Building Security Into Your Workflow with InSpec

Mandi Walls | mandi@chef.io

HI!

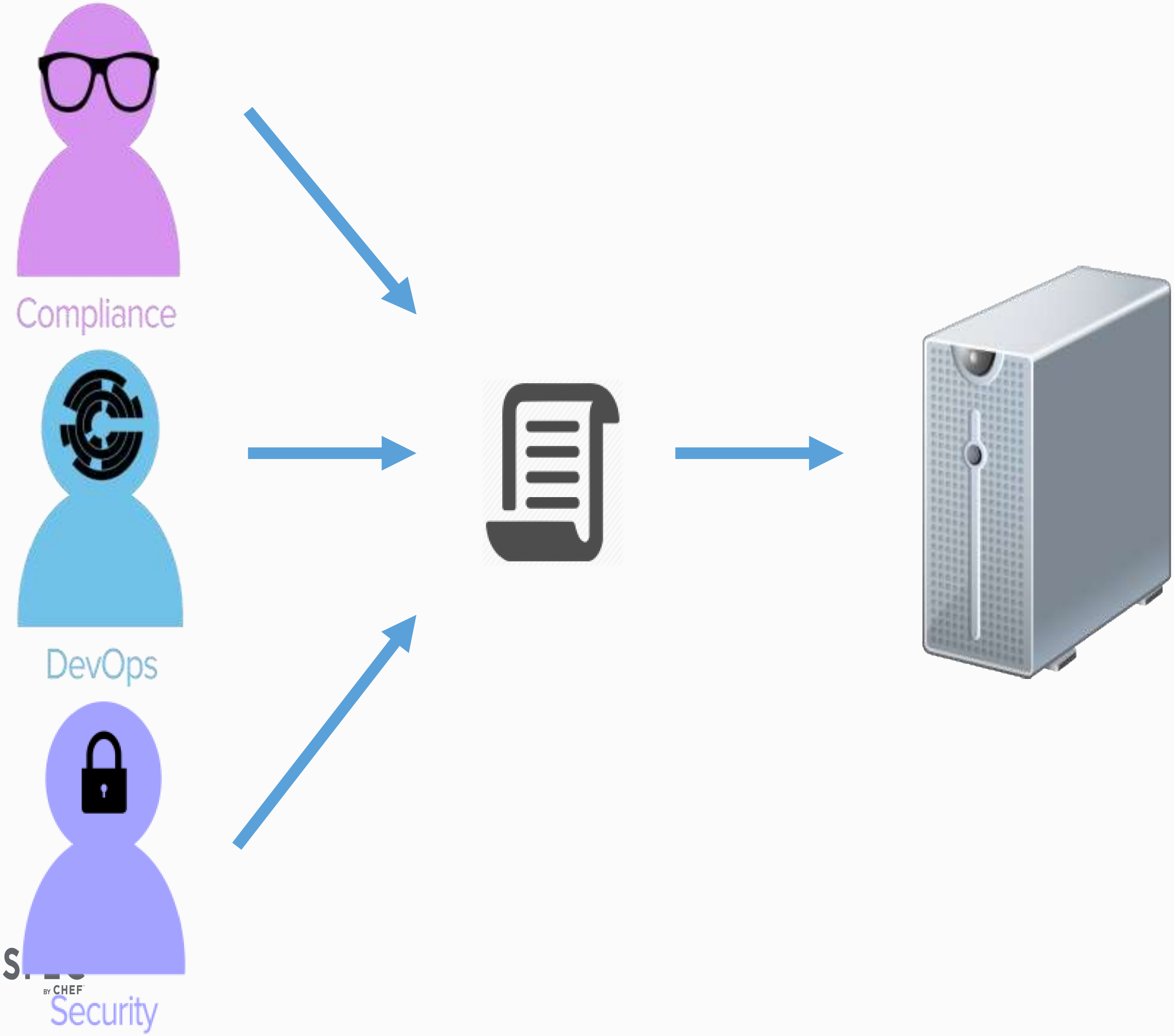
- Mandi Walls
- Technical Community Manager for Chef, EMEA
- mandi@chef.io
- @lnxchk
- https://inspec.io
- https://dev-sec.io/

