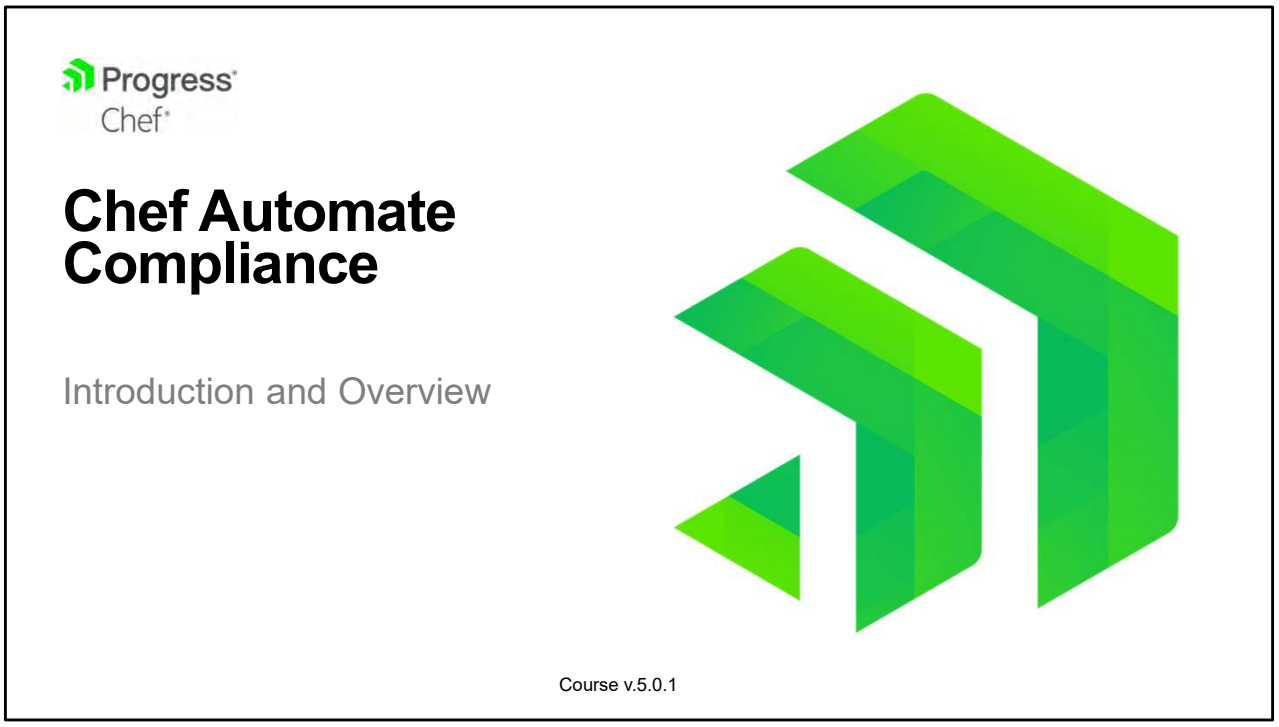
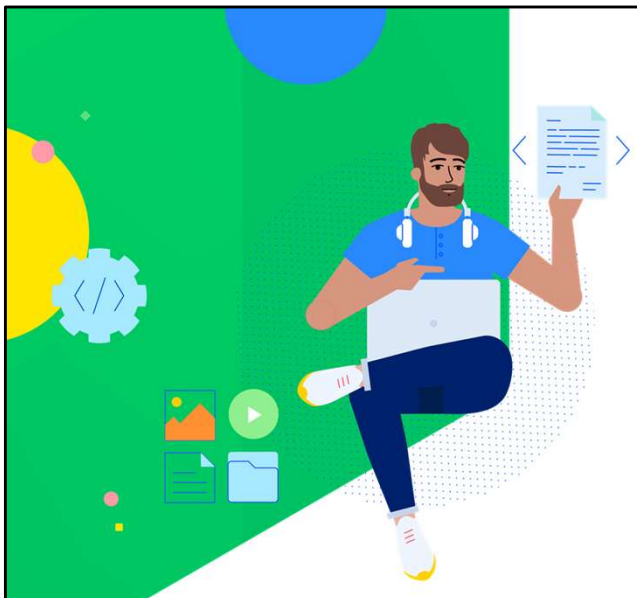



This course provides an understanding of the capabilities of Chef Automate. The first part of this course covers installation and administration then goes into using Automate to preform scans and remediate failures.

In addition, you will learn how to use InSpec to create and modify Chef Automate compliance profiles. Plus you'll learn how to use chef-client's compliance phase.

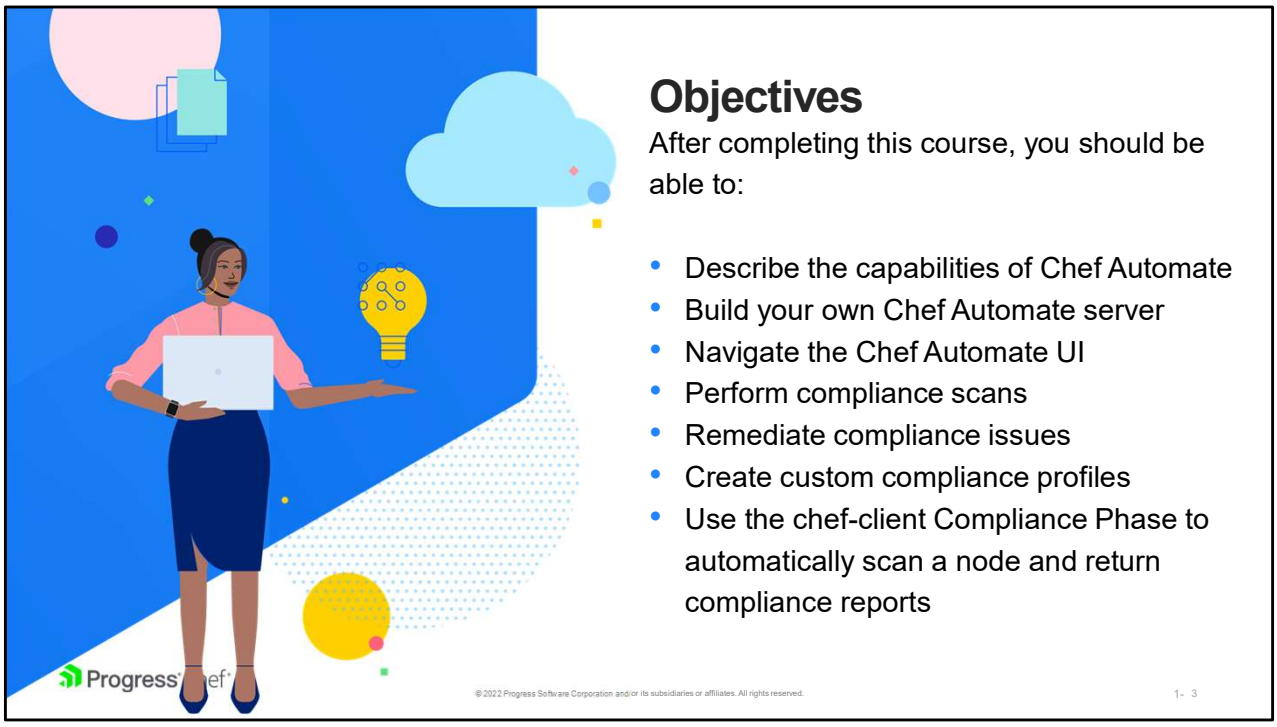
Instructor Note: Be sure to read Appendix Z at the end of this instructor guide for training lab set up notes and additional instructor notes.



## Introduce Yourselves

- Name
- Current job role
- Previous job roles/background
- Experience with Chef and/or config management

Progress Chef

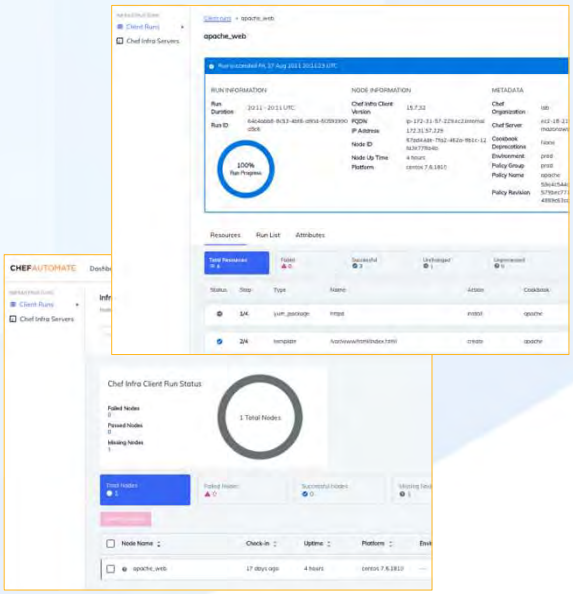



This course covers scanning and both Linux and Windows nodes.

# Concept

## Chef Automate Administration

We will begin by covering the different architecture models available, installing our own Automate server, and completing several post-installation tasks.





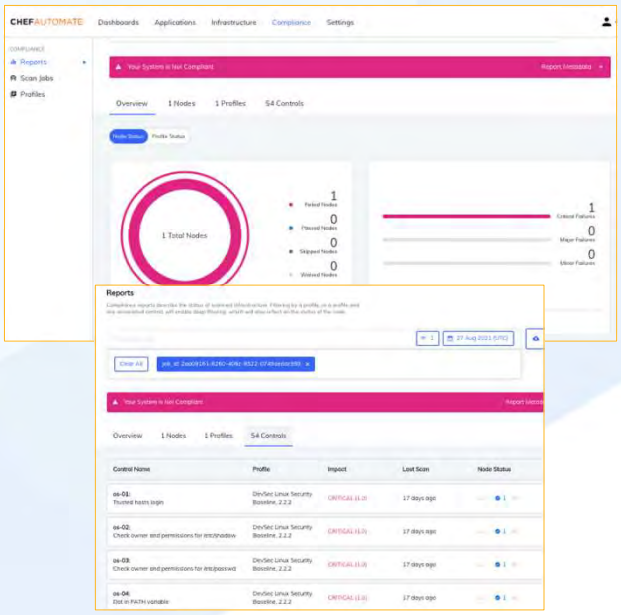



This course begins with installation, configuration, and administration of Chef Automate

# Concept

## Chef Automate UI

The Chef Automate UI is an enterprise dashboard and analytics tool enabling cross-team collaboration with actionable insights for configuration and compliance and an auditable history of changes to environments.



# Concept

## Chef Automate Compliance

Chef Automate allows you to assess your infrastructure's adherence to compliance requirements and to monitor that infrastructure on an ongoing basis. It includes:

- The Chef Automate server
- Prebuilt compliance profiles to help you get started quickly
- A language for writing audit controls, which includes audit resources that you can invoke

# Concept

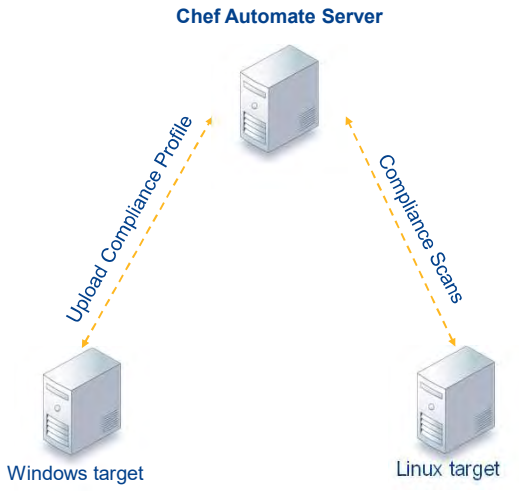
## Chef Automate Infrastructure

Chef Automate allows you to keep an eye on how your infrastructure management is doing. With all of your Chef Infra servers reporting to Automate, you get a single place to monitor for issues. A flexible filter gives you the ability to search your infrastructure for exactly what you are looking for. Run history is available at the node level so you can find out exactly when a failure appeared.

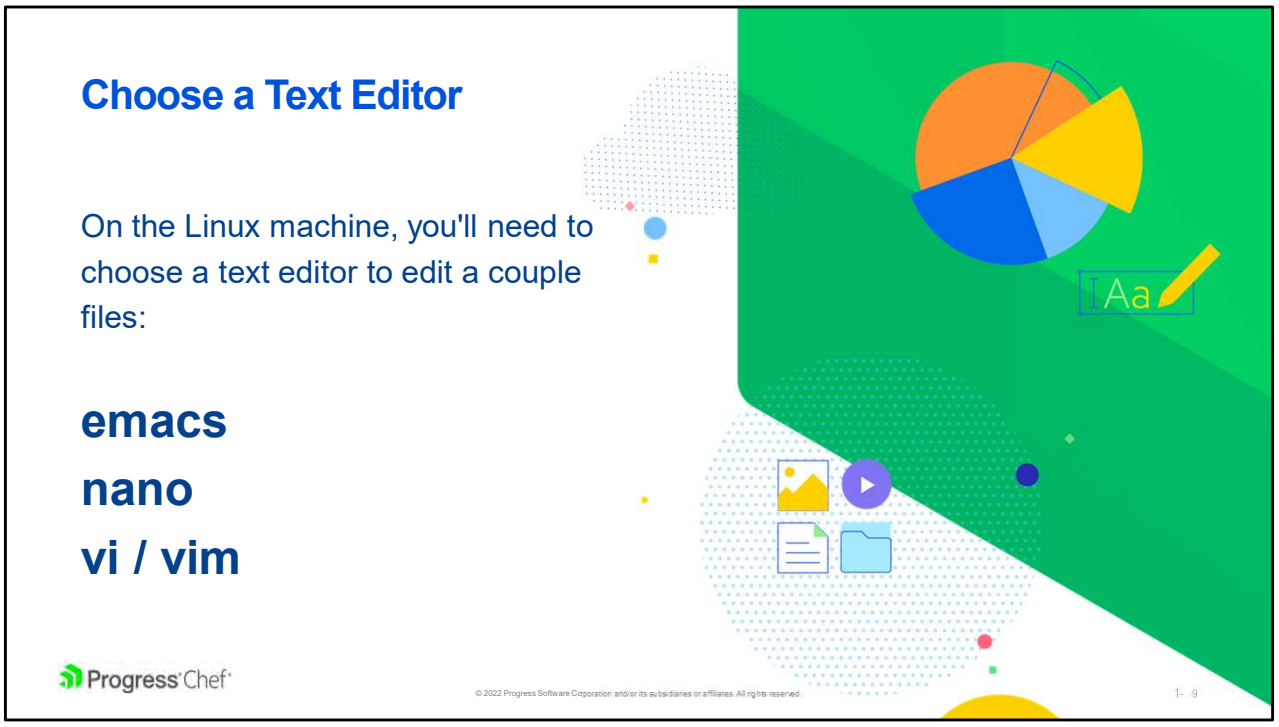
## Your Lab Environment

We will provide each of you with three AWS instances:

- A Linux instance where we will install and run Chef Automate
- A Linux instance to be used as a virtual workstation and as a scan target
- A Windows workstation that we use to create a custom compliance profile




Progress Chef




During this course we are going to use the text-based editors installed on these virtual workstations. There are at least three command-line editors that we can choose from on the Linux workstation: Emacs, Nano, or Vim.

**`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.)

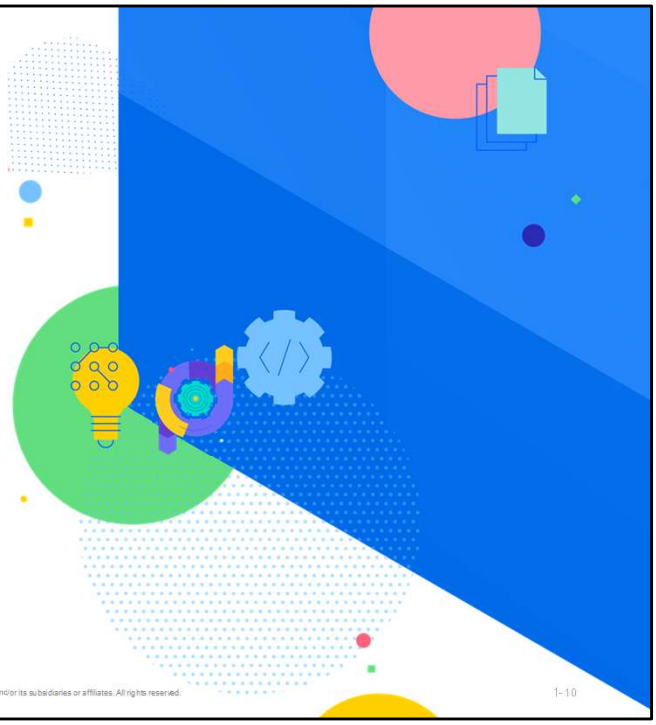
```
OPEN FILE    $ nano FILENAME
WRITE (When exiting) ctrl+x, y, ENTER
EXIT         ctrl+x
```

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

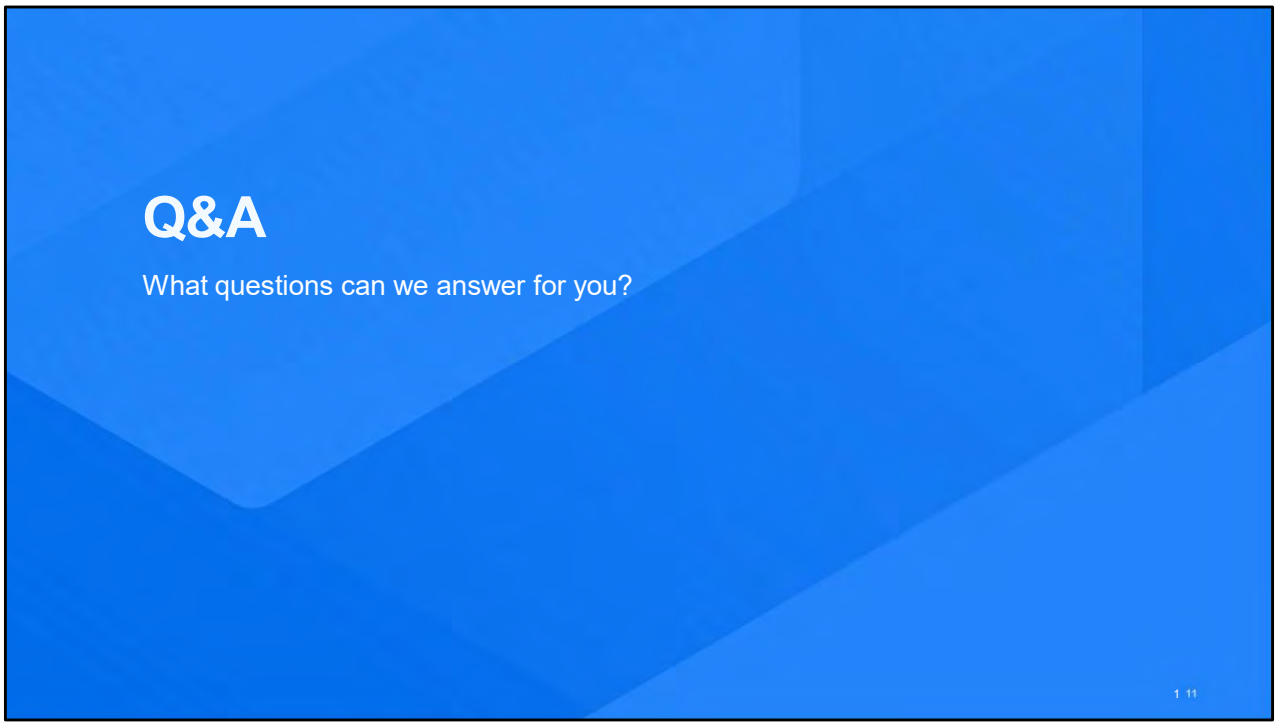
```
OPEN FILE    $ vim FILENAME
START EDITING            i
WRITE FILE   ESC, :w
EXIT         ESC, :q
EXIT (don't write)       ESC, :q!
```

## Hands-on Legend

- **GL** or **Group Lab**: All participants and the instructor do this task together with the instructor often leading the way.



Progress Chef

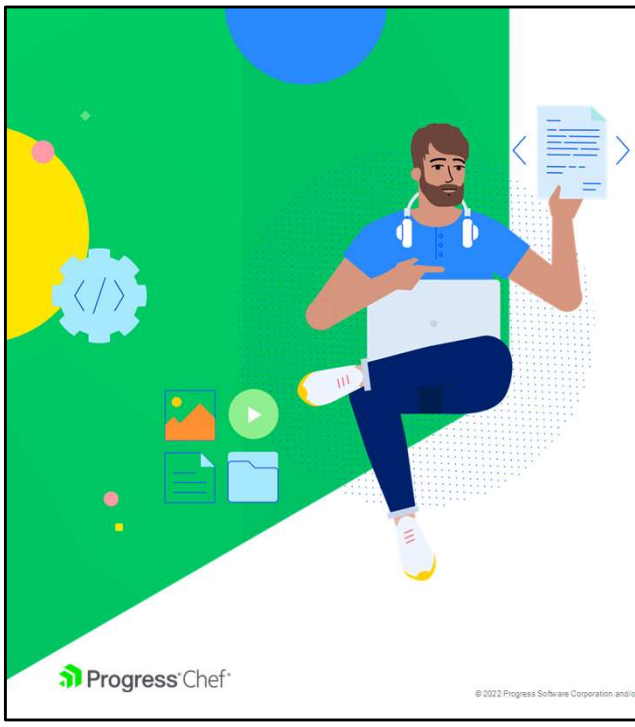

Q&A
What questions can we answer for you?
1 11


Progress Chef

Progress Chef

# Installing Chef Automate

Getting started with Automate



# Objectives

After completing this module, you should be able to:

- List the different architectures available
- Identify the tool used to manage your Chef Automate installation
- Install Chef Automate

Progress Chef

# Concept

## Chef Automate Architecture

Chef Automate can be setup in several configurations depending on your business and technical requirements. The different configurations are:

Single server – Automate and Infra installed on the same server

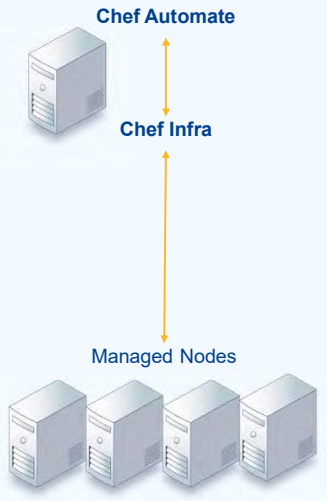
Multi-server – One or more Infra servers reporting to a one or more Automate servers

Airgapped – No inbound or outbound internet traffic

Automate HA – Highly available setup, more info in a later module

## Single Server

- Chef Infra and Chef Automate are installed on the same server
- Nodes check-in with Chef Infra
- Chef Infra reports results to Chef Automate
- Nodes can be configured to report results directly to Chef Automate



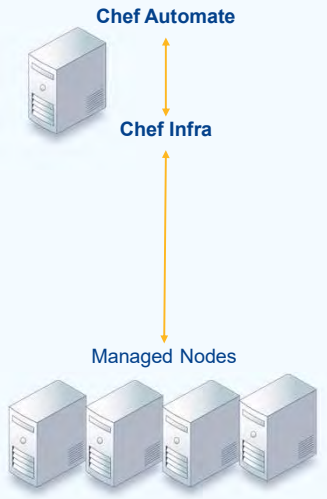

Progress



For more information on installing Chef Infra and Chef Automate on the same server visit: https://docs.chef.io/automate/infra_server/

## Single Server Minimum System Requirements

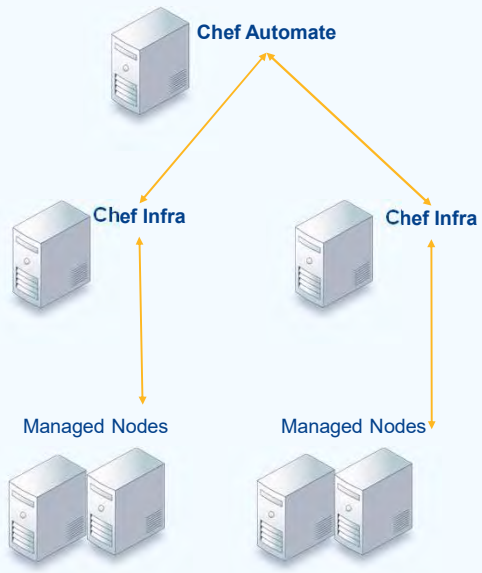
- Hardware
- up to 200 managed nodes: 8GB RAM, 2 vCPUs
- 200 - 500 managed nodes: 30GB RAM, 4 vCPUs
- 500 - 5000 managed nodes: 61GB RAM, 8vCPUs
- Disk Space: 80GB available to /hab plus 2MB per managed node
- Software
- Linux OS with kernel 3.2 or greater
- systemd as the init system
- useradd and curl or wget commands
- the install shell needs a max open file setting of greater than 65535




Progress

## Multi-Server

- Chef Infra and Chef Automate are installed on different servers
- Nodes check-in with Chef Infra
- Chef Infra reports results to Chef Automate
- Nodes can be configured to report results directly to Chef Automate



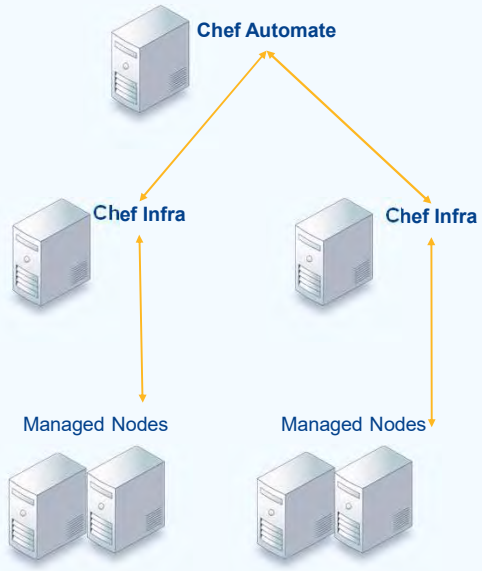

Progress



For more information on installing a stand-alone Chef Automate server visit: https://docs.chef.io/automate/install/

## Multi-Server Minimum System Requirements

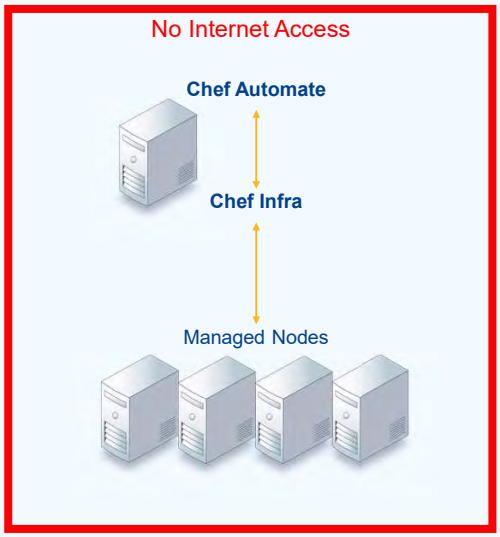
- Hardware
- RAM: 16 GB
- CPU: 4 vCPUs
- Disk Space: 80 GB available to /hab partition
- Software
- Linux OS with kernel 3.2 or greater
- system as the init system
- useradd and curl or wget commands
- The install shell needs a max open file setting of greater than 65535




Progress

## Airgapped

- Can be a single or multi-server setup
- Used when no connection to the internet is available
- Airgap Installation Bundle is created on a machine with an internet connection
- The AIB file is moved to air gapped server and used to install Chef






For more information on installing an airgapped version of Chef Automate visit: https://docs.chef.io/automate/airgapped_installation/

Concept

## Chef Automate Installation and Administration

No matter the architecture, the **chef-automate** CLI tool is used to install Chef Automate. It is also used for many other tasks such as starting and stopping Automate, upgrading the version of Automate, and creating backups of Automate. More information on the chef-automate tool is available at the link below.

https://docs.chef.io/automate/cli_chef_automate/





https://docs.chef.io/automate/cli_chef_automate/

EXERCISE

# GL: Installing Chef Automate

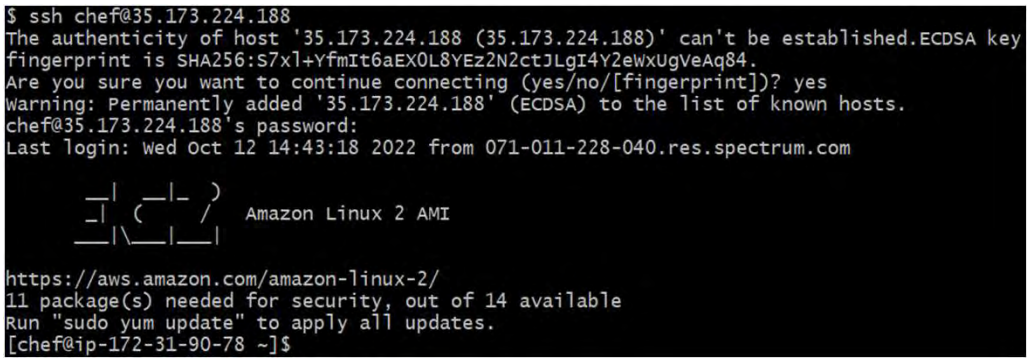
**Objective:**

- ❑ Download the chef-automate tool
- ❑ Create Airgap Installation bundle
- ❑ Create a default config file
- ❑ Update config file
- ❑ Run preflight check
- ❑ Install Chef Automate with Chef Infra

Progress Chef

## GL: Connecting to Lab Environment

To begin this lab ssh into the Linux machine designated as the Chef Automate server. You should have a list of IP addresses and the function they will serve in this course.




Progress Chef





Command:
ssh chef@IPADDRESS

Login Credentials:
Username: chef
Password: Cod3Can!

## GL: Download the chef-automate tool

After logging into the Linux machine that will become our Chef Automate server, run the following command. This will download the chef-automate tool, unpack it, and make it executable.


Progress Chef





Command:
curl https://packages.chef.io/files/current/latest/chef-automate-cli/chef-automate_linux_amd64.zip | gunzip - > chef-automate && chmod +x chef-automate

EXERCISE

# GL: Installing Chef Automate

**Objective:**

- ✓ Download the chef-automate tool
- ❑ Create Airgap Installation bundle
- ❑ Create a default config file
- ❑ Update config file
- ❑ Run preflight check
- ❑ Install Chef Automate with Chef Infra

Progress Chef

# Concept

## Locking in Chef Automate version

For this course we want to install a specific version of Automate. In order to do this we are going to create an airgap installation bundle that selects and locks in the version we want.

In a normal air gapped installation we would create the airgap installation bundle on a server with internet access and then move it to the secure environment. We are going to create the bundle and use it on the same server.

