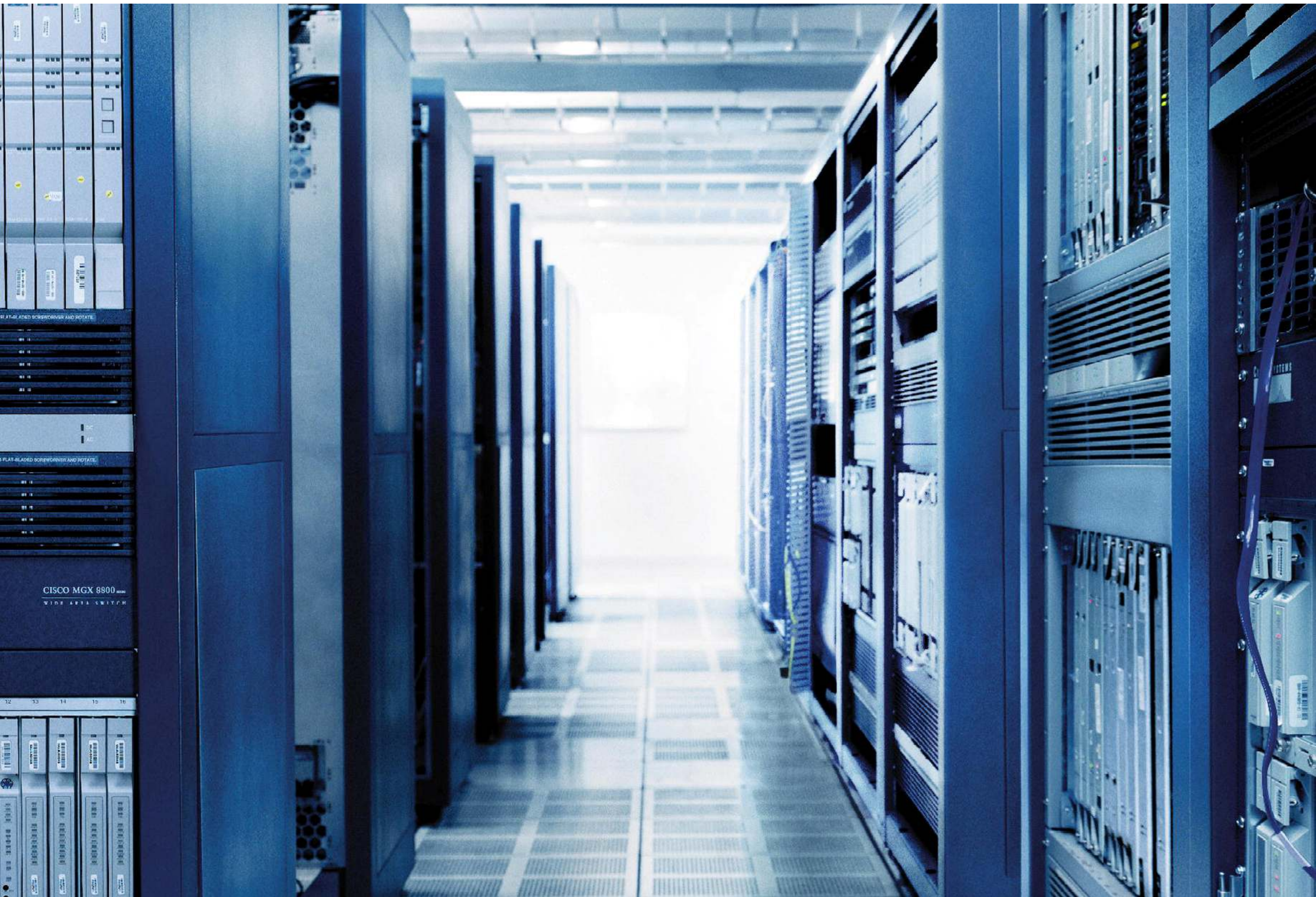
Using the Chef Client with Cisco Nexus Devices

Installation Guide



Introduction **3**

Scope and Requirements 3

Chef Client Setup on Cisco Nexus Devices **3**

Client Configuration Prerequisites 3
- Allow SSL Communication with Chef Client 3
- Configure Accurate Network Time Protocol Settings 3
- Copy Chef Client to the Device 4
- Copy Chef Validation File to Device 4

Client Installation and Activation 4
- Install Chef Client on Cisco NX-OS 4
- Activate Chef on Cisco NX-OS 5

Client Configuration 5
- Enable VTY Service 5
- Configure Domain Name Service Settings for Chef Client 6
- Configure Chef Client 6

Chef Client Registration with the Server 6
- Chef Client Run Process 6
- Run the Client on the Cisco Nexus Device 7

Configuration on Chef Server **8**

Configure Recipe on Server 8
Configure Run List for Node 9

Device Patching Test Case **10**

Software Defect Overview 10
- Bug ID 10
- Defect Background 10
- Run Chef Client with Patching Recipe 11
- Verify the Run in the Logs 11

Appendix A. Chef Concepts **14**

Chef Overview 14
Definitions 14
Flow of a Chef Run 16

For More Information **16**

Introduction

To fix problems in software, the traditional approach is to upgrade the complete software installation through either a boot variable change process or In Service Software Upgrade (ISSU).Cisco® NX-OS Software, however, provides the capability to fix the specific software defect by installing a patch, without the need for any software upgrade through ISSU or disruption of traffic.

With a Chef client, you can automate:

- Patching and depatching of a device software
- Configuration management
- Device statistics collection

This document describes the installation and configuration of a Chef client on a Cisco Nexus® device. It focuses on the use of the Chef client to install a patch to fix a software defect on Cisco Nexus 9000 Series Switches.

