Chef and HPE OneView Integration

Contents

Technical white paper

Executive summary

As information technology becomes an integral part of products, services, and experiences, a new style of infrastructure is emerging where resources can be composed, decomposed, and recomposed from a fluid pool of compute, storage, and fabric infrastructure. To respond to the requirements of different applications, this infrastructure must be software-defined, with template-driven provisioning and management to provide “infrastructure as code”. HPE’s vision of Composable Infrastructure provides this fluid pool of resources and, within that vision, HPE OneView provides the templates and APIs to deliver software-defined control of the infrastructure.

To connect a Composable Infrastructure with end-to-end provisioning of application environments, Hewlett Packard Enterprise has teamed up with Chef Software to integrate HPE OneView’s managed infrastructure with automation recipes defined in Chef. Customers use Chef for fast, scalable, flexible automation of applications. The Chef Provisioning Driver for HPE OneView allows IT administrators to use Chef recipes to automate physical provisioning including servers, network connectivity, storage, and operating system installation as part of the application deployment process. This integration accelerates time to value and increases reliability through automated, consistent provisioning.

This white paper shows IT architects and administrators how to use software-defined templates in HPE OneView to automate provisioning of physical infrastructure in conjunction with Chef.

Introducing Composable Infrastructure

CIOs are at the forefront of a major transformation in the role technology plays in achieving competitive advantage. Driving this change are new technological approaches including mobile devices, cloud services, social networks, and Big Data analytics.

At Hewlett Packard Enterprise, we call this evolution the Idea Economy because it gives IT the ability to drive new business opportunities by quickly delivering revenue-generating products, services, and experiences. Instead of providing technology to automate internal business processes, IT can now directly impact business strategy and revenues by creating software-based services.

Enterprises currently support two very different models for delivering infrastructure. Traditionally, IT architectures have focused on minimizing risk through standard methodologies, conventional vendors, and strong governance. Supporting the Idea Economy requires IT that can continuously adapt to high levels of uncertainty and change through approaches like cloud architectures and DevOps.

Hewlett Packard Enterprise believes organizations can run traditional and new workloads on a fluid pool of hardware that can be rapidly composed and decomposed to support changing business needs. These resources can be defined in software and controlled programmatically through an open, unified API. Read the white paper “HPE Composable Infrastructure: Bridging traditional IT with the Idea Economy” to learn more.

The current state of provisioning

To create an agile and efficient IT infrastructure, organizations must transform traditional rigid physical systems into flexible physical or virtual resource pools. As they do this, they are often challenged to automate the provisioning of the physical “bare metal” hardware. Provisioning is a complex process. It requires installing and configuring hardware, connectivity, OS, and application stacks on the right resources with the right configurations. This process often requires manual interaction and monitoring, which is time consuming and error prone.

This is where the Composable Infrastructure enters the picture. In a Composable Infrastructure, fluid pools of compute, storage, and networking can be shared, connected, and reused dynamically using a software-defined approach and driven programmatically using an API. The Composable Infrastructure API, native in HPE OneView, automates the provisioning, configuration, and monitoring of the infrastructure.

The Composable Infrastructure allows IT to use templates and profiles to define the infrastructure, which now becomes infrastructure-as-code. These templates allow IT to create logical infrastructures, specific to each application’s needs. It’s the first software-defined architecture that lets IT quickly and accurately configure the entire infrastructure in one step, using one programmatic interface.

In addition to initial provisioning, change operations—such as adding additional storage to a service, modifying network connectivity, or updating firmware—can also be implemented via templates, allowing changes to be implemented automatically which significantly reduces manual interaction and errors. Continuous lifecycle operations cost less, save time, and are less disruptive, all of which help ensure real-time compliance with less effort. These templates are well suited for the DevOps approach because they eliminate time-consuming provisioning processes across operational silos that often take weeks.

HPE OneView utilizes a single, unified Composable API to provide a consistent interface for discovering, searching, inventorying, configuring, provisioning, updating, and diagnosing composable resources. Using rich, high-level infrastructure constructs, it eliminates the need for multiple, time-consuming, low-level programming interfaces so administrators and software developers can be more productive.

The use case for integrating Chef with HPE OneView

The unified API in HPE OneView provides a programmatic interface for higher-level orchestration tools. One of those tools is Chef Software, a leading configuration management tool that provides fast, scalable, flexible end-to-end automation of applications. HPE OneView and Chef provide a software-defined approach to the management of the entire data center giving IT the ability to deliver new or updated services on an as needed or on-demand basis. Together, they provide IT administrators with end-to-end infrastructure to application automation.

Chef enables rapid and reliable deployment and update of application infrastructure, using recipes that can be versioned and tested just like application software. HPE OneView brings “Infrastructure as Code” to bare-metal through templates that unify the process for provisioning compute, connectivity, storage and OS deployment in a single step, just like provisioning VM instances from the public cloud. HPE OneView acts as an infrastructure provider for Chef bringing the speed of the public cloud to internal IT processes.