## GL: Create Airgap Installation bundle

Creating the airgap installation bundle is a single command, but it takes a few minutes to run.
When the command completes you should receive a message saying the .aib file was created.

```
[chef@ip-172-31-84-249 ~]$ ./chef-automate airgap bundle create --version 3.0.49

Creating Airgap Installation Bundle...
  Downloading core/hab/1.6.181/20201030172917 binary
  Downloaded core/hab/1.6.181/20201030172917 binary
  Downloading chef/mlsa/1.0.1/20210907212047
  Downloaded chef/mlsa/1.0.1/20210907212047
  Downloading chef-20160614114050
  Downloaded chef-20160614114050
  Downloading chef/automate-platform-tools/0.1.0/20220425101133
  Downloaded chef/automate-platform-tools/0.1.0/20220425101133
  Downloading core/busybox-static/1.33.0/20210826062032
  Downloaded core/busybox-static/1.33.0/20210826062032
  Downloading core-20210607174414
  Downloaded core/hab-launcher/14772/20201030181158
Success: Your Automate Install Bundle has been written to automate-3.0.49.aib. To install the
bundle on an Internet-disconnected server, copy the bundle and the chef-automate tool
to the server and run

chef-automate deploy --airgap-bundle </path/to/bundle>
```

Progress Chef





Command:
./chef-automate airgap bundle create --version 3.0.49

EXERCISE

# GL: Installing Chef Automate

**Objective:**

- ✓ Download the chef-automate tool
- ✓ Create Airgap Installation bundle
- ❑ Create a default config file
- ❑ Update config file
- ❑ Run preflight check
- ❑ Install Chef Automate with Chef Infra

Progress Chef

Concept

## Chef Automate Configuration file

Setting options for the Chef Automate install is done with a .toml file. The chef-automate tool can create a default .toml file for you that has all the fields needed for Automate to run. More information on available options is available at the link below.

https://docs.chef.io/automate/configuration/





https://docs.chef.io/automate/configuration/

## GL: Create a default config file

Using the chef-automate tool, we can create a default config file to get us started. The config.toml file that is created is very simple and contains all the fields required for a basic Chef Automate installation.

```
[chef@ip-172-31-91-83 ~]$ sudo ./chef-automate init-config
Success: Config written to config.toml
Automate Load Balancer fqdn set to [ip-172-31-91-83.ec2.internal]
When Automate is deployed you will access https://ip-172-31-91-83.ec2.internal to see the
dashboard.
If this is not a routable address please update the fqdn appropriately before deploying.
```

Progress Chef





Command:
sudo ./chef-automate init-config

EXERCISE

# GL: Installing Chef Automate

**Objective:**

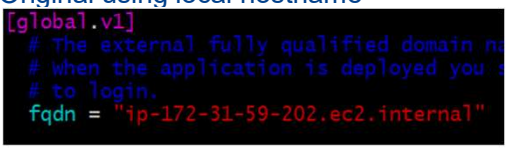
- ✓ Download the chef-automate tool
- ✓ Create Airgap Installation bundle
- ✓ Create a default config file
- ❑ Update config file
- ❑ Run preflight check
- ❑ Install Chef Automate with Chef Infra

Progress Chef

## GL: Update config file

Our default config file gives us a starting point, but is not ready to use yet. First, we need to update the FQDN to the publicly available hostname. Your values will be different, but the example below is a guide.

Original using local hostname




Updated using public hostname


Progress Chef





Command:
sudo vi config.toml

## GL: Update config file

Next, we need to update [deployment.v1.svc] to tell chef-automate that we want to install Chef Automate and Chef Infra on this server. Add **products** after **deployment_type** as shown.

Original

```
# Deployment service configuration.
[deployment.v1]
  [deployment.v1.svc]
    # Habitat channel to install hartifact from.
    # Can be 'dev', 'current', or 'acceptance'
    channel = "current"
    upgrade_strategy = "at-once"
    deployment_type = "local"
```

Updated

```
# Deployment service configuration.
[deployment.v1]
  [deployment.v1.svc]
    # Habitat channel to install hartifact from.
    # Can be 'dev', 'current', or 'acceptance'
    channel = "current"
    upgrade_strategy = "at-once"
    deployment_type = "local"
    products = ["automate", "infra-server"]
```

Progress Chef





Code:
products = ["automate", "infra-server"]

EXERCISE

# GL: Installing Chef Automate

**Objective:**

- ✓ Download the chef-automate tool
- ✓ Create Airgap Installation bundle
- ✓ Create a default config file
- ✓ Update config file
- ❑ Run preflight check
- ❑ Install Chef Automate with Chef Infra

Progress Chef

## GL: Install Chef Automate with Chef Infra

Before we start the installation, we should make sure that everything is ready. The **chef-automate preflight-check** command will do this for us. In this example there was an error and we are given a possible solution.

```
[chef@ip-172-31-91-83 ~]$ sudo ./chef-automate preflight-check

Beginning pre-flight checks
PreflightError: One or more preflight checks failed:
 OK | running as root
 OK | volume: has 11.8GB avail (need 5.0GB for installation)
 OK | SELinux is not enabled
 OK | chef-automate CLI is not in /bin
 OK | automate not already deployed
 OK | initial required ports are available
 OK | init system is systemd
 OK | found required command "useradd"
 OK | user "nobody" exists
 OK | MemTotal 8062756 kB (8.1GB) is at least 2000000 kB (2.0GB)
 OK | fs.file-max=802334 is at least 64000
FAIL| vm.max_map_count=65530 must be at least 262144
 OK | vm.dirty_ratio=20 is between 5 and 30
 OK | vm.dirty_background_ratio=10 is between 10 and 60
FAIL| vm.dirty_expire_centisecs=3000 is not between 10000 and 30000
 OK | kernel version "4.14" is at least "3.2"
 OK | https://licensing.chef.io/status is reachable
 OK | https://bldr.habitat.sh is reachable
 OK | https://raw.githubusercontent.com is reachable
 OK | https://packages.chef.io is reachable
 OK | https://github.com is reachable
 OK | https://downloads.chef.io is reachable

Fix the system tuning failures indicated above by running the following:
sysctl -w vm.max_map_count=262144
sysctl -w vm.dirty_expire_centisecs=20000

To make these changes permanent, add the following to /etc/sysctl.conf:
vm.max_map_count=262144
vm.dirty_expire_centisecs=20000
```

Progress



Command:
sudo ./chef-automate preflight-check

## GL: Install Chef Automate with Chef Infra

Let's run the commands given and see if our preflight check will pass.

```
[chef@ip-172-31-91-83 ~]$ sudo sysctl -w vm.max_map_count=262144
vm.max_map_count = 262144
[chef@ip-172-31-91-83 ~]$
[chef@ip-172-31-91-83 ~]$ sudo sysctl -w vm.dirty_expire_centisecs=20000
vm.dirty_expire_centisecs = 20000
```

Progress Chef





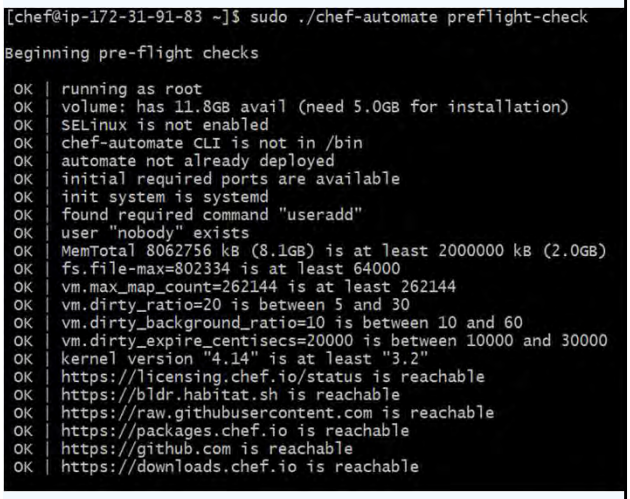
Note: Running the commands like this does not make the change permanent. In order to have these changes survive a system reboot, update the values in /etc/sysctl.conf as given by the previous command.

Commands:
sudo sysctl -w vm.max_map_count=262144

sudo sysctl -w vm.dirty_expire_centisecs=20000

## GL: Install Chef Automate with Chef Infra

Now we can run the preflight check again. Running the commands given to us allowed the preflight check to pass. We are now ready to install Chef Automate and Chef Infra.






Command:
sudo ./chef-automate preflight-check

EXERCISE

# GL: Installing Chef Automate

**Objective:**

- ✓ Download the chef-automate tool
- ✓ Create Airgap Installation bundle
- ✓ Create a default config file
- ✓ Update config file
- ✓ Run preflight check
- ❑ Install Chef Automate with Chef Infra

Progress Chef

## GL: Install Chef Automate with Chef Infra

We are going to use the **chef-automate deploy** for the installation of Chef Automate and Chef Infra. This process will take several minutes because it installs several components.


Progress Chef





Command:
sudo ./chef-automate deploy config.toml --airgap-bundle automate-3.0.49.aib --accept-terms-and-mlsa

## GL: Install Chef Automate with Chef Infra

The final steps of the install are shown below. All Chef services are restarted and checked to ensure they come up cleanly, then an admin user is created. The credentials for the admin user are stored in the file given.

```
Checking service health

Creating admin user

Deploy Complete
Your credentials have been saved to automate-credentials.toml
Access the web UI at https://ec2-44-201-73-241.compute-1.amazonaws.com/

Users of this Automate deployment may elect to share user-anonymized usage data with
Chef Software, Inc. Chef uses this shared data to improve Automate.
Please visit https://chef.io/privacy-policy for more information about the
information Chef collects, and how that information is used.
```

Progress Chef

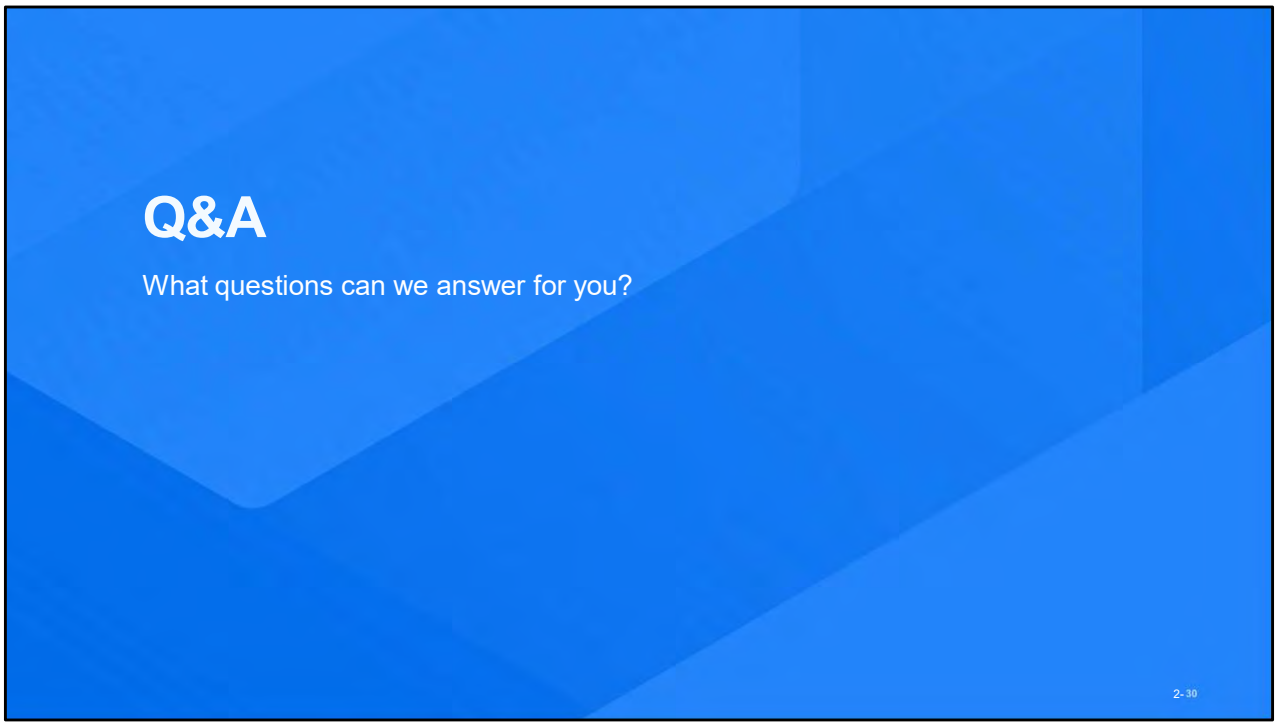
EXERCISE

# GL: Installing Chef Automate

**Objective:**

- ✓ Download the chef-automate tool
- ✓ Create Airgap Installation bundle
- ✓ Create a default config file
- ✓ Update config file
- ✓ Run preflight check
- ✓ Install Chef Automate with Chef Infra

Progress Chef

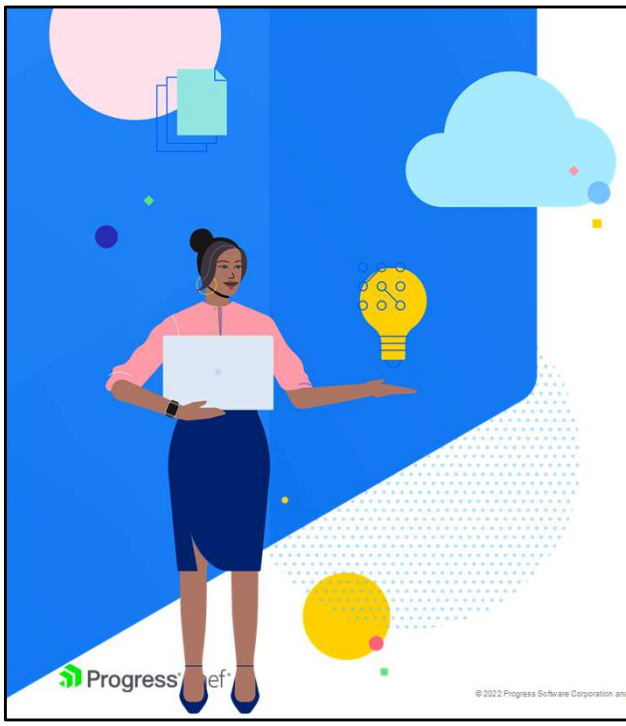

Q&A
What questions can we answer for you?
2-30


Progress Chef

Progress Chef

# Configuring Chef Automate

Preparing to use Automate



## Objectives

After completing this module, you should be able to:

- Apply additional settings to Chef Automate
- Create a Chef Infra user
- Create a Chef Infra org
- Connect Chef Automate and Chef Infra

# Concept

## Configuring Chef Automate

We have a fresh installation of Chef Automate and Chef Infra, now we need to configure them. A few tasks are required such as:

- Review available configuration options
- Create Chef Infra organization
- Create Chef Infra user
- Associate the Chef Infra user with the organization
- Connect Chef Automate and Chef Infra

Concept

## Chef Automate Configuration Options

In the previous module we created a simple config.toml file and made two small updates to it. Chef Automate has many more options that can be configured. The link below describes the available settings and gives examples of how to apply them.

https://docs.chef.io/automate/configuration/





https://docs.chef.io/automate/configuration/

# Concept

## Chef Automate Single Configuration File

When you have a simple setup, all of your configuration can be stored in a single file. Changes are applied to your system by updating your config.toml and running the command below.

- **chef-automate config set /path/to/config.toml** - replaces the current Chef Automate configuration with the provided configuration and applies any changes.

# Concept

## Chef Automate Multiple Configuration Files

As your config.toml file grows, breaking it into smaller parts might be helpful. You can create any file structure that works for your situation and use the commands below to apply them, e.g. config.toml, opensearch.toml, postgresql.toml

- **chef-automate config patch /path/to/partial-config.toml** - updates an existing Chef Automate configuration by merging the contents of your file with your current Chef Automate configuration, and applying any changes

EXERCISE

# Group Lab: Update Chef Automate Configuration

**Objective:**

- ❑ Find options
- ❑ Update config.toml
- ❑ Apply new settings

Progress Chef

## GL: Find Options

There are many options available for Chef Automate. The link at the bottom of the slide can give you an idea of what is available. For this exercise we are going to apply three changes.

- We want to upgrade Chef Automate manually so we will set the upgrade strategy to none. How to do that is covered here: https://docs.chef.io/automate/configuration/#upgrade-strategy
- An appropriate security warning needs to be added to our login page. How to do that is covered here: https://docs.chef.io/automate/disclosure_panel_and_banner/
- We need to create a new SSL certificate for the external hostname of our server

Progress Chef

https://docs.chef.io/automate/configuration/





Note: Setting the upgrade strategy is not technically needed because we have an airgapped installation. It does give us more practice editing the config.toml file and it is a commonly used setting.

https://docs.chef.io/automate/configuration/#upgrade-strategy
https://docs.chef.io/automate/disclosure_panel_and_banner/
https://docs.chef.io/automate/configuration/

EXERCISE

# Group Lab: Update Chef Automate Configuration

**Objective:**

- ✓ Find options
- ❑ Update config.toml
- ❑ Apply new settings

Progress Chef

## GL: Update config.toml

The first change is to the value of upgrade_strategy. In order to control when updates take place, set the value to “none”. After updating the file, save and close it.

```
# Deployment service configuration.
[deployment.v1]
  [deployment.v1.svc]
    # Habitat channel to install hartifact from.
    # Can be 'dev', 'current', or 'acceptance'
    channel = "current"
    upgrade_strategy = "none"
    deployment_type = "local"
    products = ["automate", "infra-server"]
```

Progress Chef

## GL: Update config.toml

Next we will create the text file that will become the disclosure panel for our login page. The message added to the disclosure panel can be plain text or HTML file. The disclosure panel can be any warning or notice you like. A simple example is shown below.

```
WARNING: This is the only Automate server, don't break it.
```

Progress Chef





Command:
vi disclosure-panel-message.txt

Code:
WARNING: This is the only Automate server, don't break it.

## GL: Update config.toml

Now we can add our new warning or notice to our config.toml. The disclosure goes into the global.v1.disclosure section. This can be added directly below the FQDN. After updating the file, save and close it.

```
[global.v1]
  # The external fully qualified domain name.
  # When the application is deployed you should be able to access 'https://<fqdn>/'
  # to login.
  fqdn = "ec2-44-211-35-224.compute-1.amazonaws.com"

  [global.v1.disclosure]
    show = true
    message_file_path = "/home/chef/disclosure-panel-message.txt"
```

Progress Chef





Code:

```
  [global.v1.disclosure]
    show = true
    message_file_path = "/home/chef/disclosure-panel-message.txt"
```

## GL: Update config.toml

When we ran the **chef-automate deploy** command a certificate was generated for us, but it used the internal hostname. We need to create a new certificate that uses the external URL and place it in our config.toml file. Creating a new certificate requires several inputs, but for our exercise only the Common Name is important. The Common Name must be the external DNS name of your Chef Automate server. An example is given below.

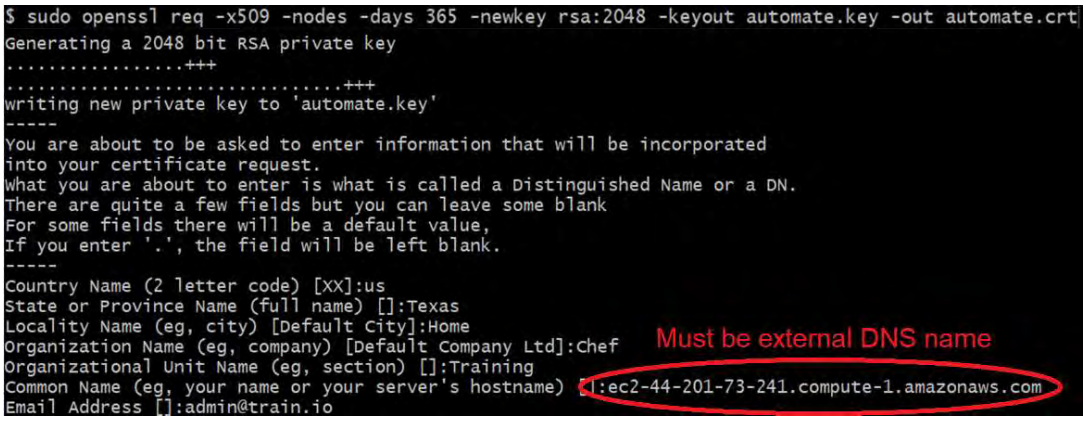
**URL:** https://ec2-44-201-73-241.compute-1.amazonaws.com

**Common Name:** ec2-44-201-73-241.compute-1.amazonaws.com

Progress Chef

## GL: Update config.toml

The command shown will create a new private key and public certificate. You will be given several prompts, The only important one is **Common Name**.




Progress Chef





Note: The hostname can be verified by running ‘curl http://169.254.169.254/latest/meta-data/public-hostname’

Command:
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout automate.key -out automate.crt

## GL: Update config.toml

Now that we have our new public cert (automate.crt) and private key (automate.key) we can add them to our config.toml. The contents of automate.crt are placed in the cert field and the contents of automate.key are placed in the key field.

```
  [[global.v1.frontend_tls]]
    # The TLS certificate for the load balancer frontend.
    cert = """-----BEGIN CERTIFICATE-----
MIIEFzCCAv+gAwIBAgIJANFkY/nITg7oMA0GCSqGSIb3DQEBCwUAMIGhMQswCQYD
VQQGEwJlczEOMAwGA1UECAwFVGV4YXMxDTALBgNVBAcMBEhvbWUxDTALBgNVBAoM
```

```
    # The TLS RSA key for the load balancer frontend.
    key = """-----BEGIN PRIVATE KEY-----
MIIEvwIBADANBgkqhkiG9w0BAQEFAASCBKkwggSlAgEAAoIBAQDPQhqKqRy3xAET
1oXnEjQCKbwPQUK3UcXxMqu1knaJ6Bm42u3IviQjjqR8rQnEBsHtnEngMTPKiZQn
```





Note: The automate.crt and automate.key files will be in whatever directory you executed the command on the previous slide.

EXERCISE

# Group Lab: Update Chef Automate Configuration

**Objective:**

- ✓ Find options
- ✓ Update config.toml
- ❑ Apply new settings

Progress Chef

## GL: Apply New Settings

Our configuration is currently in a single file, this means we can use either of the commands we learned earlier. Apply your updates using the **set** or **patch** commands. After running the command successfully, we can see the warning on our login page.

```
[chef@ip-172-31-91-83 ~]$ sudo ./chef-automate config set config.toml

Setting deployment configuration

Applying deployment configuration
  Started automate-dex
  Started event-gateway
  Started automate-load-balancer
Success: Configuration set
```

Progress Chef





Note: Ensure that you are typing ‘./chef-automate ’ and not ‘chef-automate ’. The second command will run but give an error, ‘./’ specifies the chef-automate tool in the current directory

Command:
sudo ./chef-automate config set config.toml

## GL: Apply New Settings

After running the command successfully, we can see the warning on our login page.


Progress Chef

EXERCISE

# Group Lab: Update Chef Automate Configuration

**Objective:**

- ✓ Find options
- ✓ Update config.toml
- ✓ Apply new settings

Progress Chef

Concept

## Chef Infra Server Administration

Chef Automate is the focus of this course, but Chef Infra Server is required for some features we will use later. We will setup chef-client Compliance Phase in a later module which allows compliance scans to be run along with the scheduled chef-client runs. To accomplish these tasks we need a user account and an organization on the Chef Infra Server.

Concept

## Chef Infra Server Access Control

The Chef Infra Server uses role-based access control (RBAC) to restrict access to objects—nodes, policyfiles, cookbooks, and so on. Each Chef Infra Server can host many organizations, each organization can have one or more groups, each group can have one or more users. More information on access controls is available at the link below.

https://docs.chef.io/server/server_orgs/





https://docs.chef.io/server/server_orgs/

EXERCISE

# Group Lab: Configure Chef Infra Server

**Objective:**

- ❑ Create user
- ❑ Create organization

Progress Chef

# Concept

## chef-server-ctl command

Administration of the Chef Infra Server is done with the **chef-server-ctl** command. Given below is a sample of the tasks this command can do. More information is available at the link below.

- Create Chef Infra backups
- Gather log files
- Preform maintenance tasks
- Create users
- Manage organizations

https://docs.chef.io/server/ctl_chef_server/

## GL: Create User

Before we create any users, we should look at what users currently exist. It is important to ensure you know the state of your system before and after you make a change. The command show below tells us that the only user on our Chef Infra server is the default admin account named pivotal.

```
[chef@ip-172-31-91-83 ~]$ sudo chef-server-ctl user-list
pivotal
[chef@ip-172-31-91-83 ~]$
```

https://docs.chef.io/server/ctl_chef_server/#user-list

Progress Chef





https://docs.chef.io/server/ctl_chef_server/#user-list

Command:
sudo chef-server-ctl user-list

## GL: Create User

To create a Chef Infra user we need a username, first name, last name, and a password. In order to create a key that we can give the user, we are going to use the -f option and specify a file name. Without this option the users key will be displayed in the terminal and not saved. When successful, there is no output for this command.

**chef-server-ctl user-create USER_NAME FIRST_NAME LAST_NAME EMAIL PASSWORD -f FILE_NAME**

```
[chef@ip-172-31-91-83 ~]$ sudo chef-server-ctl user-create atrain Automate Training atrain@training.io
Password1 -f atrain.key
[chef@ip-172-31-91-83 ~]$
```

https://docs.chef.io/server/ctl_chef_server/#user-create

Progress Chef





https://docs.chef.io/server/ctl_chef_server/#user-create

Command:
sudo chef-server-ctl user-create atrain Automate Training atrain@training.io Password1 -f atrain.key

## GL: Create User

After creating our user we should verify that everything worked as expected. We can see the atrain user now as well as the .key file the command created.

```
[chef@ip-172-31-91-83 ~]$ sudo chef-server-ctl user-list
atrain
pivotal
```

```
[chef@ip-172-31-91-83 ~]$ ls -l atrain*
-rw-r--r-- 1 root root 1674 Oct 12 21:19 atrain.key
```

Progress Chef





Commands:
sudo chef-server-ctl user-list
ls –l atrain*

EXERCISE

# Group Lab: Configure Chef Infra Server

**Objective:**

- ✓ Create user
- ❑ Create organization

Progress Chef

## GL: Create Organization

As before, we should verify the state of our system before and after a change is made. We can check for existing organizations with the command shown below. As this is a fresh install, we should not have anything returned.

```
[chef@ip-172-31-91-83 ~]$ sudo chef-server-ctl org-list

[chef@ip-172-31-91-83 ~]$
```

https://docs.chef.io/server/ctl_chef_server/#org-list

Progress Chef



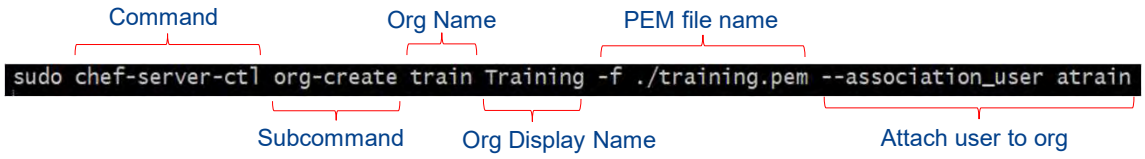


https://docs.chef.io/server/ctl_chef_server/#org-list

Command:
sudo chef-server-ctl org-list

## GL: Create Organization

Now that we have a user, we need an organization to store our nodes, cookbooks, and policyfiles. The only required information for this command is the org name and the org display name, but we are going to add two options to send the org key to a file and associate our user to the org. Similar to when we created a user, this command has no output when it is successful.




https://docs.chef.io/server/ctl_chef_server/#org-create

Progress Chef





https://docs.chef.io/server/ctl_chef_server/#org-create

Command:
sudo chef-server-ctl org-create train Training -f ./training.pem --association_user atrain

## GL: Create Organization

After creating the organization we should verify that everything worked as expected. We can see the train org now as well as the .pem file the command created.

```
[chef@ip-172-31-91-83 ~]$ sudo chef-server-ctl org-list
train
```

```
[chef@ip-172-31-91-83 ~]$ ls -l training*
-rw-r--r-- 1 root root 1674 Oct 12 21:24 training.pem
```

Progress Chef





Commands:
sudo chef-server-ctl org-list
ls –l training*

EXERCISE

# Group Lab: Configure Chef Infra Server

**Objective:**

- ✓ Create user
- ✓ Create organization

Progress Chef

Concept

## Server to Server Communication

Even though Chef Automate and Chef Infra Server are installed on the same machine, they do not know about each other. The connection between the servers must be setup manually. The connection between the servers is setup inside of Chef Automate.

EXERCISE

# Group Lab: Connect Chef Automate to Chef Infra Server

**Objective:**

- ❑ Login to Chef Automate
- ❑ Add Chef Infra Server

Progress Chef

## GL: Login to Chef Automate

To login to Chef Automate we are going to need the credentials from the automate-credentials.toml file that was created during Installation. These credentials are the only way to login until we create a user account or setup LDAP or SAML authentication. To begin, navigate to the URL in your .toml file and click **Sign in as a local user**.


Progress Chef





Command:
sudo cat automate-credentials.toml

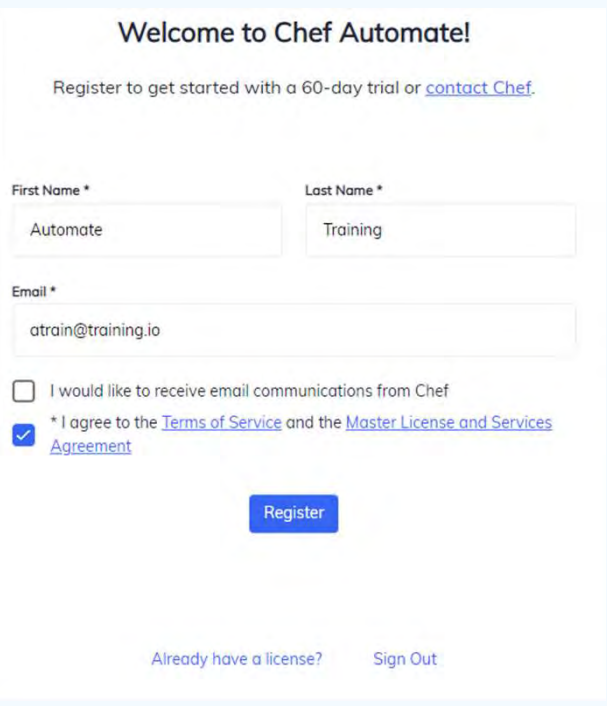
## GL: Login to Chef Automate

Now we can enter the credentials from our .toml file and click **Sign In**.

## GL: Login to Chef Automate

Once we sign in we are greeted with a form that gives us 60 days to try Chef Automate. Enter the info to the right and click **Register**.

If you already have a license for Chef Automate, you can enter it after clicking **Already have a license?** at the bottom.