Scope and Requirements

This document describes how to install and run a Chef client for device patching for the following devices:

- Cisco Nexus 3000 Series Switches
- Cisco Nexus 9000 Series Switches

Chef Client Setup on Cisco Nexus Devices

Client Configuration Prerequisites

This section provides the prerequisite configuration for the Chef client setup on Cisco Nexus devices.

Allow SSL Communication with Chef Client

The Chef server communicates with the client using SSL on port 443. The management interface of the device may drop all SSL communications depending on the platform. The Cisco Nexus3000 Series device requires additional configuration to allow SSL communication on port 443 (Figure 1).

This behavior can be allowed in either of two ways:

- Override the default SSL port setting for the Chef server.
- Remove the default configured policy for the Cisco Nexus 3000 Series management interface.

Figure 1. Allow SSL Communication with Chef Client for Cisco Nexus3000 Series

```
no mgmt-policy MGMT_DEF_POLICY106
```

Note: The configuration in Figure 1 is for Cisco Nexus3000 SeriesSwitches only.

Configure Accurate Network Time Protocol Settings

The device needs to be configured with accurate Network Time Protocol (NTP) settings (Figure 2). This configuration is required for troubleshooting and log analysis purposes. It is highly recommended that you synchronize the devices and the Chef environment with NTP for consistent timestamps.

Figure 2. Configure NTP on Cisco Nexus Device

```
ntp server 10.81.254.202 use-vrf management
```

Copy Chef Client to the Device

You need to copy the Chef client OVA file to the Cisco Nexus device to proceed with the installation (Figure 3).

Figure 3. Copy Chef Client to Cisco Nexus Device

```
copy scp://username@server/path/n9k_chef.ova bootflash: vrf management
```

Copy Chef Validation File to Device

The Chef validation file is required to register the Chef client with the server (Figures 4 and 5). This registration is required to authenticate the session between the client and the server. The chef validation file is called chef-validator.pem by default. However, a new file can be created on the server and used for the client.

Figure 4. Copy Chef Validation File to Cisco Nexus Device

```
copy scp://username@server/path/chef-validator.pem bootflash: vrf management
```

Figure 5. Copy Chef Validation File to Chef Client

```
copy bootflash:chef-validator.pem bootflash: virtual-instance/chef/rootfs/tmp/ssl
```

Client Installation and Activation

This section provides the steps needed to install and activate the Chef client on the Cisco Nexus device.

Install Chef Client on Cisco NX-OS

Use the OVA file copied to the device bootflash to install the client on the Cisco Nexus device (Figure 6). Then verify the installation (Figure 7).

Figure 6. Install Chef Client on Cisco Nexus Device

```
N9K-INS# virtual-service install name chef package  bootflash:n9k_chef.ova


Note: Installing package 'bootflash:/n9k_chef.ova' for virtual service 'chef'.
Once the install has finished, the VM may be activated. Use 'show virtual-service
list' for progress.


2013 Sep 25 00:30:22 N9K-INS %$ VDC-1 %$ %VMAN-2-INSTALL_STATE: Successfully
installed virtual service 'chef'
```

Figure 7. Verify Chef Installation on Cisco Nexus Device

```
N9K-INS# show virtual-service list


Virtual Service List:


Name                    Status            Package Name
```

```
-------------------------------------------------------------------
chef                        Installed          n9k_chef.ova
```

Activate Chef on Cisco NX-OS

After you successfully install the client, you need to activate it before starting the configuration to register it with the Chef server (Figures 8 and 9).

Figure 8. Activate Chef Service

```
N9K-INS(config)# virtual-service chef
N9K-INS(config-virt-serv)# activate

Note: Activating virtual-service 'chef', this might take a few minutes. Use 'show
virtual-service list' for progress.

N9K-INS(config-virt-serv)#

2013 Sep 25 00:41:22 N9K-INS %$ VDC-1 %$ %VMAN-2-ACTIVATION_STATE: Successfully
activated virtual service 'chef'
```

Figure 9. Verify Chef Service Activation on Cisco NX-OS

```
N9K-INS# show virtual-service list

Virtual Service List:

Name                        Status             Package Name
-------------------------------------------------------------------
chef                        Activated          n9k_chef.ova
```

Client Configuration

Enable VTY Service

Enable the VTY service (Figure 10).

Figure 10. Enable VTY Service

```
onep
  transport type tcp
  service set vty
```

Configure Domain Name Service Settings for Chef Client

Domain Name Service (DNS) configuration is highly recommended on the Cisco Nexus device for the Chef client. It is required for name resolution for Chef server and node objects (Figure 11).

Figure 11. DNS Configuration for Chef Client

```
onep applications chef
  chef v0.8
   vrf management
   name-server 10.123.123.1
   domain-name cisco.com
```

Configure Chef Client

The chef client parameters need to be configured for registration with the server (Figure 12). The parameters configured in this step are:

- Chef server fully qualified domain name (FQDN)
- Chef server port number
- Node name for the client on the Cisco Nexusdevice

Figure 12. Configure Chef Client

```
onep applications chef
  chef v0.8
    server chef-server-ins-f.cisco.com port 443
    vrf management
    validation-client-name chef-validator
    interval 60
    node-name N9K-INS
```

Chef Client Registration with the Server

This section provides a high-level overview of the Chef client run process and the required steps on the Cisco Nexus device for registration with the server.

Chef Client Run Process

The chef client can be run in any of three ways:

- Daemon: The client runs and processes the runlist at a configured interval.
- One-shot mode: The client runs and processes the runlist one time.
- Why-run mode:Use this mode to check what would be processed if the client were actually run.