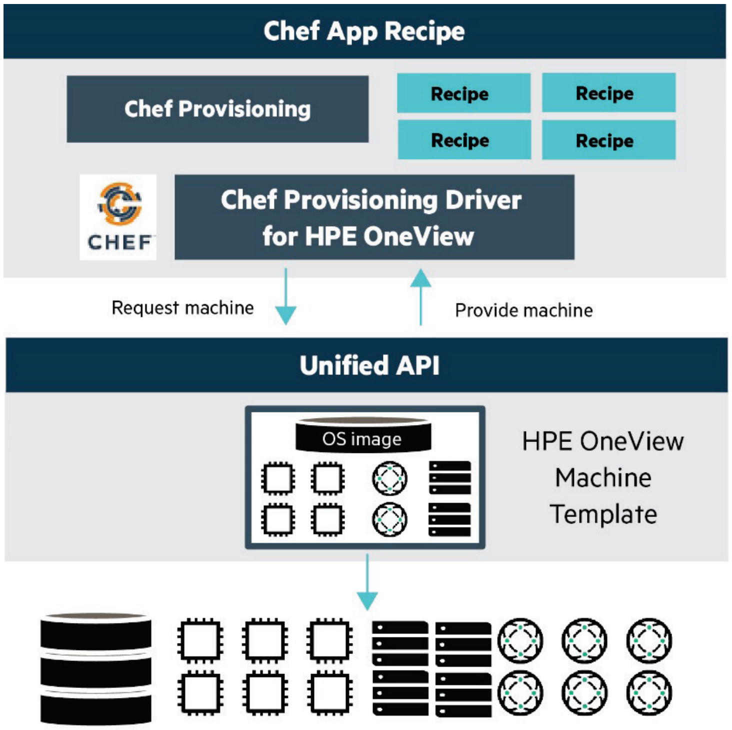
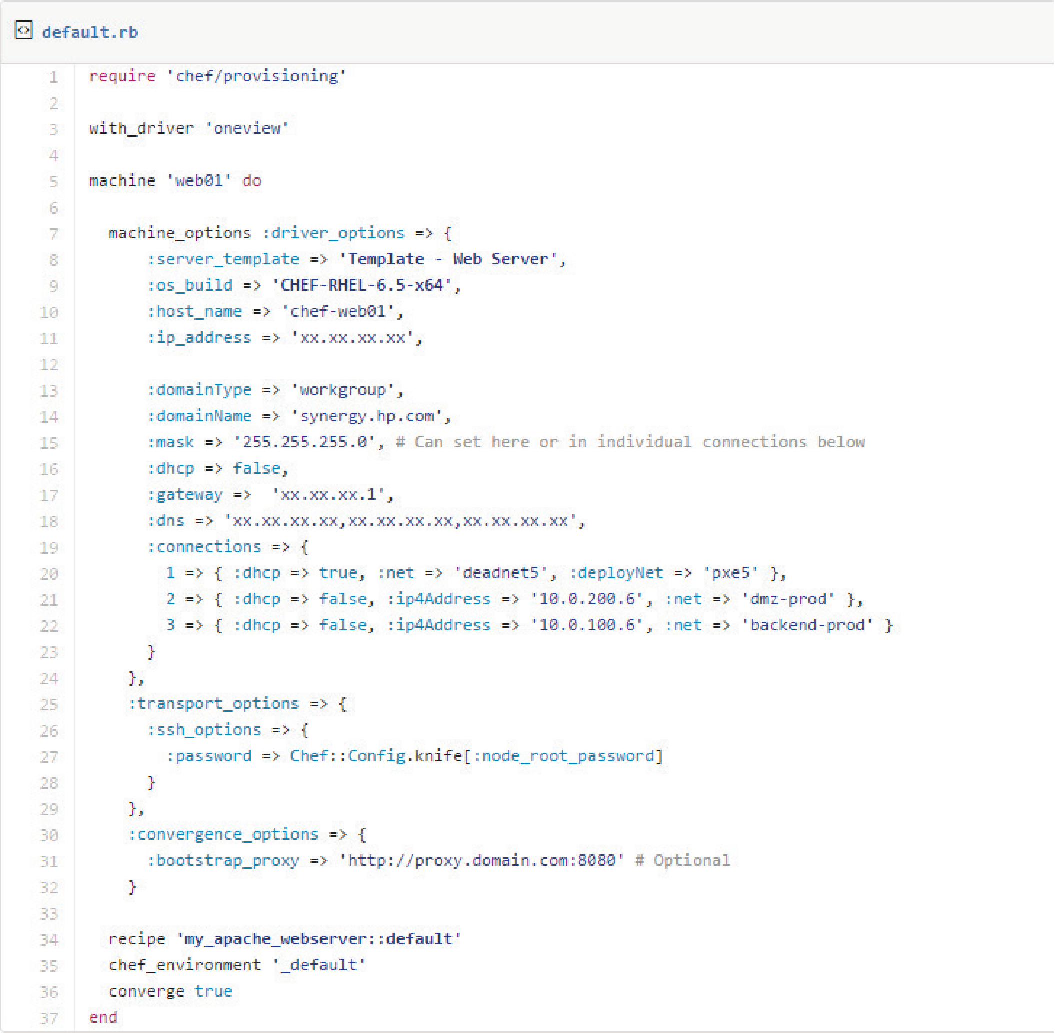


Figure 1: Overview: using HPE OneView to provision infrastructure for Chef recipes

In Chef, administrators write declarative scripts called “recipes”, that are pulled together into “cookbooks”, that can be used to automate software processes. Prior to Chef, methods existed to automate these processes, but Chef has simplified the process with its recipe approach. Chef recipes are easy to write and share among IT organizations through the Chef Supermarket.

To take advantage of the integration between Chef and HPE OneView, an administrator develops a recipe with the following steps.

1. Provision a physical server using HPE OneView template
2. Deploy operating system using an OS build template from the HPE OneView OS deployment appliance
3. Configure the application stack



default.rb

```ruby
require 'chef/provisioning'

with_driver 'oneview'

machine 'web01' do

  machine_options :driver_options => {
      :server_template => 'Template - Web Server',
      :os_build => 'CHEF-RHEL-6.5-x64',
      :host_name => 'chef-web01',
      :ip_address => 'xx.xx.xx.xx',

      :domainType => 'workgroup',
      :domainName => 'synergy.hp.com',
      :mask => '255.255.255.0', # Can set here or in individual connections below
      :dhcp => false,
      :gateway =>  'xx.xx.xx.1',
      :dns => 'xx.xx.xx.xx,xx.xx.xx.xx,xx.xx.xx.xx',
      :connections => {
        1 => { :dhcp => true, :net => 'deadnet5', :deployNet => 'pxe5' },
        2 => { :dhcp => false, :ip4Address => '10.0.200.6', :net => 'dmz-prod' },
        3 => { :dhcp => false, :ip4Address => '10.0.100.6', :net => 'backend-prod' }
      }
    },
    :transport_options => {
      :ssh_options => {
        :password => Chef::Config.knife[:node_root_password]
      }
    },
    :convergence_options => {
      :bootstrap_proxy => 'http://proxy.domain.com:8080' # Optional
    }

  recipe 'my_apache_webserver::default'
  chef_environment '_default'
  converge true
end
```

This recipe will call the Chef Provisioning Driver for HPE OneView to enable the physical provisioning and the OS builds. It will also bootstrap the Chef client onto the new system to enable further Chef operations on the application stack.

The Chef Provisioning Driver for HPE OneView is available on GitHub at github.com/HewlettPackard/chef-provisioning-OneView. The HPE Datacenter Care—Infrastructure Automation team is staffed with experts in infrastructure as code and agile processes and can assist with integrating Chef and HPE OneView in customer environments and with Chef deployments and support.

By integrating Chef with HPE OneView, Chef recipes can call HPE OneView templates for bare metal infrastructure in the same way as virtual and public cloud resources, enabling dynamic connectivity among server, storage, and networking elements to provision an application environment. IT organizations can streamline their deployment process saving time, and reducing service interruptions.

Walk through

Let's consider how we might use the Chef Provisioning Driver for HPE OneView as part of an end-to-end Chef recipe for application deployment. Using the driver, an administrator can write a recipe that specifies infrastructure parameters including which server template, which OS build plan and so on. The Chef recipe declares what you want configured in your environment, and the HPE OneView driver makes it so. No programming required. When the recipe is executed, Chef uses the Chef Provisioning Driver for HPE OneView to communicate with the HPE OneView API to build the requested server profile and execute the infrastructure provisioning process.

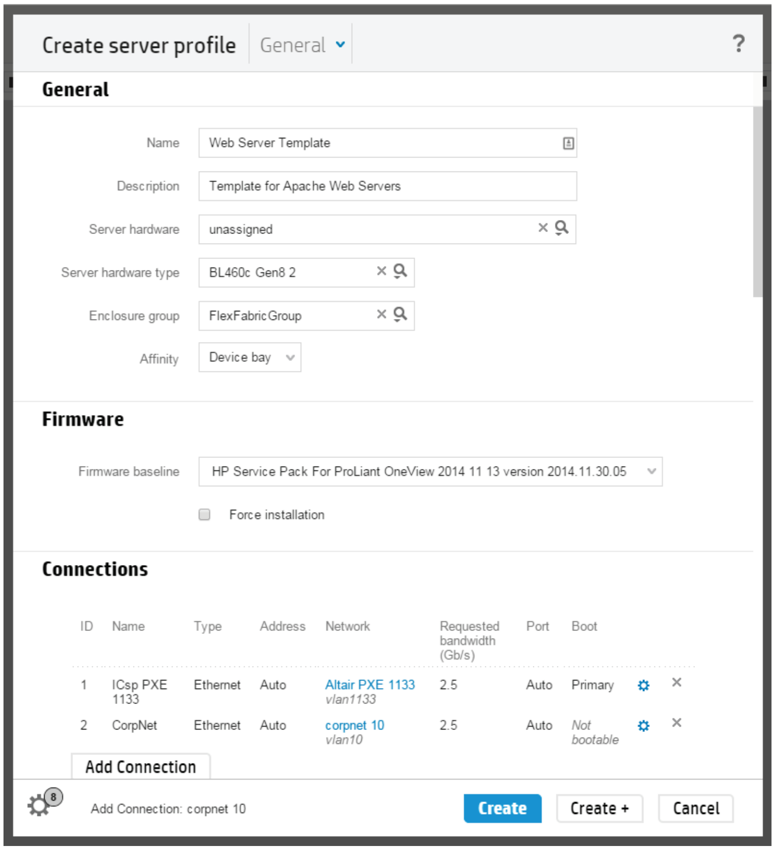
This walk-through assumes you have the following installed and running:

- An HPE OneView appliance with access to the HPE OneView UI through a browser
- Insight Control server provisioning (ICsp)
- ChefDK with access to Chef through its browser control panel

This walkthrough uses the deployment of an Apache Web server as an example, but the concepts can be applied to any Chef recipe.

