FEDERATION ENTERPRISE HYBRID CLOUD 3.5

Extending Oracle DBaaS with Puppet Enterprise

- Puppet Integration with Federation Enterprise Hybrid Cloud
- Puppet Provisioning for an Oracle Container Database (CDB) and Pluggable Database (PDB) on a Federation Enterprise Hybrid Cloud Virtual Machine
- PDB Migration between Oracle CDBs

ABSTRACT
This solution guide describes how to integrate Oracle-centric Puppet (Modules and Manifests) with virtual machines created through Federation Enterprise Hybrid Cloud 3.5.

January 2016
Federation Enterprise Hybrid Cloud 3.5: Extending Oracle DBaaS with Puppet Enterprise Solution Guide

Part Number H14726
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Executive Summary</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>Solution Architecture</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>Extending Oracle DBaaS for Federation Enterprise Hybrid Cloud 3.5 with Puppet</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>Provision an Oracle Database Virtual Machine on Federation Enterprise Hybrid Cloud 3.5 with Puppet</td>
<td>18</td>
</tr>
<tr>
<td>5</td>
<td>PDB Migration</td>
<td>29</td>
</tr>
<tr>
<td>6</td>
<td>Conclusion</td>
<td>33</td>
</tr>
<tr>
<td>7</td>
<td>References</td>
<td>35</td>
</tr>
<tr>
<td>A</td>
<td>Appendix A                                   Adding Puppet Master to vRealize Orchestrator</td>
<td>37</td>
</tr>
</tbody>
</table>
This chapter presents the following topics:

Document purpose ........................................................................................................ 5
Audience ..................................................................................................................... 5
Business challenge/value ............................................................................................. 5
Technology solution .................................................................................................... 5
Essential reading .......................................................................................................... 6
We value your feedback! ............................................................................................... 7
The Federation Enterprise Hybrid Cloud™ (FEHC) 3.5 solution supports a completely virtualized data center, fully defined and automated by software. This solution starts with a foundation that delivers IT as a Service (ITaaS) with options for high availability, backup and recovery, and disaster recovery. It also provides an extensible framework for creating and managing add-on services, either internally or using third party systems.

This Solution Guide shows how the Federation Enterprise Hybrid Cloud components, VMware vRealize® Orchestrator™ and VMware vRealize Automation™, work together with a third party system, Puppet Enterprise, to support infrastructure configuration and deliver Oracle Database as a Service (DBaaS).

This solution demonstrates how to integrate Puppet modules into EMC Federation Enterprise Hybrid Cloud for preparing, installing, and configuring Oracle 12c Container and Pluggable Databases, and provide Puppet-managed Oracle DBaaS with vRealize Orchestrator workflows.

This guide is intended for EMC® customers and qualified EMC partners. The guide assumes that users who intend to deploy this solution have the necessary training and background to install and configure Federation Enterprise Hybrid Cloud 3.5, Oracle Database 12c, and Puppet Enterprise. Users should also be familiar with the infrastructure and database security policies of the customer installation.

Many customers have implemented Puppet modules to manage their Oracle infrastructure and also made a substantial investment in scripts, procedures and code designed to automate and simplify their Oracle systems and software. Developing and using Puppet modules and methods is a critical part of their overall data center management strategy.

Customers are also looking for a way to automate common Oracle operations such as Oracle Pluggable Database (PDB) migration.

This solution guide demonstrates how a customer can move existing Puppet modules to the EMC FEHC with little effort, while continuing to derive value from their investment. It uses the vRealize Orchestrator Puppet Plug-in to utilize the customer’s existing library of Puppet modules in the new cloud-based service architecture. The customer’s existing investment in Puppet-related licenses, development time, and procedural refinement continues to provide value, without disruption to business processes.

This solution also demonstrates how to automate the migration of a PDB into an Oracle Container Database (CDB) deployed within the Federation Enterprise Hybrid Cloud.

The vRealize Orchestrator workflows that manage Oracle through Puppet provide a template which can be used for any Puppet modules, demonstrating the tight integration that can be achieved between Federation Enterprise Hybrid Cloud 3.5 and Puppet Enterprise.

Specifically, this solution covers:

- Deployment of Oracle DBaaS on a virtual machine managed by Puppet. This creates an Oracle 12c CDB and PDB in a single tenant configuration.
- Cloud-based migration of an Oracle PDB between Oracle 12c Container databases
Federation Enterprise Hybrid Cloud

Developed by EMC and VMware product and services teams, the Federation Enterprise Hybrid Cloud solution takes advantage of the strong integration between EMC technologies and the VMware vCloud® Suite. This solution includes EMC scalable storage arrays and integrated EMC and VMware monitoring and data protection to provide the foundation for cloud services within customer environments.

Federation Enterprise Hybrid Cloud is designed to deliver multitenancy and secure separation of resources. Cloud computing requires the ability to isolate resources and deliver secure access. Federation Enterprise Hybrid Cloud is structured so that each tenant can be, if required:

- Logically separated using vRealize Automation business groups. In this solution, the Oracle business group is assigned dedicated compute, storage, and network resources for software license compliance.
- Physically separated in a dedicated "workload pod." In this solution, Oracle is restricted to a dedicated vSphere High Availability (HA)/Distributed Resource Scheduler (DRS) cluster with its own servers, network, and storage resources, separate from all other tenants.

These dedicated, physically separated, clustered workload pods are mapped to the compute resource available to the vRealize Automation business group. This improves service availability and helps ensure software license compliance by restricting where the virtual machines and software are running or installed.

vRealize Orchestrator integrates with vRealize Automation and enables process automation when interfacing with third-party systems. Further, vRealize Orchestrator workflows can be deployed as custom services or actions in vRealize Automation to enable complex Day Two operations for Oracle DBaaS.

Puppet Enterprise

Puppet Enterprise is a tool that many IT administrators use to manage dozens, hundreds, or thousands of servers in their data centers. Puppet provides a way for administrators to describe the manner in which a particular machine (or machines) should be configured, and then continuously monitors those machines for "drift" from this configuration. If Puppet determines that the machine is out of its specified configuration, it remediates the machine to bring it back into configuration. Configuration descriptions can be as simple as the existence of a particular file, or as complex as entire applications being installed and running on the machine.