EXERCISE

# Group Lab: Connect Chef Automate to Chef Infra Server

**Objective:**

- ✓ Login to Chef Automate
- ❑ Add Chef Infra Server

Progress Chef

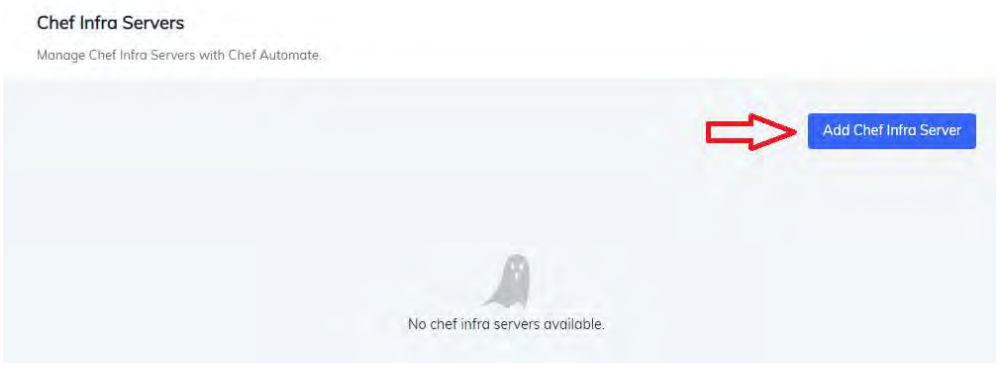
## GL: Add Chef Infra Server

Click on **Infrastructure** in the top menu, then click on **Chef Infra Servers** on the left-side menu.


Progress Chef

# GL: Add Chef Infra Server

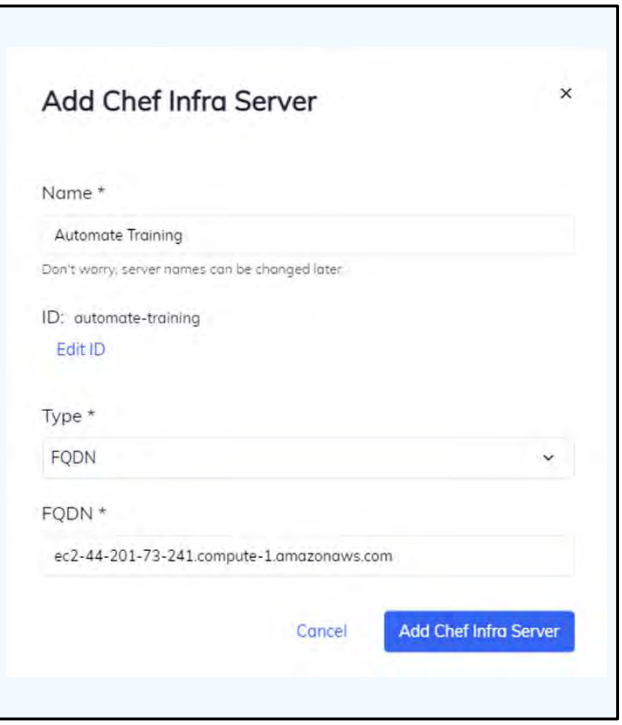
Now click on the **Add Chef Infra Server** button on the right-hand side of the screen.



## GL: Add Chef Infra Server

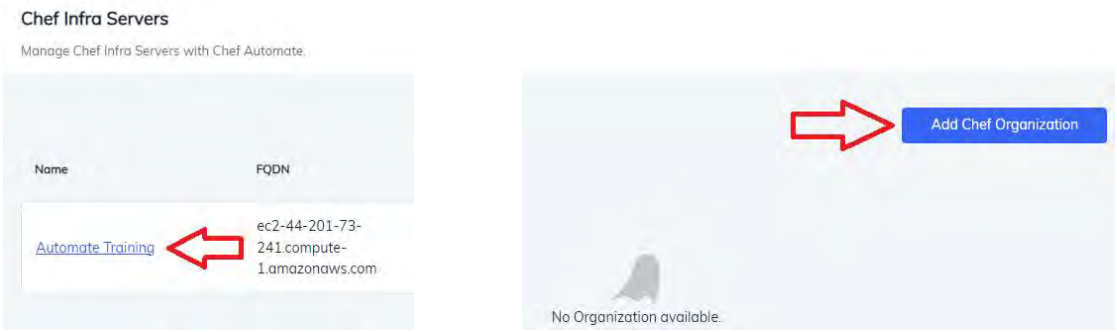
A pop-up window will appear that allows you give the Chef Infra Server a name and how to connect to it. We can connect by the FQDN or by the IP address.

The FQDN is in the URL bar because Chef Infra and Chef Automate are on the same machine.



## GL: Add Chef Infra Server

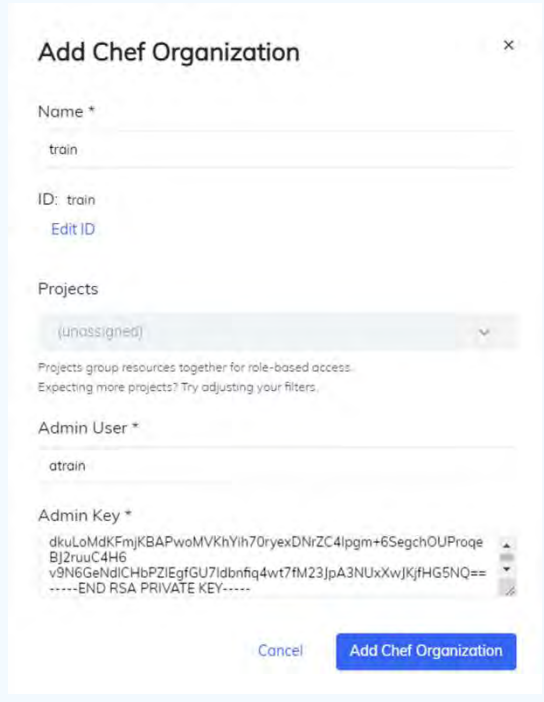
Our Chef Infra Server is now available, but the connection is not complete. We need to add the train organization from the Chef Infra Server. From the Chef Infra Servers page, click on our server, then click on the **Add Chef Organization** button.



## GL: Add Chef Infra Server

A pop-up window will appear. Enter the name of the org we created, the admin user, and the admin key.

The admin key is in the file **atrain.key** and was created when we created the atrain user.

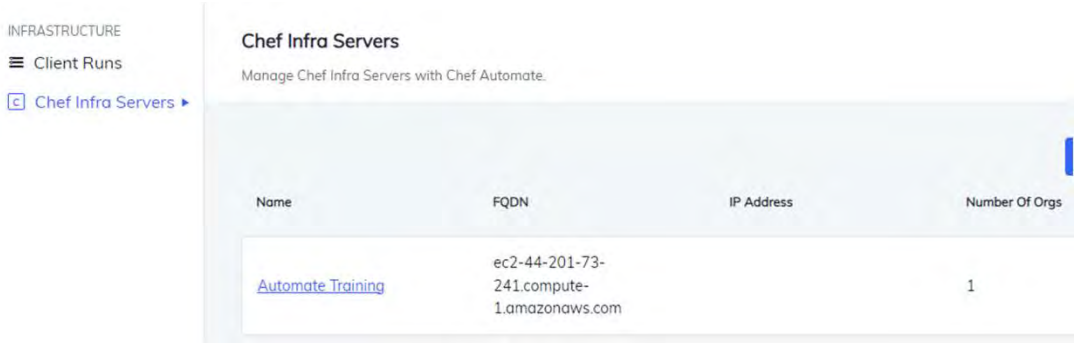





Command:
sudo cat atrain.key

## GL: Add Chef Infra Server

The connection between Chef Automate and Chef Infra Server is now complete. When chef-client runs on nodes bootstrapped to this Chef Infra Server, the results will be available in the Client Runs menu.




Progress Chef

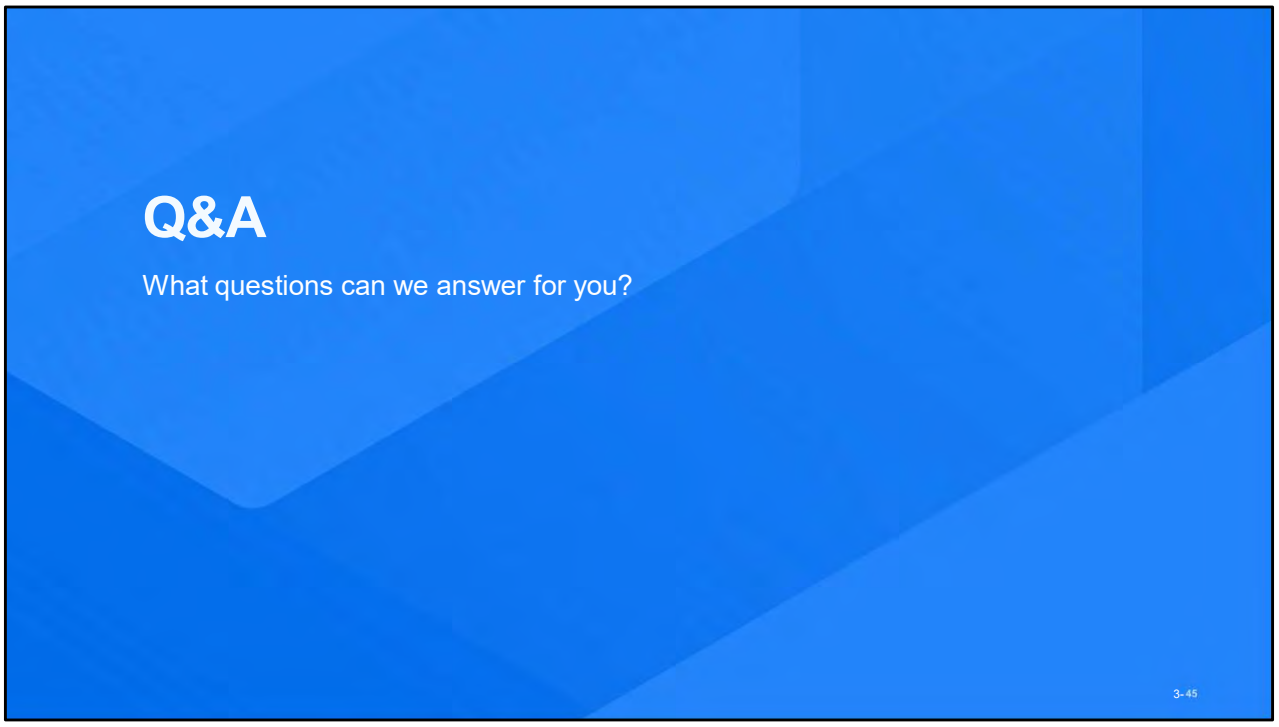
EXERCISE

# Group Lab: Connect Chef Automate to Chef Infra Server

**Objective:**

- ✓ Login to Chef Automate
- ✓ Add Chef Infra Server

Progress Chef


Q&A
What questions can we answer for you?
3-45


Progress Chef

Progress Chef

# Using the Chef Automate UI

The Chef Automate UI

# Objectives

After completing this module, you should be able to:

- Log into the Chef Automate UI
- Identify the functions available in each section
- Explain where to find Chef Automate documentation

Progress Chef

# Concept

## The Chef Automate UI

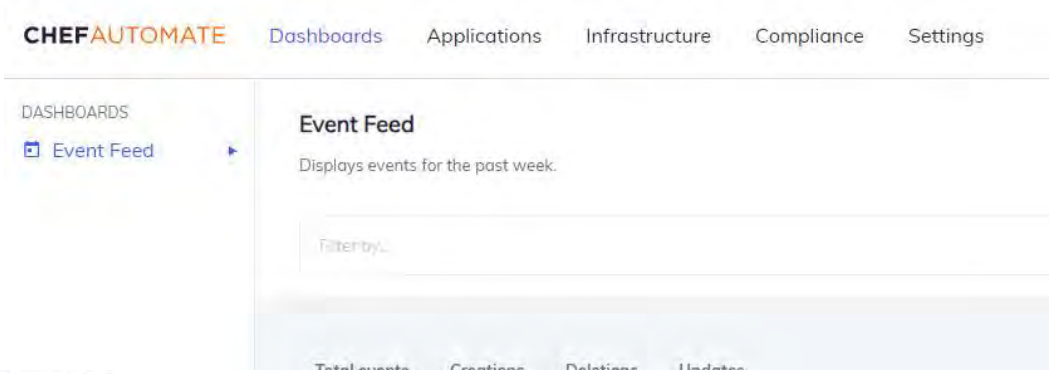
The Chef Automate UI allows you to monitor and visualize node status and convergence events (chef-client runs) from any servers or nodes that you are managing.

In addition to converge data, the UI also provides information on the compliance state of your environments. It also includes a scanner that you can use to run compliance scans.

The compliance features are what we will mostly focus on today in this course.

## Chef Automate UI Navigation

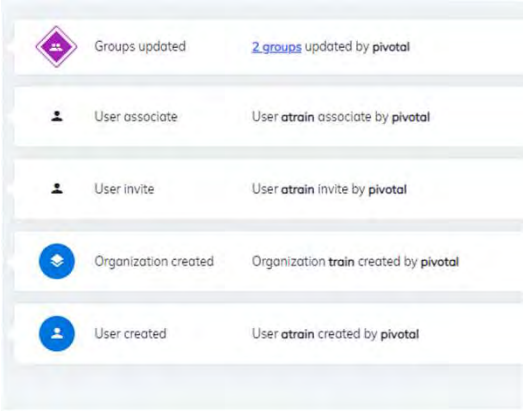
Chef Automate has five sections that are accessible from the menu at the top of the screen. Each section has its own menu on the left-hand side of the screen.




Progress Chef

## Chef Automate Dashboards

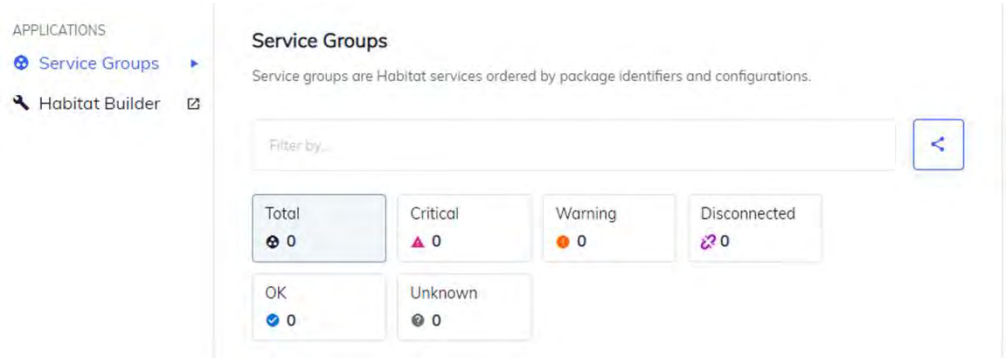
The **Dashboards** section gives you a running list of all the actions that have been taken in your environment. You can see that we have created the atrain user and the train organization here.




Progress Chef

# Chef Automate Applications

If you are using Chef Habitat in your environment, the **Applications** section is where you can interact with it. This is currently a placeholder for future functionality.




Progress Chef

## Chef Automate Infrastructure - Client Runs

- Shows the status of chef-client runs for all the nodes on your Chef Infra Servers
- A filter at the top gives you many ways to narrow down the list of nodes when you are searching for a certain node or group of nodes
- The results of chef-client runs can be downloaded in json or csv formats
- Quick filters are available for failed nodes, successful nodes, and missing nodes
- Clicking on a node gives more details about the node and the chef-client run history for the node

We will spend more time on this window during the course.

Progress Chef

## Chef Automate Infrastructure - Chef Infra Servers

- Used for adding and removing Chef Infra Servers
- Clicking on the Chef Infra Server gives you a list of the orgs on that server
- Clicking on the org gives you important items in that org such as Cookbooks, Policyfiles, Policy Groups, and Nodes

We will visit this window again during this course, but only for verifying our actions have worked as expected.

Progress Chef

## Chef Automate Compliance

The **Compliance** section is where you can setup and monitor compliance scans in your environment. Many of the future modules will be focused on this section. It provides us with the following three windows:

**Reports** – Search for and review the results of compliance scans

**Scan Jobs** – Create and run ad-hoc or scheduled compliance scans

**Profiles** – Upload custom compliance profiles or pick from hundreds of available profiles

Progress Chef

## Chef Automate Settings

The **Settings** section has many menu items available, most of them admin based. We will cover many of these items during the this course. Information on the items not covered is available at the links below.

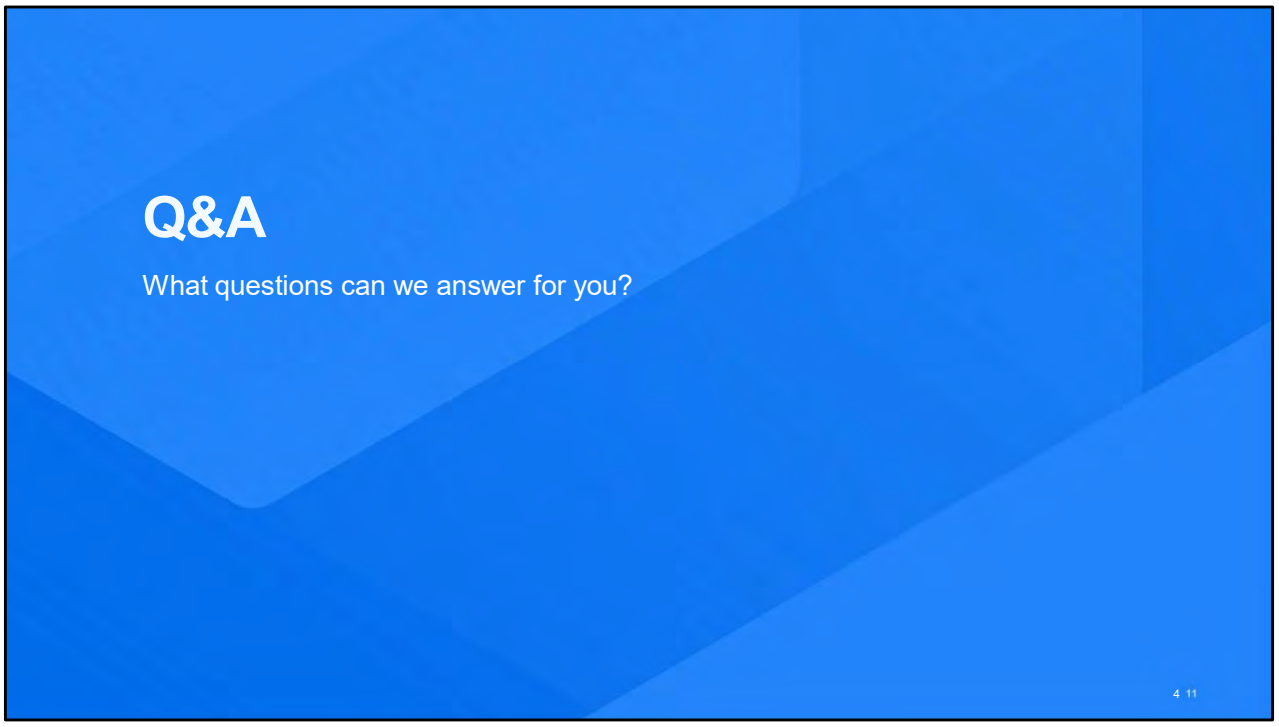
- https://docs.chef.io/automate/notifications/
- https://docs.chef.io/automate/datafeed/
- https://docs.chef.io/automate/data_lifecycle/
- https://docs.chef.io/automate/projects/

Progress Chef





https://docs.chef.io/automate/notifications/
https://docs.chef.io/automate/datafeed/
https://docs.chef.io/automate/data_lifecycle/
https://docs.chef.io/automate/projects/

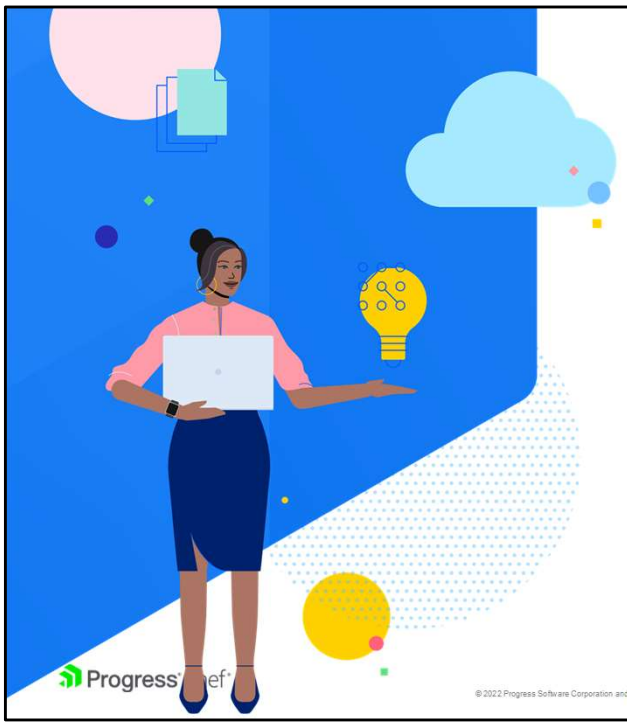

Q&A
What questions can we answer for you?
4 11


Progress Chef

Progress Chef

# Scanning with Compliance Profiles

Scanning a node for compliance violations




# Objectives

After completing this module, you should be able to:

- Describe compliance profiles
- Find an available compliance profile
- Make compliance profiles available for use
- Add a node to test for compliance
- Run a Compliance scan

# Concept

## Scanning with Chef Automate

Scanning nodes as indicated in this module can be useful for performing ad hoc scans and for learning how Chef Automate can be used to leverage the included Compliance profiles as well as how Chef Automate displays scan results.

In practice you can use a more automated way to scan nodes, such as using the Chef Infra client's (chef-client) Compliance Phase functionality, which we will cover later in this course.

# Concept

## Scanning a Node

To scan a node you'll need:

- A Compliance Profile to scan with
- The IP address or FQDN of the nodes to be tested
- Access configuration (ssh or WinRM)
- The node's username and security key

# Concept

## What is a Compliance Profile?

A compliance profile is a collection of individual controls (InSpec code blocks) that comprise a complete audit test. Hundreds of compliance profiles are available in Chef Automate.

| | |
|---|---|
| CIS CentOS Linux 7 Benchmark Level 1 - Server | 2.2.0-18 |
| CIS CentOS Linux 7 Benchmark Level 1 - Workstation | 3.1.2-2 |
| CIS CentOS Linux 7 Benchmark Level 1 - Workstation | 2.2.0-13 |
| CIS CentOS Linux 7 Benchmark Level 2 | 1.1.0-7 |
| CIS CentOS Linux 7 Benchmark Level 2 - Server | 3.1.2-3 |

## What is in a Compliance Profile?

**Control** – A unique identifier string
**Impact** – Rating of how serious the control is, scale is 0 to 1 in tenths
**Title** – More descriptive way to identify the control
**Desc** – Description of the control, sometimes has hyperlink with more information
**Describe** – The InSpec test verifying the configuration

```
control 'ssh-04' do
  impact 1.0
  title 'Client: Specify protocol version 2'
  desc "Only SSH protocol version 2 connections should be permitted.
        Version 1 of the protocol contains security vulnerabilities.
        Don't use legacy insecure SSHv1 connections anymore."
  describe ssh_config do
    its('Protocol') { should eq('2') }
  end
end

control "xccdf_org.cisecurity.benchmarks_rule_1.1.19_
         Ensure_nosuid_option_set_on_removable_media_partitions" do
  title "Ensure nosuid option set on removable media partitions"
  desc  "The nosuid mount option specifies that the filesystem cannot
    contain setuid files. Rationale: Setting this option on a
    file system prevents users from introducing privileged programs
    onto the system and allowing non-root users to execute them."
  impact 0.0
  describe 'removable media partitions' do
    skip 'Ensure nosuid option set on removable media partitions'
  end
end
```

EXERCISE

# Group Lab: Scan a Node With an Available Compliance Profile

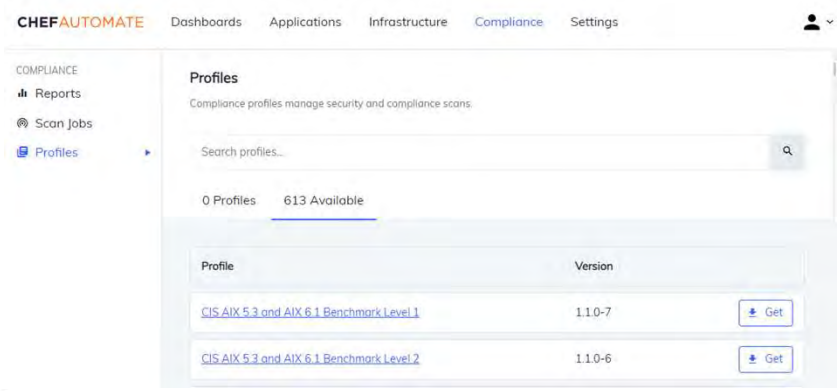
**Objective:**

- ❑ Locate Compliance Profile
- ❑ Add credentials for the node being scanned
- ❑ Add node to be scanned
- ❑ Run ad-hoc scan job
- ❑ View the results

Progress Chef

# GL: Locate Compliance Profile

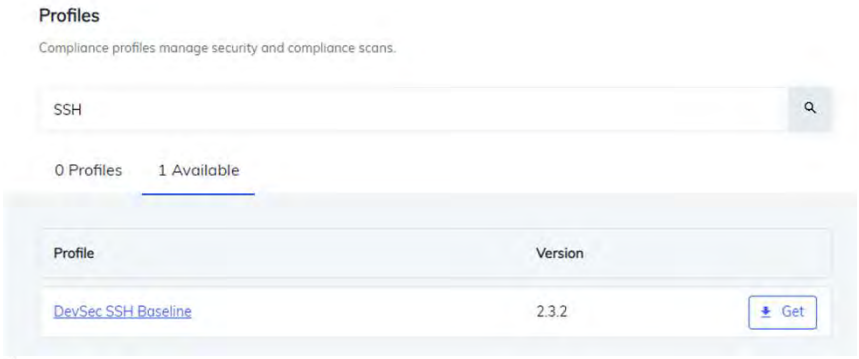
The first thing we need to scan a node is a compliance profile to scan with. These are located in the **Compliance** section, in the **Profiles** window.




Progress Chef

# GL: Locate Compliance Profile

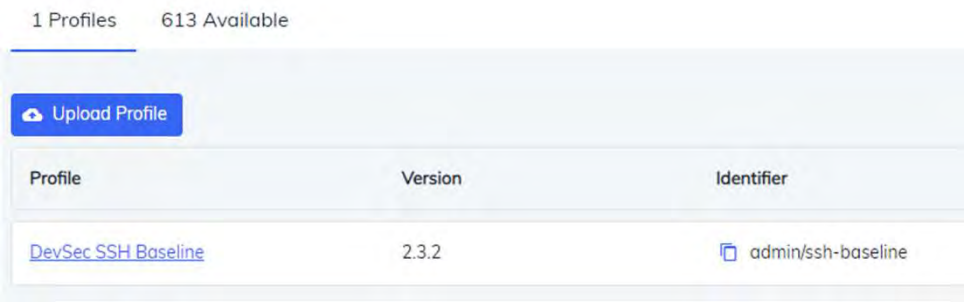
We are going to use the **DevSec SSH Baseline** compliance profile for our scan. Enter SSH into the search bar and click on the magnifying glass on the right-hand side. Then click the **Get** button on the right-hand side.




Progress Chef

# GL: Locate Compliance Profile

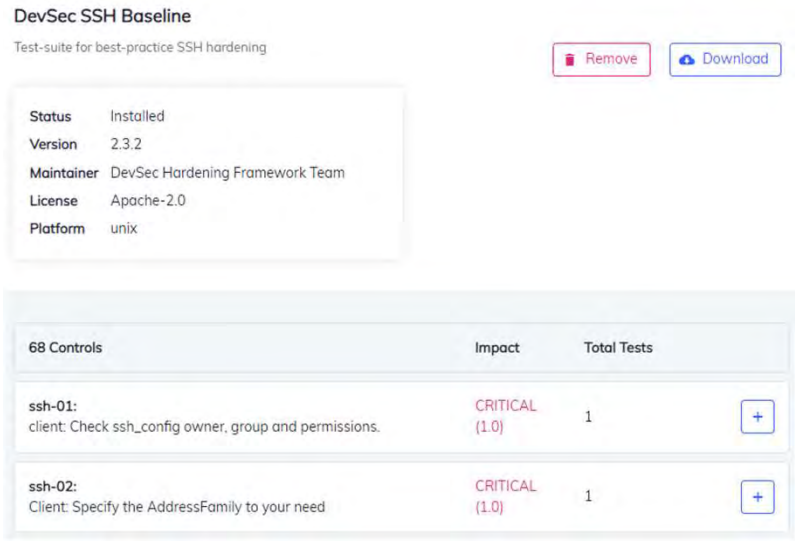
After clicking on **Get**, the profile is now shown in our Profiles tab. The identifier is a composite value of the user that made the profile available and the profile name. This is important to note for a future module.



# GL: Locate Compliance Profile

Let's take a minute to see what is in this profile. Clicking on the name of the profile will give us more information such as:

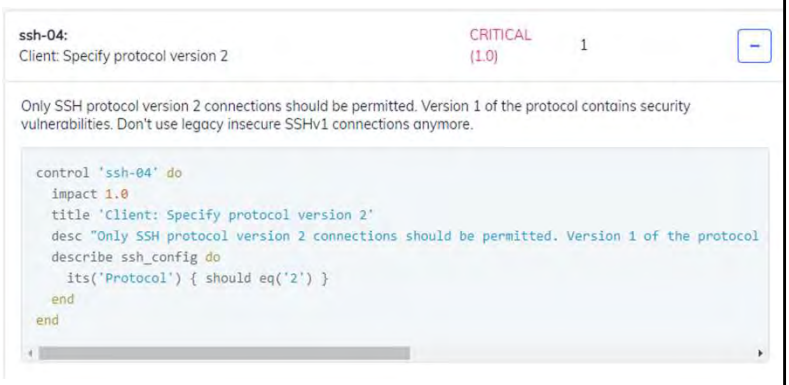
- the profile name
- a short description
- the version number
- the profile license
- the platform it works on
- the number of controls in the profile




Progress Chef

# GL: Locate Compliance Profile

Digging a little deeper, we can click the + on the right side of a control to see what it does. The top section and bottom section contain much of the same information. The main differences are the top is formatted for a web page and the bottom contains the InSpec code.



EXERCISE

# Group Lab: Scan a Node With an Available Compliance Profile

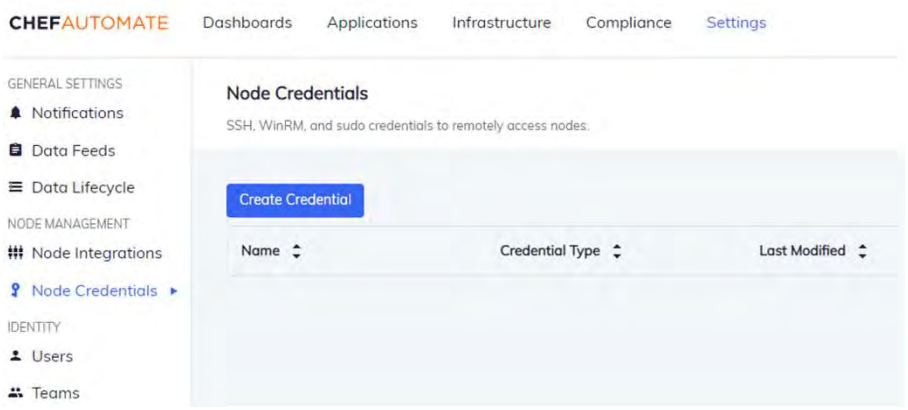
**Objective:**

- ✓ Locate Compliance Profile
- ❑ Add credentials for the node being scanned
- ❑ Add node to be scanned
- ❑ Run ad-hoc scan job
- ❑ View the results

Progress Chef

# GL: Add credentials for the node being scanned

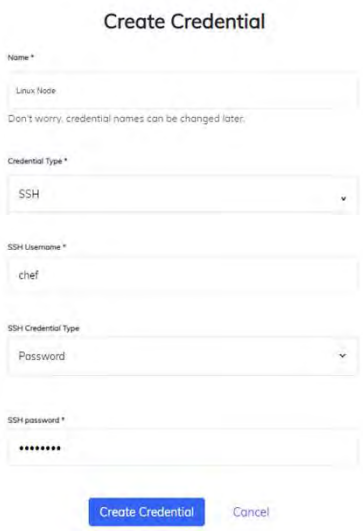
Now that our compliance profile is ready we need to give Automate the credentials to login to the node to be scanned. The is done in the **Settings** section, on the **Node Credentials** window.