Step 1: Configure the HPE OneView template

Log in to your HPE OneView appliance using your username and password and start up the HPE OneView Dashboard in your browser. Click the Server Profiles icon. We need to create a server profile for a Web server that we will be building, so click the Create Profile button on the upper left. The following popup window will appear.



General

Name: Web Server Template

Description: Template for Apache Web Servers

Server hardware: unassigned

Server hardware type: BL460c Gen8 2

Enclosure group: FlexFabric Group

Affinity: Device bay

Firmware

Firmware baseline: HP Service Pack For ProLiant OneView 2014 11 13 version 2014.11.30.05

□ Force installation

Connections

ID	Name	Type	Address	Network	Requested bandwidth (Gb/s)	Port	Boot
1	ICsp PXE 1133	Ethernet	Auto	Altair PXE 1133 *vlan1133*	2.5	Auto	Primary
2	CorpNet	Ethernet	Auto	corpnet 10 *vlan10*	2.5	Auto	*Not bootable*

Add Connection: corpnet 10

The Create Server Profile window will open. For the Name, simply enter "Web Server Template". You may supply anything you want for Description. Fill in the appropriate information for your own infrastructure, including Server Hardware, Enclosure Group, Affinity Group, and Firmware.

Now click the Create button and your new template will be saved. From here, we use Chef to do the provisioning.

Step 2: Set up the OS Build Plans

Log in to ICsp appliance with your username and password credentials. Download the script "chef-settings-Linux®" from the GitHub repository at github.com/HewlettPackard/chef-provisioning-HPE OneView. Edit this script for, providing values for these variables:

```
CHEF_PROXYSERVER: the proxy server address

YUM_REPOSITORY_INFO: the yum repository information

CHEF_CERT: the SSH certificate information to be copied into the authorization files

LOGIN_DISABLED: should be set to YES/yes to disable user login and NO/no if LOGIN using
username/password should not be disabled
```

This table below gives the various settings of the LOGIN_DISABLED and CHEF_CERT variable.

LOGIN DISABLED	CHEF_CERT	MESSAGE/action description
YES	NULL	Error message (since login disable will disable login using username/password). If the CHEF_CERT is null then access to the machine will be a problem
YES	Not null	Login using user name and password will be disabled and the key given in CHEF_CERT should be used for authentication
NO	NO	No action is performed for disabling login
NO	"YES"	No action is performed for disabling login

In the ICsp appliance create a script and save it in the appliance. It may be useful to create a naming structure that identifies the build plans for Chef such as "chef-settings-Linux" and copy paste the edited script to the script on the appliance.

Browse through the OS Build Plan list (OSBP) and select the build plan for the specific operating system that has to be installed on the target server. Save a copy of the Build plan by choosing "Save As" From "Action" drop down. Give the name "CHEF-<existing build plan name>".

Edit the OSBP. In the steps session, add steps and choose the script. This step should be right before the final "Reboot".

Step 3: Create a Chef cookbook and recipe

The Chef recipe provides the customization aspects of the server you wish to allocate. Following along with the directions in the GitHub page at github.com/HewlettPackard/chef-provisioning-HPE OneView#prerequisites, make sure your HPE OneView, ICsp, and Chef server are trusted by your certificate stores. (See ssl_issues in the GitHub project for details.) Next, follow Chef's directory structure for creating a cookbook, and start by making your knife.rb file, filling in the specific information for your environment.

```
# (knife.rb)

# (in addition to all the normal stuff like node_name, client_key, validation_client_name,
validation_key, chef_server_url, etc.)

knife[:oneview_url]        = 'https://my-oneview.my-domain.com'
knife[:oneview_username]   = 'Administrator'
knife[:oneview_password]   = 'password123'
knife[:oneview_ignore_ssl] = true # For self-signed certs

knife[:icsp_url]           = 'https://my-icsp.my-domain.com'
knife[:icsp_username]      = 'Administrator'
knife[:icsp_password]      = 'password123'
knife[:icsp_ignore_ssl]    = true # For self-signed certs

knife[:node_root_password] = 'password123'
# If your Chef server has self-signed certs:
verify_api_cert              false
ssl_verify_mode              :verify_none
```

Next, you're ready to create the recipe. Follow the example below, filling in the x's with information specific to your infrastructure, such as IP addresses, gateways, and so on. The client_name attribute and signing_key_filename attribute must have node and client creation privileges (such as validator or administrator user). Note that several items in this code are optional, and are marked with the comment #optional. The IP addresses marked optional override the properties of the same name in the structure above them. The ssl_verify_mode attribute is optional and for Chef servers that have self-signed certificates.

```
require 'chef/provisioning'

with_driver 'oneview'

with_chef_server "https://my-chef.my-domain.com/organizations/my-org",
  :client_name => Chef::Config[:node_name],
  :signing_key_filename => Chef::Config[:client_key]

machine 'web01' do
  recipe 'my_server_cookbook::default'

  machine_options :driver_options => {
      :server_template => 'Web Server Template',
      :os_build => 'CHEF-RHEL-6.5-x64',
      :host_name => 'chef-web01',
      :ip_address => 'xx.xx.xx.xx',

      :domainType => 'workgroup',
      :domainName => 'sub.domain.com',
      :mask => ‘255.255.255.0’,
      :dhcp => false,
      :gateway =>  ‘xx.xx.xx.1’,
      :dns => ‘xx.xx.xx.xx,xx.xx.xx.xx,xx.xx.xx.xx’,
      :connections => {
        #1 => { ... } (Reserved for PXE on our setup)
        2 => {
          :ip4Address => ‘xx.xx.xx.xx’,
          :mask => ‘255.255.254.0’, # Optional
          :dhcp => false
          :gateway => ‘xx.xx.xx.1’ # Optional
          :dns => ‘xx.xx.xx.xx’    # Optional
        }
      },
      :custom_attributes => {
        :chefCert => ‘ssh-rsa AA...’ # Optional
      }
    },
    :transport_options => {
      :user => ‘root’, # Optional. Defaults to ’root’
      :ssh_options => {
        :password => Chef::Config.knife[:node_root_password]
      }
    },
    :convergence_options => {
      :ssl_verify_mode => :verify_none, # Optional
      :bootstrap_proxy => ‘http://proxy.domain.com:8080’ # Optional
    }
  chef_environment ‘_default’
  converge true
end
```

