 Configuring Load Balanced External Capsules

Please ensure the Capsule servers are properly sized and meet requirements as mentioned in the Red Hat Satellite 6.2 documentation⁴

5.1 Capsule Configuration Detail

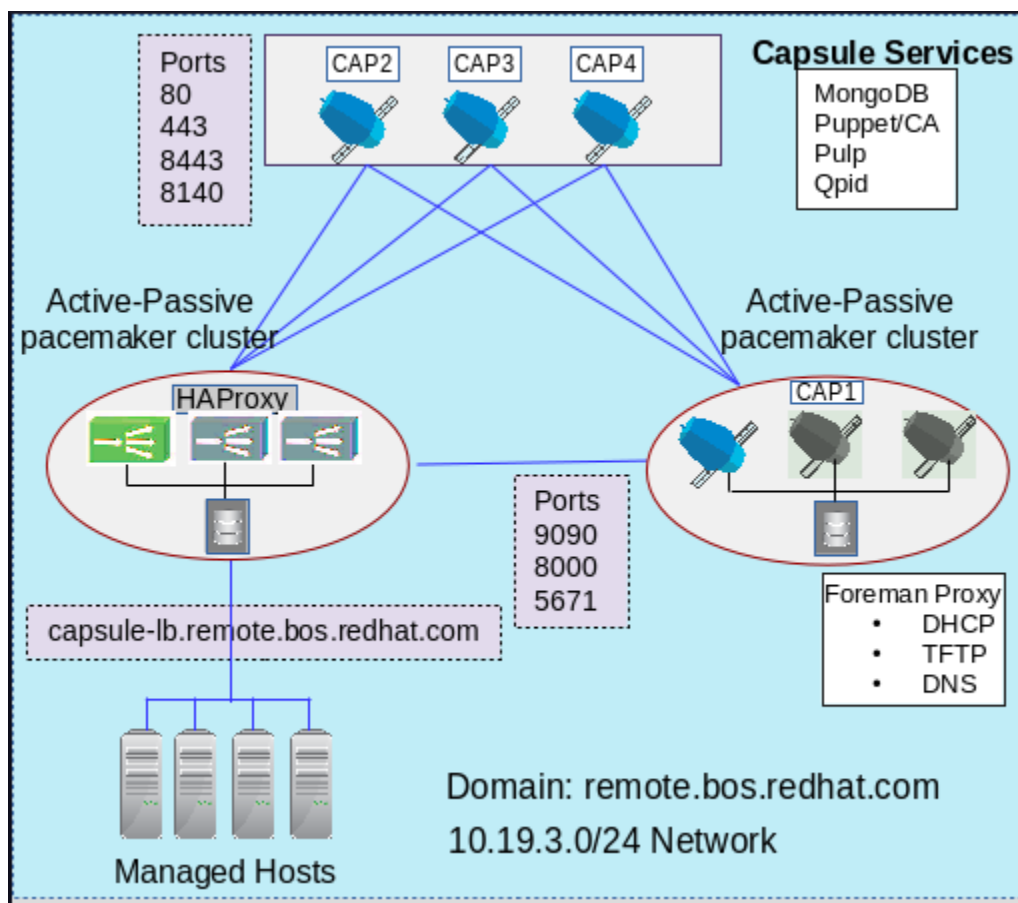


Illustration 24: Load Balanced Capsule Configuration

The reference architecture details a capsule environment that has a capsule “cap1” rendering foreman-proxy services and capsules “cap2,” “cap3” and “cap4” rendering content services. “cap2/3/4” capsules have identical configuration and offer redundancy via HAProxy load balancer. “cap1” is unique and is made highly available by an active-passive pacemaker setup across nodes “**cap1a**,” “**cap1b**” and “**cap1c**” with shared filesystems. The HAProxy

⁴ <https://access.redhat.com/documentation/en/red-hat-satellite/6.2-beta/paged/installation-guide/chapter-2-preparing-your-environment-for-installation>



loadbalancer is setup with an active-passive pacemaker cluster for high availability. The managed hosts point to a virtual capsule “capsule-lb.remote.bos.redhat.com”.

Server	IP Address	Comment
cap1.remote.bos.redhat.com	10.19.3.51	Foreman proxy Capsule Server VIP
cap1a.remote.bos.redhat.com	10.19.3.48	Cap1 Cluster node A
cap1b.remote.bos.redhat.com	10.19.3.49	Cap1 Cluster node B
cap1c.remote.bos.redhat.com	10.19.3.47	Cap1 Cluster node C
capsule-lb.remote.bos.redhat.com	10.19.3.50	Virtual Capsule IP
Hap1-sat6.remote.bos.redhat.com	10.19.3.81	HAProxy node A
Hap2-sat6.remote.bos.redhat.com	10.19.3.82	HAProxy node B
Hap3-sat6.remote.bos.redhat.com	10.19.3.83	HAProxy node C
cap2.remote.bos.redhat.com	10.19.3.52	Loadbalanced capsule
cap3.remote.bos.redhat.com	10.19.3.53	Loadbalanced capsule
cap4.remote.bos.redhat.com	10.19.3.54	Loadbalanced capsule

Table 5.1.1: Capsule Server Environment

The following lists the steps and their sequence followed in this reference architecture to configure load balanced capsule environment:

- Capsule “cap1” installation:
 - 1) Install operating system and configure servers – **cap1a**, **cap1b** and **cap1c**
 - Network, subscription, time synchronization and connectivity setup
 - Firewall settings
 - ISCSI storage setup
 - Create and mount shared filesystems (exclusive on one node)
 - 2) On Satellite server generate the following certificate bundles:
 - cap1.remote.bos.redhat.com-certs.tar
 - certs-capsule-lb.remote.bos.redhat.com.tar
 - 3) Copy over tar bundle “cap1.remote.bos.redhat.com-certs.tar” to server **cap1a** and install Capsule Server
 - 4) Copy over tar bundle “certs-capsule-lb.remote.bos.redhat.com.tar” to server **cap1a** and run “certs-update-all” command
 - 5) Verify the presence of new Capsule on Satellite UI
 - 6) Make a backup of the setup
 - 7) Delete the capsule from Satellite UI under Infrastructure → Capsules
 - 8) Unmount shared filesystems on **cap1a** and mount on **cap1b**
 - 9) Repeat steps 3,4 and 5 on **cap1b**
 - 10) Restore the backup content taken from **cap1a** on **cap1b**

- 11) Verify the capsule on Satellite UI. Restore the Capsule URL to ["https://cap1.remote.bos.redhat.com:9090"](https://cap1.remote.bos.redhat.com:9090)
- 12) Repeat steps 8,9, 10 and 11 for configuring **cap1c** instead of **cap1b**.
- 13) Install pacemaker, create the cluster and configure it for an active-passive service
- 14) Test failover

➤ Capsule "cap2/3/4" installation:

1. Install operating system and configure servers – **cap2**, **cap3** and **cap4**
 - Network, subscription, time synchronization and connectivity setup
 - Firewall settings
 - Create and mount local filesystems for content and services
2. On Satellite server generate the following certificate bundles:
 - cap2.remote.bos.redhat.com-certs.tar
 - cap3.remote.bos.redhat.com-certs.tar
 - cap4.remote.bos.redhat.com-certs.tar
3. Copy over tar bundle "*cap2.remote.bos.redhat.com-certs.tar*" to server **cap2** and install Capsule Server
4. Copy over tar bundle "*certs-capsule-lb.remote.bos.redhat.com.tar*" to server **cap2** and run "certs-update-all" command
5. Verify the capsule on Satellite UI. Restore the Capsule URL to ["https://cap2.remote.bos.redhat.com:9090"](https://cap2.remote.bos.redhat.com:9090)
6. Assign the Capsule to the Satellite, assign the Lifecycle Environment, sync content and verify the functionality
7. Repeat steps 3,4,5 and 6 on **cap3** and **cap4**

➤ HAProxy installation:

1. Install operating system and configure servers – **hap1-sat6** and **hap2-sat6**
 - Network, subscription, time synchronization and connectivity setup
 - Firewall settings
2. Install and configure HAProxy
3. Install pacemaker, create the cluster and configure it for an active-passive service
4. Restart HAProxy service and verify connectivity
5. On Satellite UI, create a new virtual capsule "**capsule-lb.remote.bos.redhat.com**"
6. On Satellite UI, create a new subnet or update an existing one that points to this capsule for DHCP, TFTP and Reverse DNS
7. On Satellite UI, create a new hostgroup or update an existing one that points to this capsule for Content Source, Puppet CA and Puppet Master.
8. Verify functionality by provisioning a new host using this subnet and hostgroup.



5.2 Installing Capsule Server For Foreman-Proxy Service

The Capsule Server *cap1.remote.bos.redhat.com* runs on an active-passive pacemaker cluster comprising of servers *cap1a* and *cap1b*.

5.2.1 Installing cap1a capsule server

The capsule server *cap1a.remote.bos.redhat.com* has been installed with RHEL7.2.

5.2.1.1 Hostname and Network

Modify the hostname to "*cap1.remote.bos.redhat.com*" by updating */etc/hostname* file to display "*cap1*" and modify the network to match with vip "*cap1*":

```
[root@cap1a ~]# cat /etc/hostname
cap1.remote.bos.redhat.com

[root@cap1a ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0
# Generated by dracut initrd
NAME="eth0"
ONBOOT=yes
NETBOOT=no
BOOTPROTO=static
TYPE=Ethernet
DEFROUTE=yes
IPADDR=10.19.3.51
PREFIX=24
GATEWAY=10.19.3.254
```

5.2.1.2 Subscriptions

The Capsule Server must be registered to the Red Hat Satellite Server to use the Red Hat Satellite Server products and subscriptions:

Install the Red Hat Satellite Server's CA certificate in the Capsule Server:

```
[root@cap1 ~]# rpm -Uvh \
http://sat62-ha.sysmgmt.bos.redhat.com/pub/katello-ca-consumer-latest.noarch
.rpm
```

Register the Capsule Server under your chosen organization's name:

```
[root@cap1 ~]# subscription-manager register --org=refarch
--activationkey=Bos-AK -force
```

Verify enabled repos

```
[root@cap1 ~]# subscription-manager repos --list
+-----+
  Available Repositories in /etc/yum.repos.d/redhat.repo
+-----+
Repo ID:   rhel-server-7-satellite-capsule-6.2-rpms
Repo Name: Red Hat Satellite Capsule 6 .2 (for RHEL 7 Server) (RPMs)
Repo URL:  https://cdn.redhat.com/content/beta/rhel/server/7/$basearch/sat-capsule/6/os
Enabled:   1
```

```
Repo ID:    rhel-ha-for-rhel-7-server-rpms
Repo Name:  Red Hat Enterprise Linux High Availability (for RHEL 7 Server)
            (RPMs)
Repo URL:   https://cdn.redhat.com/content/dist/rhel/server/7/$releasever/
            $basearch/highavailability/os
Enabled:    1
```

```
Repo ID:    rhel-7-server-rpms
Repo Name:  Red Hat Enterprise Linux 7 Server (RPMs)
Repo URL:   https://cdn.redhat.com/content/dist/rhel/server/7/$releasever/
            $basearch/os
Enabled:    1
```

5.2.1.3 Firewall Settings

Configure firewall settings as follows:

```
[root@cap1 ~]# firewall-cmd --permanent \
--add-port="53/udp"    --add-port="53/tcp" \
--add-port="67/udp"    --add-port="68/udp" \
--add-port="69/udp"    --add-port="80/tcp" \
--add-port="443/tcp"   --add-port="5647/tcp" \
--add-port="8000/tcp"  --add-port="8140/tcp" \
--add-port="8443/tcp"  --add-port="9090/tcp" \
--add-port="5671/tcp"  --add-port="8000/tcp" \
&& firewall-cmd --reload
```

```
[root@cap1 ~]# firewall-cmd --permanent--add-service=high-availability \
&& firewall-cmd --reload
```

5.2.1.4 Shared Filesystems

Create volume group `cap_vg` and the following shared filesystems. Refer to [ISCSI Settings](#) and [Create Shared Filesystems For HA LVM](#) for details.

```
[root@cap1a scripts]# df -h | grep cap_vg
/dev/mapper/cap_vg-lv_pulp          20G  /var/lib/pulp
/dev/mapper/cap_vg-lv_puppet        1G   /var/lib/puppet
/dev/mapper/cap_vg-lv_mongodb       15G  /var/lib/mongodb
/dev/mapper/cap_vg-lv_tftpboot      1G   /var/lib/tftpboot
/dev/mapper/cap_vg-lv_puppetenv     1G   /etc/puppet/environments
/dev/mapper/cap_vg-lv_dhcp          1G   /var/lib/dhcpd
/dev/mapper/cap_vg-lv_foreman_proxy 1G   /var/lib/foreman-proxy
/dev/mapper/cap_vg-lv_named         1G   /var/named
```

The filesystem sizes represents this environment and must be set appropriately to suit business requirements.



5.2.1.5 Capsule Server Certificate For Capsule Server “cap1”

Generate a Capsule Server certificate on the Satellite Server:

```
[root@satnode2 ~]# capsule-certs-generate \  
--capsule-fqdn cap1.remote.bos.redhat.com \  
-certs-tar /root/cap1.remote.bos.redhat.com-certs.tar  
Installing Done  
[100%] [.....] Success!  
To finish the installation, follow these steps:  
1. Ensure that the capsule-installer package is installed on the system.  
2. Copy /root/cap1.remote.bos.redhat.com-certs.tar to the system cap1  
3. Run the following commands on the capsule (possibly with the customized  
parameters, see capsule-installer --help and  
documentation for more info on setting up additional services):  
rpm -Uvh  
http://sat62-ha.sysmgmt.bos.redhat.com/pub/katello-ca-consumer-latest.noarch  
.rpm  
satellite-installer --scenario capsule \  
--parent-fqdn "sat62-ha.sysmgmt.bos.redhat.com" \  
--register-in-foreman "true" \  
--foreman-base-url "https://sat62-ha.sysmgmt.bos.redhat.com" \  
--trusted-hosts "sat62-ha.sysmgmt.bos.redhat.com" \  
--trusted-hosts "cap1.remote.bos.redhat.com" \  
--oauth-consumer-key "y4rqwt59G2LaXiB2WfcdDAdcqz86kLxR" \  
--oauth-consumer-secret "ZNY5QEPuysFh7CawGpCEEm4xdidUtbs" \  
--pulp-oauth-secret "ZUcax9X7dnTbm9WsnKZBgf8iQRJ6QQzv" \  
--certs-tar "/root/cap1.remote.bos.redhat.com-certs.tar"
```

Copy the resulting archive, “*cap1.remote.bos.redhat.com-certs.tar*”, from the Satellite Server to the Capsule Server.

```
[root@satnode2 ~]# scp -pr /root/cap1.remote.bos.redhat.com-certs.tar \  
cap1.remote.bos.redhat.com:/root
```

5.2.1.6 Capsule Server Install

On the Capsule server, load the installer:

```
[root@cap1 ~]# yum install -y satellite-capsule
```

Install the Capsule components based on the credentials generated by the Satellite server during the capsule-cert-generate run as mentioned in 61.

```
[root@cap1 ~]# satellite-installer --scenario capsule \  
--capsule-parent-fqdn "sat6ha.sysmgmt.bos.redhat.com" \  
--foreman-proxy-register-in-foreman "true" \  
--foreman-proxy-foreman-base-url "https://sat6ha.sysmgmt.bos.redhat.com" \  
--foreman-proxy-trusted-hosts "sat6ha.sysmgmt.bos.redhat.com" \  
--foreman-proxy-trusted-hosts "cap1.remote.bos.redhat.com" \  
--foreman-proxy-oauth-consumer-key "y4rqwt59G2LaXiB2WfcdDAdcqz86kLxR" \  
--foreman-proxy-oauth-consumer-secret "ZNY5QEPuysFh7CawGpCEEm4xdidUtbs" \  
--capsule-pulp-oauth-secret "ZUcax9X7dnTbm9WsnKZBgf8iQRJ6QQzv" \  
--capsule-certs-tar "/root/cap1.remote.bos.redhat.com-certs.tar" \  
--capsule-puppet "true" \  
--foreman-proxy-puppetca "true" \  
--foreman-proxy-dns-interface "eth0"
```

```

--foreman-proxy-dns           "true" \
--foreman-proxy-dns-zone     "remote.bos.redhat.com" \
--foreman-proxy-dns-forwarders "10.19.3.51" \
--foreman-proxy-dns-reverse  "3.19.10.in-addr.arpa" \
--foreman-proxy-dhcp        "true" \
--foreman-proxy-dhcp-interface "eth0" \
--foreman-proxy-dhcp-range   "10.19.3.101 10.19.3.150" \
--foreman-proxy-dhcp-gateway "10.19.3.254" \
--foreman-proxy-dhcp-nameservers "10.19.3.51" \
--foreman-proxy-tftp        "true" \
--foreman-proxy-tftp-servername "10.19.3.51"
Installing                Done
[100%] [.....] Success!
* Capsule is running at https://cap1.remote.bos.redhat.com:9090
The full log is at /var/log/capsule-installer/capsule-installer.log

```

The table Table 5.2.1.1: Capsule Configuration Options describes the command options for Capsule Server configuration and the values used in the reference architecture:

Option	Description	Value
--foreman-proxy-dns	Enable DNS proxy capability	true
--foreman-proxy-dns-interface	Which interface named should listen on	eth0
--foreman-proxy-dns-zone	The Forward DNS zone that the Satellite will host	remote.bos.r edhat.com
--foreman-proxy-dns-forwarders	The DNS server that unknown queries are forwarded to	10.19.2.51
--foreman-proxy-dns-reverse	The Reverse DNS zone the Satellite hosts. This is usually the first three octets of the IP address (172.17.13) reversed , and appended with ".in-addr.arpa".	3.19.10.in-a ddr.arpa
--foreman-proxy-dhcp	Enable DHCP proxy capability	true
--foreman-proxy-dhcp-interface	The interface that DHCP listens on	eth0
--foreman-proxy-dhcp-range	The range of IP addresses to issue to clients.	10.19.3.101 – 10.19.3.150
--foreman-proxy-dhcp-gateway	The default gateway IP to issue to clients.	10.19.3.51
--foreman-proxy-dhcp-nameservers	The host that the clients should use for name resolution. This should be configured with the Satellite's IP in this deployment model.	10.19.2.51
--foreman-proxy-tftp	Enable TFTP proxy capability. This is needed to PXE boot the clients.	true
--foreman-proxy-tftp-servername	Sets the TFTP host name. Set this to match the server's host name	10.19.3.51
--foreman-proxy-puppet	Enable the Puppet Master.	true
--foreman-proxy-puppetca	Enable the Puppet CA.	true

Table 5.2.1.1: Capsule Configuration Options



5.2.1.7 Multi-hosts Certificate for Virtual Capsule Server

Generate a certificate bundle on the Satellite Server with alternate DNS name options:

```
[root@satnode2 ~]# /root/katello-multi-host-certs.sh \  
capsule-lb.remote.bos.redhat.com cap1.remote.bos.redhat.com \  
cap2.remote.bos.redhat.com cap3.remote.bos.redhat.com \  
cap4.remote.bos.redhat.com
```

Refer to Multi-host Capsule Certificate Generation Script for “katello-multi-host-certs.sh” script.

Note: Only certificate bundle is required to be copied over to Capsules and the keys are not required.

