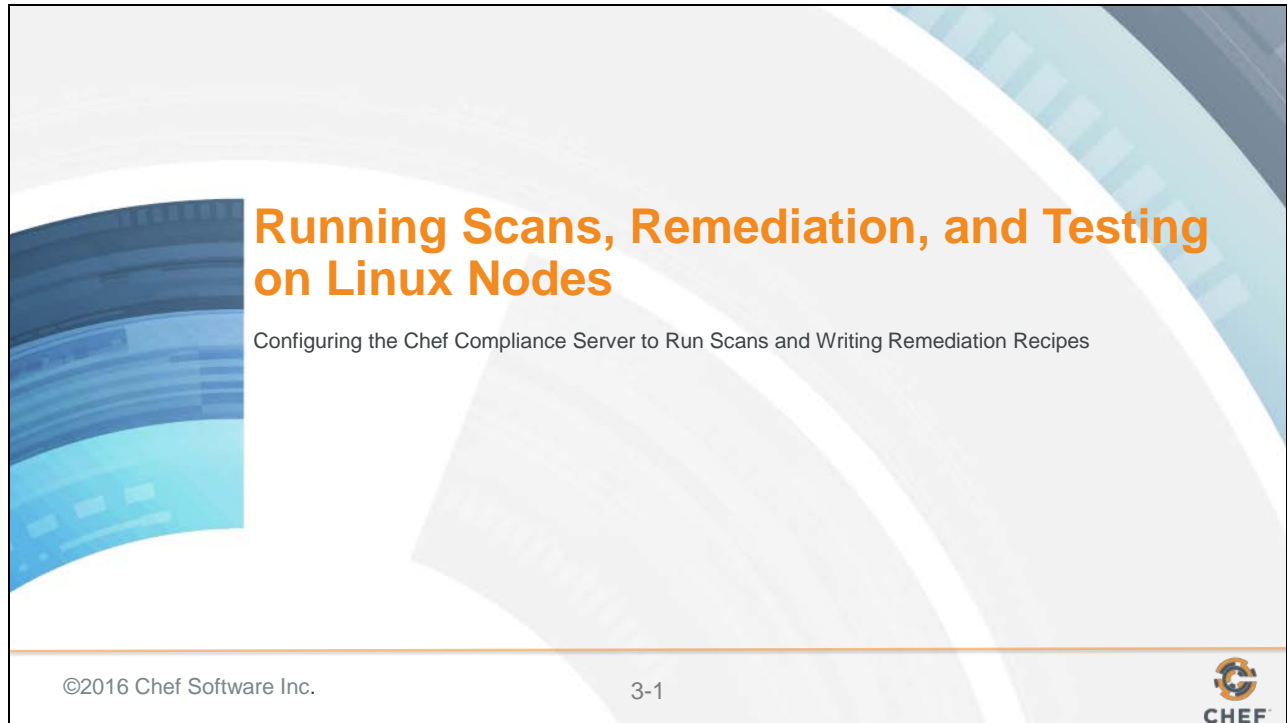


3: Running Scans, Remediation, and Testing on Linux Nodes



Instructor Note: Answers to quizzes are contained in Instructor Notes found below each quiz slide so participants won't see the answers.

Slide 2

Objectives

After completing this module, you should be able to:

- Add a node to test for compliance.
- Run a Compliance scan.
- Test for compliance with InSpec from the command line interface.
- Remediate a compliance issue.
- Use Test Kitchen to test your remediation.
- Test for compliance with InSpec from the CLI
- Rescan the node and ensure compliance.

Slide 3

CONCEPT



Adding a Node to Scan

To add a node you'll need:

- The IP address or FQDN of the nodes to be tested.
- Access configuration (ssh or WinRM).
- The node's username and password OR
- The node's username plus security key pair.

Slide 4

EXERCISE



Group Lab: Adding a Node to Scan

Objective:

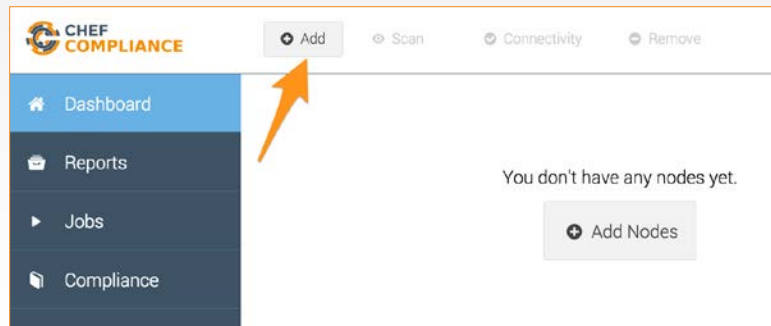
- ☐ Add a Linux Node to Scan
- ☐ Test connectivity

Note: In the next module you will perform the same exercises as in this module but using a Windows node as your target node.

Slide 5

GL: Adding a Node to Scan

1. From your Chef Compliance Dashboard, click Add Node.



Slide 6

GL: Adding a Node

2. From the resulting page, enter the node's FQDN or IP address.
3. Leave environment blank. A 'default' environment will be used
4. Accept the default **SSH** Access configuration
5. Type **chef** in the **username** field.
6. Click the **password** link as shown in this illustration.

Dashboard / Add nodes

Enter nodes (IPs or hostnames):
ec2-52-91-159-53.compute-1.amazonaws.com x Add your nodes via IP or hostname

Add to environment:
default

Access configuration:
SSH WinRM

Username
chef

Use Key Pair:
Select a login key

Add new key pairs or use login with password instead.

Sudo Configuration:
Disable sudo
Optional sudo password *

Be sure you are using the hostname of the target node that you noted previously in class.

In the workplace, the target node's username and password will likely be different than shown in this example.

We'll discuss using key pair access later in the module.

Slide 7

GL: Adding a Node to Scan

7. Type the password (**chef**) in the password field.
8. Click the **Add 1 node** button as shown in this illustration.

Add to environment:

default

Access configuration:

SSH WinRM

Username

chef

Password-based login is generally not recommended and should be limited to development and legacy systems. Make sure you have a sufficiently complex password configured.