A Puppet-monitored machine might be out of configuration because it was changed for some reason or it might have never been in the configuration in the first place. Thus, if a description states that a machine should have Oracle database software installed on it, and that description is applied to a brand new machine, Puppet will, using methods configured in the description, install Oracle database software on the machine, and perhaps create a database or other associated actions.

The ability to leverage an existing library of Puppet modules that manage machine states can be critical for organizations making the move to the Federation Enterprise Hybrid Cloud. This solution shows how that kind of integration is simple and efficient.

Essential reading

The following documents describe the architecture, components, features, and functionality of the Federation Enterprise Hybrid Cloud 3.5 solution:

- Federation Enterprise Hybrid Cloud 3.5: Concepts and Architecture Guide
- Federation Enterprise Hybrid Cloud 3.5: Administration Guide
- Federation Enterprise Hybrid Cloud 3.5: Infrastructure & Operations Management Guide
We value your feedback!

EMC and the authors of this document welcome your feedback on the solution and the solution documentation. Contact EMC.Solution.Feedback@emc.com with your comments.

Authors: Tom Hudgins, Ed Spaenij, Liam Buckley, Allan Robertson, Reed Tucker
This chapter presents the following topics:

Overview ................................................................................................................. 9
Key components ........................................................................................................ 9
Software resources .................................................................................................... 11
Overview

This solution is primarily focused on the integration of Puppet Enterprise with Federation Enterprise Hybrid Cloud. Detailed discussion of the architecture, features, and functionality of Federation Enterprise Hybrid Cloud can be found in the *Federation Enterprise Hybrid Cloud 3.5: Foundation Infrastructure Reference Architecture Guide*.

Key components

This Oracle DBaaS solution uses Puppet Enterprise to deliver Oracle DBaaS on the ITaaS foundation provided by Federation Enterprise Hybrid Cloud (FEHC). The key components of this solution are as follows:

- FEHC (particularly the vRealize Automation and vRealize Orchestrator components)
- Puppet Enterprise
- Oracle 12c Database
- Oracle Enterprise Manager 12c

A simple representation of Federation Enterprise Hybrid Cloud single-site architecture is shown in Figure 1.

![Figure 1. Federation Enterprise Hybrid Cloud single-site architecture](image)

**Note:** For a detailed overview of the key components which make up the Core, Automation, NEI and Workload Pods, refer to the *Federation Enterprise Hybrid Cloud 3.5: Foundation Infrastructure Reference Architecture Guide*. 
A typical Puppet architecture is shown in Figure 2. It is an agent/master (client/server) architecture for configuring and monitoring systems, using the Puppet agent and Puppet master applications.

Figure 2. Puppet Enterprise and Oracle components

Figure 3 shows the interrelationship among the key components. It shows the Puppet Master server, which can be located either inside or outside the FEHC architecture. It also shows the vRealize Orchestrator Puppet plug-in, which is a key element in the integration.

The workload virtual machines show that their configuration is managed by Puppet. Puppet also automatically adds any provisioned Oracle database virtual machines to Oracle Enterprise Manager, which can be used for infrastructure monitoring. Existing services such as Oracle Enterprise Manager and Puppet can be brought inside FEHC, to enable them to be centrally managed and protected.

Figure 3. Federation Enterprise Hybrid Cloud solution components
Software resources

Table 1 lists the application software components and supporting services specific to this Oracle solution for Federation Enterprise Hybrid Cloud. For a complete list of Federation Enterprise Hybrid Cloud 3.5 software requirements, refer to the relevant EMC E-Lab EMC Simple Support Matrix at elabnavigator.emc.com.

<table>
<thead>
<tr>
<th>Software</th>
<th>Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating system software</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Red Hat Enterprise Linux (64 bit)</td>
<td>6.6</td>
<td>Red Hat Enterprise Linux (RHEL)</td>
</tr>
<tr>
<td>Application software</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oracle Database 12c Release 1</td>
<td>12.1.0.2.0</td>
<td>Oracle Database software including Oracle Enterprise Manager Database Express 12c</td>
</tr>
<tr>
<td>Application monitoring software</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oracle Enterprise Manager Cloud Control 12c</td>
<td>12.1.0.4</td>
<td>Oracle integrated enterprise IT management solution</td>
</tr>
<tr>
<td>Third party management software</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Puppet Enterprise</td>
<td>3.8.2</td>
<td>Puppet Enterprise including the Puppet Master and Puppet Agents</td>
</tr>
<tr>
<td>Extensibility Software</td>
<td></td>
<td></td>
</tr>
<tr>
<td>vRealize Orchestrator Puppet Plug-in</td>
<td>1.0.0</td>
<td>Enables vRealize Automation (vRealize Automation) and Puppet to work together to achieve infrastructure configuration, application deployment, and lifecycle management</td>
</tr>
</tbody>
</table>
Chapter 3
Extending Oracle DBaaS for Federation Enterprise Hybrid Cloud 3.5 with Puppet

This chapter presents the following topics:

Overview ............................................................................................................................ 13
Puppet Enterprise concepts and assumptions................................................................. 13
VMware vRealize Orchestrator Puppet plug-in............................................................... 14
Deploying the vRealize Orchestrator Puppet plug-in .................................................. 14
Adding a Puppet Master to vRealize Orchestrator ....................................................... 14
Using vRealize Orchestrator to install and configure the Puppet Agent .................... 15
Chapter 3: Extending Oracle DBaaS for Federation Enterprise Hybrid Cloud 3.5 with Puppet

Overview

The Federation Enterprise Hybrid Cloud has Service Catalog items and actions for self-service deployment and management of Oracle databases. Some companies use third party systems such as Puppet to manage OS and application configuration management, including Oracle databases. Since the Federation Enterprise Hybrid Cloud framework is extensible through integration with Puppet, EMC provides the ultimate flexibility for end users and IT organizations.

This chapter describes how Puppet Enterprise can be integrated into the EMC Federation Enterprise Hybrid Cloud using the vRealize Orchestrator Puppet plug-in and workflows. Applications deployed on virtual machines in the Federation Enterprise Hybrid Cloud can be monitored and managed by Puppet Enterprise.

