Using the vRealize Orchestrator Puppet Plug-In 1.0

vRealize Orchestrator

This document supports the version of each product listed and supports all subsequent versions until the document is replaced by a new edition. To check for more recent editions of this document, see http://www.vmware.com/support/pubs.

EN-001700-00

vmware®

You can find the most up-to-date technical documentation on the VMware Web site at:

http://www.vmware.com/support/

The VMware Web site also provides the latest product updates.

If you have comments about this documentation, submit your feedback to:

docfeedback@vmware.com

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Contents

Using the vRealize Orchestrator Puppet Plug-In 1.0

Using the vRealize Orchestrator Puppet Plug-In 1.0 provides information and instructions about how to configure and use the VMware® vRealize Orchestrator plug-in for Puppet.

Intended Audience

This information is intended for anyone who is installing and configuring the plug-in, using the plug-in API, or using the workflow library. The information in *Using the vRealize Orchestrator Puppet Plug-In 1.0* is written for experienced users who are familiar with VMware virtual machine technology, with Orchestrator workflow development, and with Puppet.

For more information about Orchestrator, see http://www.vmware.com/support/pubs/orchestrator_pubs.html.

VMware Technical Publications Glossary

VMware Technical Publications provides a glossary of terms that might be unfamiliar to you. For definitions of terms as they are used in VMware technical documentation, go to http://www.vmware.com/support/pubs.

Introduction to the vRealize Orchestrator Puppet Plug-In 1

The vRealize Orchestrator Puppet plug-in allows interaction between vRealize Orchestrator and Puppet. The Puppet plug-in exposes common puppet tasks as workflows.

- Register and manage multiple Puppet Masters in the vRealize Orchestrator inventory.

  The Puppet plug-in uses SSH to interact with the Puppet Master.
- Install and configure the Puppet agent on Linux and Windows-based nodes by using SSH and PowerShell.
- Manage node certificates to sign a certificate signing request (CSR) on the Puppet Master.
- Create and maintain the Hiera and Manifest classification files for node classification.
- Trigger an on-demand Puppet agent run on the node by using SSH and PowerShell.
- Perform node agent certificate removal from the Puppet Master.
- Experimental support for the call to the Rake API for node classification.

This chapter includes the following topics:

Role of vRealize Orchestrator with the Puppet Plug-In

You must use the Orchestrator configuration interface to install the Puppet plug-in. You can then use the Orchestrator client to run and create workflows and access the Puppet plug-in API.

vRealize Orchestrator powers the Puppet plug-in. Orchestrator is a development and process-automation platform that provides a library of extensible workflows to manage the vCenter infrastructure and other technologies.

Orchestrator allows integration with management and administration solutions through its open plug-in architecture. The Puppet plug-in is one example of a configuration management solution that you can integrate with Orchestrator.

Installing the Puppet Plug-In

To use the Puppet plug-in, you must install the plug-in in the Orchestrator configuration interface.

Puppet Plug-In Functional Prerequisites

To install and use the Puppet plug-in, your system must meet the following product prerequisites.

vRealize Orchestrator

Verify that you have a running instance of the embedded vRealize Orchestrator 6.0 server, the embedded vCenter Orchestrator 5.5 server, or vCenter Orchestrator 5.5.2.1.

You can log in to the Orchestrator configuration interface at http://*orchestrator_server*:8282.

For information about how to set up Orchestrator, see the *vRealize Orchestrator Installation and Configuration Guide*.

Puppet Master

The Puppet infrastructure must meet the following functional prerequisites:

- Verify that Puppet Enterprise 3.7.0, Puppet Enterprise 3.3, Puppet Open Source 3.7.1, or Puppet Open Source 3.6.2 is installed.
- Verify that you can connect to the Puppet Master using SSH from the Orchestrator server
- Verify that the SSH daemon on the Puppet Master allows multiple sessions.

  The SSH daemon parameter to support multiple sessions on the Puppet Master is in the configuration file `/etc/ssh/sshd_config`. The session parameter must be set to `MaxSession=10`.

vRealize Automation

The Puppet plug-in is supported on vRealize Automation 6.2 and vCloud Automation Center 6.1.

