INSPEC

BY CHEF™

Compliance Automation with InSpec

Nathen Harvey - @nathenharvey

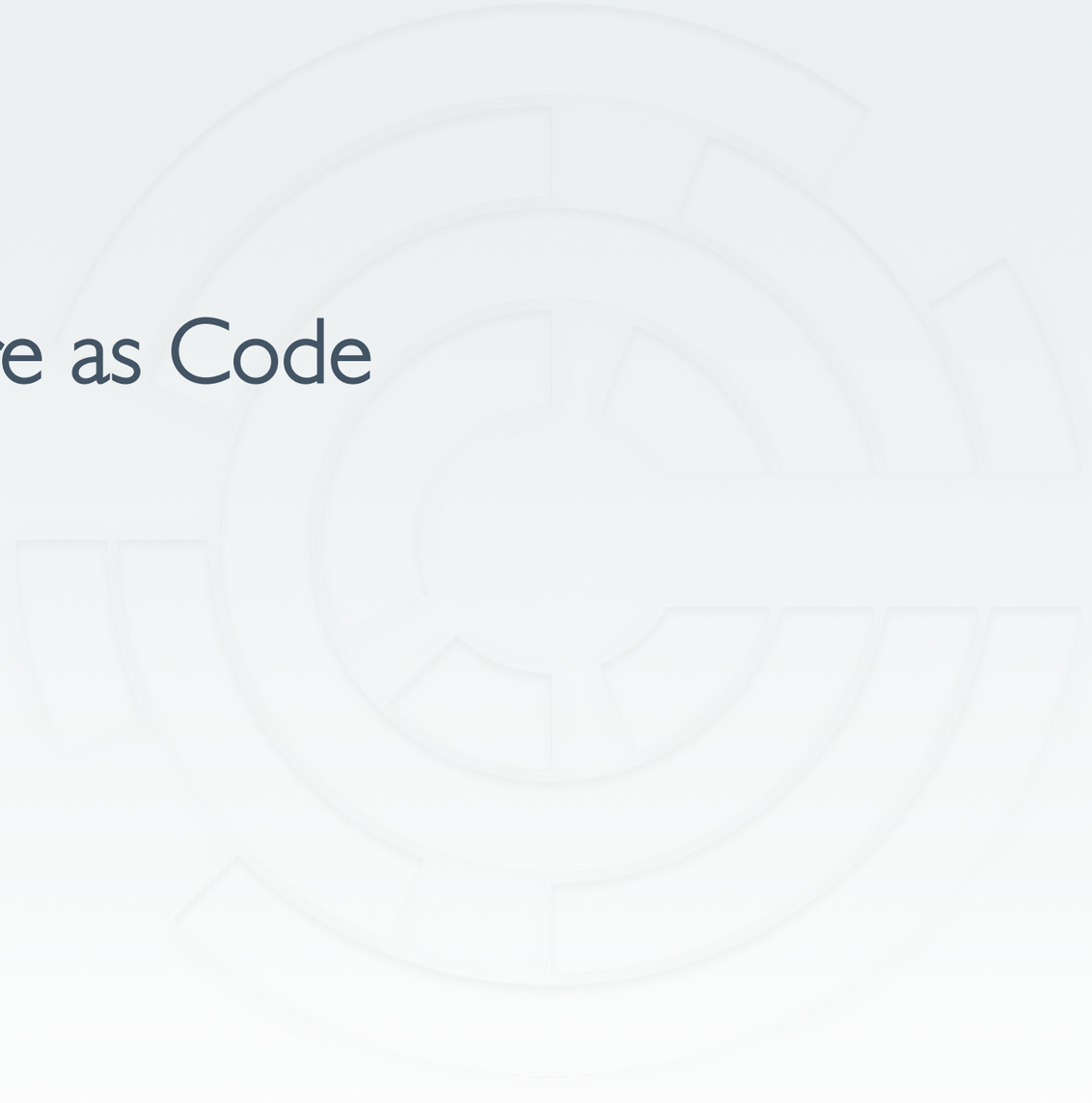
CHEF™

Agenda

- Infrastructure as Code
- Compliance as Code
- Lab access
- InSpec Command Line Interface
- Integration Testing
- Continuous Compliance
- Further Resources

http://community-slack.chef.io

Infrastructure as Code

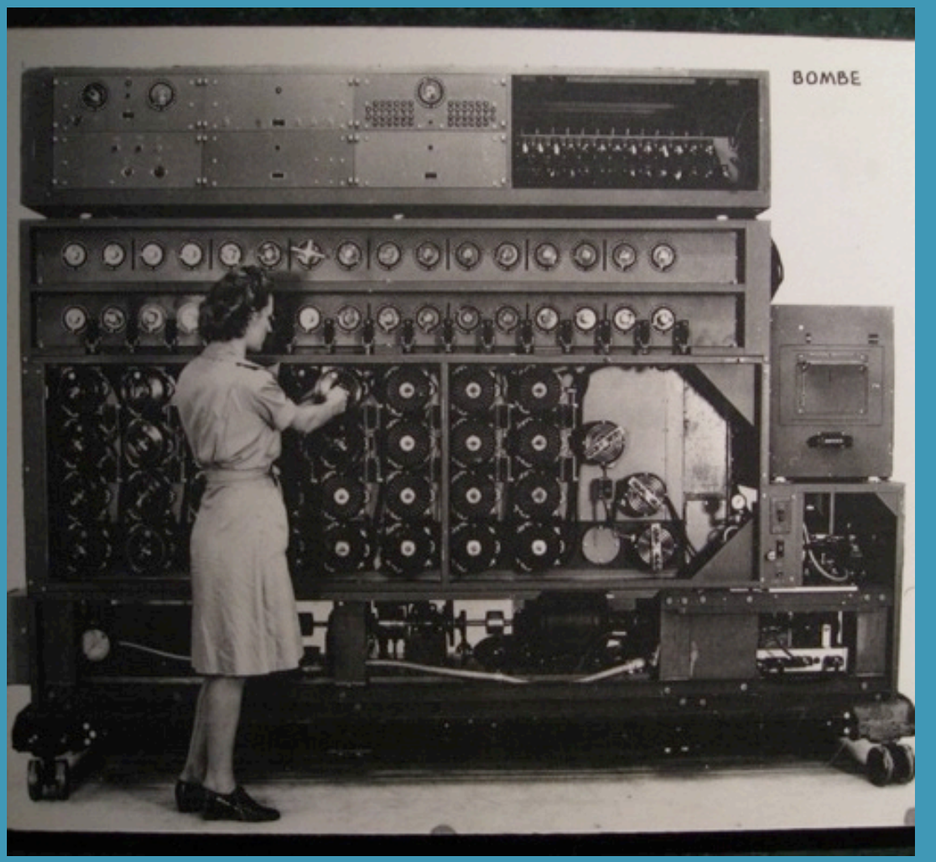


INSPEC

CHEF

Infrastructure as Code

- Programmatically provision and configure components
- Treat like any other code base
- Reconstruct business from code repository, data backup, and compute resources



INSPEC

Infrastructure as Code

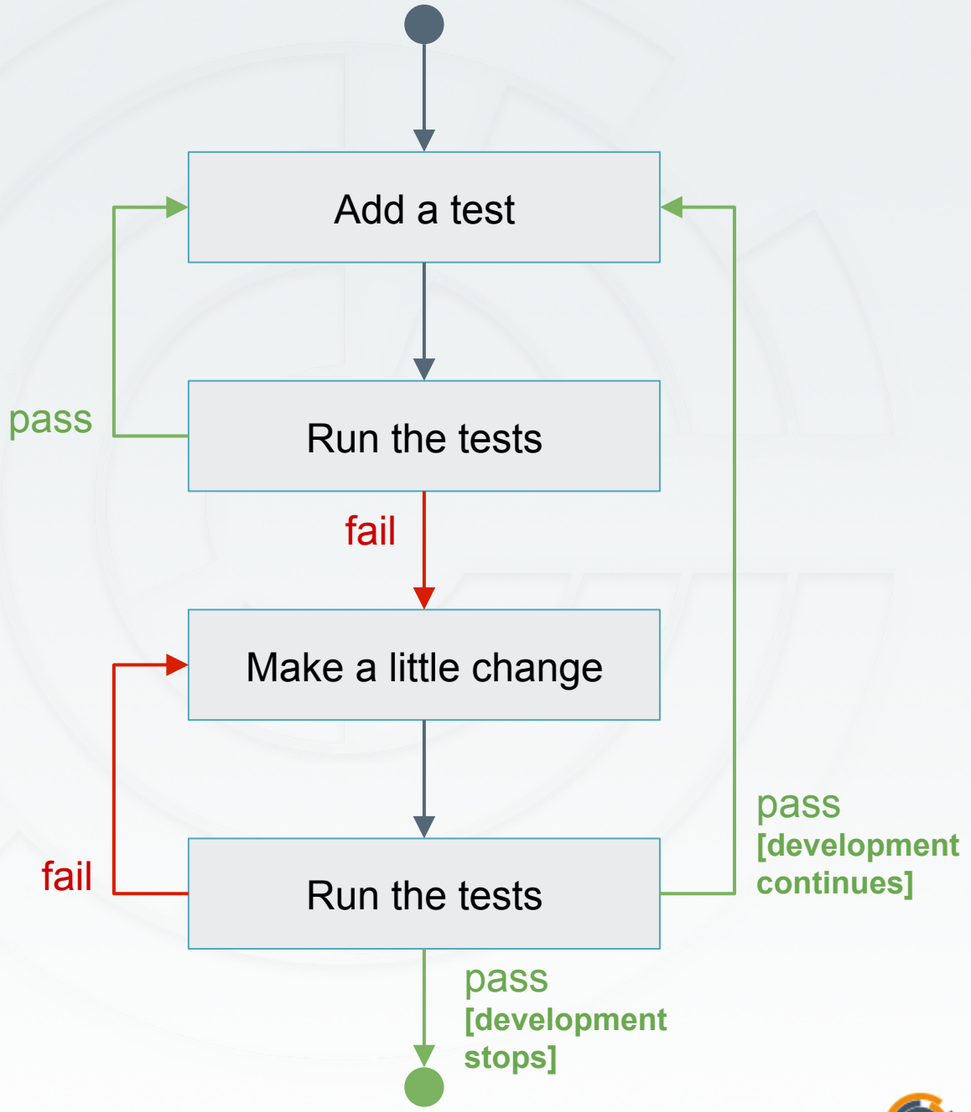
BUILD	DEPLOY	MANAGE
• Develop reusable Cookbooks • Expose tunable settings • Test locally to reduce risk, and ensure compliance	• Commit to Source Code • Automated Testing through Continuous Integration • Automatically promote across environments	• Easily deploy new configurations in a matter of minutes. • Continuously verify and repair misconfigured systems

INSPEC

CHEF

Test-driven Development

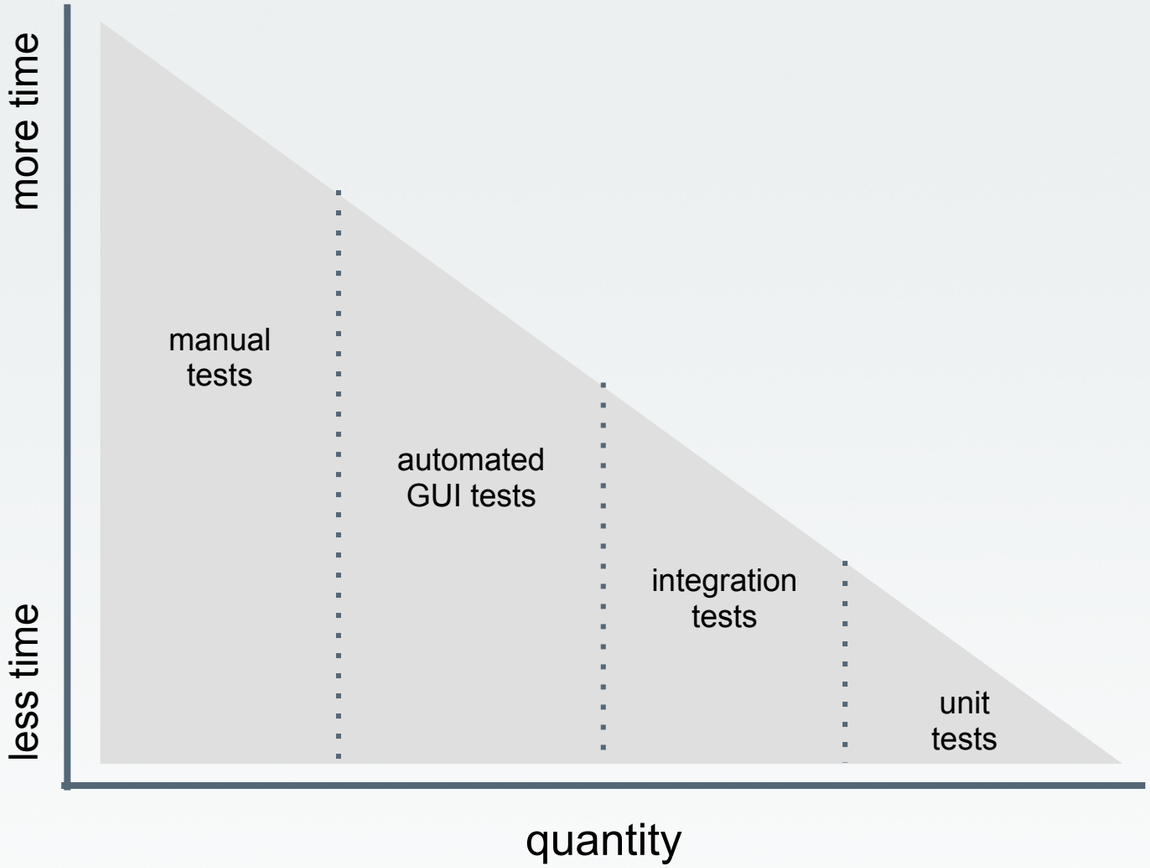
- Write a unit test, watch it fail
- Write some code
- Write and run more unit tests
- Run some integration/acceptance tests
- Code review
- Delivery pipeline to production
- Lowered chance of production failure