While this solution focuses on using Puppet to manage Oracle databases and servers, the concepts and methods can be easily leveraged to incorporate any Puppet modules that may be used to manage deployed services.

Puppet Enterprise concepts and assumptions

Puppet Enterprise is a powerful and complex application that has the flexibility to manage machines across a wide range of architectures, operating systems, and installed applications. This kind of flexibility means that there are often many ways to accomplish the same things. Different organizations have different approaches for how they use Puppet in their data centers.

This solution assumes that Puppet Enterprise is installed, configured and being used to manage data center servers (nodes). When Puppet is integrated with Federation Enterprise Hybrid Cloud and specifically the vRealize Orchestrator component, connections between vRealize Orchestrator and the Puppet Enterprise Puppet Master server will need to be established.

This solution covers the installation of the Puppet agent software on virtual machines deployed from Federation Enterprise Hybrid Cloud. The major focus of this paper is demonstrating how machines deployed from Federation Enterprise Hybrid Cloud can be automatically configured with the Puppet Agent and then continually managed by Puppet.

Node classification with Hiera

A key concept in Puppet is node classification, which is the task of assigning specific configurations to certain machines. For example, an administrator might have a Puppet specification that describes a web server – the software that should be installed and running, the user accounts that should exist, firewall settings, etc. There would likely be specifications for other types of machines as well, for example a yum repository, an NTP server, a load balancer, or an Oracle database server. Some machines might have more than one of these specifications (often called profiles in Puppet). Node classification is the task of telling Puppet what specification (or specifications) each machine should have.

Node classification in Puppet can be performed by one of several methods – entries in the site.pp file, through the node classifier in the Puppet Enterprise console, or by using Hiera. Hiera provides great flexibility in node classification and promotes the separation of code and data, which promotes reuse. This solution incorporates the use of Hiera for node classification.
VMware vRealize Orchestrator Puppet plug-in

VMware vRealize Orchestrator workflows perform tasks related to deploying and configuring Virtual Machines inside the Federation Enterprise Hybrid Cloud.

vRealize Orchestrator supports plug-ins that extend the functionality of vRealize Orchestrator and VMware offers a plug-in for Puppet that forms the core of the integration between Federation Enterprise Hybrid Cloud and Puppet Enterprise. This plug-in provides workflows and actions that enable:

- Installation of the Puppet Agent
- Configuration of the Puppet Agent (connecting it to the Puppet Master)
- Node classification with Hiera or with Manifests
- Other Puppet-related functionality

Deploying the vRealize Orchestrator Puppet plug-in

You must download and install the Puppet Plug-in from the VMware Solution Exchange. Once installed, the Puppet plug-in workflows are located in the Library->Puppet folder, as shown in Figure 4.

![Figure 4. vRealize Orchestrator Puppet plug-in](image)

Adding a Puppet Master to vRealize Orchestrator

In order for the vRealize Orchestrator Puppet plug-in to communicate with the Puppet Master, a connection must be set up using a workflow in the Puppet plug-in. This must be done as part of the setup of this Puppet integration solution. Full instructions for adding a Puppet Master to vRealize Orchestrator are included in Appendix A. See the vRealize Orchestrator Puppet plug-in documentation for detailed information and troubleshooting.

Once the Puppet Master is configured in vRealize Orchestrator, you can use the other Puppet plug-in workflows to interact with Puppet. The plug-in workflows allow you to install the
Puppet Agent on a machine, assign Puppet classes to a node with either Hiera or through Manifests, and other Puppet operations. The next sections cover some of these operations.

While these steps can be performed manually with the Puppet plug-in workflows, the goal of this Federation Enterprise Hybrid Cloud solution is to integrate Puppet with cloud operations, such as deploying new Oracle database servers, so that these operations happen automatically. Custom workflows that provide the integration will be covered in the next chapter.

Using vRealize Orchestrator to install and configure the Puppet Agent

If you have an existing Federation Enterprise Hybrid Cloud virtual machine, you can use the vRealize Orchestrator Puppet plug-in to easily install the Puppet Agent on that machine.

To install the Agent, run the Install Linux Agent with SSH workflow in the Plug-in Node Management folder. Note that there is also a workflow for installing the Agent on a Windows machine using Powershell. This document focuses only on Linux operations.

The workflow prompts you for the following parameters:

**Table 2. Parameters for installing the Linux agent with SSH**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Puppet Master</td>
<td>This is the Puppet Master object that you set up previously. Use the integrated browser to choose the item.</td>
</tr>
<tr>
<td>Installer Base URL</td>
<td>This is a URL for the puppet agent installer packages. In many cases this can be left blank. The plug-in will retrieve the packages from standard Puppet repositories.</td>
</tr>
<tr>
<td>Hostname</td>
<td>This is the name of the host you want to install the Puppet agent onto. Typically this is the DNS name of the host.</td>
</tr>
<tr>
<td>Username</td>
<td>This is typically root. Installing the Puppet agent as another user is beyond the scope of this document.</td>
</tr>
<tr>
<td>Password</td>
<td>The password for the above user.</td>
</tr>
</tbody>
</table>

Once this workflow runs, the Puppet Agent software is installed on the designated virtual machine.

Configuring the Agent

After you install the Puppet Agent software, you must configure it and connect it with the Puppet Master. To do this, run the Configure Linux Agent with SSH workflow.

There is an issue between the current version of the vRealize Orchestrator plug-in and version 3.8.2 of Puppet Enterprise and associated Agent software. The vRealize Orchestrator plug-in Configure Linux Agent with SSH workflow contains a bash script that is executed on the Linux host. This script executes ‘puppet’ commands – that is, it expects to be able to find the puppet binary in its path. Earlier versions of the Puppet Agent software package contained symlinks that were created in /usr/local/bin, which pointed to the Puppet binaries in /opt/puppet/bin. Thus, as long as /usr/local/bin was in the root user’s path, the ‘puppet’ command would be found and would run.