Copy the resulting archive, “certs-capsule-lb.remote.bos.redhat.com.tar”, from the Satellite Server to the Capsule Servers **cap1a**, **cap1b**, **cap2**, **cap3** and **cap4**.

```
[root@satnode2 ~]# scp -pr \  
/root/certs-capsule-lb.remote.bos.redhat.com.tar \  
cap1.remote.bos.redhat.com:/root
```

5.2.1.8 Certificate Update

Update the Capsule server with the certificates from multi-host tar bundle:

```
[root@cap1 ~]# satellite-installer --scenario capsule --certs-tar \  
certs-capsule-lb.remote.bos.redhat.com.tar --certs-update-all  
Marking certificate  
/root/ssl-build/cap4.remote.bos.redhat.com/cap4.remote.bos.redhat.com-puppet  
-client for update  
Marking certificate  
/root/ssl-build/cap4.remote.bos.redhat.com/cap4.remote.bos.redhat.com-qpuid-b  
roker for update  
Marking certificate  
/root/ssl-build/cap4.remote.bos.redhat.com/cap4.remote.bos.redhat.com-qpuid-c  
lient-cert for update  
Marking certificate  
/root/ssl-build/cap4.remote.bos.redhat.com/cap4.remote.bos.redhat.com-qpuid-r  
outer-client for update  
Marking certificate  
/root/ssl-build/cap4.remote.bos.redhat.com/cap4.remote.bos.redhat.com-qpuid-r  
outer-server for update  
Marking certificate  
/root/ssl-build/cap4.remote.bos.redhat.com/cap4.remote.bos.redhat.com-apache  
for update  
Marking certificate  
/root/ssl-build/cap4.remote.bos.redhat.com/cap4.remote.bos.redhat.com-forema  
n-client for update  
Marking certificate  
/root/ssl-build/cap4.remote.bos.redhat.com/cap4.remote.bos.redhat.com-forema  
n-proxy for update  
Marking certificate  
/root/ssl-build/cap4.remote.bos.redhat.com/cap4.remote.bos.redhat.com-forema  
n-proxy-client for update  
Installing Done  
[100%] [.....] Success!
```

The full log is at `/var/log/foreman-installer/capsule.log`

This update modifies the Capsule URL to “<https://capsule-lb.remote.bos.redhat.com:9090>”. Restore the Capsule URL back to “<https://cap1.remote.bos.redhat.com:9090>” by performing the following step:

Infrastructure-► Capsules-► *cap1.remote.bos.redhat.com*-► Edit

5.2.1.9 Update Template_URL

Update `template_url` to point to the virtual Capsule Server

```
[root@cap1 ~]#satellite-installer scenario capsule \  
--foreman-proxy-template-url "http://capsule.remote.bos.redhat.com:8000"
```

5.2.1.10 Configuration Backup

Make a backup of the configuration using the script as mentioned in C.9

```
[root@cap1 ~]# /root/scripts/capsule-backup.sh
```

Refer to the backup script 115

This creates a backup as listed below:

```
[root@cap1a scripts]# ll /var/tmp/rhs62backup/2016-06-03-164144/  
total 161728  
-rw-r-----. 1 root root 48065692 Jun 3 16:41 config_files.tar.gz  
-rw-r-----. 1 root root 117539838 Jun 3 16:54 mongo_data.tar.gz  
drwxr-x---. 4 root root 37 Jun 3 16:56 mongo_dump  
drwxr-xr-x. 8 apache apache 94 May 26 12:09 pulp
```

Stop services on the Capsule server

```
[root@cap1 ~]# systemctl stop dhcpd.server  
[root@cap1 ~]# systemctl stop named.server  
[root@cap1 ~]# katello-service stop
```

Unmount the shared filesystems

```
[root@cap1 ~]# umount /dev/mapper/sat_vg-lv_pulp /var/lib/pulp  
[root@cap1 ~]# umount /dev/mapper/sat_vg-lv_puppet /var/lib/puppet  
[root@cap1 ~]# umount /dev/mapper/sat_vg-lv_wwwpulp /var/www/pulp  
[root@cap1 ~]# umount /dev/mapper/sat_vg-lv_mongodb /var/lib/mongodb  
[root@cap1 ~]# umount /dev/mapper/sat_vg-lv_puppetenv /etc/puppet/environments  
[root@cap1 ~]# umount /dev/mapper/sat_vg-lv_tftpboot /var/lib/tftpboot  
[root@cap1 ~]# umount /dev/mapper/sat_vg-lv_dhcp /var/lib/dhcpd  
[root@cap1 ~]# umount /dev/mapper/sat_vg-lv_named /var/named  
[root@cap1 ~]# umount /dev/mapper/sat_vg-lv_foreman_proxy /var/lib/foreman-proxy
```

Deactivate volume group `cap_vg`

```
[root@cap1 ~]# vgchange -an cap_vg
```

On the Satellite UI, delete “*cap1.remote.bos.redhat.com*” capsule since the same steps will be repeated from server “*cap1b*” and “*cap1c*”

Restore the original hostname and IP address:

```
[root@cap1 ~]# cat /etc/hostname  
cap1a.remote.bos.redhat.com
```




```
[root@cap1 ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0
# Generated by dracut initrd
NAME="eth0"
ONBOOT=yes
NETBOOT=no
BOOTPROTO=static
TYPE=Ethernet
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
PEERDNS=no
PEERROUTES=yes
IPADDR=10.19.3.48
PREFIX=24
GATEWAY=10.19.3.254
```

5.2.2 Installing cap1b capsule server

Repeat the steps as mentioned under Installing cap1a capsule server with the exception of step Configuration Backup on Capsule server - cap1b.remote.bos.redhat.com server.

Also on server cap1b, the volume group, logical volumes and filesystems do not need to be recreated. To mount the shared filesystems, create appropriate directories, activate the volume group and mount.

Activate the volume group

```
[root@cap1b ~]# vgchange -a y cap_vg
```

Create directories

```
[root@cap1b ~]# mkdir /var/lib/pulp
[root@cap1b ~]# mkdir /var/lib/puppet
[root@cap1b ~]# mkdir /var/lib/mongodb
[root@cap1b ~]# mkdir -p /etc/puppet/environments
[root@cap1b ~]# mkdir /var/lib/tftpboot
[root@cap1b ~]# mkdir /var/lib/dhcpd
[root@cap1b ~]# mkdir /var/named
```

Mount the filesystems

```
[root@cap1b ~]# mount -o _netdev /dev/mapper/cap_vg-lv_pulp /var/lib/pulp
[root@cap1b ~]# mount -o _netdev /dev/mapper/cap_vg-lv_puppet /var/lib/puppet
[root@cap1b ~]# mount -o _netdev /dev/mapper/cap_vg-lv_mongodb /var/lib/mongodb
[root@cap1b ~]# mount -o _netdev /dev/mapper/cap_vg-lv_puppetenv /etc/puppet/environments
[root@cap1b ~]# mount -o _netdev /dev/mapper/cap_vg-lv_tftpboot /var/lib/tftpboot
[root@cap1b ~]# mount -o _netdev /dev/mapper/cap_vg-lv_dhcp /var/lib/dhcpd
[root@cap1b ~]# mount -o _netdev /dev/mapper/cap_vg-lv_foreman_proxy
/var/lib/foreman-proxy
[root@cap1b ~]# mount -o _netdev /dev/mapper/cap_vg-lv_named /var/named
```

After Capsule server has been installed and verified, copy over the backup content from server **cap1a**.

```
[root@cap1b ~]# rsync --partial --progress --delete -avz /var/tmp/rhs62backup/2016-06-03-164144
10.19.3.49:/var/tmp/rhs62backup
```

Restore the backup content on **cap1b**

```
[root@cap1b ~]# /root/scripts/satellite-restore.sh /var/tmp/rhs62backup/2016-06-03-164144
```

Assign the capsule to the organization:

refarch-► Manage Organization-► refarch-► Capsules

Select cap1.... and assign it to refarch Organization

Note: This Capsule server does not require any content. Hence Lifecycle Environment must not be assigned to the capsule.

5.2.3 Installing cap1c capsule server

Repeat steps as mentioned in Installing cap1b capsule server to install **cap1c**.

5.2.4 Pacemaker Install

Perform the following steps in both cap1a and cap1b nodes.

Install pacemaker packages and start services

```
[root@cap1b ~]# yum install -y fence-agents resource-agents pcs ccs pacemaker
[root@cap1b ~]# systemctl enable pcsd
[root@cap1b ~]# systemctl start pcsd
[root@cap1b ~]# passwd hacluster
Changing password for user hacluster.
New password: xxxxxxxx
Retype new password: xxxxxx
passwd: all authentication tokens updated successfully.
```

Verify connectivity

```
[root@cap1a scripts]# pcs cluster auth cap1a cap1b cap1c
cap1b: Already authorized
cap1a: Already authorized
```

Create a pacemaker cluster with the name "cap1-cl"

```
[root@cap1a log]# pcs cluster setup --start --name cap1-cl cap1a cap1b cap1c
Shutting down pacemaker/corosync services...
Redirecting to /bin/systemctl stop pacemaker.service
Redirecting to /bin/systemctl stop corosync.service
Killing any remaining services...
Removing all cluster configuration files...
cap1a: Succeeded
cap1b: Succeeded
cap1c: Succeeded
Starting cluster on nodes: cap1a, cap1b , cap1c...
cap1b: Starting Cluster...
cap1a: Starting Cluster...
cap1c: Starting Cluster...
Synchronizing pcsd certificates on nodes cap1a, cap1b, cap1c...
cap1b: Success
cap1a: Success
cap1c: Success
```



```
Restarting pcsd on the nodes in order to reload the certificates...
cap1b: Success
cap1a: Success
cap1c: Success
```

Create fencing for the virtual machines

```
[root@cap1a log]# pcs stonith create vmfencing fence_rhevm params \
ipaddr="se-rhevm.cloud.lab.eng.bos.redhat.com" login="admin@internal" \
passwd="xxxxxx" pcmk_host_map="cap1a:cap1a;cap1b:cap1b;cap1c:cap1c" \
pcmk_host_list="cap1a,cap1b,cap1c" power_wait="5" delay="5" ssl="yes" \
ssl_insecure="1"
```

```
[root@cap1b ~]# pcs stonith show --all
vmfencing (stonith:fence_rhevm): Started cap1a
```

```
[root@cap1a log]# pcs stonith show --full
Resource: vmfencing (class=stonith type=fence_rhevm)
Attributes: ipaddr=se-rhevm.cloud.lab.eng.bos.redhat.com
login=admin@internal passwd=xxxxxx
pcmk_host_map=cap1a:cap1a;cap1b:cap1b;cap1c:cap1c;
pcmk_host_list=cap1a,cap1b,cap1c power_wait=5 delay=5 ssl=yes ssl_insecure=1
Operations: monitor interval=60s (vmfencing-monitor-interval-60s)
```

Verify the cluster status

```
[[root@cap1a log]# pcs status
Cluster name: cap1-cl
Last updated: Fri Jun 10 16:33:02 2016      Last change: Fri Jun 10
16:13:56 2016 by root via cibadmin on cap1a
Stack: corosync
Current DC: cap1a (version 1.1.13-10.el7_2.2-44eb2dd) - partition with
quorum
2 nodes and 1 resource configured

Online: [ cap1a cap1b cap1c ]

Full list of resources:

vmfencing (stonith:fence_rhevm): Started cap1a

PCSD Status:
cap1a: Online
cap1b: Online
cap1c: Online
Daemon Status:
corosync: active/enabled
pacemaker: active/enabled
pcsd: active/enabled
```

5.2.4.1 Exclusive Activation Of Shared Volume Group

After ensuring the Capsule services are down, and the filesystems have been unmounted on all the nodes, perform the following on all the nodes:

```
[root@cap1a/b ~]# vgchange -a n cap_vg

[root@cap1a/b ~]# lvchange -aey cap_vg
```

Ensure that the shared volume group does not get activated at boot time and is activated only by pacemaker. This is achieved by making an entry of all volume groups that must be auto activated at boot time in the `/etc/lvm/lvm.conf` file.

In this environment, only “**myvg**” volume group exists and must be auto-activated.

```
[root@cap1a/b ~]# vgs | grep -v cap_vg
VG   #PV #LV #SN Attr   VSize  VFree
myvg  1  3  0 wz--n- 185.82g 60.00m
```

Edit the file to add “**myvg**” in the “`volume_list`” variable.

```
[root@cap1a/b]# cat /etc/lvm/lvm.conf | grep "volume_list =" | grep -v "#"
volume_list = [ "myvg" ]
```

```
[root@cap1a ~]# dracut -H -f /boot/initramfs-$(uname -r).img $(uname -r)
```

Ensure this change is reflected in the kernel and reboot the server:

```
[root@cap1a ~]# reboot
```

5.2.4.2 Creating Pacemaker Resources

Prework:

The resources must be created from any one node. **cap1a** in this case.

It is ideal to disable cap1b before creating resources. This step helps manually ensure that all the services are started up on the first node, instead of being scattered across different nodes. Resource constraints created in subsequent steps, will handle this automatically.

```
[root@cap1a ~]# pcs cluster standby cap1b cap1c
```

Create Resources

1. Virtual IP resource:

```
[root@cap1a ~]# pcs resource create VirtualIP IPAddr2 ip=10.19.3.51 cidr_netmask=24
```

2. Volume group

```
[root@cap1a ~]# pcs resource create cap_vg LVM volgrpname=cap_vg exclusive=true
```

3. Filesystems:

```
[root@cap1a]# pcs resource create fs_pulp Filesystem device="/dev/cap_vg/lv_pulp"
directory="/var/lib/pulp" fstype="xfs"
```

```
[root@cap1a]# pcs resource create fs_puppet Filesystem device="/dev/cap_vg/lv_puppet"
directory="/var/lib/puppet" fstype="xfs"
```

```
[root@cap1a]# pcs resource create fs_mongodb Filesystem
device="/dev/cap_vg/lv_mongodb" directory="/var/lib/mongodb" fstype="xfs"
```

```
[root@cap1a]# pcs resource create fs_puppetenv Filesystem
device="/dev/cap_vg/lv_puppetenv" directory="/etc/puppet/environments" fstype="xfs"
```

```
[root@cap1a]# pcs resource create fs_tftpboot Filesystem
device="/dev/cap_vg/lv_tftpboot" directory="/var/lib/tftpboot" fstype="xfs"
```

```
[root@cap1a]# pcs resource create fs_dhcp Filesystem device="/dev/cap_vg/lv_dhcp"
directory="/var/lib/dhcpd" fstype="xfs"
```

```
[root@cap1a]# pcs resource create fs_named Filesystem device="/dev/cap_vg/lv_named"
directory="/var/named" fstype="xfs"
```

```
[root@cap1a]# pcs resource create fs_foreman_proxy Filesystem
device="/dev/cap_vg/lv_foreman_proxy" directory="/var/lib/foreman-proxy" fstype="xfs"
```

4. Services

```
[root@cap1a]# pcs resource create rs_dhcp systemd:dhcpd op monitor interval=10s op
start interval=0s timeout=100s op stop interval=0s timeout=100s
```

```
[root@cap1a]# pcs resource create rs_named systemd:named op monitor interval=10s op
start interval=0s timeout=100s op stop interval=0s timeout=100s
```

```
[root@cap1a]# pcs resource create rs_puppet systemd:puppet op monitor interval=10s op
start interval=0s timeout=100s op stop interval=0s timeout=100s
```

```
[root@cap1a]# pcs resource create rs_mongodb systemd:mongod op monitor
interval=10s op start interval=0s timeout=100s op stop interval=0s timeout=100s
```

```
[root@cap1a]# pcs resource create rs_qpidd systemd:qpidd op monitor interval=10s op
start interval=0s timeout=100s op stop interval=0s timeout=100s
```

```
[root@cap1a]# pcs resource create rs_qdrouterd systemd:qdrouterd op monitor
interval=10s op start interval=0s timeout=100s op stop interval=0s timeout=100s
```

```
[root@cap1a]# pcs resource create rs_pulp_workers systemd:pulp_workers op monitor
interval=10s op start interval=0s timeout=100s op stop interval=0s timeout=100s
```

```
[root@cap1a]# pcs resource create rs_foreman_proxy systemd:foreman-proxy op monitor
interval=10s op start interval=0s timeout=100s op stop interval=0s timeout=100s
```

```
[root@cap1a]# pcs resource create rs_pulp_resource_manager
systemd:pulp_resource_manager op monitor interval=10s op start interval=0s
timeout=100s op stop interval=0s timeout=100s
```

```
[root@cap1a]# pcs resource create rs_pulp_celerybeat systemd:pulp_celerybeat op
monitor interval=10s op start interval=0s timeout=100s op stop interval=0s timeout=100s
```

```
[root@cap1a]# pcs resource create rs_httpd systemd:httpd op monitor interval=10s op
start interval=0s timeout=100s op stop interval=0s timeout=100s
```

5. Constraints

Colocation Constraints:

```
[root@cap1a]# pcs constraint colocation add cap_vg with VirtualIP
```

```
[root@cap1a]# pcs constraint colocation set fs_dhcp fs_named fs_puppet fs_puppetenv
```

```
fs_mongodb fs_pulp fs_tftpboot fs_foreman_proxy sequential=false set cap_vg
[root@cap1a]# pcs constraint colocation add rs_dhcp with fs_dhcp
[root@cap1a]# pcs constraint colocation add rs_named with fs_named
[root@cap1a]# pcs constraint colocation add rs_mongodb with fs_mongodb
[root@cap1a]# pcs constraint colocation add rs_foreman_proxy with fs_foreman_proxy
[root@cap1a]# pcs constraint colocation set fs_puppet fs_puppetenv rs_puppet
[root@cap1a]# pcs constraint colocation set VirtualIP rs_qpidd rs_qdrouterd
[root@cap1a]# pcs constraint colocation add rs_httpd with VirtualIP
```

6. Order constraints to start service after the filesystems

```
[root@cap1a]# pcs constraint order set VirtualIP rs_qpidd rs_qdrouterd
[root@cap1a]# pcs constraint order VirtualIP then rs_dhcp
[root@cap1a]# pcs constraint order VirtualIP then rs_named
[root@cap1a]# pcs constraint order set rs_puppet rs_dhcp rs_named sequential=false set
rs_mongodb set rs_qpidd rs_qdrouterd sequential=false set rs_pulp_workers
rs_pulp_celerybeat rs_pulp_resource_manager rs_foreman_proxy sequential=false set
rs_httpd sequential=false
```

7. Set default resource stickiness to avoid failback when the original node is restored

```
[root@cap1a]# pcs property set default-resource-stickiness=INFINITY
```

For more details on creating pacemaker resource constraints, please refer to Pacemaker resource constraint documentation⁵

5.2.4.3 Pacemaker Cluster Status

Activate the all the nodes back in the cluster:

```
[root@cap1a ~]# pcs cluster unstandby cap1b cap1c
```

Confirm that all the nodes are active:

```
[root@cap1a scripts]# pcs status
Cluster name: cap1-cl
Last updated: Wed Jun 15 16:59:05 2016          Last change: Sat Jun 11 23:01:32 2016 by root via
crm_attribute on cap1a
Stack: corosync
Current DC: cap1b (version 1.1.13-10.el7_2.2-44eb2dd) - partition with quorum
Current DC: cap1c (version 1.1.13-10.el7_2.2-44eb2dd) - partition with quorum
3 nodes and 22 resources configured
```

⁵ https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html-single/Configuring_the_Red_Hat_High_Availability_Add-On_with_Pacemaker/index.html#ch-resourceconstraints-HAAR



Online: [cap1a cap1b cap1c]

Full list of resources:

```
vmfencing      (stonith:fence_rhevm): Started cap1a
VirtualIP      (ocf::heartbeat:IPAddr2): Started cap1a
cap_vg         (ocf::heartbeat:LVM): Started cap1a
fs_pulp        (ocf::heartbeat:Filesystem): Started cap1a
fs_puppet      (ocf::heartbeat:Filesystem): Started cap1a
fs_mongodb     (ocf::heartbeat:Filesystem): Started cap1a
fs_puppetenv   (ocf::heartbeat:Filesystem): Started cap1a
fs_tftpboot    (ocf::heartbeat:Filesystem): Started cap1a
fs_dhcp        (ocf::heartbeat:Filesystem): Started cap1a
fs_named       (ocf::heartbeat:Filesystem): Started cap1a
fs_foreman_proxy (ocf::heartbeat:Filesystem): Started cap1a
rs_dhcp        (systemd:dhcpd): Stopped
rs_named       (systemd:named): Stopped
rs_puppet      (systemd:puppet): Stopped
rs_mongodb     (systemd:mongod): Stopped
rs_qpidd       (systemd:qpidd): Stopped
rs_qdrouterd   (systemd:qdrouterd): Stopped
rs_pulp_workers (systemd:pulp_workers): Stopped
rs_foreman_proxy (systemd:foreman-proxy): Stopped
rs_pulp_resource_manager (systemd:pulp_resource_manager): Stopped
rs_pulp_celerybeat (systemd:pulp_celerybeat): Stopped
rs_httpd       (systemd:httpd): Stopped
```

PCSD Status:

```
cap1a: Online
cap1b: Online
cap1c: Online
```

Daemon Status:

```
corosync: active/enabled
pacemaker: active/enabled
pcsd: active/enabled
```

5.3 Installing Capsule Servers For Content Services

5.3.1 Installing Capsule Server cap2

The capsule server cap2.remote.bos.redhat.com has been installed with RHEL7.2 with adequate storage/filesystems for content and services.

```
[root@cap2 ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0
# Generated by dracut initrd
NAME="eth0"
ONBOOT=yes
NETBOOT=no
BOOTPROTO=static
TYPE=Ethernet
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
PEERDNS=no
PEERROUTES=yes
IPADDR=10.19.3.52
PREFIX=24
GATEWAY=10.19.3.254
```