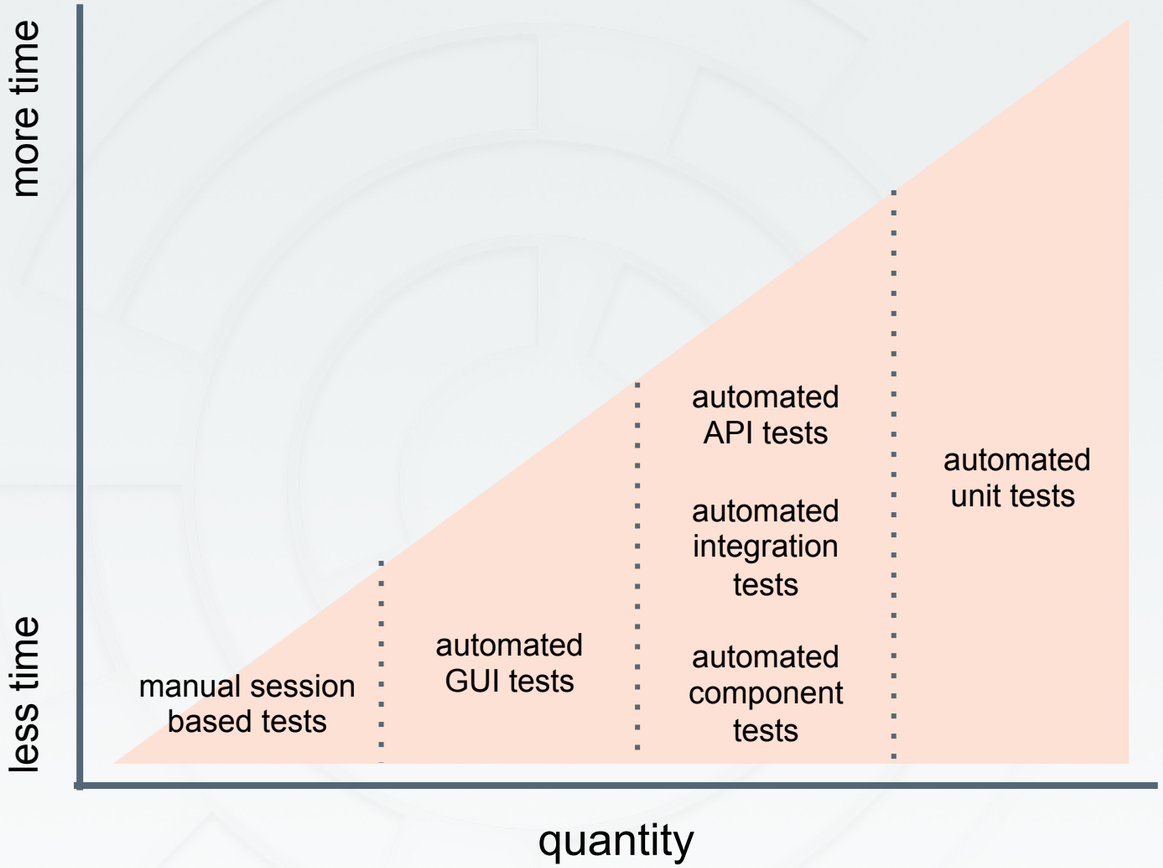


INSPEC

CHEF

Software Testing and Why it Matters





Testing **builds safety** through feedback loops

Inexpensive experiments to provide validation

Reduces risk

Optimize Testing: Do more of the inexpensive testing first!

Remember

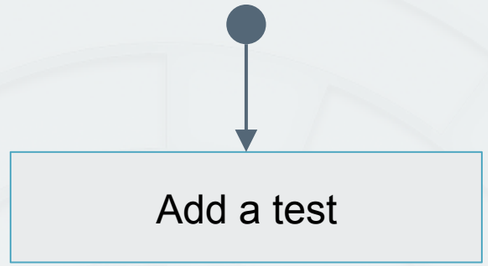
- Infrastructure policies need testing
  - Static Analysis
  - Unit Testing
  - Integration Testing
  - Compliance Testing

Infrastructre as Code should be treated like ANY other codebase.

INSPEC

Integration Testing – Add tests

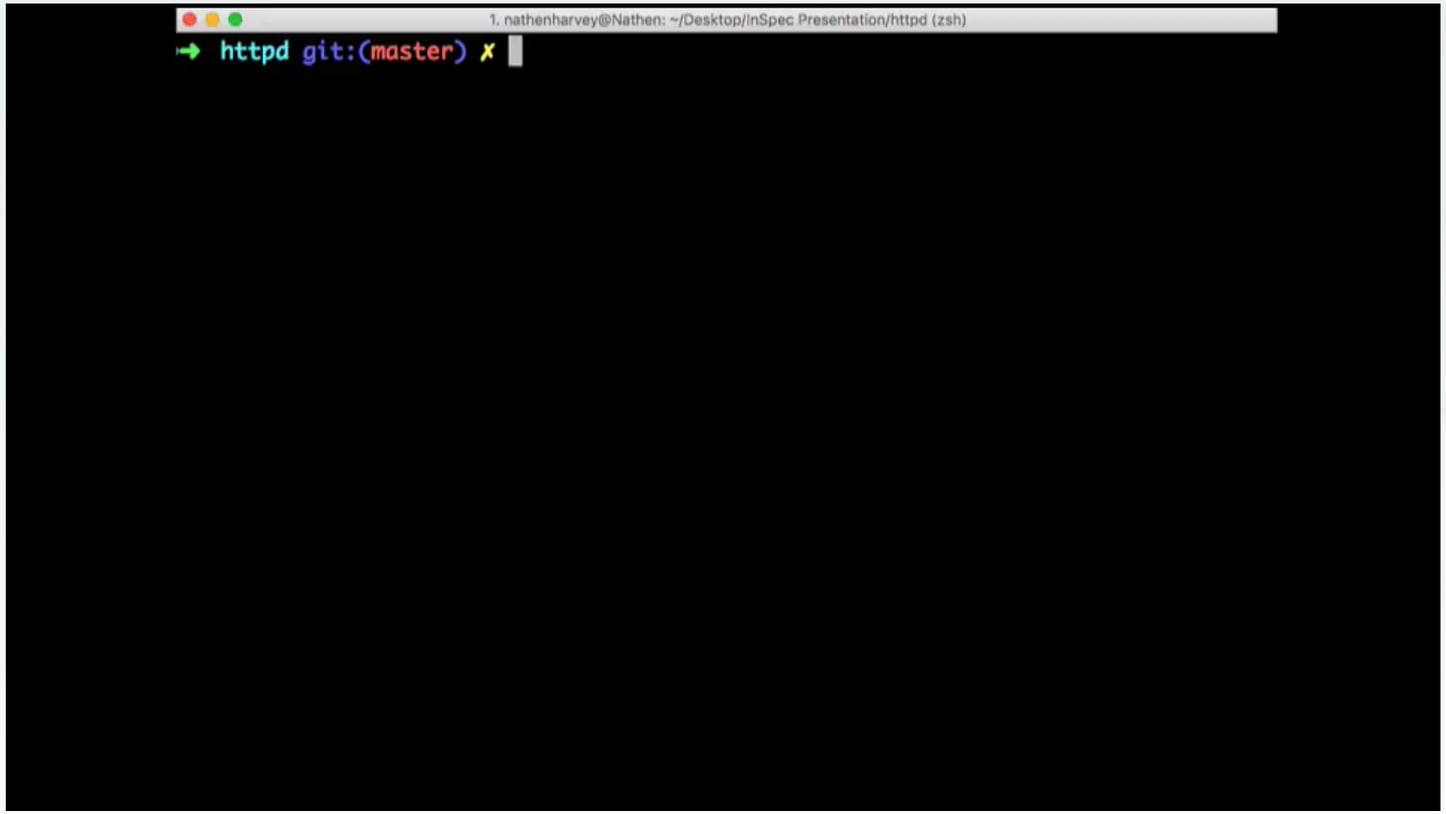
```
describe package 'httpd' do
  it { should be_installed }
end

describe service 'httpd' do
  it { should be_running }
  it { should be_enabled }
end

describe port(80) do
  it { should be_listening }
end
```

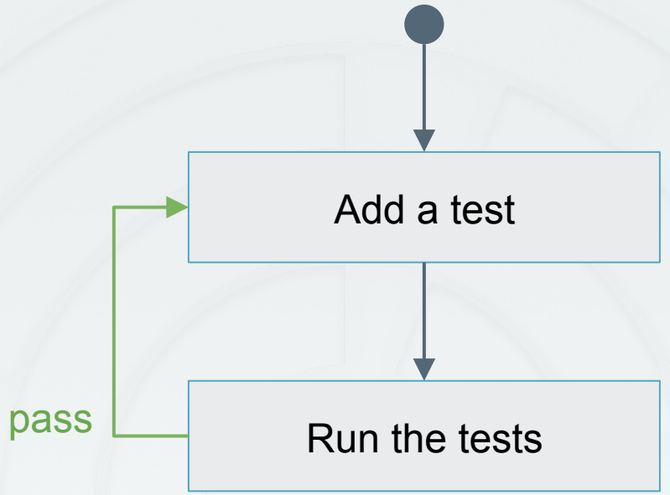


INSPEC

CHEF

Integration Testing – Run the tests



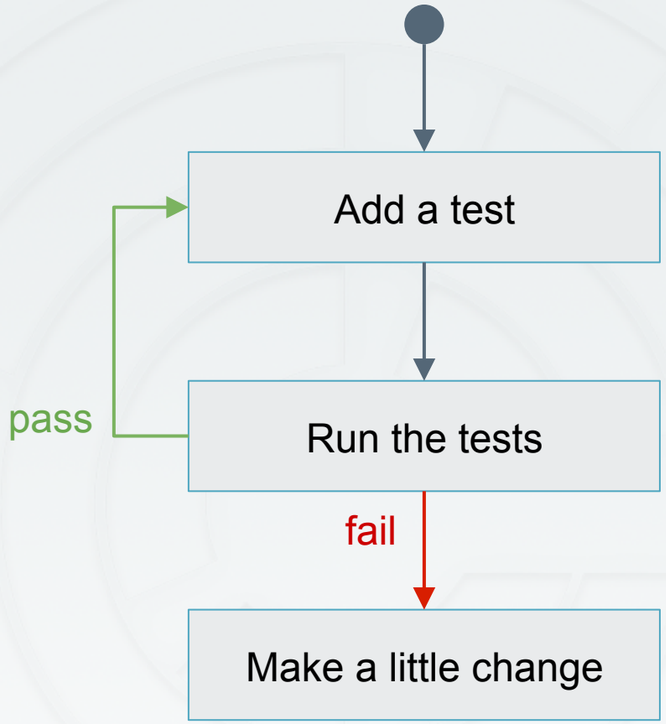


INSPEC

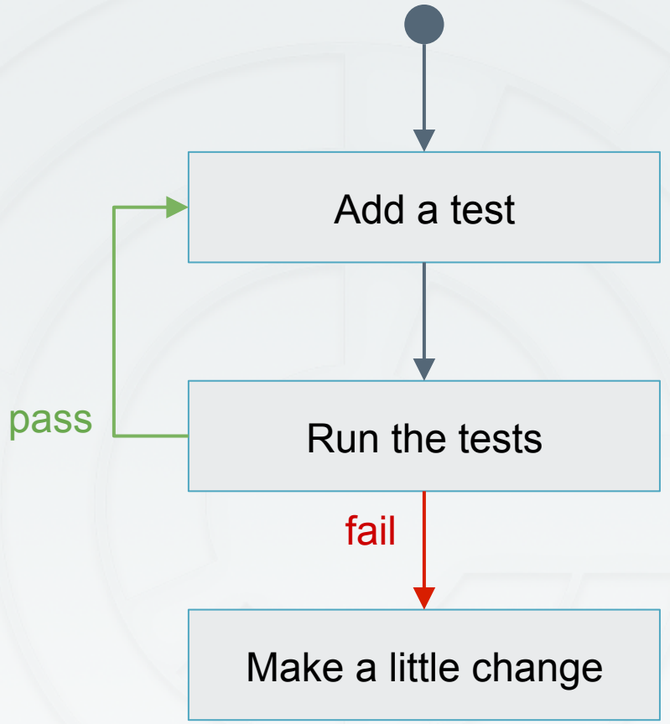
CHEF

Integration Testing – Make a change

```
package 'httpd' do
  action :install
end

service 'httpd' do
  action [ :start, :enable ]
end
```



