Introduction to Continuous Compliance and Remediation

Nathen Harvey
Chef

@nathenharvey

O'REILLY
Velocity
Introductions

- Who are you?
- Why are you here?
- What was the last show you binge watched?
Step one: Detect
Gain visibility into current status to satisfy audits and drive decision-making

55% of organizations do compliance assessments inconsistently or not at all.

Apply policies and gain a complete view across the fleet

- Accurately assess risk
- Prioritize remediation actions
- Maintain audit readiness
- Create and adjust policies

Continuous visibility means that you enter into audits knowing the outcome.

Jon Williams, NIU
Step two: Correct
Remediate issues to improve performance and security

58% of organizations need days or longer to remediate issues.

- Prioritize actions based on impact
- Improve application performance
- Close security holes
- Prove policy compliance

Web & Media Giant
Can patch 250,000 nodes within 6 hours of a patch being made available
A tale of three personas...
... and a single language.
From lemons...

```
$ grep "^key" /etc/tac_plus/tac_plus.conf | sed 's/key = //'
s00persecretkey
$
```
... create lemonade!

control 'sox-404.3.5' do
  title 'Network Device to Central Auth Encryption'
  impact 1.0
  desc "
    All communication between network devices and central auth
    must be encrypted. Our TACACS+ servers encrypt all the time
    and the presence of a pre-shared key proves it."

describe ini('/etc/tac_plus/tac_plus.conf') do
  its('key') { should_not be_nil }
end
end

@nathenharvey
Map Documentation to Controls

404.3.5: Communication between network devices and central authentication systems must be encrypted at all times.

control 'sox-404.3.5' do
  title 'Network Device to Central Auth Encryption'
  impact 1.0
  desc "All communication between network devices and central auth must be encrypted. Our TACACS+ servers encrypt all the time and the presence of a pre-shared key proves it."
  describe ini('/etc/tac_plus/tac_plus.conf') do
    its('key') { should_not be_nil }
  end
end
404.3.5: Communication between network devices and central authentication systems must be encrypted at all times.

All communication between network devices and central auth must be encrypted. Our TACACS+ servers encrypt all the time and the presence of a pre-shared key proves it.

describe ini('/etc/tac_plus/tac_plus.conf') do
  its('key') { should_not be_nil }
end
end
Automate Test Execution

404.3.5: Communication between network devices and central authentication systems must be encrypted at all times.

control 'sox-404.3.5' do
  title 'Network Device to Central Auth Encryption'
  impact 1.0
  desc "All communication between network devices and central auth must be encrypted. Our TACACS+ servers encrypt all the time and the presence of a pre-shared key proves it."

describe ini('/etc/tac_plus/tac_plus.conf') do
  its('key') { should_not be_nil }
end
end
Today's Workshop

- Detect a compliance failure with InSpec and Chef Automate
- Create a Chef cookbook to remediate the failure
- Test the cookbook with Test Kitchen
- Remediate the failure with the new cookbook
- Validate our remediation in Chef Automate
Learning Environment

CHEF
AUTOMATE

@nathenharvey
Learning Environment

CHEF

AUTOMATE

@nathenharvey
Learning Environment

CHEF AUTOMATE

Nodes

Node data
Learning Environment

CHEF
AUTOMATE

Node data

Laptop

Nodes

@nathenharvey
Learning Environment

CHEF

AUTOMATE

Node data

Laptop

Chef Development Workstation

Nodes

@nathenharvey
Learning Environment

CHEF

AUTOMATE

Node data

Laptop

ssh

Chef Development Workstation

Nodes
Learning Environment

**CHEF**

AUTOMATE

- Laptop
- ssh
- Node data
- Nodes
- Chef Development Workstation

@nathenharvey
Access the Learning Environment

- Login to Chef Automate
- Find your workstation/node
- Find your workstation’s IP address
- Login to your workstation
Let's log in to Chef Automate!

