

## Puppet Enterprise and Razor on Cisco Unified Computing System (UCS)



When adding bare metal, Virtual Machine (VM), or container hosts to an application workload infrastructure, a server administrator would typically have to manually install the host OSes and configure various server, networking, storage, cluster, and other settings. This traditional manual build process could take hours or days, depending on the number of servers and configuration requirements. With Cisco UCS Manager integrated with Puppet Enterprise and Razor, multiple complex server builds can be programmatically defined and completed automatically in less than an hour.

### Solution Highlights

Cisco UCS, Puppet Enterprise, and Razor provide fast, flexible definition and management of complete infrastructure as code solutions. They can be used by administrators, developers, and operations staff for routine infrastructure lifecycle management, including the following:

- Day Zero configuration and server role assignment with UCS Service Profiles
  - An integral part of UCS programmability are its Service Profiles which provide identity for the infrastructure. The infrastructure elements are stateless, and the network, storage, and server profiles create the identity for each of these elements through UCS Manager. UCS's combination of a model driven architecture plus Service Profiles enables safe, fast automation.
- Day One rapid provisioning with Razor
  - Razor provides a rich set of programmatic interfaces for provisioning including fine grained control of the target OS (Red Hat, Windows, etc.) and target workload (the server's "role").
  - UCS Service Profiles allow for upfront, flexible definition of the intended server role which is passed automatically to Razor. They unify the process for provisioning compute, connectivity, storage and OS deployment in a single step, so they eliminate time-consuming provisioning processes across operational silos that often take days or weeks.

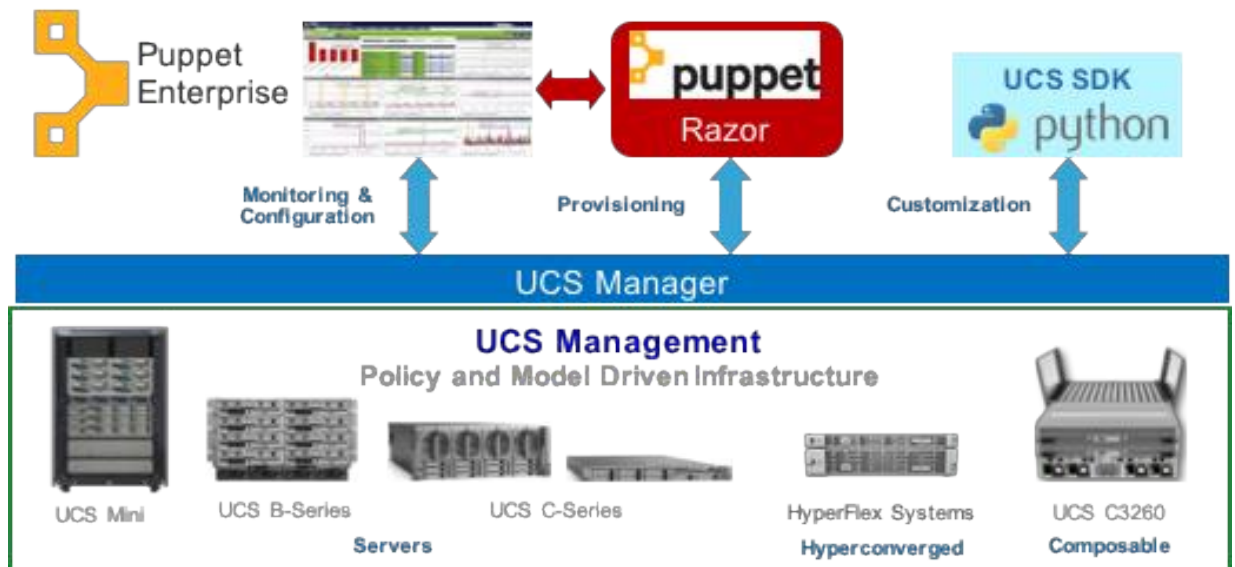
- Developers can use Razor to regularly wipe and re-provision test machines on Cisco UCS and Cisco HyperFlex systems.
- Razor seamlessly integrates with Puppet Enterprise and provisioned nodes are immediately brought under management.
- Day Two and beyond management and monitoring with Puppet Enterprise
  - Puppet Enterprise can install and manage all aspects of a target workload with information passed up from UCS Service Profiles. This allows for upfront definition of a target OS and workload which is automatically provisioned and brought into the desired state by Puppet Enterprise.
  - Puppet Enterprise’s rich set of modules allows for a wide range of managed workloads across all major OS, Virtual Machine, and Containerized environments.

Using Cisco UCS Manager integrated with Puppet Enterprise and Razor, this whitepaper will show:

- How quick and easy it is to create server workload “roles” via service profiles.
- Automatic installation of host OSES with Puppet Enterprise’s Razor.
- Automatic configuration of the hosts, including the ability to deploy and manage complete application workload stacks with Puppet Enterprise.

An overview of solution components is provided below.

**Figure 1.** Overview of Puppet Enterprise Integration with UCS Management



## Cisco Unified Computing System (UCS)

Cisco Unified Computing System (UCS) delivers smart, programmable infrastructure that simplifies and speeds enterprise-class application and service deployment in bare-metal, virtualized, and cloud-computing environments.

---

Unified, model-based management, end-to-end provisioning, and migration support come together in this next-generation data center platform to accelerate and simplify application deployment with greater reliability and security. UCS Manager automates the provisioning, configuration, and monitoring of the infrastructure. It includes an open API that serves as a unified control plane for integration with Puppet and a wide range of ISV configuration, orchestration and monitoring tools.