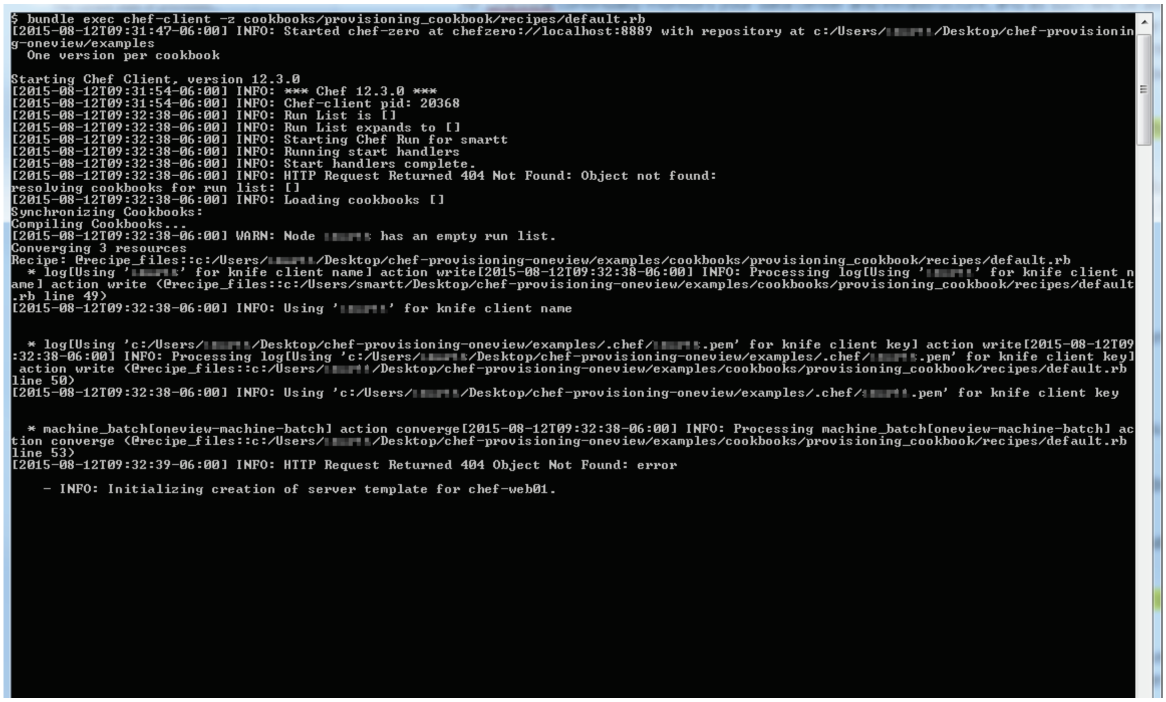
Step 4: Start the provisioning

You now have everything in place, and the provisioning can begin. Chef will use the recipe you created in conjunction with the HPE OneView driver to provision the Web server: from bare-metal hardware, to the installation and configuration of the application stack, through continual monitoring from Chef.

Open a command window. Switch to the directory containing your new Chef recipe and enter the following to execute the recipe, filling in the appropriate path to your recipe:

```
$ bundle exec chef-client -z path_to_recipe.rb
```

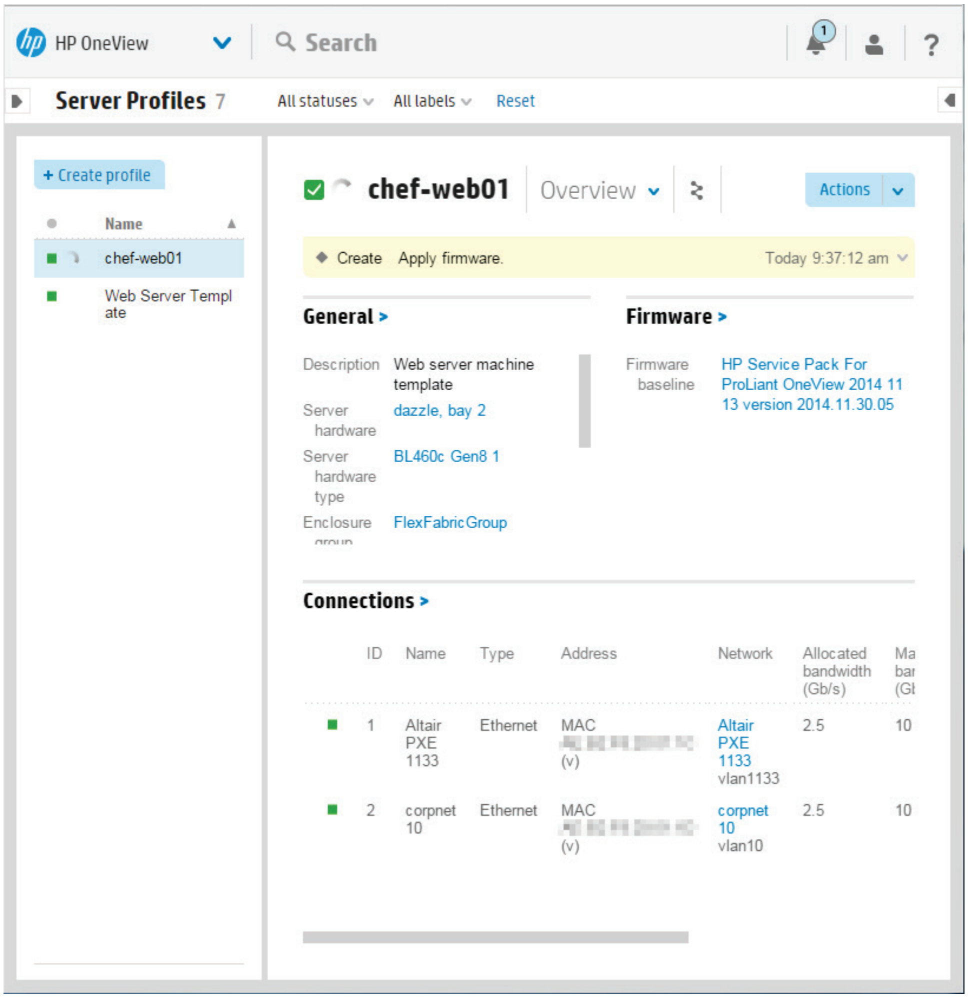
Chef will immediately begin running your recipe; the command window will look similar to the following.



```
$ bundle exec chef-client -z cookbooks/provisioning_cookbook/recipes/default.rb
[2015-08-12T09:31:47-06:00] INFO: Started chef-zero at chefzero://localhost:8889 with repository at c:/Users/[redacted]/Desktop/chef-provisionin
g-oneview/examples
  One version per cookbook

Starting Chef Client, version 12.3.0
[2015-08-12T09:31:54-06:00] INFO: *** Chef 12.3.0 ***
[2015-08-12T09:31:54-06:00] INFO: Chef-client pid: 20368
[2015-08-12T09:32:38-06:00] INFO: Run List is []
[2015-08-12T09:32:38-06:00] INFO: Run List expands to []
[2015-08-12T09:32:38-06:00] INFO: Starting Chef Run for smartt
[2015-08-12T09:32:38-06:00] INFO: Running start handlers
[2015-08-12T09:32:38-06:00] INFO: Start handlers complete.
[2015-08-12T09:32:38-06:00] INFO: HTTP Request Returned 404 Not Found: Object not found:
resolving cookbooks for run list: []
[2015-08-12T09:32:38-06:00] INFO: Loading cookbooks []
Synchronizing Cookbooks:
Compiling Cookbooks...
[2015-08-12T09:32:38-06:00] WARN: Node [redacted] has an empty run list.
Converging 3 resources
Recipe: @recipe_files::c:/Users/[redacted]/Desktop/chef-provisioning-oneview/examples/cookbooks/provisioning_cookbook/recipes/default.rb
  * log[Using '[redacted]' for knife client name] action write[2015-08-12T09:32:38-06:00] INFO: Processing log[Using '[redacted]' for knife client n
ame] action write (@recipe_files::c:/Users/smartt/Desktop/chef-provisioning-oneview/examples/cookbooks/provisioning_cookbook/recipes/default
.rb line 49)
[2015-08-12T09:32:38-06:00] INFO: Using '[redacted]' for knife client name


  * log[Using 'c:/Users/[redacted]/Desktop/chef-provisioning-oneview/examples/.chef/[redacted].pem' for knife client key] action write[2015-08-12T09
:32:38-06:00] INFO: Processing log[Using 'c:/Users/[redacted]/Desktop/chef-provisioning-oneview/examples/.chef/[redacted].pem' for knife client key]
 action write (@recipe_files::c:/Users/[redacted]/Desktop/chef-provisioning-oneview/examples/cookbooks/provisioning_cookbook/recipes/default.rb
line 50)
[2015-08-12T09:32:38-06:00] INFO: Using 'c:/Users/[redacted]/Desktop/chef-provisioning-oneview/examples/.chef/[redacted].pem' for knife client key


  * machine_batch[oneview-machine-batch] action converge[2015-08-12T09:32:38-06:00] INFO: Processing machine_batch[oneview-machine-batch] ac
tion converge (@recipe_files::c:/Users/[redacted]/Desktop/chef-provisioning-oneview/examples/cookbooks/provisioning_cookbook/recipes/default.rb
line 53)
[2015-08-12T09:32:39-06:00] INFO: HTTP Request Returned 404 Object Not Found: error

    - INFO: Initializing creation of server template for chef-web01.
```