- https://velocity-workshop.community.chefdemo.net
- Uses a self-signed certificate in this lab
- Username: admin
- Password: chef-automate
Browse to your node
Browse to your node
Browse to your node

<table>
<thead>
<tr>
<th>Node Name</th>
<th>Check-in</th>
<th>Uptime</th>
<th>Platform</th>
<th>Environment</th>
<th>Policy Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>blue-hearts-02</td>
<td>12 minutes ago</td>
<td>11 minutes</td>
<td>centos</td>
<td>_default</td>
<td>-</td>
</tr>
<tr>
<td>blue-hearts-03</td>
<td>12 minutes ago</td>
<td>11 minutes</td>
<td>centos</td>
<td>_default</td>
<td>-</td>
</tr>
<tr>
<td>blue-hearts-04</td>
<td>12 minutes ago</td>
<td>11 minutes</td>
<td>centos</td>
<td>_default</td>
<td>-</td>
</tr>
<tr>
<td>blue-hearts-05</td>
<td>12 minutes ago</td>
<td>11 minutes</td>
<td>centos</td>
<td>_default</td>
<td>-</td>
</tr>
</tbody>
</table>
View details of your node

Node blue-hearts-03

Use the run history list to examine recent Chef client runs for this node.

This run succeeded on 06/10/2018 at 11:16 PM. All resources ran successfully!

- Run Duration: 11:16 PM - 11:16 PM
- Run Initiator: Not Available
- Run Type: Not Available
- Run ID: 51febe43-1519-4cff-a6ee-1e984c45ce5f
- Uptime: 11 minutes
- Environment: _default
- Platform(s): centos
- IP Address: 172.31.20.230
- FQDN: ip-172-31-20-230.us-west-2.compute.internal

<table>
<thead>
<tr>
<th>Resources</th>
<th>Run List</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Resources: 0</td>
<td>Failed: 0</td>
<td>Successful: 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unchanged: 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unprocessed: 0</td>
</tr>
</tbody>
</table>

@nathenharvey
View details of your node

```json
{
    "audit": {...},
    "block_device": {...},
    "chef_environment": "_default",
    "chef_guid": "5244176a-5482-4c6e-81b3-c85c4e099e5b",
    "chef_packages": {...},
    "cloud": {...},
    "command": {...},
    "counters": {...},
    "cpu": {...},
    "current_user": "root",
    "dmi": {...},
    "docker": {...},
    "domain": "us-west-2.compute.internal",
    "ec2": {...},
    "etc": {...},
}
```
Find the IP of your node

```json
   "network_interfaces_macs" : { ... },
   "placement_availability_zone" : "us-west-2b",
   "product_codes" : "aw0evgkw8e5c1q415zgyspjce",
   "profile" : "default-hvm",
   "public_hostname" : "ec2-54-149-195-207.us-west-2.compute.amazonaws.com",
   "public_ipv4" : "54.149.195.207",
```

@nathenharvey
Log in to your remote workstation

$ ssh chef@12.34.56.78
Using PuTTY on Windows

- Host Name (or IP address): 1234.56.78
- Connection type: SSH
- Open button
Log in to your remote workstation

$ ssh chef@12.34.56.78

The authenticity of host 12.34.56.78 (12.34.56.78)' can't be established.  
ECDSA key fingerprint is SHA256:zAtoeO29XbhRNvwg542cuh4qsKCEaX8hNI1EOCbgd3I.  
Are you sure you want to continue connecting (yes/no)?
Log in to your remote workstation

$ ssh chef@12.34.56.78

The authenticity of host 12.34.56.78 (12.34.56.78)' can't be established.
ECDSA key fingerprint is SHA256:zAtoeO29XbhRNvwg542cuh4qsKCEaX8hNIleOCbgd3I.
Are you sure you want to continue connecting (yes/no)? yes
Using PuTTY on Windows

PuTTY Security Alert

The server’s host key is not cached in the registry. You have no guarantee that the server is the computer you think it is.
The server’s ssh-ed25519 key fingerprint is:
ssh-ed25519 2059 e5bdcce7fa8a53cd81b22a569e878912b1d5d
If you trust this host, hit Yes to add the key to PuTTY’s cache and carry on connecting.
If you want to carry on connecting just once, without adding the key to the cache, hit No.
If you do not trust this host, hit Cancel to abandon the connection.