Use login with [public key](#) instead

Sudo Configuration:

☒ Disable sudo

Optional sudo password

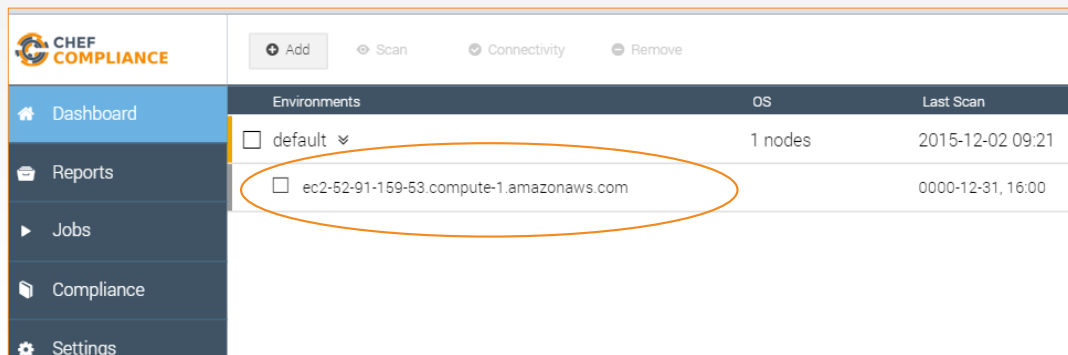
Add 1 node

Instructor Note: If a Linux target node's image has `/etc/sudoers` ``Defaults requiretty`` uncommented, then the Compliance server won't be able to connect to the target node unless we disable sudo on this page. Once the issue is fixed it should not matter if the target node `/etc/sudoers` ``Defaults requiretty`` is uncommented. The Linux AMI used in this course has `/etc/sudoers` ``Defaults requiretty`` commented so no worries."

Slide 8

GL: Adding a Node to Scan

At this point your Compliance Dashboard should list the node you just added.

CHEF
COMPLIANCE

Dashboard

Reports

Jobs

Compliance

Settings

Add

Scan

Connectivity

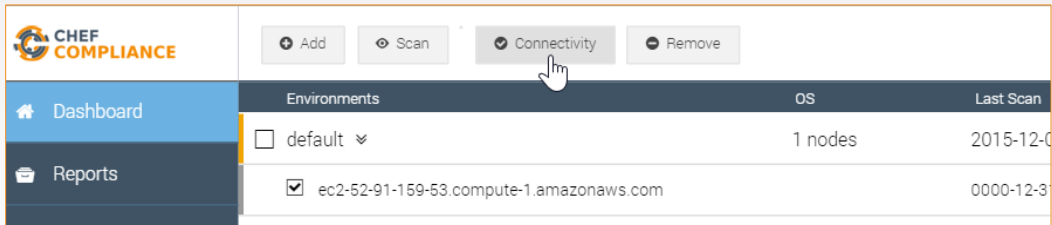
Remove

Environments	OS	Last Scan
<input type="checkbox"/> default ▾	1 nodes	2015-12-02 09:21
<input type="checkbox"/> ec2-52-91-159-53.compute-1.amazonaws.com		0000-12-31, 16:00

Slide 9

GL: Testing Connectivity to Your Node

1. Click the **check box** next to your node and then click the **Connectivity** button.



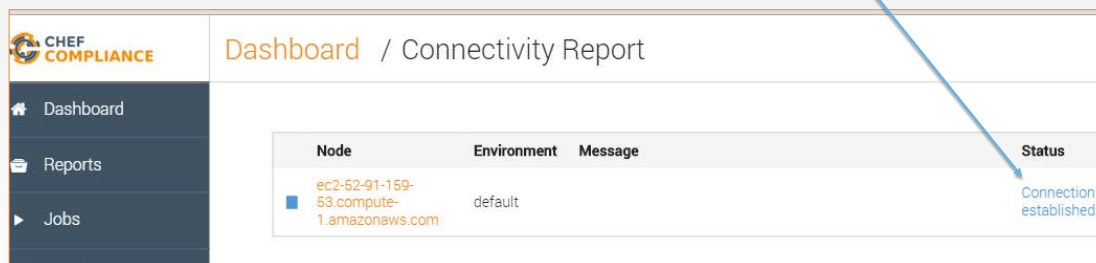
The screenshot shows the Chef Compliance web interface. On the left is a sidebar with 'Dashboard' and 'Reports'. The main area has a top bar with 'Add', 'Scan', 'Connectivity', and 'Remove' buttons. Below is a table with columns 'Environments', 'OS', and 'Last Scan'. The 'Environments' column has a dropdown menu with 'default' and a checked checkbox next to 'ec2-52-91-159-53.compute-1.amazonaws.com'. The 'OS' column shows '1 nodes' for the default environment and '0000-12-31' for the selected node. The 'Last Scan' column shows '2015-12-01' for the default environment and '0000-12-31' for the selected node. A hand cursor is pointing at the 'Connectivity' button.

Environments	OS	Last Scan
<input type="checkbox"/> default	1 nodes	2015-12-01
<input checked="" type="checkbox"/> ec2-52-91-159-53.compute-1.amazonaws.com		0000-12-31

Slide 10

GL: Testing Connectivity to Your Node

The Status column of you node should now indicate **Connection established**.



Dashboard / Connectivity Report			
Node	Environment	Message	Status
ec2-52-91-159-53.compute-1.amazonaws.com	default		Connection established

If your Status column does not indicate **Connection established**, please notify the instructor.

Slide 11

Adding Nodes in Bulk

You could add additional nodes by simply repeating the previous steps.

You could also add a number of nodes at once by separating each hostname or IP address with a comma or a space, as shown in this illustration.

Chef Compliance also supports bulk loading of nodes via API.

Dashboard / Add nodes

Enter nodes (IPs or hostnames):

ec2-52-23-162-169.compute-1.amazonaws.com x ec2-52-91-130-201.compute-1.amazonaws.com x

Add your nodes via IP or hostname

Add to environment:

default

Access configuration:

SSH WinRM

Username:

chef

Password-based login is generally not recommended and should be limited to development and legacy systems. Make sure you have a sufficiently complex password configured.

Use login with **public key** instead.

Sudo Configuration:


☐ Disable sudo

Optional sudo password

Add 2 nodes

©2016 Chef Software Inc.

3-11

 CHEF

As you may have noticed, you could add additional nodes by simply repeating the previous steps.

You could also add a number of nodes at once by separating each hostname or IP address with a comma or a space, as shown in this illustration.

Chef Compliance also supports bulk loading of nodes via API.

Slide 12

Adding Nodes in Bulk via API

After class you can go to the following link.