Operating Systems for the Puppet Node

The following operating systems are supported on the Puppet node:

- Centos 7.0
- Centos 6.5
- Centos 5.10
- Red Hat Enterprise Linux 7.0
- Red Hat Enterprise Linux 6.5
- SUSE Linux Enterprise Server 11 Service Pack 3
- Ubuntu 12.0.4
- Windows 8.1
- Windows 7.0
- Windows Server 2012 R2
- Windows Server 2008 R2

Verify that the WinRM HTTP or HTTPS protocol for Windows is enabled. See the Configure WinRM to Use HTTP and Configure WinRM to Use HTTPS topics in the VMware vRealize Orchestrator Plug-Ins Documentation Center.

Verify that enough memory is allocated on the node to run Puppet in a remote shell using WinRM and PowerShell. Use the `winrm set winrm/config/winrs @{MaxMemoryPerShellMB="4096"}` command to increase the memory allocation.

Install the Puppet Plug-In

You can install the Puppet plug-in from the Orchestrator configuration interface.

Prerequisites

- Verify that you are logged in to the Orchestrator configuration interface at http://*orchestrator_server*:8282.
- Verify that you have downloaded the `.vmoapp` file from http://www.vmware.com/products/datacenter-virtualization/vcenter-orchestrator/plugins.html.

Procedure

1. On the **General** tab, click **Install Application**.
2. Upload the Puppet plug-in.
   a. Click the magnifying glass icon.
   b. Select the `.vmoapp` file to install.
   c. Click **Open**.
   d. Click **Install**.

   A message appears after successful installation. The Puppet plug-in is installed without a tab in the Orchestrator configuration interface.
3. On the **Startup Options** tab, click **Restart service** to complete the plug-in installation.

Configuring the Puppet Plug-In 2

With the Puppet plug-in configuration workflows, you can register and manage the Puppet Master connection details in the vRealize Orchestrator inventory.

This chapter includes the following topics:

Add a Puppet Master

To perform node level configuration you must register a Puppet Master to the Puppet plug-in.

Prerequisites

- Verify that you are logged in to the Orchestrator client as an administrator.
- Verify that the supported version of the Puppet Master is installed. See "Puppet Plug-In Functional Prerequisites," on page 8.

Procedure

1 From the drop-down menu in the Orchestrator client, select **Run** or **Design**.

2 Click the **Workflows** view in the Orchestrator client left pane.

3 Select **Library > Puppet > Configuration**.

4 Right-click the **Add a Puppet Master** workflow and select **Start workflow**.

5 Enter a unique name in the **Puppet Master name** text box.

6 Enter the SSH information to connect to the Puppet Master host in the **IP Address** text box.

7 Accept the default Puppet Master SSH port number.

8 Enter the Puppet Master root login credentials.

9 Click **Submit** to run the workflow.

What to do next

Run a validation workflow to verify that the SSH connection to the Puppet Master or the login credentials are working. See "Validate the Puppet Master," on page 12.

Validate the Puppet Master

This workflow validates the SSH connection to the Puppet Master.

The workflow also refreshes the Puppet Master cached information in the Puppet plug-in such as, the Puppet Master version information stored in the vRealize Orchestrator inventory.

Prerequisites

- Verify that you are logged in to the Orchestrator client as an administrator.
- Verify that a Puppet Master is available.

Procedure

1. From the drop-down menu in the Orchestrator client, select **Run** or **Design**.
2. Click the **Workflows** view in the Orchestrator client left pane.
3. Select **Library > Puppet > Configuration**.
4. Right-click the **Validate a Puppet Master** workflow and select **Start workflow**.
5. Click the **Puppet Master** text box to select a Puppet Master.

   The Inventory dialog-box opens.
6. Expand the Puppet plug-in.
7. Select the Puppet Master to validate and click **Select**.
8. Click **Submit** to run the workflow.

If the login credential information is incorrect, the workflow displays the error message, `java.lang.RuntimeException: com.jcraft.jsch.JSchException: Auth cancel.`

You also receive an error message from the workflow if the SSH connection to the Puppet Master is not working or the Puppet Master is disabled.

Troubleshoot and fix the connection or invalid credential problem and rerun the validation workflow. Remove the disabled Puppet Master and rerun the validation workflow. See "Remove a Puppet Master," on page 13.

Update the Puppet Master

The Update a Puppet Master workflow updates existing Puppet Master properties, such as IP address, port number, and user login credentials.