Yes  No  Cancel  Help
Using PuTTY on Windows

@nathenharvey
Log in to your remote workstation

$ ssh chef@12.34.56.78

The authenticity of host 12.34.56.78 (12.34.56.78)' can't be established.
ECDSA key fingerprint is SHA256:zAtoeO29XbhRNvwg542cuh4qsKCEaX8hN1E0Cbgd3I.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '12.34.56.78' (ECDSA) to the list of known hosts.

chef@12.34.56.78's password:
Log in to your remote workstation

$ ssh chef@12.34.56.78

The authenticity of host 12.34.56.78 (12.34.56.78)' can't be established.
ECDSA key fingerprint is SHA256:zAtoeO29XbhRNvwg542cuh4qsKCEaX8hNI1E0Cbqd3I.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '12.34.56.78' (ECDSA) to the list of known hosts.

chef@12.34.56.78's password: velocity
Using PuTTY on Windows
Create a file with your name

$ touch firstname-lastname
Create a file with your name

$ touch will-robinson
List your home directory

$ ls -t

will-robinson  cookbooks  Berksfile  profiles
nodes          Berksfile.lock  config.json
Verify the installation

$ which inspec

/opt/chefdk/bin/inspec
Verify the installation

$ inspec version

2.1.72

Your version of InSpec is out of date! The latest version is 2.2.10.
Verify the installation

$ which chef

/opt/chefdk/bin/chef
Verify the installation

$ chef --version

Chef Development Kit Version: 3.0.36
chef-client version: 14.1.12
delivery version: master (7206afaf4cf29a17d2144bb39c55b7212cfafcc7)
berks version: 7.0.2
kitchen version: 1.21.2
inspec version: 2.1.72
### Chef DK - The Chef Development Kit

<table>
<thead>
<tr>
<th>Foodcritic</th>
<th>CookStyle</th>
<th>ChefSpec</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Test Your &quot;Chef Style&quot;</strong></td>
<td><strong>Validate your Ruby</strong></td>
<td><strong>Simulate Chef</strong></td>
</tr>
<tr>
<td>● Validate your Chef code against Chef best practices</td>
<td>● Validate your Chef code against Ruby best practices</td>
<td>● Validate your Chef code will run</td>
</tr>
<tr>
<td>● Extend with rules to enforce organizational Chef development best practices</td>
<td>● Identify potential Ruby errors (unclosed strings, etc.)</td>
<td>● Testing for more Chef advanced used cases</td>
</tr>
<tr>
<td>● Enforce compliance &amp; security practices</td>
<td>● Identify style/convention that helps write better code (single quotes vs. double quotes)</td>
<td>● Useful for regression testing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test Kitchen</th>
<th>InSpec</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Let's do this (almost) for real</strong></td>
<td><strong>Verify automation results &amp; ensure compliance</strong></td>
</tr>
<tr>
<td>● Validate your Chef code against Chef best practices</td>
<td>● Assert the intention of your Chef code</td>
</tr>
<tr>
<td>● Extend with rules to enforce organizational Chef development best practices</td>
<td>● Verify on live systems that your Chef code produced the correct result</td>
</tr>
<tr>
<td>● Enforce compliance &amp; security practices</td>
<td>● Confirm your Chef code did not produce compliance drift or failures</td>
</tr>
</tbody>
</table>

@nathenharvey
Running Chef on the Node
Go home

$ cd ~
Run chef

$ run_chef

Starting Chef Client, version 14.1.12
resolving cookbooks for run list: []
Synchronizing Cookbooks:
Installing Cookbook Gems:
Compiling Cookbooks...
[2018-06-11T03:36:50+00:00] WARN: Node blue-hearts-03 has an empty run list.
Converging 0 resources

Running handlers:
Running handlers complete
Chef Client finished, 0/0 resources updated in 01 seconds
Check the converge status in Automate
Check the converge status in Automate
Check the converge status in Automate

@nathenharvey
Compliance data in Automate

Your Chef Automate license will expire in 59 days. Get a license from Chef or apply a license if you already have one.

Overview
0 Nodes
0 Profiles

Node Status
Profile Status

Compliance status and the severity levels for your nodes will appear here.

Learn how to start your first compliance scan.

@nathenharvey
# Compliance data in Automate

<table>
<thead>
<tr>
<th>Overview</th>
<th>0 Nodes</th>
<th>0 Profiles</th>
</tr>
</thead>
</table>

- **Node Status**
- **Profile Status**

Compliance status and the severity levels for your nodes will appear here. Learn how to start your first compliance scan.