**Read more** [about Cisco Unified Computing System](#).

## Puppet Enterprise (PE)

Since 2005, Puppet has become one of the industry's most important, de facto standards for IT automation. Puppet Enterprise's extensible plug-in architecture, RESTful API, and powerful declarative language provide a flexible, easy to use platform that seamlessly integrates the unique capabilities of Cisco's UCS and Nexus solutions. The combination of Cisco UCS and Puppet Enterprise provides the following benefits:

**Unparalleled Cost Efficiencies:** Cisco UCS and Nexus open architectures provide Puppet Enterprise with extremely rich and detailed levels of metadata which Puppet Enterprise uses to eliminate costly, repetitive and error prone tasks.

**Accelerated Time to Value:** The model driven approaches of Cisco UCS Manager and Puppet Enterprise create a flexible platform that eliminates manual, time consuming tasks and provides the ability to quickly provision, deploy, and monitor mission critical applications in real time.

**Security from the Ground Up:** The granular levels of security embedded in the Cisco UCS and Puppet Enterprise solutions provide the ability to manage an entire application's infrastructure in a proactive and consistent way. This reduces risk and improves IT governance while at the same time providing a truly scalable infrastructure as code environment.

**Declarative Management:** Puppet Enterprise provides declarative management that allows the user to specify and track desired state without having to specify a procedure to achieve the desired state (the user declares how they want the system configured). Puppet Enterprise typically manages resources through agents that run on the resource or through broker services that interface with managed resources.

**Read more** [about Puppet Enterprise](#).

**Note:** You can [try Puppet Enterprise for free on up to 10 nodes](#).

## Automate Bare Metal Provisioning with Puppet's Razor Module

Puppet's Razor bare metal provisioning module works with Cisco UCS B and C series servers, UCS Mini, Cisco HyperFlex™ hyperconverged infrastructure and UCS C3260 composable infrastructure to create completely autonomous, end-to-end infrastructure as code solutions. Razor's open source heritage makes it highly customizable and superior to the more cumbersome menu driven approach of typical Preboot eXecution Environment (PXE) based systems.

Combining UCS hardware and integrated management with Puppet's Razor bare metal provisioning capabilities provides the following benefits:

**Operational Efficiency:** Puppet's Razor module eliminates wasteful, haphazard provisioning of hardware and operating system combinations by automatically matching the right hardware with the right OS at the right time.

**Simplicity:** Razor bare metal provisioning can be completed in a minimal number of steps. When combined with other PE tools, Razor can automatically discover, categorize, deploy and then boot the most appropriate OS and UCS hardware combinations.

**Security:** The combination of powerful UCS security features like TPM and Razor's node tagging and policy enforcement features provide an opportunity to instantly embed security from the ground up for security sensitive applications.

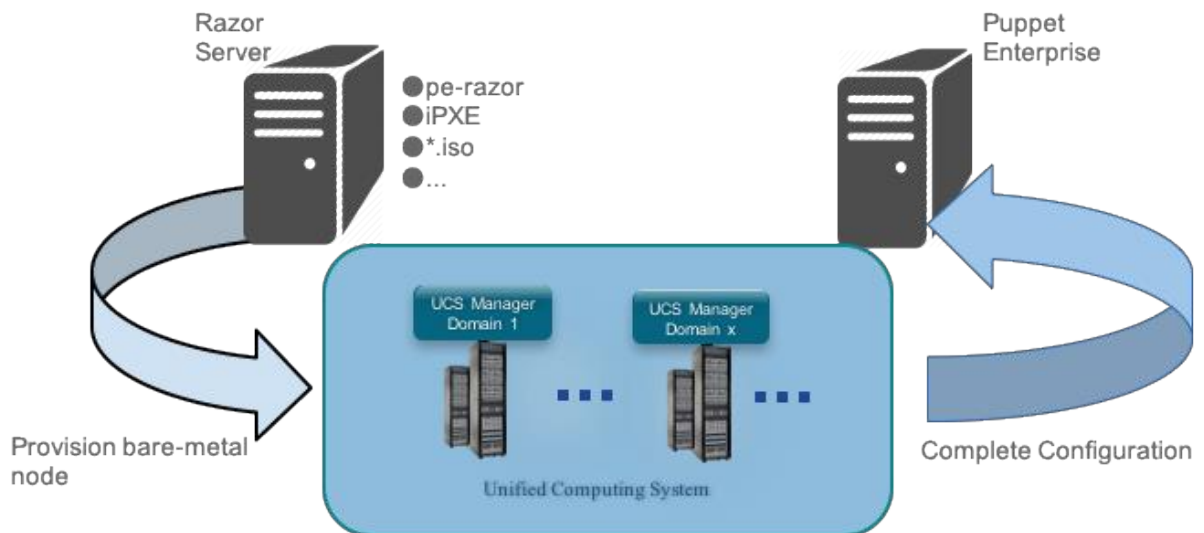
**Flexibility:** Razor can function as a standalone system or as part of a completely integrated Puppet Enterprise solution. Razor modules can be customized to meet a particular set of needs or it can be used in conjunction with other popular PE tools like Factor and MCollective to create a truly comprehensive solution.

Read more [about Puppet's Razor](#).