EVERY business is a software business

> We’re going to be a software company with airplanes.
>
> – CIO, Alaska Airlines

What We Have Here Is A Communications Problem

Compliance
Security
DevOps
X
SHELL
rb


Compliance
DevOps
Security

What Is InSpec

InSpec

- Human-readable specification language for tests related to security and compliance
- Includes facilities for creating, sharing, and reusing profiles
- Extensible language so you can build your own rules for your applications and systems
- Command-line tools for plugging into your existing workflows / build servers
- Integrates with Test Kitchen for fast-feedback local testing by developers

SSH Example

- If your security team sends you a directive:

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

How Do You Go About Fixing It?

- Identify the file and file location to check your systems
- Create a file with a new setting
- Push out changes
- What’s the plan for the currently used images?
  - Rebuild?
  - Remediate at instantiation?
- Did you test it?

Lifecycle

- When you get a mandate from security, how often is it checked?
- Single big scan, report mailed out with a “due date”?
- Yearly or twice-yearly massive scans with remediation projects?

Using InSpec

User: chef
Pass: dodams2018

Find It!

- http://inspec.io/
- Open Source!
- The “spec” is a hint
- It comes installed as part of the Chef Developer’s Kit, ChefDK, or on its own
- It’s on your host

```
which inspec
```

- https://downloads.chef.io/chefdk

```
curl -o chefdk.rpm https://packages.chef.io/files/stable/chefdk/2.3.4/el/7/chefdk-2.3.4-1.el7.x86_64.rpm
sudo rpm -Uhv chefdk.rpm
```

Check that sshd_config

```
describe sshd_config do
  impact 1.0

  title 'SSH Version 2'

  desc <<-EOF
    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.
  EOF

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

Resources

- InSpec includes built-in resources for common services, system files, and configurations

  See http://inspec.io/docs/reference/resources/ for the current list!

- Built-in resources work on several platforms of Linux. There are also Windows-specifics
- A resource has characteristics that can be verified for your requirements, and Matchers that work with those characteristics

- Resources take the “grep for x” out of the testing phase
- Parsers included in the InSpec software do the work for you
- It’s built off the premises of rSpec, and meant to be human readable

its.... should...

- it { should exist }
- it { should be_installed }
- it { should be_enabled }
- its('max_log_file') { should cmp 6 }
- its('exit_status') { should eq 0 }
- its('gid') { should eq 0 }

Run It

- InSpec is command line
  - Installs on your workstation as a ruby gem or as part of the ChefDK
- Can be run locally, test the machine it is executing on
- Or remotely
  - InSpec will log into the target and run the tests for you
- Also a REPL
  - https://www.inspec.io/docs/reference/shell/

Create a Basic Test – test.rb

- Let’s write a basic test to make sure /tmp is a directory
- It also should be owned by root
- And its mode should be `01777` – open to all (plus sticky bit!)
- Let’s check out the docs for the “file” resource for InSpec

File Resources in InSpec

- https://www.inspec.io/docs/reference/resources/file/
- We want:

  Directory

  Owner

  Mode

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

test.rb

```
describe file("/tmp") do
  it { should exist }
  its('type') { should cmp 'directory' }
  its('owner') { should eq 'root' }
  its('mode') { should cmp '01777' }
end
```

Test Any Target

```
inspec exec test.rb

inspec exec test.rb -i ~/.aws/mandi_eu.pem -t
ssh://ec2-user@54.152.7.203

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

inspec exec test.rb -t docker://3dda08e75838
```

Execute InSpec

```
[chef@ip-172-31-38-151 ~]$ inspec exec ./test.rb
Profile: tests from ./test.rb
Version: (not specified)
Target:  local://
  File /tmp
     ✓  should exist
     ✓  should be directory
     ✓  should be owned by "root"
     ✓  mode should cmp == "01777"

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

Failures

- InSpec runs with failed tests return a non-zero return code

```
Profile Summary: 0 successful, 1 failures, 0 skipped
[chef@ip-172-31-29-25 ~]$ echo $?
1
[chef@ip-172-31-29-25 ~]$
```

- Passing tests have 0 return code

```
Profile Summary: 1 successful, 0 failures, 0 skipped
[chef@ip-172-31-29-25 ~]$ echo $?
0
[chef@ip-172-31-29-25 ~]$
```

Profiles

- InSpec profiles allow you to package and share sets of InSpec tests for your organization or for a specific application set
- Each profile can have multiple test files included
- The test files generally test for one required outcome, but can look at different objects to meet requirements
- Flexible!