---

Global Compliance

@nathenharvey
Run Chef with the audit cookbook

$ run_chef "recipe[audit::default]"

Starting Chef Client, version 14.1.12
...

- Chef::Handler::AuditReport

Running handlers complete

Chef Client finished, 0/2 resources updated in 03 seconds
Check the converge status in Automate

<table>
<thead>
<tr>
<th>Status</th>
<th>Step</th>
<th>Type</th>
<th>Name</th>
<th>Action</th>
<th>Cookbook</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>1/2</td>
<td>inspec_gem</td>
<td>inspec</td>
<td>install</td>
<td>audit</td>
<td>- -</td>
</tr>
<tr>
<td>🔄</td>
<td>2/2</td>
<td>inspec_gem</td>
<td>inspec</td>
<td>nothing</td>
<td>audit</td>
<td>- -</td>
</tr>
</tbody>
</table>
Check the compliance status in Automate

- Your System is Not Compliant

Overview
- 1 Nodes
- 1 Profiles

Node Status

- 1 Total Nodes
  - 1 Failed Nodes
  - 0 Passed Nodes
  - 0 Skipped Nodes

Profile Status

Global Compliance

Severity of Node Failures

@nathennnarvey
Check the compliance status in Automate
Check the compliance status in Automate

<table>
<thead>
<tr>
<th>Overview</th>
<th>1 Nodes</th>
<th>1 Profiles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Status</td>
<td>Profile Status</td>
<td></td>
</tr>
</tbody>
</table>

@nathenharvey
Check the compliance status in Automate

Your System is Not Compliant

Overview

1 Nodes
1 Profiles

Nodes  Platform  Environment  Last Scan  Control Failures

- blue-hearts-03  centos  _default  4 minutes ago  1 FAILED

@nathenharvey
Check the compliance status in Automate

<table>
<thead>
<tr>
<th>Control</th>
<th>Test Results</th>
<th>Severity</th>
<th>Root Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>sshd-1.0: SSH Version 2</td>
<td></td>
<td>CRITICAL (0.7)</td>
<td>ssh</td>
</tr>
</tbody>
</table>
Check the compliance status in Automate

@nathenharvey

Failed test result

SSHd Configuration Protocol should cmp == 2

expected: 2
got:

(compared using `cmp` matcher)
Check the compliance status in Automate

sshd-1.0: SSH Version 2

failed test result

SSHD Configuration Protocol should cmp == 2

expected: 2
got: 
(compared using `cmp` matcher)
Check the compliance status in Automate

<table>
<thead>
<tr>
<th>sshd-1.0: SSH Version 2</th>
<th>1</th>
<th>CRITICAL (0.7)</th>
<th>ssh</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>failed test result</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSHD Configuration Protocol should cmp == 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>expected: 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>got:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(compared using <code>cmp</code> matcher)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Check the compliance status in Automate

sshd-1.0: SSH Version 2
Only SSH version 2 should be enabled

control 'sshd-1.0' do
  impact 0.7
  title 'SSH Version 2'
  desc 'Only SSH version 2 should be enabled'
  describe sshd_config do
    its('Protocol') { should cmp 2 }
  end
end
Review the Setup

tying it all together…

a.k.a. "How the heck did that happen?"
Go home again

```bash
$ cd ~
```
List contents

$ ls

will-robinson  cookbooks  Berksfile  profiles
nodes          Berksfile.lock  config.json
List cookbooks

$  ls  cookbooks

audit
Audit Cookbook

- Installs InSpec (if necessary - included in Chef 13 by default)
- Run InSpec profiles
- Report results to Chef Automate
Attributes for the Audit cookbook

$ cat config.json

```json
{
    "audit": {
        "collector": "chef-automate",
        "profiles": [
            {
                "name": "ssh",
                "path": "/home/chef/profiles/ssh"
            }
        ]
    }
}
```
Our ssh InSpec profile

$ tree profiles/ssh

profiles/ssh/
├── controls
│   └── ssh.rb
└── inspec.yml

1 directory, 2 files
Our ssh InSpec profile

$ cat profiles/ssh/controls/ssh.rb

control 'sshd-1.0' do
  impact 0.7
  title 'SSH Version 2'
  desc 'Only SSH version 2 should be enabled'
  describe sshd_config do
    its('Protocol') { should cmp 2 }
  end
end
Run locally with InSpec

```bash
$ inspec exec profiles/ssh
```