5.3.1.1 Subscriptions

The Capsule Server must be registered to the Red Hat Satellite Server to use the Red Hat Satellite Server products and subscriptions:

Install the Red Hat Satellite Server's CA certificate in the Capsule Server:

```
[root@cap2 ~]# rpm -Uvh
http://sat62-ha.sysmgmt.bos.redhat.com/pub/katello-ca-consumer-latest.noarch
.rpm
```

Register the Capsule Server under your chosen organization's name:

```
[root@cap1 ~]# subscription-manager register --org=refarch
--activationkey=Bos-AK -force
```

```
[root@cap2 ~]# subscription-manager repos --list
+-----+
  Available Repositories in /etc/yum.repos.d/redhat.repo
+-----+
Repo ID:    rhel-server-7-satellite-capsule-6.2-rpms
Repo Name:  Red Hat Satellite Capsule 6 .2 (for RHEL 7 Server) (RPMs)
Repo URL:   https://cdn.redhat.com/content/beta/rhel/server/7/$basearch/sat-capsule/6/os
Enabled:    1

Repo ID:    rhel-ha-for-rhel-7-server-rpms
Repo Name:  Red Hat Enterprise Linux High Availability (for RHEL 7 Server)
(RPMs)
Repo URL:   https://cdn.redhat.com/content/dist/rhel/server/7/$releasever/
$basearch/highavailability/os
Enabled:    1
```




```
Repo ID:    rhel-7-server-rpms
Repo Name:  Red Hat Enterprise Linux 7 Server (RPMs)
Repo URL:   https://cdn.redhat.com/content/dist/rhel/server/7/$releasever/
           $basearch/os
Enabled:    1
```

5.3.1.2 Firewall Settings

```
[root@cap2 ~]# firewall-cmd --permanent \
--add-port="80/tcp" --add-port="443/tcp" \
--add-port="5647/tcp" --add-port="8000/tcp" \
--add-port="8140/tcp" --add-port="8443/tcp" \
--add-port="9090/tcp" && firewall-cmd --reload
```

5.3.1.3 Capsule Server Certificate For Capsule Server “cap2”

Generate a Capsule Server certificate on the Satellite Server:

```
[root@satnode2 ~]# capsule-certs-generate \
--capsule-fqdn cap2.remote.bos.redhat.com \
--certs-tar /root/cap2.remote.bos.redhat.com-certs.tar
Installing Done
[100%] [.....]
Success!
To finish the installation, follow these steps:
1. Ensure that the capsule-installer package is installed on the system.
2. Copy /root/cap2.remote.bos.redhat.com-certs.tar to the system
cap1.remote.bos.redhat.com
3. Run the following commands on the capsule (possibly with the customized
parameters, see capsule-installer --help and
documentation for more info on setting up additional services):

rpm -Uvh \
http://sat62-ha.sysmgmt.bos.redhat.com/pub/katello-ca-consumer-latest.noarch
.rpm
subscription-manager register --org "Default_Organization"
satellite-installer --scenario capsule \
--parent-fqdn "sat62-ha.sysmgmt.bos.redhat.com" \
--register-in-foreman "true" \
--foreman-base-url "https://sat62-ha.sysmgmt.bos.redhat.com" \
--trusted-hosts "sat62-ha.sysmgmt.bos.redhat.com" \
--trusted-hosts "cap1.remote.bos.redhat.com" \
--oauth-consumer-key "y4rqwt59G2LaXiB2WfcdDAdcqz86kLxR" \
--oauth-consumer-secret "ZNY5QEPuysFh7CawGpCEEm4xdidUtbs" \
--pulp-oauth-secret "ZUcax9X7dnTbm9WsnKZBgf8iQRJ6QQzv" \
--certs-tar "/root/cap1.remote.bos.redhat.com-certs.tar"
satellite-installer --scenario capsule \
```

Copy the resulting archive, “*cap2.remote.bos.redhat.com-certs.tar*”, from the Satellite Server to the Capsule Server.

```
[root@satnode2 ~]# scp -pr /root/cap2.remote.bos.redhat.com-certs.tar \
cap2.remote.bos.redhat.com:/root
```

5.3.1.4 Capsule Server Install

On the Capsule server, load the installer:

```
[root@cap2 ~]# yum install -y satellite-capsule
```

Install the Capsule components based on the credentials generated by the Satellite server during the capsule-cert-generate run as mentioned in Capsule Server Certificate For Capsule Server “cap2”

```
[root@cap2 ~]# satellite-installer --scenario capsule \
--capsule-parent-fqdn "sat6ha.sysmgmt.bos.redhat.com" \
--foreman-proxy-register-in-foreman "true" \
--foreman-proxy-foreman-base-url "https://sat6ha.sysmgmt.bos.redhat.com" \
--foreman-proxy-trusted-hosts "sat6ha.sysmgmt.bos.redhat.com" \
--foreman-proxy-trusted-hosts "cap2.remote.bos.redhat.com" \
--foreman-proxy-oauth-consumer-key "y4rqwt59G2LaXiB2WfcdDAdcqz86kLxR" \
--foreman-proxy-oauth-consumer-secret "ZNY5QEPuysFh7CawGpCEEm4xdidUtbsc" \
--capsule-pulp-oauth-secret "ZUcax9X7dnTbm9WsnKZBgf8iQRJ6QQzv" \
--capsule-certs-tar "/root/cap2.remote.bos.redhat.com-certs.tar"
```

```
Installing Done
[100%]
[.....]
Success!
* Capsule is running at https://cap2.remote.bos.redhat.com:9090
The full log is at /var/log/capsule-installer/capsule-installer.log
```

5.3.1.5 Certificate update

Update the multi-host certificate as mentioned in Certificate Update

5.3.1.6 Assign Capsule and Synchronize Content

Assign the capsule to the organization:

refarch-► Manage Organization-► refarch-► Capsules

Select cap2.... and assign it to refarch Organization

Assign Lifecycle Environment to “Library” and Submit.

Infrastructure-► Capsules-► cap1.remote.bos.redhat.com-► Edit-► Lifecycle Environment

Select **cap2.remote.bos.redhat.com** in the capsules list and select Synchronize.

5.3.1.7 Installing Capsule Server cap3 and cap4

Repeat the steps as mentioned in Installing Capsule Server cap2 for Capsule servers cap3, cap4 and on any additional Capsule servers.



5.4 HAProxy installation

HAProxy service is installed on a three node pacemaker cluster in active-passive mode running RHEL 7.2.

Cluster node - **hap1-sat6.remote.bos.redhat.com** (10.19.3.81)

Cluster node - **hap2-sat6.remote.bos.redhat.com** (10.19.3.82)

Cluster node - **hap3-sat6.remote.bos.redhat.com** (10.19.3.83)

Virtual server - **capsule-lb..remote.bos.redhat.com** (10.19.3.50)

Note: The loadbalancer functionality in this environment is provided by HAProxy. This setup can be complimented by any other hardware based or software based loadbalancer provider. Also high availability of HAProxy can be configured with pacemaker active-passive cluster instead of using keepalived.

5.4.1 Subscription

```
[root@hap1-sat6 ~]# subscription-manager repos --list-enabled
+-----+
Available Repositories in /etc/yum.repos.d/redhat.repo
+-----+
Repo ID:    rhel-ha-for-rhel-7-server-rpms
Repo Name:  Red Hat Enterprise Linux High Availability (for RHEL 7 Server)
            (RPMs)
Repo URL:   https://cdn.redhat.com/content/dist/rhel/server/7/$releasever/
            $basearch/highavailability/os
Enabled:    1

Repo ID:    rhel-7-server-rpms
Repo Name:  Red Hat Enterprise Linux 7 Server (RPMs)
Repo URL:   https://cdn.redhat.com/content/dist/rhel/server/7/$releasever/
            $basearch/os
Enabled:    1
```

5.4.2 Firewall Settings

```
[root@hap1 ~]# firewall-cmd --permanent \
--add-port="80/tcp" --add-port="443/tcp" \
--add-port="8443/tcp" && firewall-cmd --reload

[root@hap1 ~]# firewall-cmd --permanent--add-service=high-availability \
&& firewall-cmd --reload
```

5.4.3 Shared Filesystems

Create volume group hap_vg and the following shared filesystems. Refer to ISCSI Settings and Create Shared Filesystems For HA LVM for details.

```
[root@hap1--sat6 ~]# df -h | grep cap_vg
/dev/mapper/hap_vg-lv_log          10G  /var/log/haproxy
/dev/mapper/hap_vg-lv_haproxy     1G   /etc/haproxy
```

The filesystem sizes represents this environment and must be set appropriately to suit business requirements.

5.4.4 Install and Configure HAProxy

```
[root@hap1-sat6 ~]# yum install haproxy
```

5.4.4.1 HAProxy configuration:

Foreman-proxy service (port 9090) and qpid (port 5671) are pointed to cap1.

Apache (port 80,443,8443) and puppet (port 8140) are pointed to cap2, cap3 and cap4 Capsules.

```
[root@hap1-sat6 ~]# cat /etc/haproxy/haproxy.cfg
```

```
global
  log          127.0.0.1 local2
  pidfile      /var/run/haproxy.pid
  maxconn      4000
  user         haproxy
  group        haproxy
  daemon

  # turn on stats unix socket
  stats socket /var/lib/haproxy/stats

defaults
  log          global
  option      tcplog
  option      dontlognull
  option      redispatch
  retries     3
  timeout http-request 10s
  timeout queue 1m
  timeout connect 10s
  timeout client 1m
  timeout server 1m
  timeout check 10s
  maxconn     3000

#
listen apache
  bind 10.19.3.50:80
  mode tcp
  balance roundrobin
  server cap2 10.19.3.52:80 check inter 1s
  server cap3 10.19.3.53:80 check inter 1s
  server cap4 10.19.3.54:80 check inter 1s

#
listen puppet
  bind 10.19.3.50:8140
  mode tcp
  option tcplog
  option      ssl-hello-chk
  balance roundrobin
  server cap2 10.19.3.52:8140 check inter 1s
```

```
server cap3 10.19.3.53:8140 check inter 1s
server cap4 10.19.3.54:8140 check inter 1s
#
listen apache-443
  bind 10.19.3.50:443
  mode tcp
  option tcplog
  balance roundrobin
  server cap2 10.19.3.52:443 check inter 1s
  server cap3 10.19.3.53:443 check inter 1s
  server cap4 10.19.3.54:443 check inter 1s
#
listen apache-8443
  bind 10.19.3.50:8443
  mode tcp
  option tcplog
  balance roundrobin
  server cap2 10.19.3.52:8443 check inter 1s
  server cap3 10.19.3.53:8443 check inter 1s
  server cap4 10.19.3.54:8443 check inter 1s
#
listen foreman-proxy
  bind 10.19.3.50:9090
  mode tcp
  option tcplog
  balance roundrobin
  server cap1 10.19.3.51:9090 check inter 1s
#
listen qpid
  bind 10.19.3.50:5671
  mode tcp
  option tcplog
  balance roundrobin
  server cap1 10.19.3.51:5671 check inter 1s
#
listen template
  bind 10.19.3.50:8000
  mode tcp
  option tcplog
  balance roundrobin
  server cap1 10.19.3.51:8000 check inter 1s
```

5.4.4.2 HAProxy logging

Add logging for HAProxy

```
[root@hap1-sat6 ~]# echo "local2.* /var/log/haproxy/haproxy.log" >> \
/etc/sysconfig/rsyslog
[root@hap1-sat6 ~]# touch /var/log/haproxy/haproxy.log
[root@hap1-sat6 ~]# systemctl restart rsyslog.conf
[root@hap1-sat6 ~]# systemctl restart haproxy
```

Note: This reference architecture describes a functional configuration of HAProxy. Additional configuration may be required to utilize web based HAProxy log outputs.

5.4.4.3 Manually Adding Virtual IP

```
[root@hap1-sat6 log]# ip addr add 10.19.3.50 dev eth0

[root@hap1-sat6 log]# ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP qlen 1000
    link/ether 00:01:a4:ac:32:aa brd ff:ff:ff:ff:ff:ff
    inet 10.19.3.81/24 brd 10.19.3.255 scope global eth0
        valid_lft forever preferred_lft forever
    inet 10.19.3.50/32 scope global eth0
        valid_lft forever preferred_lft forever
    inet 10.19.3.50/24 scope global secondary eth0
        valid_lft forever preferred_lft forever
    inet6 2620:52:0:1303:201:a4ff:feac:32aa/64 scope global mngtmpaddr
dynamic
    valid_lft 2591908sec preferred_lft 604708sec
    inet6 fe80::201:a4ff:feac:32aa/64 scope link
        valid_lft forever preferred_lft forever
```

5.4.4.4 Start HAProxy Services

```
[root@hap1-sat6 ~]# systemctl start haproxy.service
```

Verify if the ports are being forwarded to the right server

```
[root@hap1-sat6 log]# telnet capsule-lb 8443
Trying 10.19.3.50...
Connected to capsule-lb.
Escape character is '^']
```

Stop Virtual IP, unmount shared filesystems, mount them on node hap2-sat6 and repeat the installation as mentioned in steps Manually Adding Virtual IP and Start HAProxy Services

5.4.5 Pacemaker Install

Perform the following steps in both hap1-sat6 and hap2-sat6 and hap3-sat6 nodes.

Install pacemaker packages and start services

```
[root@hap1-sat6 ~]# yum install -y fence-agents resource-agents pcs ccs
pacemaker
[root@hap1-sat6 ~]# systemctl enable pcsd
[root@hap1-sat6 ~]# systemctl start pcsd
[root@hap1-sat6 ~]# passwd hacluster
Changing password for user hacluster.
New password: xxxxxxxx
Retype new password: xxxxxx
passwd: all authentication tokens updated successfully.
```

Verify connectivity

```
[root@hap1-sat6 ~]# pcs cluster auth hap1-sat6 hap2-sat6 hap3-sat6
hap1-sat6: Already authorized
hap2-sat6: Already authorized
```



```
hap3-sat6: Already authorized
```

Create a pacemaker cluster with the name “hap1-cl”

```
[root@cap1a log]# pcs cluster setup --start --name hap-cl hap1-sat6 hap2-sat6
hap3-sat6
Shutting down pacemaker/corosync services...
Redirecting to /bin/systemctl stop pacemaker.service
Redirecting to /bin/systemctl stop corosync.service
Killing any remaining services...
Removing all cluster configuration files...
hap1-sat6: Succeeded
hap1-sat6: Succeeded
hap1-sat6: Succeeded
Starting cluster on nodes: hap1-sat6 , hap2-sat6 , hap3-sat6 ...
hap1-sat6: Starting Cluster...
hap2-sat6: Starting Cluster...
hap3-sat6: Starting Cluster...
Synchronizing pcsd certificates on nodes hap1-sat6 , hap2-sat6 , hap3-sat6 ...
hap1-sat6: Succeeded
hap2-sat6: Succeeded
hap3-sat6: Succeeded
```

Create fencing for the virtual machines

```
[root@hap1-sat6 ]# pcs stonith create vmfencing fence_rhevm params \
ipaddr="se-rhevm.cloud.lab.eng.bos.redhat.com" login="admin@internal" \
passwd="xxxxxx"
pcmk_host_map="hap1-sat6:hap1-sat6;hap2-sat6:hap2-sat6;hap3-sat6:hap3-sat6"\
pcmk_host_list="hap1-sat6,hap2-sat6,hap3-sat6" power_wait="5" delay="5"
ssl="yes" \ ssl_insecure="1"

[root@hap1-sat6 ]# pcs stonith show --all
vmfencing (stonith:fence_rhevm): Started hap1-sat6

[root@cap1a log]# pcs stonith show --full
Resource: vmfencing (class=stonith type=fence_rhevm)
Attributes: ipaddr=se-rhevm.cloud.lab.eng.bos.redhat.com
login=admin@internal passwd=xxxxxx
pcmk_host_map="hap1-sat6:hap1-sat6;hap2-sat6:hap2-sat6;hap3-sat6:hap3-sat6";
pcmk_host_list=hap1-sat6,hap2-sat6, hap3-sat6 power_wait=5 delay=5 ssl=yes
ssl_insecure=1
Operations: monitor interval=60s (vmfencing-monitor-interval-60s)
```

5.4.5.1 Exclusive Activation Of Shared Volume Group

Follow the same steps for volume group hap_vg as mentioned in Exclusive Activation Of Shared Volume Group

5.4.5.2 Creating Pacemaker Resources

Prework:

The resources must be created from any one node. **hap1-sat6** in this case.

It is ideal to disable hap2-sat6 and hap3-sat6 before creating resources. This step helps manually ensure that all the services are started up on the first node, instead of being scattered across different nodes. Resource constraints created in subsequent steps, will handle this automatically.

```
[root@cap1a ~]# pcs cluster standby hap2-sat6 hap2-sat6
```

Create Resources

1. Virtual IP resource:

```
[root@hap1-sat6 ~]# pcs resource create VirtualIP IPAddr2 ip=10.19.3.50 cidr_netmask=24
```

2. Volume group

```
[root@hap1-sat6 ~]# pcs resource create hap_vg LVM volgrpname=hap_vg exclusive=true
```

3. Filesystems:

```
[root@hap1-sat6 ~]# pcs resource create fs_log Filesystem device="/dev/hap_vg/lv_log" \
directory="/var/log/haproxy" fstype="xfs"
```

```
[root@hap1-sat6 ~]# pcs resource create fs_haproxy Filesystem \
device="/dev/hap_vg/lv_haproxy" directory="/etc/haproxy" fstype="xfs"
```

4. Services

```
[root@hap1-sat6 ~]# pcs resource create rs_haproxy systemd:haproxy op monitor \
interval=10s op start interval=0s timeout=100s op stop interval=0s timeout=100s
```

5. Constraints

Colocation Constraints:

```
[root@hap1-sat6 ~]# pcs constraint colocation add hap_vg with VirtualIP
```

```
[root@cap1a ~]# pcs constraint colocation set fs_log fs_haproxy sequential=false set \
hap_vg
```

```
[root@cap1a ~]# pcs constraint colocation add rs_haproxy with fs_log
```

```
[root@cap1a ~]# pcs constraint colocation set VirtualIP rs_haproxy
```

6. Order constraints to start service after the filesystems

```
[root@cap1a ~]# pcs constraint order VirtualIP then rs_named
```

7. Set default resource stickiness to avoid failback when the original node is restored

```
[root@cap1a ~]# pcs property set default-resource-stickiness=INFINITY
```

5.4.5.3 Pacemaker Cluster Activate Nodes

Activate the all the nodes back in the cluster:

```
[root@cap1a ~]# pcs cluster unstandby hap2-sat6 hap3-sat6
```




5.4.5.4 Creating Virtual Capsule Server

Infrastructure-► Capsules-► New Capsule

Enter the values:

Name – capsule-lb.remote.bos.redhat.com

URL – <https://capsule-lb.remote.bos.redhat.com:9090>

Lifecycle Environment – Do not assign anything since there is no content sync required

Locations – remote

Organization – refarch

Note: When creating virtual Capsule Server, URL <https://capsule-lb.remote.bos.redhat.com:9090> should be able to return an active foreman-proxy environment (in this case running on **cap1**). Hence, virtual Capsule must be created after HAProxy or any other loadbalancer is functional to redirect **capsule-lb:9090** to foreman-proxy service **cap1:9090**.

List all the configured Capsule servers:

```
[root@sat62-ha ~]# hammer -u admin -p xxxxxx - capsule list
```

ID	NAME	URL	FEATURES
4	cap1	https://cap1.remote.bos.redhat.com:9090	Templates, Pulp Node, ...
3	cap2	https://cap2.remote.bos.redhat.com:9090	Templates, Pulp Node, ...
6	cap3	https://cap3.remote.bos.redhat.com:9090	Templates, Pulp Node, ...
8	cap4	https://cap4.remote.bos.redhat.com:9090	Templates, Pulp Node, ...
7	capsule-lb	https://capsule-lb.remote.bos.redhat.com:9090	Templates, Pulp Node, ...
1	sat62-ha	https://sat62-ha.sysmgmt.bos.redhat.com:9090	Pulp, Puppet, Puppet C...