  Create your own profiles for specific software you use

Hardening with InSpec

- Centos 7 host
- *os-hardening* cookbook from https://supermarket.chef.io
- */dev-sec/linux-baseline* InSpec profile from https://github.com/dev-sec/linux-baseline

What’s in the *linux-baseline* Profile

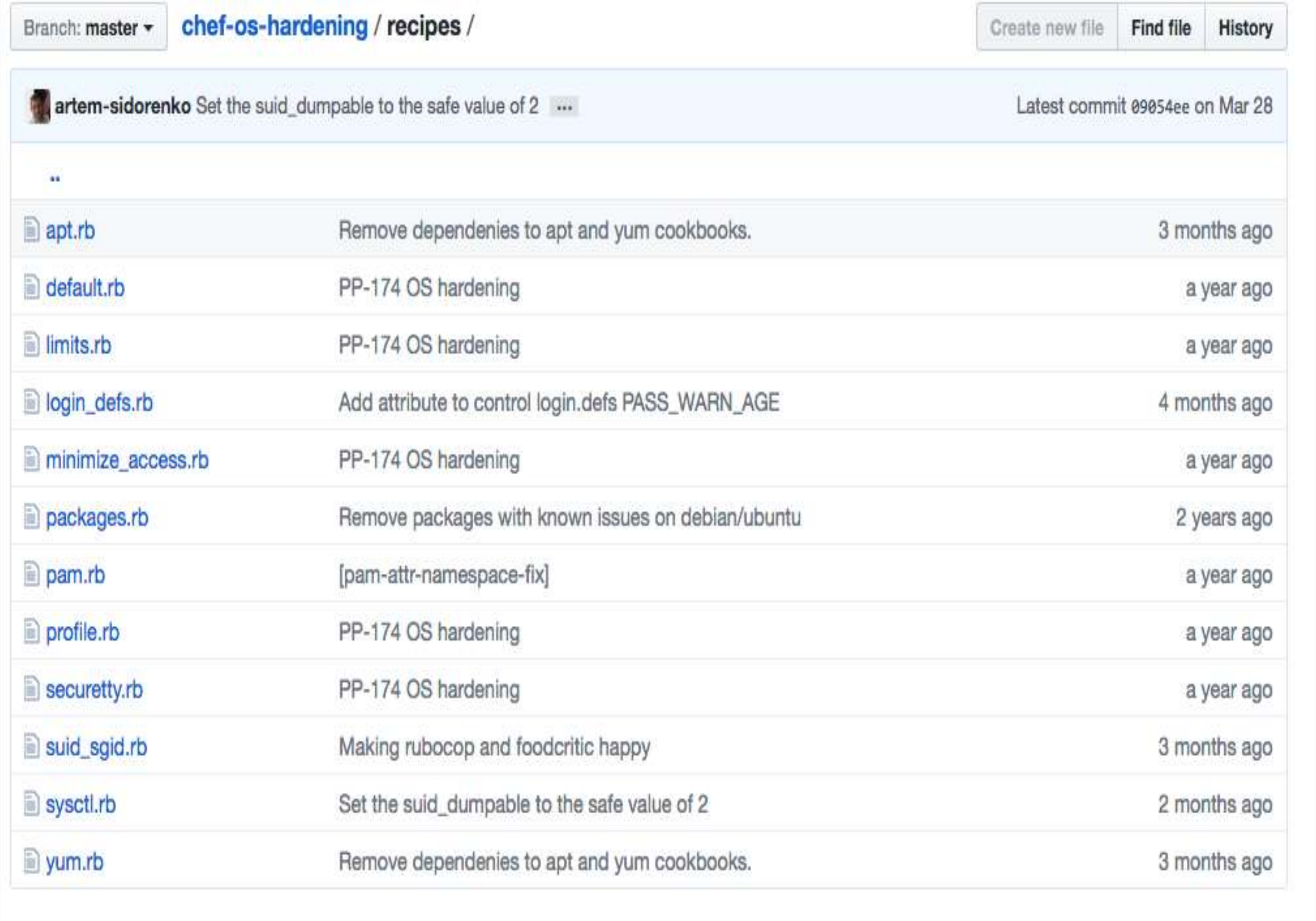
```
control 'os-02' do
  impact 1.0
  title 'Check owner and permissions for /etc/shadow'
  desc 'Check periodically the owner and permissions for /etc/shadow'
  describe file('/etc/shadow') do
    it { should exist }
    it { should be_file }
    it { should be_owned_by 'root' }
    its('group') { should eq shadow_group }
    it { should_not be_executable }
    it { should be_writable.by('owner') }
...
```