The resulting `kitchen_sink.rb` will step you through how to upload nodes in bulk.

```
1  ### Script to export Chef Server nodes and import them to Chef Compliance
2  ### Change the 'api_url', 'api_user' and 'api_pass' variables below
3  ### Go to your chef-repo and check Chef Server access first
4  # cd chef-repo; knife environment list
5  ### Save this Ruby script as kitchen_sink.rb and run it like this:
6  # cat kitchen_sink.rb | knife exec
7  ### Chef Compliance API docs: https://docs.chef.io/api_compliance.html
8
9  require 'json'
10 require 'uri'
11 require 'net/http'
12 require 'openssl'
13
14 # This extracts data from the Chef Server. Auth done by 'knife exec'
15 # Change loginkey and any other details that will be posted to the Chef Compliance API:
16 nodes_array = []
17 nodes.find(':*') { |n|
18   nodes_array << { id: n.name,
19                     name: n.name,
```

<https://gist.github.com/alexp0p/01b0bba8d259adeeee320>

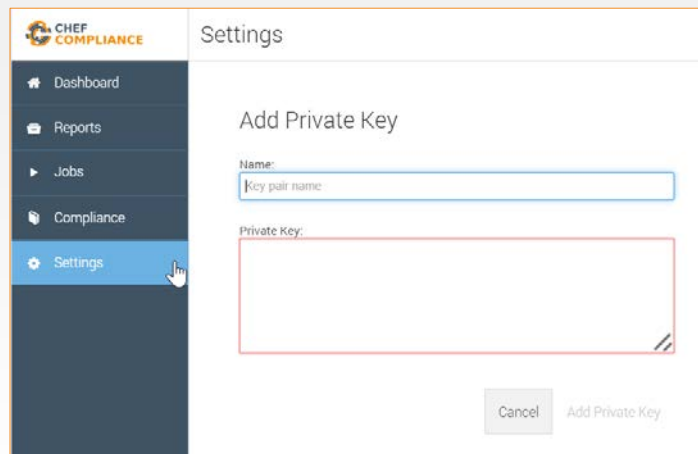
Instructor Note: If you have extra time, you can walk the participants through this file.

Slide 13

Private Keys

In the workplace, using a security key would be a more secure method for connecting to nodes than using the password method.

By clicking **Settings > Add Private Key** you will see where to paste a private key.



The screenshot shows the 'Settings' page in the Chef Compliance interface. On the left is a dark sidebar with a menu containing 'Dashboard', 'Reports', 'Jobs', 'Compliance', and 'Settings'. The 'Settings' item is highlighted with a blue bar and a mouse cursor. The main content area is titled 'Settings' and 'Add Private Key'. It contains two input fields: 'Name:' with a placeholder 'Key pair name' and 'Private Key:' with a large text area. At the bottom right are two buttons: 'Cancel' and 'Add Private Key'.

In the workplace, using security key pairs would be a more secure method for connecting to nodes than using the password method we are using in class.

By clicking **Settings > Add Private Key** you will see where to paste your private key.

Slide 14

CONCEPT



Running Compliance Scans

You can run Compliance scans on demand or schedule them to run at a later time.

Chef Compliance maintains profiles as a collection of individual controls that comprise a complete audit.

As mentioned previously, Chef Compliance comes with a few reference profiles of various compliance policies that you can leverage or use as examples to create your own.

Slide 15

Compliance Profiles Used in Scans

This image shows the default Compliance Profiles as accessed from the Scan Nodes page.

You should be thoughtful with which profiles choose.

Notice how you can also choose to run a scan on demand or schedule a scan.

©2016 Chef Software Inc. 3-15

This image shows the default Compliance Profiles as accessed from the Scan Nodes page. This page displays when you select nodes to scan and then click the Scan button.

You'll access the profiles in a moment. These profiles determine what will be scanned on your nodes.

You should be thoughtful with which profiles choose since the more you choose to run, the longer it will take to execute the scan.

Notice how you can also choose to run a scan on demand (Scan now) or schedule a scan to run at a later time.

Slide 16

EXERCISE



Group Lab: Running a Scan

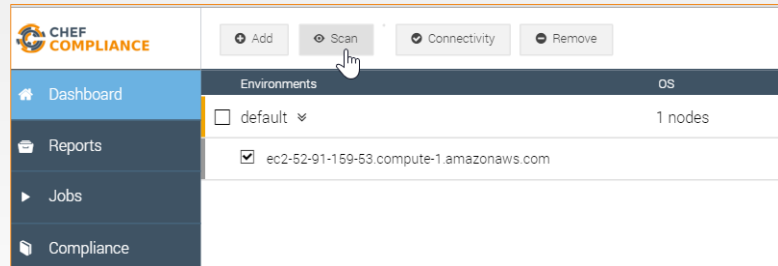
Objective:

- ☐ Run a Compliance scan.
- ☐ View the output of a scan.

Slide 17

GL: Running a Scan

1. Click the **check box** next to your node and then click the **Scan** button.




Slide 18

GL: Running a Scan

2. From the resulting page, check the **base/ssh** profile and uncheck any other check boxes.
3. Click the **Scan now** button.

Dashboard / Scan nodes

Target nodes: 1 host 
ec2-52-91-159-53.compute-1.amazonaws.com

Patch Management ☐ Operating System Patch Level

Compliance:

- ☐ base/apache
- ☐ base/linux
- ☐ base/mysql
- ☐ base/postgres
- ☒ base/ssh
- ☐ base/windows
- ☐ cis/cis-ubuntu-level1
- ☐ cis/cis-ubuntu-level2

Scan now Schedule

Slide 19

Scan Results

A Compliance Report should now display and your scan results should be similar to that shown here.

Notice how in the upper Summary section in this example, 10 tests were compliant and 6 tests show critical issues with ssh.

Reports / Compliance Report

Summary

Time: 2015-12-02 13:12 (Ran for a few seconds)
Targets: 1 nodes

Compliance

- Compliant: 10 tests
- Minor Issues: 14 tests
- Major Issues: 32 tests
- Critical Issues: 6 tests
- Other: 0 rules

Compliance Overview

Major 32

Hostname	Compliant	Minor Issues	Major Issues	Critical Issues	Skipped
ec2-52-91-159-53.compute-1.amazonaws.com	10	14	32	6	0

base/ssh: Client: Set SSH protocol version to 2
base/ssh: Server: Enable strict mode
base/ssh: Server: Disable empty passwords
base/ssh: Server: Disable X11 forwarding
base/ssh: Server: Ignore legacy .rhosts configuration
base/ssh: Server: If X11 is used, enforce localhost
base/ssh: Server: Configure a listen address
base/ssh: Server: Configure the service port

Critical issues
Critical issues
Critical issues
Critical issues
Critical issues
Critical issues
Major issues
Major issues

©2016 Chef Software Inc. 3-19

CHEF

There are also 6 critical issues related to ssh on the target node. Your results may be slightly different than this example.