6 Provisioning

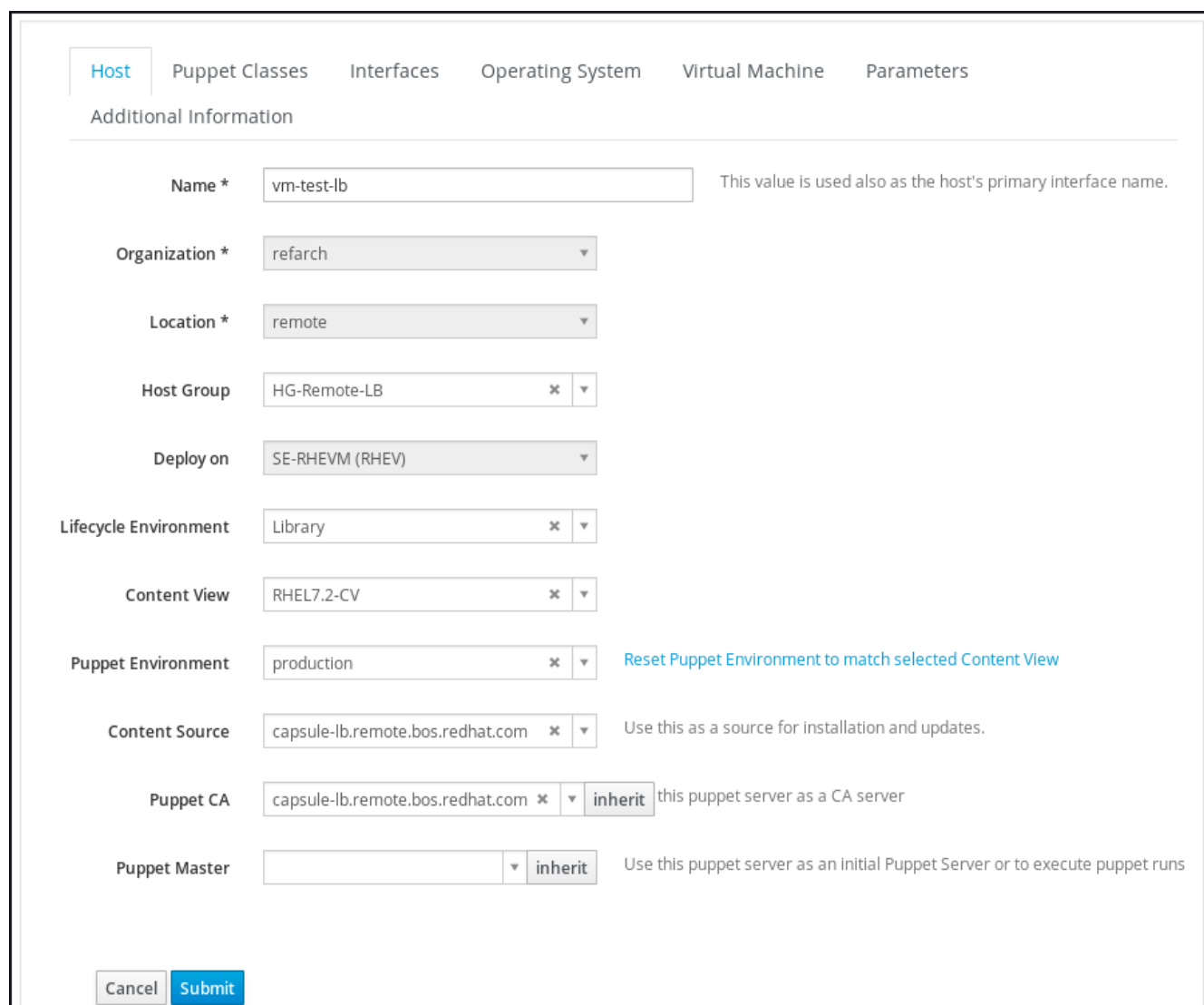
6.1 Provisioning A Host Using PXE

Create a new host:

Hosts - ► New Host

Enter the values as displayed in the illustration

Host information



The screenshot shows a web form for creating a new host. At the top, there are tabs: Host (selected), Puppet Classes, Interfaces, Operating System, Virtual Machine, and Parameters. Below the tabs is a section titled "Additional Information". The form contains several fields:

- Name ***: Input field with "vm-test-lb". A note says: "This value is used also as the host's primary interface name."
- Organization ***: Dropdown menu with "refarch" selected.
- Location ***: Dropdown menu with "remote" selected.
- Host Group**: Input field with "HG-Remote-LB" and a dropdown arrow.
- Deploy on**: Dropdown menu with "SE-RHEVM (RHEV)" selected.
- Lifecycle Environment**: Input field with "Library" and a dropdown arrow.
- Content View**: Input field with "RHEL7.2-CV" and a dropdown arrow.
- Puppet Environment**: Input field with "production" and a dropdown arrow. A link says: "Reset Puppet Environment to match selected Content View".
- Content Source**: Input field with "capsule-lb.remote.bos.redhat.com" and a dropdown arrow. A note says: "Use this as a source for installation and updates."
- Puppet CA**: Input field with "capsule-lb.remote.bos.redhat.com" and a dropdown arrow, followed by a button labeled "inherit". A note says: "this puppet server as a CA server".
- Puppet Master**: Input field with a dropdown arrow, followed by a button labeled "inherit". A note says: "Use this puppet server as an initial Puppet Server or to execute puppet runs".

At the bottom left, there are two buttons: "Cancel" and "Submit".

Illustration 25: Provisioning – Host Information

Network information

Provide the network interface device and network information

Interface ✕

Type

MAC address

Device identifier ⓘ

DNS name ⓘ

Domain

Subnet ✕ ▾

IP address ⓘ
[Suggest new](#)

Managed ⓘ

Primary ⓘ

Provision ⓘ

Remote execution ⓘ

Virtual NIC ⓘ

RHEV

Name

Network ▾

Illustration 26: Provisioning - Network Information

Operating System Information

New Host | vm-test-lb.remote.bos.redhat.com

Host Puppet Classes Interfaces **Operating System** Virtual Machine Parameters Additional Information

Architecture * x86_64 x ▼

Operating system * RedHat 7.2 x ▼

Provisioning Method * Network Based Image Based

Build mode Enable this host for provisioning

Media Selection Synced Content All Media
Select the installation media that will be used to provision this host. Choose 'Synced Content' for Synced Kickstart Repositories or 'All Media' for other media.

Synced Content Red Hat Enterprise Linux 7 Server Kickstart x86_64 7.2 x ▼

Partition table * ▼

Custom partition table
What ever text(or ERB template) you use in here, would be used as your OS disk layout options If you want to

Root password * Password must

Provisioning templates
Display the templates that will be used to provision this host

Illustration 27: Provisioning - OS Information

Virtual Machine Information

New Host | vm-test-lb.remote.bos.redhat.com

Host Puppet Classes Interfaces Operating System **Virtual Machine** Parameters Additional Information

Cluster Syseng

Template Select template
RHEV/RHEV template to use

Cores 1

Memory 1 GB

Start Power ON this machine

Network interfaces
Network interfaces management has been moved to the interfaces tab. Please set your interfaces there.

Storage

Size (GB) 10

Storage domain Data-MD3000I

Preallocate disk Uses thin provisioning if unchecked

Bootable Only one volume can be bootable

+ Add Volume

Cancel Submit

Illustration 28: Provisioning - VM Information

Parameter

Host Puppet Classes Interfaces Operating System Virtual Machine **Parameters** Additional Information

Puppet class parameters

Puppet class	Name	Value
--------------	------	-------

Global parameters

Name	Value
kt_activation_keys	<input type="button" value="⊕"/> Bos-AK

Host parameters

Name	Value
------	-------

+ Add Parameter

Illustration 29: Provisioning - Activation Key

Virtual Host creation

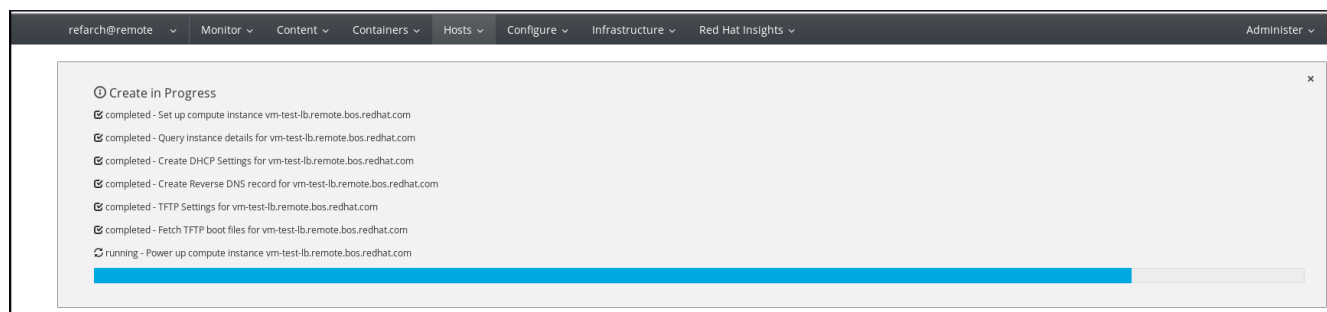


Illustration 30: Provisioning - Host Creation

New Host Information

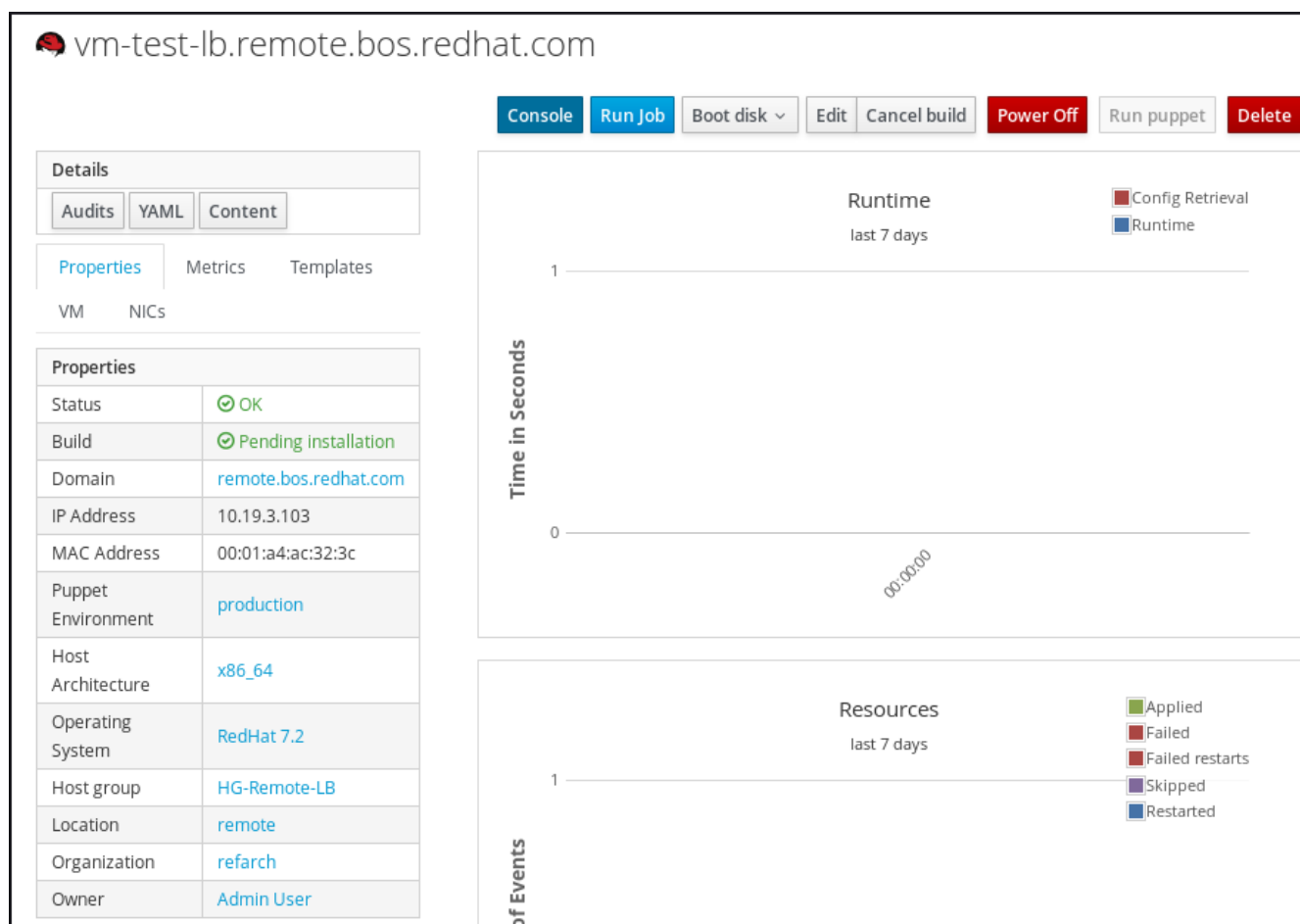
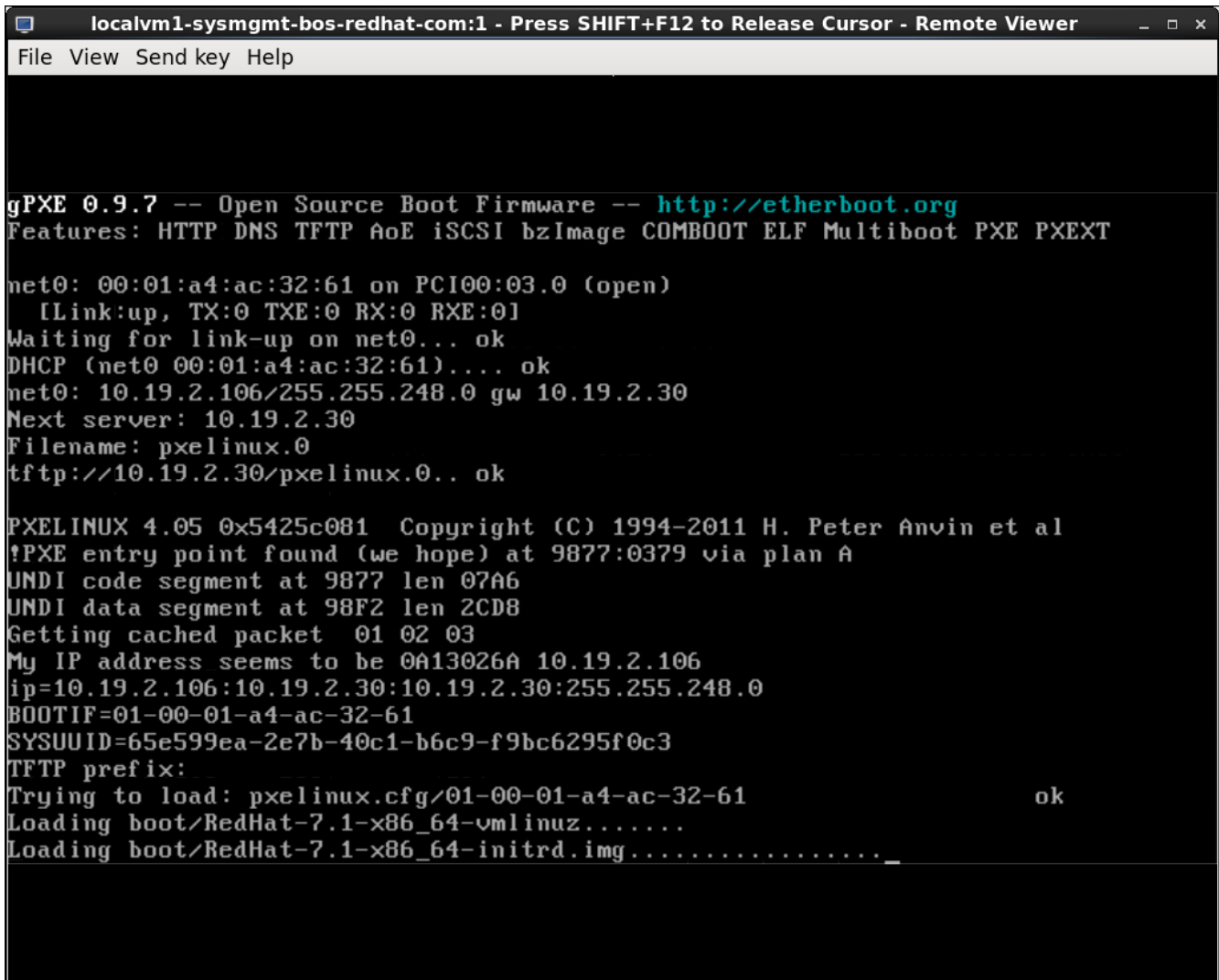


Illustration 31: Provision - New Host Information

Pxe boot



```
localvm1-sysmgmt-bos-redhat-com:1 - Press SHIFT+F12 to Release Cursor - Remote Viewer
File View Send key Help

gPXE 0.9.7 -- Open Source Boot Firmware -- http://etherboot.org
Features: HTTP DNS TFTP AoE iSCSI bzImage COMBOOT ELF Multiboot PXE PXEXT

net0: 00:01:a4:ac:32:61 on PCI00:03.0 (open)
  [Link:up, TX:0 TXE:0 RX:0 RXE:0]
Waiting for link-up on net0... ok
DHCP (net0 00:01:a4:ac:32:61)... ok
net0: 10.19.2.106/255.255.248.0 gw 10.19.2.30
Next server: 10.19.2.30
Filename: pxelinux.0
tftp://10.19.2.30/pxelinux.0.. ok

PXELINUX 4.05 0x5425c081 Copyright (C) 1994-2011 H. Peter Anvin et al
!PXE entry point found (we hope) at 9877:0379 via plan A
UNDI code segment at 9877 len 07A6
UNDI data segment at 98F2 len 2CD8
Getting cached packet 01 02 03
My IP address seems to be 0A13026A 10.19.2.106
ip=10.19.2.106:10.19.2.30:10.19.2.30:255.255.248.0
BOOTIF=01-00-01-a4-ac-32-61
SYSUUID=65e599ea-2e7b-40c1-b6c9-f9bc6295f0c3
TFTP prefix:
Trying to load: pxelinux.cfg/01-00-01-a4-ac-32-61 ok
Loading boot/RedHat-7.1-x86_64-vmlinuz.....
Loading boot/RedHat-7.1-x86_64-initrd.img....._
```

Illustration 32: Provisioning- PXE Boot

6.2 Test For Capsule Failure While Provisioning

6.2.1 Failure Of Capsule Server With Content

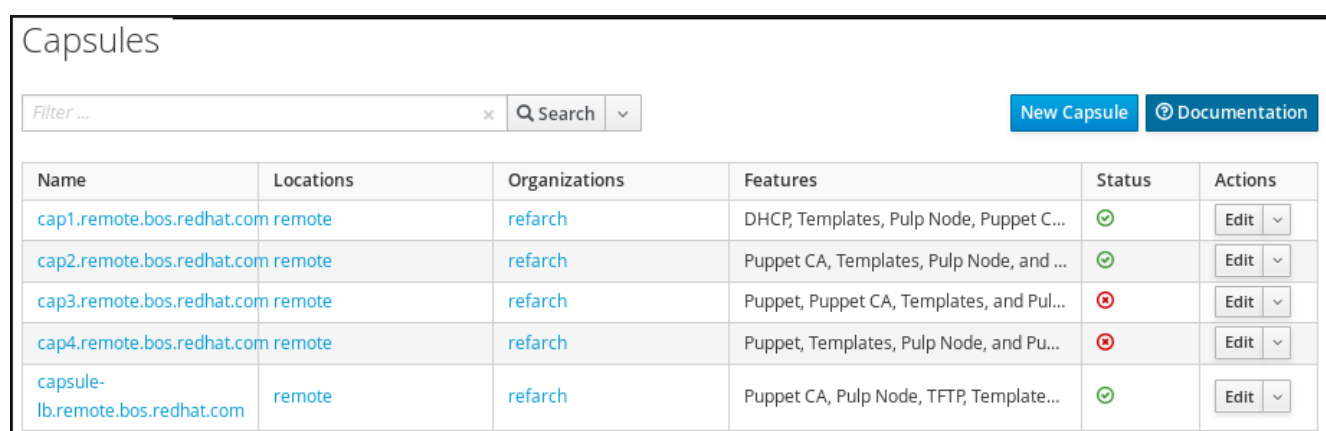
This test involves rebuilding the “*vm-test-lb*” and inducing a Capsule server failure during provisioning.