Prerequisites

- Verify that you are logged in to the Orchestrator client as an administrator.
- Verify that the SSH network connectivity works between your vRealize Orchestrator instance and the Puppet Master.

Procedure

1. From the drop-down menu in the Orchestrator client, select **Run** or **Design**.
2. Click the **Workflows** view in the Orchestrator client left pane.
3. Select **Library > Puppet > Configuration**.
4. Right-click the **Update a Puppet Master** workflow and select **Start workflow**.

5 Click the **Puppet Master** text box to select a Puppet Master.

The Inventory dialog-box opens.

6 Expand the Puppet plug-in.

7 Select the Puppet Master to modify and click **Select**.

8 Enter the new Puppet Master name in the **Name** text box.

9 Enter the new SSH information to connect the Puppet Master host in the **IP Address** text box.

10 (Optional) Enter the new Puppet Master SSH port number.

11 Enter the new Puppet Master root login credentials.

12 Click **Submit** to run the workflow.

Remove a Puppet Master

The Remove a Puppet Master workflow removes a Puppet Master from the Puppet plug-in.

Prerequisites

- Verify that you are logged in to the Orchestrator client as an administrator.
- Verify that a Puppet Master is available.

Procedure

1 From the drop-down menu in the Orchestrator client, select **Run** or **Design**.

2 Click the **Workflows** view in the Orchestrator client left pane.

3 Select **Library > Puppet > Configuration**.

4 Right-click the **Remove a Puppet Master** workflow and select **Start workflow**.

5 Click the **Puppet Master** text box to select a Puppet Master.

The Inventory dialog-box opens.

6 Expand the Puppet plug-in.

7 Select the Puppet Master to remove and click **Select**.

8 Click **Submit** to run the workflow.

Using the Puppet Plug-In Workflows 3

The Puppet plug-in workflow library contains workflows that allow you to manage and classify nodes.

This chapter includes the following topics:

Using the Puppet Plug-In Inventory

You can use the **Inventory** view to run workflows on Puppet objects.

To display the workflows that are available for an inventory object, navigate to **Tools > User preferences > Inventory** and select the **Use contextual menu in inventory** check box. After the option is enabled, when you right-click an object in the Orchestrator inventory, all available workflows for the object are displayed.

Using the Puppet Plug-In Node Management Workflows

You can use the Puppet plug-in Node Management workflows to accept or revoke agent certificates on the Puppet Master, install, configure, and remediate Linux and Windows Puppet agents.

You can find these workflows on the **Workflows** view of the Orchestrator client, in the **Node Management** subdirectory of the Puppet plug-in library.

Table 3-1. Node Management Workflows

Workflow	Description
Install Linux Agent with SSH	Uses SSH to install the Puppet agent on a Linux node. To run this workflow, complete the following parameters: ■ Add a Puppet Master that was registered in the vRealize Orchestrator inventory in the Puppet Master text box. ■ (Optional) Override the default Puppet agent download path in the Installer Base URL text box. For example, the default agent download URLs for Puppet Open Source are `http://yum.puppetlabs.com` and `http://apt.puppetlabs.com`. For Puppet Enterprise, the Administrator must install the appropriate agent package from the Puppet Master package repository on the Puppet Master. ■ Enter the Linux node IP address in the **Hostname** text box. ■ Enter the root login credentials in the **Username** text box. ■ Enter the root password.
Configure Linux Agent with SSH	Uses SSH to configure the Puppet agent on a Linux node. To run this workflow, complete the following parameters: ■ Add a Puppet Master that was registered in the vRealize Orchestrator inventory in the Puppet Master text box. ■ Enter the Linux node IP address in the Puppet **Node Hostname** text box. ■ Enter the root login credentials in the Puppet **Node Username** text box. ■ Enter the root password. ■ Define an environment that the Puppet node belongs to such as test, QE, or production. ■ (Optional) Click **Yes** to register the Puppet Master host name to the IP address in the node hosts file.
Install Windows Agent with PowerShell	Uses PowerShell to install the Puppet agent on a Windows node. To run this workflow, complete the following parameters: ■ Add a Puppet Master that was registered in the vRealize Orchestrator inventory in the Puppet Master text box. ■ (Optional) Override the default Puppet agent download path in the **Installer Base URL** text box. The default agent download URL for the Puppet Enterprise `https://s3.amazonaws.com/pe-builds/released` and the Puppet Open Source `https://downloads.puppetlabs.com/windows`. ■ Select an HTTP or HTTPS protocol from the **WinRM transport protocol** drop-down menu. The WinRM protocol determines the authentication mechanism. The supported authentication mechanism is Basic. ■ Enter the Windows node IP address in the **Hostname** text box. ■ Enter the Administrator login credentials in the **Username** text box. ■ Enter the password.