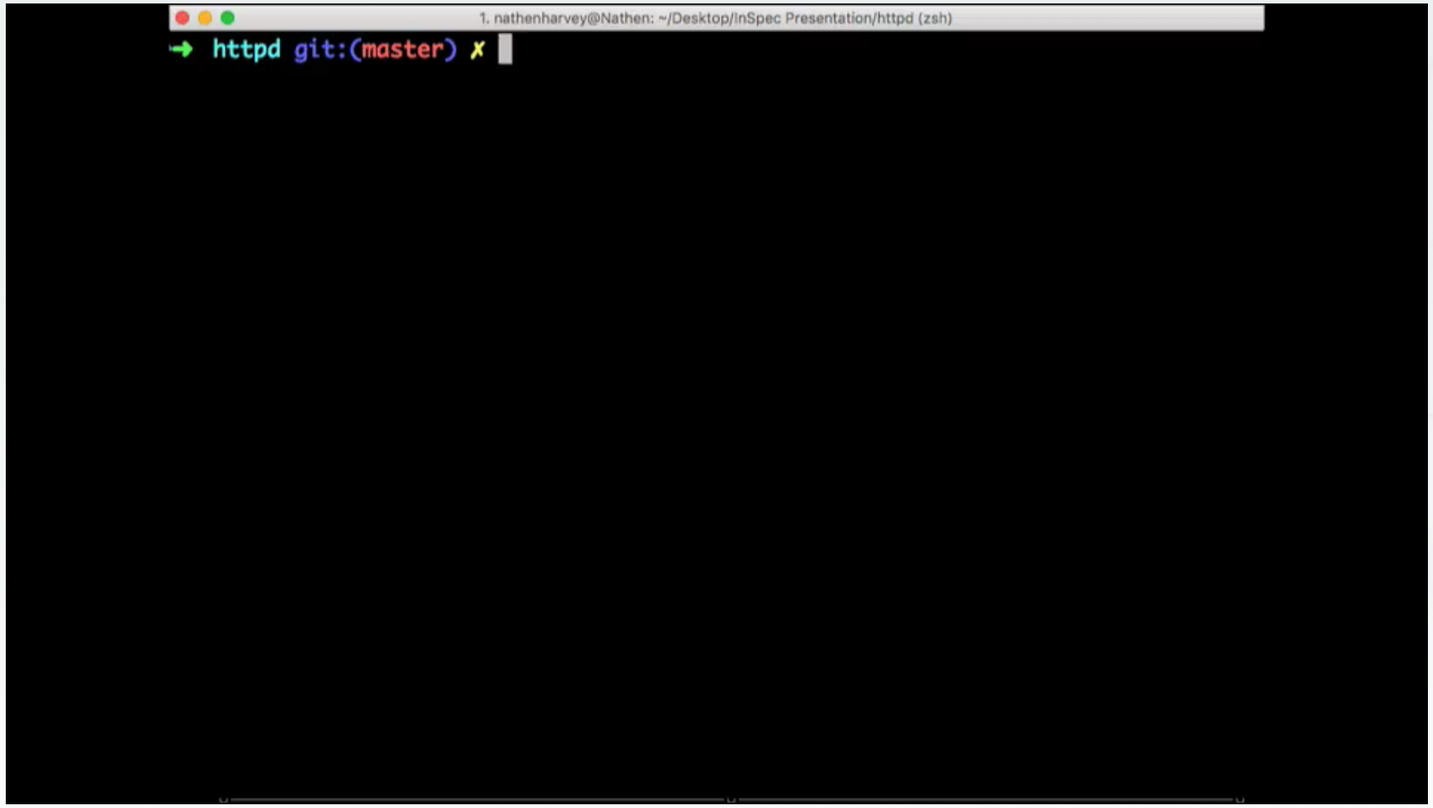
Integration Testing – Apply the change

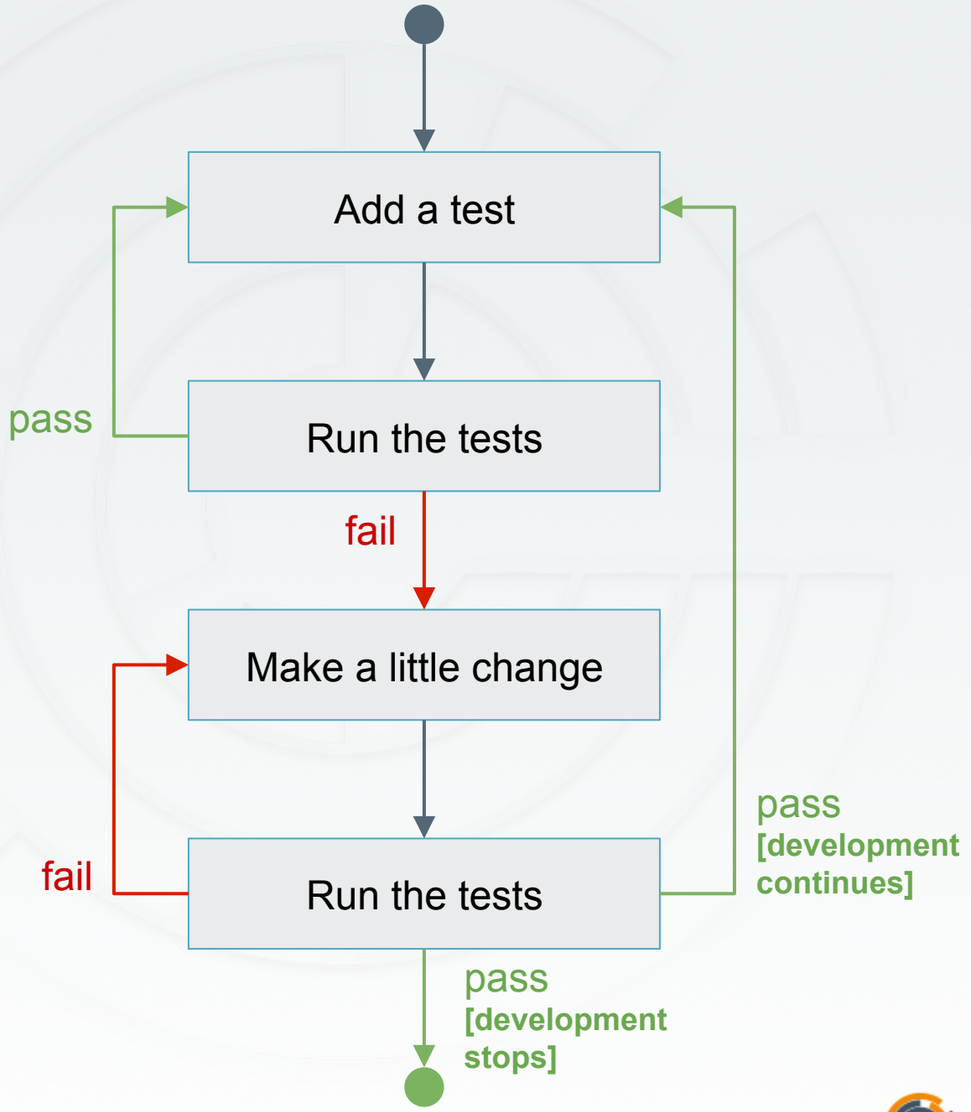


INSPEC

CHEF

Integration Testing – Run the tests

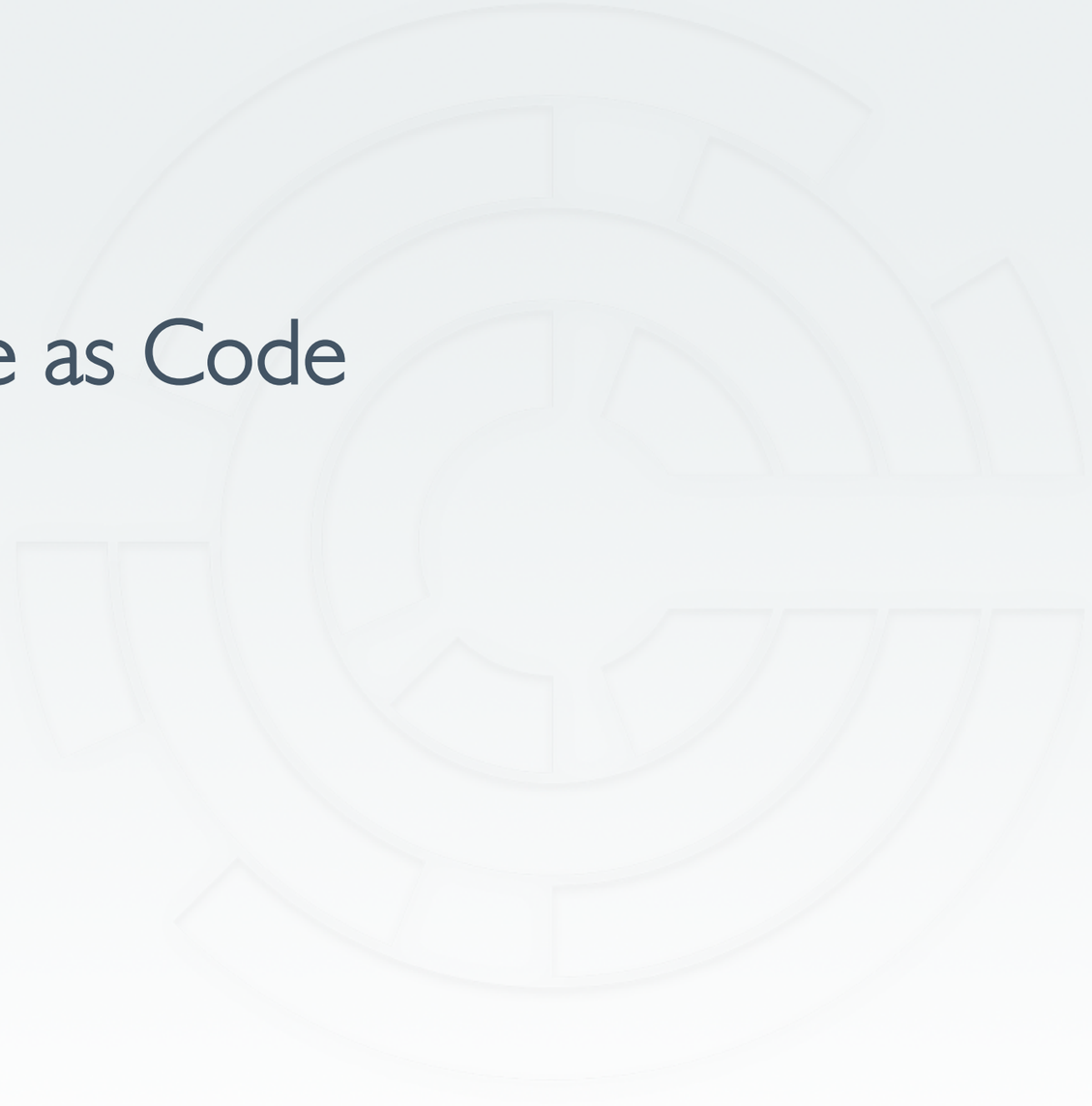




INSPEC

CHEF

Compliance as Code



INSPEC

CHEF

"Society's ability to regulate industries effectively is limited by it's ability to access and understand code, as we saw with the VW emissions scandal." @richardjpope

Pinned Tweet

Justin Arbuckle @dromologue · Mar 11

Arbuckle's law... The actual compliance of an organisation is in direct proportion to the degree to which its policies are expressed as code

RETWEETS 44 FAVORITES 34

INSPEC

CHEF

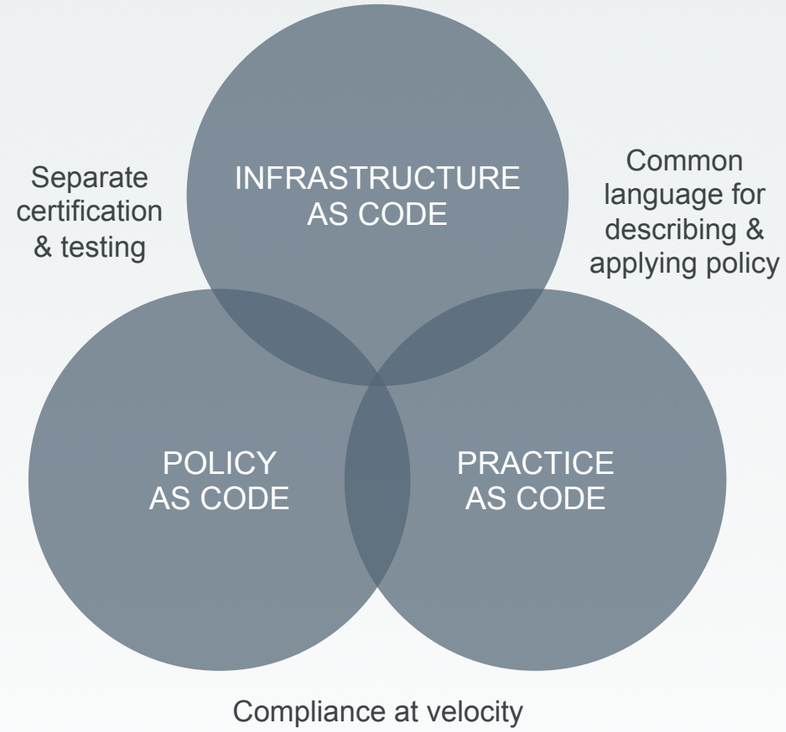
INSPEC

BY CHEF™

InSpec is compliance as code – a human-readable language for automating the continuous testing and compliance auditing of your entire infrastructure.

Compliance as Code

ACCELERATED CYCLE



ROLE OF THE COMPLIANCE OFFICER

One language, One workflow

SSH Control

SSH supports two different protocol versions. The original version, SSHv1, was subject to a number of security issues. Please use SSHv2 instead to avoid these.

Differences in verifying compliance policy

DOCUMENTATION

SSH supports two different protocol versions. The original version, SSHv1, was subject to a number of security issues. Please use SSHv2 instead to avoid these.

SCRIPTING TOOLS

```
 > grep "^Protocol" /etc/ssh/
sshd_config | sed 's/Protocol //'
 2
```

Differences in verifying compliance policy

DOCUMENTATION

SSH supports two different protocol versions. The original version, SSHv1, was subject to a number of security issues. Please use SSHv2 instead to avoid these.