In the current version of the Agent software installation package, those symlinks are not created until the Puppet Agent runs. Thus, when the configuration script attempts to run ‘puppet’ commands, they fail. A workaround is to edit the script contained in the workflow to add the /opt/puppet/bin directory to the PATH variable in the script.
To make this change, perform the following steps:

1. Select the **Configure Linux Agent with SSH** workflow.
2. Put the workflow into edit mode by clicking the pencil icon.
3. Click the **General** tab.
4. Click the Value field of the configureScript Attribute. This opens a text box that contains the script. You can resize this text box to make it easier to see the script.
5. Scroll to the top of the script and find the line that begins with “export PATH”.
6. At the end of this line, append the text, as shown in Figure 5:
   
   ```
   /opt/puppet/bin
   ```

   The full line should be (all on one line):

   ```
   export PATH=$PATH:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/opt/puppet/bin
   ```

   ![Figure 5. Linux Agent configuration PATH variable modification](image)

7. Click the **Ok** button to save this change and then click **Save and Close** to save the workflow.

This change enables the script to run ‘puppet’ commands and complete successfully. Among other actions, the configure workflow starts the Puppet agent on the host and, after a few minutes, the Puppet agent creates the symlinks to Puppet binaries in `/usr/local/bin`, so this workaround is only needed for this operation.

**Signing the node certificate**

Once the Puppet Agent is configured and the Agent is running, it registers itself with the specified Puppet Master. When the node registers with the Puppet Master, a notification appears in the Puppet Master console application that must be acknowledged by an Administrator before any Puppet modules can be run on the node. This manual step can be limiting if you are trying to automate the use of Puppet on a new virtual machine.

To automatically accept/sign the node certificate on the master, you can run the Puppet plug-in workflow **Sign Node Certificate**. This completes the process of registering the node to the Puppet Master so that Puppet modules applied to the node are processed by the Puppet Agent.
After you install and configure the Puppet Agent on a node, you need to tell Puppet which Puppet modules should be applied to this node. This is usually called “classifying” the node. As mentioned earlier, there are several ways to do this. See the Puppet Enterprise documentation for information on the options for classifying nodes.

One way to classify nodes is through the use of Hiera. The vRealize Orchestrator Puppet plug-in provides a workflow to accomplish this. When you run the workflow, you supply the node you want to classify, and the list of Puppet classes and their parameters that you want to assign to this node. The workflow then creates a Hiera YAML file for the node on the Puppet Master listing these classes and parameters. When the Puppet Agent runs on the node, Hiera reads this file and passes the information to Puppet to apply to the node.

The Puppet Agent runs every 30 minutes by default, but you can use the Remediate Node workflow to initiate an ad hoc run of the Puppet agent to apply your new classification immediately.

The next chapters take these core concepts and put them together into an integrated solution that installs and configures an Oracle database server that is deployed from Federation Enterprise Hybrid Cloud and is configured with Puppet modules.
Chapter 4 Provision an Oracle Database Virtual Machine on Federation Enterprise Hybrid Cloud 3.5 with Puppet

This chapter presents the following topics:

Overview ..................................................................................................................19
Puppet integration in Federation Enterprise Hybrid Cloud 3.5.................................19
Installing Oracle with Puppet ....................................................................................21
Puppet solution ..........................................................................................................23
Integration with Federation Enterprise Hybrid Cloud ..............................................26
Overview

This chapter describes the process of provisioning a virtual machine through Federation Enterprise Hybrid Cloud and using Puppet Enterprise to automatically configure it as an Oracle database server. This process involves steps in four main components – vRealize Orchestrator, vRealize Automation, VMware vCenter, and Puppet Enterprise. The overall process is:

1. Establish a virtual machine blueprint in vRealize Automation that is the base for the virtual machine and create a catalog item that uses this blueprint.
2. Create Puppet modules that install and configure the Oracle database software.
3. Create a workflow in vRealize Orchestrator that installs the Puppet Agent, configures it, adds it to the Puppet Master and signs the certificate, and classifies the node as an Oracle database server. Assign this workflow to the Machine Provisioned workflow stub in vRealize Automation.
4. Create a workflow in vRealize Orchestrator that requests the catalog item above.
5. Create a Catalog Item that invokes this workflow.

When the user requests the Oracle database server item in the catalog, the machine is provisioned and then Puppet automatically installs the Oracle software on it.

Puppet integration in Federation Enterprise Hybrid Cloud 3.5

Federation Enterprise Hybrid Cloud 3.5 integrates with Puppet through the use of vRealize Orchestrator and the vRealize Orchestrator Puppet plug-in.

Process overview

The high level steps involved in successfully deploying an Oracle database virtual machine using vRealize Automation, vRealize Orchestration, VMware vCenter and Puppet Enterprise are shown in Figure 6.
1. A user requests a vRealize Automation catalog item ‘Deploy Oracle VM’.
2. The request starts vRealize Orchestrator workflow ‘Deploy Oracle VM’, which:
   a. Asks the user for required input – such as oracle username and password, etc.
   b. Requests a Linux VM catalog item and passes on all user-specified attributes
3. The requested Linux catalog item is associated with a vRealize Automation blueprint, which is linked to a vCenter RHEL 6.5 virtual machine template. This vRealize Automation blueprint is configured with custom properties which hold the values of the user-specified input (the Oracle username, password, and other deployment options selected by the requesting user).
4. vCenter clones the new virtual machine from the template, starts the virtual machine and applies the customization specification.
5. The vRealize Automation blueprint also has a custom property set, which activates the MachineProvisioned workflow stub after the virtual machine is deployed.
6. The MachineProvisioned Workflow completes the following:
   a. Registers the new virtual machine in DNS
   b. Uses the vRealize Orchestrator Puppet plug-in to install Puppet Agent on the new virtual machine. In Puppet terms, the new virtual machine is a new ‘node’.
   c. Reads the custom properties from the virtual machine, including the user-specified inputs, and uses them as inputs for the Puppet plug-in. The Puppet plug-in then classifies the node with Hiera by writing the values to the node’s Hiera file on the Puppet Master.
d. Remediates the node, which immediately causes Puppet to apply the classes specified in the node’s Hiera file to the node. These Puppet classes contain the information that allows Puppet to install and configure Oracle on the node.

Developing with Puppet
The flexibility of Puppet means that any particular goal, such as installing Oracle software, can be solved in a number of different ways. This solution presents one approach.