Step 5: Monitor the progress

HPE OneView continually monitors your data center and infrastructure, so it’s aware of what is happening as the provisioning is taking place. If you wish to see what’s happening, open the HPE OneView Dashboard in your Web browser, and return to the Server Profiles window. You will see the new server’s name on the left. Click on it. At the top of the right-hand pane, you will see a yellow box showing the steps that are taking place.



◆ Create Apply firmware. Today 9:37:12 am

General >

Description	Web server machine template
Server hardware	dazzle, bay 2
Server hardware type	BL460c Gen8 1
Enclosure group	FlexFabricGroup

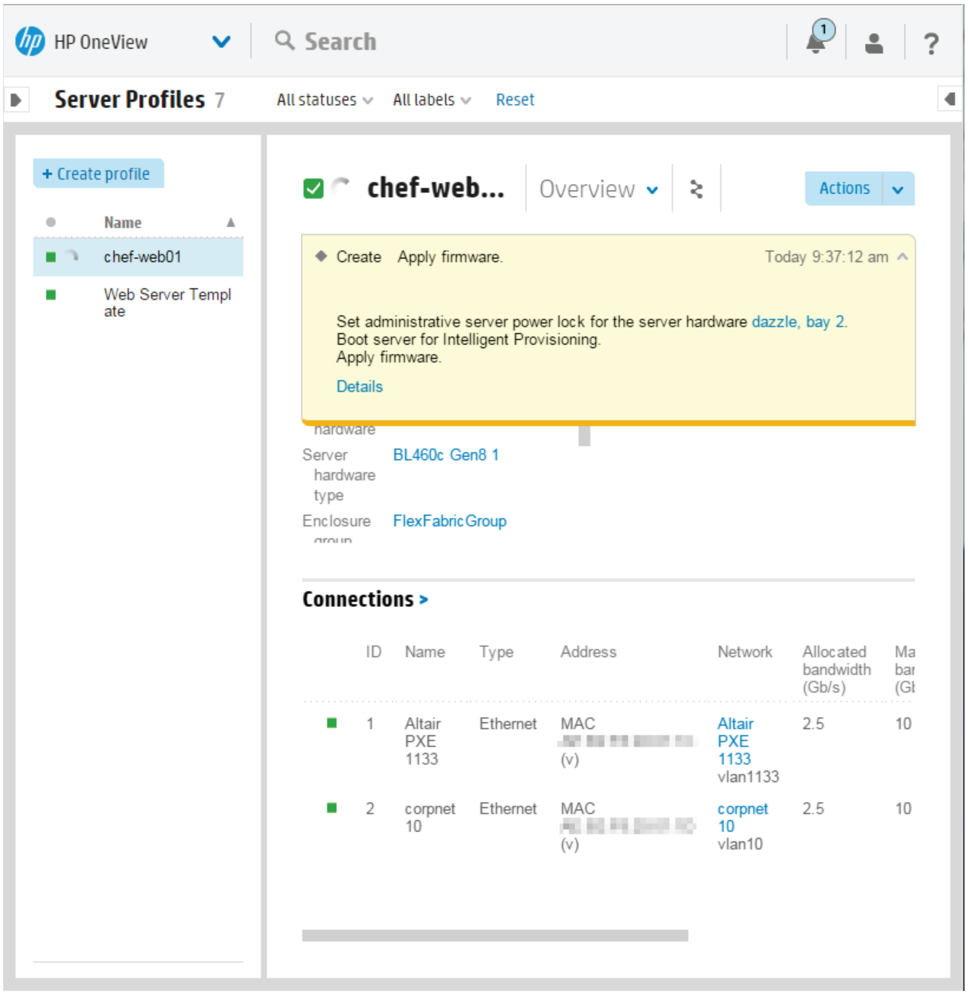
Firmware >

Firmware baseline	HP Service Pack For ProLiant OneView 2014 11 13 version 2014.11.30.05

Connections >

	ID	Name	Type	Address	Network	Allocated bandwidth (Gb/s)	Ma bar (Gb
🟩	1	Altair PXE 1133	Ethernet	MAC [redacted] (v)	Altair PXE 1133 vlan1133	2.5	10
🟩	2	corpnet 10	Ethernet	MAC [redacted] (v)	corpnet 10 vlan10	2.5	10

As the provisioning occurs, the yellow box will update in real-time. Click on the yellow box during this time, and it will expand and show you more detail of what is happening with each step, similar to the following screen.



Create Apply firmware. Today 9:37:12 am

Set administrative server power lock for the server hardware dazzle, bay 2.
Boot server for Intelligent Provisioning.
Apply firmware.