SCRIPTING TOOLS

```
 > grep "^Protocol" /etc/ssh/
sshd_config | sed 's/Protocol //'
 2
```

COMPLIANCE LANGUAGE

```
control 'ssh-1234' do
  impact 1.0
  title 'Server: Set protocol version to SSHv2'
  desc "
    Set the SSH protocol version to 2. Don't use legacy
    insecure SSHv1 connections anymore...
  "

  describe sshd_config do
   its('Protocol') { should eq 2 }
  end
end
```

INSPEC

CHEF

InSpec

- ONE LANGUAGE

- InSpec for Windows

```
control 'windows-base-201' do
  impact 1.0
  title 'Strong Windows NTLMv2 Authentication Enabled; Weak LM
           Disabled'
  desc '
    @link: http://support.microsoft.com/en-us/kb/823659
  '


  describe registry_key
      ('HKLM\System\CurrentControlSet\Control\Lsa') do
    it { should exist }
    its('LmCompatibilityLevel') { should eq 4 }
  end
end
```

INSPEC

CHEF

InSpec

- ONE LANGUAGE

Oracle Solaris 11

- **Baremetal**
- **VMs**
- **Containers**

INSPEC

CHEF

Different ways to run InSpec

Test your machine locally

```
> inspec exec test.rb
```

Test a machine remotely via SSH

```
> inspec exec test.rb -i identity.key -t ssh://root@172.17.0.1
```

No ruby/agent on the node

Test a machine remotely via WinRM

```
> inspec exec test.rb -t winrm://Admin@192.168.1.2 --password super
```

No ruby/agent on the node

Test Docker Container

```
> inspec exec test.rb -t docker://5cc8837bb6a8
```

no SSH/agent in the container

InSpec

- ONE LANGUAGE

Oracle Solaris 11

- Baremetal
- VMs
- Containers
- **Databases**
- **API endpoints (e.g. cloud)**

- Database Testing

```
describe mysql_session.query("SELECT
user,host FROM mysql.user WHERE host = '%'")
do
  its(:stdout) { should be empty }
end
```

- Cloud Provider Testing

```
security_groups.each do |security_group|
  describe security_group do
    it { should_not
have_inbound_rule().with_source('0.0.0.0/0')
}
  end
end
```

INSPEC

CHEF

InSpec

- ONE WORKFLOW
- Security meets operations

InSpec

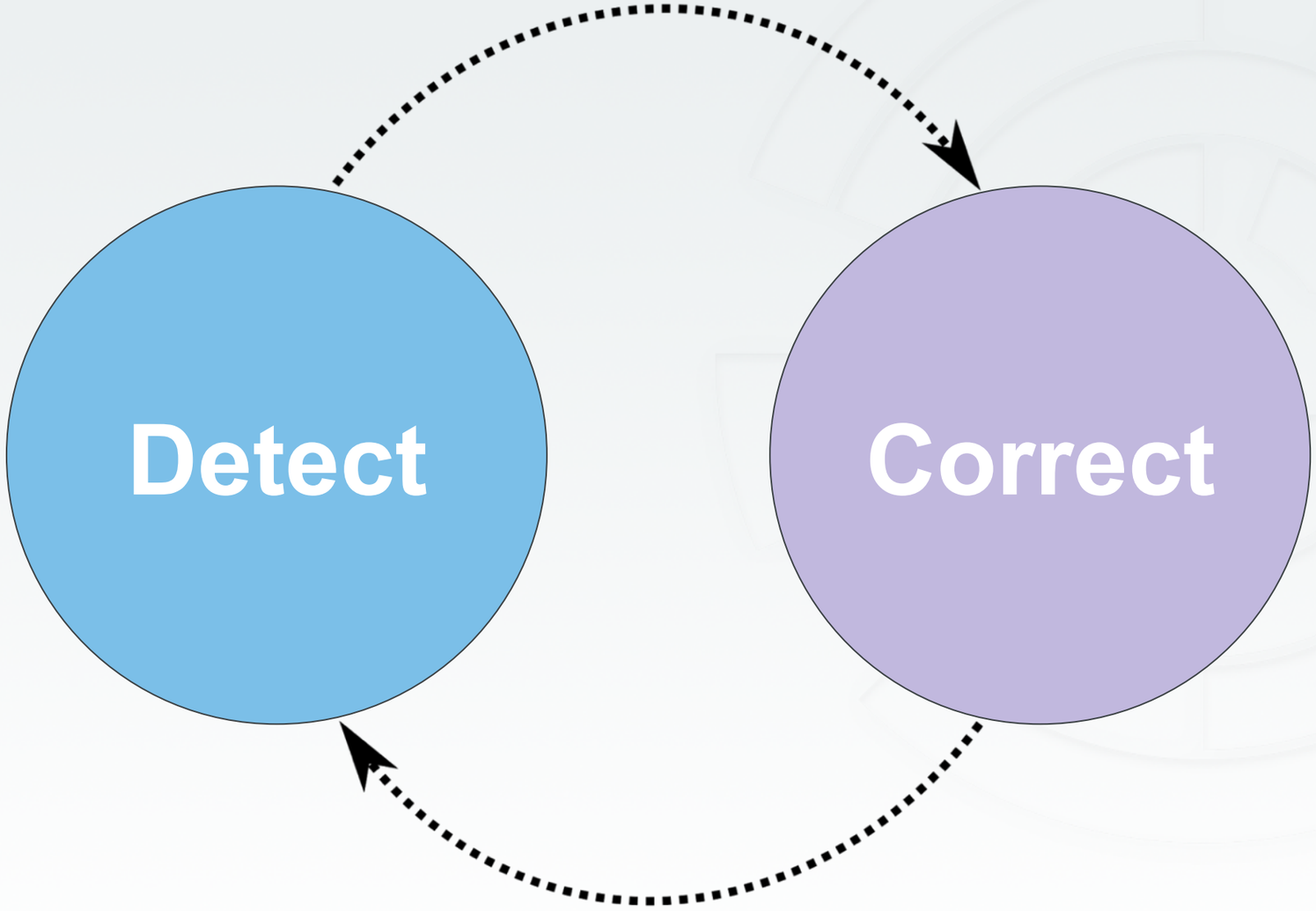
- Each team uses separate tools

- Unified language

INSPEC

CHEF

Continuous Workflow



INSPEC

CHEF

InSpec

Translate compliance into Code
Clearly express statements of policy
Move risk to build/test from runtime
Find issues early
Write code quickly
Run code anywhere
Inspect machines, data, and APIs

INSPEC

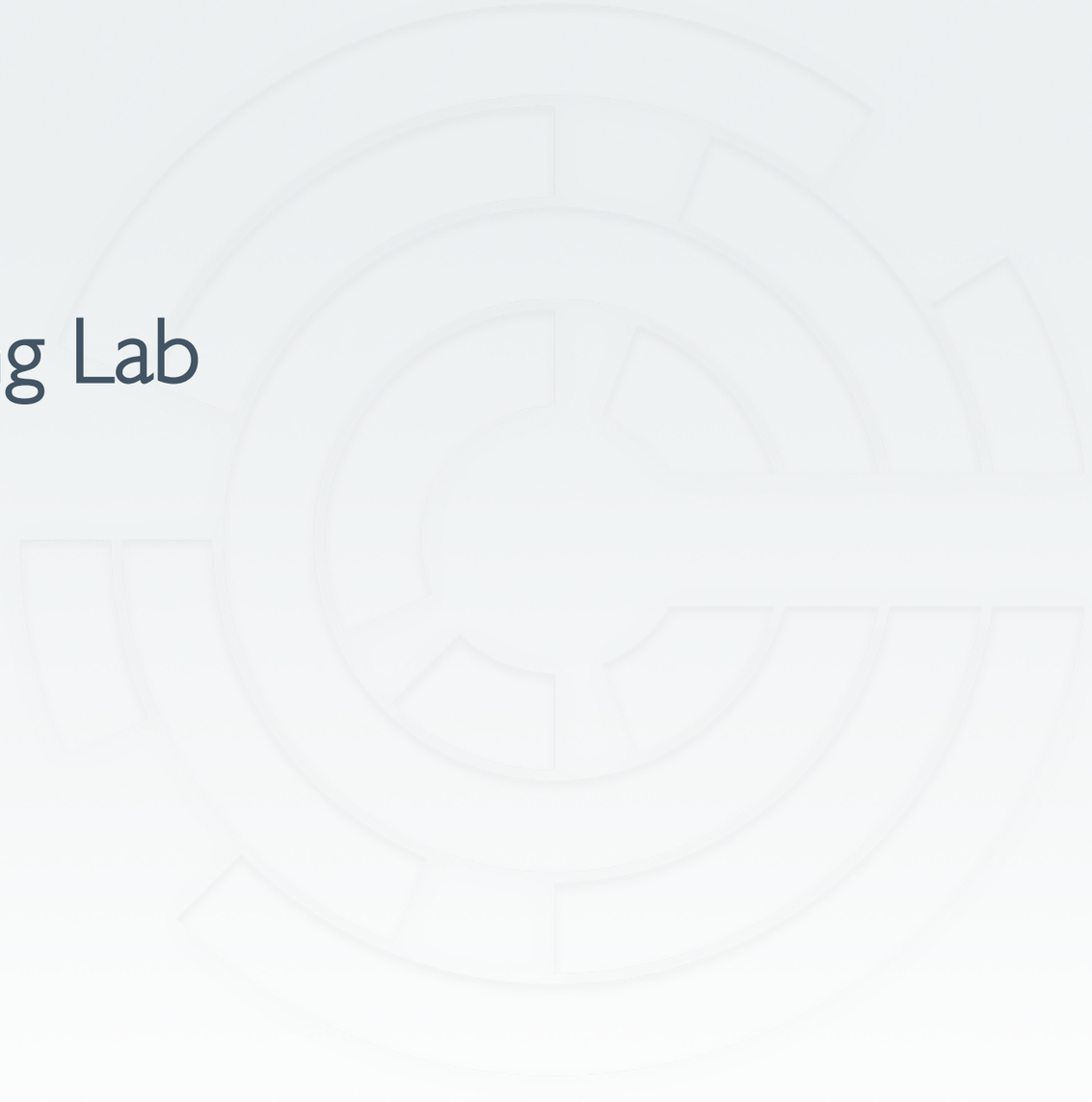
PART OF A PROCESS OF CONTINUOUS COMPLIANCE

A SIMPLE EXAMPLE OF AN INSPEC CIS RULE

```
control 'cis-1.4.1' do
                title '1.4.1 Enable SELinux in /etc/grub.conf'
                desc '
                                        Do not disable SELinux and
enforcing in your GRUB configuration. These are important
security features that prevent attackers from escalating their
access to your systems. For reference see …
                '
                impact 1.0
                expect(grub_conf.param 'selinux').to_not eq '0'
        expect(grub_conf.param 'enforcing').to_not eq '0'
end
```

INSPEC

Learning Lab



INSPEC

CHEF

Find your IP Address

https://goo.gl/WHu8Wg

Login to remote workstation

```
$ ssh chef@IP_ADDRESS
```

```
The authenticity of host '52.54.113.210 (52.54.113.210)' can't be
established.
ECDSA key fingerprint is
SHA256:zAtoeO29XbhRNvwg542cuh4qsKCEaX8hNIlEOCbgd3I.
Are you sure you want to continue connecting (yes/no)?
```

Login to remote workstation

```
$ ssh chef@IP_ADDRESS
```

```
The authenticity of host '52.54.113.210 (52.54.113.210)' can't be
established.
ECDSA key fingerprint is
SHA256:zAtoeO29XbhRNvwg542cuh4qsKCEaX8hNIlEOCbgd3I.
Are you sure you want to continue connecting (yes/no)? yes
```

Login to remote workstation

```
$ ssh chef@IP_ADDRESS
```

```
The authenticity of host '52.54.113.210 (52.54.113.210)' can't be
established.
ECDSA key fingerprint is
SHA256:zAtoeO29XbhRNvwg542cuh4qsKCEaX8hNIlEOCbgd3I.
Are you sure you want to continue connecting (yes/no)? Yes
Warning: Permanently added '52.54.113.210' (ECDSA) to the list of known
hosts.
chef@52.54.113.210's password:
```

Login to remote workstation

```
$ ssh chef@IP_ADDRESS
```

```
The authenticity of host '52.54.113.210 (52.54.113.210)' can't be
established.
ECDSA key fingerprint is
SHA256:zAtoeO29XbhRNvwg542cuh4qsKCEaX8hNIlEOCbgd3I.
Are you sure you want to continue connecting (yes/no)? Yes
Warning: Permanently added '52.54.113.210' (ECDSA) to the list of known
hosts.
chef@52.54.113.210's password: chef
```

Verify the installation

```
$ which inspec
```

```
/opt/chefdk/bin/inspec
```

Verify the installation

```
$ inspec version
```

```
1.11.0

Your version of InSpec is out of date! The latest version is 1.15.0.
```

Verify the installation

```
$ which chef
```

```
/opt/chefdk/bin/chef
```

INSPEC

CHEF

Verify the installation

```
$ chef --version
```

```
Chef Development Kit Version: 1.2.22
chef-client version: 12.18.31
delivery version: master (0b746cafed65a9ea1a79de3cc546e7922de9187c)
berks version: 2017-03-02T09:46:48.762338 20503]
2017-03-02T09:46:48.762505 20503] 2017-03-02T09:46:48.762618 20503]
2017-03-02T09:46:48.762722 20503] 2017-03-02T09:46:48.791141 20503]
2017-03-02T09:46:48.791248 20503] 5.6.0
kitchen version: 1.15.0
```

Chef DK - The Chef Development Kit

. Definitive tooling for local development of Chef code & Infrastructure as Code development

FAST INEXPENSIVE TESTING

Foodcritic

Test Your "Chef Style"

- Validate your Chef code against Chef best practices
- Extend with rules to enforce organizational Chef development best practices
- Enforce compliance & security practices

CookStyle

Validate your Ruby

- Validate your Chef code against Ruby best practices
- Identify potential Ruby errors
  Unclosed strings, etc.
- Identify style/convention that helps write better code
  Single quotes vs. double quotes

ChefSpec

Simulate Chef

- Validate your Chef code will run
- Testing for more Chef advanced use cases
- Useful for regression testing

DEEP INTEGRATION TESTING

Test Kitchen

Let's do this (almost) for real

- Executes your Chef code on an instance or container
- Integrates with Cloud and Virtualization providers
- Validate your Chef code locally before sharing
- Speed development of Chef Cookbooks

InSpec

Verify automation results & ensure compliance

- Assert the intention of your Chef code
- Verify on live systems that your Chef code produced the correct result
- Confirm your Chef code didn't not produce compliance drift

INSPEC

CHEF

http://community-slack.chef.io

InSpec CLI

Command Line Interface

INSPEC

CHEF

Objectives

Create and execute a simple compliance check

LAB

Ensure SSH Protocol is set to 2

- ❑ Review the Center for Internet Security control
- ❑ Create an InSpec profile to verify the control
- ❑ Execute the InSpec profile to determine current system state

CHEF

Compliance Mandate

Center for Internet Security

5.2.2 Ensure SSH Protocol is set to 2 (Scored)

Profile Applicability:

- Level 1 - Server
- Level 1 - Workstation

Description:

SSH supports two different and incompatible protocols: SSH1 and SSH2. SSH1 was the original protocol and was subject to security issues. SSH2 is more advanced and secure.