Run the Client on the Cisco Nexus Device

The Chef client will be run in one-shot mode for the patching use case presented here. The client run will be performed after initial configuration to register the client with the server (Figure 13). The node will be registered on the server after the first client run (Figure 14).

Figure 13. Client Processing on Cisco Nexus Device

```
execute onep application chef v0.8 chef client-oneshot
```

Figure 14. Client Registration on Chef Server

```
TME-1-9508-2# show onep app chef v0.8 chef client last-exec-log

# Logfile created on 2013-09-30 12:18:11 +0000 by logger.rb/31641
[2013-09-30T12:18:11+00:00] INFO: Forking chef instance to converge...
[2013-09-30T12:18:11+00:00] INFO: Fork successful. Waiting for new chef pid:
15601
[2013-09-30T12:18:11+00:00] INFO: Forked instance now converging
[2013-09-30T12:18:11+00:00] INFO: *** Chef 11.4.0 ***
[2013-09-30T12:18:12+00:00] WARN: unable to detect ipaddress
[2013-09-30T12:18:12+00:00] WARN: unable to detect macaddress
[2013-09-30T12:18:12+00:00] WARN: unable to detect ip6address
[2013-09-30T12:18:12+00:00] INFO: Run List is []
[2013-09-30T12:18:12+00:00] INFO: Run List expands to []
[2013-09-30T12:18:12+00:00] INFO: Starting Chef Run for TME-1-9508-2
[2013-09-30T12:18:12+00:00] INFO: Running start handlers
[2013-09-30T12:18:12+00:00] INFO: Start handlers complete.
[2013-09-30T12:18:12+00:00] INFO: Loading cookbooks []
[2013-09-30T12:18:12+00:00] WARN: Node TME-1-9508-2 has an empty run list.
[2013-09-30T12:18:12+00:00] INFO: Chef Run complete in 0.228609717 seconds
[2013-09-30T12:18:12+00:00] INFO: Running report handlers
[2013-09-30T12:18:12+00:00] INFO: Creating Cisco JSON run report
[2013-09-30T12:18:12+00:00] INFO: Report handlers complete
[2013-09-30T12:18:12+00:00] INFO: Forked child successfully reaped (pid: 15601)
```

Configuration on Chef Server

Configure Recipe on Server

The recipe relevant to the action to be performed on the device is configured on the Chef workstation and uploaded to the server. This section provides an extract from the recipe created to install the patch on the Cisco Nexus device (Figure 15).

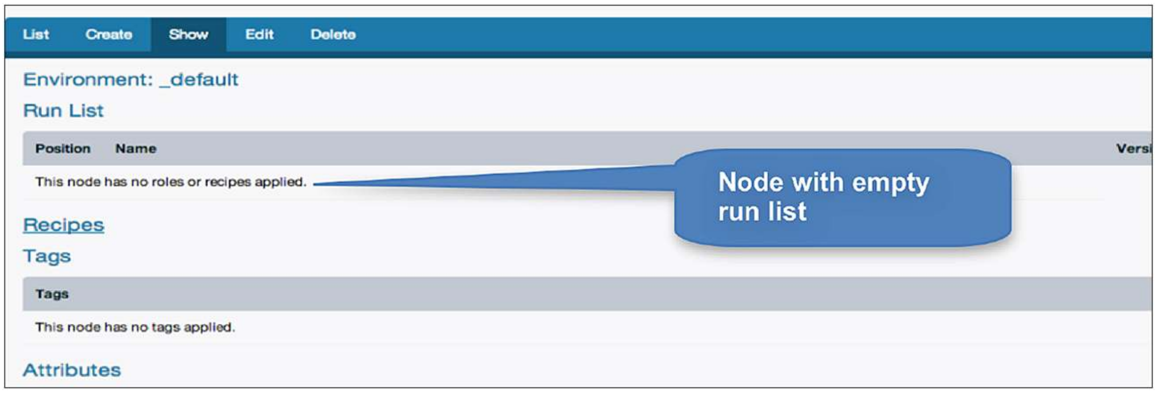
Figure 15. Chef Recipe for Patching Cisco Nexus9000 Series Device

```
cisco_device "#{name}" do
  username "admin"
  password "xxx"
  action :create
end

..
..
..
..
..

cisco_package "TME-1-9508-2/n9000_CSCuj11591.gbin" do
  source "bootflash:///n9000_CSCuj11591.gbin"
  action :activate
end

..
..
..

cisco_device "TME-1-9508-2" do
  action :destroy
end
```

Configure Run List for Node

After the node is registered on the Chef server, the run list can be configured on the server with the relevant recipes that will run on the node every time the client is run (Figures 16 and 17).

Figure 16. Empty Run List for New Node



Environment: _default

Run List

Position	Name	Versi
This node has no roles or recipes applied.		