Table 3-1. Node Management Workflows (Continued)

Workflow	Description
Configure Windows Agent with PowerShell	Uses PowerShell to configure the Puppet agent on a Windows node. To run this workflow, complete the following parameters: ■ Add a Puppet Master that was registered in the vRealize Orchestrator inventory in the Puppet Master text box. ■ Enter the password. ■ Select an HTTP or HTTPS protocol from the **WinRM transport protocol** drop-down menu. The WinRM protocol determines the authentication mechanism. The supported authentication mechanism is Basic. ■ Enter the Windows node IP address in the **Hostname** text box. ■ Enter the Administrator login credentials in the **Username** text box. ■ Enter the password. ■ Define an environment that the Puppet node belongs to such as test, QE, or production. ■ (Optional) Click **Yes** to register the Puppet Master host name to the IP address in the node hosts file.
Remediate Linux Node with SSH	Triggers an on-demand Puppet agent run of the Linux node. To run this workflow, complete the following parameters: ■ Enter the Linux node IP address in the **Hostname** text box. ■ Enter the root login credentials in the Puppet **Node Username** text box. ■ Enter the root password.
Remediate Windows Node with PowerShell	Triggers an on-demand Puppet agent run of the Windows node. To run this workflow, complete the following parameters: ■ Select an HTTP or HTTPS protocol from the **WinRM transport protocol** drop-down menu. The WinRM protocol determines the authentication mechanism. The supported authentication mechanism is Basic. ■ Enter the Windows node IP address in the **Hostname** text box. ■ Enter the Administrator login credentials in the **Username** text box. ■ Enter the password.

Table 3-1. Node Management Workflows (Continued)

Workflow	Description
Sign Node Certificate	Signs node certificate signing request on Puppet Master. To run this workflow, complete the following parameters: ■ Add a Puppet Master that was registered in the vRealize Orchestrator inventory in the Puppet Master text box. ■ Enter the node name to accept the certificate signing request on the Puppet Master. ■ Accept the default setting to allow the Puppet Master to retry the connection in case of a network lag.
Clean Node Certificate	Revokes applicable node certificate and removes all of the files related to that node from the Puppet Master certificate storage. To run this workflow, complete the following parameters: ■ Add a Puppet Master that was registered in the vRealize Orchestrator inventory in the Puppet Master text box. ■ Enter the node name to revoke the certificate from and remove from the Puppet certificate storage.

Using the Puppet Plug-In Hiera Workflows

You can use the Puppet plug-in Hiera workflows to classify nodes on a Puppet Master using Hiera files.

You can find these workflows on the **Workflows** view of the Orchestrator client, in the **Hiera** subdirectory of the Puppet plug-in library.

Table 3-2. Hiera Workflows

Workflow	Description
Classify Node with Hiera	Defines Puppet node classifications using Hiera files. To run this workflow, complete the following parameters: ■ Add a Puppet Master that was registered in the vRealize Orchestrator inventory in the Puppet Master text box. ■ Select the supported JSON or YAML Hiera backend type for the node classification. ■ Enter the Hiera data directory path. The Hiera data directory path is defined in the Hiera configuration file on the Puppet Master under the hierarchy settings. ■ Enter the node name to classify. ■ (Optional) Click **Yes** to replace the existing classification information. ■ Enter the class for the Puppet node. For example, you can use the Apache or Tomcat class. ■ (Optional) Override the default class parameters. The class parameter values, Array and Hashes use the JSON format.
Delete Node Hiera	Deletes the Hiera data files from the Hiera data directory. To run this workflow, complete the following parameters: ■ Add a Puppet Master that was registered in the vRealize Orchestrator inventory in the Puppet Master text box. ■ Select the supported JSON or YAML Hiera backend type for the node classification. ■ Enter the Hiera data directory path. The Hiera data directory path is defined in the Hiera configuration file on the Puppet Master under the hierarchy settings. ■ Enter the node name to remove the JSON or YAML Hiera file.