A great benefit of using Puppet is the fact that many users have contributed modules into the Puppet Forge (forge.puppetlabs.com). Using these pre-built and tested modules greatly simplifies the process of using Puppet to accomplish a task. This solution combines several modules from the Puppet Forge along with some additional modules that are customized for this solution.

These modules are installed onto the Puppet Master in standard module locations. It is assumed that the reader is familiar with the use of Puppet modules both from the Puppet Forge as well as custom modules and knows how to integrate them into the Puppet Master.

Installing Oracle with Puppet
Before Puppet can be integrated with Federation Enterprise Hybrid Cloud, functioning Puppet code that installs Oracle software on virtual or physical machines is required. This code is integrated with Federation Enterprise Hybrid Cloud.

Puppet roles and profiles
Designing Puppet modules that do all of the things that are required to install and configure an Oracle database can be complex. Oracle prerequisites, such as kernel parameters and installed packages, have to be satisfied, disks have to be identified and configured, and the installation software has to be loaded and then executed. To help organize a complex Puppet configuration, a design pattern based on roles and profiles is often used.

Roles and profiles are Puppet modules that simply call (or include) other, lower level Puppet modules and thereby establish a hierarchy of module organization. A profile is a logical functionality stack that is implemented by lower level Puppet modules. An example of a profile might be “Oracle prerequisites”. Another example might be “Oracle installation media”. Several profiles, or chunks of functionality, might be required for a full Oracle installation. A shop might have profiles for many other types of servers and software as well. These profiles are then grouped together into a single role, “oracle” for example, in the role module, and this role is then assigned to the node. In this way, a node is only directly assigned one role – the role that defines what that node should be. Based on the profiles assigned to that role, and the lower level modules that actually do the work, Puppet configures the node to the desired state. Figure 7 shows the relationship of roles, profiles, modules and resources.

![Figure 7. Roles and profiles relationships](image)

In summary a roles and profiles design can be simplified as:

- A node includes only one role.
- A role includes one or more profiles to define the type of server.
- A profile includes and manages modules to define a technical stack.
- Modules manage resources.
- Modules are only responsible for managing aspects of the component for which they are written.
**Hiera**

Hiera is utilized in this solution to manage the environment configuration data. It enables a separation between Puppet code and data which results in a cleaner environment and the creation of more re-usable Puppet modules. With the data separated from the Puppet code some tasks are easier to complete, for example encrypting sensitive values, such as passwords.

Ultimately, Hiera and its associated YAML files define which Puppet module or modules should be assigned to a node and also any parameters those modules might require. A simple Hiera YAML file for an Oracle node, based on the use of Puppet roles, might look like this:

```yaml
classes:
  - 'roles::oracle'
  'roles::oracle::ora_sw__main_version': 'Oracle 12c'
  'roles::oracle::ora_sw__db_edition': 'Enterprise Edition'
```

This tells Puppet to assign the role 'oracle' to the node and also defines some parameters to use when setting up the role.

The combination of the Roles and Profiles design pattern and the use of Hiera for data separation can be visualized as shown in Figure 8.

---

**Figure 8.** Puppet module design
Chapter 4: Provision an Oracle Database Virtual Machine on Federation Enterprise Hybrid Cloud 3.5 with Puppet

Puppet solution

Using the Puppet Design principles described in the topic Developing with Puppet, we can build a set of Puppet modules, roles, and profiles that install and Configure Oracle on a new machine.

Puppet modules

This solution makes use of several modules from the Puppet Forge. These modules must be installed onto your Puppet Master (puppet module install <modulename>). The modules required are listed below in Table 3:

Table 3. Puppet Forge module descriptions

<table>
<thead>
<tr>
<th>Puppet Forge Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>biemond/oradb</td>
<td>Provides comprehensive control of Oracle software installation and database configuration.</td>
</tr>
<tr>
<td><a href="https://forge.puppetlabs.com/biemond/oradb">https://forge.puppetlabs.com/biemond/oradb</a></td>
<td></td>
</tr>
<tr>
<td>puppetlabs(concat)</td>
<td>The concat module lets you construct files from multiple ordered fragments of text.</td>
</tr>
<tr>
<td><a href="https://forge.puppetlabs.com/puppetlabs(concat)">https://forge.puppetlabs.com/puppetlabs(concat)</a></td>
<td></td>
</tr>
<tr>
<td>puppetlabs(stdlib)</td>
<td>Adds a standard library of resources for Puppet modules.</td>
</tr>
<tr>
<td><a href="https://forge.puppetlabs.com/puppetlabs(stdlib)">https://forge.puppetlabs.com/puppetlabs(stdlib)</a></td>
<td></td>
</tr>
<tr>
<td>fiddyspence(sysctl)</td>
<td>Edits Linux kernel params using sysctl.</td>
</tr>
<tr>
<td><a href="https://forge.puppetlabs.com/fiddyspence(sysctl)">https://forge.puppetlabs.com/fiddyspence(sysctl)</a></td>
<td></td>
</tr>
<tr>
<td>erwbgy/limits</td>
<td>Sets entries in /etc/security/limits.conf.</td>
</tr>
<tr>
<td><a href="https://forge.puppetlabs.com/erwbgy/limits">https://forge.puppetlabs.com/erwbgy/limits</a></td>
<td></td>
</tr>
<tr>
<td>puppetlabs/lvm</td>
<td>Provides Logical Resource Management (LVM) features for Puppet.</td>
</tr>
<tr>
<td><a href="https://forge.puppetlabs.com/puppetlabs(lvm)">https://forge.puppetlabs.com/puppetlabs(lvm)</a></td>
<td></td>
</tr>
</tbody>
</table>

Additionally, several custom modules, as shown in Table 4, are used to implement the roles and profiles design pattern, which leverage the modules from the forge.

Table 4. Custom Puppet Forge modules

<table>
<thead>
<tr>
<th>Custom Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roles</td>
<td>This module defines the Oracle role by referencing profiles in the profiles module.</td>
</tr>
<tr>
<td>Profiles</td>
<td>Defines the profiles that can be used in the definition of different roles.</td>
</tr>
</tbody>
</table>

Profiles and roles created in this solution are described in further detail in Table 5 and Table 6.

Profile and role definitions

A profile is a logical functionality stack that is implemented by lower level Puppet modules.