Progress Chef

## GL: Add credentials for the node being scanned

Clicking on the **Create Credential** button will launch the pop-up window show to the right. Enter the information given in the notes below and click **Create Credential**.




Progress Chef

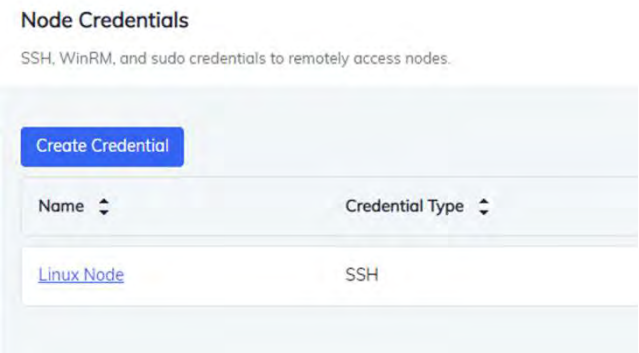




**Name:** Linux Node
**Credential Type:** SSH
**SSH Username:** chef
**SSH Credential Type:** Password
**SSH Password:** Cod3Can!

# GL: Add credentials for the node being scanned

The credential for the node we will scan is ready to be used now. We will see this credential available when we setup our scan.




Progress Chef

EXERCISE

# Group Lab: Scan a Node With an Available Compliance Profile

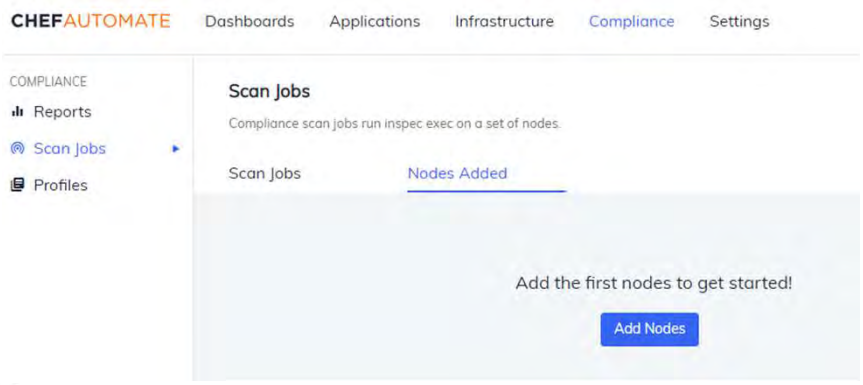
**Objective:**

- ✓ Locate Compliance Profile
- ✓ Add credentials for the node being scanned
- ❑ Add node to be scanned
- ❑ Run ad-hoc scan job
- ❑ View the results

Progress Chef

# GL: Add node to be scanned

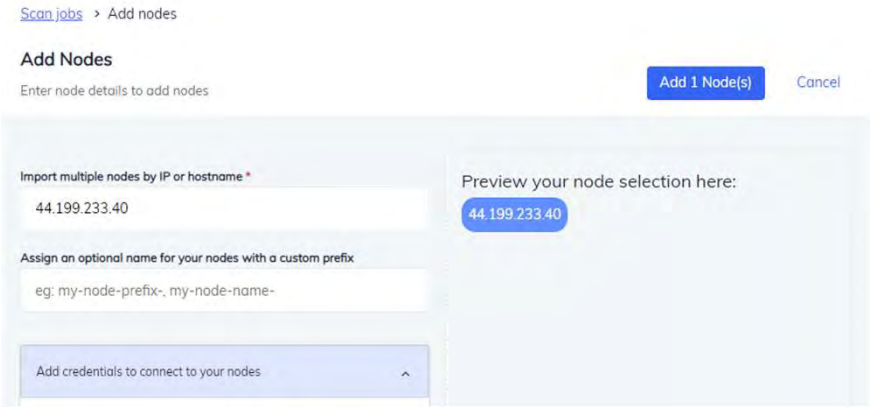
The next step is to add the node, we do this in the **Compliance** section on the **Scan Jobs** window in the **Nodes Added** tab. Once in this location, click on the **Add Nodes** button.




Progress Chef

# GL: Add node to be scanned

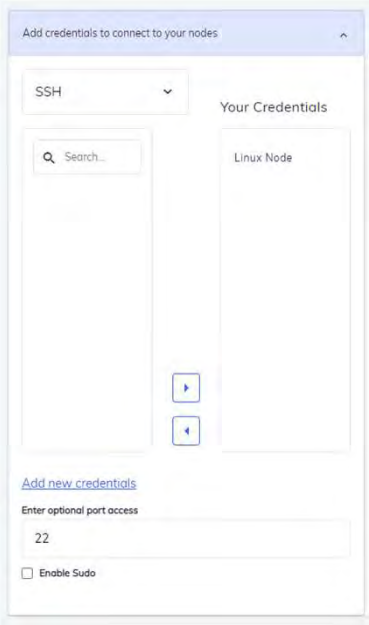
On the next window, begin by filling in the node's IP address.




Progress Chef

# GL: Add node to be scanned

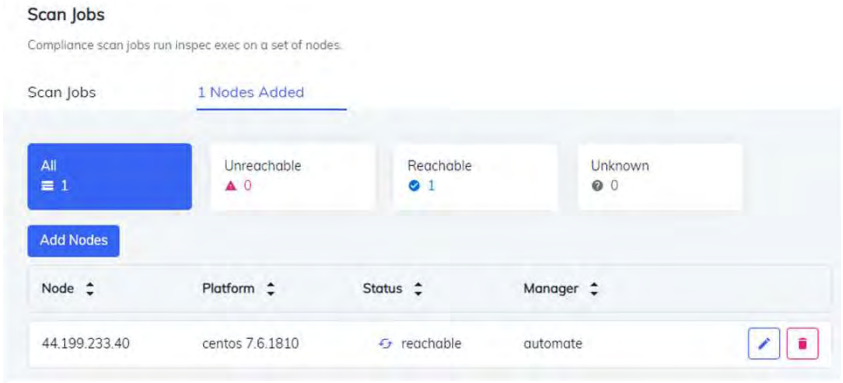
Now we can scroll down and select the credential to use for this node. Highlight Linux Node and click on the triangle pointing to the right. Scroll back to the top and click on the **Add 1 Node(s)** button.




Progress Chef

# GL: Add node to be scanned

Back on the **Nodes** tab you will see your node listed. At first the status will be unknown, but it should change to reachable after a few seconds. If it does not, then something is wrong. Verify the machine is running and the credentials are correct.




Progress Chef

EXERCISE

# Group Lab: Scan a Node With an Available Compliance Profile

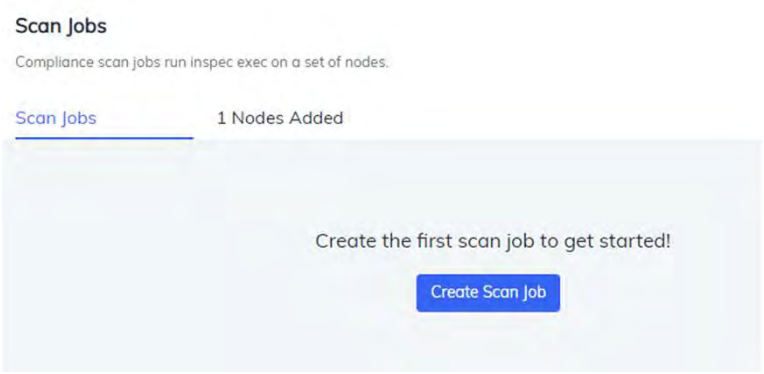
**Objective:**

- ✓ Locate Compliance Profile
- ✓ Add credentials for the node being scanned
- ✓ Add node to be scanned
- ❑ Run ad-hoc scan job
- ❑ View the results

Progress Chef

# GL: Run ad-hoc scan job

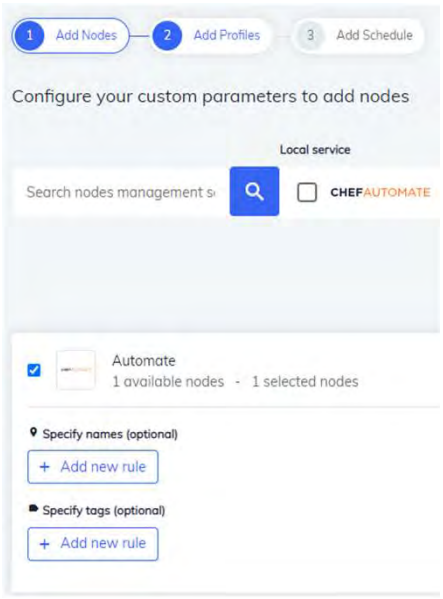
We now have all the pieces in place to scan a node. Move over to the **Scan Jobs** tab in the **Scan Jobs** window and click on the **Create Scan Job** button.




Progress Chef

# GL: Run ad-hoc scan job

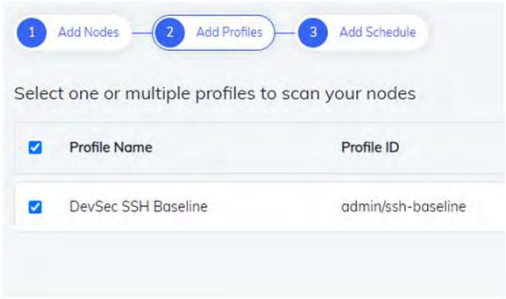
The first step is to select a node to scan. Check the box next to **Automate** to open a new section, our lone node is already selected. After ensuring your node is selected, click the **Next** button.




Progress Chef

# GL: Run ad-hoc scan job

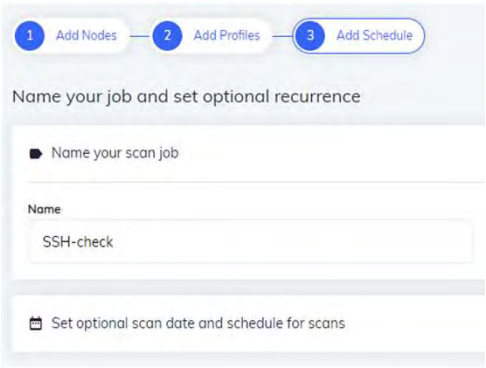
The next step is to select the profile that you want to use for your scan. Check the box next to the **DevSec SSH Baseline** profile and click the **Next** button.




Progress Chef

# GL: Run ad-hoc scan job

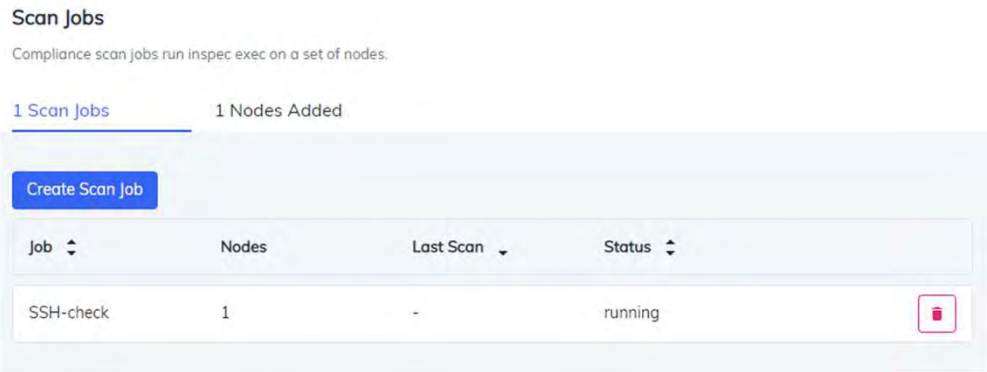
The final step is to give the scan job a name. Every scan must have a name. We also have the option to create a schedule and have this job run automatically. Enter a name and click the **Save** button.




Progress Chef

# GL: Run ad-hoc scan job

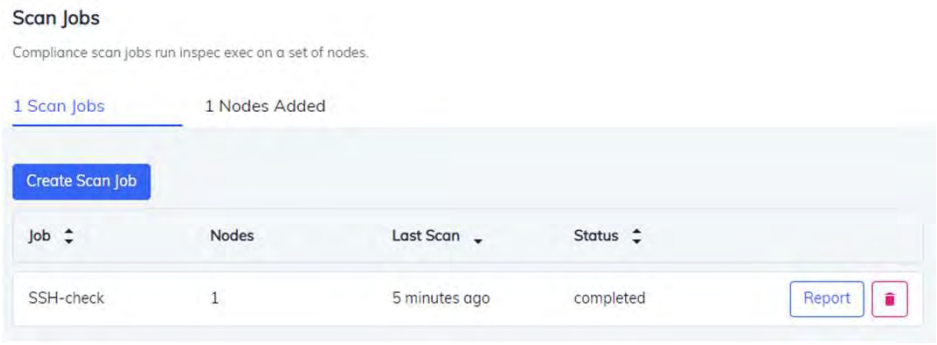
After clicking the **Save** button, you will be returned to the **Scan Jobs** tab and you job will begin running immediately.




Progress Chef

# GL: Run ad-hoc scan job

Depending on the number of nodes being scanned and the number of controls being tested, the scan can take anywhere from under a minute to several minutes to complete. Once the job is complete, you will be able to view the results by clicking on the **Report** button.



EXERCISE

# Group Lab: Scan a Node With an Available Compliance Profile

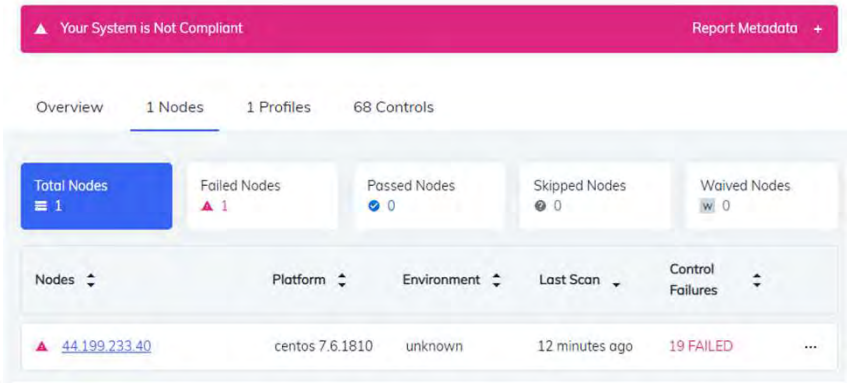
**Objective:**

- ✓ Locate Compliance Profile
- ✓ Add credentials for the node being scanned
- ✓ Add node to be scanned
- ✓ Run ad-hoc scan job
- ❑ View the results

Progress Chef

# GL: View the results

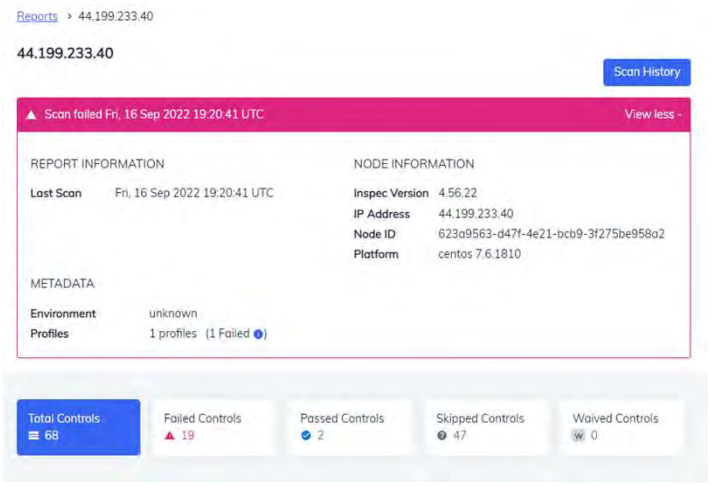
Clicking on the **Report** button takes us directly to the **Reports** window. Several options exist for looking at scan results. We can focus on the nodes scanned, the compliance profiles used, or the individual controls used in the scan. Let's focus on the node by clicking on the **Nodes** tab.




Progress Chef

# GL: View the results

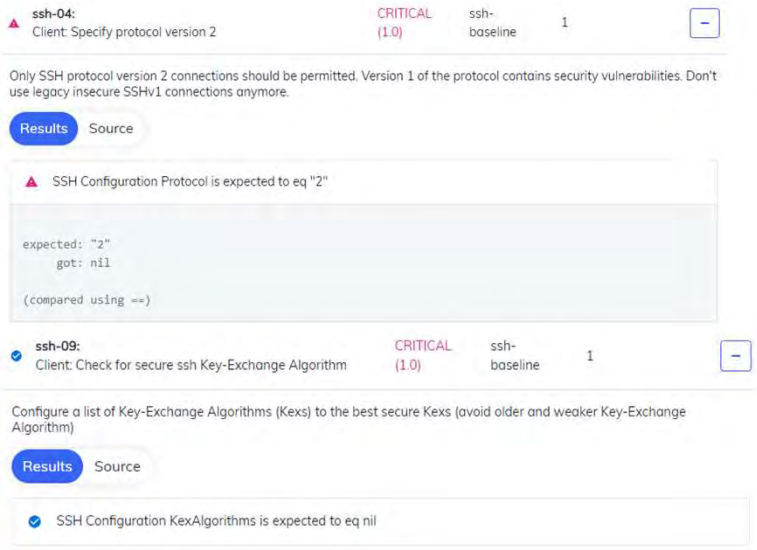
Clicking on the node IP address displays the results of the scan. We can quickly sort by failed controls, passed controls, skipped controls, and waived controls.




Progress Chef

# GL: View the results

Scrolling down, we can expand the controls and see the results. Some controls passed and some controls failed. We will cover remediation in the next module.




Progress Chef

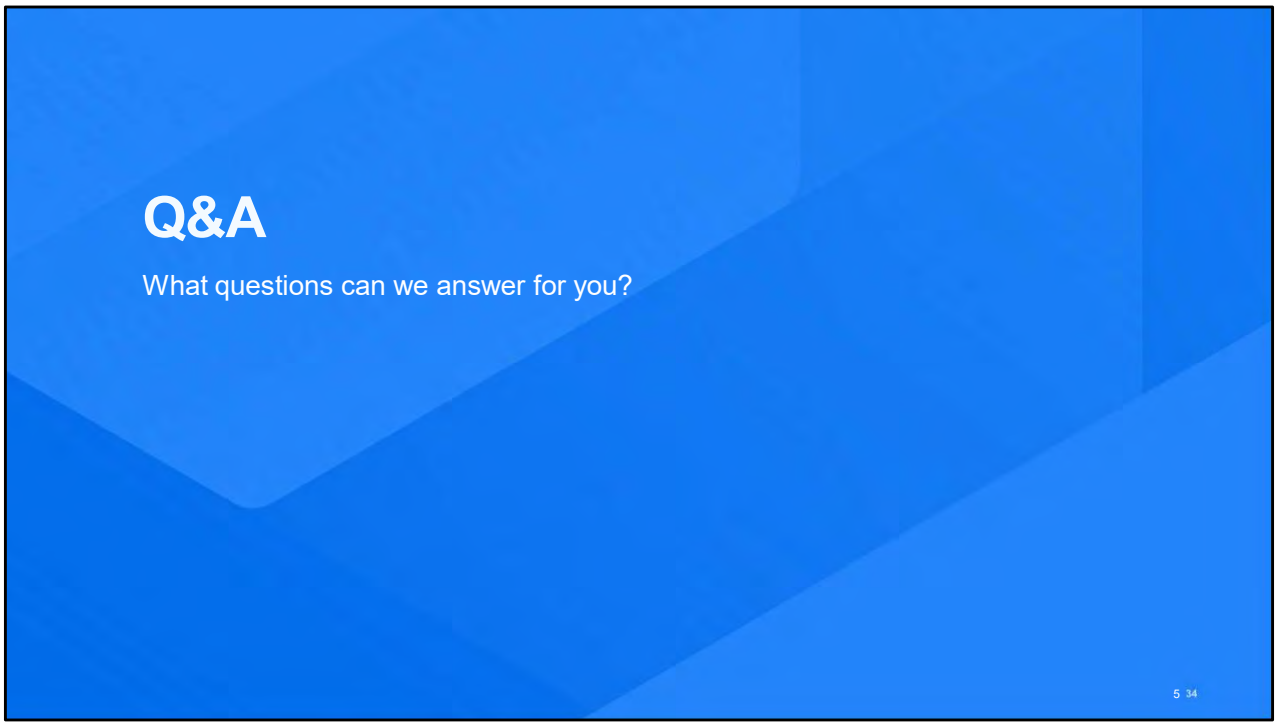
EXERCISE

# Group Lab: Scan a Node With an Available Compliance Profile

**Objective:**

- ✓ Locate Compliance Profile
- ✓ Add credentials for the node being scanned
- ✓ Add node to be scanned
- ✓ Run ad-hoc scan job
- ✓ View the results

Progress Chef


Q&A
What questions can we answer for you?
5 34


Progress Chef

Progress Chef

# Remediating Compliance Profiles

Addressing Failures in the Compliance Report

## Objectives

After completing this module, you should be able to:

- Find compliance scan reports
- Apply cookbook to node
- Re-scan node with compliance profile

Progress Chef

# Concept

## Compliance scan failures

The compliance scan from our previous module had several failures. These items could be addressed one at a time manually, but who wants to do that?

A better solution is to create a cookbook that addresses each control in the compliance scan. This gives us the ability to apply the cookbook to any node that fails the scan.

Concept

## Correcting Failures With Cookbooks

Writing cookbooks is not the focus of this course, so we are going to use an existing cookbook that addresses one control in the compliance scan.

Information on creating cookbooks, policyfiles, and more is covered in the Chef Infra Foundations course.

EXERCISE

# Group Lab: Remediate Scan Failure

**Objective:**

- ❑ Run chef-client on Linux node
- ❑ Scan Linux node again

Progress Chef



6 5

## GL: Run chef-client on Linux Node

This exercise will be done on our Linux Node. The instructor should have provided an IP address to you, login to the machine now using the credentials below.

```
$ ssh chef@54.237.181.43
chef@54.237.181.43's password:
Last login: Thu Oct 13 15:22:01 2022 from 071-011-228-040.res.spectrum.com
[chef@ip-172-31-65-135 ~]$
```

Progress Chef



6 6

Command:
ssh chef@LINUX_NODE_IP_ADDRESS

Username: chef
Password: Cod3Can!

## GL: Run chef-client on Linux Node

We will start by moving to the directory holding our cookbooks, **~/cookbooks**. Before we apply the **ssh-remediation** cookbook, let's look at the default recipe and see what it does. This recipe has a single template resource that manages **/etc/ssh/ssh_config**.

```
#
# Cookbook:: ssh-remediation
# Recipe:: default
#
# Copyright:: 2018, The Authors, All Rights Reserved.

template '/etc/ssh/ssh_config' do
  source 'ssh_config.erb'
end
```

Progress Chef



6 7

Commands:
cd ~/cookbooks/
cat ssh-remediation/recipes/default.rb

## GL: Run chef-client on Linux Node

We are going to use the **chef-client** command to apply the cookbook. By default the **chef-client** command will look to the Chef Infra Server to find what cookbooks should be applied, but we have not applied a cookbook to this node. We are going to use **chef-client** in local mode and specify the cookbook to be applied.

```
[chef@ip-172-31-75-89 cookbooks]$ sudo chef-client --local-mode -r "recipe[ssh-remediation]"
[2022-10-10T18:35:13+00:00] WARN: No config file found or specified on command line. Using command line options instead.
Chef Infra Client, version 17.10.0
Patents: https://www.chef.io/patents
Infra Phase starting
Resolving cookbooks for run list: ["ssh-remediation"]
Synchronizing cookbooks:
  - ssh-remediation (0.1.0)
Installing cookbook gem dependencies:
Compiling cookbooks...
Loading Chef InSpec profile files:
Loading Chef InSpec input files:
Loading Chef InSpec waiver files:
Converging 1 resources
```

Progress Chef



6 8

Command:
sudo chef-client --local-mode -r "recipe[ssh-remediation]"

EXERCISE

# Group Lab: Remediate Scan Failure

**Objective:**

- ✓ Run chef-client on Linux Node
- ❑ Scan Linux node again

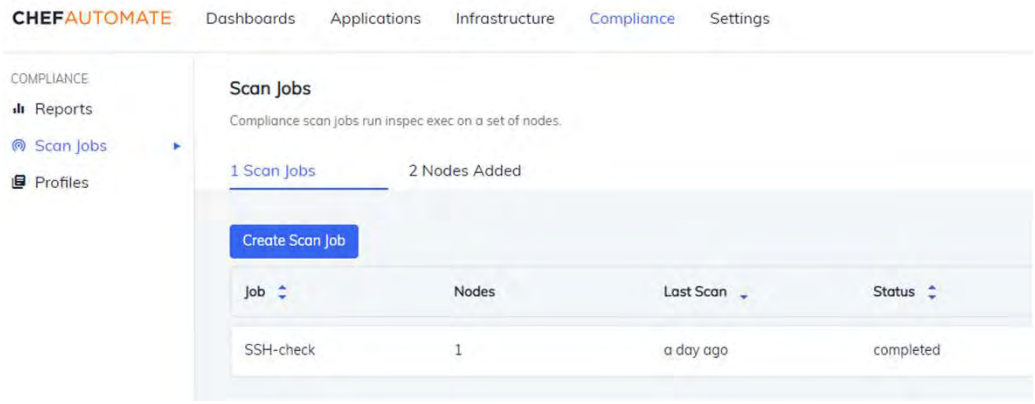
Progress Chef



6 9

# GL: Scan Linux node again

To re-scan our Linux node, go to the **Scan Jobs** window in the **Compliance** section and click the **Create Scan Job** button.



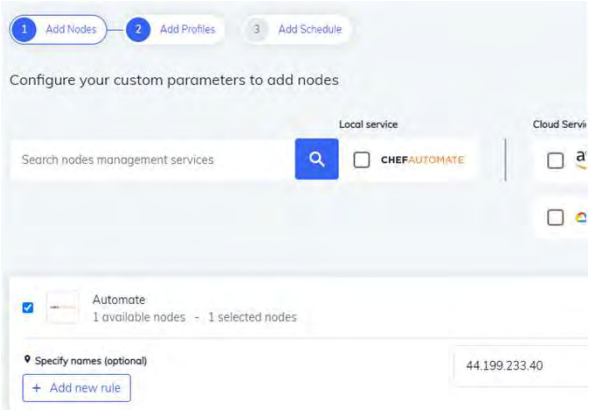

Progress Chef



6 10

# GL: Scan Linux node again

The first step is to add the node to the job. Check the box next to **Automate**, the node IP should be added, and click **Next**.



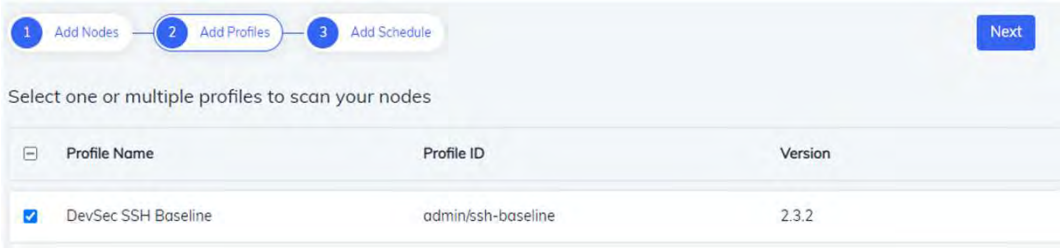

Progress Chef



6 11

## GL: Scan Linux node again

Next we add the profile to scan with. Check the box next to the **DevSec SSH Baseline** profile and click **Next**.



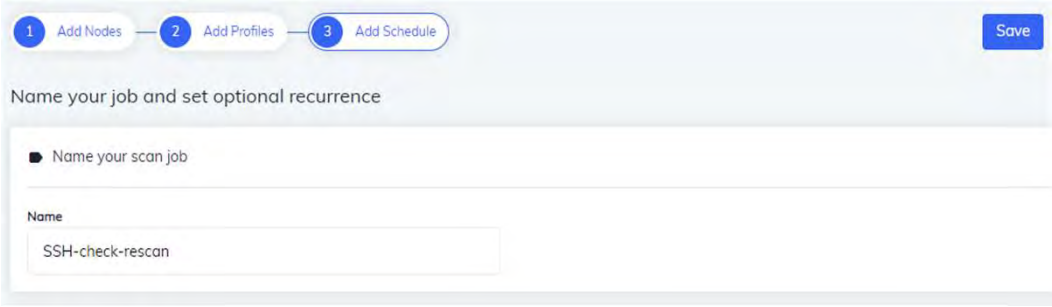

Progress Chef



6 12

## GL: Scan Linux node again

Finally, we give the scan job a name. Enter a name and click **Save**. After clicking **Save** the scan job will begin to run immediately.



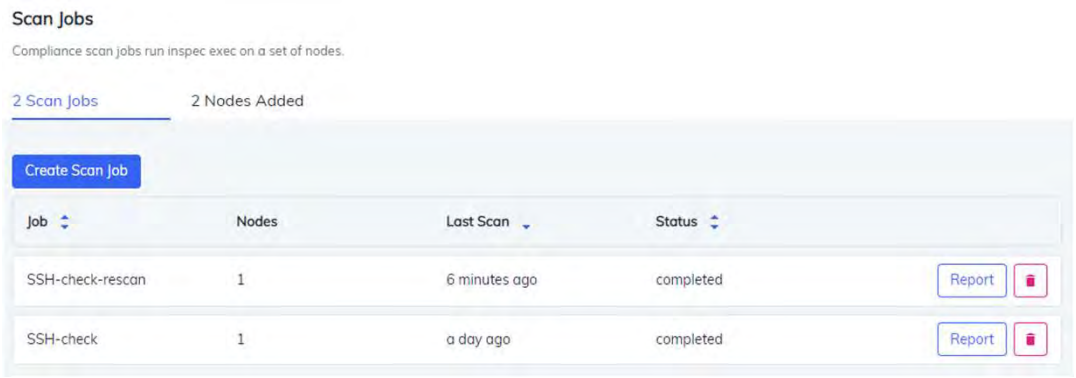

Progress Chef



6 13

# GL: Scan Linux node again

When the scan completes, click on the **Report** button to the right side to see the results.