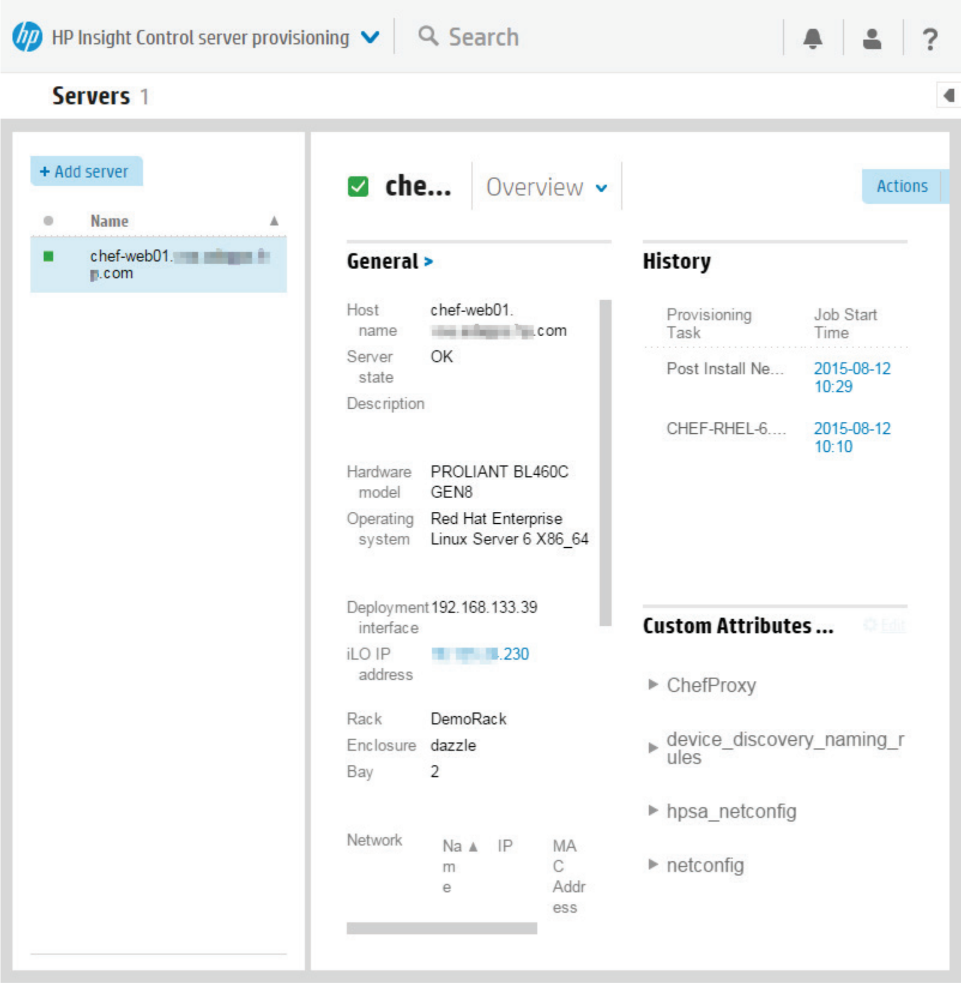
Details

Server hardware type	BL460c Gen8 1
Enclosure group	FlexFabricGroup

Connections >

	ID	Name	Type	Address	Network	Allocated bandwidth (Gb/s)	Ma bar (Gb
🟩	1	Altair PXE 1133	Ethernet	MAC [redacted] (v)	Altair PXE 1133 vlan1133	2.5	10
🟩	2	corpnet 10	Ethernet	MAC [redacted] (v)	corpnet 10 vlan10	2.5	10

When the physical configuration is complete, go over to your Insight Control server provisioning dashboard to see that everything is configured, as in the following image.



chef-web01.[redacted]p.com

General >

Host name: chef-web01.[redacted].com
Server state: OK
Description:
Hardware model: PROLIANT BL460C GEN8
Operating system: Red Hat Enterprise Linux Server 6 X86_64
Deployment interface: 192.168.133.39
iLO IP address: [redacted].230
Rack: DemoRack
Enclosure: dazzle
Bay: 2
Network: Name ▲ | IP | MAC Address

History

Provisioning Task	Job Start Time
Post Install Ne...	2015-08-12 10:29
CHEF-RHEL-6....	2015-08-12 10:10

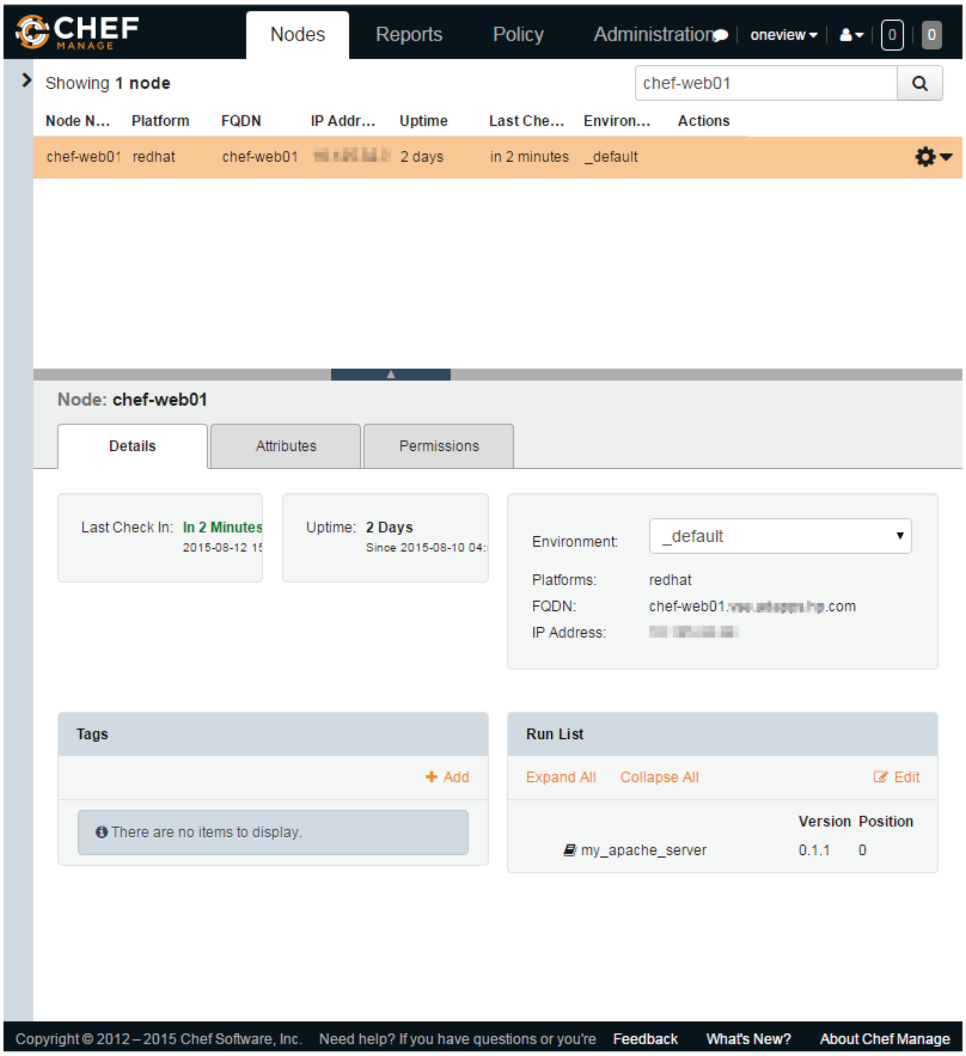
Custom Attributes ...

- ▶ ChefProxy
- ▶ device_discovery_naming_rules
- ▶ hpsa_netconfig
- ▶ netconfig

There you can see, for example, the IP addresses and other network configurations that you specified in your Chef recipe.

Over on the right-hand side of the screen you can see the history for this new server, including the point where it installed the operating system and the post-install networking configuration.

Now if you switch over to the Chef dashboard, you can see that the node now exists and that it is configured using the specifications from the Chef recipe, as in the following image.



Showing **1 node**

chef-web01

Node N...	Platform	FQDN	IP Addr...	Uptime	Last Che...	Environ...	Actions
chef-web01	redhat	chef-web01	[redacted]	2 days	in 2 minutes	_default	

Node: **chef-web01**

Last Check In: **In 2 Minutes** 2015-08-12 15

Uptime: **2 Days** Since 2015-08-10 04:

Environment: _default

Platforms: redhat

FQDN: chef-web01.[redacted].com

IP Address: [redacted]

Tags

There are no items to display.