Stop services on “*cap3*” and “*cap4*”

```
[root@cap3 ~]# katello-service stop
```

```
[root@cap4 ~]# katello-service stop
```

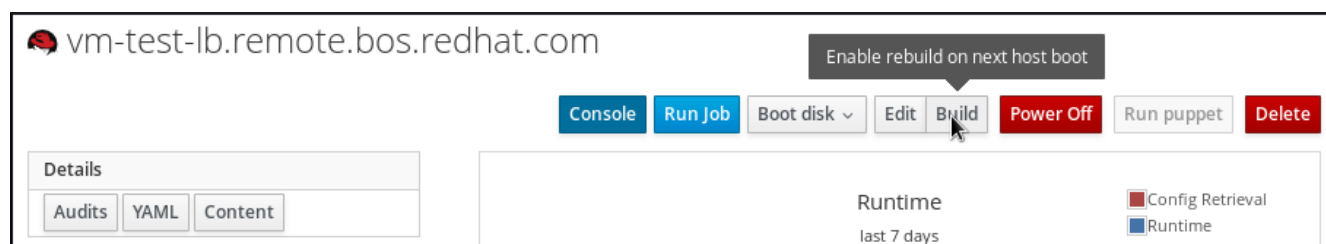
The provisioning is initiated with only *cap2* active as displayed in the illustration



Name	Locations	Organizations	Features	Status	Actions
cap1.remote.bos.redhat.com	remote	refarch	DHCP, Templates, Pulp Node, Puppet C...	✔	Edit
cap2.remote.bos.redhat.com	remote	refarch	Puppet CA, Templates, Pulp Node, and ...	✔	Edit
cap3.remote.bos.redhat.com	remote	refarch	Puppet, Puppet CA, Templates, and Pul...	✘	Edit
cap4.remote.bos.redhat.com	remote	refarch	Puppet, Templates, Pulp Node, and Pu...	✘	Edit
capsule-lb.remote.bos.redhat.com	remote	refarch	Puppet CA, Pulp Node, TFTP, Template...	✔	Edit

Illustration 33: Only Capsule "cap2" Active

Rebuild the VM



vm-test-lb.remote.bos.redhat.com

Enable rebuild on next host boot

Console Run Job Boot disk Edit **Build** Power Off Run puppet Delete

Details

Audits YAML Content

Runtime last 7 days

Config Retrieval Runtime

Illustration 34: Rebuild vm-test-lb



Upon rebuild, provisioning continues with “**cap2**” active

```
File View Send key Help
Checking storage configuration...
=====
Installation
1) [x] Language settings          2) [x] Timezone settings
   (English (United States))      (UTC timezone)
3) [x] Installation source        4) [x] Software selection
   (http://capsule-lb.remote.bos.r  (Custom software selected)
   edhat.com/pulp/repos/refarch/Li
   brary/RHEL7_2-CV/content/dist/r
   hel/server/7/7.2/x86_64/kicksta
   rt)
5) [x] Installation Destination  8) [ ] User creation
   (Automatic partitioning selecte  (No user will be created)
   d)
7) [x] Network configuration
   (Wired (eth0) connected)
=====
Progress
Setting up the installation environment
.
Creating disklabel on /dev/uda
.
Creating xfs on /dev/uda1
.
Creating lvm on /dev/uda2
.
Creating swap on /dev/mapper/rhel_vm--test--lb-swap
.
Creating xfs on /dev/mapper/rhel_vm--test--lb-root
.
Starting package installation process
Preparing transaction from installation source
Installing libgcc (1/328)
Installing redhat-release-server (2/328)
Installing setup (3/328)
Installing filesystem (4/328)
Installing tzdata (5/328)
Installing basesystem (6/328)
Installing emacs-filesystem (7/328)
Installing kbd-misc (8/328)
Installing ncurses-base (9/328)
Installing nss-softokn-freebl (10/328)
Installing glibc-common (11/328)
[anaconda] 1:main* 2:shell 3:log 4:storage-log 5:program-log  Switch tab: alt+tab  Help: F1
```

Illustration 35: Provisioning With Only Cap2 Active

Start katello-services on “**cap4**” and shutdown on “**cap2**”

```
[root@cap4 ~]# katello-service start
```

```
[root@cap2 ~]# katello-service stop
```

Notice the timestamp when the package download stops on “**cap2**” by verifying the “**capsule_access.log**”.

```
[root@cap2 httpd]# tail -5 /var/log/httpd/capsule_access.log
10.19.3.81 - - [23/Jun/2016:14:40:33 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/pkgconfig-0.27.1-4.el7.x86_64.rpm HTTP/1.1" 200 54972 "-"
"urlgrabber/3.10 yum/3.4.3"
10.19.3.81 - - [23/Jun/2016:14:40:34 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/shared-mime-info-1.1-9.el7.x86_64.rpm HTTP/1.1" 200 379496 "-"
"urlgrabber/3.10 yum/3.4.3"
10.19.3.81 - - [23/Jun/2016:14:40:35 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/libpgp-error-1.12-3.el7.x86_64.rpm HTTP/1.1" 200 89376 "-"
"urlgrabber/3.10 yum/3.4.3"
10.19.3.81 - - [23/Jun/2016:14:40:38 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/libattr-2.4.46-12.el7.x86_64.rpm HTTP/1.1" 200 18592 "-"
"urlgrabber/3.10 yum/3.4.3"
```

```
10.19.3.81 - - [23/Jun/2016:14:40:39 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/libacl-2.2.51-12.el7.x86_64.rpm HTTP/1.1" 200 27532 "-"
"urlgrabber/3.10 yum/3.4.3"
```

The Capsule list displays the status of the capsules:

Name	Locations	Organizations	Features	Status	Actions
cap1.remote.bos.redhat.com	remote	refarch	DHCP, Templates, Pulp Node, Puppet C...	⊕	Edit
cap2.remote.bos.redhat.com	remote	refarch	Puppet CA, Templates, Pulp Node, and ...	⊖	Edit
cap3.remote.bos.redhat.com	remote	refarch	Puppet, Puppet CA, Templates, and Pul...	⊖	Edit
cap4.remote.bos.redhat.com	remote	refarch	Puppet, Templates, Pulp Node, and Pu...	⊕	Edit
capsule-lb.remote.bos.redhat.com	remote	refarch	Puppet CA, Pulp Node, TFTP, Template...	⊕	Edit

Illustration 36: Only Capsule "cap4" Active

Notice provisioning continues:

```
File View Send key Help
Installing dbus-python (124/328)
Installing gzip (125/328)
Installing python-iniparse (126/328)
Installing python-dateutil (127/328)
Installing cracklib (128/328)
Installing python-decorator (129/328)
Installing m2crypto (130/328)
Installing newt-python (131/328)
Installing pygobject3-base (132/328)
Installing cracklib-dicts (133/328)
Installing pam (134/328)
Installing libpwquality (135/328)
Installing systemd-libs (136/328)
Installing libgudev1 (137/328)
Installing procps-ng (138/328)
Installing libmount (139/328)
Installing gettext (140/328)
Installing python-ethtool (141/328)
Installing python-lxml (142/328)
Installing redhat-logos (143/328)
Installing plymouth-core-libs (144/328)
Installing grubby (145/328)
Installing avahi-autoipd (146/328)
Installing libutempter (147/328)
Installing python-pyudev (148/328)
Installing pylibzma (149/328)
Installing python-configobj (150/328)
Installing pyOpenSSL (151/328)
Installing rhnlib (152/328)
Installing pygobject2 (153/328)
Installing python-gudev (154/328)
Installing yum-metadata-parser (155/328)
Installing libselinux-python (156/328)
Installing python-slip (157/328)
Installing python-slip-dbus (158/328)
Installing puxattr (159/328)
Installing libxml2-python (160/328)
Installing python-dmidecode (161/328)
Installing python-perf (162/328)
Installing xdg-utils (163/328)
Installing nss-sysinit (164/328)
Installing nss (165/328)
Installing NetworkManager-libnm (166/328)
Installing nss-tools (167/328)
Installing logrotate (168/328)
Installing alsa-lib (169/328)
[anaconda1 i:main* 2:shell 3:log 4:storage-log 5:program-log Switch tab: Alt+Tab | Help: F1
```

Illustration 37: Provisioning Continues On Cap4



Notice the timestamp when the package download resumes on “**cap4**” by verifying the “*capsule_access.log*”.

```
[root@cap4 ~]# cat /var/log/httpd/capsule_access.log
...

10.19.3.81 - - [23/Jun/2016:14:29:21 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/iwl6000g2b-firmware-17.168.5.2-43.el7.noarch.rpm HTTP/1.1" 200
314664 "-" "urlgrabber/3.10 yum/3.4.3"
10.19.3.81 - - [23/Jun/2016:14:29:22 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/iwl2030-firmware-18.168.6.1-43.el7.noarch.rpm HTTP/1.1" 200
248544 "-" "urlgrabber/3.10 yum/3.4.3"
10.19.3.81 - - [23/Jun/2016:14:29:22 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/iwl1000-firmware-39.31.5.1-43.el7.noarch.rpm HTTP/1.1" 200 215464
 "-" "urlgrabber/3.10 yum/3.4.3"
10.19.3.81 - - [23/Jun/2016:14:29:23 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/iwl100-firmware-39.31.5.1-43.el7.noarch.rpm HTTP/1.1" 200 150576
 "-" "urlgrabber/3.10 yum/3.4.3"

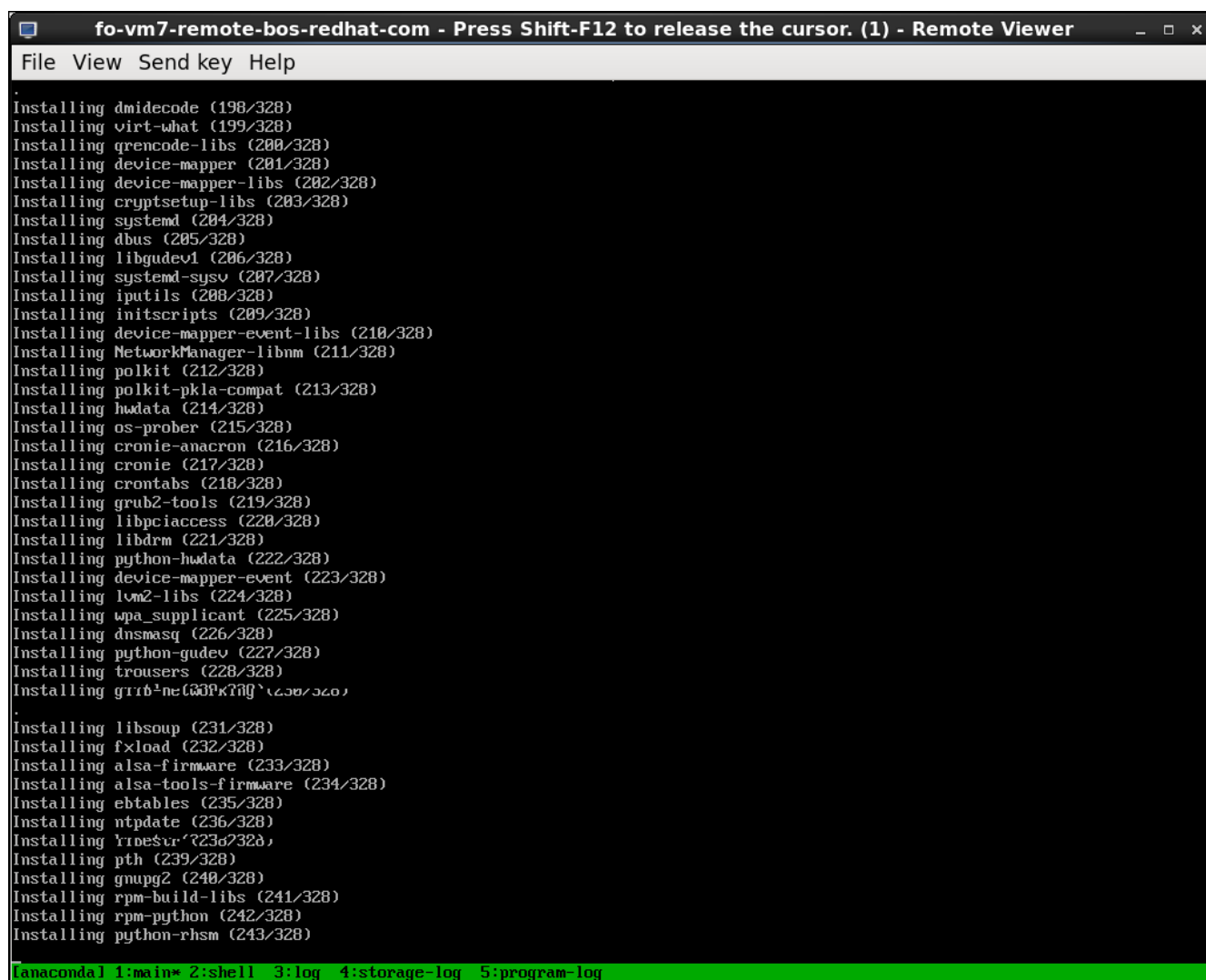
10.19.3.81 - - [23/Jun/2016:14:40:35 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/libgcrypt-1.5.3-12.el7_1.1.x86_64.rpm HTTP/1.1" 200 269032 "-"
"urlgrabber/3.10 yum/3.4.3"
10.19.3.81 - - [23/Jun/2016:14:40:38 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/libcap-2.22-8.el7.x86_64.rpm HTTP/1.1" 200 48224 "-"
"urlgrabber/3.10 yum/3.4.3"
10.19.3.81 - - [23/Jun/2016:14:40:39 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/cpio-2.11-24.el7.x86_64.rpm HTTP/1.1" 200 215084 "-"
"urlgrabber/3.10 yum/3.4.3"
10.19.3.81 - - [23/Jun/2016:14:40:41 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/libnl3-3.2.21-10.el7.x86_64.rpm HTTP/1.1" 200 201924 "-"
"urlgrabber/3.10 yum/3.4.3"
10.19.3.81 - - [23/Jun/2016:14:40:41 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/expat-2.1.0-8.el7.x86_64.rpm HTTP/1.1" 200 82372 "-"
"urlgrabber/3.10 yum/3.4.3"
10.19.3.81 - - [23/Jun/2016:14:40:42 -0400] "GET
/pulp/repos/refarch/Library/RHEL7_2-CV/content/dist/rhel/server/7/7.2/x86_64
/kickstart/p11-kit-0.20.7-3.el7.x86_64.rpm HTTP/1.1" 200 109564 "-"
"urlgrabber/3.10 yum/3.4.3"
```

6.3 Satellite Fail Over During Provisioning

Initiate a new host provisioning described in Error: Reference source not found

While the provisioning in progress simulate a node failure to trigger a cluster node failover

The Illustration 38: Host provisioning in Progress before Satellite failover describes a host being provisioned before Satellite node failover. The cluster is currently active on **satnode1**.



```
fo-vm7-remote-bos-redhat-com - Press Shift-F12 to release the cursor. (1) - Remote Viewer
File View Send key Help
Installing dmidecode (198/328)
Installing virt-what (199/328)
Installing qrencode-libs (200/328)
Installing device-mapper (201/328)
Installing device-mapper-libs (202/328)
Installing cryptsetup-libs (203/328)
Installing systemd (204/328)
Installing dbus (205/328)
Installing libgudev1 (206/328)
Installing systemd-sysv (207/328)
Installing iputils (208/328)
Installing initscripts (209/328)
Installing device-mapper-event-libs (210/328)
Installing NetworkManager-libnm (211/328)
Installing polkit (212/328)
Installing polkit-pkla-compat (213/328)
Installing hwdata (214/328)
Installing os-prober (215/328)
Installing cronie-anacron (216/328)
Installing cronie (217/328)
Installing crontabs (218/328)
Installing grub2-tools (219/328)
Installing libpciaccess (220/328)
Installing libdrm (221/328)
Installing python-hwdata (222/328)
Installing device-mapper-event (223/328)
Installing lvm2-libs (224/328)
Installing wpa_supplicant (225/328)
Installing dnsmasq (226/328)
Installing python-gudev (227/328)
Installing trousers (228/328)
Installing grub2-neo (229/328)
Installing libsoup (231/328)
Installing fxload (232/328)
Installing alsa-firmware (233/328)
Installing alsa-tools-firmware (234/328)
Installing ebttables (235/328)
Installing ntpdate (236/328)
Installing trousers (238/328)
Installing pth (239/328)
Installing gnupg2 (240/328)
Installing rpm-build-libs (241/328)
Installing rpm-python (242/328)
Installing python-rhsm (243/328)
anaconda1 1:main* 2:shell 3:log 4:storage-log 5:program-log
```

Illustration 38: Host provisioning in Progress before Satellite failover



```
[root@satnode1 /]# pcs status
```

```
Cluster name: sat62-ha
```

```
...
```

```
(Output truncated)
```

```
fs_mongodb (ocf::heartbeat:Filesystem): Started satnode1
fs_pgsql (ocf::heartbeat:Filesystem): Started satnode1
fs_puppetenv (ocf::heartbeat:Filesystem): Started satnode1
fs_tftpbboot (ocf::heartbeat:Filesystem): Started satnode1
fs_dhcp (ocf::heartbeat:Filesystem): Started satnode1
fs_named (ocf::heartbeat:Filesystem): Started satnode1
rs_dhcp (systemd:dhcpd): Started satnode1
rs_named (systemd:named): Started satnode1
rs_puppet (systemd:puppet): Started satnode1
rs_pgsql (systemd:postgresql): Started satnode1
rs_mongodb (systemd:mongod): Started satnode1
rs_qpidd (systemd:qpidd): Started satnode1
rs_qdrouterd (systemd:qdrouterd): Started satnode1
```

```
...
```

```
(Output truncated)
```

Set satnode1 to standby mode to trigger a cluster failover

```
[root@satnode1 /]# pcs cluster standby satnode1
```

```
[root@satnode1 scripts]# pcs status
```

```
Cluster name: sat62-ha
```

```
...
```

```
(Output truncated)
```

```
Node satnode1: standby
```

```
Online: [ satnode2 satnode3 ]
```

```
Full list of resources:
```

```
fs_mongodb (ocf::heartbeat:Filesystem): Started satnode3
fs_pgsql (ocf::heartbeat:Filesystem): Started satnode3
fs_puppetenv (ocf::heartbeat:Filesystem): Started satnode3
fs_tftpbboot (ocf::heartbeat:Filesystem): Started satnode3
fs_dhcp (ocf::heartbeat:Filesystem): Started satnode3
fs_named (ocf::heartbeat:Filesystem): Started satnode3
rs_dhcp (systemd:dhcpd): Started satnode3
rs_named (systemd:named): Started satnode3
rs_puppet (systemd:puppet): Started satnode3
rs_pgsql (systemd:postgresql): Started satnode3
rs_mongodb (systemd:mongod): Started satnode3
rs_qpidd (systemd:qpidd): Started satnode3
rs_qdrouterd (systemd:qdrouterd): Started satnode3
rs_tomcat (systemd:tomcat): Started satnode3
rs_pulp_workers (systemd:pulp_workers): Started satnode3
rs_foreman-proxy (systemd:foreman-proxy): Started satnode3
```

```
...
```

```
(Output truncated)
```

The provisioning continues uninterrupted and completes as described in Illustration 39: Provisioning Completed After Satellite Node Failover.

```
(Press Ctrl+Alt to release pointer) fo-vm7-remote-bos-redhat-com - Press Shift-F12 to release the cu _ □ ×
File View Send key Help
Red Hat Enterprise Linux Server 7.1 (Maipo)
Kernel 3.10.0-229.el7.x86_64 on an x86_64
fo-vm7 login: _
```

Illustration 39: Provisioning Completed After Satellite Node Failover

7 Conclusion

Satellite plays a pivotal role in provisioning, upgrading and maintaining the IT Infrastructure.

This reference architecture describes Red Hat's approach to high availability of Red Hat Satellite 6 server. The steps described in this reference architecture were performed on realistic hardware in the Systems Engineering lab using only released code.

Although Red Hat Satellite Server supports many provisioning and management features, this document focuses on a solution comprised entirely from shipping Red Hat software components. Red Hat tests and supports all of these components. These components are bundled into a single solution to ensure that Red Hat supports and stands behind the end-to-end stack.

This document covers each of these topics in sufficient detail to allow Red Hat customers to reproduce them in their own environments. The environment can be customized and expanded to meet specific.