Profile: SSH Configuration (ssh)
Version: 0.1.0
Target: local://

- sshd-1.0: SSH Version 2
  - SSHD Configuration Protocol should cmp == 2

  expected: 2
  got:

  (compared using `cmp` matcher)

Profile Summary: 0 successful controls, 1 control failure, 0 controls skipped
Test Summary: 0 successful, 1 failure, 0 skipped
Next Steps

- Automate the remediation of the failing control
- Test the remediation before deploying
- Deploy the remediation, and use the audit cookbook to report back to Automate
- View the compliant node in Automate
Create an SSH Chef Cookbook

- A recipe to deploy a proper sshd_config configuration file
- A local test environment configured to test our changes
Move to the cookbooks directory

$ cd ~/cookbooks
Generate a new ssh cookbook

$ chef generate cookbook ssh

Generating cookbook ssh
- Ensuring correct cookbook file content
- Committing cookbook files to git
- Ensuring delivery configuration
- Ensuring correct delivery build cookbook content
- Adding delivery configuration to feature branch
- Adding build cookbook to feature branch
- Merging delivery content feature branch to master

Your cookbook is ready. Type `cd ssh` to enter it.

There are several commands you can run to get started locally developing and testing your cookbook. Type `delivery local --help` to see a full list.

Why not start by writing a test? Tests for the default recipe are stored at:

test/integration/default/default_test.rb

If you'd prefer to dive right in, the default recipe can be found at:

recipes/default.rb
Add a server recipe to the ssh cookbook

$ chef generate recipe ssh server

Recipe: code_generator::recipe
  * directory[./ssh/spec/unit/recipes] action create (up to date)
  * cookbook_file[./ssh/spec/spec_helper.rb] action create_if_missing (up to date)
  * template[./ssh/spec/unit/recipes/server_spec.rb] action create_if_missing
    - create new file ./ssh/spec/unit/recipes/server_spec.rb
    - update content in file ./ssh/spec/unit/recipes/server_spec.rb from none to 7c8724
      (diff output suppressed by config)
  * directory[./ssh/test/integration/default] action create (up to date)
  * template[./ssh/test/integration/default/server_test.rb] action create_if_missing
    - create new file ./ssh/test/integration/default/server_test.rb
    - update content in file ./ssh/test/integration/default/server_test.rb from none to f2f1c1
      (diff output suppressed by config)
  * template[./ssh/recipes/server.rb] action create
    - create new file ./ssh/recipes/server.rb
    - update content in file ./ssh/recipes/server.rb from none to f29497
      (diff output suppressed by config)
Add a template to the cookbook

$ chef generate template ssh sshd_config -s /etc/ssh/sshd_config

Recipe: code_generator::template
* directory[./ssh/templates/default] action create
  - create new directory ./ssh/templates/default
* file[./ssh/templates/sshd_config.erb] action create
  - create new file ./ssh/templates/sshd_config.erb
  - update content in file ./ssh/templates/sshd_config.erb from none to a16b11
    (diff output suppressed by config)
Server Recipe

~/.cookbooks/ssh/recipes/server.rb

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

Never used a command-line text editor before? Type: nano cookbooks/ssh/recipes/server.rb
Using nano

Press Control-x
Using nano

Press Y

@nathenharvey
Using nano

Press Enter to confirm filename
Remember...

Infrastructure policies need testing!
- Linting
- Static analysis
- Unit testing
- Integration Testing
- Compliance Testing

"Infrastructure as Code" should be tested like ANY other codebase.

@nathenharvey
Test-driven Development

- Write a test, watch it fail
- Write some code
- Write and run more tests
- Code review
- Delivery pipeline to production
- Lowered chance of production failure
Testing the change

- kitchen create
- kitchen converge
- kitchen verify
- kitchen destroy

Create Virtual Machine
Install Chef Tools
Copy Cookbooks
Run/Apply Cookbooks
Verify Assumptions
Destroy Virtual Machine

driver
provisioner
busser

@nathenharvey
Test Kitchen Configuration (1 of 3)

~/.kitchen.yml

```yaml
---
driver:
- name: vagrant
+ name: docker
...
```
~/cookbooks/ssh/.kitchen.yml

...

platforms:

- name: ubuntu-16.04

- name: centos-7.2

+ name: centos-7.3

...