### Provisioning Infrastructure and Razor Overview

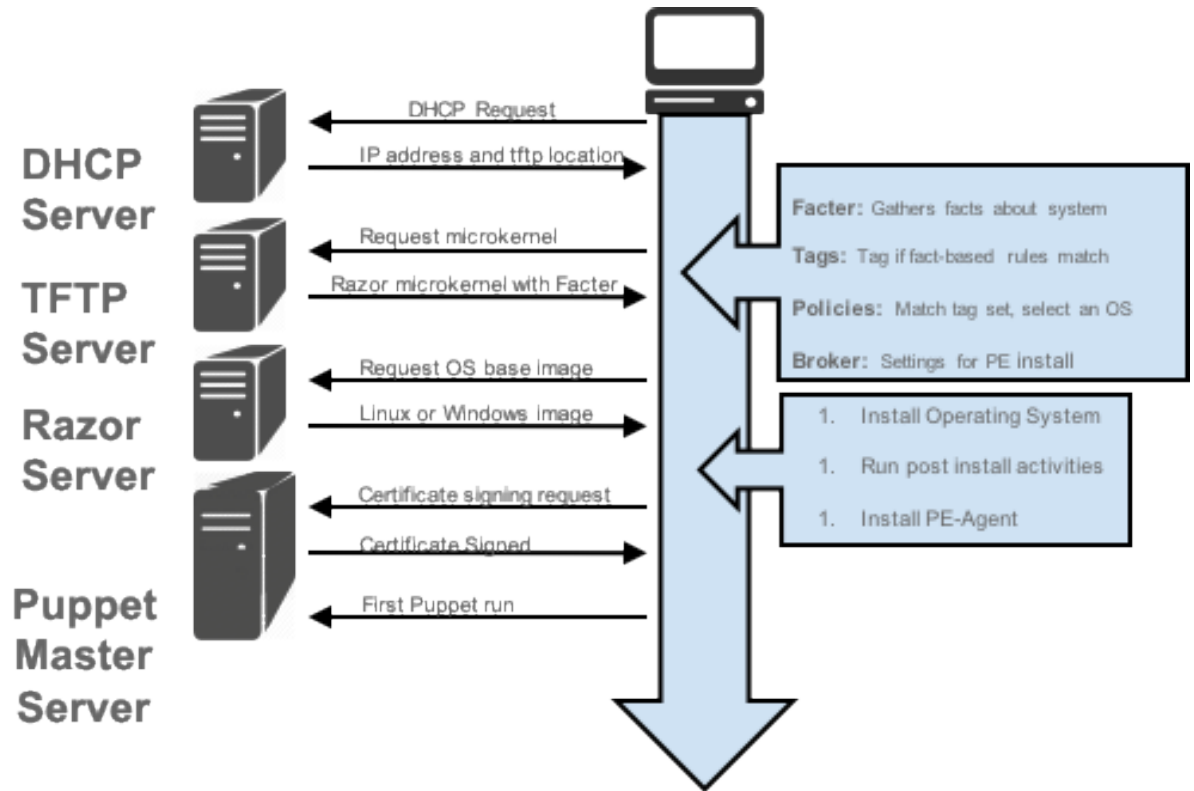
An example workflow is provided below and describes Puppet Enterprise (PE) managing nodes including a separate Razor server. In the example, host server roles are created in Razor and PE, roles are associated with servers through UCS Service Profiles, and Razor is used to provision hosts into the desired role.

**Figure 2.** Infrastructure Overview



The Razor server used to provision nodes also provides DHCP, TFTP, and PXE boot services to UCS host nodes. When a UCS node is PXE booted by Razor, Razor gathers facts about the node, matches facts to a policy, installs the appropriate OS, and brings the node under management by Puppet Enterprise.

**Figure 3.** Razor Provisioning Process



In the example below, policy matching is simplified by using UCSM Service Profiles to directly specify the intended server role. With UCSM and Service Profiles, the user is able to completely manage end-to-end server provisioning. Here is the simplified workflow:

- Create a server role (OS and application workload stack) and give it a name through tags and policies in Razor and PE.
- Create a UCSM Service Profile with the role name. Service Profiles allow for complete specification of server HW configuration (networking, BIOS policies, etc.) and allow for seamless migration of the profile from one physical server instance to another. When the Profile is assigned to a physical server, the server PXE boots and is provisioned by Razor.
- Razor installs a host OS for the assigned role and brings the server under management by Puppet Enterprise.