Instructor Note: This and the following slide should be used for a discussion of the scan results. The group exercise continues after that.

Slide 20

Scan Results

The bottom half of the Compliance Report shown here has a table of details of test results.

These are sorted by severity.

If you click an issue as shown here, a bit more information about the issue displays.

Hostname	Compliant	Minor Issues	Major Issues	Critical Issues	Skipped
ec2-52-91-159-53.compute-1.amazonaws.com	10	14	32	6	0
base/ssh: Client: Set SSH protocol version to 2				Critical Issues	■
SSH Configuration Protocol should eq "2"				10.0	
base/ssh: Server: Enable strict mode				Critical Issues	■
base/ssh: Server: Disable empty passwords				Critical Issues	■
base/ssh: Server: Disable X11 forwarding				Critical Issues	■
base/ssh: Server: Ignore legacy .rhosts configuration				Critical Issues	■
base/ssh: Server: If X11 is used, enforce localhost				Critical Issues	■
base/ssh: Server: Configure a listen address				Major Issues	■
base/ssh: Server: Configure the service port				Major Issues	■
base/ssh: /etc/ssh should have limited access to 0755				Major Issues	■

The bottom half of the Compliance Report has a table of details of test results.

These are sorted by severity so the critical issues are listed at the top and the compliant items are at the bottom of the list.

If you click an issue as shown here, a bit more information about the issue displays, but that's not really telling us much.

Slide 21

GL: Profile

To view the InSpec code that comprises this profile, do the following:

1. Click the **Compliance** button.
2. Click the relevant profile (**Basic SSH**).
3. Scroll down and click the **`Set SSH protocol version to 2`** profile.

The screenshot shows the Chef Compliance web interface. On the left is a navigation menu with 'Dashboard', 'Reports', 'Jobs', 'Compliance', and 'Settings'. The 'Compliance' menu item is highlighted. The main content area is titled 'Reports / Compliance Report' and contains a 'Summary' section with status indicators (Compliant, Minor issues, Major issues, Critical issues, Other). To the right is a 'Compliance profiles' table with a list of profiles: Basic Apache 2, Basic Linux, Basic MySQL, Basic PostgreSQL, and Basic SSH. The 'Basic SSH' profile is highlighted. Below the table, the InSpec code for the 'Set SSH protocol version to 2' profile is displayed. The code includes a control block for 'ssh-4' with an impact of 1.0, a title, a description, and a describe block for 'ssh_config' that checks if the protocol is set to '2'.

Name
Basic Apache 2
Basic Linux
Basic MySQL
Basic PostgreSQL
Basic SSH

```
control 'ssh-4' do
  impact 1.0
  title 'Client: Set SSH protocol version to 2'
  desc "
    Set the SSH protocol version to 2. Don't use legacy insecure SSHv1 connections anymore.
  "
  describe ssh_config do
    its('Protocol') { should eq('2') }
  end
end
```

Instructor Note: Now we continue the group Lab but you should stop as needed to explain what this code means.

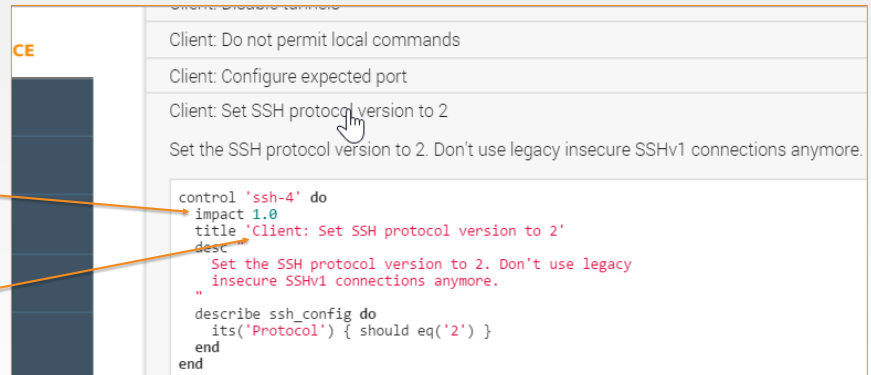
Slide 22

Discussion: InSpec Profile Code

Let's discuss what this profile is doing.

The `impact` of 1.0 indicates this is a Critical issue.

The `title` is what populates the Compliance Report issue title.



```
Client: Do not permit local commands
Client: Configure expected port
Client: Set SSH protocol version to 2
Set the SSH protocol version to 2. Don't use legacy insecure SSHv1 connections anymore.

control 'ssh-4' do
  impact 1.0
  title 'Client: Set SSH protocol version to 2'
  desc
    Set the SSH protocol version to 2. Don't use legacy
    insecure SSHv1 connections anymore.
  describe ssh_config do
    its('Protocol') { should eq('2') }
  end
end
```

Let's discuss what this profile is doing.

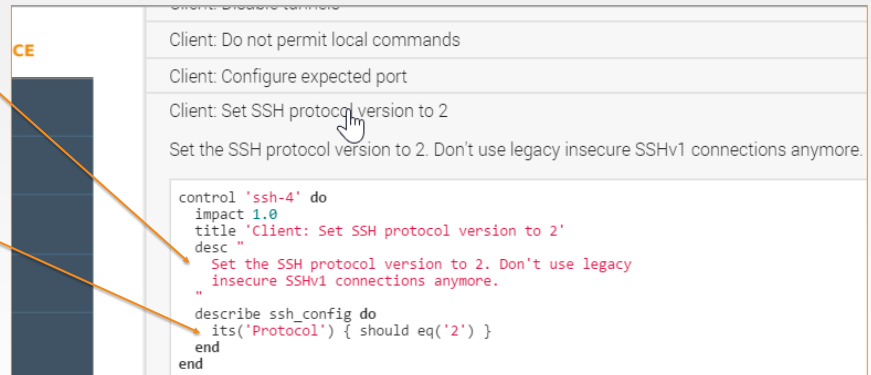
The impact of 1.0 indicates this is a critical issue if it the scanned node violates what is in this code. We'll discuss severity mapping in a moment.

Slide 23

Discussion: InSpec Profile Code

The **desc** is typically human-readable description sourced from the CIS or source doc.