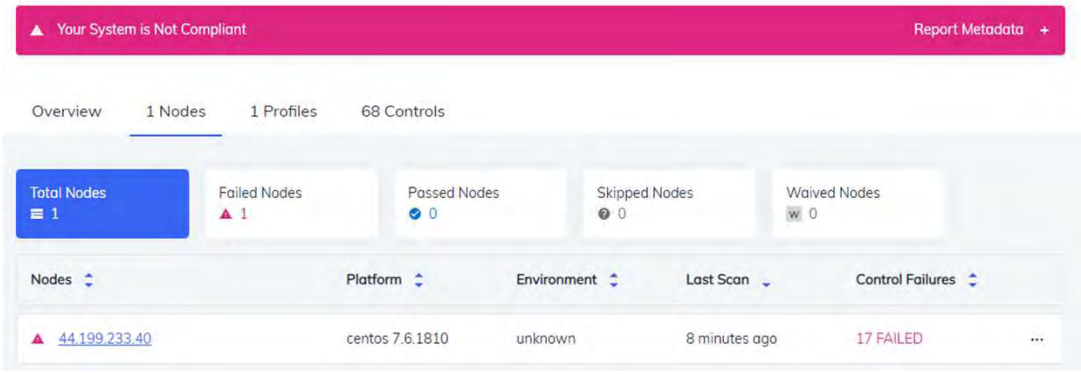

Progress Chef



6 14

# GL: Scan Linux node again

Clicking the **Reports** button takes us to the **Reports** window. Click on the **Nodes** tab to find our Linux node. Then click on the node IP address.



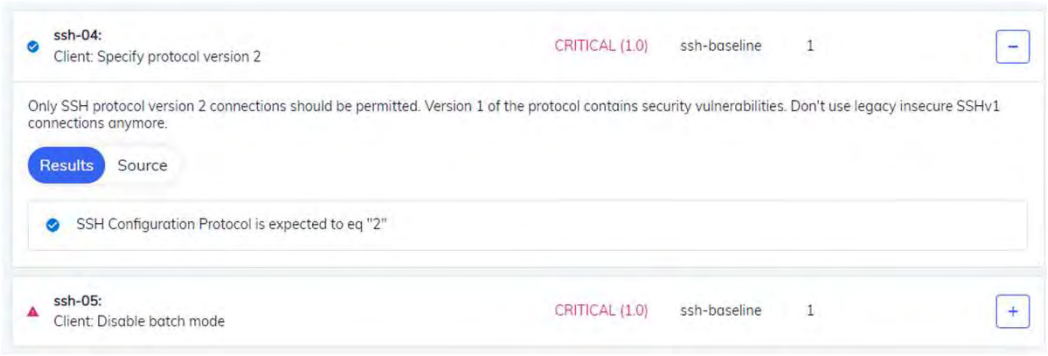

Progress Chef



6 15

# GL: Scan Linux node again

We can now scroll down to the controls and see that control ssh-04 is now passing because the cookbook corrected the issue in our SSH config.




Progress Chef



6 16

EXERCISE

# Group Lab: Remediate Scan Failure

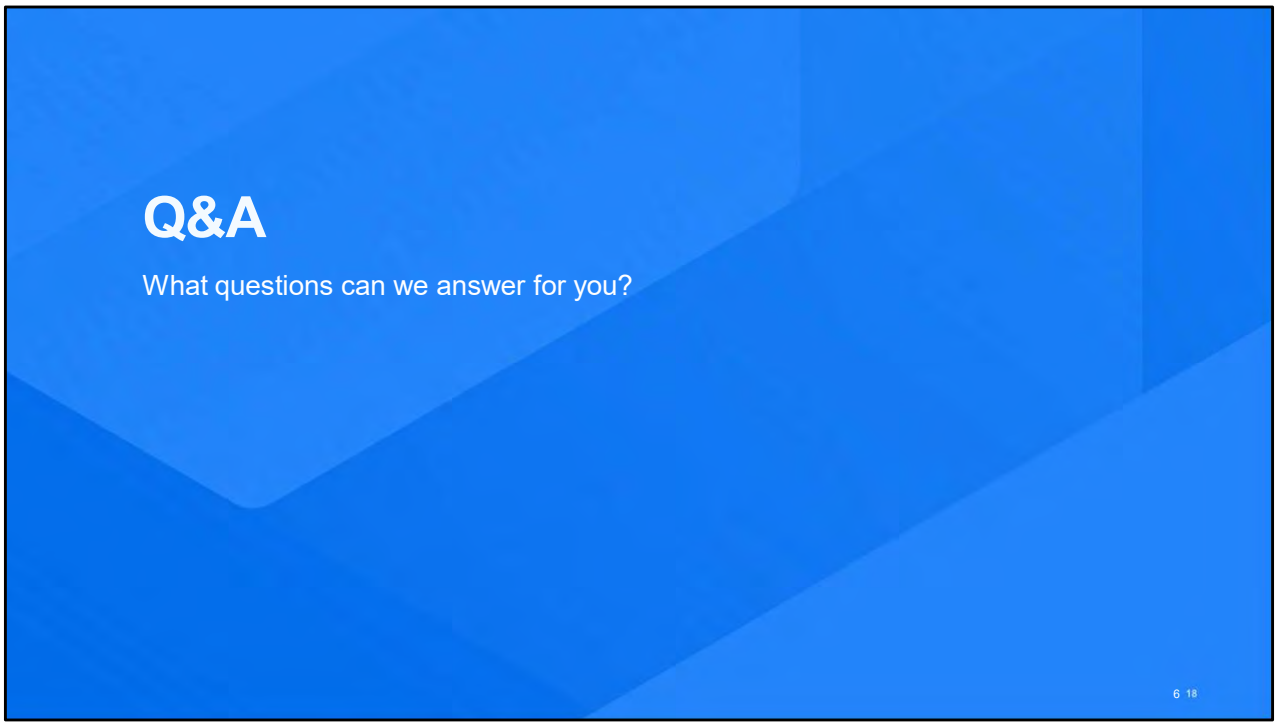
**Objective:**

- ✓ Run chef-client on Linux Node
- ✓ Scan Linux node again

Progress Chef



6 17


Q&A
What questions can we answer for you?
6 18


Progress Chef

Progress Chef

# Creating Custom Compliance Profiles

Creating and Uploading Compliance Profiles to the Automate Server

Instructor Note: This module uses the Windows VM. The students will need the IP address for the Windows node

# Objectives

After completing this module, you should be able to:

- Write a custom compliance profile
- Use InSpec to test your custom profile
- Upload a custom compliance profile to the Chef Automate server

Progress Chef

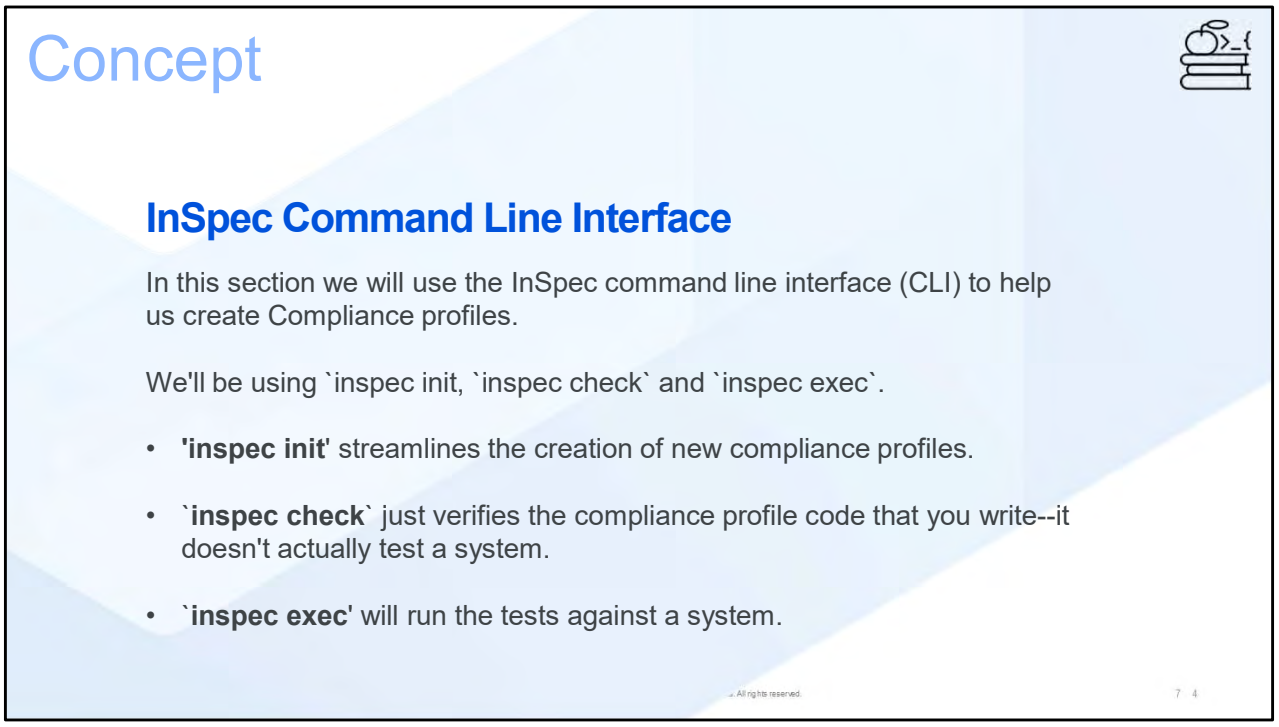
# Concept

## Creating a Custom Profile

If you are testing for compliance to an uncommon audit or even one that was created by your company, then you may not find a profile available. This is when you want to create a custom compliance profile.

Custom profiles are created using InSpec, just like the existing profiles were created.

After we have created a custom profile, we will upload it to Chef Automate so it can be used to test for compliance issues.




`inspec init` can create all the directories and basic files that the Compliance Server and `inspec check` and `inspec exec` require.

## Log In To Your Windows Node

This module is the only one in this course that uses the Windows node provided by the instructor. Use RDP to login to the IP address provided to you. The credentials are given below.

We will be using Chef Workstation, VS Code, and a web browser to complete the labs in this module.

Progress Chef



7 5

**Username:** administrator
**Password:** Cod3Can!

EXERCISE

# Group Lab: Creating a Custom Profile

**Objective:**

- ❑ Create a custom profile
- ❑ Add controls to profile
- ❑ Test your profile for errors with InSpec
- ❑ Test your profile for functionality with InSpec
- ❑ Upload your compliance profile to Automate

Progress Chef

## GL: Create a custom profile

The first thing we need to do after logging in is launch Chef Workstation. If you have never used it before, Chef Workstation is command line based so there is no GUI. On our Windows node it launches PowerShell and loads some extra pieces for us. There is a shortcut on the desktop named CW PowerShell, double-click on it.


Progress Chef



7 7

## GL: Create a custom profile

The **inspec** command is new to us, lets see what it can do.

```
PS C:\Users\Administrator\Desktop> inspec --help
Commands:
  inspec archive PATH                                          # archive a profile to...
  inspec artifact SUBCOMMAND                                   # Manage Chef InSpec A...
  inspec automate SUBCOMMAND or compliance SUBCOMMAND          # Chef Automate commands
  inspec check PATH                                            # verify all tests at ...
  inspec clear_cache                                           # clears the InSpec ca...
  inspec detect                                                # detect the target OS
  inspec env                                                   # Output shell-appropr...
  inspec exec LOCATIONS                                        # Run all tests at LOC...
  inspec habitat SUBCOMMAND                                    # Manage Habitat with ...
  inspec help [COMMAND]                                        # Describe available c...
  inspec init SUBCOMMAND                                       # Generate InSpec code
  inspec json PATH                                             # read all tests in PA...
  inspec plugin SUBCOMMAND                                     # Manage Chef InSpec a...
  inspec shell                                                 # open an interactive ...
  inspec supermarket SUBCOMMAND ...                            # Supermarket commands
  inspec vendor PATH                                           # Download all depende...
  inspec version                                               # prints the version o...
```

Progress Chef



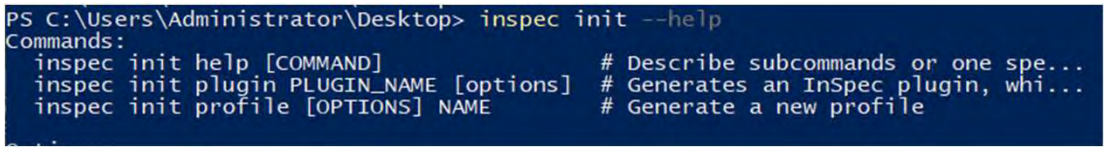
7 8

More information about the inspec command is available here:
https://docs.chef.io/inspec/cli/

Command:
inspec --help

## GL: Create a custom profile

We want to create a custom profile, **inspec init** looks helpful.




Progress Chef



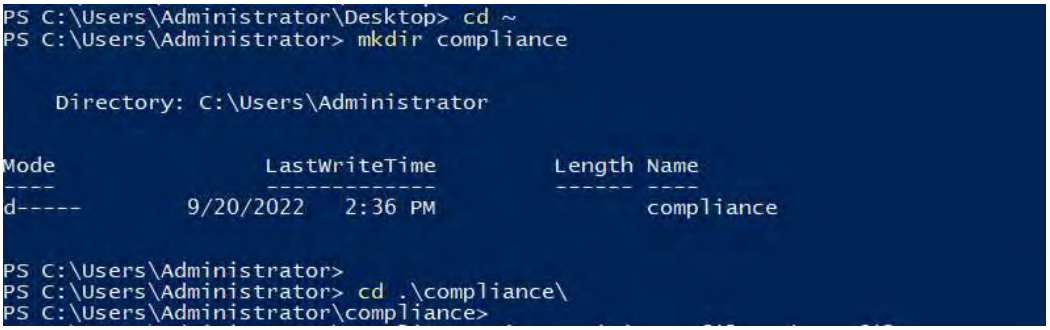
7 9

The `inspec init` command enables you to create new Compliance profiles with less manual intervention than in previous versions of InSpec.

Command:
inspec init --help

## GL: Create a custom profile

We should create a directory dedicated for our compliance profiles.




Progress Chef



7 10

Commands:
cd ~
mkdir compliance
cd .\compliance

## GL: Create a custom profile

Now that we are in our profiles directory, we can create our first profile.

```
PS C:\Users\Administrator\compliance> inspec init profile windows_profile

                    InSpec Code Generator

Creating new profile at C:/Users/Administrator/compliance/windows_profile
 • Creating file README.md
 • Creating directory controls
 • Creating file controls/example.rb
 • Creating file inspec.yml
```

https://docs.chef.io/inspec/profiles/

Progress Chef



7 11

Note: you may be asked to accept the Chef Workstation license
More information about InSpec profiles is available here:
https://docs.chef.io/inspec/profiles/

Command:
inspec init profile windows_profile

Concept

## Controls directory

Now that we have created a profile, let's take a look at what it contains.

All of the controls in your profile are stored in the Controls directory. An example.rb file is created when can be used as a template if you are new to building compliance profiles.

# Concept

## inspec.yml

The **inspec.yml** file is similar to the metadata.rb file in a cookbook. It holds the profile name, version, and dependencies.

```
name: windows_profile
title: InSpec Profile
maintainer: The Authors
copyright: The Authors
copyright_email: you@example.com
license: Apache-2.0
summary: An InSpec Compliance Profile
version: 0.1.0
supports:
  platform: os
```

https://docs.chef.io/inspec/profiles/#inspecyml

https://docs.chef.io/inspec/profiles/#inspecyml

Concept

## InSpec Profile Dependencies

A Chef InSpec profile can bring in the controls and custom resources from another Chef InSpec profile. Additionally, when inheriting the controls of another profile, a profile can skip or even modify those included controls.

These methods are similar to using a cookbook in the supermarket. We will cover the concepts now, but not use them in this exercise. We will use the **include_controls** method in another module. More information is available at the link below.

https://docs.chef.io/inspec/profiles/#profile-dependencies

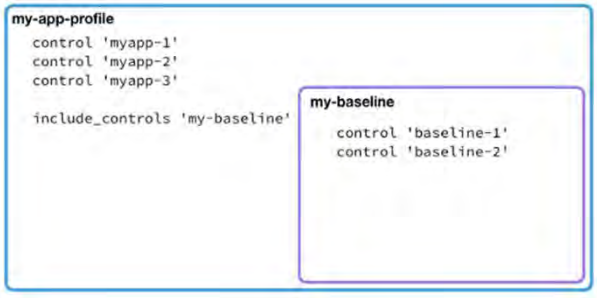


7 14

https://docs.chef.io/inspec/profiles/#profile-dependencies

## include_controls

With the **include_controls** method in a profile, all controls from the named profile will be executed every time the including profile is executed. We will see an example of this in a future module.



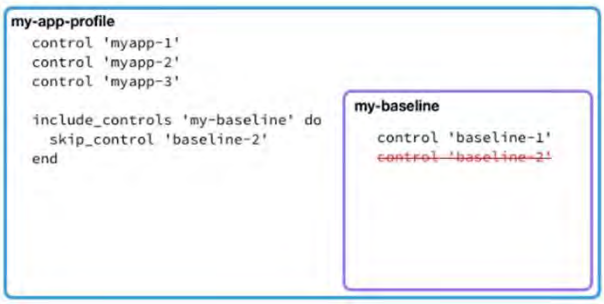

Progress

https://docs.chef.io/inspec/profiles/#including-all-controls-from-a-profile

## skip_control

What if one of the controls from the included profile does not apply to your environment? Luckily, it is not necessary to maintain a slightly-modified copy of the included profile just to delete a control. The **skip_control** method tells Chef InSpec to not run a particular control.



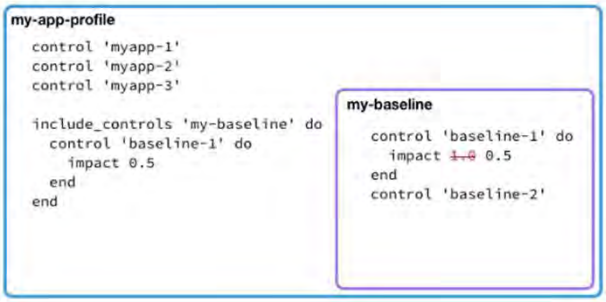

Progress 7 16 

https://docs.chef.io/inspec/profiles/#skipping-a-control-from-a-profile

## Modifying A Control

Let's say a particular control from an included profile should still be run, but the impact isn't appropriate? Perhaps the test should still run, but if it fails, it should be treated as low severity instead of high severity?

When a control is included, it can also be modified!

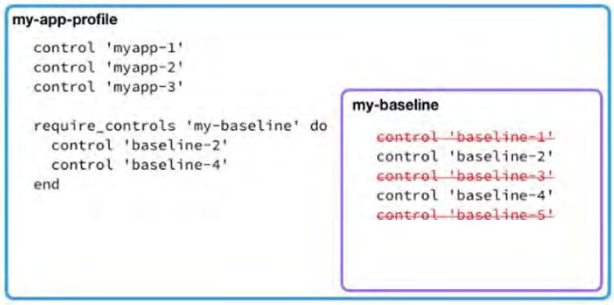



Progress 7 17 

https://docs.chef.io/inspec/profiles/#modifying-a-control

## require_controls

If there are only a handful of controls that should be executed from an included profile, it's not necessary to skip all the unneeded controls, or worse, copy/paste those controls bit-for-bit into your profile. Instead, use the **require_controls** method.




Progress

https://docs.chef.io/inspec/profiles/#selectively-including-controls-from-a-profile

EXERCISE

# Group Lab: Creating a Custom Profile

**Objective:**

- ✓ Create a custom profile
- ❑ Add controls to profile
- ❑ Test your profile for errors with InSpec
- ❑ Test your profile for functionality with InSpec
- ❑ Upload your compliance profile to Automate

Progress Chef

## GL: Add controls to profile

We are ready to add controls to our profile now. The first step is to rename the **example.rb** file in the controls directory to something more appropriate.


Progress Chef

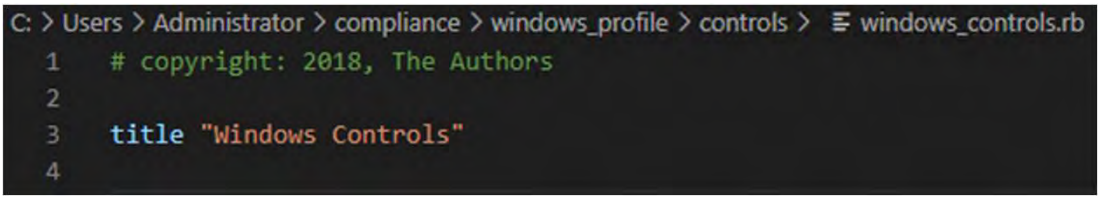


7 20

Commands:
cd .\windows_profile\controls
rename-item .\example.rb windows_controls.rb

## GL: Add controls to profile

Now we can open **windows_controls.rb** with VS Code and remove the example code. Remove everything below line 4 and update the title on line 3.




Progress Chef

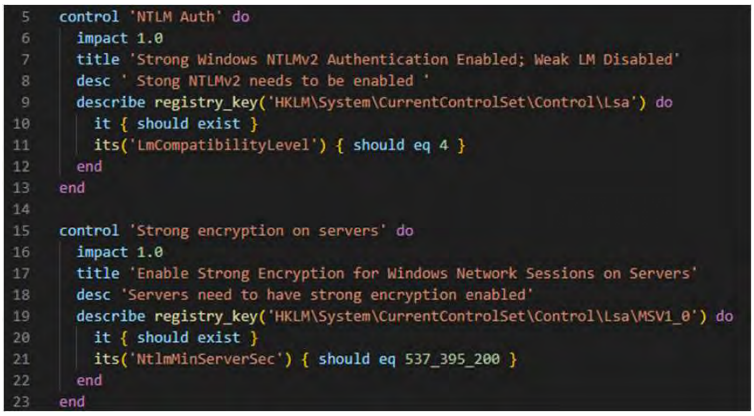


7 21

Command:
code windows_controls.rb

## GL: Add controls to profile

We are creating two security related tests for this lab. Add the code given below to your **windows_controls.rb file.**



```
control 'NTLM Auth' do
  impact 1.0
  title 'Strong Windows NTLMv2 Authentication Enabled; Weak LM Disabled'
  desc ' Stong NTLMv2 needs to be enabled '
  describe registry_key('HKLM\System\CurrentControlSet\Control\Lsa') do
    it { should exist }
    its('LmCompatibilityLevel') { should eq 4 }
  end
end

control 'Strong encryption on servers' do
  impact 1.0
  title 'Enable Strong Encryption for Windows Network Sessions on Servers'
  desc 'Servers need to have strong encryption enabled'
  describe registry_key('HKLM\System\CurrentControlSet\Control\Lsa\MSV1_0') do
    it { should exist }
    its('NtlmMinServerSec') { should eq 537_395_200 }
  end
end
```

Progress Chef



7 22

Note: if any yellow boxes are shown in VS Code after pasting the code below, delete them and add a space. Right-click on one yellow box and select change all occurrences, then press backspace button and then the space bar.

Code:

```
control 'NTLM Auth' do
  impact 1.0
  title 'Strong Windows NTLMv2 Authentication Enabled; Weak LM Disabled'
  desc ' Stong NTLMv2 needs to be enabled '
  describe registry_key('HKLM\System\CurrentControlSet\Control\Lsa') do
    it { should exist }
    its('LmCompatibilityLevel') { should eq 4 }
  end
end

control 'Strong encryption on servers' do
  impact 1.0
  title 'Enable Strong Encryption for Windows Network Sessions on Servers'
  desc 'Servers need to have strong encryption enabled'
  describe registry_key('HKLM\System\CurrentControlSet\Control\Lsa\MSV1_0') do
    it { should exist }
    its('NtlmMinServerSec') { should eq 537_395_200 }
  end
end
```

EXERCISE

# Group Lab: Creating a Custom Profile

**Objective:**

- ✓ Create a custom profile
- ✓ Add controls to profile
- ❑ Test your profile for errors with InSpec
- ❑ Test your profile for functionality with InSpec
- ❑ Upload your compliance profile to Automate

Progress Chef

## GL: Test your profile for errors with InSpec

Now that our custom profile is complete, we can check it for any errors before it is made available to everyone else. All the commands we will run need to be executed at the same level as our profile, so lets move to the compliance directory.

```
PS C:\Users\Administrator\compliance\windows_profile\controls> cd ../..
PS C:\Users\Administrator\compliance> _
```

Progress Chef



7 24

Command:
cd ../..

## GL: Test your profile for errors with InSpec

We will use the **inspec check** command to ensure our profile does not have any errors.

```
PS C:\Users\Administrator\compliance> inspec check .\windows_profile\
Location :   .\windows_profile\
Profile :    windows_profile
Controls :   2
Timestamp :  2022-09-21T15:15:05+00:00
Valid :      true

No errors or warnings
PS C:\Users\Administrator\compliance> _
```

https://docs.chef.io/inspec/cli/#check

Progress Chef



7 25

https://docs.chef.io/inspec/cli/#check

Command:
inspec check .\windows_profile

EXERCISE

# Group Lab: Creating a Custom Profile

**Objective:**

- ✓ Create a custom profile
- ✓ Add controls to profile
- ✓ Test your profile for errors with InSpec
- ❑ Test your profile for functionality with InSpec
- ❑ Upload your compliance profile to Automate

Progress Chef

## GL: Test your profile for functionality with InSpec

With our syntax check passing, we can move on to testing the functionality of the profile. We will use **inspec exec** for this. We should see two passing controls and two failing controls. The specified registry keys exist, but the value is incorrect.

```
PS C:\Users\Administrator\compliance> inspec exec .\windows_profile\

Profile: InSpec Profile (windows_profile)
Version: 0.1.0
Target:  local://
```

```
Profile Summary: 0 successful controls, 2 control failures, 0 controls skipped
Test Summary: 2 successful, 2 failures, 0 skipped
```

https://docs.chef.io/inspec/cli/#exec

Progress Chef



7 27

https://docs.chef.io/inspec/cli/#exec

Command:
inspec exec .\windows_profile

EXERCISE

# Group Lab: Creating a Custom Profile

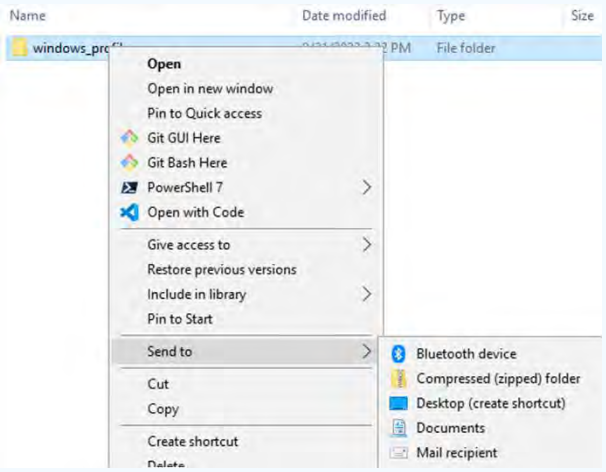
**Objective:**

- ✓ Create a custom profile
- ✓ Add controls to profile
- ✓ Test your profile for errors with InSpec
- ✓ Test your profile for functionality with InSpec
- ❑ Upload your compliance profile to Automate

Progress Chef

## GL: Upload your compliance profile to Automate

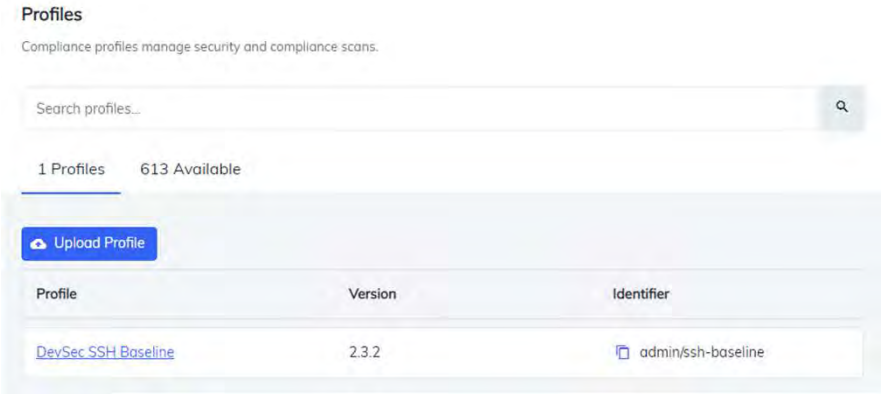
The Chef Automate UI accepts .zip and .tar.gz files for uploading profiles. Navigate to **C:\Users\Administrator\compliance** in Windows Explorer and send **windows_profile** to a zipped folder



## GL: Upload your compliance profile to Automate

We can now upload our profile into Automate. This is done in the **Profiles** window of the **Compliance** tab. Click on the **Upload Profile** button to launch a pop-up window.



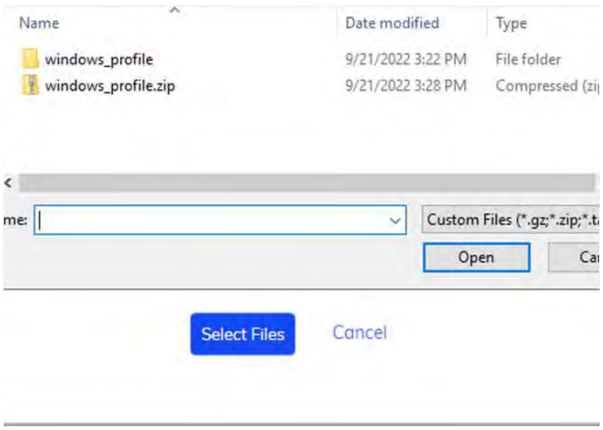

Progress Chef



7 30

## GL: Upload your compliance profile to Automate

On the pop-up window, click the **Select Files** button to launch a file manager window. Navigate to the compliance directory and select the .zip file we created, then click **Open**.



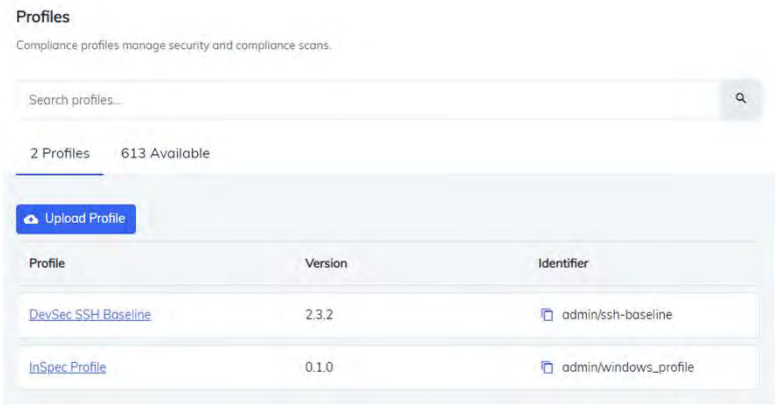

Progress Chef



7 31

## GL: Upload your compliance profile to Automate

After clicking **Open**, there is no indication that the profile is uploading, but it should appear after a short wait.