### Example Host Provisioning Workflow

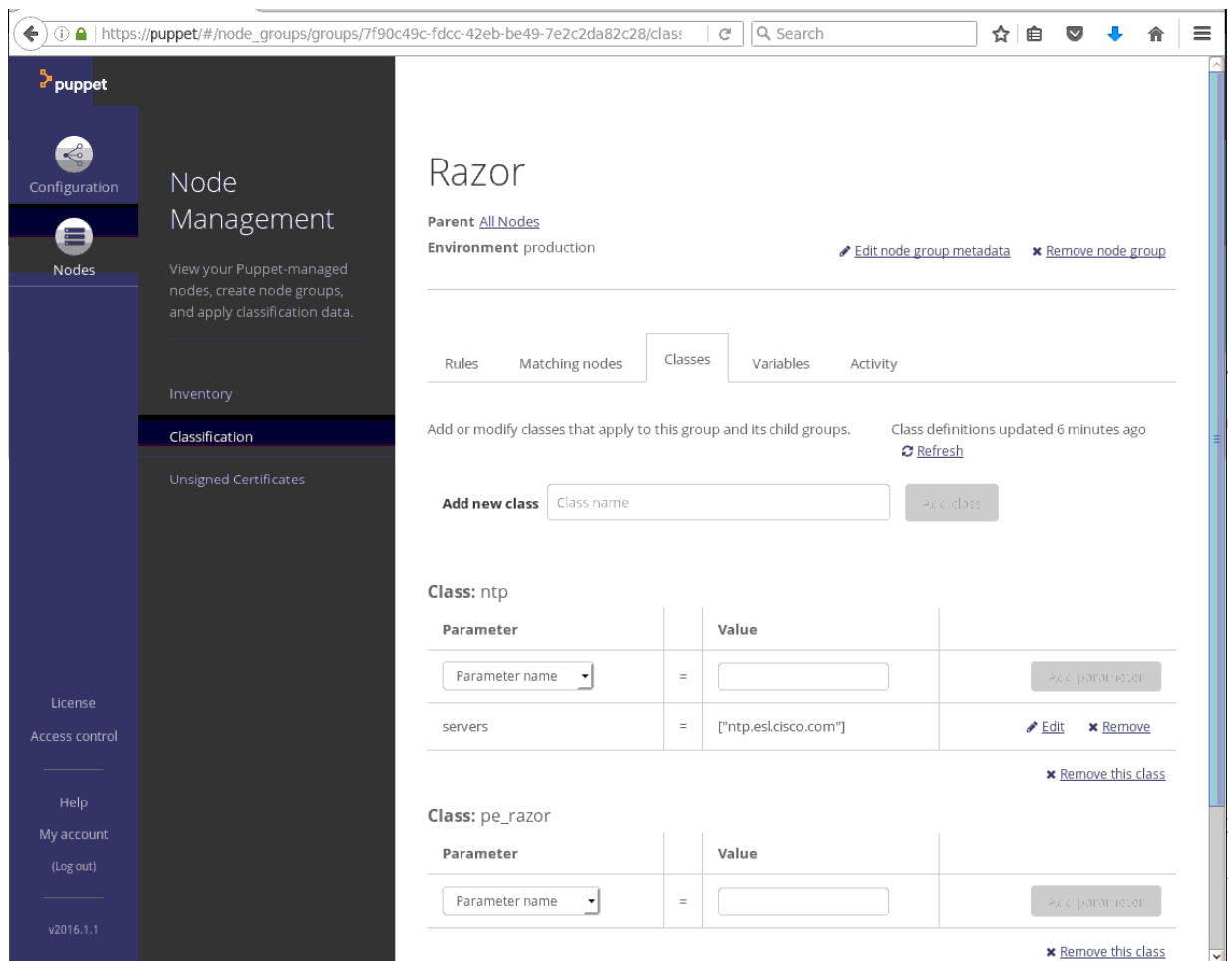
Puppet Enterprise and Razor are supported on several OS distributions. For detailed instructions, please see the following:

- [Puppet Enterprise installation instructions.](#)
- [Razor installation instructions.](#)

There are several options for running Puppet Enterprise (PE) and Razor, and the following setup assumes an existing Puppet Enterprise and Razor installation is in place along with basic DHCP, TFTP, and PXE services from the Razor host. The following steps were performed with PE and Razor installed on Red Hat Enterprise Linux (RHEL) with an active Red Hat subscription. Installation in other environments is possible, but those instructions are not provided here.

Once you have a Puppet Enterprise (PE) and Razor servers active in your environment, your Razor server should be part of your PE managed domain which will allow you to manage additional server deployments through the PE console.

**Figure 4.** PE Console showing example Razor node group and classes



With PE and Razor configured, you can assign server roles and automatically provision new hosts with the following steps:

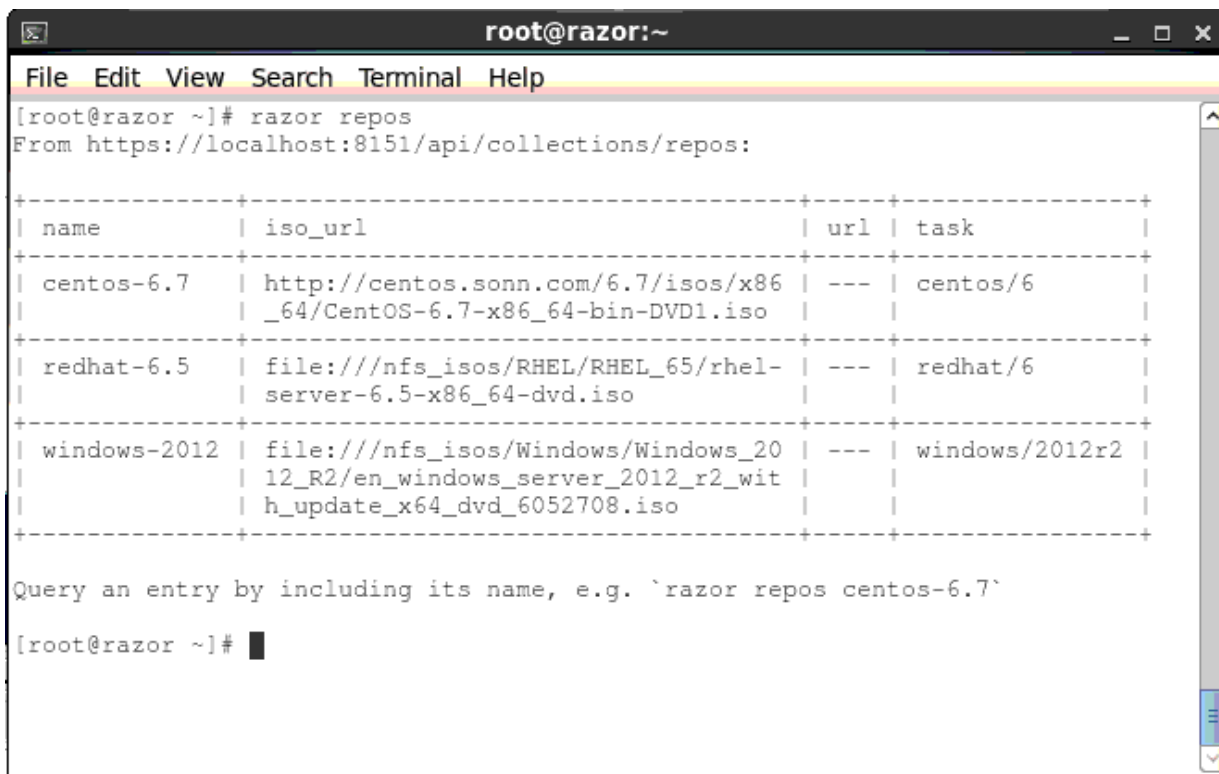
Step 1. Create Razor repos with desired host Operating Systems for deployment.