The **`describe`** section is the actual test that is executed.



```
Client: Do not permit local commands
Client: Configure expected port
Client: Set SSH protocol version to 2
Set the SSH protocol version to 2. Don't use legacy insecure SSHv1 connections anymore.

control 'ssh-4' do
  impact 1.0
  title 'Client: Set SSH protocol version to 2'
  desc "
    Set the SSH protocol version to 2. Don't use legacy
    insecure SSHv1 connections anymore.
  "
  describe ssh_config do
    its('Protocol') { should eq('2') }
  end
end
```

The **desc** is typically human-readable description sourced from the CIS or source doc.

The describe value is the actual test. In this case, this is saying the protocol for `ssh_config` Protocol should be `2`. If the actual value from the node is not Protocol 2, the Critical issue is reported as in this case.

So when you run a scan, the Compliance Server connects to the node using the configuration we specified, in this case ssh, and then it will run this set of code, this InSpec control, on that node. The Compliance Server translates the InSpec code into ssh commands when it transmits across the wire.

No agent or client is required to be running on the target node for this work.

Slide 24

Compliance Profile Severity Mapping

The table below shows the current mapping of Compliance Profile **impact** numbering to severity.

```
Set the SSH protocol version to 2. Don't use legacy insecure S
control 'ssh-4' do
  impact 1.0
  title 'Client: Set SSH protocol version to 2'
  desc "
    Set the SSH protocol version to 2. Don't use legacy
    insecure SSHv1 connections anymore.
  "
  describe ssh_config do
    its('Protocol') { should eq('2') }
  end
end
```

Impact Numbering	Severity Designation
0.7 - 1.0	Critical Issues
0.4 - <0.7	Major Issues
0 - <0.4	Minor Issues

<https://nvd.nist.gov/cvss.cfm>

Critical Issues	■
Critical Issues	■
Critical Issues	■
Major Issues	■
Major Issues	■
Major Issues	■
Minor Issues	■
Minor Issues	■

Here is the current mapping of Compliance Profile **impact** numbering to severity.

The image at the top-right shows a Compliance Profile's impact numbering.

The table at the bottom-left shows the current mapping of Compliance Profile impact numbering to severity.

The image at the bottom-right is an example of the severities listed in the reports in the Compliance web UI.

The mapping is currently analogous to the Common Vulnerability Scoring System (CVSS) framework, which can be viewed via the link at the bottom of this slide.

This mapping will be made configurable in the future.

Example: Node's ssh config



```
$ more /etc/ssh/ssh_config
```

```
# IdentityFile ~/.ssh/identity
# IdentityFile ~/.ssh/id_rsa
# IdentityFile ~/.ssh/id_dsa
# Port 22
# Protocol 2,1
# Ciphers 3des
```

Slide 26

CONCEPT



Let's Remediate the Issue

Now that we've identified the ssh version issue, let's write a recipe on the target node to remediate the issue.

Then we'll run the compliance scan again to see if we successfully remediated the issue.

Note: In this course we will write a recipe directly on the node that we're running scans on. Of course in a production environment you will likely write such recipes locally and upload them to Chef Server. Then the nodes would converge the recipes on their next chef-client run.

Slide 27

EXERCISE



GL: Remediating the Issue

Objective:

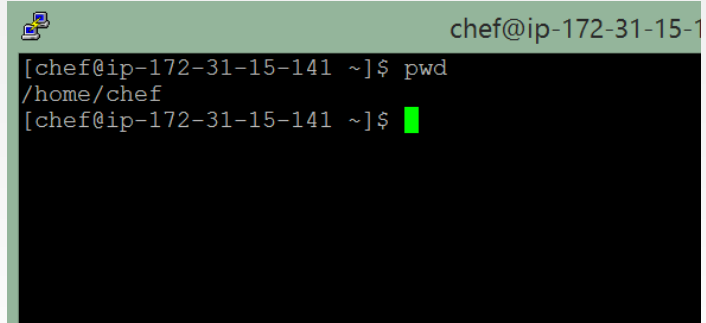
- ☐ Start writing a remediation recipe on that node.
- ☐ Test the recipe with Test Kitchen.
- ☐ Test for compliance with InSpec from the command line interface (CLI)
- ☐ Converge the recipe.
- ☐ Rescan the node and ensure compliance.

Slide 28

GL: Remediating the Issue

Log in to your **target** node (not your compliance server node) using ssh and ensure you are in the **home directory**.

Note: emacs, nano, and vim/vi are installed on your Linux nodes. Some tips for using them can be found below in your participant guide.



```
chef@ip-172-31-15-141 ~$ pwd
/home/chef
[chef@ip-172-31-15-141 ~]$
```

Emacs: (Emacs is fairly straightforward for editing files.)

OPEN FILE `$ emacs FILENAME`

WRITE FILE `ctrl+x, ctrl+w`

EXIT `ctrl+x, ctrl+c`

Nano: (Nano is usually touted as the easiest editor to get started with editing through the command-line.)

OPEN FILE `$ nano FILENAME`

WRITE (When exiting) `ctrl+x, y, ENTER`

EXIT `ctrl+x`

VIM: (Vim, like vi, is more complex because of its different modes.)

OPEN FILE `$ vim FILENAME`

START EDITING `i`

WRITE FILE `ESC, :w`

EXIT `ESC, :q`

EXIT (don't write) `ESC, :q!`

GL: Create and Change to a 'cookbooks' Directory



```
$ mkdir -p cookbooks  
$ cd cookbooks
```

From the home directory, create a **`cookbooks`** directory and navigate into it.

GL: Create an SSH Cookbook



```
$ chef generate cookbook ssh
```

```
Compiling Cookbooks...
```

```
Recipe: code_generator::cookbook
```

```
  * directory[/home/chef/cookbooks/ssh] action create
```

```
    - create new directory /home/chef/cookbooks/ssh
```

```
...
```

```
  - create new file /home/chef/cookbooks/ssh/recipes/default.rb
```

```
    - update content in file
```

```
/home/chef/cookbooks/ssh/recipes/default.rb from none to b702c7
```

```
(diff output suppressed by config)
```

GL: Create an SSH Client Recipe



```
$ chef generate recipe ssh client
```

```
Compiling Cookbooks...
```

```
Recipe: code_generator::recipe
```

```
  * directory[./ssh/spec/unit/recipes] action create (up to date)
```

```
  * cookbook_file[./ssh/spec/spec_helper.rb] action  
  create_if_missing (up to date)
```

```
...
```

```
- create new file ./ssh/recipes/client.rb
```

```
  - update content in file ./ssh/recipes/client.rb from none to  
  9c833a
```

```
    (diff output suppressed by config)
```

In this step, instead of modifying the default recipe, we will create a new `ssh client` recipe.

This is because a default ssh cookbook probably affects an ssh server config and ssh client config and we only want to affect an ssh client.