Rationale:

SSH v1 suffers from insecurities that do not affect SSH v2.

Audit:

Run the following command and verify that output matches:

```
# grep "^Protocol" /etc/ssh/sshd_config
Protocol 2
```

Remediation:

Edit the `/etc/ssh/sshd_config` file to set the parameter as follows:

```
Protocol 2
```

CIS Red Hat Enterprise Linux 7 Benchmark

v2.1.0 - 06-02-2016

Run InSpec

```
$ inspec
```

```
Commands:
  inspec archive PATH                # archive a profile to tar.gz (default) or zip
  inspec artifact SUBCOMMAND ...     # Sign, verify and install artifacts
  inspec check PATH                  # verify all tests at the specified PATH
  inspec compliance SUBCOMMAND ...   # Chef Compliance commands
  inspec detect                      # detect the target OS
  inspec env                         # Output shell-appropriate completion configuration
  inspec exec PATHS                  # run all test files at the specified PATH.
  inspec help [COMMAND]              # Describe available commands or one specific command
  inspec init TEMPLATE ...           # Scaffolds a new project
  inspec json PATH                   # read all tests in PATH and generate a JSON summary
  inspec shell                       # open an interactive debugging shell
  inspec supermarket SUBCOMMAND ...  # Supermarket commands
  inspec vendor PATH                 # Download all dependencies and generate a lockfile in a `vendor` directory
  inspec version                     # prints the version of this tool

Options:
  l, [--log-level=LOG_LEVEL]          # Set the log level: info (default), debug, warn, error
      [--log-location=LOG_LOCATION]   # Location to send diagnostic log messages to. (default: STDOUT or STDERR)
      [--diagnose], [--no-diagnose]   # Show diagnostics (versions, configurations)
```

Questions

What version of InSpec is installed?
What type(s) of projects with inspec init generate?

Build scaffold for an ssh profile

```
$ inspec init profile ssh
```

```
Create new profile at /home/chef/ssh
 * Create file README.md
 * Create directory controls
 * Create file controls/example.rb
 * Create file inspec.yml
 * Create directory libraries
```

Open the example control

/home/chef/ssh/controls/example.rb

```
# encoding: utf-8
# copyright: 2015, The Authors
# license: All rights reserved

title 'sample section'

# you can also use plain tests
describe file('/tmp') do
  it { should be_directory }
end

# you add controls here
control 'tmp-1.0' do                        # A unique ID for this control
  impact 0.7                                # The criticality, if this control fails.
  title 'Create /tmp directory'             # A human-readable title
  desc 'An optional description...'
  describe file('/tmp') do                  # The actual test
    it { should be_directory }
  end
end
```

Execute the example control

```
$ inspec exec ssh
```

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


  ✔  tmp-1.0: Create /tmp directory
     ✔  File /tmp should be directory

  File /tmp
     ✔  should be directory

Profile Summary: 1 successful, 0 failures, 0 skipped
Test Summary: 2 successful, 0 failures, 0 skipped
```

Rename the example control

```
$ mv ~/ssh/controls/example.rb ~/ssh/controls/server.rb
```

Rewrite the control

/home/chef/ssh/controls/server.rb

```
# 5.2.2 Ensure SSH Protocol is set to 2
#
# grep "^Protocol" /etc/ssh/sshd_config
# Protocol 2
#
describe file('/etc/ssh/sshd_config') do
  its('content') { should match /^Protocol 2/ }
end
```

Execute the control

```
$ inspec exec ssh
```

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



  File /etc/ssh/sshd_config
     ∅  content should match /^Protocol 2/
     expected "# This config file was generated by Chef\n\n#       $OpenBSD: sshd_config,v 1.93 2014/01/10 05:59:19...XMODIFIERS\n\n#
override default of no subsystems\nSubsystem sftp  /usr/libexec/openssh/sftp-server" to match /^Protocol 2/
     Diff:
     @@ -1,2 +1,78 @@
     -/^Protocol 2/
     +# This config file was generated by Chef
...
 +Subsystem sftp  /usr/libexec/openssh/sftp-server



Test Summary: 0 successful, 1 failures, 0 skipped
```

DISCUSSION

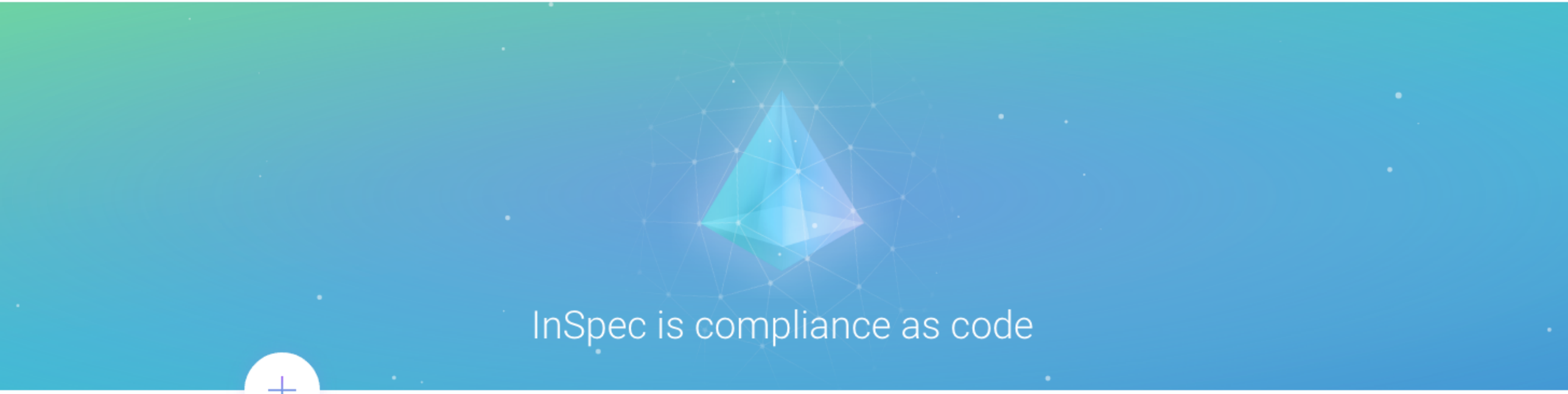
Wait a minute…

Where did `its('content')` come from?
What other file attributes can we write tests for?
Where does one go to find out more information about these resources?

inspec.io

INSPEC by CHEF

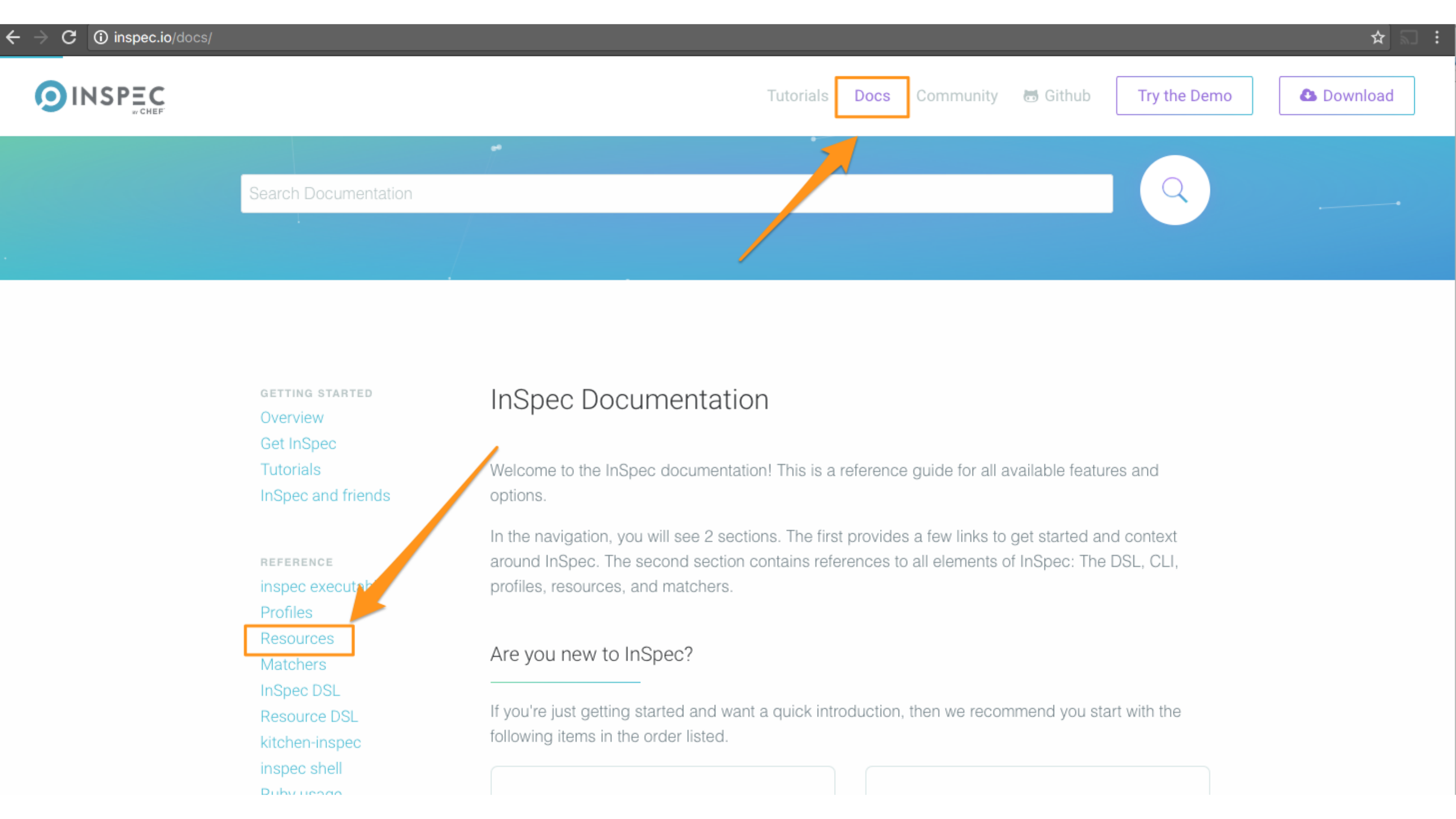
Tutorials Docs Community Github Try the Demo Download



InSpec is compliance as code

Automated testing, codified

InSpec is an open-source testing framework for infrastructure with a human-readable language for specifying compliance, security and other policy requirements. Easily integrate automated tests that check for adherence to policy into any stage of your deployment pipeline.


inspec.io/docs/
INSPEC by CHEF
Tutorials
Docs
Community
Github
Try the Demo
Download
Search Documentation
GETTING STARTED
Overview
Get InSpec
Tutorials
InSpec and friends
REFERENCE
inspec executable
Profiles
Resources
Matchers
InSpec DSL
Resource DSL
kitchen-inspec
inspec shell
InSpec Documentation
Welcome to the InSpec documentation! This is a reference guide for all available features and options.
In the navigation, you will see 2 sections. The first provides a few links to get started and context around InSpec. The second section contains references to all elements of InSpec: The DSL, CLI, profiles, resources, and matchers.
Are you new to InSpec?
If you're just getting started and want a quick introduction, then we recommend you start with the following items in the order listed.

InSpec Resources Reference

The following InSpec audit resources are available:

file

Use the `file` InSpec audit resource to test all system file types, including files, directories, symbolic links, named pipes, sockets, character devices, block devices, and doors.

Syntax

A `file` resource block declares the location of the file type to be tested, what type that file should be (if required), and then one (or more) matchers:

```
describe file('path') do
  it { should MATCHER 'value' }
end
```

where

`('path')` is the name of the file and/or the path to the file
`MATCHER` is a valid matcher for this resource
`'value'` is the value to be tested

Matchers

This InSpec audit resource has the following matchers:

be

Use the `be` matcher to use a comparison operator—`=` (equal to), `>` (greater than), `<` (less than), `>=` (greater than or equal to), and `<=` (less than or equal to)—to compare two values: `its('value') { should be >= value }`, `its('value') { should be < value }`, and so on.

be_block_device

File Resource

```
describe file('path') do
  it { should MATCHER 'value' }
end
```

Use the file resource to test all system file types, including files, directories, symbolic links, named pipes, sockets, character devices, block devices, and doors.

Test if a file exists

```
describe file('/tmp') do
 it { should exist }
end
```

Test if a path is a directory

```
describe file('/tmp') do
 its('type') { should eq :directory }
 it { should be_directory }
end
```

Content Matcher

```
describe file('/etc/ssh/sshd_config') do
  its('content') { should match /^Protocol 2/ }
end
```

The content matcher tests if contents in the file match the value specified in a regular expression. The values of the content matcher are arbitrary and depend on the file type being tested and also the type of information that is expected to be in that file

LAB

Ensure SSH Protocol is set to 2

- ✓ Review the Center for Internet Security control
- ✓ Create an InSpec profile to verify the control
- ✓ Execute the InSpec profile to determine current system state

CHEF

PROBLEM

There are some problems!

Location of the SSH server configuration is hard-coded
Regular expressions are difficult

LAB

Refactor our control

- Use a different resource

DISCUSSION

Which resource

Is there a better resource that we could use?
What might a refactored test look like?

Refactored Control

/home/ssh/controls/server.rb

```
describe sshd_config do
  its('Protocol') { should cmp 2 }
end
```

Resource: sshd_config

```
describe sshd_config('path') do
  its('name') { should include('foo') }
end
```

where

`name` is a configuration setting in sshd_config

`('path')` is the non-default /path/to/sshd_config

`{ should include('foo') }` tests the value of name as read from sshd_config versus the value declared in the test

Use the sshd_config resource to test configuration data for the OpenSSH daemon located at /etc/ssh/sshd_config on Linux and Unix platforms.