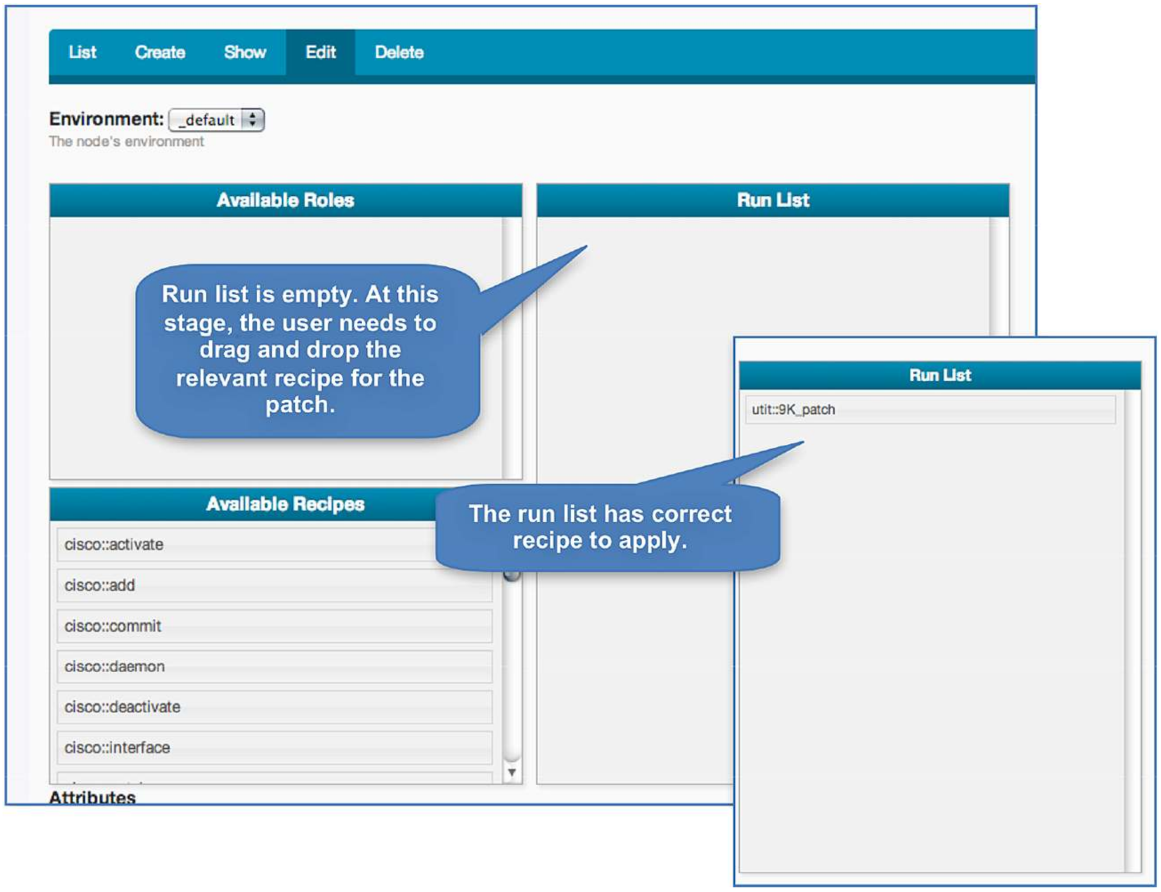
Recipes

Tags

Tags
This node has no tags applied.

Attributes

Figure 17. Add Recipe to Node Run List


List
Create
Show
Edit
Delete
Available Roles
Run List
Run list is empty. At this stage, the user needs to drag and drop the relevant recipe for the patch.
Available Recipes
The run list has correct recipe to apply.
Attributes

Environment: _default

The node's environment

Available Recipes: cisco::activate, cisco::add, cisco::commit, cisco::daemon, cisco::deactivate, cisco::interface

Run List: utit::9K_patch

Device Patching Test Case

Software Defect Overview

Bug ID

CSCuj11591 FD leak in securityd for security_nginx.out:feature nginx enable/disable.

Defect Background

Feature nginx is used to enable HTTP access to the Cisco Nexus9000 SeriesSwitch through the interactive API (iAPI). The iAPI will not be able to communicate with the Cisco Nexus9000 SeriesSwitch if the nginx feature is not enabled in Cisco NX-OS.

Figure 18 shows the procedure to enable the nginx feature.

Figure 18. Enable nginx Feature

```
N9K-INS(config)# feature nginx
N9K-INS(config)# end
```

On every run of nginxfeature activation and deactivation, file descriptor (FD) leaks in the security ID (security) were detected (Figures 19 and 20).

Figure 19. Enable and Disable nginx Feature Defects

```
N9K-INS(config)# no feature nginx
N9K-INS(config)# feature nginx
N9K-INS(config)# no feature nginx
N9K-INS(config)# no feature nginx
N9K-INS(config)# feature nginx
N9K-INS(config)# no feature nginx
```

Figure 20. Memory Leaks in Security as a Result of Software Defect

```
COMMAND    PID   USER   FD   TYPE   DEVICE SIZE/OFF   NODE NAME
securityd 8203   root   16r   REG   0,20   10 37760 /var/tmp/security_nginx.out
(deleted)
securityd 8203   root   17r   REG   0,20   10 66744 /var/tmp/security_nginx.out
(deleted)
securityd 8203   root   18r   REG   0,20    5 37863 /var/tmp/security_nginx.out
(deleted)
securityd 8203   root   19r   REG   0,20    5 65969 /var/tmp/security_nginx.out
(deleted)
nginx     8301   root   16r   REG   0,20   10 37760 /var/tmp/security_nginx.out
(deleted)
nginx     8301   root   17r   REG   0,20   10 66744 /var/tmp/security_nginx.out
(deleted)
nginx     8301   root   18r   REG   0,20    5 37863 /var/tmp/security_nginx.out
(deleted)
```