The profiles used in this solution are outlined in Table 5:

Table 5. Profile definitions

<table>
<thead>
<tr>
<th>Profile</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ora_media</td>
<td>• Ensures that an NFS Share containing the Oracle 12c installation files is mounted</td>
</tr>
</tbody>
</table>
### Table 6. Role definitions

<table>
<thead>
<tr>
<th>Role</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>oracle</td>
<td>Includes the following profiles:</td>
</tr>
<tr>
<td></td>
<td>- Oracle Media</td>
</tr>
<tr>
<td></td>
<td>- Oracle Database 12c prereqs</td>
</tr>
<tr>
<td></td>
<td>- Oracle Database 12c software</td>
</tr>
<tr>
<td></td>
<td>- Oracle Database 12c CDB</td>
</tr>
<tr>
<td></td>
<td>- Oracle Database 12c PDB</td>
</tr>
<tr>
<td></td>
<td>- Oracle Database 12c EM Agent</td>
</tr>
</tbody>
</table>

Together, these modules create a hierarchy, such that when the oracle role is applied to a node, Puppet installs and configures the Oracle Database. This hierarchy is outlined in Table 7.
Table 7. Puppet role to resource hierarchy

<table>
<thead>
<tr>
<th>Role</th>
<th>Profile</th>
<th>Module</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle</td>
<td>ora_media</td>
<td>Puppet core</td>
<td>mount</td>
</tr>
<tr>
<td></td>
<td>ora_preregs</td>
<td>Puppet core</td>
<td>file</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>group</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>user</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>package</td>
</tr>
<tr>
<td></td>
<td>puppetlabs/lvm</td>
<td>volume_group</td>
<td></td>
</tr>
<tr>
<td></td>
<td>fiddyspence/sysct</td>
<td>sysctl</td>
<td></td>
</tr>
<tr>
<td></td>
<td>erwbgy/limits</td>
<td>limits</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ora_sw</td>
<td>biemond/oradb</td>
<td>installdb</td>
</tr>
<tr>
<td></td>
<td>ora_cdb</td>
<td>biemond/oradb</td>
<td>database</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>autostartdatabase</td>
</tr>
<tr>
<td></td>
<td>ora_pdb</td>
<td>biemond/oradb</td>
<td>database_pluggable</td>
</tr>
<tr>
<td></td>
<td>ora_oem_agent</td>
<td>biemond/oradb</td>
<td>installem_agent</td>
</tr>
</tbody>
</table>

Node classification file

With all of the roles, profiles, and modules defined, we can now describe a Hiera YAML file that, when applied to a node, causes the Oracle software to be installed and configured. A sample Hiera YAML file for a node is listed below.

```yaml
classes:
- 'roles::oracle'
  'roles::oracle::ora_prereqs__ora_user': oracle
  'roles::oracle::ora_prereqs__ora_pwd': '****'
  'roles::oracle::ora_sw__main_version': 'Oracle 12c'
  'roles::oracle::ora_sw__db_edition': 'Enterprise Edition'
  'roles::oracle::ora_pdb__create_yn': 'True'
  'roles::oracle::ora_pdb__name': pdb1
  'roles::oracle::ora_pdb__admin_user': pdbadmin
  'roles::oracle::ora_pdb__admin_pwd': '****'
  'roles::oracle::ora_oem_agent__add_oem': 'False'
```

The first section specifies the Puppet classes that should be applied to the node. Per the roles and profiles design pattern, we have defined the Oracle role. Applying the Oracle role to the node causes the associated profiles and related Puppet modules to be classified to this node. The remaining entries are parameter values that are specific to this particular node. The user is prompted for these values when requesting the new Oracle server which are then passed to Puppet by being written into this Hiera file.

The '****' values for the password parameters in the sample above represent encrypted passwords. Puppet can be configured to automatically decrypt encrypted values in Hiera files by using the eYaml Ruby gem on the Puppet Master. The eYaml package also provides command line tools for encrypting and decrypting strings.

Since this file is what is required for Puppet to install and configure Oracle, the task of integrating Puppet with Federation Enterprise Hybrid Cloud comes down to configuring Federation Enterprise Hybrid Cloud to deploy the virtual machine, install the Puppet Agent on the virtual machine and configure it with the Puppet Master, and then create the Hiera YAML file. Once the virtual machine is deployed and the corresponding Hiera YAML file is written to the Puppet Master, Puppet remediates the node. It reads the Hiera YAML file and configures the node as an Oracle database server per the corresponding Puppet role and profiles associated with the role.
Integration with Federation Enterprise Hybrid Cloud

Integrating Puppet with Federation Enterprise Hybrid Cloud primarily involves installing and configuring the Puppet Agent on newly deployed virtual machines, and then classifying the virtual machines with the appropriate Puppet modules. This solution includes several vRealize Orchestrator workflows and actions that perform these steps as shown in Figure 6. The workflows leverage the VMware Puppet plug-in for vRealize Orchestrator.

To maximize flexibility, the vRealize Orchestrator workflows make use of parameter key-value pairs that are stored in configuration elements and attributes. The first element is OracleDBaaS and contains the attributes shown in Table 8:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmRootPassword</td>
<td>The root password for the VM template used</td>
</tr>
<tr>
<td>vCACHost</td>
<td>The vCAC Host</td>
</tr>
</tbody>
</table>

The second element is puppetInfo and contains the attributes shown in Table 9:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>puppetMaster</td>
<td>The Puppet Master object created by the Puppet Plug-in &quot;Add a Puppet Master&quot; workflow</td>
</tr>
<tr>
<td>hieraDataDir</td>
<td>The path to the Hiera data directory on the Puppet Master. For example,&quot;/etc/puppetlabs/puppet/environments/production/hieradata&quot;</td>
</tr>
<tr>
<td>environment</td>
<td>The name of the Puppet environment to use, for example, &quot;production&quot;</td>
</tr>
<tr>
<td>hieraBackend</td>
<td>YAML</td>
</tr>
</tbody>
</table>

Workflows: Deploy Oracle VM

The workflow that starts the process of requesting a virtual machine to use as an Oracle database is "Deploy Oracle VM", as shown in Figure 9.