Appendix A: Troubleshooting

A.1 HAProxy – SELinux Issue

HAProxy fails to function in Selinux enforcing mode

```
[root@hap1-sat6 etc]# audit2why -a
type=AVC msg=audit(1466698201.905:1493): avc: denied { name_connect } for
pid=80434 comm="haproxy" dest=5671 scontext=system_u:system_r:haproxy_t:s0
tcontext=system_u:object_r:amqp_port_t:s0 tclass=tcp_socket
Was caused by:
The boolean haproxy_connect_any was set incorrectly.
Description:
Allow haproxy to connect any
Allow access by executing:
# setsebool -P haproxy_connect_any 1
```

```
[root@hap1-sat6 etc]# setsebool -P haproxy_connect_any 1
```

Verify connectivity via HAProxy

```
[root@hap1-sat6 etc]# telnet capsule-lb 9090
Trying 10.19.3.50...
Connected to capsule-lb.
Escape character is '^]'.
```

A.2 Registration Issues

A.2.1 Capsule Installation Failure

```
Proxy cap1.remote.bos.redhat.com cannot be registered (422 Unprocessable
Entity): Unable to communicate with the Capsule: ERF12-2530
[ProxyAPI::ProxyException]: Unable to detect features
([OpenSSL::SSL::SSLError]: SSL_connect returned=1 errno=0 state=SSLv3 read
server session ticket A: sslv3 alert ba...) for Capsule
https://cap1.remote.bos.redhat.com:9090/features Please check the Capsule is
configured and running on the host.
```

```
/Stage[main]/Foreman_proxy::Register/Foreman_smartproxy[cap1.remote.bos.redh
at.com]/ensure: change from absent to present failed: Proxy
cap1.remote.bos.redhat.com cannot be registered (422 Unprocessable Entity):
Unable to communicate with the Capsule: ERF12-2530
[ProxyAPI::ProxyException]: Unable to detect features
([OpenSSL::SSL::SSLError]: SSL_connect returned=1 errno=0 state=SSLv3 read
server session ticket A: sslv3 alert ba...) for Capsule
https://cap1.remote.bos.redhat.com:9090/features Please check the Capsule is
configured and running on the host.
```

```
(ApipieBindings::MissingArgumentsError: id): N/A
Installing Done
```




```
[100%] [.....]  
Something went wrong! Check the log for ERROR-level output  
The full log is at /var/log/foreman-installer/capsule.log
```

Check if the capsule is registered to the satellite. If not register it.

```
#rpm -Uvh  
http://sat62-ha.sysmgmt.bos.redhat.com/pub/katello-ca-consumer-latest.noarch  
.rpm subscription-manager register --org=refarch --activationkey=Bos-AK
```

A.2.2 Time Synchronization Issues

Capsule installation registration failed:

```
[root@cap1 scripts]# subscription-manager register --org=refarch  
--activationkey=Bos-AK --force  
Unable to verify server's identity: certificate verify failed
```

Check time settings and configure ntp or chrony to ensure the capsule and satellite times are in sync.

A.3 Certificate update failure:

```
Proxy cap3.remote.bos.redhat.com cannot be registered (422 Unprocessable  
Entity): URL Only one declaration of a proxy is allowed  
  
/Stage[main]/Foreman_proxy::Register/Foreman_smartproxy[cap3.remote.bos.redh  
at.com]/url: change from https://cap3.remote.bos.redhat.com:9090 to  
https://capsule-lb.remote.bos.redhat.com:9090 failed: Proxy  
cap3.remote.bos.redhat.com cannot be registered (422 Unprocessable Entity):  
URL Only one declaration of a proxy is allowed  
Installing          Debug: Executing '/usr/share/katello-installer-bas  
[99%]  
[..... ]
```

While performing certificate update, the capsule URL is set to URL <https://capsule-lb.remote.bos.redhat.com:9090>.

For Ex: When certificate update as in step Certificate Update the URL for “**cap1a**” gets updated to <https://capsule-lb.remote.bos.redhat.com:9090>. After the update has been completed, manually restore the original URL as <https://cap1a.remote.bos.redhat.com:9090>. This would allow to continue to update “**cap1b**” since there is no other declaration of the same URL with “**capsule-lb**” virtual capsule name.

Appendix B: Configuration

B.1 Configuration Files

```
[root@satnode3 ~]# cat /etc/dhcp/dhcpd.conf
# dhcpd.conf
omapi-port 7911;

default-lease-time 43200;
max-lease-time 86400;

ddns-update-style none;

option domain-name "sysmgmt.bos.redhat.com";
option domain-name-servers 10.19.2.10;

allow booting;
allow bootp;

option fqdn.no-client-update    on; # set the "O" and "S" flag bits
option fqdn.rcode2              255;
option pxegrub code 150 = text ;

# PXE Handoff.
next-server 10.19.2.10;
filename "pxelinux.0";

log-facility local7;

include "/etc/dhcp/dhcpd.hosts";
#####
# sysmgmt.bos.redhat.com
#####
subnet 10.19.2.0 netmask 255.255.255.0 {
    pool
    {
        range 10.19.2.101 10.19.2.150;
    }

    option subnet-mask 255.255.248.0;
    option routers 10.19.2.10;
}
```



B.2 Provisioning Output Yaml File

vm-test-lb Yaml file output:

```
---
classes: {}
parameters:
  puppetmaster: capsule-lb.remote.bos.redhat.com
  domainname: ''
  hostgroup: HG-Remote-LB
  location: remote
  organization: refarch
  root_pw: "$5$Lrdv1XI$g7vjQC09gmEMidpteFZsUC3SqhEqc0XAd8M8E.UE0C7"
  puppet_ca: capsule-lb.remote.bos.redhat.com
  foreman_env: production
  owner_name: Admin User
  owner_email: root@sysmgmt.bos.redhat.com
  foreman_subnets:
  - network: 10.19.3.0
    mask: 255.255.255.0
    name: remote-net
    vlanid: ''
    gateway: 10.19.3.254
    dns_primary: 10.19.3.51
    dns_secondary: 10.19.143.247
    from: 10.19.3.101
    to: 10.19.3.150
    boot_mode: DHCP
    ipam: DHCP
  foreman_interfaces:
  - mac: 00:01:a4:ac:32:3c
    ip: 10.19.3.103
    type: Interface
    name: vm-test-lb.remote.bos.redhat.com
    attrs: {}
    virtual: false
    link: true
    identifier: eth0
    managed: true
    primary: true
    provision: true
    subnet:
      network: 10.19.3.0
      mask: 255.255.255.0
      name: remote-net
      vlanid: ''
      gateway: 10.19.3.254
      dns_primary: 10.19.3.51
      dns_secondary: 10.19.143.247
      from: 10.19.3.101
      to: 10.19.3.150
      boot_mode: DHCP
      ipam: DHCP
  kt_activation_keys: Bos-AK
  remote_execution_ssh_user: root
```

```
remote_execution_effective_user_method: sudo
kt_env: Library
kt_cv: RHEL7_2-CV
lifecycle_environment: Library
content_view: RHEL7_2-CV
kickstart_repository:
Red_Hat_Enterprise_Linux_7_Server_Kickstart_x86_64_7_2
environment: production
```

B.3 Pacemaker Configuration Output

```
[root@satnode2 ~]# pcs config show
Cluster Name: sat62-ha
Corosync Nodes:
  satnode1 satnode2 satnode3
Pacemaker Nodes:
  satnode1 satnode2 satnode3

Resources:
Resource: VirtualIP (class=ocf provider=heartbeat type=IPaddr2)
  Attributes: ip=10.19.2.10 cidr_netmask=24
  Operations: start interval=0s timeout=20s (VirtualIP-start-timeout-20s)
              stop interval=0s timeout=20s (VirtualIP-stop-timeout-20s)
              monitor interval=10s timeout=20s
(VirtualIP-monitor-interval-10s)
Resource: sat_vg (class=ocf provider=heartbeat type=LVM)
  Attributes: volgrpname=sat_vg exclusive=true
  Operations: start interval=0s timeout=30 (sat_vg-start-timeout-30)
              stop interval=0s timeout=30 (sat_vg-stop-timeout-30)
              monitor interval=10 timeout=30 (sat_vg-monitor-interval-10)
Resource: fs_pulp (class=ocf provider=heartbeat type=Filesystem)
  Attributes: device=/dev/sat_vg/lv_pulp directory=/var/lib/pulp fstype= xfs
  Operations: start interval=0s timeout=60 (fs_pulp-start-timeout-60)
              stop interval=0s timeout=60 (fs_pulp-stop-timeout-60)
              monitor interval=20 timeout=40 (fs_pulp-monitor-interval-20)
Resource: fs_candlepin (class=ocf provider=heartbeat type=Filesystem)
  Attributes: device=/dev/sat_vg/lv_candlepin directory=/var/lib/candlepin
fstype= xfs
  Operations: start interval=0s timeout=60 (fs_candlepin-start-timeout-60)
              stop interval=0s timeout=60 (fs_candlepin-stop-timeout-60)
              monitor interval=20 timeout=40
(fs_candlepin-monitor-interval-20)
Resource: fs_foreman (class=ocf provider=heartbeat type=Filesystem)
  Attributes: device=/dev/sat_vg/lv_foreman directory=/var/lib/foreman
fstype=xfs
  Operations: start interval=0s timeout=60 (fs_foreman-start-timeout-60)
              stop interval=0s timeout=60 (fs_foreman-stop-timeout-60)
              monitor interval=20 timeout=40
(fs_foreman-monitor-interval-20)
Resource: fs_puppet (class=ocf provider=heartbeat type=Filesystem)
  Attributes: device=/dev/sat_vg/lv_puppet directory=/var/lib/puppet
```



```
fstype=xfst
  Operations: start interval=0s timeout=60 (fs_puppet-start-timeout-60)
              stop interval=0s timeout=60 (fs_puppet-stop-timeout-60)
              monitor interval=20 timeout=40 (fs_puppet-monitor-interval-20)
Resource: fs_wwwpulp (class=ocf provider=heartbeat type=Filesystem)
  Attributes: device=/dev/sat_vg/lv_wwwpulp directory=/var/www/pulp
fstype=xfst
  Operations: start interval=0s timeout=60 (fs_wwwpulp-start-timeout-60)
              stop interval=0s timeout=60 (fs_wwwpulp-stop-timeout-60)
              monitor interval=20 timeout=40
(fs_wwwpulp-monitor-interval-20)

Resource: fs_mongodb (class=ocf provider=heartbeat type=Filesystem)
  Attributes: device=/dev/sat_vg/lv_mongodb directory=/var/lib/mongodb
fstype=xfst
  Operations: start interval=0s timeout=60 (fs_mongodb-start-timeout-60)
              stop interval=0s timeout=60 (fs_mongodb-stop-timeout-60)
              monitor interval=20 timeout=40
(fs_mongodb-monitor-interval-20)
Resource: fs_pgsqldata (class=ocf provider=heartbeat type=Filesystem)
  Attributes: device=/dev/sat_vg/lv_pgsqldata directory=/var/lib/pgsqldata
fstype=xfst
  Operations: start interval=0s timeout=60 (fs_pgsqldata-start-timeout-60)
              stop interval=0s timeout=60 (fs_pgsqldata-stop-timeout-60)
              monitor interval=20 timeout=40 (fs_pgsqldata-monitor-interval-20)
Resource: fs_puppetenv (class=ocf provider=heartbeat type=Filesystem)
  Attributes: device=/dev/sat_vg/lv_puppetenv
              directory=/etc/puppet/environments fstype=xfst
  Operations: start interval=0s timeout=60 (fs_puppetenv-start-timeout-60)
              stop interval=0s timeout=60 (fs_puppetenv-stop-timeout-60)
              monitor interval=20 timeout=40
(fs_puppetenv-monitor-interval-20)
Resource: fs_tftpbboot (class=ocf provider=heartbeat type=Filesystem)
  Attributes: device=/dev/sat_vg/lv_tftpbboot directory=/var/lib/tftpbboot
fstype=xfst
  Operations: start interval=0s timeout=60 (fs_tftpbboot-start-timeout-60)
              stop interval=0s timeout=60 (fs_tftpbboot-stop-timeout-60)
              monitor interval=20 timeout=40
(fs_tftpbboot-monitor-interval-20)
Resource: fs_dhcp (class=ocf provider=heartbeat type=Filesystem)
  Attributes: device=/dev/sat_vg/lv_dhcp directory=/var/lib/dhcpd fstype=xfst
  Operations: start interval=0s timeout=60 (fs_dhcp-start-timeout-60)
              stop interval=0s timeout=60 (fs_dhcp-stop-timeout-60)
              monitor interval=20 timeout=40 (fs_dhcp-monitor-interval-20)
Resource: fs_named (class=ocf provider=heartbeat type=Filesystem)
  Attributes: device=/dev/sat_vg/lv_named directory=/var/named fstype=xfst
  Operations: start interval=0s timeout=60 (fs_named-start-timeout-60)
              stop interval=0s timeout=60 (fs_named-stop-timeout-60)
              monitor interval=20 timeout=40 (fs_named-monitor-interval-20)
Resource: rs_dhcp (class=systemd type=dhcpd)
  Operations: monitor interval=10s (rs_dhcp-monitor-interval-10s)
              start interval=0s timeout=100s (rs_dhcp-start-interval-0s)
              stop interval=0s timeout=100s (rs_dhcp-stop-interval-0s)
Resource: rs_named (class=systemd type=named)
  Operations: monitor interval=10s (rs_named-monitor-interval-10s)
```

```
        start interval=0s timeout=100s (rs_named-start-interval-0s)
        stop interval=0s timeout=100s (rs_named-stop-interval-0s)
Resource: rs_puppet (class=systemd type=puppet)
  Operations: monitor interval=10s (rs_puppet-monitor-interval-10s)
             start interval=0s timeout=100s (rs_puppet-start-interval-0s)
             stop interval=0s timeout=100s (rs_puppet-stop-interval-0s)
Resource: rs_pgsql (class=systemd type=postgresql)
  Operations: monitor interval=10s (rs_pgsql-monitor-interval-10s)
             start interval=0s timeout=100s (rs_pgsql-start-interval-0s)
             stop interval=0s timeout=100s (rs_pgsql-stop-interval-0s)
Resource: rs_mongodb (class=systemd type=mongod)
  Operations: monitor interval=10s (rs_mongodb-monitor-interval-10s)
             start interval=0s timeout=100s (rs_mongodb-start-interval-0s)
             stop interval=0s timeout=100s (rs_mongodb-stop-interval-0s)
Resource: rs_qpidd (class=systemd type=qpidd)
  Operations: monitor interval=10s (rs_qpidd-monitor-interval-10s)
             start interval=0s timeout=100s (rs_qpidd-start-interval-0s)
             stop interval=0s timeout=100s (rs_qpidd-stop-interval-0s)
Resource: rs_qdrouterd (class=systemd type=qdrouterd)
  Operations: monitor interval=10s (rs_qdrouterd-monitor-interval-10s)
             start interval=0s timeout=100s
(rs_qdrouterd-start-interval-0s)
             stop interval=0s timeout=100s (rs_qdrouterd-stop-interval-0s)
Resource: rs_tomcat (class=systemd type=tomcat)
  Operations: monitor interval=10s (rs_tomcat-monitor-interval-10s)
             start interval=0s timeout=100s (rs_tomcat-start-interval-0s)
             stop interval=0s timeout=100s (rs_tomcat-stop-interval-0s)
Resource: rs_pulp_workers (class=systemd type=pulp_workers)
  Operations: monitor interval=10s (rs_pulp_workers-monitor-interval-10s)
             start interval=0s timeout=100s
(rs_pulp_workers-start-interval-0s)
             stop interval=0s timeout=100s
(rs_pulp_workers-stop-interval-0s)
Resource: rs_foreman-proxy (class=systemd type=foreman-proxy)
  Operations: monitor interval=10s (rs_foreman-proxy-monitor-interval-10s)
             start interval=0s timeout=100s
(rs_foreman-proxy-start-interval-0s)
             stop interval=0s timeout=100s
(rs_foreman-proxy-stop-interval-0s)
Resource: rs_pulp_resource_manager (class=systemd
type=pulp_resource_manager)
  Operations: monitor interval=10s
(rs_pulp_resource_manager-monitor-interval-10s)
             start interval=0s timeout=100s
(rs_pulp_resource_manager-start-interval-0s)
             stop interval=0s timeout=100s
(rs_pulp_resource_manager-stop-interval-0s)
Resource: rs_pulp_celerybeat (class=systemd type=pulp_celerybeat)
  Operations: monitor interval=10s (rs_pulp_celerybeat-monitor-interval-10s)
             start interval=0s timeout=100s
(rs_pulp_celerybeat-start-interval-0s)
             stop interval=0s timeout=100s
(rs_pulp_celerybeat-stop-interval-0s)
```



```
Resource: rs_httpd (class=systemd type=httpd)
  Operations: monitor interval=10s (rs_httpd-monitor-interval-10s)
              start interval=0s timeout=100s (rs_httpd-start-interval-0s)
              stop interval=0s timeout=100s (rs_httpd-stop-interval-0s)
Resource: rs_foreman-tasks (class=systemd type=foreman-tasks)
  Operations: monitor interval=10s timeout=30s
(rs_foreman-tasks-monitor-interval-10s)
              start interval=0s timeout=100s
(rs_foreman-tasks-start-interval-0s)
              stop interval=0s timeout=100s
(rs_foreman-tasks-stop-interval-0s)
```

Stonith Devices:

```
Resource: satnode1-fence (class=stonith type=fence_ipmilan)
  Attributes: login=root passwd=100Mgmt- action=reboot ipaddr=10.19.143.24
lanplus=1 verbose= pcmk_host_list=satnode1
  Operations: monitor interval=60s (satnode1-fence-monitor-interval-60s)
Resource: satnode2-fence (class=stonith type=fence_ipmilan)
  Attributes: login=root passwd=100Mgmt- action=reboot ipaddr=10.19.143.23
lanplus=1 verbose= pcmk_host_list=satnode2
  Operations: monitor interval=60s (satnode2-fence-monitor-interval-60s)
Resource: satnode3-fence (class=stonith type=fence_ipmilan)
  Attributes: login=root passwd=100Mgmt- action=reboot ipaddr=10.19.143.22
lanplus=1 verbose= pcmk_host_list=satnode3
  Operations: monitor interval=60s (satnode3-fence-monitor-interval-60s)
```

Fencing Levels:

Location Constraints:

Ordering Constraints:

```
start VirtualIP then start sat_vg (kind:Mandatory)
(id:order-VirtualIP-sat_vg-mandatory)
start fs_wwwpulp then start rs_httpd (kind:Mandatory)
(id:order-fs_wwwpulp-rs_httpd-mandatory)
start VirtualIP then start rs_dhcp (kind:Mandatory)
(id:order-VirtualIP-rs_dhcp-mandatory)
start VirtualIP then start rs_named (kind:Mandatory)
(id:order-VirtualIP-rs_named-mandatory)
```