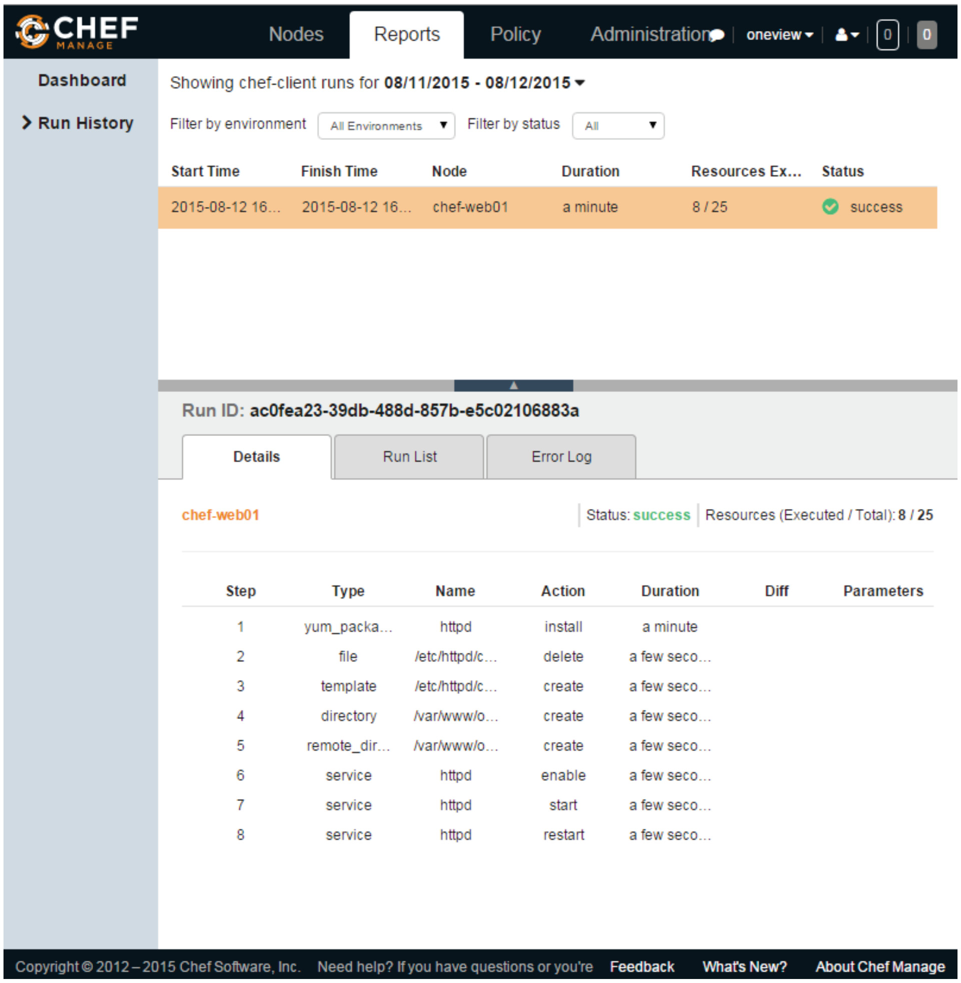
Run List

	Version	Position
my_apache_server	0.1.1	0

Copyright © 2012 – 2015 Chef Software, Inc. Need help? If you have questions or you're

Included in the Chef configuration will be a serial number that Chef uses to identify this server. Included in this you will see that the apache configurations also took place, and that the apache Web server is now running on this server.

The Chef dashboard also includes a report section whereby you can see the run history for this recipe, including the time it took and the steps that were performed, as shown below.



Showing chef-client runs for **08/11/2015 - 08/12/2015**

Filter by environment: All Environments Filter by status: All

Start Time	Finish Time	Node	Duration	Resources Ex...	Status
2015-08-12 16...	2015-08-12 16...	chef-web01	a minute	8 / 25	✓ success

Run ID: **ac0fea23-39db-488d-857b-e5c02106883a**

chef-web01 | Status: success | Resources (Executed / Total): **8 / 25**

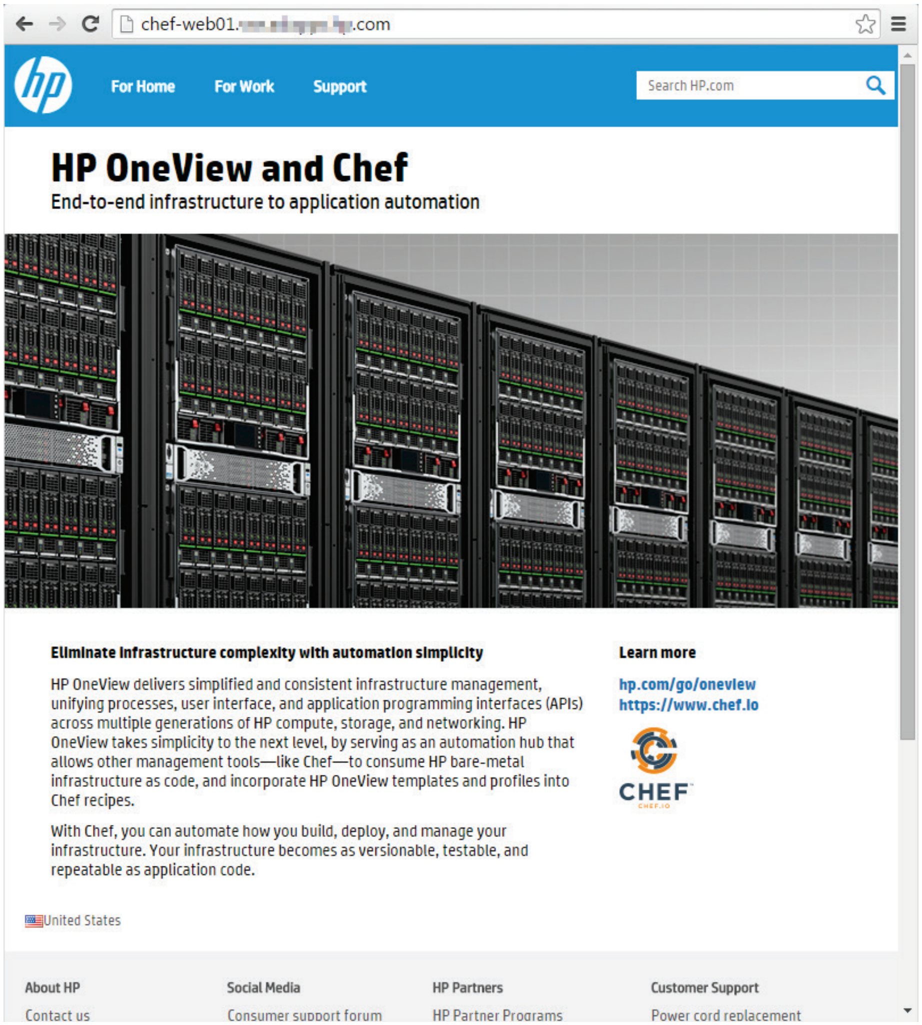
Step	Type	Name	Action	Duration	Diff	Parameters
1	yum_packa...	httpd	install	a minute		
2	file	/etc/httpd/c...	delete	a few seco...		
3	template	/etc/httpd/c...	create	a few seco...		
4	directory	/var/www/o...	create	a few seco...		
5	remote_dir...	/var/www/o...	create	a few seco...		
6	service	httpd	enable	a few seco...		
7	service	httpd	start	a few seco...		
8	service	httpd	restart	a few seco...		