Using the Puppet Plug-In Manifest Workflows

You can use the Puppet plug-in Manifest workflows to classify nodes on a Puppet Master using Manifest files.

You can find these workflows on the **Workflows** view of the Orchestrator client, in the **Manifest** subdirectory of the Puppet plug-in library.

Table 3-3. Manifest Workflows

Workflow	Description
Classify Node with Manifest	Defines Puppet node classifications using Manifest files. To run this workflow, complete the following parameters: ■ Add a Puppet Master that was registered in the vRealize Orchestrator inventory in the Puppet Master text box. ■ Enter the Manifest data directory path. The Manifest data directory path is defined in the Puppet Master configuration. ■ Enter the node name to classify. ■ (Optional) Click **Yes** to replace the existing classification information. ■ Accept the default setting to trigger a Manifest reload. ■ Define an environment that the Puppet node belongs to such as test, QE, or production. ■ Enter the class for the Puppet node. For example, you can use the Apache or Tomcat class. ■ (Optional) Override the default class parameters. The class parameter values, Array and Hashes use the JSON format.
Delete Node Manifest	Deletes the Manifest data files from the Manifest directory. To run this workflow, complete the following parameters: ■ Add a Puppet Master that was registered in the vRealize Orchestrator inventory in the Puppet Master text box. ■ Enter the Manifest data directory path. The Manifest data directory path is defined in the Puppet Master configuration. ■ Enter the node name to remove the Manifest file.

Using the Puppet Plug-In Samples Workflows

The Puppet plug-in Samples workflows show how you can combine the workflows and create custom processes.

You can find these workflows on the **Workflows** view of the Orchestrator client, in the **Samples** subdirectory of the Puppet plug-in library.

These workflows automate the following tasks:

- Installing and configuring the Puppet agent.
- Registering the Puppet agent certificates with the Puppet Master.
- Classifying the node with a class or multiple classes on the Puppet Master.
- Triggering an on-demand Puppet run of the Puppet agent on the newly configured node.

NOTE While vRealize Orchestrator is querying the Puppet Master class parameter list to populate a newly set parameter, the vRealize Orchestrator user interface becomes unresponsive for 10 seconds.

Table 3-4. Samples Workflows

Workflow	Description
Single Class - Install, Configure, Sign, Classify and Remediate Node	Shows how you can connect multiple Puppet workflows to automate the process of adding and classifying a Puppet node. To run this workflow, complete the following parameters: ■ Add a Puppet Master that was registered in the vRealize Orchestrator inventory in the Puppet Master text box. ■ Define an environment that the Puppet node belongs to such as test, QE, or production. ■ Select a Manifest, Hiera, or experimental Rake classification from the drop-down menu. Set the applicable parameters for the classification type. See "Using the Puppet Plug-In Manifest Workflows," on page 19, "Using the Puppet Plug-In Hiera Workflows," on page 18, or "Experimental Puppet Plug-In Features," on page 23. **NOTE** After you set a classification type for a node, do not use another classification type on that node. ■ Select an agent action from the **Puppet Agent Setup Action** drop-down menu. The None action applies to a node that has an agent installed and configured with the Puppet Master. The node name for this action refers to the Puppet node to classify. The Install, Configure, and Sign action lets you automate installing the agent, configuring with the Puppet Master, and signing a node certificate request on the Puppet Master for a new node. The Configure and Sign action lets you automate the configuring with the Puppet Master, and signing a node certificate request on the Puppet Master for a node with the agent preinstalled. The Sign action lets you automate the signing of a node certificate request on the Puppet Master for an existing node. The node name for this action refers to the Puppet node to classify. ■ Accept the default setting to trigger an on-demand Puppet agent run. ■ Select the node operating system from the **Machine Type** drop-down menu. Set the applicable parameters for the operating system. See "Using the Puppet Plug-In Node Management Workflows," on page 15. ■ (Optional) Override the default Puppet agent download path in the Installer Base URL override text box. For example, the default agent download URLs for Puppet Open Source are `http://yum.puppetlabs.com` and `http://apt.puppetlabs.com`. For Puppet Enterprise, the Administrator must install the appropriate agent package from the Puppet Master package repository on the Puppet Master.