Execute the control

```
$ inspec exec ssh
```

```
 SSH Configuration
     Ø  Protocol should cmp == 2

     expected: 2
          got:

     (compared using `cmp` matcher)


Test Summary: 0 successful, 1 failures, 0 skipped
```

LAB

Refactor our control

- ✓ Use a different resource

LAB

Execute profile on a remote machine

- Execute your ssh profile against the instructor's machine

Different ways to run InSpec

Test your machine locally

```
> inspec exec test.rb
```

Test a machine remotely via SSH

```
> inspec exec test.rb -i identity.key -t ssh://root@172.17.0.1
```

No ruby/agent on the node

Test a machine remotely via WinRM

```
> inspec exec test.rb -t winrm://Admin@192.168.1.2 --password super
```

No ruby/agent on the node

Test Docker Container

```
> inspec exec test.rb -t docker://5cc8837bb6a8
```

no SSH/agent in the container

Execute the control

```
$ inspec exec ssh -t ssh://ec2-35-156-226-39.eu-central-1.compute.amazonaws.com --user=chef --password=chef
```

```
 SSH Configuration
     Ø  Protocol should cmp == 2

     expected: 2
          got:

     (compared using `cmp` matcher)


Test Summary: 0 successful, 1 failures, 0 skipped
```

LAB

Execute profile on a remote machine

- ✓ Execute your ssh profile against the instructor's machine

LAB

Enrich our profile

- Add additional metadata to our control

Compliance Mandate

Center for Internet Security

5.2.2 Ensure SSH Protocol is set to 2 (Scored)

Profile Applicability:

- Level 1 - Server
- Level 1 - Workstation

Description:

SSH supports two different and incompatible protocols: SSH1 and SSH2. SSH1 was the original protocol and was subject to security issues. SSH2 is more advanced and secure.

Rationale:

SSH v1 suffers from insecurities that do not affect SSH v2.

Audit:

Run the following command and verify that output matches:

```
# grep "^Protocol" /etc/ssh/sshd_config
Protocol 2
```

Remediation:

Edit the `/etc/ssh/sshd_config` file to set the parameter as follows:

```
Protocol 2
```

CIS Red Hat Enterprise Linux 7 Benchmark

v2.1.0 - 06-02-2016

Server Control

/home/ssh/controls/server.rb

```
describe sshd_config do
  its('Protocol') { should cmp 2 }
end
```

Enriched Server Control

/home/ssh/controls/server.rb

```
control '5.2.2' do
    impact 1.0

    title 'Ensure SSH Protocol is set to 2'

    desc <<-EOF
      SSH supports two different and incompatible protocols: SSH1 and SSH2.
      SSH1 was the original protocol and was subject to security issues.
      SSH2 is more advanced and secure.

      SSH v1 suffers from insecurities that do not affect SSH v2.
    EOF

    tag 'ssh', 'sshd', 'server', 'workstation'

    ref 'SSH Protocol', url: 'https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/System_Administrators_Guide...'

    describe sshd_config do
        its('Protocol') { should cmp 2 }
    end
end
```

Additional metadata for controls

The following example illustrates various ways to add tags and references to `control`

```
control 'ssh-1' do
  impact 1.0

  title 'Allow only SSH Protocol 2'
  desc 'Only SSH protocol version 2 connections should be permitted.
        The default setting in /etc/ssh/sshd_config is correct, and can be
        verified by ensuring that the following line appears: Protocol 2'

  tag 'production','development'
  tag 'ssh','sshd','openssh-server'

  tag cce: 'CCE-27072-8'
  tag disa: 'RHEL-06-000227'

  tag remediation: 'stig_rhel6/recipes/sshd-config.rb'
  tag remediation: 'https://supermarket.chef.io/cookbooks/ssh-hardening'

  ref 'NSA-RH6-STIG - Section 3.5.2.1', url: 'https://www.nsa.gov/ia/_files/os/re
  ref 'http://people.redhat.com/swells/scap-security-guide/RHEL/6/output/ssg-cent

  describe ssh_config do
    its ('Protocol') { should eq '2'}
  end
end
```

Compliance Profile Severity Mapping

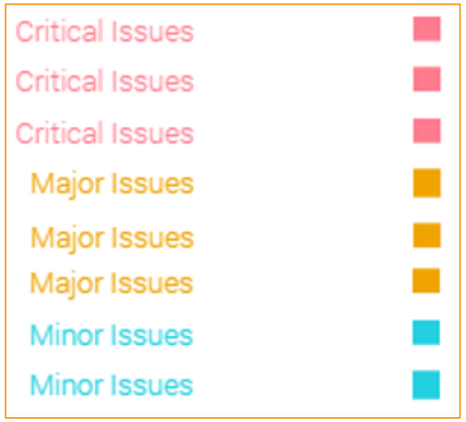
The table below shows the current mapping of Compliance Profile **impact** numbering to severity.

Set the SSH protocol version to 2. Don't use legacy insecure S

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

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

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



Execute the control

```
$ inspec exec ssh
```

```
×  5.2.2: Ensure SSH Protocol is set to 2 (
   expected: 2
        got:

   (compared using `cmp` matcher)
   )
   ×  SSH Configuration Protocol should cmp == 2

   expected: 2
        got:

   (compared using `cmp` matcher)
```

LAB

Enrich our profile

- ✓ Add additional metadata to our control

CHEF

Objectives

- ✓ Execute an InSpec test on a local machine
- ✓ Execute an InSpec test on a remote machine
- ✓ Generate an InSpec profile

Add InSpec-based integration test to a Chef cookbook

Run InSpec-based integrations tests during Chef cookbook development

List additional resources and places to look for support with InSpec

InSpec for Integration Testing

LAB

Simple Web Server Cookbook

- ❑ The Apache web server should be running
- ❑ The Apache web server should be listening on the default port
- ❑ The Apache web server should be configured to start on boot

Create a directory for cookbooks

```
$ mkdir -p ~/cookbooks
```

Move into the directory for cookbooks

```
$ cd ~/cookbooks
```

Generate an apache cookbook

```
$ chef generate cookbook apache
```

```
Generating cookbook apache
- 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 apache` 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/smoke/default/default_test.rb

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

recipes/default.rb
```

Move into the apache cookbook's directory

```
$ cd ~/cookbooks/apache
```

```
Generating cookbook apache
- 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 apache` 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/smoke/default/default_test.rb

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

recipes/default.rb
```

Remember…

Infrastructure policies need testing

- ↳ Linting
- ↳ Static Analysis
- ↳ Unit Testing
- ↳ Integration Testing
- ↳ Compliance Testing

“Infrastructure as Code” should be tested like ANY other codebase.

Remember…

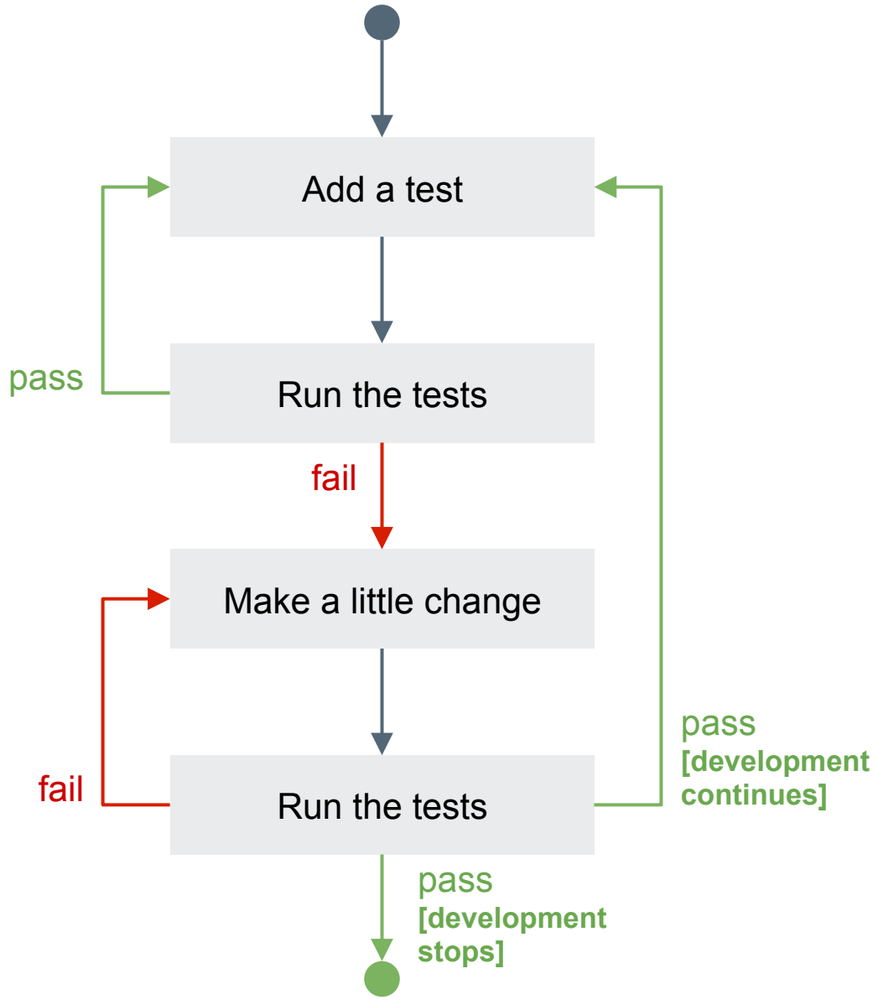
Infrastructure policies need testing

- ↳ Linting
- ↳ Static Analysis
- ↳ Unit Testing
- ↳ Integration Testing
- ↳ Compliance Testing

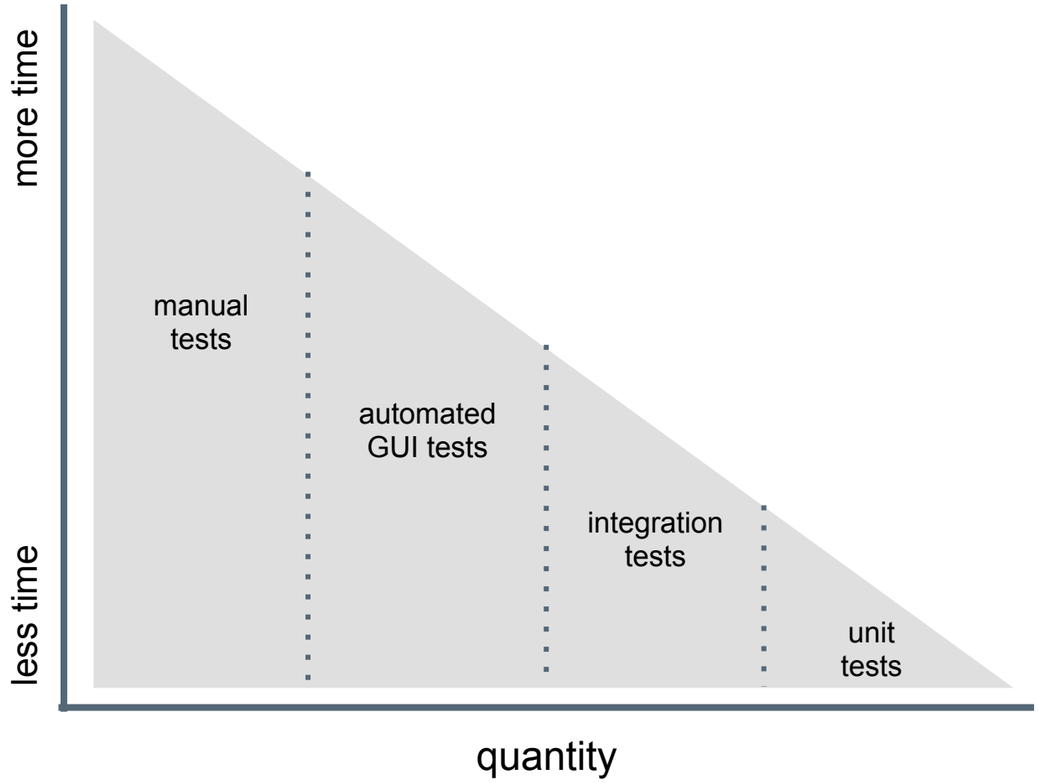
“Infrastructure as Code” should be tested like ANY other codebase.