Resource Sets:

```
set sat_vg (id:pcs_rsc_set_sat_vg-1) set fs_dhcp fs_named fs_puppet
fs_puppetenv fs_pgsq1 fs_mongodb fs_pulp fs_wwwpulp fs_foreman fs_candlepin
fs_tftpboot sequential=false
(id:pcs_rsc_set_fs_dhcp_fs_named_fs_puppet_fs_puppetenv_fs_pgsq1_fs_mongodb_
fs_pulp_fs_wwwpulp_fs_foreman_fs_candlepin_fs_tftpboot-1)
(id:pcs_rsc_order_sat_vg_set_fs_dhcp_fs_named_fs_puppet_fs_puppetenv_fs_pgsq
l_fs_mongodb_fs_pulp_fs_wwwpulp_fs_foreman_fs_candlepin_fs_tftpboot)
set sat_vg fs_dhcp rs_dhcp (id:pcs_rsc_set_sat_vg_fs_dhcp_rs_dhcp)
(id:pcs_rsc_order_sat_vg_fs_dhcp_rs_dhcp)
set sat_vg fs_named rs_named (id:pcs_rsc_set_sat_vg_fs_named_rs_named)
(id:pcs_rsc_order_sat_vg_fs_named_rs_named)
set sat_vg fs_puppet fs_puppetenv rs_puppet
(id:pcs_rsc_set_sat_vg_fs_puppet_fs_puppetenv_rs_puppet)
(id:pcs_rsc_order_sat_vg_fs_puppet_fs_puppetenv_rs_puppet)
set sat_vg fs_pgsq1 rs_pgsq1 (id:pcs_rsc_set_sat_vg_fs_pgsq1_rs_pgsq1)
(id:pcs_rsc_order_sat_vg_fs_pgsq1_rs_pgsq1)
set sat_vg fs_mongodb rs_mongodb
```

```
(id:pcs_rsc_set_sat_vg_fs_mongodb_rs_mongodb)
(id:pcs_rsc_order_sat_vg_fs_mongodb_rs_mongodb)
    set sat_vg fs_pulp fs_wwwpulp rs_pulp_workers rs_pulp_celerybeat
(id:pcs_rsc_set_sat_vg_fs_pulp_fs_wwwpulp_rs_pulp_workers_rs_pulp_celerybeat
)
(id:pcs_rsc_order_sat_vg_fs_pulp_fs_wwwpulp_rs_pulp_workers_rs_pulp_celerybe
at)
    set sat_vg fs_foreman rs_foreman-proxy
(id:pcs_rsc_set_sat_vg_fs_foreman_rs_foreman-proxy)
(id:pcs_rsc_order_sat_vg_fs_foreman_rs_foreman-proxy)

    set VirtualIP rs_qpidd rs_qdrouterd
(id:pcs_rsc_set_VirtualIP_rs_qpidd_rs_qdrouterd-1)
(id:pcs_rsc_order_VirtualIP_rs_qpidd_rs_qdrouterd)
    set rs_puppet rs_dhcp rs_named rs_pgsql sequential=false
(id:pcs_rsc_set_rs_puppet_rs_dhcp_rs_named_rs_pgsql) set rs_mongodb
(id:pcs_rsc_set_rs_mongodb) set rs_qpidd rs_qdrouterd sequential=false
(id:pcs_rsc_set_rs_qpidd_rs_qdrouterd) set rs_tomcat rs_pulp_workers
rs_pulp_celerybeat rs_pulp_resource_manager rs_foreman-proxy
sequential=false
(id:pcs_rsc_set_rs_tomcat_rs_pulp_workers_rs_pulp_celerybeat_rs_pulp_resourc
e_manager_rs_foreman-proxy) set rs_httpd rs_foreman-tasks sequential=false
(id:pcs_rsc_set_rs_httpd_rs_foreman-tasks)
(id:pcs_rsc_order_rs_puppet_rs_dhcp_rs_named_rs_pgsql_set_rs_mongodb_set_rs_
qpidd_rs_qdrouterd_set_rs_tomcat_rs_pulp_workers_rs_pulp_celerybeat_rs_pulp_
resource_manager_rs_foreman-proxy_set_rs_httpd_rs_foreman-tasks)
Colocation Constraints:
    sat_vg with VirtualIP (score:INFINITY)
(id:colocation-sat_vg-VirtualIP-INFINITY)
    rs_dhcp with fs_dhcp (score:INFINITY)
(id:colocation-rs_dhcp-fs_dhcp-INFINITY)
    rs_named with fs_named (score:INFINITY)
(id:colocation-rs_named-fs_named-INFINITY)
    rs_pgsql with fs_pgsql (score:INFINITY)
(id:colocation-rs_pgsql-fs_pgsql-INFINITY)
    rs_mongodb with fs_mongodb (score:INFINITY)
(id:colocation-rs_mongodb-fs_mongodb-INFINITY)
    rs_foreman-proxy with fs_foreman (score:INFINITY)
(id:colocation-rs_foreman-proxy-fs_foreman-INFINITY)
    rs_tomcat with VirtualIP (score:INFINITY)
(id:colocation-rs_tomcat-VirtualIP-INFINITY)
    rs_httpd with VirtualIP (score:INFINITY)
(id:colocation-rs_httpd-VirtualIP-INFINITY)
    rs_foreman-tasks with fs_foreman (score:INFINITY)
(id:colocation-rs_foreman-tasks-fs_foreman-INFINITY)
Resource Sets:
    set fs_dhcp fs_named fs_puppet fs_puppetenv fs_pgsql fs_mongodb fs_pulp
fs_wwwpulp fs_foreman fs_candlepin fs_tftboot sequential=false
(id:pcs_rsc_set_fs_dhcp_fs_named_fs_puppet_fs_puppetenv_fs_pgsql_fs_mongodb_
fs_pulp_fs_wwwpulp_fs_foreman_fs_candlepin_fs_tftboot) set sat_vg
(id:pcs_rsc_set_sat_vg) setoptions score=INFINITY
(id:pcs_rsc_colocation_fs_dhcp_fs_named_fs_puppet_fs_puppetenv_fs_pgsql_fs_m
ongodb_fs_pulp_fs_wwwpulp_fs_foreman_fs_candlepin_fs_tftboot_set_sat_vg)
```




```
    set fs_puppet fs_puppetenv rs_puppet
(id:pcs_rsc_set_fs_puppet_fs_puppetenv_rs_puppet) setoptions score=INFINITY
(id:pcs_rsc_colocation_fs_puppet_fs_puppetenv_rs_puppet)
    set fs_pulp fs_wwwpulp rs_pulp_workers
(id:pcs_rsc_set_fs_pulp_fs_wwwpulp_rs_pulp_workers) setoptions
score=INFINITY (id:pcs_rsc_colocation_fs_pulp_fs_wwwpulp_rs_pulp_workers)
    set VirtualIP rs_qpidd rs_qdrouterd
(id:pcs_rsc_set_VirtualIP_rs_qpidd_rs_qdrouterd) setoptions score=INFINITY
(id:pcs_rsc_colocation_VirtualIP_rs_qpidd_rs_qdrouterd)
    set fs_pulp fs_wwwpulp rs_pulp_resource_manager
(id:pcs_rsc_set_fs_pulp_fs_wwwpulp_rs_pulp_resource_manager) setoptions
score=INFINITY
(id:pcs_rsc_colocation_fs_pulp_fs_wwwpulp_rs_pulp_resource_manager)
    set fs_pulp fs_wwwpulp rs_pulp_celerybeat
(id:pcs_rsc_set_fs_pulp_fs_wwwpulp_rs_pulp_celerybeat) setoptions
score=INFINITY (id:pcs_rsc_colocation_fs_pulp_fs_wwwpulp_rs_pulp_celerybeat)
```

Cluster Properties:

```
cluster-infrastructure: corosync
cluster-name: SAT62-HA
dc-version: 1.1.13-a14efad
default-resource-stickiness: INFINITY
have-watchdog: false
last-lrm-refresh: 1448469373
```

Node Attributes:

```
satnode2: standby=on
satnode3: standby=on
```

Appendix C: Scripts

The following scripts and config files can be downloaded from:

<https://access.redhat.com/node/2089291/40/0>

C.1 Create Shared Filesystems Script

```
[root@satnode1 scripts]# cat /root/scripts/create-fileSYSTEMS.sh
#!/bin/bash
#Create and mount filesystems required for the satellite
#environment that would be controlled by pacemaker eventually
#TO BE RUN ON THE FIRST NODE ONLY
### Create Physical Volume ###
#
pvcreate /dev/sdc
#
### Create Volume group ###
#
vgcreate sat_vg /dev/sdc
#
### Create Logical volumes ###
lvcreate -L 500G -n lv_pulp sat_vg
lvcreate -L 10G -n lv_foreman sat_vg
lvcreate -L 10G -n lv_puppet sat_vg
lvcreate -L 10G -n lv_wwwpulp sat_vg
lvcreate -L 10G -n lv_candlepin sat_vg
lvcreate -L 100G -n lv_mongodb sat_vg
lvcreate -L 10G -n lv_psqldata sat_vg
lvcreate -L 10G -n lv_puppetenv sat_vg
lvcreate -L 10G -n lv_tftpboot sat_vg
lvcreate -L 10G -n lv_dhcp sat_vg
lvcreate -L 10G -n lv_named sat_vg
#
### Create filesystems using these volumes ###
for i in $(lvs | grep sat_vg | awk '{print $1}'); do mkfs.xfs /dev/sat_vg/
$i; done
#
### Create mount points ###
#
mkdir -p /var/lib/pulp
mkdir -p /var/lib/candlepin
mkdir -p /var/lib/foreman
mkdir -p /var/lib/puppet
mkdir -p /var/www/pulp
mkdir -p /var/lib/mongodb
mkdir -p /var/lib/pgsql
mkdir -p /etc/puppet/environments
mkdir -p /var/lib/tftpboot
mkdir -p /var/lib/dhcpd
systemctl stop named.service
```



```
mv /var/named /var/named.orig
mkdir -p /var/named
#
### Mount the filesystems ###
#
mount /dev/mapper/sat_vg-lv_pulp /var/lib/pulp
mount /dev/mapper/sat_vg-lv_candlepin /var/lib/candlepin
mount /dev/mapper/sat_vg-lv_foreman /var/lib/foreman
mount /dev/mapper/sat_vg-lv_puppet /var/lib/puppet
mount /dev/mapper/sat_vg-lv_wwwpulp /var/www/pulp
mount /dev/mapper/sat_vg-lv_mongodb /var/lib/mongodb
mount /dev/mapper/sat_vg-lv_psqldata /var/lib/pgsql
mount /dev/mapper/sat_vg-lv_puppetenv /etc/puppet/environments
mount /dev/mapper/sat_vg-lv_tftpboot /var/lib/tftpboot
mount /dev/mapper/sat_vg-lv_named /var/named
# Copy over contents of /var/named.orig to /var/named
cp -pr /var/named.orig/* /var/named
# Restore selinux booleans
restorecon -Rv /var/named
```

C.2 Cleanup Filesystem Script

```
[root@satnode1 scripts]# cat /root/scripts/cleanup.sh
#!/bin/bash
#
#This script stops Satellite services on the node, unmounts the
#filesystems and deactivates the shared volume group
#
#Stop services
katello-service stop
systemctl stop postgresql.service
systemctl stop puppet.service
systemctl stop dhcpd.service
systemctl stop named.service
#
#unmount the filesystems
#
umount /dev/mapper/sat_vg-lv_pulp /var/lib/pulp
umount /dev/mapper/sat_vg-lv_candlepin /var/lib/candlepin
umount /dev/mapper/sat_vg-lv_foreman /var/lib/foreman
umount /dev/mapper/sat_vg-lv_puppet /var/lib/puppet
umount /dev/mapper/sat_vg-lv_wwwpulp /var/www/pulp
umount /dev/mapper/sat_vg-lv_mongodb /var/lib/mongodb
umount /dev/mapper/sat_vg-lv_psqldata /var/lib/pgsql
umount /dev/mapper/sat_vg-lv_puppetenv /etc/puppet/environments
umount /dev/mapper/sat_vg-lv_tftpboot /var/lib/tftpboot
umount /dev/mapper/sat_vg-lv_dhcp /var/lib/dhcpd
umount /dev/mapper/sat_vg-lv_named /var/named
#
#Deactivate volume group
#
vgchange -a n sat_vg
```

C.3 Move Shared Filesystem Script

```
[root@satnode1 scripts]# cat /root/scripts/move-filefilesystems.sh
#!/bin/bash
# This script mounts the shared filesystems to the secondary node
#Activate the volume group
vgchange -a y sat_vg
#
#Create mount points
#
mkdir -p /var/lib/pulp
mkdir -p /var/lib/candlepin
mkdir -p /var/lib/foreman
mkdir -p /var/lib/puppet
mkdir -p /var/www/pulp
mkdir -p /var/lib/mongodb
mkdir -p /var/lib/pgsql
mkdir -p /etc/puppet/environments
mkdir -p /var/lib/tftpboot
mkdir -p /var/lib/dhcpd
systemctl stop named.service
mv /var/named /var/named.orig
mkdir -p /var/named
#
#Mount the filesystems
mount -o _netdev /dev/mapper/sat_vg-lv_pulp /var/lib/pulp
mount -o _netdev /dev/mapper/sat_vg-lv_candlepin /var/lib/candlepin
mount -o _netdev /dev/mapper/sat_vg-lv_foreman /var/lib/foreman
mount -o _netdev /dev/mapper/sat_vg-lv_puppet /var/lib/puppet
mount -o _netdev /dev/mapper/sat_vg-lv_wwwpulp /var/www/pulp
mount -o _netdev /dev/mapper/sat_vg-lv_mongodb /var/lib/mongodb
mount -o _netdev /dev/mapper/sat_vg-lv_psqldata /var/lib/pgsql
mount -o _netdev /dev/mapper/sat_vg-lv_puppetenv /etc/puppet/environments
mount -o _netdev /dev/mapper/sat_vg-lv_tftpboot /var/lib/tftpboot
mount -o _netdev /dev/mapper/sat_vg-lv_dhcp /var/lib/dhcpd
mount -o _netdev /dev/mapper/sat_vg-lv_named /var/named

###NOTE: mount option _netdev maybe needed in using ISCSI disk and adding an
#entry in /etc/fstab file for any reason.
```

C.4 Manual Satellite Start And Stop Scripts

```
[root@satnode1 scripts]# cat satellite-start.sh
systemctl start named.service
systemctl start dhcpd.service
systemctl start puppet.service
systemctl start postgresql.service
katello-service start
```

```
[root@satnode1 scripts]# cat satellite-stop.sh
katello-service stop
systemctl stop postgresql.service
systemctl stop puppet.service
```



```
systemctl stop dhcpd.service
systemctl stop named.service
```

C.5 Mount And Unmount Filesystems Scripts

```
[root@satnode1 scripts]# cat mount_filesystems.sh
#!/bin/bash
#Activate the volume group
vgchange -a y sat_vg
#
#Mount the filesystems
mount -o _netdev /dev/mapper/sat_vg-lv_named /var/named
mount -o _netdev /dev/mapper/sat_vg-lv_pulp /var/lib/pulp
mount -o _netdev /dev/mapper/sat_vg-lv_candlepin /var/lib/candlepin
mount -o _netdev /dev/mapper/sat_vg-lv_foreman /var/lib/foreman
mount -o _netdev /dev/mapper/sat_vg-lv_puppet /var/lib/puppet
mount -o _netdev /dev/mapper/sat_vg-lv_wwwpulp /var/www/pulp
mount -o _netdev /dev/mapper/sat_vg-lv_mongodb /var/lib/mongodb
mount -o _netdev /dev/mapper/sat_vg-lv_psqldata /var/lib/pgsql
mount -o _netdev /dev/mapper/sat_vg-lv_puppetenv /etc/puppet/environments
mount -o _netdev /dev/mapper/sat_vg-lv_tftpboot /var/lib/tftpboot
mount -o _netdev /dev/mapper/sat_vg-lv_dhcp /var/lib/dhcpd
mount -o _netdev /dev/mapper/sat_vg-lv_named /var/named
```

```
[root@satnode1 scripts]# cat mount_filesystems.sh
#!/bin/bash
# Unmount the shared filesystems
umount /dev/mapper/sat_vg-lv_pulp /var/lib/pulp
umount /dev/mapper/sat_vg-lv_candlepin /var/lib/candlepin
umount /dev/mapper/sat_vg-lv_foreman /var/lib/foreman
umount /dev/mapper/sat_vg-lv_puppet /var/lib/puppet
umount /dev/mapper/sat_vg-lv_wwwpulp /var/www/pulp
umount /dev/mapper/sat_vg-lv_mongodb /var/lib/mongodb
umount /dev/mapper/sat_vg-lv_psqldata /var/lib/pgsql
umount /dev/mapper/sat_vg-lv_puppetenv /etc/puppet/environments
umount /dev/mapper/sat_vg-lv_tftpboot /var/lib/tftpboot
umount /dev/mapper/sat_vg-lv_dhcp /var/lib/dhcpd

umount /dev/mapper/sat_vg-lv_named /var/named
```

C.6 Satellite Backup Script

```
[root@satnode1 scripts]# cat satellite-backup.sh
#!/bin/bash
# Satellite 6.2
# Disclaimer | This script is NOT SUPPORTED by Red Hat Global Support
Service.
# Global Variables
umask 0027
export RETENTION=15
export VER=`date +%Y-%m-%d-%H%M%S`
export BDIR=/var/tmp/rhs6backup
export LOGFILE=/var/log/satellite6-backup.log
export DIRECTORIES='/etc/sudoers /etc/sudoers.d /etc/foreman-proxy
/etc/foreman /etc/hammer /etc/tomcat /etc/httpd /etc/satellite-installer
```

```
/etc/pki /etc/puppet /srv /var/lib/candlepin /var/lib/foreman-proxy
/var/lib/foreman /var/lib/puppet /var/www/pulp /etc/default /etc/katello
/etc/candlepin /etc/pulp /etc/pki/katello /etc/pki/pulp
/etc/sysconfig/katello /root/ssl-build /var/www/html/pub
/usr/share/foreman-proxy /var/named /usr/share/katello /var/lib/tftpboot
/var/lib/dhcpd /etc/dhcp/'
# Function to launch commands
execom () {
command=$1
if [ ` /bin/echo $command | grep -P .+ | wc -l ` -ne 0 ]
then
datecom=`/bin/date +%Y%m%d%H%M%S`
/bin/echo "$datecom Executting command : '$command >> $LOGFILE 2>&1'" >>
$LOGFILE 2>&1
/bin/sh -c "$command >> $LOGFILE 2>&1"
RET=$?
/bin/echo "$datecom Return from command '$command >> $LOGFILE 2>&1' is
'$RET'" >> $LOGFILE 2>&1
if [ ! "$RET" == "0" ]
then
/bin/echo "$datecom Return from command '$command >> $LOGFILE 2>&1' '$RET' !
= '0' . Aborting script : 'exit 1'" >> $LOGFILE 2>&1
/bin/echo "$datecom Return from command '$command >> $LOGFILE 2>&1' '$RET' !
= '0' . Aborting script : 'exit 1'"
/usr/bin/katello-service start >> $LOGFILE 2>&1
exit 1
fi
fi
}
# Script begin
/bin/echo "Starting execution of $0 $1 at $VER" >> $LOGFILE 2>&1
# Generate skel dir with permissions
execom "/bin/mkdir -p ${BDIR}"
execom "/bin/chmod 755 ${BDIR}"
execom "/bin/mkdir -p ${BDIR}/${VER}"
execom "/bin/chgrp postgres ${BDIR}/${VER}"
execom "/bin/chmod 770 ${BDIR}/${VER}"
# Clean up old backups, they have been captured by NetBackup
for i in $(/usr/bin/find ${BDIR}/ -type d -mtime +${RETENTION})
do
execom "/bin/rm -rf $i"
done
# Backup the configuration and data files
execom "/bin/tar --selinux -czvf ${BDIR}/${VER}/config_files.tar.gz
$DIRECTORIES"
execom "/bin/tar --selinux -tzvf ${BDIR}/${VER}/config_files.tar.gz"
# Online Repositories Backup
/bin/echo "Trying to make online backup /var/lib/pulp and /var/www/pub via
rsync" >> $LOGFILE 2>&1
/usr/bin/rsync --partial --progress --delete -alv /var/lib/pulp ${BDIR}/$
{VER}/ >> $LOGFILE 2>&1
CHKREPO=$(/bin/find /var/lib/pulp -printf '%T@\n' | md5sum)
execom "/usr/bin/rsync --partial --progress --delete -alv /var/lib/pulp $
```



```
{BDIR}/${VER}/"
CHKREPOB=$(/bin/find /var/lib/pulp -printf '%T@\n' | md5sum)
/bin/echo "MD5 before tar: $CHKREPO after tar: $CHKREPOB" >> $LOGFILE 2>&1
if [ ! "$CHKREPO" == "$CHKREPOB" ]
then
/bin/echo "ERROR doing online repositories backup. Aborting backup process"
>> $LOGFILE 2>&1 exit 1
fi
# Optional: Online Database Backup
# rhs6 must be stopped
execom "/usr/bin/katello-service stop"
execom "/bin/tar --selinux -czvf ${BDIR}/${VER}/mongo_data.tar.gz
/var/lib/mongodb"
execom "/bin/tar --selinux -tzvf ${BDIR}/${VER}/mongo_data.tar.gz"
execom "/bin/tar --selinux -czvf ${BDIR}/${VER}/pgsql_data.tar.gz
/var/lib/pgsql/data"
execom "/bin/tar --selinux -tzvf ${BDIR}/${VER}/pgsql_data.tar.gz"
execom "/usr/bin/katello-service start"
# Postgres online database backup
# Extract schema names from "grep db_name
/etc/katello/katello-configure.conf"
execom "/sbin/runuser - postgres -c \"pg_dump -Fc candlepin > ${BDIR}/${
VER}/candlepin.dump\""
execom "/sbin/runuser - postgres -c \"pg_dump -Fc foreman > ${BDIR}/${
VER}/foreman.dump\""
# Mongodb online database backup
execom "/usr/bin/mongodump --host localhost --out ${BDIR}/${VER}/mongo_dump"
# End script
/bin/echo "Execution of $0 $1 at $VER finished with return = 0" >> $LOGFILE
2>&1
```