Figure 9. Workflow: Deploy Oracle VM

This workflow makes a request for the vRealize Automation catalog item that corresponds to the blueprint for the virtual machine to deploy as the new Oracle Database server. The workflow takes options supplied by the requestor, such as Oracle username and password, and saves them as custom properties on the virtual machine.

This workflow is attached to a Service Blueprint in vRealize Automation called "Deploy Oracle VM" which appears in the vRealize Automation catalog. When the user requests that item, this workflow is run.
To automatically configure a new VM (node) with the Puppet Agent and subsequently configure it as an Oracle Database server, a workflow is attached to the Machine Provisioned workflow stub in vRealize Automation, as shown in Figure 10. This workflow is called by vRealize Automation once the virtual machine has been cloned and is ready for use.

**Workflow: Add VM to DNS**

This workflow, shown in Figure 11, uses the vRealize Orchestrator Powershell plug-in to execute a Powershell script, which adds the new virtual machine to the DNS server so it can be referenced by name in addition to IP address.

**Workflow: Install and Configure Puppet**

This workflow, shown in Figure 12, uses workflows from the vRealize Orchestrator Puppet Plug-in to install the Puppet agent on the new virtual machine, configure the Puppet Agent and add it to the Puppet Master. It then completes the installation by signing the node certificate on the Puppet Master.

**Workflow: Configure machine with Puppet**

This workflow, shown in Figure 13, creates the Hiera YAML file, which classifies the node in the Oracle role. The user-specified options (e.g., oracle username and password) are read from the virtual machine custom properties and used to create the node YAML file on the Puppet Master. That configuration is then applied to the node using the Remediate Linux Node workflow from the vRealize Orchestrator Puppet plug-in. This causes Puppet to apply all the profiles associated with the Oracle Role, causing the Oracle database software to be installed and initial databases to be created.
To clean up the Federation Enterprise Hybrid Cloud and Puppet systems when a virtual machine is destroyed, this workflow is attached to the Machine Disposing workflow stub and called by vRealize Automation when the user requests that the virtual machine be destroyed. This workflow, shown in Figure 14, deletes the nodes Hiera YAML file, unregisters the node from the Puppet Master, and removes the virtual machine from the DNS.
This chapter presents the following topics:

Overview .......................................................................................................................... 30
vRealize Orchestrator workflow design ........................................................................ 31
Chapter 5: PDB Migration

Overview

Once a database server is created in Federation Enterprise Hybrid Cloud, many ongoing maintenance and operational requirements still need to be performed.

This chapter shows how workflows can be built in Federation Enterprise Hybrid Cloud to enable the migration of an Oracle 12c PDB from one server to another. There are several options available for doing this, which are outlined in the next section, including the option selected for this solution and why it was chosen.

PDB copy options and choice

The options available to a DBA for copying a PDB are:

1. Copy PDB over DB Link, metadata only
2. Copy PDB over DB Link, including data
3. Transportable Tablespace
4. Cloning a PDB from an RMAN CDB backup set
5. Duplicating using RMAN on an active CDB
6. Unplug PDB from source and plug-in PDB to destination using manifest file (XML)

In deciding which alternative to use, there are several factors to be considered, such as:

- Is all or part of the data (database, tablespace, ...) available during the process?
- How accessible is the data (read-only or read-write) during the process?
- Are other tools needed ( datapump, RMAN, SQL*Plus, SSH, ...)?
- What is the impact on performance?
- What is the complexity?
- Do data governance, privacy, protection and ownership even allow copying this data?

In most business environments different teams or projects have a copy of a data set, for example for development, testing, QA, and so on. However, to comply with governance, privacy and protection rules, this data usually needs to be restricted (subsetted) and masked. Organizations have their own tools and procedures for these tasks, which are outside the scope of this document.

What does need to be copied is the database structures, or metadata. This can be supported by the Federation Enterprise Hybrid Cloud environment and made available to the end user through the Federation Enterprise Hybrid Cloud service catalog.

The option that is most efficient, least complex, has the least impact, and is fastest for copying the PDB metadata only is "Copy PDB over DB Link, metadata only." This creates a database link to the source database, and copies the PDB metadata over the link. It can be implemented in Oracle 12c, using only SQL.

To make this available for orchestration in the Federation Enterprise Hybrid Cloud, it is implemented in a vRealize Orchestrator workflow.
vRealize Orchestrator workflow design

The steps to perform the PDB copy comprise various SQL commands that need to be executed on the associated source and target CDBs. The most efficient way to execute these commands in vRO is to create an action that connects to the Oracle databases and executes the commands. This results in a single workflow that simply executes that action.

Workflow: Copy PDB

This workflow, shown in Figure 15, executes all required SQL to setup the database link and to copy the PDB.

All SQL statements are grouped in the action component.

The SQL statements are sent from vRealize Orchestrator to the Oracle Server using the SQL plug-in in vRealize Orchestrator. The SQL plug-in API, which is based on Java Database Connectivity (JDBC), provides a call-level API for SQL-based database access.

Figure 15. Copy PDB workflow

SQL steps

The statements and connections required are outlined below. It is a very efficient and low-impact solution to copy PDB metadata. The statements implemented assume that a PDB clone user already exists in the source (pluggable) database.

Source database steps

```
-- connected as a privileged user
ALTER SESSION SET CONTAINER=pdb1;
CREATE USER pdb_clone_user IDENTIFIED BY pdb_clone_user;
GRANT CREATE SESSION, CREATE PLUGGABLE DATABASE TO pdb_clone_user;
ALTER PLUGGABLE DATABASE pdb1 OPEN READ ONLY FORCE;
```

Target database steps

```
-- connected as a privileged user
CREATE USER c##pdb_clone_user IDENTIFIED BY pdb_clone_user;
GRANT CREATE SESSION, CREATE PLUGGABLE DATABASE, CREATE DATABASE LINK TO c##pdb_clone_user;
GRANT CREATE PLUGGABLE DATABASE TO c##pdb_clone_user;
GRANT SYSOPER TO c##pdb_clone_user;
CONN c##pdb_clone_user/pdb_clone_user;
CREATE DATABASE LINK pdbclone
    CONNECT TO pdb_clone_user IDENTIFIED BY pdb_clone_user
    USING '//server:port/pdb1.domain';
```
Chapter 5: PDB Migration

CREATE PLUGGABLE DATABASE pdb2
FROM pdb1@pdbclone
FILE_NAME_CONVERT = ('/u02/app/oracle/oradata/orcl/pdb1',
                     '/u02/app/oracle/oradata/orcl/pdb2')
NO DATA;

-- sysoper needed to alter pdb
CONN c##pdb_clone_user/pdb_clone_user AS SYSOPER;

-- for sysoper connection with jdbc
-- jdbc:oracle:thin:@server.domain:port/dbname.domain
-- user is 'c##pdb_clone_user as sysoper';

ALTER PLUGGABLE DATABASE pdb2 OPEN;
ALTER PLUGGABLE DATABASE pdb2 SAVE STATE;

Notes

(1) Source PDB mode

We have successfully run the PDB Copy with the source database in read write mode. However, the Oracle Documentation states to bring the source database to READ ONLY mode.