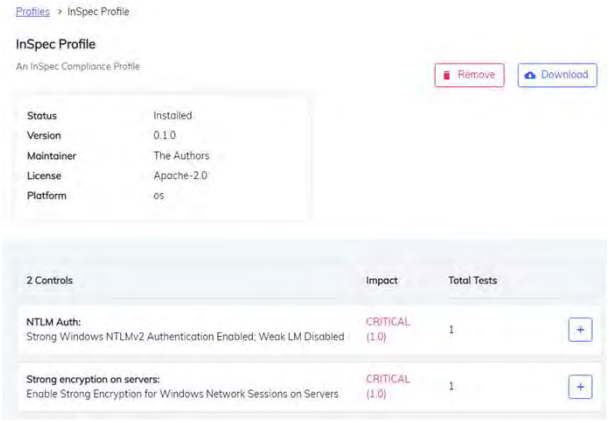

Progress Chef



7 32

## GL: Upload your compliance profile to Automate

We can click on our profile and view the controls the same way we viewed the **DevSec SSH Baseline** profile.




Progress Chef



7 33

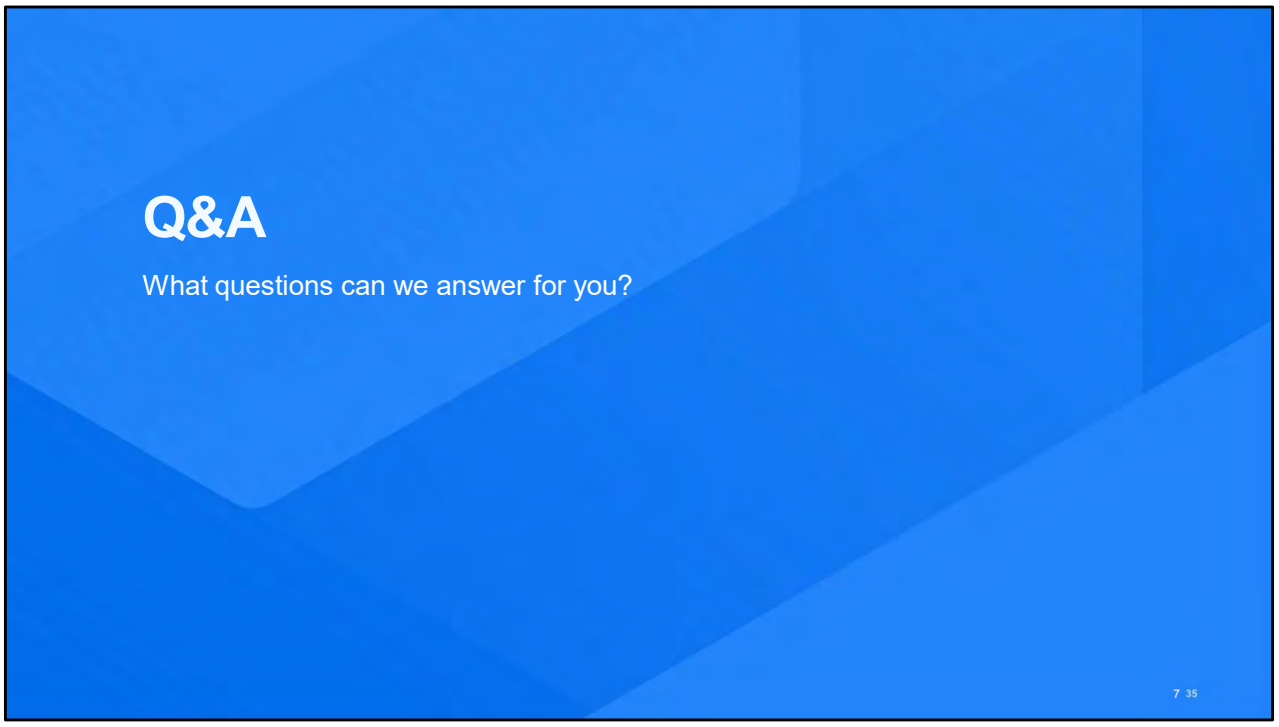
EXERCISE

# Group Lab: Creating a Custom Profile

**Objective:**

- ✓ Create a custom profile
- ✓ Add controls to profile
- ✓ Test your profile for errors with InSpec
- ✓ Test your profile for functionality with InSpec
- ✓ Upload your compliance profile to Automate

Progress Chef


Q&A
What questions can we answer for you?
7 35


Note: We will not be using the Windows Node in any more, it can shutdown now.

Progress Chef

# chef-client Compliance Phase

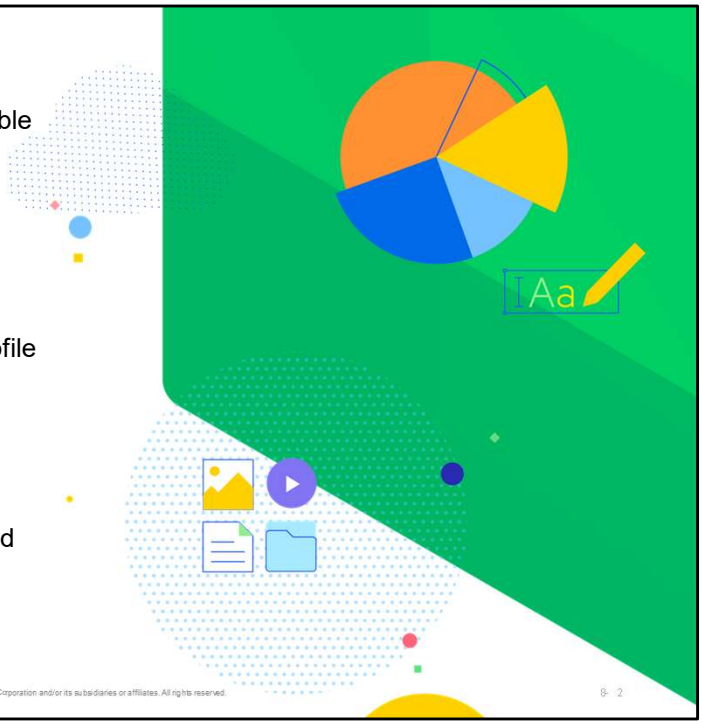
Using Compliance Phase to Automatically Generate Compliance Reports

8- 1

# Objectives

After completing this module, you should be able to:

- Explain the uses of the chef-client's Compliance Phase
- Configure the client-run cookbook to use a "reporter", a "fetcher" and a compliance profile
- Upload the client-run cookbook's Policyfile.lock to Chef server
- Converge a node to run the client-run cookbook
- View the compliance scan results generated by chef-client Compliance Phase




Progress Chef

# Concept

## chef-client Compliance Phase

**chef-client Compliance Phase** runs InSpec compliance profiles as part of a chef-client run.

Earlier in this course we manually ran ad hoc compliance scans against nodes so you could understand how Chef Automate works.

In practice you will probably use the **chef-client Compliance Phase** to automatically run your compliance scans.

https://docs.chef.io/chef_compliance_phase/





https://docs.chef.io/chef_compliance_phase/

# Concept

## chef-client Compliance Phase

**chef-client Compliance Phase** makes it easy to retrieve compliance profiles, execute compliance scans and report results via chef-client runs.

In this way, compliance scans are automatically run each time chef-client runs on a node that is configured with compliance phase.

It downloads configured compliance profiles from various sources like the Chef Automate server (compliance profiles asset store), or Git and reports compliance scan results to the Chef Automate UI.

## Reporters

When you configure compliance phase, you will need to set **reporters** and **fetchers**.

**reporters** specify where you want your scan results go. For example:

- Chef Automate proxied by Chef Infra Server

```
default['audit']['reporter'] = %w(chef-server-automate)
```

- Directly to Chef Automate (requires additional authentication)

```
default['audit']['reporter'] = (chef-automate)
```

Progress Chef





As you'll learn later in this module, you can set the reporters in ~/*cookbookname*/attributes/default.rb.

## Fetchers

**fetchers** specify from where you want to get your compliance profile(s). For example:

- Chef Automate proxied by Chef Infra Server

```
default['audit']['fetcher'] = 'chef-server-automate'
```

- Directly to Chef Automate (requires additional authentication)

```
default['audit']['fetcher'] = 'chef-automate'
```

Progress Chef





As you'll learn later in this module, you can set the fetchers in ~/*cookbookname*/attributes/default.rb.

# Concept

## Node management

Nodes are not required to be managed by Chef to run compliance scans. All of the scans we have executed so far have been on nodes that are not managed by Chef.

However, in order to use **chef-client Compliance Phase**, the node must be managed by a Chef Infra Server. The process of adding a node to a Chef Infra Server is called bootstrapping.

Our first exercise will be logging into our Linux node, connecting the Chef Infra account we created, and then bootstrapping the node.

EXERCISE

# Group Lab: Connect Node to Chef Infra Server

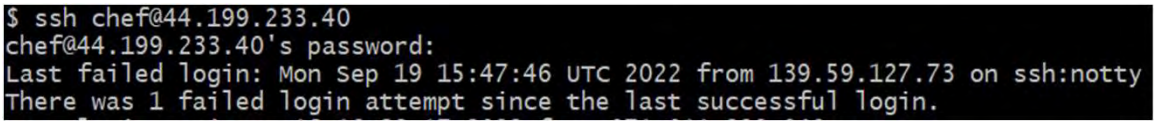
**Objective:**

- ❑ Setup knife
- ❑ Bootstrap Linux node

Progress Chef

## GL: Setup knife

We begin by logging into the Linux node, if you are not already logged in. Open a new terminal window or putty session because we are going to need access to the Automate server and the Linux node at the same time.




Progress Chef



8- 9

**Command:**
ssh chef@IP_ADDRESS

**Password:** Cod3Can!

## GL: Setup knife

Next we need to get the contents of the **atrain.key** file from our Automate server to this node. On the Automate server, use **cat** to view the file. Copy the contents and paste it into a new file on your Linux node named **~/.chef/atrain.pem**.


Progress Chef



8-10

**Command on Automate Server:**
cat atrain.key

**Command on Linux node:**
vi ~/.chef/atrain.pem

Concept

## The Knife Command

**knife** is a command-line tool that provides an interface between the system you work on and the Chef Infra Server. knife helps users to manage:

- Nodes
- Cookbooks and recipes
- Roles, Environments, and Data Bags
- The installation of Chef Infra Client onto nodes

https://docs.chef.io/workstation/knife/



8-11

https://docs.chef.io/workstation/knife/

## GL: Setup knife

Because we are setting up knife for the first time on this machine, we execute **knife configure init-config**. It will ask for two inputs, Chef Server URL and username. These inputs will be used to create a credentials file.

```
[chef@ip-172-31-75-89 ~]$ knife configure init-config
WARNING: No knife configuration file found. See https://docs.chef.io/config_rb_knife.htm
l for details.
Please enter the chef server URL: [https://ip-172-31-75-89.ec2.internal/organizations/my
org] https://ec2-44-201-73-241.compute-1.amazonaws.com/organizations/train
Please enter an existing username or clientname for the API: [chef] atrain
*****

You must place your client key in:
  /home/chef/.chef/atrain.pem
Before running commands with Knife

*****
Knife configuration file written to /home/chef/.chef/credentials
```

Progress Chef



8-12

**Command on Linux node:**
Knife configure init-config

**Inputs**
**Chef Server URL:** AUTOMATE_URL/organizations/train
**Username:** atrain

## GL: Setup knife

Viewing the credentials file that was created, we can see that it is very simple. It defines a username, key, and URL all under the name [default]. The credentials file can store logins for multiple Chef Infra servers, as long as they have unique names. More information about setting up Knife is available at the link below.

```
[chef@ip-172-31-75-89 ~]$ cat .chef/credentials
[default]
client_name     = 'atrain'
client_key      = '/home/chef/.chef/atrain.pem'
chef_server_url = 'https://ec2-44-201-73-241.compute-1.amazonaws.com/organizations/train'
```

https://docs.chef.io/workstation/knife_setup/

Progress Chef



8-13

**Command on Linux node:**
cat .chef/credentials

## GL: Setup knife

Knife should be ready to communicate with the Chef Infra Server now. The first command we want to use is **knife ssl fetch**. This should retrieve the certificate and place it in our trusted_certs directory.

```
[chef@ip-172-31-75-89 ~]$ knife ssl fetch
WARNING: Certificates from ec2-44-201-73-241.compute-1.amazonaws.com will be fetched and placed in
 your trusted_cert
        directory (/home/chef/.chef/trusted_certs).

        Knife has no means to verify these are the correct certificates. You should
        verify the authenticity of these certificates after downloading.
Adding certificate for ec2-44-201-73-241_compute-1_amazonaws_com in /home/chef/.chef/trusted_certs
/ec2-44-201-73-241_compute-1_amazonaws_com.crt
```

Progress Chef



8-14

**Command on Linux node:**
knife ssl fetch

## GL: Setup knife

Now we want to verify that everything is ok with the certificate we pulled. We can do this with **knife ssl check**.

```
[chef@ip-172-31-75-89 ~]$ knife ssl check
Connecting to host ec2-44-201-73-241.compute-1.amazonaws.com:443
Successfully verified certificates from `ec2-44-201-73-241.compute-1.amazonaws.com'
```

Progress Chef



8-15

**Command on Linux node:**
knife ssl check

## GL: Setup knife

With everything working with our certificate, we can try out another knife command. **Knife client list** will return all of the clients on our Chef Infra Server.

It is important to note that clients and nodes are not the same thing in Chef Infra Server. Nodes are items managed by Chef, while clients are items that connect to the Chef Infra Server API.

```
[chef@ip-172-31-75-89 ~]$ knife client list
train-validator
```

Progress Chef



8-16

**Command on Linux node:**
knife client list

EXERCISE

# Group Lab: Connect Node to Chef Infra Server

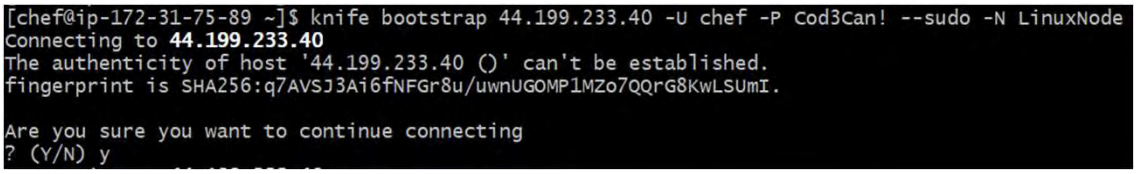
**Objective:**

- ✓ Setup knife
- ❑ Bootstrap Linux node

Progress Chef

## GL: Bootstrap Linux node

Now that we have knife setup we can bootstrap our Linux node. The **knife bootstrap** command is used to connect a node to a Chef Infra Server. We need to give this command the node IP address, a username and password that can SSH into the node, and the name we want to give the node.




https://docs.chef.io/workstation/knife_bootstrap/

Progress Chef



8-18

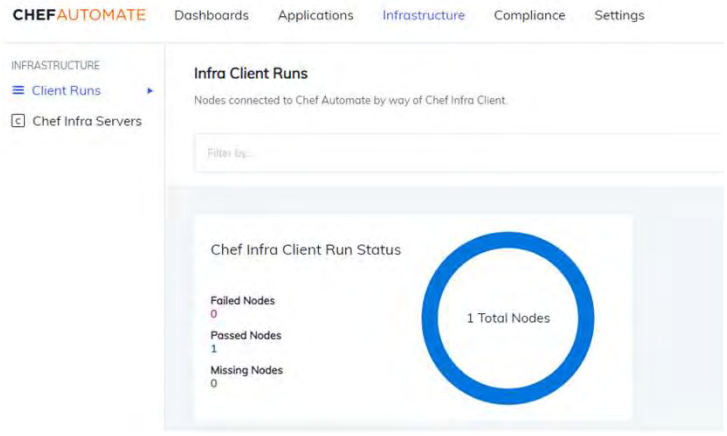
**Note:** In practice you would rarely, if ever, bootstrap your workstation node.

https://docs.chef.io/workstation/knife_bootstrap/

**Command on Linux node:**
knife bootstrap IP_ADDRESS -U chef -P Cod3Can! --sudo -N LinuxNode

## GL: Bootstrap Linux node

After the bootstrap process is complete we can verify that it worked by going to the **Infrastructure** section and looking at the **Client Runs** window.




Progress Chef



8-19

EXERCISE

# Group Lab: Connect Node to Chef Infra Server

**Objective:**

- ✓ Setup knife
- ✓ Bootstrap Linux node

Progress Chef

Concept

## Reviewing cookbook

We will use the **client-run** cookbook in our cookbooks directory to implement **chef-client Compliance Phase**. Before we do anything let's review the cookbook to see what is required.



8- 21

## View the Cookbook's Default Recipe

Start by moving into the **client-run** cookbook, then print **recipes/default.rb** to the terminal. Line 7 adds an additional compliance profile and line 9 starts a cron resource. More information on profiles is available at the link below.

https://docs.chef.io/inspec/profiles/

```
1  #
2  # Cookbook:: client-run
3  # Recipe:: default
4  #
5  # Copyright:: 2021, John Tonello, All Rights Reserved.
6
7  include_profile 'client-run::client-run'
8
9  cron 'chef-client' do
10   minute '*/59'
11   command '/usr/bin/chef-client'
12   action :create
13 end
```

Progress 8- 22 

https://docs.chef.io/inspec/profiles/

**Note:** The code in the slide is shown in an IDE to make it more readable.

**Command on Linux node:**
cd cookbooks/client-run/
cat recipes/default.rb

## View the Compliance Profile

The compliance profile referenced in the default recipe in located at **compliance/profiles/client-run/controls/default.rb**. This profile has some logic in it, but is simply checking that **chef-client** exists and is scheduled to run.

```
# copyright: 2021, John Tonello

control 'client-run' do
  impact 0.7
  title 'Run the chef-client once an hour'
  if os.windows?
    describe windows_task('run-chef-client') do
      it { should exist }
      it { should be_enabled }
    end
  else
    describe crontab do
      its('commands') { should include '/usr/bin/chef-client' }
      its('minutes') { should include '*/59' }
    end
  end
end
```

Progress 8- 23 

**Command on Linux node:**
cat compliance/profiles/client-run/controls/default.rb

## View the Attributes file

Four important values are set here:

1. First we set the value of **compliance_phase** to true
2. The reporter tells chef-client where to send the results of the scan
3. The fetcher tells chef-client where to get the profile to scan with
4. The final line tells the chef-client what profile to scan with

```
default['audit']['compliance_phase'] = true
default['audit']['reporter'] = %w(chef-server-automate cli)
default['audit']['fetcher'] = 'chef-server'

default['audit']['profiles']['cis-centos7-level1'] = {
 compliance: 'admin/cis-centos7-level1',
}
```

EXERCISE

# Group Lab: Execute chef-client Compliance Phase

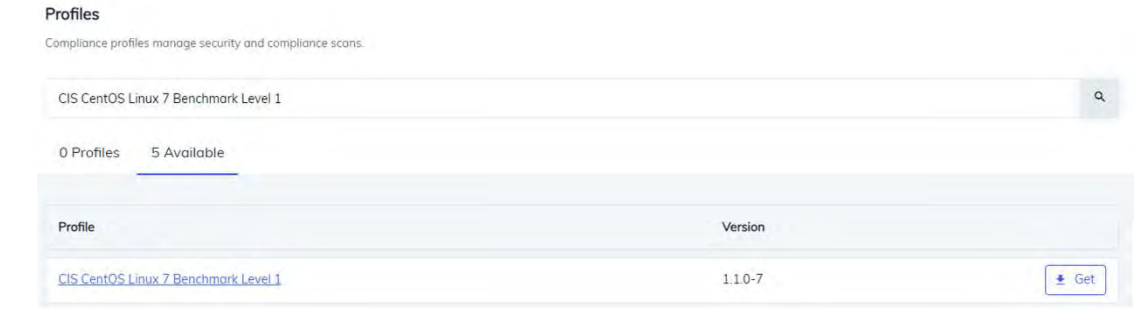
**Objective:**

- ❑ Activate required profile
- ❑ Create API token
- ❑ Add API token to node
- ❑ Apply policyfile to node
- ❑ View compliance scan results

Progress Chef

## GL: Activate required profile

We need to make the profile referenced in the attributes file available. Go to the **Profiles** window of the **Compliance** section, then go to the **Available** tab and search for **CIS CentOS Linux 7 Benchmark Level 1**. Click on **Get** to the right side of the top profile.




Progress Chef



8-26

## GL: Activate required profile

Looking back at the **Profiles** tab, we see that the profile is now available. It is important that the profile identifier matches exactly what is in the attributes file.


Progress Chef



8-27

EXERCISE

# Group Lab: Execute chef-client Compliance Phase

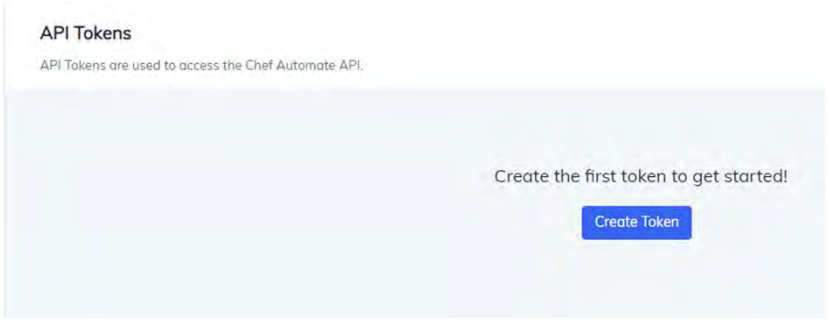
**Objective:**

- ✓ Activate required profile
- ❑ Create API token
- ❑ Add API token to node
- ❑ Apply policyfile to node
- ❑ View compliance scan results

Progress Chef

## GL: Create API token

Our cookbook tells **chef-client** to send the compliance scan results directly to the Chef Automate server. In order to do this the node needs an API token added to its **client.rb** file. In the Automate UI, navigate to the **API Tokens** window in the **Settings** section. Then click on the **Create Token** button.



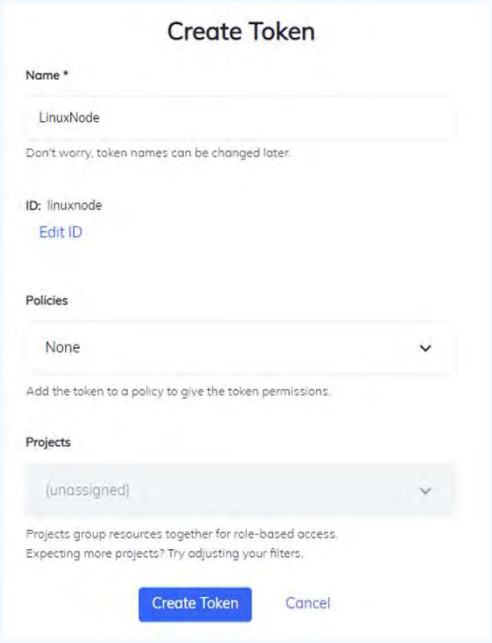

Progress Chef



8-29

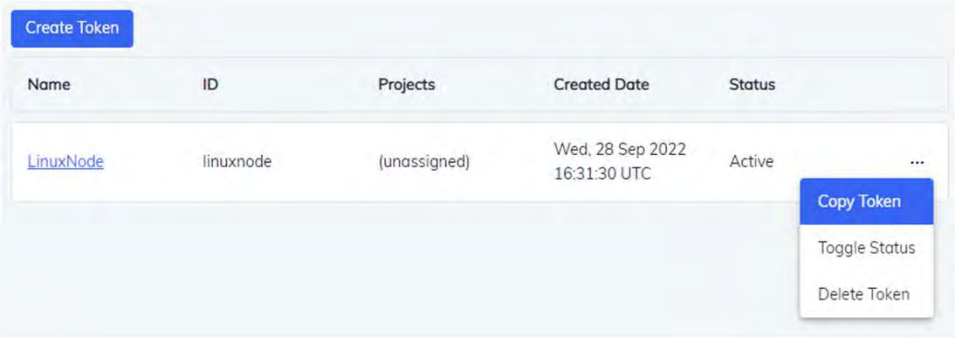
## GL: Create API token

Clicking on **Create Token** launches a pop-up window. The only required field is **Name**, enter a name and click the **Create Token** button.



## GL: Create API token

Our API token is now ready to go. In order to use it, click on the three dots at the far right side and select **Copy Token**.




Progress Chef



8-31

EXERCISE

# Group Lab: Execute chef-client Compliance Phase

**Objective:**

- ✓ Activate required profile
- ✓ Create API token
- ❑ Add API token to node
- ❑ Apply policyfile to node
- ❑ View compliance scan results

Progress Chef

## GL: Add API token to node

The API token is added to the **client.rb** file on our node. Back on our Linux node, edit the file **/etc/chef/client.rb** to include the three settings shown on lines 8-10. The values for **.server_url** and **.token** will be different for everyone.

```
1  chef_server_url  "https://ec2-44-201-73-241.compute-1.amazonaws.com/organizations/train"
2  validation_client_name "train-validator"
3  chef_license "accept"
4  log_location   STDOUT
5  node_name "LinuxNode"
6  trusted_certs_dir "/etc/chef/trusted_certs"
7
8  ssl_verify_mode :verify_none
9  data_collector.server_url "https://44.201.73.241/data-collector/v0/"
10 data_collector.token "MT14M9kfbfSis2_8Fby_oTeAosQ="
```

Progress Chef



8-33

Code:
ssl_verify_mode :verify_none
data_collector.server_url "https://AUTOMATE_IP_ADDRESS/data-collector/v0/"
data_collector.token “TOKEN_COPIED_FROM_AUTOMATE”

EXERCISE

# Group Lab: Execute chef-client Compliance Phase

**Objective:**

- ✓ Activate required profile
- ✓ Create API token
- ✓ Add API token to node
- ❑ Apply policyfile to node
- ❑ View compliance scan results

Progress Chef

## GL: Apply policyfile to node

Applying the policyfile to our Linux node takes a few steps. Move to the **client-run** cookbook if you are not there already. Then, update the **Policyfile.lock.json** file.

```
[chef@ip-172-31-75-89 client-run]$ chef update Policyfile.rb
Building policy client-run
Expanded run list: recipe[client-run::default]
Caching Cookbooks...
Installing client-run >= 0.0.0 from path

Lockfile written to /home/chef/automate-repo/cookbooks/client-run/Policyfile.lock.json
Policy revision id: 41ff1b5dbfeb99ce0e4f9b8a7906ff9b2dcbfe9180f6e971425eed63b2a00bc1
```

Progress Chef



8-35

**Command on Linux node:**
chef update Policyfile.rb

## GL: Apply policyfile to node

Now we can push the updated .lock.json file to the Chef Infra Server with **chef push prod Policyfile.lock.json**. We can verify that the policyfile was pushed successfully with **chef show-policy**.

```
[chef@ip-172-31-75-89 client-run]$ chef push prod Policyfile.lock.json
Uploading policy client-run (41ff1b5dbf) to policy group prod
Uploaded client-run 0.1.0 (5321eb4c)
```

```
[chef@ip-172-31-75-89 client-run]$ chef show-policy
client-run
==========

* prod:  41ff1b5dbf
```

Progress Chef



8-36

**Command on Linux node:**
chef push prod Policyfile.lock.json
chef show-policy

## GL: Apply policyfile to node

The next step is to assign the client-run policyfile to our Linux node and verify that it was assigned correctly.

```
[chef@ip-172-31-75-89 client-run]$ knife node policy set LinuxNode prod client-run
Successfully set the policy on node LinuxNode
[chef@ip-172-31-75-89 client-run]$
[chef@ip-172-31-75-89 client-run]$ knife node show LinuxNode
Node Name:   LinuxNode
Policy Name:  client-run
Policy Group: prod
FQDN:        ip-172-31-75-89.ec2.internal
IP:          44.199.233.40
Run List:
Recipes:
Platform:    centos 7.6.1810
Tags:
```

Progress Chef



8-37

**Command on Linux node:**
knife node policy set LinuxNode prod client-run
knife node show LinuxNode

## GL: Apply policyfile to node

Finally, we can run **chef-client**. In the output of **chef-client**, we can see when the Infra Phase ends and the Compliance Phase begins.

We also see the profile we included at the very end.

```
[chef@ip-172-31-75-89 client-run]$ sudo chef-client
Chef Infra Client, version 17.10.0
Patents: https://www.chef.io/patents
Infra Phase starting
Resolving cookbooks for run list: ["client-run::default@0.1.0 (5321eb4)"]
Synchronizing cookbooks:
  - client-run (0.1.0)
Installing cookbook gem dependencies:
Compiling cookbooks...
Loading Chef InSpec profile files:
  - client-run::client-run (0.1.1)
Loading Chef InSpec input files:
Loading Chef InSpec waiver files:
Converging 1 resources
Recipe: client-run::default
  * cron[chef-client] action create
    - add crontab entry for cron[chef-client]

Running handlers:
Running handlers complete
Infra Phase complete, 1/1 resources updated in 02 seconds

Compliance report:
 * CIS CentOS Linux 7 Benchmark Level 1
     Create Separate Partition for /tmp
       - Mount /tmp is expected to be mounted
```

```
* Run chef-client
    Run the chef-client once an hour
      + crontab for current user commands is expected to include "/usr/bin/chef-client"
      + crontab for current user minutes is expected to include "*/59"
```

Progress 8- 38 

**Command on Linux node:**
sudo chef-client

EXERCISE

# Group Lab: Execute chef-client Compliance Phase

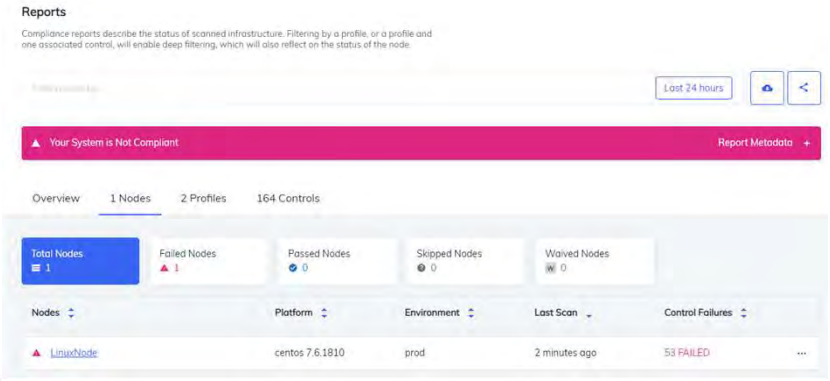
**Objective:**

- ✓ Activate required profile
- ✓ Create API token
- ✓ Add API token to node
- ✓ Apply policyfile to node
- ❑ View compliance scan results

Progress Chef

## GL: View compliance scan results

The results of the compliance scan were sent to the Chef Automate server. We can verify that by going to the **Nodes** tab in the **Reports** window in the **Compliance** section.




Progress Chef



8-40

## GL: View compliance scan results

Clicking on the node name will take us to the report. Notice that two profiles were used in the scan.