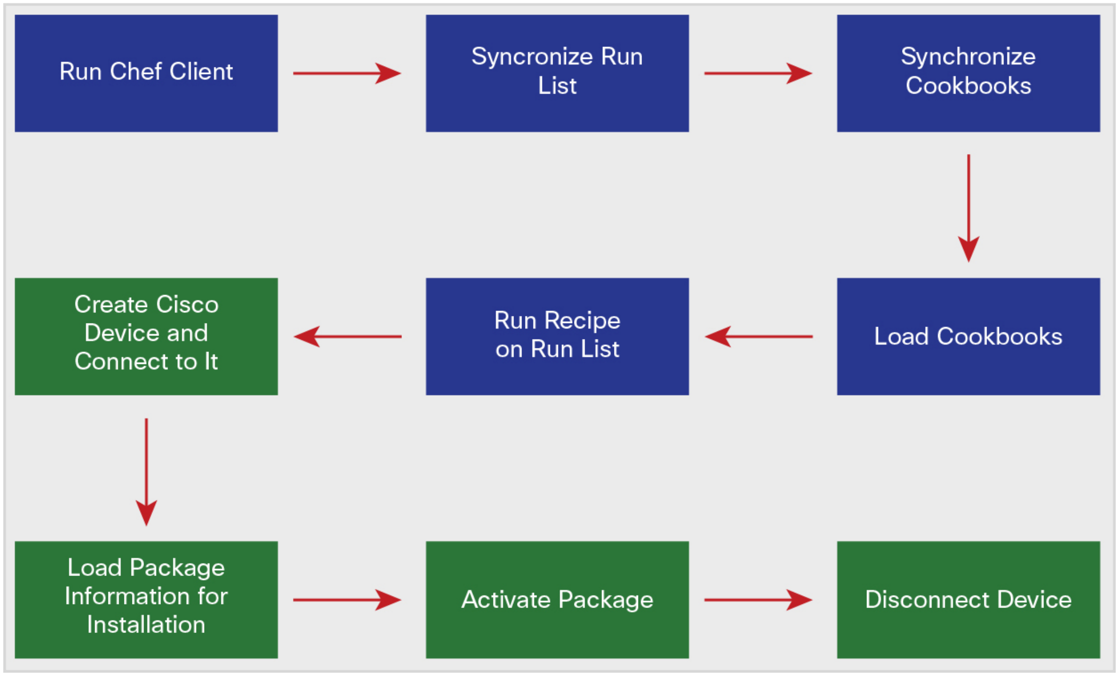
Run Chef Client with Patching Recipe

This scenario shows the Chef client run in one-shot mode to patch the Cisco Nexus device (Figure 21). The device has the correct recipe applied to install the device patch (Figure 22).

Figure 21. Run Chef Client in One-Shot Mode

```
N9K-INS# execute onep application chef v0.8 chef client-oneshot

Execute operation in progress, please see status in log using 'sh onep
applications chef <ver> <instance name> client last-exec-log'
```

Figure 22. Chef Client Processing to Patch Cisco Nexus Device



Verify the Run in the Logs

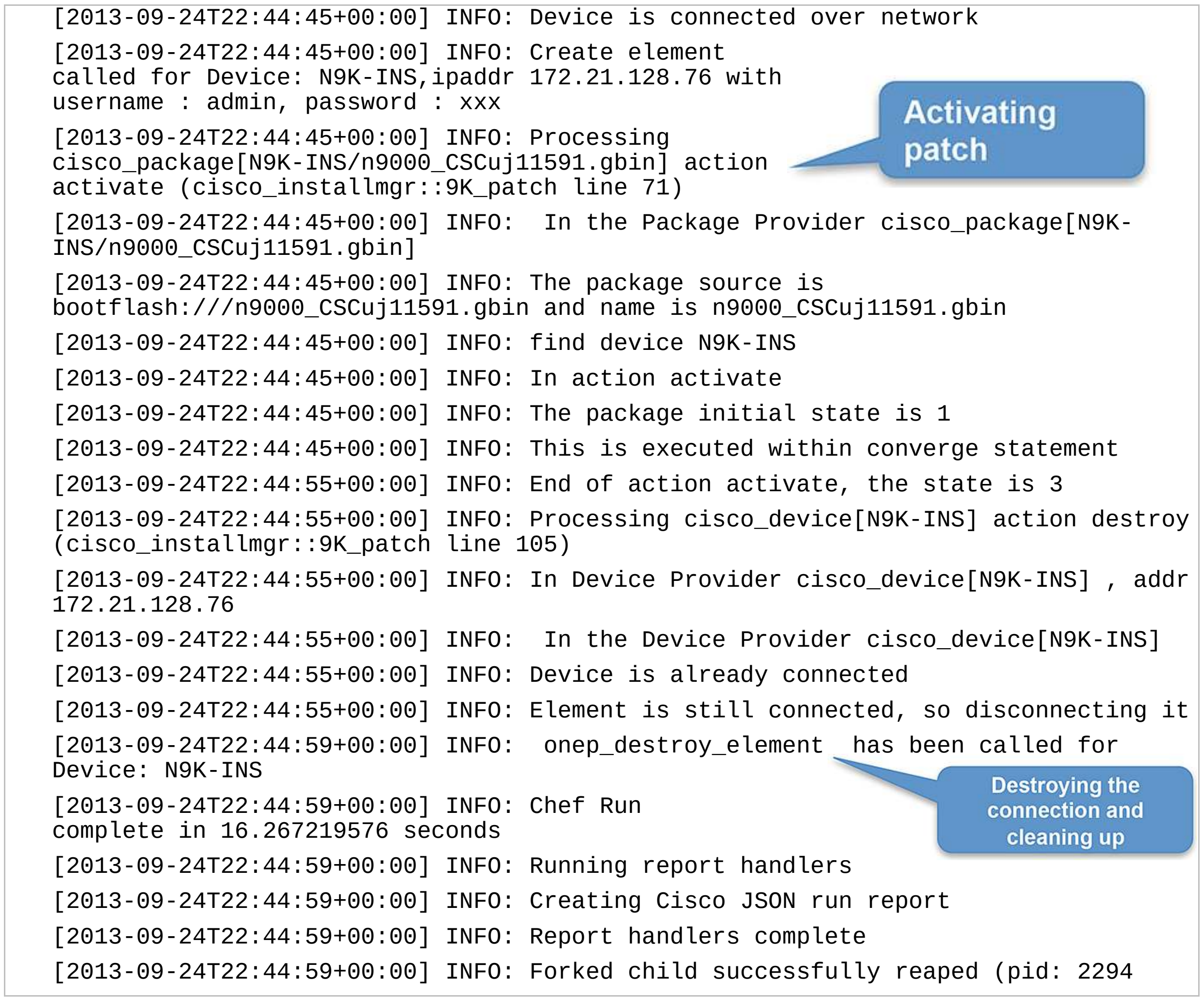
Check the logs to verify the Chef run (Figures 23 and 24).