- a. Here is an example Razor command line to create a Red Hat 6.5 repo from a iso image on a local NFS share:

```
razor create-repo --name redhat-6.5 --task redhat/6 --iso-url
file:///nfs_isos/RHEL/RHEL_65/rhel-server-6.5-x86_64-dvd.iso
```

- b. You can check repos currently available with the “razor repos” command:

**Figure 5.** Current Razor repos



```
root@razor:~  
File Edit View Search Terminal Help  
[root@razor ~]# razor repos  
From https://localhost:8151/api/collections/repos:  


| name         | iso_url                                                                                             | url | task           |
|--------------|-----------------------------------------------------------------------------------------------------|-----|----------------|
| centos-6.7   | http://centos.sonn.com/6.7/isos/x86_64/CentOS-6.7-x86_64-bin-DVD1.iso                               | --- | centos/6       |
| redhat-6.5   | file:///nfs_isos/RHEL/RHEL_65/rhel-server-6.5-x86_64-dvd.iso                                        | --- | redhat/6       |
| windows-2012 | file:///nfs_isos/Windows/Windows_2012_R2/en_windows_server_2012_r2_wit_h_update_x64_dvd_6052708.iso | --- | windows/2012r2 |

  
Query an entry by including its name, e.g. `razor repos centos-6.7`  
[root@razor ~]#
```

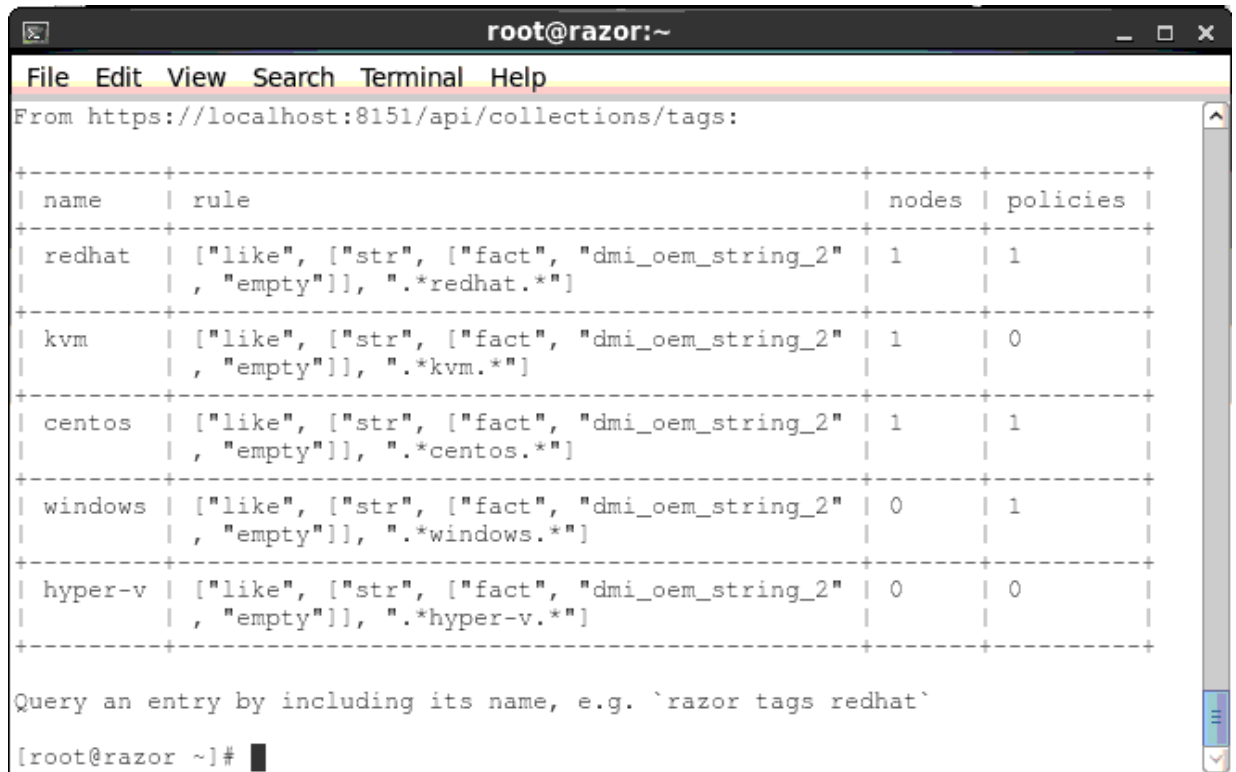
**Step 2.** Create a server role with Razor tags and policies.