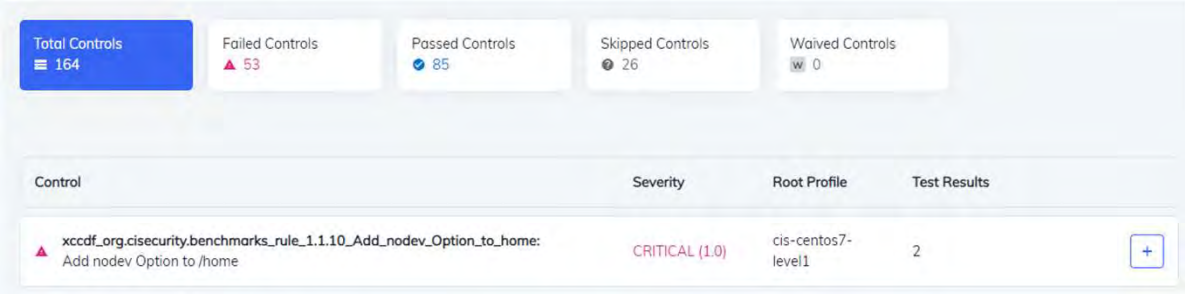

Progress Chef



8-41

## GL: View compliance scan results

Further down the report we can see a few ways to quickly filter the results. We can focus on Failed Controls, Passed Controls, Skipped Controls, and Waived Controls. Below those filters we can dig into the individual controls in the profile.



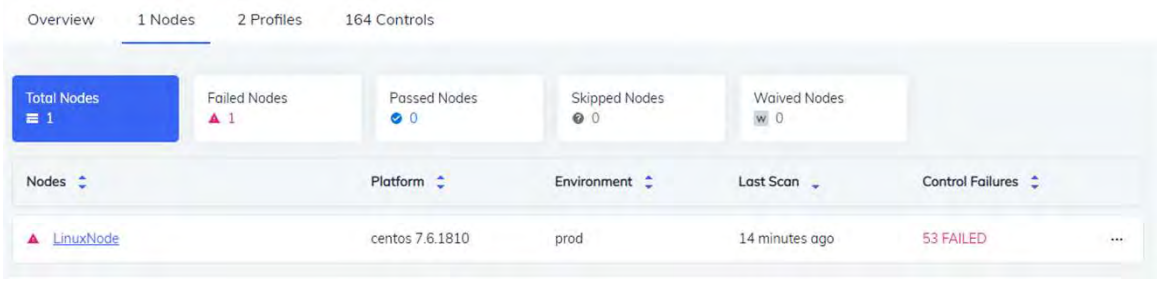

Progress Chef



8-42

## GL: View compliance scan results

Because we schedule chef-client to run every 59 minutes, this compliance scan will continue to run and stay up-to-date.




Progress Chef



8-43

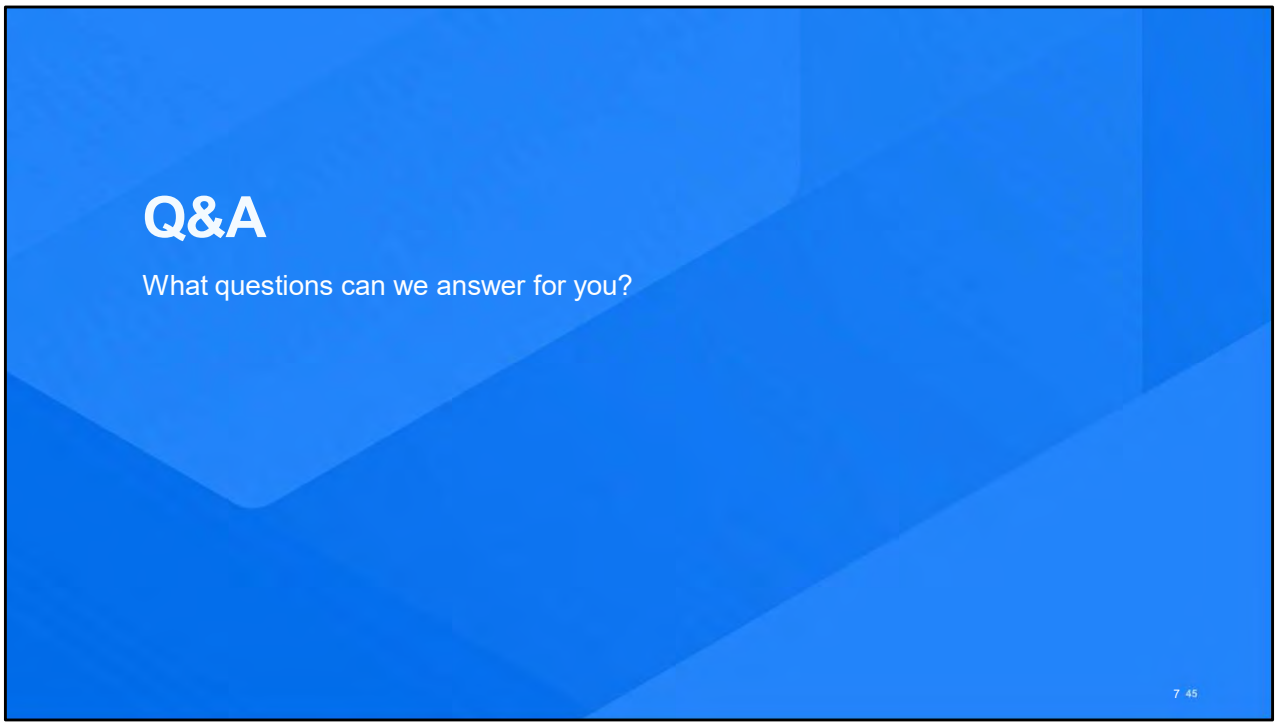
EXERCISE

# Group Lab: Execute chef-client Compliance Phase

**Objective:**

- ✓ Activate required profile
- ✓ Create API token
- ✓ Add API token to node
- ✓ Apply policyfile to node
- ✓ View compliance scan results

Progress Chef


Q&A
What questions can we answer for you?
7 45


Progress Chef

Progress Chef

# Compliance Profile Waivers

Creating an exception when needed

# Objectives

After completing this module, you should be able to:

- Explain what a waiver is
- Create a waiver file
- Use a waiver file with inspec exec
- Use a waiver file in a compliance phase scan

Progress Chef


https://docs.chef.io/inspec/waivers/

# Concept

## Waiver file usage

Waiver files can be used in two different ways. The format for the waiver file does not change depending on how it is used.

- When running the **inspec exec** command by using the **--waiver-file** switch
- Declared in a cookbook's attributes file

EXERCISE

# Group Lab: Manual scan with waiver file

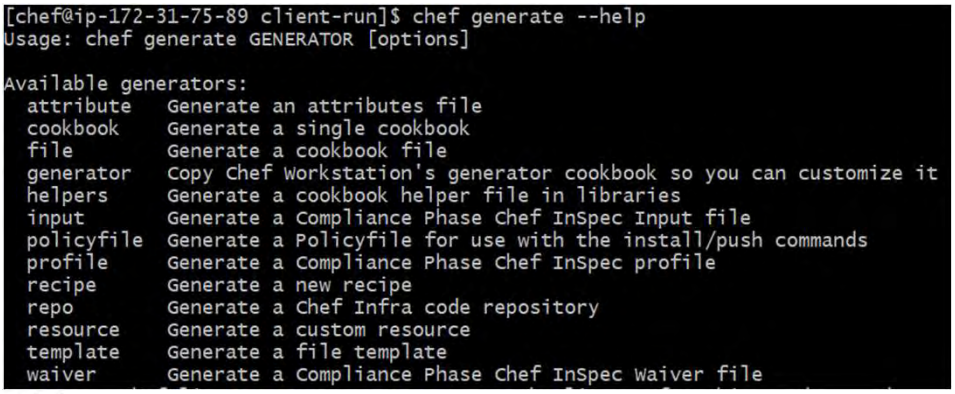
**Objective:**

- ❑ Create waiver file
- ❑ Use waiver file on a manual compliance scan

Progress Chef

## GL: Create waiver file

As with many other components in Chef, a waiver file can be created by using the **chef generate** command.




Progress Chef



9 6

Command:
chef generate --help

## GL: Create waiver file

We can find the full syntax by running **chef generate waiver --help**. Here we can see that the path to the cookbook is optional and the only required parameter is the name.

```
[chef@ip-172-31-75-89 client-run]$ chef generate waiver --help
Usage: chef generate waiver [path/to/cookbook] NAME [options]
        --chef-license ACCEPTANCE    Accept the license for this product and any contained product
s ('accept', 'accept-no-persist', or 'accept-silent')
    -c, --config CONFIG_FILE         Path to configuration file
    -C, --copyright COPYRIGHT        Name of the copyright holder - defaults to 'The Authors'
    -D, --debug                      Enable stacktraces and other debug output
    -m, --email EMAIL                Email address of the author - defaults to 'you@example.com'
    -a, --generator-arg KEY=VALUE    Use to set arbitrary attribute KEY to VALUE in the code_gener
ator cookbook
    -h, --help                       Show this message
    -I, --license LICENSE            all_rights, apachev2, mit, gplv2, gplv3 - defaults to all_rig
hts
```

Progress Chef



9 7

Command:
chef generate waiver --help

## GL: Create waiver file

From inside the **client-run** cookbook execute the command **chef generate waiver exemptions**. The output tells us that a new directory was created for our waiver, and the waiver file itself was also created.

```
[chef@ip-172-31-75-89 client-run]$ chef generate waiver exemptions
Recipe: code_generator::waiver
  * directory[/home/chef/automate-repo/cookbooks/client-run/compliance/waivers] action create
    - create new directory /home/chef/automate-repo/cookbooks/client-run/compliance/waivers
  * template[/home/chef/automate-repo/cookbooks/client-run/compliance/waivers/exemptions.yml] acti
on create
    - create new file /home/chef/automate-repo/cookbooks/client-run/compliance/waivers/exemptions.
yml
    - update content in file /home/chef/automate-repo/cookbooks/client-run/compliance/waivers/exem
ptions.yml from none to 8ded67
    (diff output suppressed by config)
```

Progress Chef

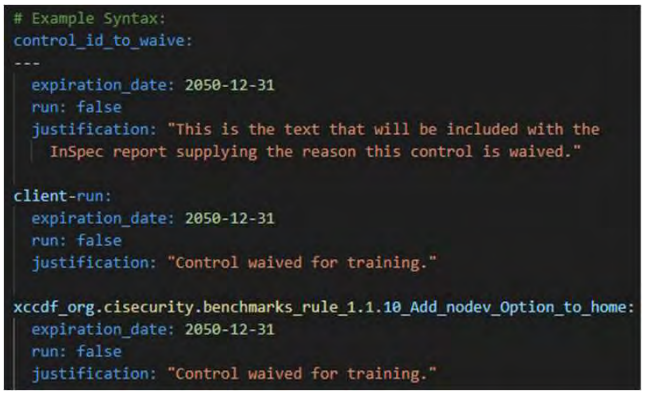


9 8

Command:
chef generate waiver exemptions

## GL: Create waiver file

Our new waiver file has an example waiver for us. Using this we can create two new waivers for some of the controls in our compliance scan. Add the code shown and given below to your waiver file.




Progress Chef



9 9

Note: Copy and paste can sometimes pull invisible characters. If you receive an error, try to delete the spaces and type them in again.

Code:

```
client-run:
  expiration_date: 2050-12-31
  run: false
  justification: "Control waived for training."

xccdf_org.cisecurity.benchmarks_rule_1.1.10_Add_nodev_Option_to_home:
  expiration_date: 2050-12-31
  run: false
  justification: "Control waived for training."
```

EXERCISE

# Group Lab: Manual scan with waiver file

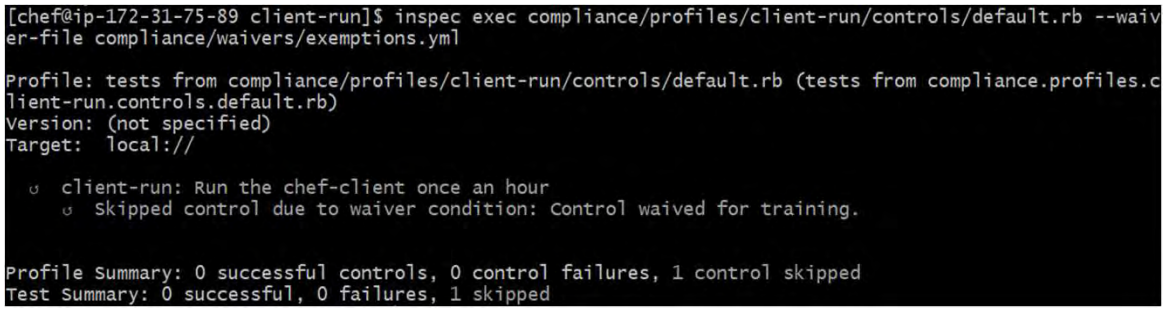
**Objective:**

- ✓ Create waiver file
- ❑ Use waiver file on a manual compliance scan

Progress Chef

## GL: Use waiver file on a manual compliance scan

Using the command shown below we can see the results of the scan. The only control in the profile was exempted because of our waiver file. We are given the justification for the control being skipped and the summary shows one control skipped.




Progress Chef



9 11

Command:
inspec exec compliance/profiles/client-run/controls/default.rb --waiver-file compliance/waivers/exemptions.yml

EXERCISE

# Group Lab: Manual scan with waiver file

**Objective:**

- ✓ Create waiver file
- ✓ Use waiver file on a manual compliance scan

Progress Chef

Concept

## Including Waiver File In Compliance Phase Scan

In order to use a waiver file in the compliance phase, the waiver file needs to be on the node so **chef-client** can read it.

To accomplish this we are first going to create a cookbook file using the **chef generate file** command. Then we are going to send it to the node with a cookbook_file resource in our default recipe.

https://docs.chef.io/resources/cookbook_file/



9.13

https://docs.chef.io/resources/cookbook_file/

EXERCISE

# Group Lab: Compliance Phase with waiver file

**Objective:**

- ❑ Create cookbook file
- ❑ Update default recipe
- ❑ Update attributes file
- ❑ Update Policyfile
- ❑ Run chef-client
- ❑ View compliance scan results

Progress Chef

## GL: Create cookbook file

As with most components of chef, we can create a cookbook file with the **chef generate** command. The first command shown creates the files directory in your cookbook and a blank file named **exemptions.yml**. With the second command we are overwriting the blank file with the waiver file we created earlier.

```
[chef@ip-172-31-75-89 client-run]$ chef generate file exemptions.yml
Recipe: code_generator::cookbook_file
  * directory[/home/chef/automate-repo/cookbooks/client-run/files/] action create
    - create new directory /home/chef/automate-repo/cookbooks/client-run/files/
  * template[/home/chef/automate-repo/cookbooks/client-run/files/exemptions.yml] action create
    - create new file /home/chef/automate-repo/cookbooks/client-run/files/exemptions.yml
    - update content in file /home/chef/automate-repo/cookbooks/client-run/files/exemptions.yml from none to e3b0c4
    (diff output suppressed by config)
[chef@ip-172-31-75-89 client-run]$
[chef@ip-172-31-75-89 client-run]$ mv compliance/waivers/exemptions.yml files/exemptions.yml
```

Progress Chef



9 15

Commands:
chef generate file exemptions.yml
mv compliance/waivers/exemptions.yml files/exemptions.yml

EXERCISE

# Group Lab: Compliance Phase with waiver file

**Objective:**

- ✓ Create cookbook file
- ❑ Update default recipe
- ❑ Update attributes file
- ❑ Update Policyfile
- ❑ Run chef-client
- ❑ View compliance scan results

Progress Chef

## GL: Update default recipe

Next we need to create a cookbook_file resource in our default recipe. This resource creates a copy of **files/exemptions.yml** on our node at **/tmp/exemptions.yml**.

```
include_profile 'client-run::client-run'

cron 'chef-client' do
  minute '*/59'
  command '/usr/bin/chef-client'
  action :create
end

cookbook_file '/tmp/exemptions.yml' do
  source 'exemptions.yml'
  action :create
end
```

Progress Chef



9 17

Code:

```
cookbook_file '/tmp/exemptions.yml' do
  source 'exemptions.yml'
  action :create
end
```

EXERCISE

# Group Lab: Compliance Phase with waiver file

**Objective:**

- ✓ Create cookbook file
- ✓ Update default recipe
- ❑ Update attributes file
- ❑ Update Policyfile
- ❑ Run chef-client
- ❑ View compliance scan results

Progress Chef

## GL: Update attributes file

Now that the file will be available on our node we can create an attribute in our attributes file that points to the waiver file.

```
1  default['audit']['reporter'] = %w(chef-server-automate cli)
2  default['audit']['fetcher'] = 'chef-server'
3
4  default['audit']['profiles']['cis-centos7-level1'] = {
5   compliance: 'admin/cis-centos7-level1',
6  }
7
8  default['audit']['waiver_file'] = '/tmp/exemptions.yml'
```

Progress Chef



9 19

Note: You should not place your waivers in /tmp. Every organization will have their own standards, we are using /tmp as an example.

Code:
default['audit']['waiver_file'] = '/tmp/exemptions.yml'

EXERCISE

# Group Lab: Compliance Phase with waiver file

**Objective:**

- ✓ Create cookbook file
- ✓ Update default recipe
- ✓ Update attributes file
- ❑ Update Policyfile
- ❑ Run chef-client
- ❑ View compliance scan results

Progress Chef

## GL: Update policyfile

We have made several changes to our cookbook, so we need to update the policyfile and send the new lock.json file to our Chef Infra Server. The commands shown below will take care of those steps for us.

```
[chef@ip-172-31-75-89 client-run]$ chef update Policyfile.rb
Building policy client-run
Expanded run list: recipe[client-run::default]
Caching Cookbooks...
Installing client-run >= 0.0.0 from path

Lockfile written to /home/chef/automate-repo/cookbooks/client-run/Policyfile.lock.json
Policy revision id: e21b926ca68900e01bbcbb1c32b54c5e2325b68467dae3c8d6480eea724a4e06
[chef@ip-172-31-75-89 client-run]$
[chef@ip-172-31-75-89 client-run]$
[chef@ip-172-31-75-89 client-run]$
[chef@ip-172-31-75-89 client-run]$ chef push prod Policyfile.lock.json
Uploading policy client-run (e21b926ca6) to policy group prod
Uploaded client-run 0.1.0 (53306d38)
```

Progress Chef



9 21

Commands:
chef update Policyfile.rb
chef push prod Policyfile.lock.json

EXERCISE

# Group Lab: Compliance Phase with waiver file

**Objective:**

- ✓ Create cookbook file
- ✓ Update default recipe
- ✓ Update attributes file
- ✓ Update Policyfile
- ❑ Run chef-client
- ❑ View compliance scan results

Progress Chef

## GL: Run chef-client

Now we can run **chef-client** to apply the updated policyfile. We can see that this time a waiver file is loaded.

```
[chef@ip-172-31-75-89 client-run]$ sudo chef-client
Chef Infra Client, version 17.10.0
Patents: https://www.chef.io/patents
Infra Phase starting
Resolving cookbooks for run list: ["client-run::default@0.1.0 (b488ab2)"]
Synchronizing cookbooks:
  - client-run (0.1.0)
Installing cookbook gem dependencies:
Compiling cookbooks...
Loading Chef InSpec profile files:
  - client-run::client-run (0.1.1)
Loading Chef InSpec input files:
Loading Chef InSpec waiver files:
  - client-run::exemptions
Converging 1 resources
Recipe: client-run::default
  * cron[chef-client] action create (up to date)

Running handlers:
Running handlers complete
Infra Phase complete, 0/1 resources updated in 02 seconds
```

Progress Chef



9_23

Command:
sudo chef-client

EXERCISE

# Group Lab: Compliance Phase with waiver file

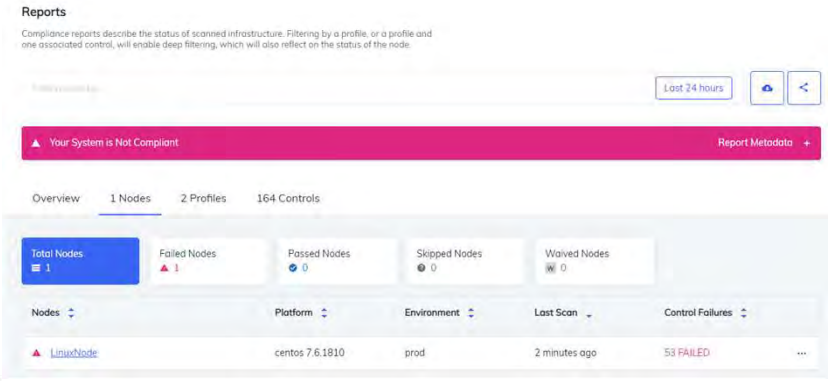
**Objective:**

- ✓ Create cookbook file
- ✓ Update default recipe
- ✓ Update attributes file
- ✓ Update Policyfile
- ✓ Run chef-client
- ❑ View compliance scan results

Progress Chef

## GL: View compliance scan results

The results of the compliance scan were sent to the Chef Automate server. We can verify that by going to the **Nodes** tab in the **Reports** window in the **Compliance** section.




Progress Chef



9_25

## GL: View compliance scan results

Clicking on the node name will take us to the report. Notice that two profiles were used in the scan.

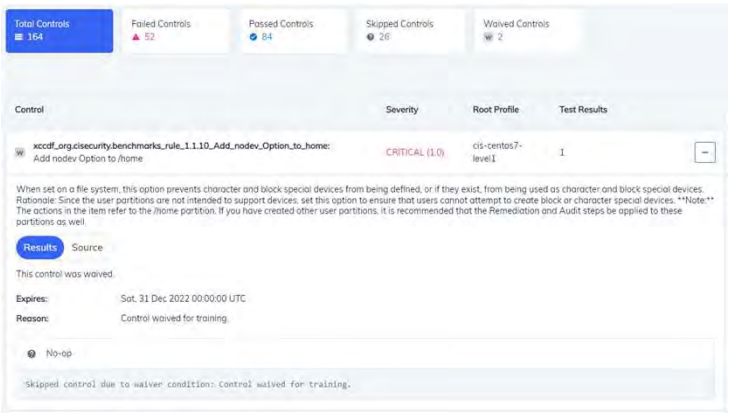

Progress Chef



9 26

## GL: View compliance scan results

Below the summary we can see that two controls were skipped. When we expand the first control, we can see the justification from our waiver file.




Progress Chef



9 27

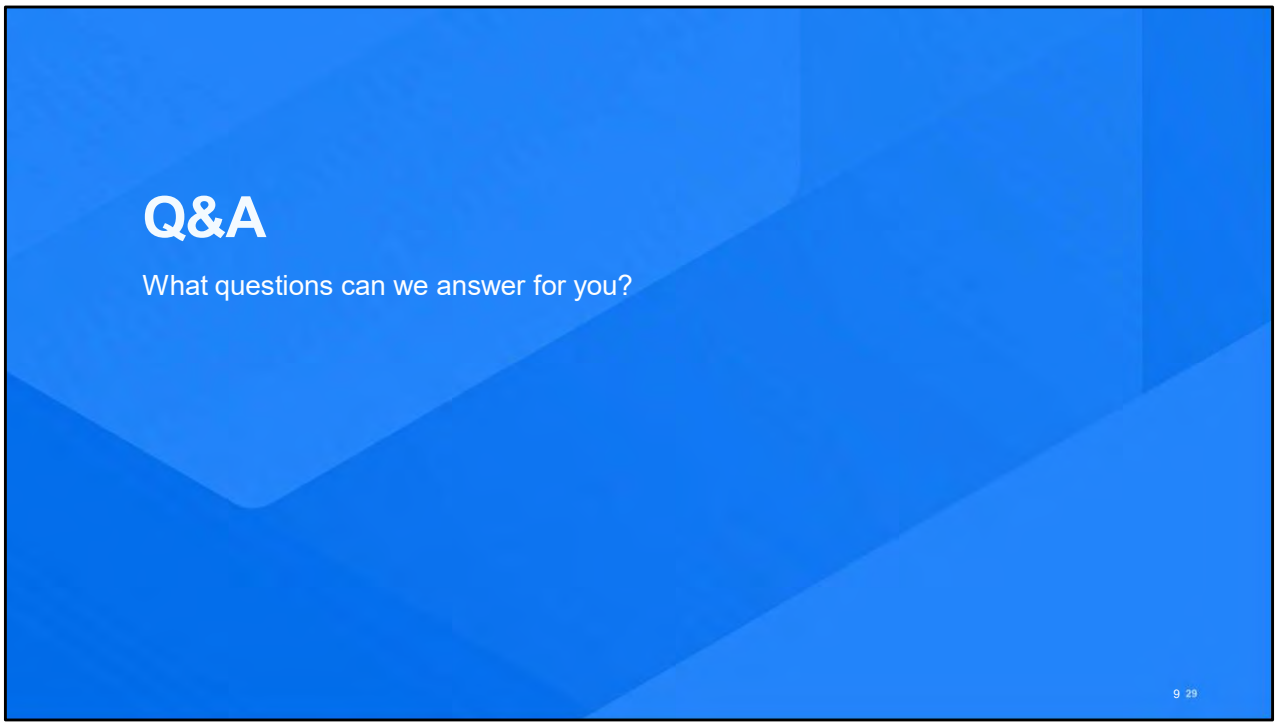
EXERCISE

# Group Lab: Compliance Phase with waiver file

**Objective:**

- ✓ Create cookbook file
- ✓ Update default recipe
- ✓ Update attributes file
- ✓ Update Policyfile
- ✓ Run chef-client
- ✓ View compliance scan results

Progress Chef


Q&A
What questions can we answer for you?
9 29


Progress Chef

Progress Chef

# Expanded Operations

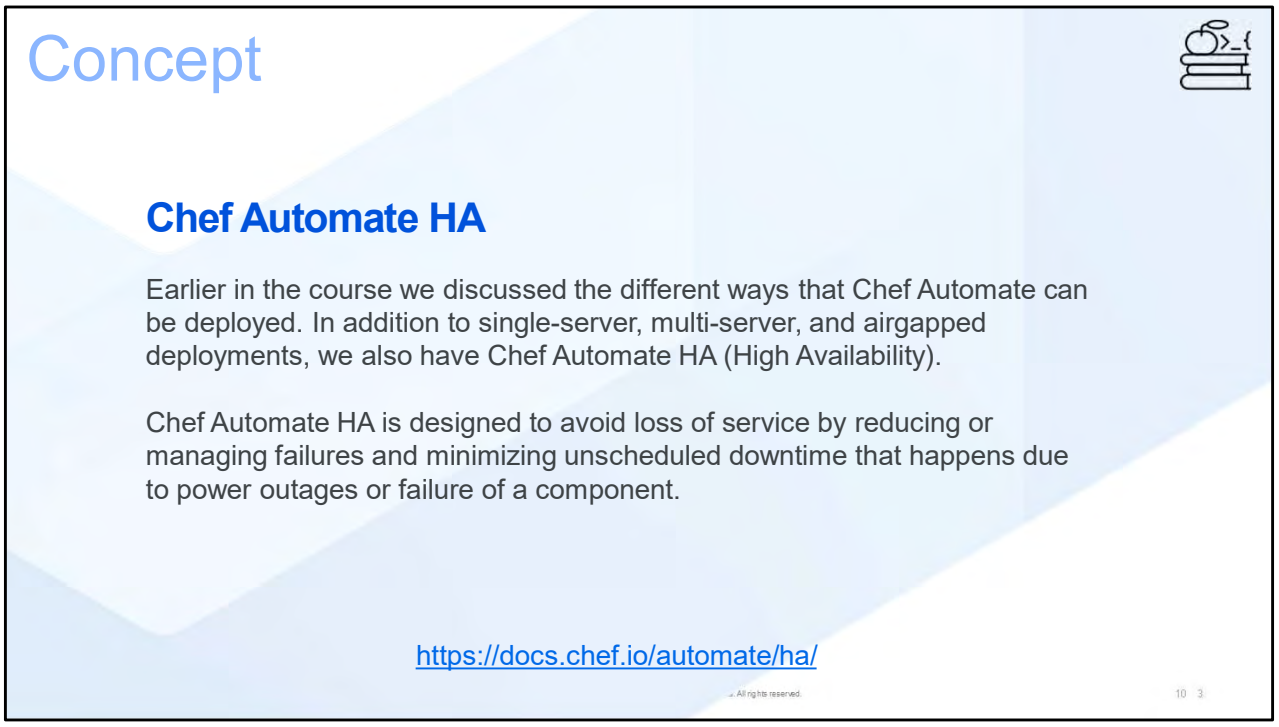
Too much goodness for one class

# Objectives

After completing this module, you should:

- Explain the basic architecture of Chef Automate HA
- Describe what is needed to connect to the Chef Automate API
- Explain what is included with Chef Premium Content
- Describe the benefits of Chef Cloud Security

Progress Chef




https://docs.chef.io/automate/ha/

## Chef Automate HA topology

Frontend

- Chef Automate and Chef Infra Server clusters – Minimum of two nodes in each

Backend

- Postgres and Opensearch clusters – Minimum of three nodes in each

Others

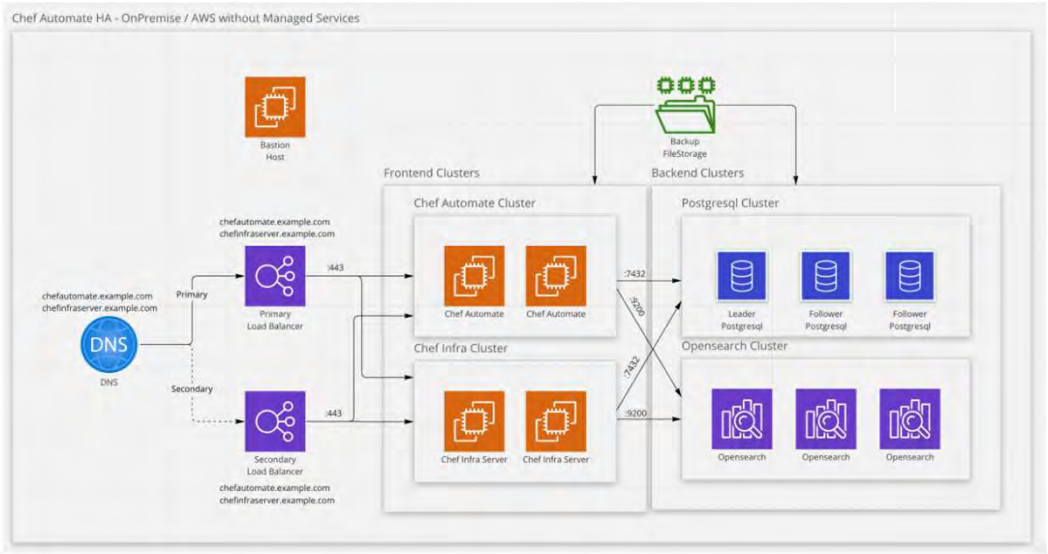
- Bastion Host – Used for maintenance of Chef Automate HA setup
- File storage for backup – Backups for all systems stored here
- Load balancers – One or two balancers for accessing the Automate cluster

Progress Chef



10 4

## Chef Automate HA topology example




Progress Chef



10 5

## Chef Automate HA requirements

Supported Operating Systems

- RHEL 7 or 8
- Ubuntu 16.04, 18.04, or 20.04
- Centos 7
- Amazon Linux 2
- SUSE Linux Enterprise Server 12

Hardware requirements depend on many factors. Sizing guidelines and a sizing calculator are available at:

https://docs.chef.io/automate/ha_platform_support/#hardware-requirements

Progress Chef



10 6

https://docs.chef.io/automate/ha_platform_support/#hardware-requirements

## Chef Automate HA deployment types

Automate HA is able to be deployed in three different scenarios:

- On-Premise – Using physical or virtual machines that reside in your environment
- AWS – Using AWS EC2 instances
- AWS Managed Services – Using AWS managed services such as RDS

Progress Chef



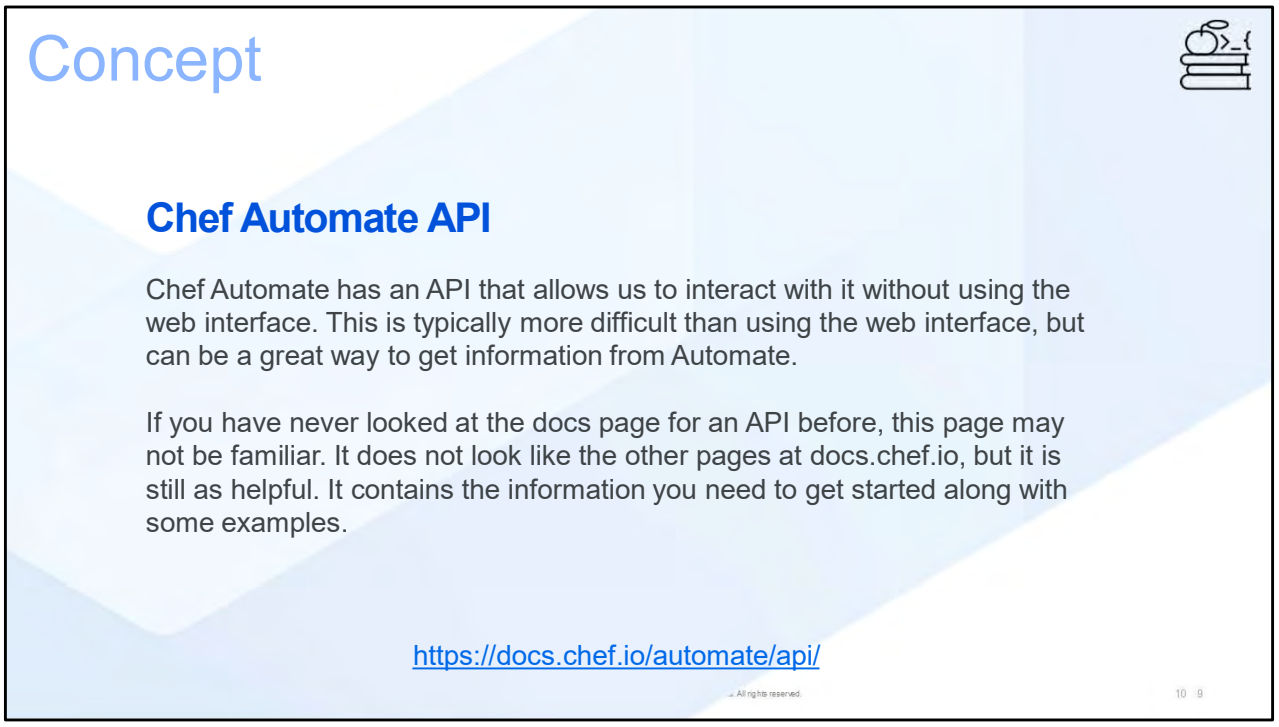
10 7

## Chef Automate HA decision

Implementing Chef Automate HA is not a trivial task. There are many factors that need to be considered

The information given in this module only gives a high-level view of what Automate HA is and how it is setup. If your organization is interested in implementing Automate HA, your Customer Success Manager can provide much more detail. Contact them and schedule a specialized consultation.