Figure 23. Use Chef Run Logs for Verification

```
N9K-INS# show onep application chef v0.8 chef client last-exec-log

# Logfile created on 2013-09-24 22:44:42 +0000 by logger.rb/31641
[2013-09-24T22:44:42+00:00] INFO: Forking chef instance to converge...
[2013-09-24T22:44:42+00:00] INFO: Fork successful. Waiting for new chef pid: 2294
[2013-09-24T22:44:42+00:00] INFO: Forked instance now converging
[2013-09-24T22:44:42+00:00] INFO: *** Chef 11.4.0 ***
[2013-09-24T22:44:43+00:00] WARN: unable to detect ipaddress
[2013-09-24T22:44:43+00:00] WARN: unable to detect macaddress
[2013-09-24T22:44:43+00:00] WARN: unable to detect ip6address
```

```
[2013-09-24T22:44:43+00:00] INFO: Run List is [recipe[utit::9K_patch]]
[2013-09-24T22:44:43+00:00] INFO: Run List expands to [utit::9K_patch]
[2013-09-24T22:44:43+00:00] INFO: Starting Chef Run for N9K-INS
[2013-09-24T22:44:43+00:00] INFO: Running start handlers
[2013-09-24T22:44:43+00:00] INFO: Start handlers complete.
[2013-09-24T22:44:43+00:00] INFO: Loading cookbooks [utit]
[2013-09-24T22:44:43+00:00] INFO: Removing cookbooks/cisco_installmgr/results
from the cache; its cookbook is no longer needed on this client.
[2013-09-24T22:44:43+00:00] INFO: Removing
cookbooks/cisco_installmgr/CHANGELOG.md from the cache; its cookbook is no longer
needed on this client.
..
..
..
..
[2013-09-24T22:44:43+00:00] INFO: Removing
cookbooks/cisco_installmgr/recipes/9K_patch.rb from the cache; its cookbook is no
longer needed on this client.
[2013-09-24T22:44:43+00:00] INFO: Removing cookbooks/cisco_installmgr/metadata.rb
from the cache; its cookbook is no longer needed on this client.
[2013-09-24T22:44:43+00:00] INFO: Storing updated
cookbooks/cisco_installmgr/recipes/default.rb in the cache.
[2013-09-24T22:44:43+00:00] INFO: Storing updated
cookbooks/cisco_installmgr/recipes/rp32.rb in the cache.
..
..
..
..
[2013-09-24T22:44:44+00:00] INFO: Storing updated
cookbooks/cisco_installmgr/recipes/rp28.rb in the cache.
[2013-09-24T22:44:44+00:00] INFO: Storing updated
cookbooks/cisco_installmgr/metadata.rb in the cache.
[2013-09-24T22:44:44+00:00] INFO: Storing updated
cookbooks/cisco_installmgr/CHANGELOG.md in the cache.
[2013-09-24T22:44:44+00:00] INFO: Storing updated
cookbooks/cisco_installmgr/results in the cache.
[2013-09-24T22:44:44+00:00] INFO: Storing updated
cookbooks/cisco_installmgr/README.md in the cache.
[2013-09-24T22:44:45+00:00] WARN: Cloning resource attributes for
cisco_device[N9K-INS] from prior resource (CHEF-3694)
[2013-09-24T22:44:45+00:00] WARN: Previous cisco_device[N9K-INS]:
/var/chef/cache/cookbooks/cisco_installmgr/recipes/9K_patch.rb:47:in `from_file'
[2013-09-24T22:44:45+00:00] WARN: Current  cisco_device[N9K-INS]:
/var/chef/cache/cookbooks/cisco_installmgr/recipes/9K_patch.rb:105:in `from_file'
[2013-09-24T22:44:45+00:00] INFO: Processing cisco_device[N9K-INS] action create
(cisco_installmgr::9K_patch line 47)
[2013-09-24T22:44:45+00:00] INFO: In Device Provider cisco_device[N9K-INS] , addr
172.21.128.76
[2013-09-24T22:44:45+00:00] INFO:  In the Device Provider cisco_device[N9K-INS]
[2013-09-24T22:44:45+00:00] INFO: dev: N9K-INS, login: admin, psswd xxx
[2013-09-24T22:44:45+00:00] INFO: app is nill...so creating
[2013-09-24T22:44:45+00:00] INFO: element is nill, so calling new
```