GL: Create an SSH Config Template



```
$ chef generate template ssh ssh_config.erb -s /etc/ssh/ssh_config
```

```
Compiling Cookbooks...
```

```
Recipe: code_generator::template
```

```
  * directory[./ssh/templates/default] action create
```

```
    - create new directory ./ssh/templates/default
```

```
  * file[./ssh/templates/default/ssh_config.erb] action create
```

```
    - create new file ./ssh/templates/default/ssh_config.erb
```

```
    - update content in file
```

```
./ssh/templates/default/ssh_config.erb from none to 86eb9b
```

```
(diff output suppressed by config)
```

In this step, we want to create a template file to manage our `ssh_config` file.

The `-s` option in this command takes the contents of the current `/etc/ssh/ssh_config` file and places it in the `ssh_config.erb` file.

Instructor Note: At this time you might want to show the class the contents of `/home/chef/cookbooks/ssh/templates/default/ssh_config.erb` to illustrate how the contents of the current `/etc/ssh/ssh_config` file is now in the `ssh_config.erb` file.

Slide 33

GL: Write the Client Recipe



```
$ ~/cookbooks/ssh/recipes/client.rb
```

```
# Cookbook Name:: ssh
# Recipe:: client
# Copyright (c) 2035 The Authors, All Rights Reserved.

template '/etc/ssh/ssh_config' do
  source 'ssh_config.erb'
  owner 'root'
  group 'root'
  mode '0644'
end
```

Edit the `~/cookbooks/ssh/recipes/client.rb` file and add the contents shown here.

EXERCISE



GL: Testing the Recipe

Objective:

- ✓ Write a remediation recipe on that node.
- ☐ Test the recipe with Test Kitchen.
- ☐ Test for compliance with InSpec from the command line interface (CLI)
- ☐ Converge the recipe.
- ☐ Rescan the node and ensure compliance.

Slide 35

GL: Navigate to your SSH Cookbook



```
$ cd ~/cookbooks/ssh/
```

To test your recipe, first navigate to where the recipe lives.

Slide 36

GL: Edit your .kitchen.yml -- Part 1

 ~/cookbooks/ssh/.kitchen.yml

```
---  
driver:  
  name: docker  
  
provisioner:  
  name: chef_zero
```

Edit your ``.kitchen.yml`` as shown here and on the following slide.

Because our node is an EC2 AWS instance, we need to change the driver from `vagrant` to `docker`. `docker` should already be installed on the EC2 AWS training instances.

Slide 37

GL: Edit your .kitchen.yml -- Part 2

 `~/cookbooks/ssh/.kitchen.yml`

```
platforms:
#  - name: ubuntu-14.04
  - name: centos-6.7

suites:
  - name: default
    run_list:
      - recipe[ssh::default]
    attributes:
```

Also comment the ubuntu platform line and change the centos platform to `centos-6.7`, which should be the version running on the training instance.

To confirm the centos release, you could execute `more /etc/*-release`

```
.....
```

```
/etc/centos-release
```

```
.....
```

```
CentOS release 6.7 (Final)
```

Slide 38

GL: Edit your .kitchen.yml -- Part 3

 ~/cookbooks/ssh/.kitchen.yml

```
platforms:
#  - name: ubuntu-14.04
  - name: centos-6.7

suites:
- name: client
  run_list:
    - recipe[ssh::client]
  attributes:
```

Finally, change the suites name to `client` and the run_list recipe name to `ssh:client`.

run_list:

- recipe[ssh::client]

Slide 39

GL: Run `kitchen list` from ~/cookbooks/ssh/



```
$ kitchen list
```

Instance	Driver	Provisioner	Verifier	Transport	Last Action
client-centos-67	Docker	ChefZero	Busser	Ssh	<Not Created> :

Now run `kitchen list` from the ~/cookbooks/ssh directory. This command will tell you if you have a typo in your `.kitchen.yml`.

Slide 40

GL: Run `kitchen converge`



```
$ kitchen converge
```

```
-----> Starting Kitchen (v3.4.2)
-----> Creating <client-centos-67>...
      Sending build context to Docker daemon 32.26 kB
      Sending build context to Docker daemon
      Step 0 : FROM centos:centos6
      ---> 3bbbf0aca359
...
Chef Client finished, 0/3 resources updated in 03 seconds
      Finished converging <client-centos-67> (0m28.27s).
-----> Kitchen is finished. (0m30.32s)
zlib(finalizer): the stream was freed prematurely.
```

Now run `kitchen converge` from the `~/cookbooks/ssh` directory.

`kitchen converge` will:

- Launch a docker container.
- Place the cookbook into the docker container.
- Install chef-client in the docker container.
- Run chef zero (i.e., chef-client in local mode) across the client recipe.

The end result will be that it should write out the `ssh_conf` to the appropriate location (i.e., `/etc/ssh/ssh_config`).

It could take a couple minutes or so for this command to complete.

Slide 41

CONCEPT



What We've Done So Far

In the preceding exercises, we began writing a remediation recipe on our target node.

We also tested the recipe with Test Kitchen.

But have we even addressed the "Set the SSH protocol version to 2" issue?

©2016 Chef Software Inc. 3-41 

In the preceding exercises we began writing a remediation recipe on our target node.

We also tested the recipe with Test Kitchen.

But have we even addressed the "Set the SSH protocol version to 2" issue?

If you answered "no", you are correct. In a little while we will modify our recipe to address the "Set the SSH protocol version to 2" issue.

EXERCISE



GL: Using InSpec for Verification

Objective:

- ✓ Write a remediation recipe on that node.
- ✓ Test the recipe with Test Kitchen.
- ❑ Test for compliance with InSpec from the command line interface (CLI)
- ❑ Converge the recipe .
- ❑ Rescan the node and ensure compliance.