Progress Chef




https://docs.chef.io/automate/api/

## Chef Automate API interaction

We send and receive information from the Automate API using the HTTP methods GET, POST, PUT, PATCH, and DEL

The API is not open to anyone on the network. In order to communicate with the API a token is required. This process of creating the API token is covered in the link below. Once you have the token, it is sent with all communication to the API

https://docs.chef.io/automate/api/#section/Authentication

Progress Chef



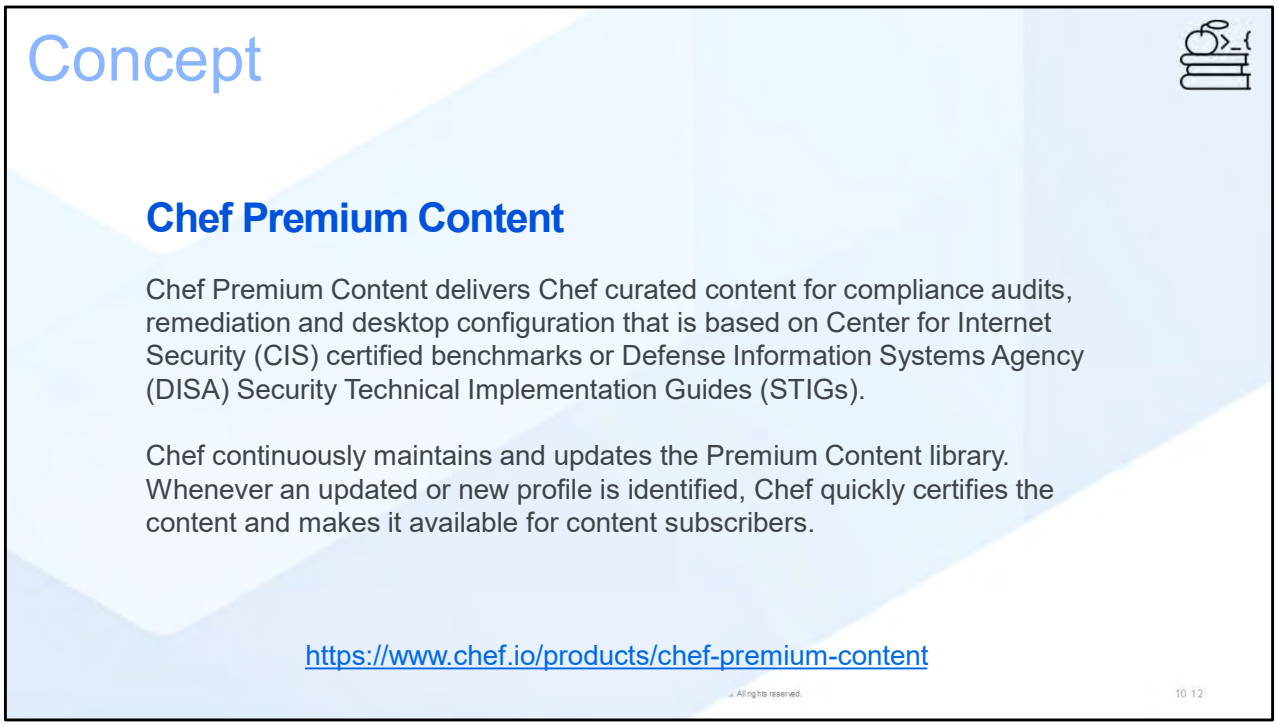


https://docs.chef.io/automate/api/#section/Authentication

## Chef Automate API Usage

The Chef Automate API can be used to create more detailed reports than what is available in the UI. Scripts can be written that pull certain pieces of data and format it how you want. These scripts can then be run on a schedule or ad-hoc to meet your reporting requirements.

Progress Chef




https://www.chef.io/products/chef-premium-content

## Chef Premium Content Categories

Chef Premium Content provides compliance scans for three different categories:

- **Operating Systems** – CIS Amazon Linux 2, DISA STIG Windows Server 2019, and others
- **Applications** – CIS Apache HTTP 2.2, CIS NGINX, and more
- **Cloud Environment** – CIS Amazon Web Services Foundation Benchmark, CIS Kubernetes Benchmark, and more

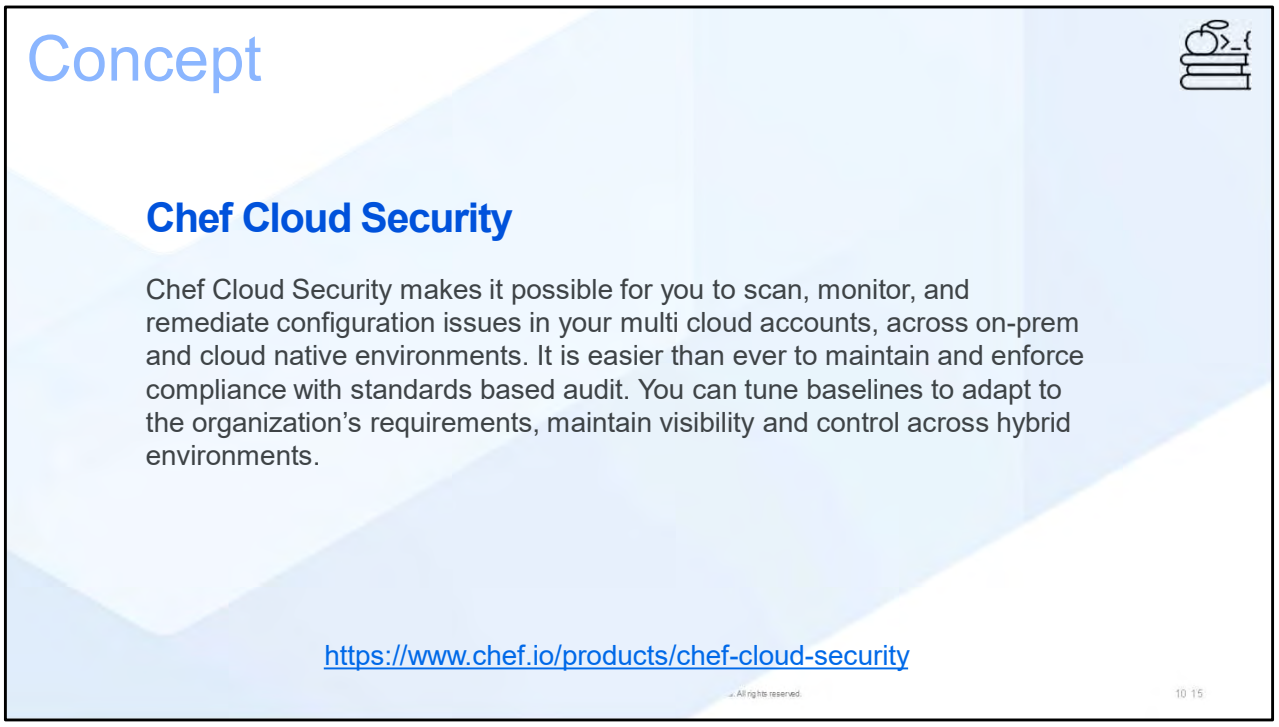
Progress Chef

## Chef Premium Content Contents

Chef Premium Content is not only additional compliance scans. Many of the compliance scans are paired with a cookbook that can remediate any deficiencies that are found by the scan.

This saves on the development time required to pass security audits. As long as your chef-client runs are successful then your nodes will be brought into compliance and stay that way.

Progress Chef




https://www.chef.io/products/chef-cloud-security

## Chef Cloud Security Components

Chef Cloud Security is powered by four Chef components:

- Chef Automate provides comprehensive security and compliance visibility across environments
- Chef Infra automates infrastructure configuration for any data center or cloud environment
- Chef InSpec automates security testing to ensure compliance of servers, containers, or cloud environments
- Chef Premium Content provides CIS based profiles and remediation content for a range of enterprise assets

Progress Chef



10 16

## Chef Cloud Security Use Cases

**Secure Hybrid Cloud Management** – Manage both cloud and on-prem environments (Cloud Instances, VMs, Multi-Tier Apps, Jenkins, Azure DevOps, GitHub) using the same tools and processes

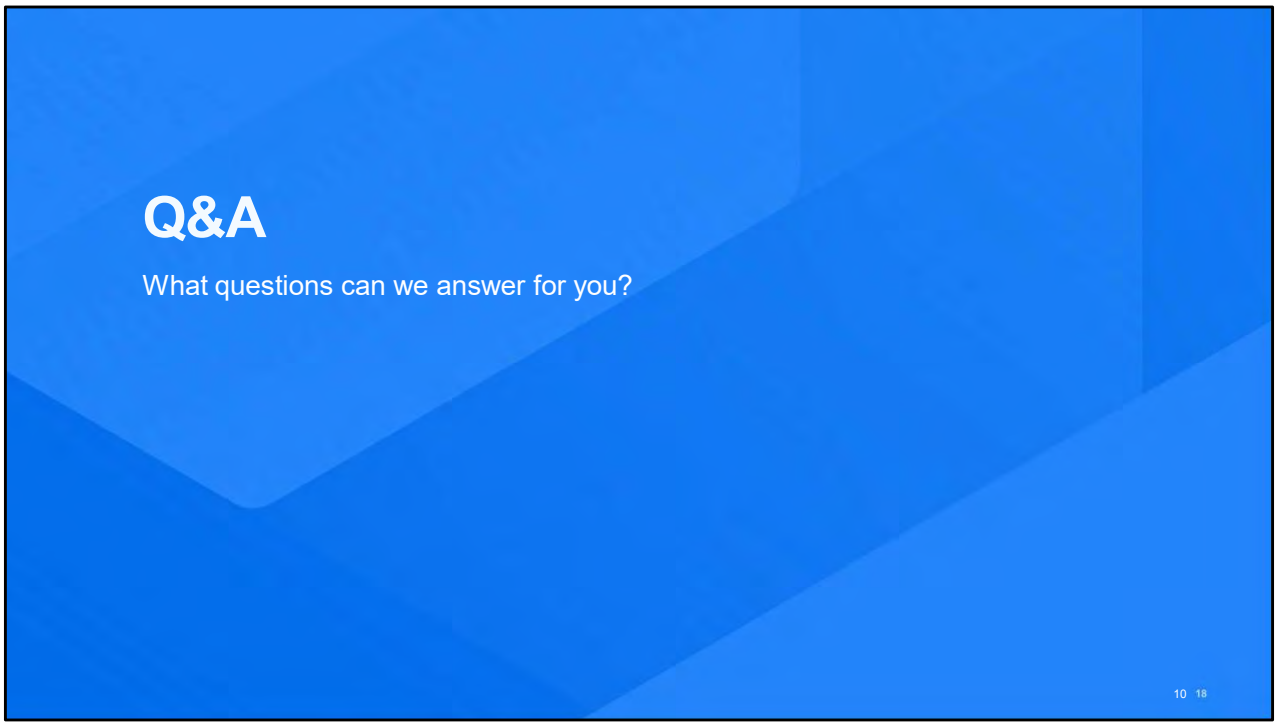
**Multi-Cloud** – Continuously audit cloud accounts and services for security risks and misconfigurations. Achieve consistent security across AWS, Azure, Google, and Alibaba Cloud

**Audit for Compliance** – Automate CIS benchmark tests for Cloud Fundamentals, Kubernetes, and Docker

Progress Chef



10 17


Q&A
What questions can we answer for you?
10 18


Progress Chef

Progress Chef

# Further Resources

Other Places to Talk About, Practice, and Learn Chef Automate Compliance

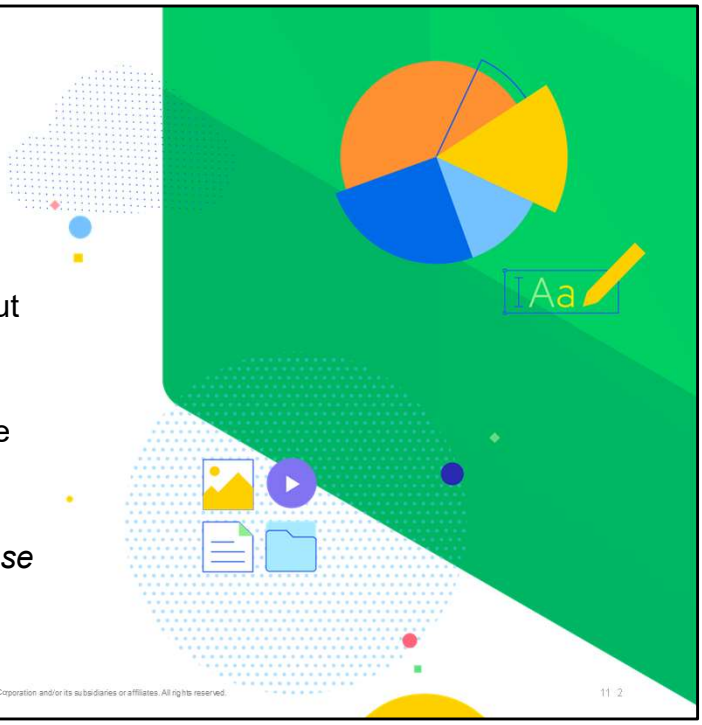
11- 1

## Going Forward

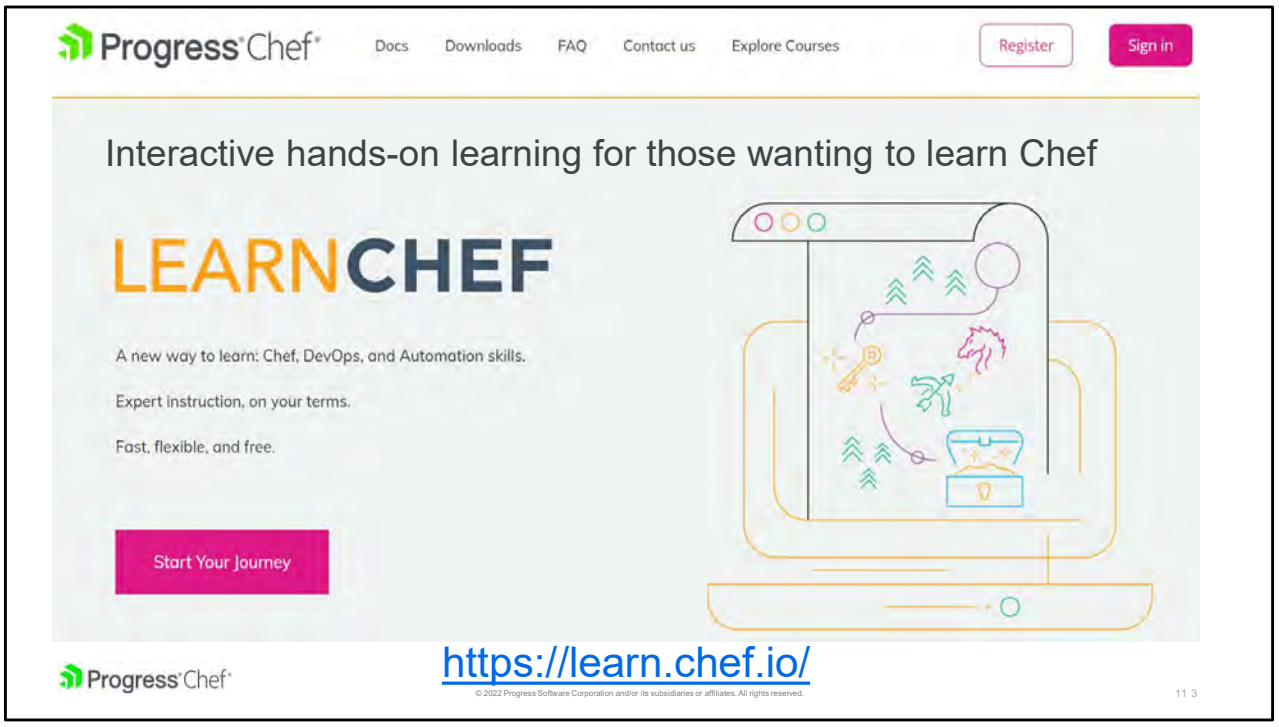
There are many Chef resources available to you outside this class. During this module we will talk about just a few of those resources.

But…remember what we said at the beginning of this class:

*The best way to learn Chef is to use Chef*




Progress Chef




https://learn.chef.io/

## docs.chef.io

Docs are available to you, 24 hours a day, 7 days a week.

Any question you have, you probably will find the answer for on our Docs site.

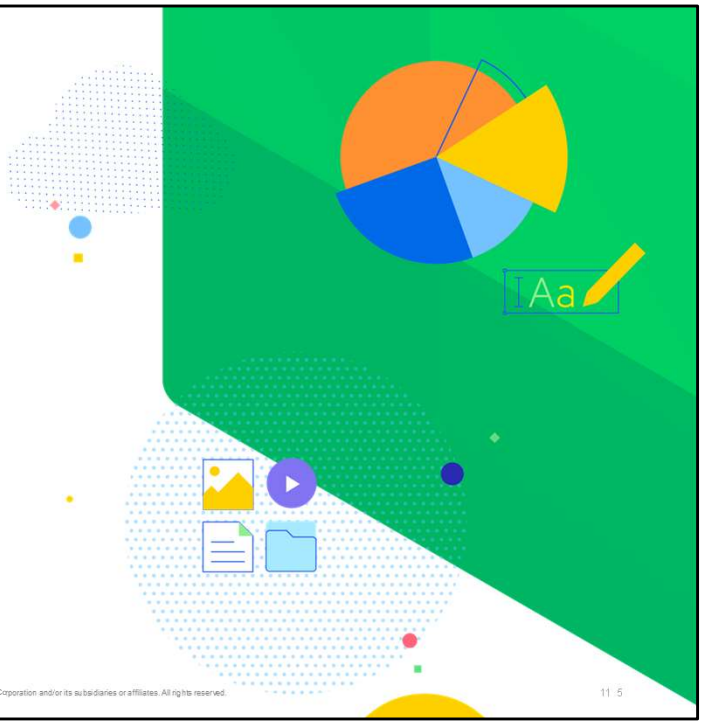
Progress Chef

## docs.chef.io

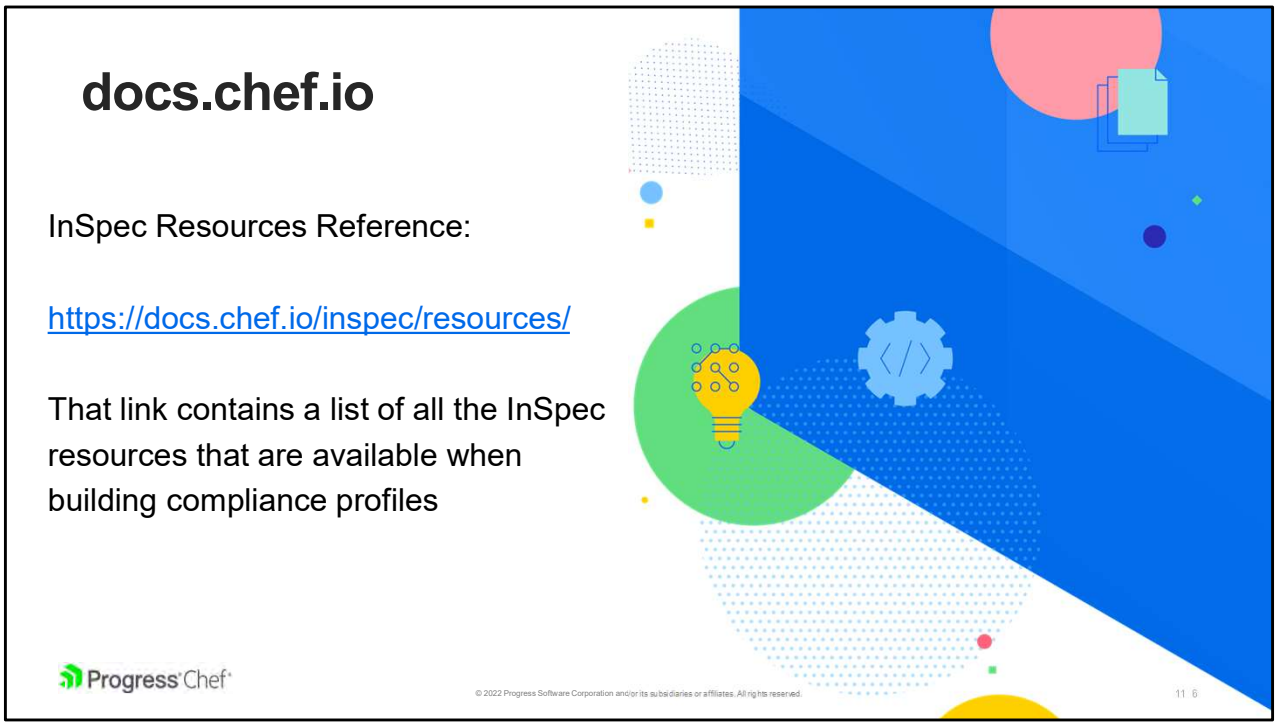
Many different topics are available specific to Chef Automate such as:

- Getting Started
- Install
- High Availability
- Configure
- Manage
- Settings

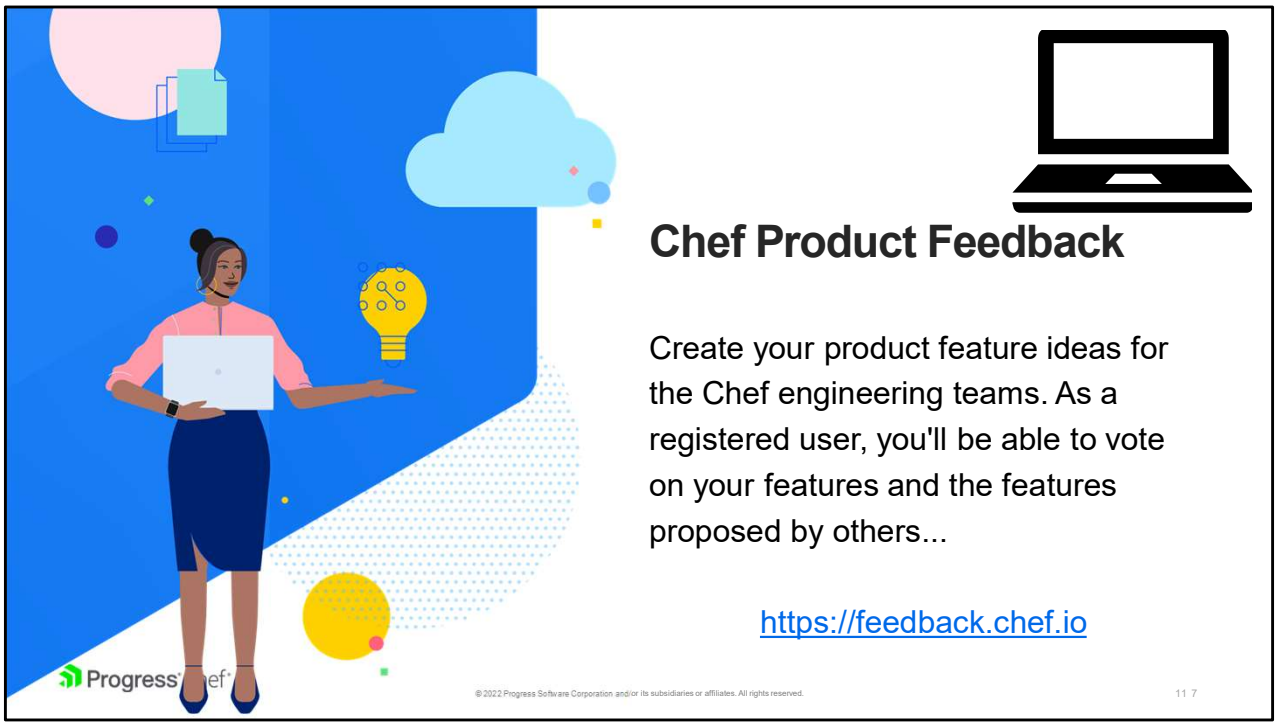
https://docs.chef.io/automate/




Progress Chef

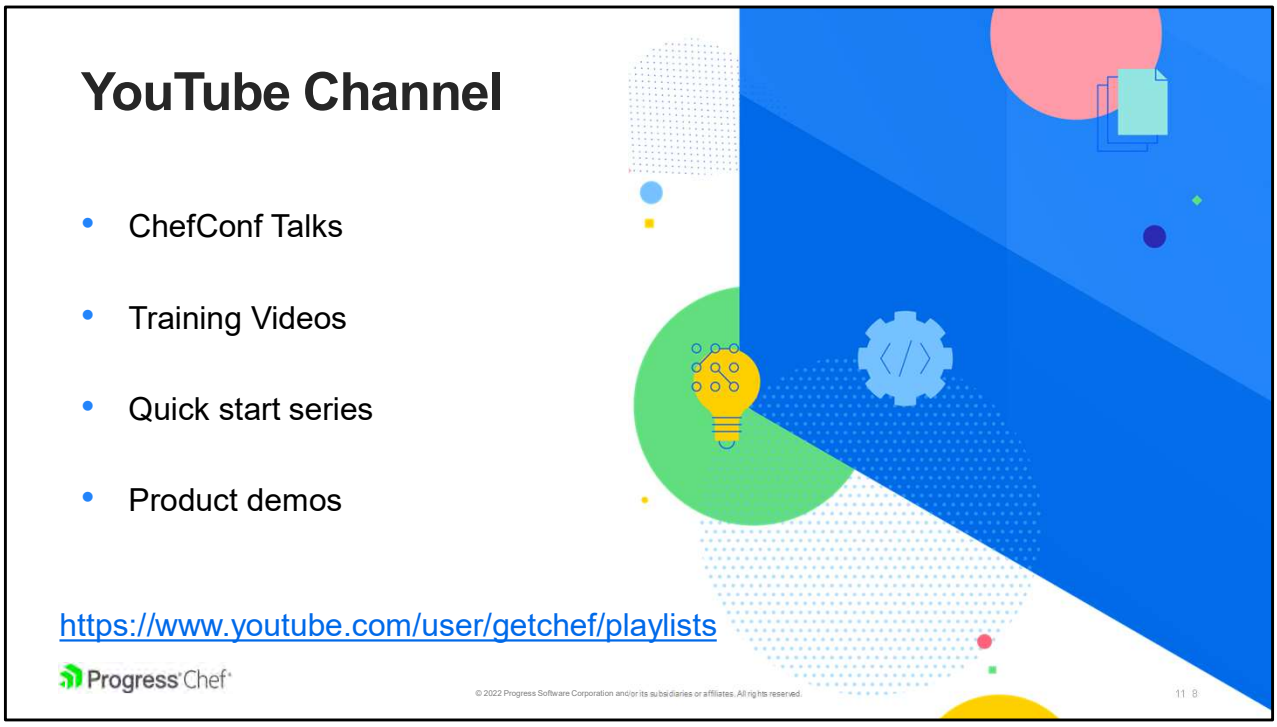



https://docs.chef.io/inspec/resources/




Help us build the best product. If you have an idea we would love to hear more about it. Or come and vote on other features proposed by others.

https://feedback.chef.io




We have uploaded a number of videos to the Chef YouTube channel, including training videos and talks from past Chef conferences.

https://www.youtube.com/user/getchef/playlists


# Thank You!

Progress Chef