C.7 Satellite Restore Script

```
[root@satnode1 scripts]# cat satellite-restore.sh
#!/bin/bash
# Satellite 6.2
# Disclaimer | This script is NOT SUPPORTED by Red Hat Global Support
Service.
# Global Variables
export LOGFILE=/var/log/satellite6-restore.log
# First and unique argument must be back dir to restore
RESDIR=$1
# Function to launch commands
execom () {
command=$1
if [ ` /bin/echo $command | grep -P .+ | wc -l ` -ne 0 ]
then
datecom=`/bin/date +%Y%m%d%H%M%S`
/bin/echo "$datecom Executing command : '$command >> $LOGFILE 2>&1'" >>
$LOGFILE 2>&1
/bin/sh -c "$command >> $LOGFILE 2>&1"
RET=$?
/bin/echo "$datecom Return from command '$command >> $LOGFILE 2>&1' is
'$RET'" >> $LOGFILE 2>&1
if [ ! "$RET" == "0" ]
```

```
then
/bin/echo "$datecom Return from command '$command >> $LOGFILE 2>&1' '$RET' !
= '0' . Aborting script : 'exit 1'" >> $LOGFILE 2>&1
/bin/echo "$datecom Return from command '$command >> $LOGFILE 2>&1' '$RET' !
= '0' . Aborting script : 'exit 1'"
/usr/bin/katello-service stop >> $LOGFILE 2>&1
exit 1
fi
fi
}
# Script begin
/bin/echo "Starting execution of $0 $1 at $VER" >> $LOGFILE 2>&1
# Stop sat6
execom "/usr/bin/katello-service stop"
sleep 6
execom "/usr/bin/systemctl stop postgresql.service "
sleep 6
# Restore system files
execom "/bin/tar --selinux -xzvf ${RESDIR}/config_files.tar.gz -C /"
execom "/bin/tar --selinux -xzvf ${RESDIR}/pgsql_data.tar.gz -C /"
execom "/bin/tar --selinux -xzvf ${RESDIR}/mongo_data.tar.gz -C /"
execom "/usr/bin/rsync --partial --progress --delete -alv ${RESDIR}/pulp/
/var/lib/pulp/"
execom "/usr/bin/rsync --partial --progress --delete -alv ${RESDIR}/pulp/
/var/lib/pulp/"
# Drop the existing Red Hat Satellite PostgreSQL databases if any exist
execom "/usr/bin/systemctl start postgresql.service "
sleep 30
execom "/sbin/runuser - postgres -c \"dropdb candlepin\""
execom "/sbin/runuser - postgres -c \"dropdb foreman\""
# Restore Red Hat Satellite PostgreSQL databases with the following commands
execom "/sbin/runuser - postgres -c \"pg_restore -C -d postgres $
${RESDIR}/candlepin.dump\""
execom "/sbin/runuser - postgres -c \"pg_restore -C -d postgres $
${RESDIR}/foreman.dump\""
# Ensure MongoDB is running, delete the old data and restore new one
execom "/usr/bin/systemctl start mongod.service"
sleep 30
execom "/bin/echo 'db.dropDatabase();' | mongo pulp_database"
execom "/usr/bin/mongorestore --host localhost $
${RESDIR}/mongo_dump/pulp_database/"
# Start Sat6 and verify it
execom "/usr/bin/katello-service start"
execom "/usr/bin/systemctl start postgresql.service"
###execom "/sbin/reboot"
# End script
/bin/echo "Execution of $0 $1 at $VER finished with return = 0" >> $LOGFILE
2>&1
```


C.8 Create Pacemaker Resources Script

```
[root@satnode1 scripts]# cat pacemaker-resource.sh
#!/bin/bash
#This script creates all the required resources for the SAT62-HA pacemaker
cluster
#and sets colocation constraints and order constraints to ensure that all
the
#services are colocated on the same node and start in the correct order.
#VIP resource
#
pcs resource create VirtualIP IPaddr2 ip=10.19.2.10 cidr_netmask=24
# LVM and filesystem creation
pcs resource create sat_vg LVM volgrpname=sat_vg exclusive=true
pcs resource create fs_pulp Filesystem device="/dev/sat_vg/lv_pulp"
directory="/var/lib/pulp" fstype="xfs"
pcs resource create fs_candlepin Filesystem
device="/dev/sat_vg/lv_candlepin" directory="/var/lib/candlepin"
fstype="xfs"
pcs resource create fs_foreman Filesystem device="/dev/sat_vg/lv_foreman"
directory="/var/lib/foreman" fstype="xfs"
pcs resource create fs_puppet Filesystem device="/dev/sat_vg/lv_puppet"
directory="/var/lib/puppet" fstype="xfs"
pcs resource create fs_wwwpulp Filesystem device="/dev/sat_vg/lv_wwwpulp"
directory="/var/www/pulp" fstype="xfs"
pcs resource create fs_mongodb Filesystem device="/dev/sat_vg/lv_mongodb"
directory="/var/lib/mongodb" fstype="xfs"
pcs resource create fs_pgsqldata Filesystem device="/dev/sat_vg/lv_pgsqldata"
directory="/var/lib/pgsqldata" fstype="xfs"
pcs resource create fs_puppetenv Filesystem
device="/dev/sat_vg/lv_puppetenv" directory="/etc/puppet/environments"
fstype="xfs"
pcs resource create fs_tftpboot Filesystem device="/dev/sat_vg/lv_tftpboot"
directory="/var/lib/tftpboot" fstype="xfs"
pcs resource create fs_dhcp Filesystem device="/dev/sat_vg/lv_dhcp"
directory="/var/lib/dhcpd" fstype="xfs"
pcs resource create fs_named Filesystem device="/dev/sat_vg/lv_named"
directory="/var/named" fstype="xfs"
#
# Service resource creation
#
pcs resource create rs_dhcp systemd:dhcpd op monitor interval=10s op start
interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_named systemd:named op monitor interval=10s op start
interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_puppet systemd:puppet op monitor interval=10s op
start interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_pgsqldata systemd:postgresql op monitor interval=10s op
start interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_mongodb systemd:mongod op monitor interval=10s op
start interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_qpidd systemd:qpidd op monitor interval=10s op start
interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_qdrouterd systemd:qdrouterd op monitor interval=10s
```

```
op start interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_tomcat systemd:tomcat op monitor interval=10s op
start interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create op monitor interval=10s
pcs resource create rs_pulp_workers systemd:pulp_workers op monitor
interval=10s op start interval=0s timeout=100s op stop interval=0s
timeout=100s
pcs resource create rs_foreman-proxy systemd:foreman-proxy op monitor
interval=10s op start interval=0s timeout=100s op stop interval=0s
timeout=100s
pcs resource create rs_pulp_resource_manager systemd:pulp_resource_manager
op monitor interval=10s op start interval=0s timeout=100s op stop
interval=0s timeout=100s
pcs resource create rs_pulp_celerybeat systemd:pulp_celerybeat op monitor
interval=10s op start interval=0s timeout=100s op stop interval=0s
timeout=100s
pcs resource create rs_httpd systemd:httpd op monitor interval=10s op start
interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_foreman-tasks systemd:foreman-tasks op monitor
interval=10s timeout=30s op start interval=0s timeout=100s op stop
interval=0s timeout=100s
#
### Creating Constraints ###
#
# Colocation Constraints
#
pcs constraint colocation add sat_vg with VirtualIP
pcs constraint colocation set fs_dhcp fs_named fs_puppet fs_puppetenv
fs_pgsql fs_mongodb fs_pulp fs_wwwpulp fs_foreman fs_candlepin fs_tftpboot
sequential=false set sat_vg
pcs constraint colocation add rs_dhcp with fs_dhcp
pcs constraint colocation add rs_named with fs_named
pcs constraint colocation add rs_pgsql with fs_pgsql
pcs constraint colocation add rs_mongodb with fs_mongodb
pcs constraint colocation add rs_foreman-proxy with fs_foreman
pcs constraint colocation set fs_puppet fs_puppetenv rs_puppet
pcs constraint colocation set fs_pulp fs_wwwpulp rs_pulp_workers
pcs constraint colocation set VirtualIP rs_qpidd rs_qdrouterd
pcs constraint colocation add rs_tomcat with VirtualIP
pcs constraint colocation add rs_httpd with VirtualIP
pcs constraint colocation set fs_pulp fs_wwwpulp rs_pulp_resource_manager
pcs constraint colocation set fs_pulp fs_wwwpulp rs_pulp_celerybeat
pcs constraint colocation add rs_foreman-tasks with fs_foreman
#
# Order constraints to start service after the filesystems
#
pcs constraint order set sat_vg set fs_dhcp fs_named fs_puppet fs_puppetenv
fs_pgsql fs_mongodb fs_pulp fs_wwwpulp fs_foreman fs_candlepin fs_tftpboot
sequential=false
pcs constraint order VirtualIP then sat_vg
pcs constraint order set sat_vg fs_dhcp rs_dhcp
pcs constraint order set sat_vg fs_named rs_named
pcs constraint order set sat_vg fs_puppet fs_puppetenv rs_puppet
```



```
pcs constraint order set sat_vg fs_pgsql rs_pgsql
pcs constraint order set sat_vg fs_mongodb rs_mongodb
pcs constraint order set sat_vg fs_pulp fs_wwwpulp rs_pulp_workers
rs_pulp_celerybeat
pcs constraint order set sat_vg fs_foreman rs_foreman-proxy
pcs constraint order start fs_wwwpulp then rs_httpd
#
# Order constraints to start services after parent service
#
pcs constraint order set VirtualIP rs_qpidd rs_qdrouterd
pcs constraint order VirtualIP then rs_dhcp
pcs constraint order VirtualIP then rs_named

pcs constraint order set rs_puppet rs_dhcp rs_named rs_pgsql
sequential=false set rs_mongodb set rs_qpidd rs_qdrouterd sequential=false
set rs_tomcat rs_pulp_workers rs_pulp_celerybeat rs_pulp_resource_manager
rs_foreman-proxy sequential=false set rs_httpd rs_foreman-tasks
sequential=false
#
# Set default resource stickiness to avoid failback when the original node
is restored
#
pcs property set default-resource-stickiness=INFINITY
```

C.9 Capsule Backup Script:

```
[root@cap1a scripts]# cat capsule-backup.sh
datecom=`/bin/date +%Y%m%d%H%M%S`
/bin/echo "$datecom Executting command : '$command >> $LOGFILE 2>&1'" >>
$LOGFILE 2>&1
/bin/sh -c "$command >> $LOGFILE 2>&1"
RET=$?
/bin/echo "$datecom Return from command '$command >> $LOGFILE 2>&1' is
'$RET'" >> $LOGFILE 2>&1
if [ ! "$RET" == "0" ]
then
/bin/echo "$datecom Return from command '$command >> $LOGFILE 2>&1' '$RET' !
= '0' . Aborting script : 'exit 1'" >> $LOGFILE 2>&1
/bin/echo "$datecom Return from command '$command >> $LOGFILE 2>&1' '$RET' !
= '0' . Aborting script : 'exit 1'"
/usr/bin/katello-service start >> $LOGFILE 2>&1
exit 1
fi
fi
}
# Script begin
/bin/echo "Starting execution of $0 $1 at $VER" >> $LOGFILE 2>&1
# Generate skel dir with permissions
execom "/bin/mkdir -p ${BDIR}"
execom "/bin/chmod 755 ${BDIR}"
execom "/bin/mkdir -p ${BDIR}/${VER}"
#execom "/bin/chgrp postgres ${BDIR}/${VER}"
execom "/bin/chmod 770 ${BDIR}/${VER}"
# Backup the configuration and data files
execom "/bin/tar --selinux -czvf ${BDIR}/${VER}/config_files.tar.gz
$DIRECTORIES"
```

```
execom "/bin/tar --selinux -tzvf ${BDIR}/${VER}/config_files.tar.gz"
# Online Repositories Backup
/bin/echo "Trying to make online backup /var/lib/pulp and /var/www/pub via
rsync" >> $LOGFILE 2>&1
/usr/bin/rsync --partial --progress --delete -alv /var/lib/pulp ${BDIR}/$
{VER}/ >> $LOGFILE 2>&1
CHKREPO=$(/bin/find /var/lib/pulp -printf '%T@\n' | md5sum)
execom "/usr/bin/rsync --partial --progress --delete -alv /var/lib/pulp $
{BDIR}/${VER}/"
CHKREPOB=$(/bin/find /var/lib/pulp -printf '%T@\n' | md5sum)
/bin/echo "MD5 before tar: $CHKREPO after tar: $CHKREPOB" >> $LOGFILE 2>&1
if [ ! "$CHKREPO" == "$CHKREPOB" ]
then
/bin/echo "ERROR doing online repositories backup. Aborting backup process"
>> $LOGFILE 2>&1 exit 1
fi
# Optional: Onine Database Backup
# rhs6 must be stopped
execom "/usr/bin/katello-service stop"
execom "/bin/tar --selinux -czvf ${BDIR}/${VER}/mongo_data.tar.gz
/var/lib/mongodb"
execom "/bin/tar --selinux -tzvf ${BDIR}/${VER}/mongo_data.tar.gz"
execom "/usr/bin/katello-service start"
# Postgres online database backup
# Extract schema names from "grep db_name
/etc/katello/katello-configure.conf"
# End script
/bin/echo "Execution of $0 $1 at $VER finished with return = 0" >> $LOGFILE
2>&1
```

C.10 Mult-host Capsule Certificate Generation Script

```
[root@sat62-ha ~]# cat katello-multi-host-certs.sh
#!/bin/env bash

if [ -z "$1" -o -z "$2" ]; then
    echo "Usage: ./katello-multi-host-certs.sh real.name.example.com
alternative.name.example.com"
    exit 1
fi

FQDN=$1

shift
for alternative_fqdn in "$@"; do
    set_cname="$set_cname --set-cname $alternative_fqdn"
done

katello-ssl-tool --gen-server \
    --set-hostname $FQDN \
    --server-cert custom-cert.crt \
    --server-cert-req custom-cert.crt.req \
    --server-key custom-cert.key \
```

```
        $set_cname \  
        -p file:/etc/pki/katello/private/katello-default-ca.pwd \  
        --ca-cert ~/ssl-build/katello-default-ca.crt \  
        --ca-key ~/ssl-build/katello-default-ca.key  
  
#if [ "$FQDN" = $HOSTNAME ]; then  
#    katello-installer --certs-server-cert ~/ssl-build/$FQDN/custom-cert.crt \  
#    \  
#        --certs-server-cert-req ~/ssl-build/  
$FQDN/custom-cert.crt.req \  
#        --certs-server-key ~/ssl-build/$FQDN/custom-cert.key \  
#    \  
#        --certs-server-ca-cert  
~/ssl-build/katello-default-ca.crt \  
#        --certs-update-server --certs-update-server-ca  
#  
#else  
capsule-certs-generate --capsule-fqdn $FQDN \  
                        --certs-tar ~/certs-$FQDN.tar \  
                        --server-cert ~/ssl-build/  
$FQDN/custom-cert.crt \  
                        --server-cert-req ~/ssl-build/  
$FQDN/custom-cert.crt.req \  
                        --server-key ~/ssl-build/  
$FQDN/custom-cert.key \  
                        --server-ca-cert  
~/ssl-build/katello-default-ca.crt \  
                        --certs-update-server  
#fi
```

```
[root@cap1a ~]# /root/scripts/cat cap-resources
```

```
#!/bin/bash  
set -x  
#This script creates all the required resources for the SAT62-HA pacemaker  
cluster  
#and sets colocation constraints and order constraints to ensure that all  
the  
#services are colocated on the same node and start in the correct order.  
#VIP resource  
#  
###pcs resource create VirtualIP IPaddr2 ip=10.19.3.51 cidr_netmask=24  
# LVM and filesystem creation  
###/pcs resource create cap_vg LVM volgrpname=cap_vg exclusive=true  
pcs resource create fs_pulp Filesystem device="/dev/cap_vg/lv_pulp"  
directory="/var/lib/pulp" fstype="xfs"  
pcs resource create fs_puppet Filesystem device="/dev/cap_vg/lv_puppet"  
directory="/var/lib/puppet" fstype="xfs"  
pcs resource create fs_mongodb Filesystem device="/dev/cap_vg/lv_mongodb"  
directory="/var/lib/mongodb" fstype="xfs"  
pcs resource create fs_puppetenv Filesystem  
device="/dev/cap_vg/lv_puppetenv" directory="/etc/puppet/environments"  
fstype="xfs"  
pcs resource create fs_tftpboot Filesystem device="/dev/cap_vg/lv_tftpboot"  
directory="/var/lib/tftpboot" fstype="xfs"  
pcs resource create fs_dhcp Filesystem device="/dev/cap_vg/lv_dhcp"
```

```
directory="/var/lib/dhcpd" fstype="xfs"
pcs resource create fs_named Filesystem device="/dev/cap_vg/lv_named"
directory="/var/named" fstype="xfs"
pcs resource create fs_foreman_proxy Filesystem
device="/dev/cap_vg/lv_foreman_proxy" directory="/var/lib/foreman-proxy"
fstype="xfs"
#
# Service resource creation
#
pcs resource create rs_dhcp systemd:dhcpd op monitor interval=10s op start
interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_named systemd:named op monitor interval=10s op start
interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_puppet systemd:puppet op monitor interval=10s op
start interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_mongodb systemd:mongod op monitor interval=10s op
start interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_qpidd systemd:qpidd op monitor interval=10s op start
interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_qdrouterd systemd:qdrouterd op monitor interval=10s
op start interval=0s timeout=100s op stop interval=0s timeout=100s
pcs resource create rs_pulp_workers systemd:pulp_workers op monitor
interval=10s op start interval=0s timeout=100s op stop interval=0s
timeout=100s
pcs resource create rs_foreman_proxy systemd:foreman-proxy op monitor
interval=10s op start interval=0s timeout=100s op stop interval=0s
timeout=100s
pcs resource create rs_pulp_resource_manager systemd:pulp_resource_manager
op monitor interval=10s op start interval=0s timeout=100s op stop
interval=0s timeout=100s
pcs resource create rs_pulp_celerybeat systemd:pulp_celerybeat op monitor
interval=10s op start interval=0s timeout=100s op stop interval=0s
timeout=100s
pcs resource create rs_httpd systemd:httpd op monitor interval=10s op start
interval=0s timeout=100s op stop interval=0s timeout=100s
#
### Creating Constraints ###
#
# Colocation Constraints
#
pcs constraint colocation add cap_vg with VirtualIP
pcs constraint colocation set fs_dhcp fs_named fs_puppet fs_puppetenv
fs_mongodb fs_pulp fs_tftpboot fs_foreman_proxy sequential=false set cap_vg
pcs constraint colocation add rs_dhcp with fs_dhcp
pcs constraint colocation add rs_named with fs_named
pcs constraint colocation add rs_mongodb with fs_mongodb
pcs constraint colocation add rs_foreman_proxy with fs_foreman_proxy
pcs constraint colocation set fs_puppet fs_puppetenv rs_puppet
pcs constraint colocation set VirtualIP rs_qpidd rs_qdrouterd
pcs constraint colocation add rs_httpd with VirtualIP
#
# Order constraints to start service after the filesystems
#
```



```
pcs constraint order set cap_vg set fs_dhcp fs_named fs_puppet fs_puppetenv
fs_mongodb fs_pulp fs_tftpboot fs_foreman_proxy sequential=false
pcs constraint order VirtualIP then cap_vg
pcs constraint order set cap_vg fs_dhcp rs_dhcp
pcs constraint order set cap_vg fs_named rs_named
pcs constraint order set cap_vg fs_puppet fs_puppetenv rs_puppet
pcs constraint order set cap_vg fs_mongodb rs_mongodb
pcs constraint order set cap_vg fs_pulp rs_pulp_workers rs_pulp_celerybeat
pcs constraint order set cap_vg fs_foreman_proxy rs_foreman_proxy
#
# Order constraints to start services after parent service
#
pcs constraint order set VirtualIP rs_qpidd rs_qdrouterd
pcs constraint order VirtualIP then rs_dhcp
pcs constraint order VirtualIP then rs_named

pcs constraint order set rs_puppet rs_dhcp rs_named sequential=false set
rs_mongodb set rs_qpidd rs_qdrouterd sequential=false set rs_pulp_workers
rs_pulp_celerybeat rs_pulp_resource_manager rs_foreman_proxy
sequential=false set rs_httpd sequential=false
#
# Set default resource stickiness to avoid failback when the original node
is restored
#
pcs property set default-resource-stickiness=INFINITY
```

Appendix D: Contributors

Name	Contribution	Title
Brett Thurber	Technical and Review	Software Engineering Manager
Andrew Beekhof	Technical	Principal Software Engineer
Stephen Benjamin	Technical	Senior Software Engineer
David Caplan	Review	Product Manager
Rich Jerrido	Review	Principal Technical Product Marketing Manager

Appendix E: Revision History

Revision 1.0	Tuesday December 9, 2015	Balaji Jayavelu
Revision 2.0	Wednesday July 27, 2016	Balaji Jayavelu