- a. To determine a server’s intended role, Razor matches server facts with defined tags. Server roles are created with several tags to match against. For example, a “redhat” tag is used to determine that Red Hat should be installed (a later step will show how UCS Service Profiles populate the “dmi\_oem\_string\_2” tag being matched):

```
razor create-tag --name redhat --rule '["like", ["str", ["fact",  
"dmi_oem_string_2", "empty"]], ".*redhat.*"]'
```

- b. You can view currently defined tags with the “razor tags” command:

Figure 6. Current Razor tags



```
root@razor:~  
File Edit View Search Terminal Help  
From https://localhost:8151/api/collections/tags:  
-----  
| name | rule | nodes | policies |  
-----  
| redhat | ["like", ["str", ["fact", "dmi_oem_string_2" | 1 | 1  
| , "empty"]], ".*redhat.*"]  
-----  
| kvm | ["like", ["str", ["fact", "dmi_oem_string_2" | 1 | 0  
| , "empty"]], ".*kvm.*"]  
-----  
| centos | ["like", ["str", ["fact", "dmi_oem_string_2" | 1 | 1  
| , "empty"]], ".*centos.*"]  
-----  
| windows | ["like", ["str", ["fact", "dmi_oem_string_2" | 0 | 1  
| , "empty"]], ".*windows.*"]  
-----  
| hyper-v | ["like", ["str", ["fact", "dmi_oem_string_2" | 0 | 0  
| , "empty"]], ".*hyper-v.*"]  
-----  
Query an entry by including its name, e.g. `razor tags redhat`  
[root@razor ~]#
```

**Note:** Tags used in this example are visible to Razor using custom facts. Please see the [GitHub updated fact module for Razor](#) and the [GitHub custom extensions module for Razor](#) for more information.

Step 3. Configure Razor to assign servers to define roles.

- a. Razor policies control how nodes are provisioned. In this example, policies are set so that nodes with UCS Service Profiles matching certain tags are installed with the appropriate OS. For example, Red Hat installations will use the “redhat” tag to match the policy:

```
razor create-policy --name redhat --repo redhat-6.5 --broker pe-ucs --tag  
redhat --hostname 'redhat-${id}.ucsdemo.cisco.com' --root-password supersecret
```

**Note:** In this example, a custom pe-ucs broker is being used to pass additional role specifiers (e.g., “kvm” for a Red Hat KVM host) to Puppet Enterprise for complete workload install and configuration management. You can read more [about Razor brokers](#).

- b. Use the “razor policies” command to view current policies and status:



Figure 7. Razor policies



```
root@razor:~  
File Edit View Search Terminal Help  
[root@razor ~]# razor policies  
From https://localhost:8151/api/collections/policies:  


| name    | repo          | task            | broker | enable | max_co | tags     | nodes |
|---------|---------------|-----------------|--------|--------|--------|----------|-------|
| centos  | centos-6.7    | centos/6        | pe-ucs | true   |        | centos   | 1     |
| redhat  | redhat-6.5    | redhat/6        | pe-ucs | true   |        | redhat   | 1     |
| windows | window-s-2012 | window-s/2012r2 | pe-ucs | true   |        | window-s | 0     |

  
Query an entry by including its name, e.g. `razor policies centos`  
[root@razor ~]#
```

- Step 4. Create service profile templates and service profiles for the desired server roles. If you're unfamiliar with profile templates (or any other aspects of UCS), there are training demos on the [Cisco Demo Cloud \(dCloud\)](#). Search for "Cisco Unified Computing System".
- All aspects of each logical server's compute, network, and storage configuration can be specified through its service profile. This includes vNIC placement, PXE boot related networking configuration, and policies such as boot order.
  - Through service profiles, the intended server role is also defined by setting the profile name to match a defined Razor policy.

**Figure 8.** Service Profile template named to match a Razor policy

Create Service Profile Template

## Unified Computing System Manager

Create Service Profile Template

1. **Identify Service Profile Template**
2. Storage Provisioning
3. Networking
4. SAN Connectivity
5. Zoning
6. vNIC/vHBA Placement
7. vMedia Policy
8. Server Boot Order
9. Maintenance Policy
10. Server Assignment
11. Operational Policies

### Identify Service Profile Template

You must enter a name for the service profile template and specify the template type. You can also specify how a UUID will be assigned to this template and enter a description.

Name :

The template will be created in the following organization. Its name must be unique within this organization.  
Where : **org-root**

The template will be created in the following organization. Its name must be unique within this organization.  
Type :  Initial Template  Updating Template

Specify how the UUID will be assigned to the server associated with the service generated by this template.  
**UUID**

UUID Assignment:

The UUID will be assigned from the selected pool.  
The available/total UUIDs are displayed after the pool name.