Recipe downloaded from the server



```
[2013-09-24T22:44:45+00:00] INFO: Device is connected over network
[2013-09-24T22:44:45+00:00] INFO: Create element
called for Device: N9K-INS,ipaddr 172.21.128.76 with
username : admin, password : xxx
[2013-09-24T22:44:45+00:00] INFO: Processing
cisco_package[N9K-INS/n9000_CSCuj11591.gbin] action
activate (cisco_installmgr::9K_patch line 71)
[2013-09-24T22:44:45+00:00] INFO:  In the Package Provider cisco_package[N9K-
INS/n9000_CSCuj11591.gbin]
[2013-09-24T22:44:45+00:00] INFO: The package source is
bootflash:///n9000_CSCuj11591.gbin and name is n9000_CSCuj11591.gbin
[2013-09-24T22:44:45+00:00] INFO: find device N9K-INS
[2013-09-24T22:44:45+00:00] INFO: In action activate
[2013-09-24T22:44:45+00:00] INFO: The package initial state is 1
[2013-09-24T22:44:45+00:00] INFO: This is executed within converge statement
[2013-09-24T22:44:55+00:00] INFO: End of action activate, the state is 3
[2013-09-24T22:44:55+00:00] INFO: Processing cisco_device[N9K-INS] action destroy
(cisco_installmgr::9K_patch line 105)
[2013-09-24T22:44:55+00:00] INFO: In Device Provider cisco_device[N9K-INS] , addr
172.21.128.76
[2013-09-24T22:44:55+00:00] INFO:  In the Device Provider cisco_device[N9K-INS]
[2013-09-24T22:44:55+00:00] INFO: Device is already connected
[2013-09-24T22:44:55+00:00] INFO: Element is still connected, so disconnecting it
[2013-09-24T22:44:59+00:00] INFO:  onep_destroy_element  has been called for
Device: N9K-INS
[2013-09-24T22:44:59+00:00] INFO: Chef Run
complete in 16.267219576 seconds
[2013-09-24T22:44:59+00:00] INFO: Running report handlers
[2013-09-24T22:44:59+00:00] INFO: Creating Cisco JSON run report
[2013-09-24T22:44:59+00:00] INFO: Report handlers complete
[2013-09-24T22:44:59+00:00] INFO: Forked child successfully reaped (pid: 2294
```

Figure 24. Verify Installed Packages on Cisco Nexus Device

```
TME-1-9508-2# sh install active
Boot Images:
        Kickstart Image: bootflash:/n9000_dk9.6.1.1.257.gbin
        System Image: package:/isanboot/bin/images/sys

Active Packages:
        n9000_CSCuj11591.gbin
```

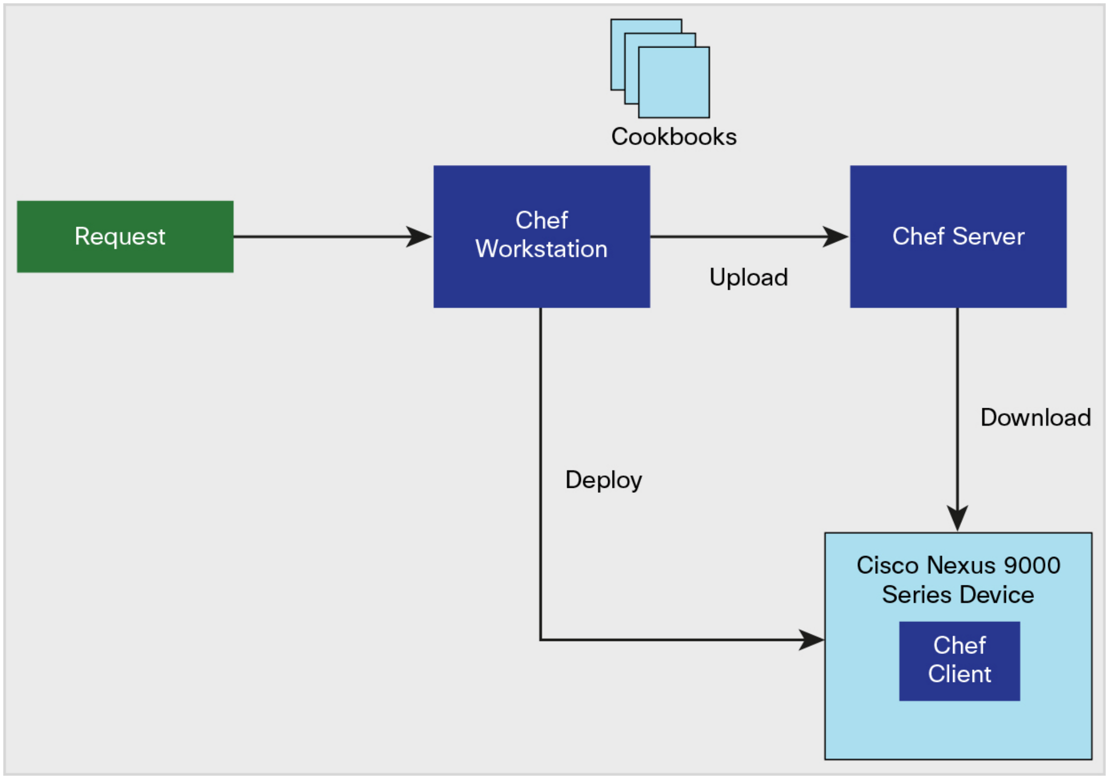
Appendix A. Chef Concepts

Chef Overview

Chef is a systems and cloud infrastructure automation framework that makes it easy to deploy servers and applications to any physical, virtual, or cloud location, no matter the size of the infrastructure.

Chef uses the knife tool to perform various tasks in the environment. The knife tool can be run on the server itself, or a dual-server model can be usedin which a server and a workstation running the knife tool are implemented. The administrator users loginto the relevant system running knife to perform the various chef tasks, such as creation and modification of cookbooks and node management. The cookbooks are configured or uploaded to the server, using knife. Then they are pushed to the managed nodes (Figure 25).