Slide 43

GL: Create the `inspec` Directory



```
$ mkdir -p ~/cookbooks/ssh/test/integration/client/inspec
```

Slide 44

GL: Create the `client_spec.rb` file

 `~/cookbooks/ssh/test/integration/client/inspec/client_spec.rb`

```
control 'ssh-4' do
  impact 1.0
  title 'Client: Set SSH protocol version to 2'
  desc "
    Set the SSH protocol version to 2. Don't use legacy
    insecure SSHv3 connections anymore.
  "
  describe ssh_config do
    its('Protocol') { should eq('2') }
  end
end
```

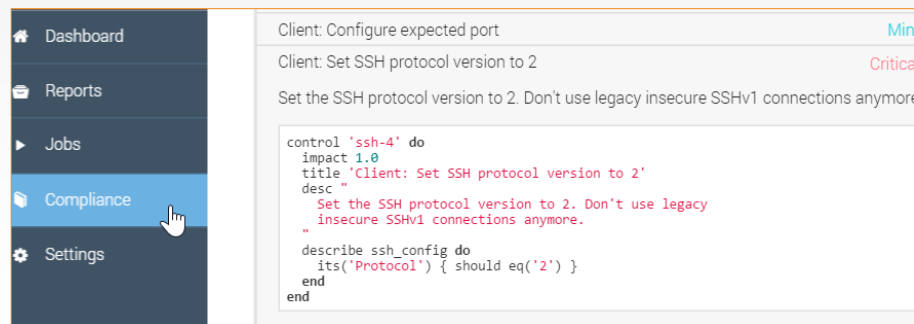
See the following slide for an example of a handy way to populate this file.

Slide 45

Example of Creating the `client_spec.rb` file

One handy way to populate the preceding `client_spec.rb` is to use the Compliance Web UI and copy the InSpec code found in the relevant Compliance profile:

Compliance > Base SSH > Set the SSH protocol version to 2



One handy way to populate the preceding `client_spec.rb` is to use the Compliance Web UI and copy the InSpec control code found in the relevant Compliance profile.

Then you can paste it into the `client_spec.rb` file to save yourself some typing.

Instructor Note: It could also be good for the instructor to demonstrate using the InSpec verifier in test kitchen locally with Vagrant to show the students that it can be done.

Slide 46

CONCEPT



Running InSpec from the Command Line Interface (CLI)

InSpec is an executable application.

InSpec can execute on remote hosts, including docker containers.

You can use 'inspec exec' to run tests at a specified path.

Slide 47

GL: Change Owner of `/var/run/docker.sock`



```
$ sudo chown root:dockerroot /var/run/docker.sock
```


First, we need to run this command because we are using Docker solely for testing and placing it in this configuration is not secure. We are doing it here because it is necessary if we do not want to prefix `sudo` in front of the commands we execute.

So it's done here namely for speed and ease of training so you can focus on Compliance. On your local system you may use vagrant, ec2, or the azure driver and those will not have the same concern that we are experiencing here.


Instructor Note: This command is done in order to put the chef user in the dockerroot group and make /var/run/docker.sock's group dockerroot. This change would not persist when making part of the AMI so we run the command here.

Slide 48


GL: What is your Docker ID?

 `$ sudo docker ps`

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
5b51a4237437	d5b8fd3299b4	"/usr/sbin/sshd -D -	41 minutes ago	Up 41 minutes	0.0.0.0:32768->22/tcp	grave_davinci



CONTAINER ID	IMAGE	COMMAND	CREATED
5b51a4237437	d5b8fd3299b4	"/usr/sbin/sshd -D -	41 minutes ago

©2016 Chef Software Inc.
3-48


Below is an example of the details of the `sudo docker ps` command. This shows one docker container running.

You should only have only one docker container running too.

You'll need the Container ID for the next step so copy your Container ID, which is the first value that is not a header.

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
5b51a4237437	d5b8fd3299b4	"/usr/sbin/sshd -D -	41 minutes ago	Up 41
minutes	0.0.0.0:32768->22/tcp	grave_davinci		

Slide 49

GL: Running InSpec from the CLI



```
$ inspec exec
~/cookbooks/ssh/test/integration/client/inspec/client_spec.rb -t
docker://CONTAINER_ID
```

```
Failures:
  1) SSH Configuration Protocol should eq "2"
     Failure/Error: its('Protocol') { should eq('2') }

     expected: "2"
     got: nil

     (compared using ==)
     # ./test/integration/client/inspec/client_spec.rb:9:in `block (3 levels) in load'

Finished in 0.79369 seconds (files took 0.7207 seconds to load)
1 example, 1 failure

Failed examples:

rspec # SSH Configuration Protocol should eq "2"
```

Run this inspec command using the container ID you just copied, replacing CONTAINER_ID in the example.

```
`inspec exec ~/cookbooks/ssh/test/integration/client/inspec/client_spec.rb -t
docker://CONTAINER_ID`
```

Running InSpec in this way can uncover more complex issues than the basic issue we are remediating in this module.

While the image of the output may be hard to see, key parts of the output is also pasted below. Notice how inspec from the command line also found the "SSH Configuration Protocol should eq "2" non compliance issue.

Key parts of the output is here:
Failures:

```
1) SSH Configuration Protocol should eq "2"
   Failure/Error: its('Protocol') { should eq('2') }
   expected: "2"
   got: nil
```

...

Failed examples:
rspec # SSH Configuration Protocol should eq "2"

GL: Update the Template

 ~/cookbooks/ssh/templates/default/ssh_config.erb

```
# IdentityFile ~/.ssh/id_rsa
# IdentityFile ~/.ssh/id_dsa
# Port 22
# Protocol 2,1
```

- Edit the ~/cookbooks/ssh/templates/default/ssh_config.erb file.
- Uncomment the `# Protocol 2,1` line.
- Change the protocol version to `2` only.

Slide 51

GL: Update the Template

 `~/cookbooks/ssh/templates/default/ssh_config.erb`

```
# IdentityFile ~/.ssh/id_rsa
# IdentityFile ~/.ssh/id_dsa
# Port 22
Protocol 2
```

Results: Your `~/cookbooks/ssh/templates/default/ssh_config.erb` file's Protocol line should now look like this example.

Slide 52

GL: Ensure you are in ~/cookbooks/ssh



```
$ cd ~/cookbooks/ssh
```

Change to ~/cookbooks/ssh if not there already.

Slide 53

GL: Run `kitchen converge`



```
$ kitchen converge
```

```
...
+++ /etc/ssh/.ssh_config20151209-412-cf7gd7 2015-12-09
20:35:29.734689138 +0000
@@ -37,7 +37,7 @@
# IdentityFile ~/.ssh/id_rsa
# IdentityFile ~/.ssh/id_dsa
# Port 22
-# Protocol 2,1
+Protocol 2
# Cipher 3des
```

©2016 Chef Software Inc.