@nathenharvey
Test Kitchen Configuration (3 of 3)

~/.kitchen.yml

suites:
- name: default
+ name: server

run_list:
- recipe[ssh::default]
+ recipe[ssh::server]

verifier:
  inspec_tests:
- test/smoke/default
+ /home/chef/profiles/ssh

attributes:
Move to the ssh cookbook directory

$ cd ~/cookbooks/ssh
List the kitchens

$ kitchen list

<table>
<thead>
<tr>
<th>Instance</th>
<th>Driver</th>
<th>Provisioner</th>
<th>Verifier</th>
<th>Transport</th>
<th>Last Action</th>
<th>Last Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>server-centos-73</td>
<td>Docker</td>
<td>ChefZero</td>
<td>Inspec</td>
<td>Ssh</td>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
</tr>
</tbody>
</table>
$ kitchen converge

-----> Starting Kitchen (v1.21.2)
-----> Creating <server-centos-73>...
    setsebool: SELinux is disabled.
    Sending build context to Docker daemon 209.9 kB
    Step 1/16 : FROM centos:centos7
...
    Running handlers:
    Running handlers complete
    Chef Client finished, 1/1 resources updated in 01 seconds
    Downloading files from <server-centos-73>
    Finished converging <server-centos-73> (0m21.21s).
-----> Kitchen is finished. (1m3.05s)
Test-Driven Development

1. Add a test
2. Run the tests
   - If pass, stop
   - If fail, make a little change
3. Run the tests
   - If pass, development continues
   - If fail, make a little change again
$ kitchen verify

-----> Verifying <server-centos-73>...
   Loaded ssh
[2018-06-11T04:06:59+00:00] ERROR: Cannot find a UUID for your node.

Profile: SSH Configuration (ssh)
Version: 0.1.0
Target: ssh://kitchen@localhost:32768

× ssdh-1.0: SSH Version 2
   × SSHD Configuration Protocol should cmp == 2
      expected: 2
      got:
      (compared using `cmp` matcher)

Profile Summary: 0 successful controls, 1 control failure, 0 controls skipped
Test Summary: 0 successful, 1 failure, 0 skipped
Test-Driven Development

1. Add a test
2. Run the tests
   - If pass, done
   - If fail, make a little change
Edit the SSH Configuration Template

```
~/cookbooks/ssh/templates/sshd_config.erb

#ListenAddress 0.0.0.0
#ListenAddress ::

# The default requires explicit activation of protocol 1
-
#Protocol 2
+
Protocol 2

# HostKey for protocol version 1
```
Test-Driven Development

1. Add a test
2. Run the tests
   - If pass, go to the next step.
   - If fail, go to the next step.
3. Make a little change
4. Run the tests
   - If pass, development continues.
   - If fail, development stops.
5. Development stops.
Converge (apply our new cookbook change)

$ kitchen converge

-----> Starting Kitchen (v1.21.2)
-----> Converging <server-centos-73>...
...
   # The default requires explicit activation of protocol 1
   -#Protocol 2
   +Protocol 2
...
   Running handlers:
   Running handlers complete
   Chef Client finished, 1/1 resources updated in 00 seconds
   Downloading files from <server-centos-73>
   Finished converging <server-centos-73> (0m4.74s).
-----> Kitchen is finished. (0m6.71s)
Verify the Kitchen

$ kitchen verify

-----> Starting Kitchen (v1.21.2)
-----> Setting up <server-centos-73>...
    Finished setting up <server-centos-73> (0m0.00s).
-----> Verifying <server-centos-73>...
    Loaded ssh
[2018-06-11T04:09:36+00:00] ERROR: Cannot find a UUID for your node.