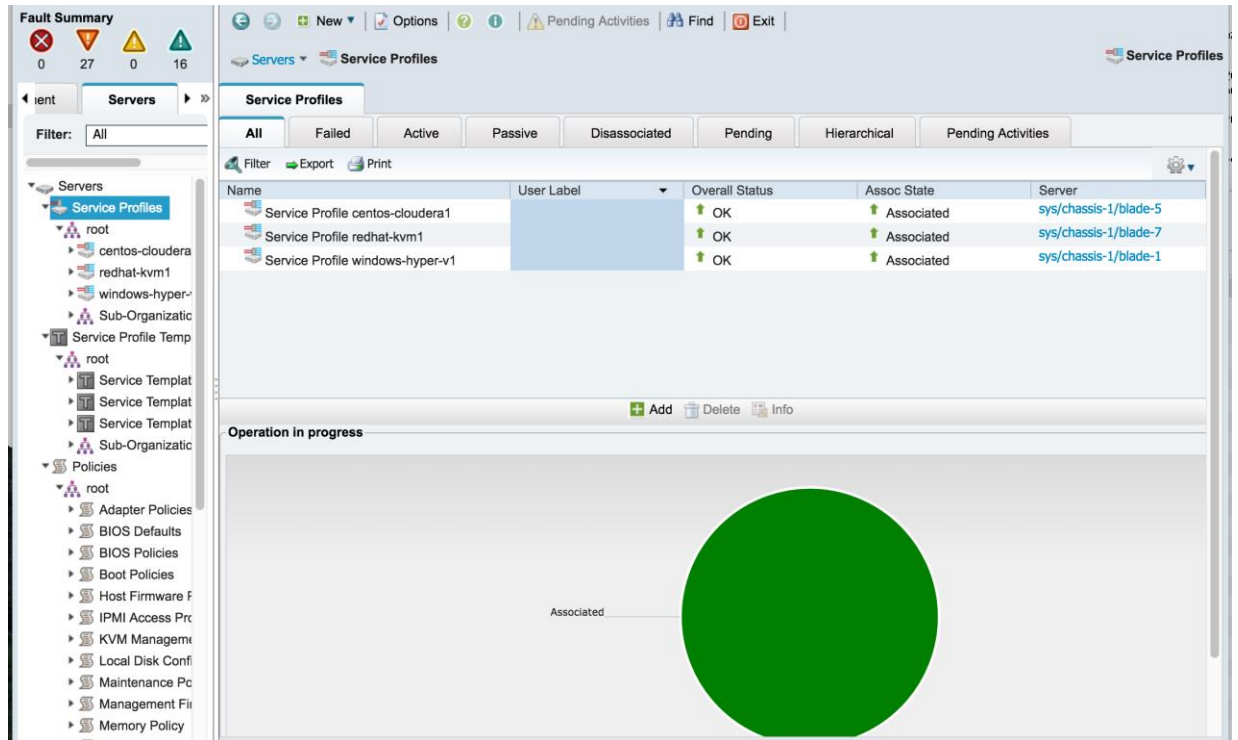
Optionally enter a description for the profile. The description can contain information about when and where the service profile should be used.

< Prev   Next >   Finish   Cancel

**Note:** The service profile name must contain an exact match of the role name created with a Razor tag (e.g., the substring “redhat” must be present in a service profile name to match the associated redhat Razor policy). The profile name can have other leading or trailing characters (e.g., “redhat-kvm1” will still match the “redhat” policy).

- c. When service profiles are associated with physical servers and the desired power state is “up”, each server will begin booting and is ready for provisioning.

**Figure 9.** Service Profiles Associated and Booting



**Note:** All aspects of service profile creation including policy definition can be automated with the UCS Manager Python SDK. To get started automating infrastructure as code with Cisco UCS, read more about the Cisco supported [Python SDK](#) and see the comprehensive set of [Python SDK sample code](#).

- Step 5. *Optional:* Observe server boot on the UCSM KVM (Keyboard, Video, Mouse) Console.
- a. Once service profiles are associated with a physical server, each server should be PXE booted by Razor. You can check boot progress on the KVM console.

Figure 10. KVM Console output of Razor microkernel PXE boot

```
File View Macros Tools Virtual Media Help
Boot Server Shutdown Server Reset
KVM Console Properties

All rights reserved

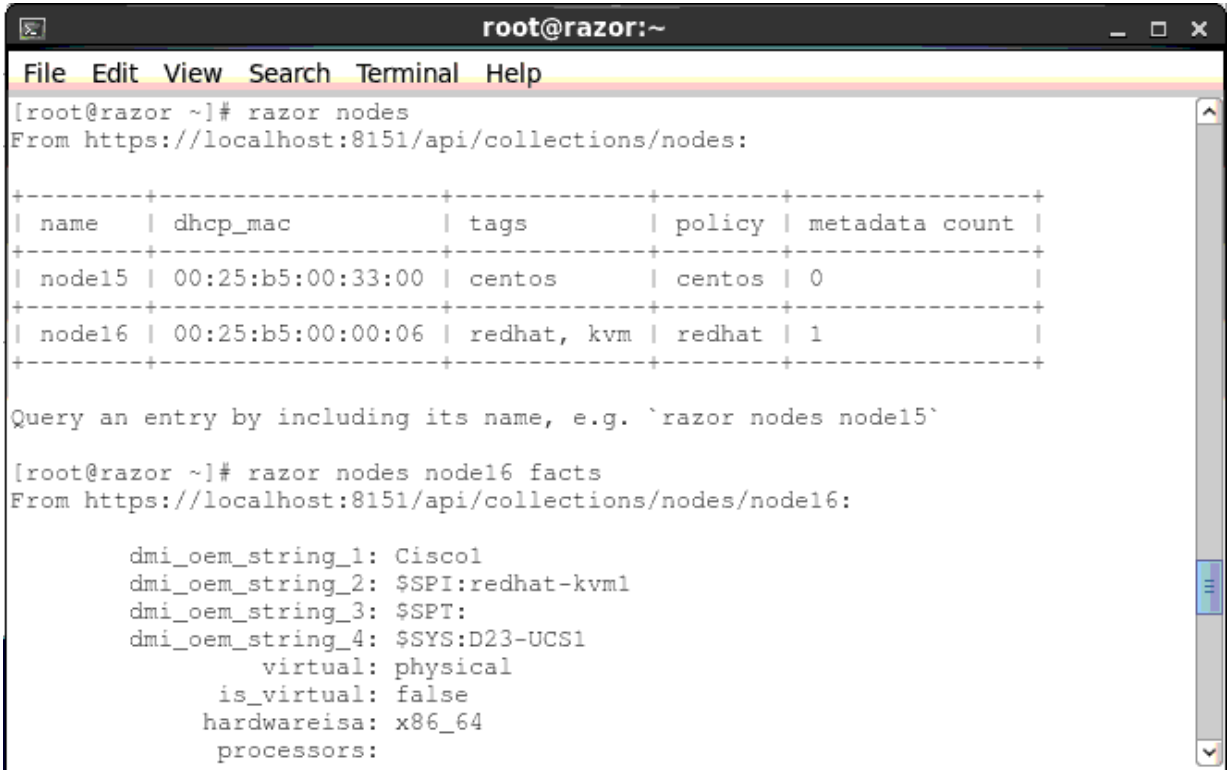
CLIENT MAC ADDR: 00 25 B5 00 33 00  GUID: 0C9B4A5E-6C5C-11E5-3300-000000000001
CLIENT IP: 192.168.1.142  MASK: 255.255.255.0  DHCP IP: 192.168.1.1
GATEWAY IP: 192.168.1.1
PXE->EB: !PXE at 9E19:0020, entry point at 9E19:0104
         UNDI code segment 9E19:0A70, data segment 99E9:4300 (615-635kB)
         UNDI device is PCI 07:00.0, type DIX+802.3
         615kB free base memory after PXE unload
iPXE initialising devices...ok

iPXE 1.0.0+ (c6c80) -- Open Source Network Boot Firmware -- http://ipxe.org
Features: HTTP iSCSI DNS TFTP AoE bzImage ELF MBOOT PXE PXEXT Menu

net0: 00:25:b5:00:33:00 using undionly on UNDI-PCI07:00.0 (open)
      [Link:up, TX:0 TXE:0 RX:0 RXE:0]
Configuring (net0 00:25:b5:00:33:00)... ok
net0: 192.168.1.142/255.255.255.0 gw 192.168.1.1
Next server: 192.168.1.1
Filename: bootstrap.ipxe
tftp://192.168.1.1/bootstrap.ipxe... ok
http://razor:8150/svc/boot?net0=00-25-b5-00-33-00&dhcp_mac=00-25-b5-00-33-00&ser
ial=FCH18427K31&asset=%20&uuid=5e4a9b0c-5c6c-e511-3300-000000000001...
```