Figure 25. Chef Operating Model



Definitions

- Chef server: The server acts as a hub for configuration data. The server stores cookbooks, the policies that are applied to nodes, and metadata that describes each registered node that is being managed by the Chef client.
- Chef workstation: A workstation is a computer that is configured to run the knife tool to synchronize with the Chef repository (chef-repo) and interact with a single server. The workstation is the location from which most users will do most of their work, including:
    - Developing cookbooks and recipes (and authoring them using Ruby)
    - Keeping chef-repo synchronized with version source control
    - Using the knife tool to upload items from chef-repo to the server

- Configuring organization policy, including defining roles and environments and helping ensure that critical data is stored in data bags
- Interacting with nodes as required, such as to perform a bootstrap operation

Note: The knife tool can be run on the Chef server, which would eliminate the need to have a dedicated workstation to run it.

- Knife: Knife is a command-line tool that provides an interface between a local chef-repo and the server. Knife helps users manage:
  - Nodes
  - Cookbooks and recipes
  - Roles
  - Stores of JSON data (data bags), including encrypted data
  - Environments
  - Cloud resources, including provisioning
  - Installation of the Chef client on management workstations
  - Searches of indexed data on the server
- Chef client: A Chef client is an agent that runs locally on every node that is registered with the server. When a Chef client is run, it performs all the steps required to bring the node into the expected state.
- Node: A node is any server or virtual server that is configured to be maintained by a Chef client. A node can be any physical, virtual, or cloud devicethat can run the Chef client.
- Cookbook: A cookbook is the fundamental unit of configuration and policy distribution. Each cookbook defines a scenario, such as everything needed to install and configure MySQL, and it contains all the components required to support that scenario.
- Recipe: A recipe is the most fundamental configuration element within the organization. A recipe:
  - Is authored using Ruby, which is a programming language designed to read and behave in a predictable manner
  - Is mostly a collection of resources in Ruby syntax with some helper code around it
  - Must define everything that is required to configure part of a system
  - Must be stored in a cookbook
  - May be included in a recipe
  - May use the results of a search query and read the contents of a data bag (including an encrypted data bag)
  - May be dependent on one (or more) recipes
  - May be tagged to facilitate the creation of arbitrary groupings that exist outside the normal naming conventions an organization may have
  - Must be added to a run list before it can be used by the Chef client
  - Is always processed in the same order as listed in a run list

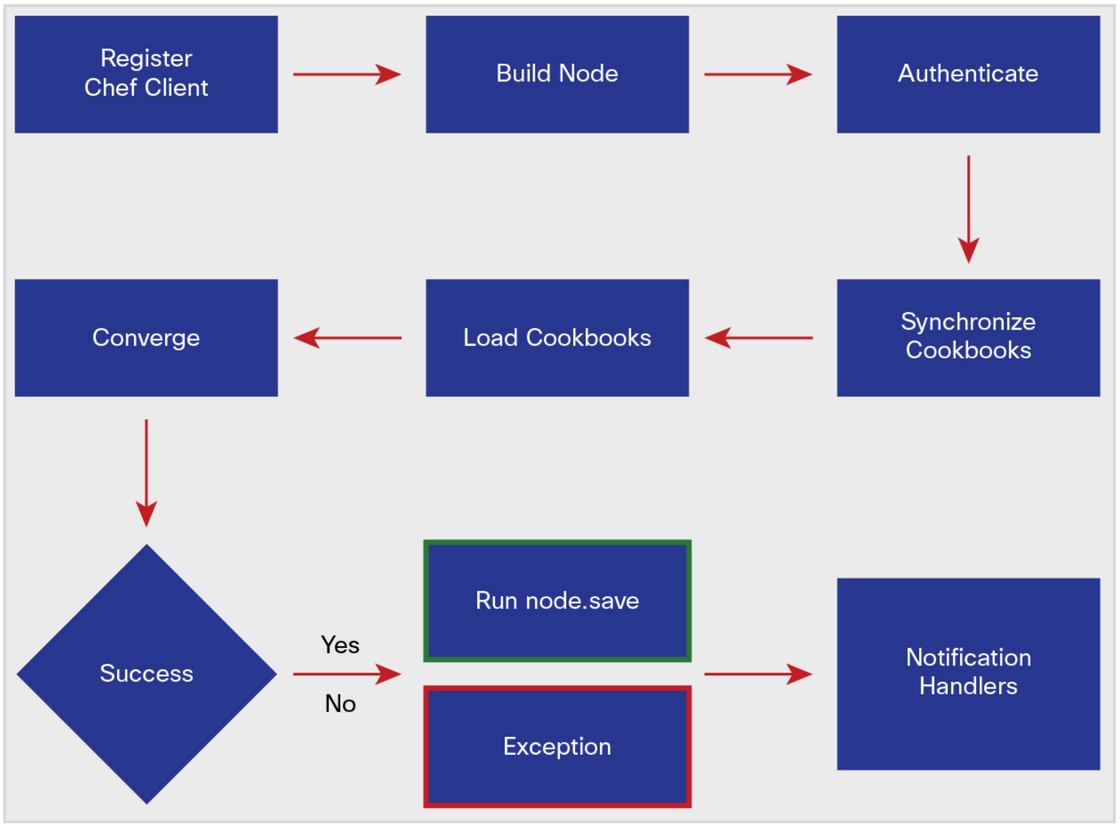
- Run list: A runlist is an ordered list of roles and recipes that are run in an exact order. A runlist is always specific to the node on which it runs, though multiple nodes can have runlists that are similar or even identical.

Flow of a Chef Run

The following steps occur during a Chef client run (Figure 26):

- The node is registered and authenticated with the server.
- The node object is built on the server.
- Cookbooks are synchronized.
- Resource collection is compiled by loading required:
  - Cookbooks
  - Recipes
  - Attributes
  - Any other dependencies
- Actions appropriate to the runlist are performed to configure the node.
- Exception and notification analysis and handling is performed.

Figure 26. Flow of a Chef Run



For More Information

See docs.opscode.com.