Table 3-4. Samples Workflows (Continued)

Workflow	Description
	The default Windows agent download URL for the Puppet Enterprise is `https://s3.amazonaws.com/pe-builds/released` and the Puppet Open Source is `https://downloads.puppetlabs.com/windows`. ■ (Optional) Click **Yes** to register the Puppet Master host name to the IP address in the node hosts file. ■ (Optional) Click **Yes** to replace the existing classification information. ■ (Optional) Enter the class to add to the Puppet node. For example, you can use the Apache or Tomcat class. ■ (Optional) Override the default class parameters. To override the parameter value remove the prefix `default` and the parentheses and add a new value. For example, the class parameter `default (80)` value changes to `90`.
Multiple Classes - Install, Configure, Sign, Classify and Remediate Node	Shows how you can connect multiple Puppet workflows to automate the process of adding a node to your Puppet environment and classify multiple classes at the same time. To run this workflow, complete the following parameters: ■ Add a Puppet Master that was registered in the vRealize Orchestrator inventory in the Puppet Master text box. ■ Define an environment that the Puppet node belongs to such as test, QE, or production. ■ Select a Manifest, Hiera, or experimental Rake classification from the drop-down menu. Set the applicable parameters for the classification type. See "Using the Puppet Plug-In Manifest Workflows," on page 19, "Using the Puppet Plug-In Hiera Workflows," on page 18, or "Experimental Puppet Plug-In Features," on page 23. **Note** After you set a classification type for a node, do not use another classification type on that node. ■ Select an agent action from the **Puppet Agent Setup Action** drop-down menu. The None action applies to a node that has an agent installed and configured with the Puppet Master. The node name for this action refers to the Puppet node to classify. The Install, Configure, and Sign action lets you automate installing the agent, configuring with the Puppet Master, and signing a node certificate request on the Puppet Master for a new node. The Configure and Sign action lets you automate the configuring with the Puppet Master, and signing a node certificate request on the Puppet Master for a node with the agent preinstalled.

Table 3-4. Samples Workflows (Continued)

Workflow	Description
	The Sign action lets you automate the signing of a node certificate request on the Puppet Master for an existing node. The node name for this action refers to the Puppet node to classify. ■ Accept the default setting to trigger an on-demand Puppet agent run. ■ Select the node operating system from the **Machine Type** drop-down menu. Set the applicable parameters for the operating system. See "Using the Puppet Plug-In Node Management Workflows," on page 15. ■ (Optional) Override the default Puppet agent download path in the Installer Base URL override text box. For example, the default agent download URLs for Puppet Open Source are `http://yum.puppetlabs.com` and `http://apt.puppetlabs.com`. For Puppet Enterprise, the Administrator must install the appropriate agent package from the Puppet Master package repository on the Puppet Master. The default Windows agent download URL for the Puppet Enterprise is `https://s3.amazonaws.com/pe-builds/released` and the Puppet Open Source is `https://downloads.puppetlabs.com/windows`. ■ (Optional) Click **Yes** to register the Puppet Master host name to the IP address in the node hosts file. ■ (Optional) Click **Yes** to replace the existing classification information. ■ (Optional) Enter the classes to add to the Puppet node. For example, you can use the Apache and Tomcat classes. ■ (Optional) Override the default class parameters. To override the parameter value remove the prefix `default` and the parentheses and add a new value. For example, the class parameter `default (80)` value changes to `90`.

Experimental Puppet Plug-In Features

The Puppet plug-in provides the experimental Rake workflows and caching features.

Using the Experimental Puppet Plug-In Rake Workflows

You can use the Puppet plug-in Rake workflows to add a node to the Puppet Dashboard or Console, add a class, retrieve the node class list and node list, and remove a node with Rake API.

You can find these workflows in the **Workflows** view of the Orchestrator client, in the **Rake** subdirectory of the Puppet plug-in library.

Table 3-5. Rake Workflows

Workflow	Description
Add a Node	Adds a node to the Puppet Dashboard or Console.
Classify Node with Rake	Defines Puppet node classifications using Rake API.
Delete Node Rake	Deletes the node from the Puppet Dashboard or Console.
Get Node Class List	Retrieves the node class list from Puppet Dashboard or Console.
Get Node List	Retrieves the node list from Puppet Dashboard or Console.

Experimental Puppet Plug-In Caching

You can find caching configuration details in the **Design > Configurations** view of the Orchestrator client, in the **Puppet** subdirectory.

Index