(2) NO DATA clause

Taken from the Oracle 12c SQL Language Reference, from statement CREATE PLUGGABLE DATABASE:

The NO DATA clause is available starting with Oracle Database 12c Release 1 (12.1.0.2).

The NO DATA clause applies only when cloning a PDB. This clause specifies that the source PDB's data model definition is cloned, but not the PDB's data. The dictionary data in the source PDB is cloned, but all user-created table and index data from the source PDB is discarded.

Restrictions on the NO DATA Clause

The following restrictions apply to the NO DATA clause:

- You cannot specify NO DATA when cloning a non-CDB.
- You cannot specify NO DATA if the source PDB contains clustered tables, Advanced Queuing (AQ) tables, index-organized tables, or tables that contain abstract data type columns.
This chapter presents the following topics:

Summary ........................................................................................................................................... 34
Findings ........................................................................................................................................... 34
Summary

Federation Enterprise Hybrid Cloud enables customers to build an enterprise-class, scalable, multitenant platform for complete infrastructure service lifecycle management.

The ability to integrate third party products like Puppet Enterprise into Federation Enterprise Hybrid Cloud allows users to exploit the full potential of Puppet Enterprise for management of any resources in a company’s environment.

The integration of Oracle Database as a Service into the Federation Enterprise Hybrid Cloud fully supports Oracle users with a self-service environment. This reduces “time to value” because it allows for more efficiency and a focus on tasks that matter. When there is a need to obtain a metadata copy of an Oracle 12c pluggable database, this can now be dealt with in an efficient manner, using the self-service capabilities of the Federation Enterprise Hybrid Cloud environment.

Findings

This solution enables resource management by integrating Federation Hybrid Cloud with Puppet Enterprise. The key findings of this solution are as follows:

- Puppet Enterprise integrates seamlessly in the Federation Enterprise Hybrid Cloud environment:
  - Both new and existing Puppet Modules can be used without any special modifications.
  - Combining the Roles and Profiles design pattern with Hiera provides a solid and robust method for Puppet to classify nodes and manage configuration data.
  - The Puppet plug-in for vRealize Orchestrator enables a full integration of any Puppet solution in the Federation Enterprise Hybrid Cloud environment.
- Integration of Oracle related tasks into the Federation Enterprise Hybrid Cloud self-service environment, such as obtaining a metadata copy of a pluggable database, has never been easier:
  - Using the vRealize Automation and Orchestration services, Oracle users can now develop the automation to enable self-service for obtaining a metadata copy of a Pluggable Database in an Oracle 12c multitenant environment.
  - The SQL plug-in for vRealize Orchestrator enables flexible integration of any Oracle SQL tasks, while keeping orchestration control inside vRealize Orchestrator.
This chapter presents the following topics:

- EMC Documentation ................................................................. 36
- VMware Documentation ............................................................ 36
- Oracle Documentation ............................................................... 36
- Other Documentation ............................................................... 36
EMC Documentation

The following documents, located on EMC Online Support or EMC.com, provide additional and relevant information. Access to these documents depends on your login credentials. If you do not have access to a document, contact your EMC representative.

These documents describe the architecture, components, features, and functionality of the Federation Enterprise Hybrid Cloud 3.5 solution:

- Federation Enterprise Hybrid Cloud 3.5: Foundation Infrastructure Reference Architecture Guide
- Federation Enterprise Hybrid Cloud 3.5: Concepts and Architecture Solution Guide
- Federation Enterprise Hybrid Cloud 3.5: Administration Guide

The following guide provides further information about various aspects of the Federation Enterprise Hybrid Cloud solution:

- Federation Enterprise Hybrid Cloud 3.5: Security Management Solution Guide

The following solution guide describes an Oracle Database as a Service solution:

- Federation Enterprise Hybrid Cloud 3.1: Oracle Database as a Service Solution Guide

VMware Documentation

Refer to the following documentation on the VMware website:

- VMware vRealize Automation documentation

Oracle Documentation

Refer to the following documentation on the Oracle website:

- Oracle Database 12c Documentation
- How To Create a Single PDB from another PDB (Doc ID 1991517.1)

Other Documentation

- Puppet Enterprise 3.8 Documentation
  http://docs.puppetlabs.com/pe/3.8/
- Designing Puppet – Roles and Profiles,
  https://puppetlabs.com/presentations/designing-puppet-rolesprofiles-pattern
- Best Practices for Building Puppet Modules,
  https://puppetlabs.com/blog/best-practices-building-puppet-modules
This appendix presents the following topics:

Adding Puppet Master to vRealize Orchestrator (Manually?) ........................................38
Appendix A: Adding Puppet Master to vRealize Orchestrator

Adding Puppet Master to vRealize Orchestrator

To add the Puppet Master to vRealize Orchestrator, perform the following steps:

1. Execute the Puppet plug-in **Add a Puppet Master** workflow. You must provide a name, which can be any name that helps to identify the Puppet Master. Also complete the IP Address, Port, and Username and password fields (typically the root user) as shown in Figure 16.

![Start Workflow: Add a Puppet Master](image)

**Figure 16. Workflow: Add a Puppet Master**

2. Once this workflow runs, a Puppet Resource appears in vRealize Orchestrator, which represents the connection to the Puppet Master.

3. For completeness, also run the **Validate a Puppet Master** workflow. This verifies the connection and ensures proper operation of the plug-in.