Copyright © 2012 – 2015 Chef Software, Inc. Need help? If you have questions or you're

Step 6: Verifying the provisioning

Now is the time to test that the server is usable. We know from both HPE OneView and Chef that the server is alive and configured correctly with the networking we specified, and that the OS and apache were installed and configured. The fact that these servers are showing up in HPE OneView and Chef shows that there is network connectivity and that the server is reporting back. Now let's see the application work.

Open up a new window in your Web browser. Enter the name of the server in the address bar, and you see the "It works" welcome page from apache, as shown below.



chef-web01.[redacted].com

HP OneView and Chef

End-to-end infrastructure to application automation

Eliminate infrastructure complexity with automation simplicity

HP OneView delivers simplified and consistent infrastructure management, unifying processes, user interface, and application programming interfaces (APIs) across multiple generations of HP compute, storage, and networking. HP OneView takes simplicity to the next level, by serving as an automation hub that allows other management tools—like Chef—to consume HP bare-metal infrastructure as code, and incorporate HP OneView templates and profiles into Chef recipes.

With Chef, you can automate how you build, deploy, and manage your infrastructure. Your infrastructure becomes as versionable, testable, and repeatable as application code.

Learn more

hp.com/go/oneview
https://www.chef.io

Your provisioned server is ready and running with the application stack you chose. You can now continue managing and configuring it with your Chef recipes.

Understand the provisioning

The HPE OneView driver performs the following 5 steps during the provisioning process:

1. **Locates and allocates compatible server hardware** from the available pool of servers managed my HPE OneView, using the HPE OneView search REST API.
2. **Creates new instances from the selected Server Profile Template** using the HPE OneView REST API:
   a. **Firmware, BIOS, and local storage configuration:** HPE OneView uses the embedded HPE Intelligent Provisioning (IP), a Linux pre-boot environment embedded on the iLO baseboard management card (BMC) processor, and the embedded SmartStart tooling.
   b. **Network connectivity:** HPE OneView uses the virtual connect networking platform to dynamically provisioning network connectivity for servers, from a pool of network connections.
   c. **SAN storage:** HPE OneView automates SAN storage provisioning for HPE 3PAR SAN storage systems and Fibre Channel switches including Brocade, Cisco, and HPE Networking.
3. **Deploys the selected OS** once the hardware provisioning is complete:
   a. The OS deployment network connection in the server profile is configured to PXE boot to the OS deployment server.
   b. The server hardware is powered on and PXE boots.
   c. The server registers with the OS deployment server.
   d. The OneView driver detects the server is registered and schedules the OS installation using the HPE OneView REST API for OS deployment (Insight Control server provisioning).
4. **Configures the host networking**
   a. IP address, subnet, hostname, etc. for different network adapters and virtual network adapters, allowing you to build sophisticated private networking arrangements.
5. **Chef deploys the application stack on top of the provisioned server OS**
   a. Bootstraps the Chef client onto the new server, first downloading the Chef client software and then registering itself with the Chef server.
   b. Start a Chef-client run on the new server, which will pull the necessary cookbook from the Chef server and apply the recipes to configure the Web server.

Best practices

A robust, automated data center takes time to organize, build, maintain, and support. Here are some best practices you should consider for your environment.

Divide your work

When creating Chef recipes, don't let them get very big; instead, maintain small recipes that serve specific purposes. In doing so, consider the likelihood that the recipes you're creating will need to be used again. If you're doing a large amount of provisioning, it's very likely you will need to reuse them. If you want, you can even share them with others.

Don't hardcode

This means you don't want to have to open each recipe and modify it each time you use it and fill in the individual parameters such as IP addresses and domain names. Doing so is called hardcoding your parameters, and software professionals advise against it.

Instead, code your Chef recipes to be generic, and put your parameters into a data bag, environment definition, or an attribute file. Then, when running the recipe, you can simply make changes to one of these resources, and all the recipes in your set will use those parameters.

Don't reinvent the wheel

Chef has an enormous number of features. It was built from the ground up specifically for application and infrastructure automation, ranging from individual servers to huge, globally dispersed data centers. No matter what type of work you're trying to accomplish, Chef can probably handle it, either natively or through simple scripts that use the native functionality.

In other words, make sure you're familiar with all aspects of Chef and what it provides natively. Study the documentation and learn what Chef is capable of. You don't want to try to write a cookbook that duplicates the capabilities of built-in Chef features or does the same thing as an existing cookbook on Chef's public Supermarket (supermarket.chef.io).

The Chef Supermarket is filled with thousands of cookbooks built by avid Chef users. It provides useful searching features to narrow results by platform (such as Windows®, Red Hat,® or Ubuntu), and is a great place to see who the authors are, how many times the cookbook has been downloaded, and often provides links to the source code so you can go inspect it for yourself. Note that cookbooks are not compiled, so you should always examine the code before using it and deploying it. If the code is hosted publicly on sites like GitHub, you should also check the reported issues and look at the recent activity to see if the cookbook is being actively maintained and issues are being addressed.

Provisioning and de-provisioning through recipes

Your templates and profiles work together with Chef recipes for provisioning and for ongoing management and configuration. Remember, however, that shutting down a server to be returned to the pool is not just a matter of turning it off. Instead, you'll want to build a recipe that does the de-provisioning as well.

Leverage HPE expertise to accelerate your Chef deployments today!

Data center technology is evolving and making use of the models, and learning the new models can be overwhelming for teams. It's hard for any one individual to be on top of all the newest and important technologies. Hewlett Packard Enterprise has decades of experience and our team of devoted developers and engineers know the best practices in creating and managing an automated infrastructure. It is our pleasure to share these best practices with you. We've only listed a few here to help you get started, and would be happy to talk to you about how you can learn more and how we can help. You can find more information about how Hewlett Packard Enterprise can help by visiting the HPE Datacenter Care—Infrastructure Automation website at www8.hp.com/us/en/business-services/it-services/datacenter-infrastructure-automation.html.

Summary

The data center is evolving and IT administrators are finding new ways to simplify the management of large numbers of servers and hardware. Hewlett Packard Enterprise has always been an innovator in the data center infrastructure industry. A major transformation is taking place as data centers are becoming automated with the goal of saving resources and maximizing both profits and customer experience. Hewlett Packard Enterprise is at the forefront and is part of the evolution we are calling the Idea Economy.

Today's enterprises are finding difficulty in mixing old-style data centers with new Idea Economy applications. Consumers and customers want results quickly; they want continuous software delivery, and consistent and continuous connectivity with your data center. Is your data center ready?

Composable Infrastructure streamlines the automation of complex data centers, taking your infrastructure into this new world and ahead of your competition. HPE OneView, together with Chef, streamlines the automation of your infrastructure and prepares you to succeed in the Idea Economy.

For information on HPE OneView integrations with other software, visit: hpe.com/info/composablepartners.

For more information on HPE Datacenter Care—Infrastructure Automation, visit: hpe.com/services/dcia.

Hewlett Packard Enterprise is proud to support the open source community, and has made our Chef recipes available on GitHub at github.com/HewlettPackard/chef-provisioning-oneview. We have built a community around our recipes, and encourage other developers to fork them, modify them, add new features, and then perform a pull request so we can add the features back in.

Learn more at

hpe.com/info/composable
hpe.com/info/composableprogram
chef.io/

Sign up for updates

★ Rate this document

Hewlett Packard
Enterprise

4AA6-1024ENW, January 2016, Rev. 1