Profile: SSH Configuration (ssh)
Version: 0.1.0
Target: ssh://kitchen@localhost:32768

✔ ssdh-1.0: SSH Version 2
  ✔ SSHD Configuration Protocol should cmp == 2

Profile Summary: 1 successful control, 0 control failures, 0 controls skipped
Test Summary: 1 successful, 0 failures, 0 skipped
  Finished verifying <server-centos-73> (0m1.20s).
-----> Kitchen is finished. (0m3.17s)
Test-Driven Development

1. Add a test
2. Run the tests
   - If pass, development continues
   - If fail, make a little change
3. Run the tests
   - If pass, development stops
   - If fail, make a little change
$ kitchen test

-----> Starting Kitchen (v1.21.2)
-----> Cleaning up any prior instances of <server-centos-73>
-----> Destroying <server-centos-73>...
...
-----> Testing <server-centos-73>
-----> Creating <server-centos-73>...
...
-----> Converging <server-centos-73>...
...
-----> Installing Chef Omnibus (install only if missing)
...
End-to-End Kitchen Test (2 of 2)

$ kitchen test

------> Setting up <server-centos-73>...
...
------> Verifying <server-centos-73>...
...
Target: ssh://kitchen@localhost:32769

✔ ssdh-1.0: SSH Version 2
✔ SSHD Configuration Protocol should cmp == 2

Profile Summary: 1 successful control, 0 control failures, 0 controls skipped
Test Summary: 1 successful, 0 failures, 0 skipped
...
------> Destroying <server-centos-73>...
...
------> Kitchen is finished. (0m23.89s)
What's next?

- Test-driven development cycle is complete
- Deploy the change (with confidence!)
Remediate with Chef

```
$ run_chef "recipe[ssh::server],recipe[audit::default]"
```

Starting Chef Client, version 14.1.12
...
Synchronizing Cookbooks:
  - audit (7.0.0)
  - ssh (0.1.0)
...
  -#Protocol 2
  +Protocol 2
...
Chef Client finished, 1/3 resources updated in 04 seconds
## Verify Converge Status in Automate

<table>
<thead>
<tr>
<th>Resources</th>
<th>Run List</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Resources</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Failed</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Successful</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Unchanged</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Unprocessed</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status</th>
<th>Step</th>
<th>Type</th>
<th>Name</th>
<th>Action</th>
<th>Cookbook</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>1/3</td>
<td>inspec_gem</td>
<td>inspec</td>
<td>install</td>
<td>audit</td>
<td>-</td>
</tr>
<tr>
<td>✔️</td>
<td>2/3</td>
<td>template</td>
<td>/etc/ssh/sshd_config</td>
<td>create</td>
<td>ssh</td>
<td>-</td>
</tr>
<tr>
<td>❓</td>
<td>3/3</td>
<td>inspec_gem</td>
<td>inspec</td>
<td>nothing</td>
<td>audit</td>
<td>-</td>
</tr>
</tbody>
</table>
Verify Compliance Status in Automate

Your System is Compliant

Overview

1 Nodes
1 Profiles

Node Status

Total Nodes: 1
- Failed Nodes: 0
- Passed Nodes: 1
- Skipped Nodes: 0

Global Compliance

Severity of Node Failures

CRITICAL
MAJOR
MINOR
# Verify Compliance Status in Automate

<table>
<thead>
<tr>
<th>Control</th>
<th>Test Results</th>
<th>Severity</th>
<th>Root Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>sshd-1.0: SSH Version 2</td>
<td>1</td>
<td>CRITICAL (0.7)</td>
<td>ssh</td>
</tr>
</tbody>
</table>

@nathenharvey
Verify Compliance Status in Automate

<table>
<thead>
<tr>
<th>Control</th>
<th>Test Results</th>
<th>Severity</th>
<th>Root Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>sshd-1.0: SSH Version 2</td>
<td>1</td>
<td>CRITICAL (0.7)</td>
<td>ssh</td>
</tr>
</tbody>
</table>

Passed test result: SSHD Configuration Protocol should cmp == 2
Ready for more?

• Learn Chef Rally
  learn.chef.io

• Classroom-style Training

@nathenharvey
Get started with **CHEF AUTOMATE**

- [https://learn.chef.io/modules/chef-automate-pilot/](https://learn.chef.io/modules/chef-automate-pilot/)
  *Set up your own demo environment*

- [https://automate.chef.io/](https://automate.chef.io/)
  *Install on-prem, generate a trial license*

- AWS OpsWorks for Chef Automate
  *Managed service*

- AWS and Azure Marketplace
Join us on Slack!

- [http://community-slack.chef.io](http://community-slack.chef.io)
- [#general (for Chef stuff)](#general (for Chef stuff))
- [#inspec](#inspec)