- Step 6. *Optional:* Once each node boots and is claimed by Razor, the Razor server should report facts from the node including the intended role. In this example, the Service Profile name for the booted node is visible as a `dmi_oem_string_2` fact which can be matched against Razor tags.

Figure 11. Razor nodes and facts information



```
root@razor:~  
File Edit View Search Terminal Help  
[root@razor ~]# razor nodes  
From https://localhost:8151/api/collections/nodes:  
  
+-----+-----+-----+-----+-----+  
| name | dhcp_mac | tags | policy | metadata count |  
+-----+-----+-----+-----+-----+  
| node15 | 00:25:b5:00:33:00 | centos | centos | 0 |  
+-----+-----+-----+-----+-----+  
| node16 | 00:25:b5:00:00:06 | redhat, kvm | redhat | 1 |  
+-----+-----+-----+-----+-----+  
  
Query an entry by including its name, e.g. `razor nodes node15`  
  
[root@razor ~]# razor nodes node16 facts  
From https://localhost:8151/api/collections/nodes/node16:  
  
    dmi_oem_string_1: Cisco1  
    dmi_oem_string_2: $SPI:redhat-kvm1  
    dmi_oem_string_3: $SPT:  
    dmi_oem_string_4: $SYS:D23-UCS1  
                virtual: physical  
                is_virtual: false  
    hardwareisa: x86_64  
    processors:
```

- Step 7. With Puppet Enterprise (PE) used as the node broker, nodes should also be visible in assigned node groups and under management by PE.
- a. In this example, a RedHat-KVM node group has been created and NTP brought under management by Puppet Enterprise on the node.

Figure 12. Razor nodes and facts information

The screenshot displays the Puppet web interface for a node named 'redhat-host11'. On the left is a dark sidebar with the Puppet logo and navigation options: Configuration, Nodes (highlighted), Inventory, Classification, and Unsigned Certificates. The main content area shows the node name 'redhat-host11' with links for 'View node graph' and 'Run Puppet... Why?'. Below this is a horizontal menu with tabs for Facts, Classes (selected), Variables, Reports, Groups, and Activity. A table lists the classes applied to the node, with columns for 'Class' and 'Source group'.

Class	Source group
ntp	<a href="#">RedHat-KVM</a>
puppet_enterprise	<a href="#">PE Infrastructure</a>
puppet_enterprise::profile::agent	<a href="#">PE Agent</a>
puppet_enterprise::profile::mcollective::agent	<a href="#">PE MCollective</a>

---

## For More Information

Read more on the [Cisco Marketplace for Puppet Enterprise and Razor on Cisco UCS and Nexus](#).

Visit the [Cisco UCS Communities pages](#) for additional information, video demos, and support forums.

## Solution Q&A

- Q.** What UCS Hardware is supported by this solution?
- A.** Any UCS Manager domain can have roles assigned through Service Profiles that are automatically passed to Razor and Puppet Enterprise.
- Q.** How do I add Razor to an existing Puppet Enterprise environment?
- A.** Razor is part of Puppet Enterprise (PE) and PE can manage your Razor server in the same way other nodes are managed. Read more [about Razor](#).
- Q.** Can I use the UCS Platform Emulator with Puppet Enterprise and Razor?
- A.** The Platform Emulator does not provide host OS boot capabilities, but all aspects of Service Profile and UCS Manager policy setting can be tested and verified on the Platform Emulator. Read more [about the UCS Platform Emulator](#).



---

**Americas Headquarters**  
Cisco Systems, Inc.  
San Jose, CA

**Asia Pacific Headquarters**  
Cisco Systems (USA) Pte. Ltd.  
Singapore

**Europe Headquarters**  
Cisco Systems International BV Amsterdam,  
The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at [www.cisco.com/go/offices](http://www.cisco.com/go/offices).

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: [www.cisco.com/go/trademarks](http://www.cisco.com/go/trademarks). Third party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)