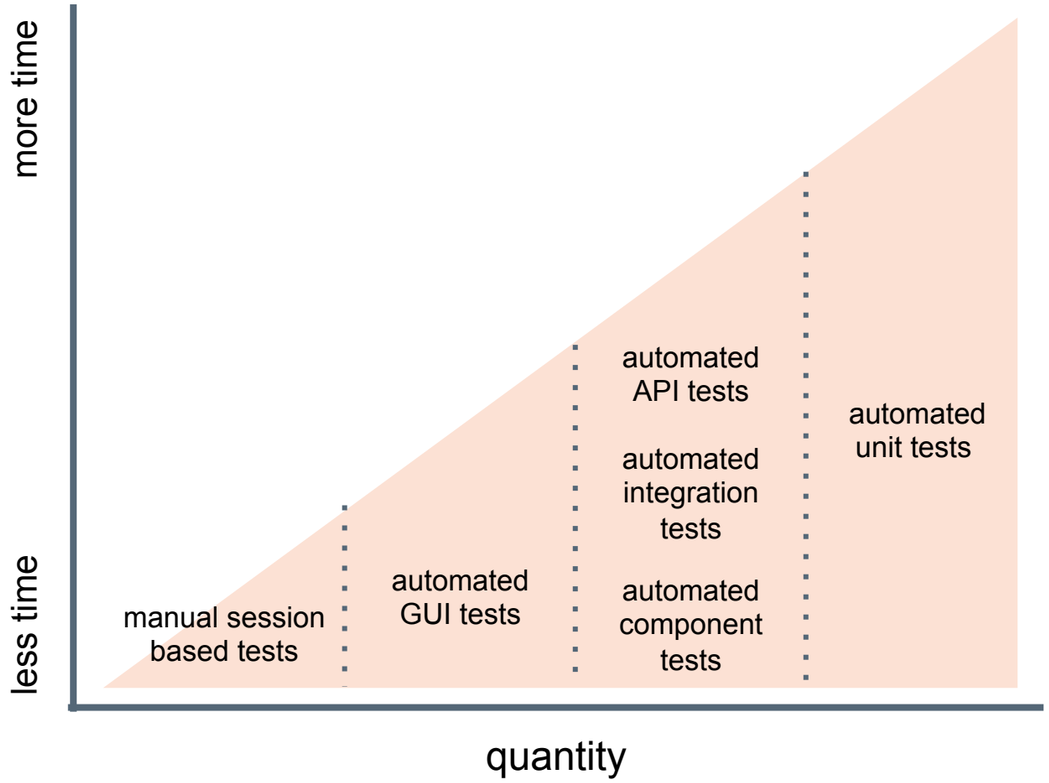
Test-driven Development

- 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



Software Testing and Why it Matters





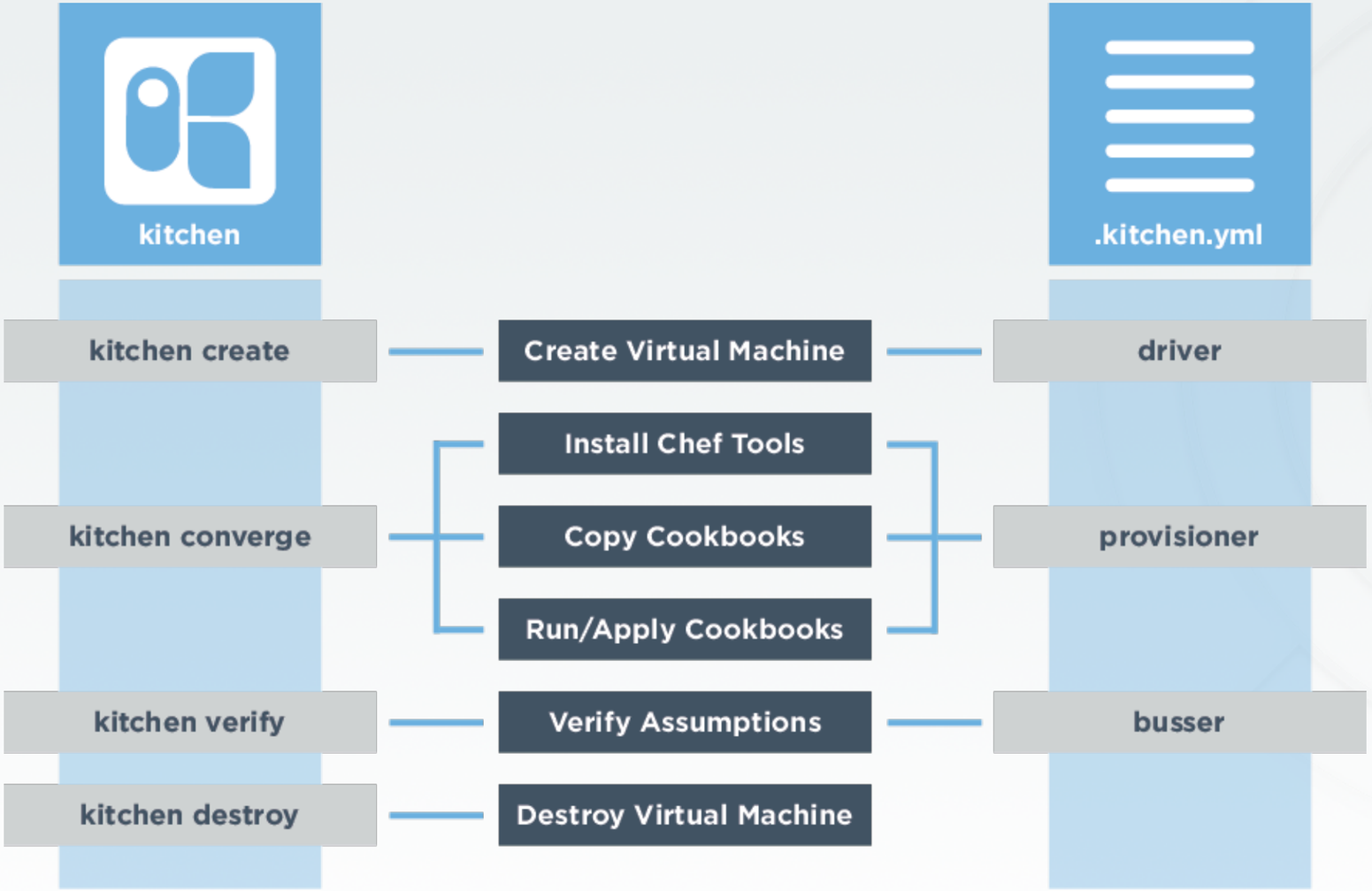
Testing **builds safety** through feedback loops

Inexpensive experiments to provide validation

Reduces risk

Optimize Testing: Do more of the inexpensive testing first!

How do we write and test our intended change?



amazon web services

Microsoft Azure

vmware®

openstack™

INSPEC

CHEF

What can delivery local do?

```
$ delivery local --help
```

```
delivery-local
Run Delivery phases on your local workstation.

USAGE:
    delivery local [FLAGS] <phase>

FLAGS:
    -h, --help               Prints help information
        --no-spinner         Disable the spinner
        --non-interactive    Disable cli interactions
    -V, --version            Prints version information

ARGS:
    <phase>    Delivery phase to execute [values: unit, lint, syntax, provision, deploy, smoke, cleanup]
```

Test Kitchen

kitchen list
kitchen create
kitchen converge
kitchen verify
kitchen destroy
kitchen test
kitchen login

Delivery

[No matching delivery command]
delivery local provision
delivery local deploy
delivery local smoke
delivery local cleanup
delivery local all
[No matching delivery command]

`~/cookbooks/apache/.delivery/project.toml`

Update the Kitchen Configuration

~/cookbooks/apache/.kitchen.yml

```
---
driver:
  name: vagrant
  name: docker
```

Update the Kitchen Configuration

~/cookbooks/apache/.kitchen.yml

```
platforms:
  - name: ubuntu-16.04
  - name: centos-7.2
  - name: centos-6.8
```

Final .kitchen.yml

~/cookbooks/apache/.kitchen.yml

```
---
driver:
  name: docker

provisioner:
  name: chef_zero
  # You may wish to disable always updating cookbooks in CI or other testing environments.
  # For example:
  #   always_update_cookbooks: <%= !ENV['CI'] %>
  always_update_cookbooks: true

verifier:
  name: inspec

platforms:
  - name: centos-6.8

suites:
  - name: default
    run_list:
      - recipe[apache::default]
    verifier:
      inspec_tests:
        - test/smoke/default
    attributes:
```

List Kitchen Instances

```
$ kitchen list
```

```
Instance             Driver   Provisioner  Verifier  Transport  Last Action    Last Error
default-centos-68    Docker   ChefZero     Inspec    Ssh        <Not Created>  <None>
```

Run the integration tests

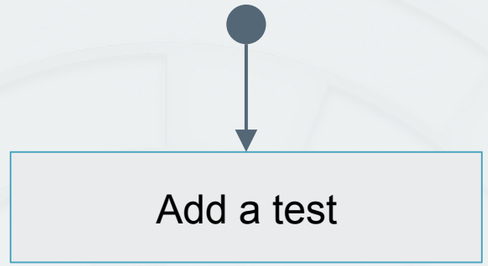
```
$ delivery local smoke
```

```
Chef Delivery
Running Smoke Phase
-----> Starting Kitchen (v1.14.2)
-----> Creating <default-centos-68>...
       Sending build context to Docker daemon 193.5 kB
       Step 1 : FROM centos:centos7
 ...
-----> Verifying <default-centos-68>...
       Loaded

Target:  ssh://kitchen@localhost:32768


  User root
     ✔  should exist
     ↻  This is an example test, replace with your own test.
  Port 80
     ✔  should not be listening
     ↻  This is an example test, replace with your own test.

Test Summary: 2 successful, 0 failures, 2 skipped
       Finished verifying <default-centos-68> (0m0.45s).
-----> Kitchen is finished. (1m4.44s)
```


CHEF

Integration Testing – Add tests

```
describe package 'httpd' do
  it { should be_installed }
end

describe service 'httpd' do
  it { should be_running }
  it { should be_enabled }
end

describe port(80) do
  it { should be_listening }
end
```



INSPEC

CHEF

Resource: package

```
describe package('name') do
  it { should be_installed }
end
```

where

```
('name') must specify the name of a package, such as 'nginx'
be_installed is a valid matcher for this resource
```

Use the package resource to test if the named package and/or package version is installed on the system.

Resource: service

```
describe service('service_name') do
  it { should be_installed }
  it { should be_enabled }
  it { should be_running }
end
```

Use the service resource to test if the named service is installed, running and/or enabled.

Resource: port

```
describe port(514) do
  it { should be_listening }
end
```

Use the port InSpec audit resource to test basic port properties, such as port, process, if it's listening.

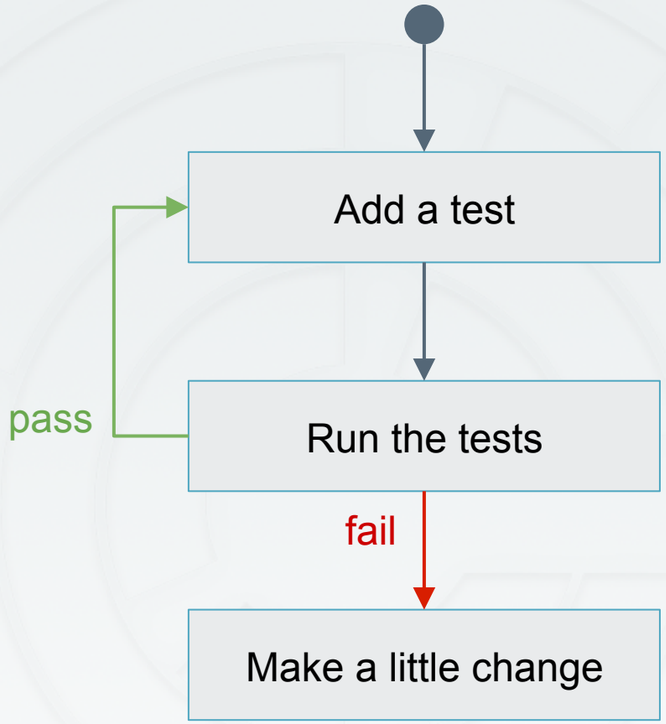
Run the Integration Test

```
$ delivery local smoke
```

```
System Package
   Ø  httpd should be installed
   expected that `System Package httpd` is installed
Service httpd
   Ø  should be running
   expected that `Service httpd` is running
   Ø  should be enabled
   expected that `Service httpd` is enabled
Port 80
   Ø  should be listening
   expected `Port 80.listening?` to return true, got false

Test Summary: 0 successful, 4 failures, 0 skipped
```

Integration Testing – Make a change

```
package 'httpd' do
  action :install
end

service 'httpd' do
  action [ :start, :enable ]
end
```



Apply the change

```
$ delivery local deploy
```

```
Chef Delivery
Running Deploy Phase
-----> Starting Kitchen (v1.14.2)
-----> Converging <default-centos-68>...
       Preparing files for transfer
       Preparing dna.json
       Resolving cookbook dependencies with Berkshelf 5.2.0...
...
Recipe: apache::default
 * yum_package[httpd] action install
   - install version 2.2.15-56.el6.centos.3 of package httpd
 * service[httpd] action start
   - start service service[httpd]
 * service[httpd] action enable
   - enable service service[httpd]

Running handlers:
Running handlers complete
Chef Client finished, 3/3 resources updated in 06 seconds
Finished converging <default-centos-68> (0m11.54s).
-----> Kitchen is finished. (0m12.42s)
```

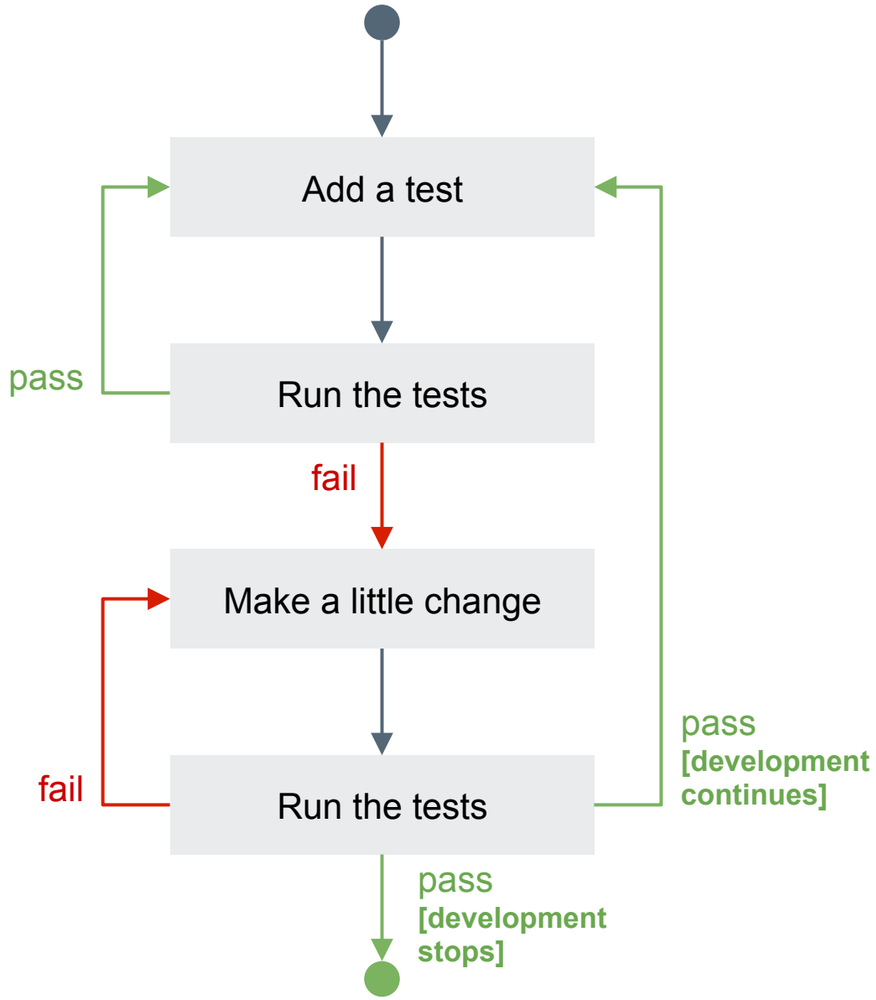
Run the Integration Test

```
$ delivery local smoke
```

```
System Package
   ✔  httpd should be installed
Service httpd
   ✔  should be running
   ✔  should be enabled
Port 80
   ✔  should be listening

Test Summary: 4 successful, 0 failures, 0 skipped
     Finished verifying <default-centos-68> (0m0.58s).
-----> Kitchen is finished. (0m1.46s)
```