3-53



You should now see that only Protocol version 2 is currently set in test kitchen.

```
- update content in file /etc/ssh/ssh_config from 86eb9b to 065f90
--- /etc/ssh/ssh_config 2015-08-13 09:58:26.000000000 +0000
+++ /etc/ssh/.ssh_config20151209-412-cf7gd7 2015-12-09 20:35:29.734689138
+0000
@@ -37,7 +37,7 @@
# IdentityFile ~/.ssh/id_rsa
# IdentityFile ~/.ssh/id_dsa
# Port 22
-# Protocol 2,1
+Protocol 2
# Cipher 3des
# Ciphers aes128-ctr,aes192-ctr,aes256-ctr,arcfour256,arcfour128,aes128-
cbc,3des-cbc
```

Running handlers:

Chef Client finished, 3/3 resources updated in 03 seconds
Finished converging <client-centos-67> (0m8.22s).

Slide 54

GL: Running InSpec from the CLI



```
$ inspec exec  
~/cookbooks/ssh/test/integration/client/inspec/client_spec.rb -t  
docker://CONTAINER_ID
```

```
.  
  
Finished in 0.21546 seconds (files took 0.3575  
seconds to load)  
1 example, 0 failures
```

Run this inspec command again using the container ID you copied previously, replacing CONTAINER_ID in the example.

```
`inspec exec ~/cookbooks/ssh/test/integration/client/inspec/client_spec.rb -t  
docker://CONTAINER_ID`
```

You should now see that the test has passed. In addition to the output text that says there were 0 failures, the single dot at the top-left of the output means there was one test made and that it passed.

Slide 55

GL: Apply the New SSH Recipe



```
$ sudo chef-client --local-mode -r 'recipe[ssh::client]'
```

```
...  
+++ /etc/ssh/.ssh_config20151209-10413-hlk9ow      2015-12-09  
20:37:07.621689137 +0000  
@@ -37,7 +37,7 @@  
#   IdentityFile ~/.ssh/id_rsa  
#   IdentityFile ~/.ssh/id_dsa  
#   Port 22  
-#   Protocol 2,1  
+Protocol 2  
#   Cipher 3desources updated in 3.29477735 seconds
```

Now we need to actually apply the change to the node. We'll do this using chef-client in local mode. You should then see that only Protocol version 2 is currently set on the node.

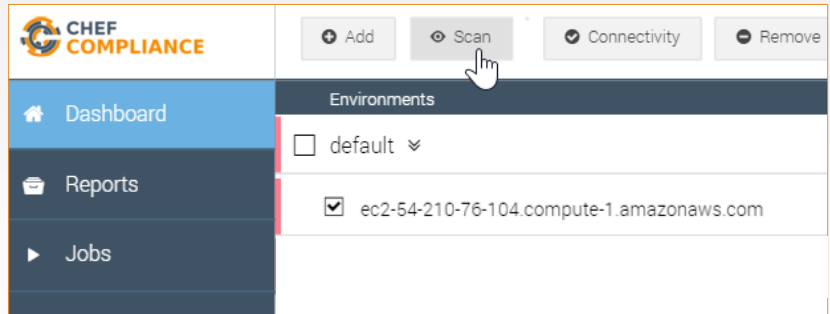
Of course in a production environment chef-client would most likely be set to run automatically to download and converge these changes from Chef Server.

Slide 56

GL: Re-run the Compliance Scan

Return to the Compliance Web UI and re-run the scan on your target node.

Be sure to run only the base/ssh scan as shown on the next slide.



Slide 57

GL: Re-run the Compliance Scan

Run only the base/ssh scan.

Dashboard / Scan nodes

Target nodes: 1 host
ec2-54-210-76-104.compute-1.amazonaws.com

Patch Management

Compliance:

- ☐ base/apache
- ☐ base/linux
- ☐ base/mysql
- ☐ base/postgres
- ☒ base/ssh
- ☐ base/windows
- ☐ cis/cis-ubuntu-level1
- ☐ cis/cis-ubuntu-level2

Scan now Schedule

Slide 58


GL: Results of this Exercise

Your scan should show that the ssh protocol issue is now compliant.

base/ssh: Server: Specify a valid address family	Minor Issues	■
base/ssh: /etc/ssh should be a directory	Compliant	■
base/ssh: /etc/ssh should be owned by root	Compliant	■
base/ssh: sshd_config should be owned by root	Compliant	■
base/ssh: sshd_config should not be writable/executable to others	Compliant	■
base/ssh: Client: Set SSH protocol version to 2	Compliant	■
SSH Configuration Protocol should eq "2"	10.0	■
base/ssh: Server: Set protocol version to SSHv2	Compliant	■
base/ssh: Server: Do not permit root-based login	Compliant	■
base/ssh: sshd_config should not be group-writable/executable	Compliant	■
base/ssh: sshd_config should not be group-writable/executable	Compliant	■
base/ssh: Server: Disable challenge-response authentication	Compliant	■
base/ssh: sshd_config should not be accessible to others	Compliant	■


Slide 59

DISCUSSION



Conclusion

- ✓ Log in to your target node.
- ✓ Write a remediation recipe on that node.
- ✓ Test the recipe with Test Kitchen.
- ✓ Test for compliance with InSpec from the CLI
- ✓ Converge the recipe.
- ✓ Rescan the node and ensure compliance.

©2016 Chef Software Inc. 3-59 

In this module we scanned a node for compliance issues. We identified an issue and then wrote a remediation recipe directly on the node scanned. We also tested our recipe with test kitchen.

As mentioned previously, in a production environment, you will likely write such recipes locally, add them to the node's run list, and then upload them to Chef Server.

Then the nodes would download the recipes from Chef Server on their next chef-client run and also convergence the recipes.

Slide 60

Review Questions

1. When adding a node to the Compliance server's dashboard, should you use the node's FQDN or just its IP address?
2. What can `inspec exec` be used for?
3. How are compliance severities defined?
4. Using the image on the right, what section is the actual test?

```
control 'ssh-4' do
  impact 1.0
  title 'Client: Set SSH protocol version to 2'
  desc "
    Set the SSH protocol version to 2. Don't use legacy
    insecure SSHv1 connections anymore.
  "
  describe ssh_config do
    its('Protocol') { should eq('2') }
  end
end
```

Instructor Note Answers:

1. It doesn't matter...you could use the node's FQDN or just its IP address.
2. `inspec exec` can be used to test a compliance profile against remote hosts, including docker containers.
3. The `impact` value in a Compliance Profile/control defines severity. See slide 3-22 through slide 3-24 for examples.
4. The `describe` section is the actual test.

Slide 61

Review Questions

5. If a compliance scan tells you that a node is unreachable, what might you use to troubleshoot the connection?
6. What language is used to define controls?

Instructor Note Answers:

5. You could use the Dashboard's "check the connectivity" test, ssh into the target, and/or check the node's configuration in the Web UI (IP address, login credentials.)
6. InSpec.

Slide 62