Integration Testing – Run the tests

```
-----> Verifying <default-centos-72>...
       Loaded

Target:  ssh://vagrant@127.0.0.1:2222


  System Package
     ✔  httpd should be installed
  Service httpd
     ✔  should be running
     ✔  should be enabled
  Port 80
     ✔  should be listening

Test Summary: 4 successful, 0 failures, 0 skipped
       Finished verifying <default-centos-72> (0m0.91s).
```



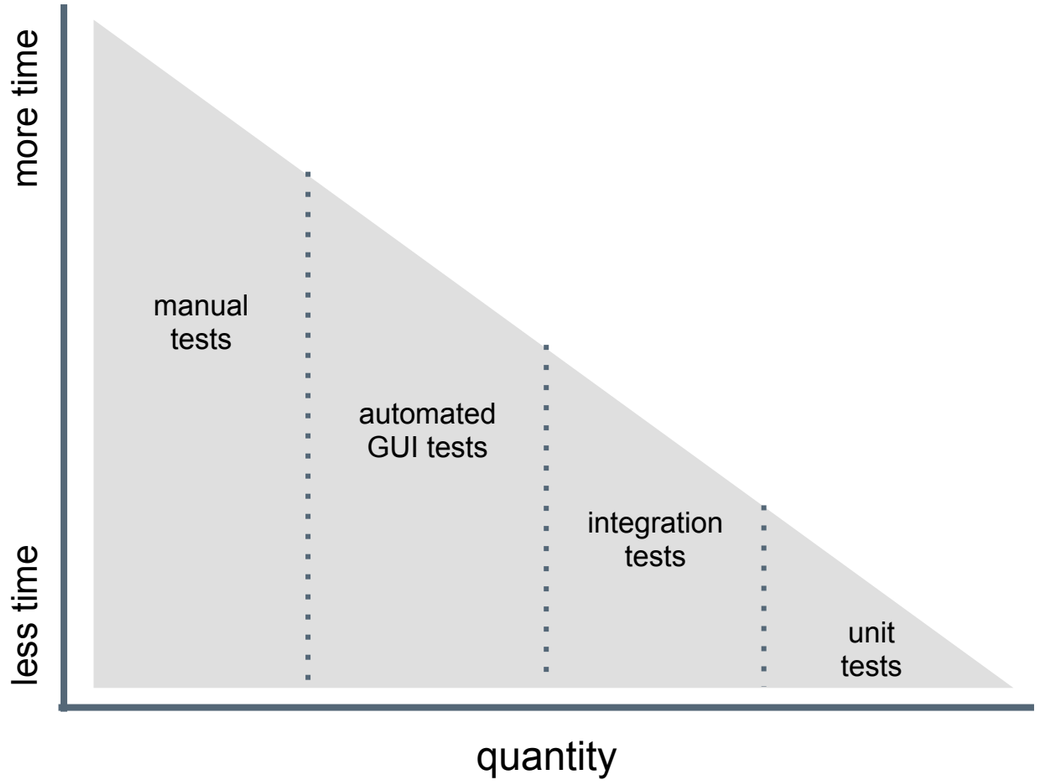
LAB

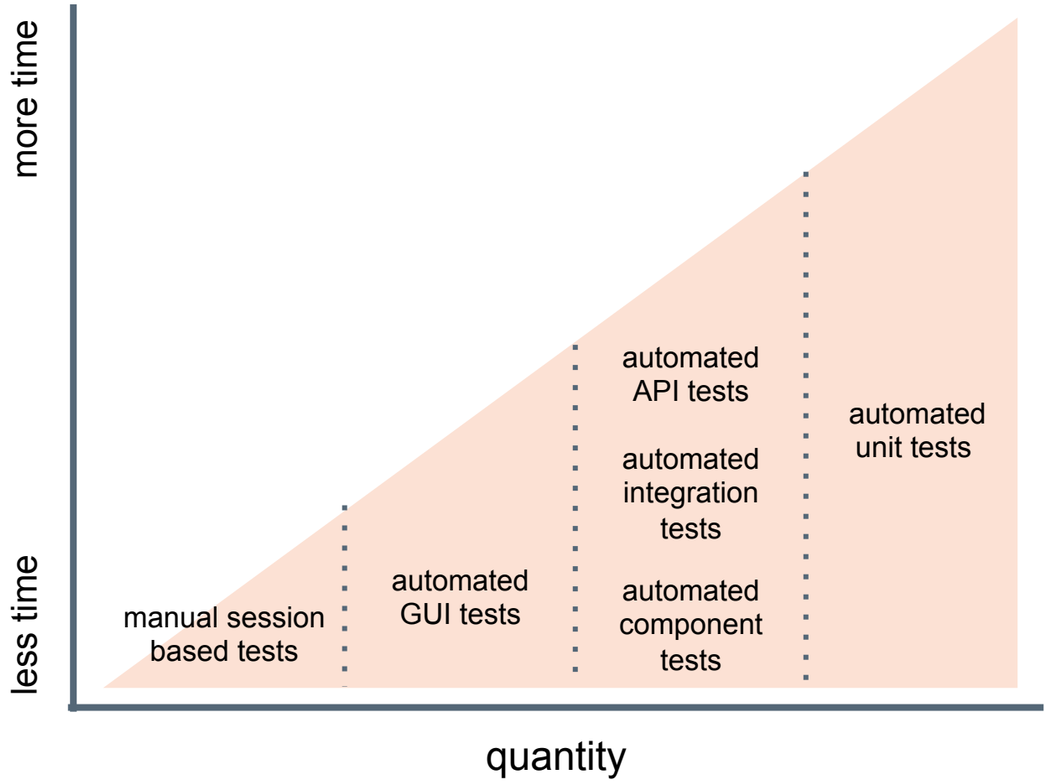
Simple Web Server Cookbook

- ✓ The Apache web server should be running
- ✓ The Apache web server should be listening on the default port
- ✓ The Apache web server should be configured to start on boot

CHEF

Software Testing and Why it Matters





Testing **builds safety** through feedback loops

Inexpensive experiments to provide validation

Reduces risk

Optimize Testing: Do more of the inexpensive testing first!

DISCUSSION

Additional Testing

What command would you use to complete lint testing?
What command would you use to complete syntax testing?
What command would you run to complete unit tests?
What additional kitchen-related commands should we run?
How would you InSpec exec your tests over the docker protocol?

Objectives

- ✓ Execute an InSpec test on a local machine
- ✓ Execute an InSpec test on a remote machine
- ✓ Generate an InSpec profile
- ✓ Add InSpec-based integration test to a Chef cookbook
- ✓ Run InSpec-based integrations tests during Chef cookbook development

List additional resources and places to look for support with InSpec

Continuous Compliance

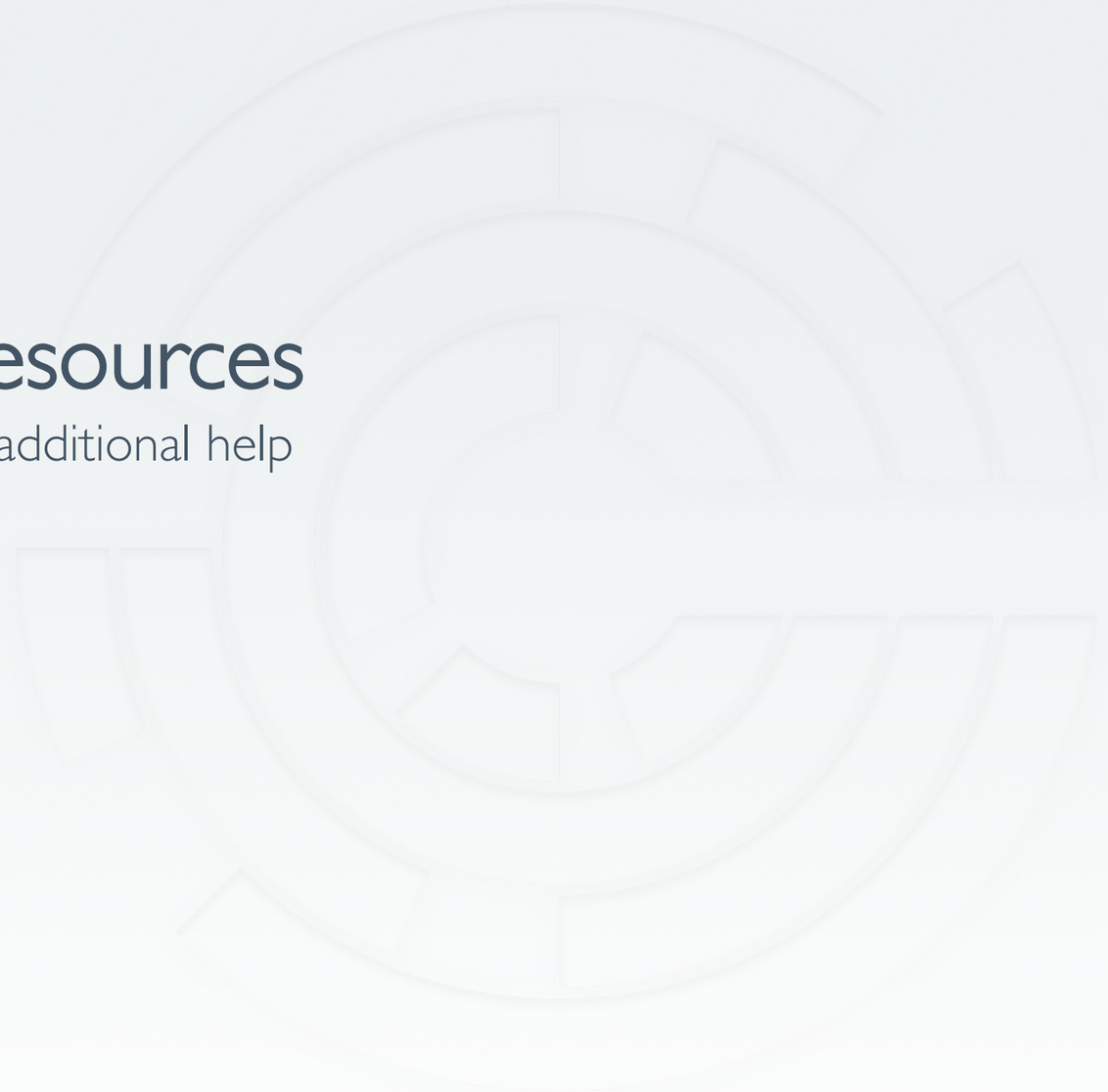
Stick with me…but the slides aren't really ready, we're doing this bit live

INSPEC

CHEF

Further Resources

Where to go for additional help



Community Resources

InSpec Website, includes tutorials and docs - http://inspec.io/

#inspec channel of the Chef Community Slack - http://community-slack.chef.io/

InSpec category of the Chef Mailing List - https://discourse.chef.io/c/inspec

Compliance Profiles on the Supermarket - https://supermarket.chef.io/tools?type=compliance_profile

Open Source Project - https://github.com/chef/inspec

DAY 1 // MAY 22

- ★ Workshops & Chef Training
- ★ DevOps Leadership Summit
- ★ Community Summit
- ★ Partner Summit
- ★ Welcome Reception
- ★ Customer Dinner
- ★ Analyst Day

DAY 2 // MAY 23

- ★ Keynotes
- ★ Technical Sessions
- ★ Happy Hour
- ★ Game Night
- ★ Executive Dinner

DAY 3 // MAY 24

- ★ Keynotes
- ★ Technical Sessions
- ★ Awesome Chef Awards
- ★ Community Celebration

DAY 4 // MAY 25

- ★ Hackday

• Exhibit Hall Open & Sales suites available • chefconf.chef.io •


MAY 22-24 | AUSTIN
CHEFCONF
2017
chefconf.chef.io
REGISTER NOW
$995 EARLY
BIRD PRICE
ENDS MARCH 31st

Workshop this afternoon

CHEF™

More hands-on Exercises

Still have time to play around?

LAB

Remediate ssh Protocol

- Write a cookbook to manage ssh
- Verify the Protocol is set appropriately
- Apply the recipe to your local machine with `chef-client --local` (if you dare!)
- Verify your colleague's work

LAB

Build additional CIS controls

- https://github.com/chef-training/workshops/tree/master/InSpec

LAB

Supermarket Profiles

- ❑ List profiles available on the Supermarket
- ❑ View information about the dev-sec/ssh-baseline profile
- ❑ Execute the dev-sec/ssh-baseline profile from the Supermarket
- ❑ Write a cookbook to remediate failing controls from dev-sec/ssh-baseline

LAB

Unit Testing

- ❑ Add unit testing to our apache cookbook
- ❑ Create a cookbook to remediate the failing ssh controls; include unit tests