Use the Profile

```
$ git clone https://github.com/dev-sec/linux-baseline
...

$ sudo inspec exec linux-baseline
Profile Summary: 26 successful controls, 27 control
failures, 1 control skipped
Test Summary: 80 successful, 45 failures, 1 skipped
$
```

What’s in the os-hardening Cookbook



chef-os-hardening / recipes /

artem-sidorenko Set the suid_dumpable to the safe value of 2 ... Latest commit 09054ee on Mar 28

..		
apt.rb	Remove dependenies to apt and yum cookbooks.	3 months ago
default.rb	PP-174 OS hardening	a year ago
limits.rb	PP-174 OS hardening	a year ago
login_defs.rb	Add attribute to control login.defs PASS_WARN_AGE	4 months ago
minimize_access.rb	PP-174 OS hardening	a year ago
packages.rb	Remove packages with known issues on debian/ubuntu	2 years ago
pam.rb	[pam-attr-namespace-fix]	a year ago
profile.rb	PP-174 OS hardening	a year ago
securetty.rb	PP-174 OS hardening	a year ago
suid_sgid.rb	Making rubocop and foodcritic happy	3 months ago
sysctl.rb	Set the suid_dumpable to the safe value of 2	2 months ago
yum.rb	Remove dependenies to apt and yum cookbooks.	3 months ago

Use Chef to Repair the Findings

```
$ chef generate cookbook harden
(ignore git's complaints, it's ok)
```

Edit harden/metadata.rb

```
name 'harden'
maintainer 'The Authors'
maintainer_email 'you@example.com'
license 'All Rights Reserved'
description 'Installs/Configures harden'
...
...
depends 'os-hardening'
```

Create a Cookbooks Package

```
$ cd harden
$ berks install
$ berks package
$ cd ..
$ tar –xzvf harden/cookbooks-VERSION.tar.gz
```

Run chef-client to remediate failed tests

```
$ sudo chef-client -r "recipe[os-hardening]" --local-mode
```

Rerun the Tests

```
$ sudo inspec exec linux-baseline/
...
Profile Summary: 51 successful controls, 2 control failures, 1 control skipped
Test Summary: 123 successful, 2 failures, 1 skipped
```

What’s Still Failing?

- Find the controls that aren’t passing
- Decide if you want to fix them or forget them
- Let’s fix one and forget the others

Error 1: Entropy, os-08

```
control 'os-08' do
  impact 1.0
  title 'Entropy'
  desc 'Check system has enough entropy - greater than 1000'
  describe file('/proc/sys/kernel/random/entropy_avail').content.to_i do
    it { should >= 1000 }
  end
end
```

https://github.com/dev-sec/linux-baseline/blob/master/controls/os_spec.rb

Fix it with `rngd`

```
$ vi harden/recipes/default.rb

package 'rng-tools'

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

Install the Package

Turn on the Service

Berks Update

```
$ cd ~/harden
$ berks package
```

Install new cookbooks and run chef-client

```
$ cd ~
$ tar -xzvf harden/cookbooks-NEWVERSION.tar.gz
$ sudo chef-client -r “recipe[harden],recipe[os-
hardening]” --local-mode
…
Recipe: harden::default
  * yum_package[rng-tools] action install
    - install version 0:5-13.el7.x86_64 of package
rng-tools
  * service[rngd] action start
    - start service service[rngd]
  * service[rngd] action enable (up to date)
```

Check the InSpec Output Now

```
$ sudo inspec exec linux-baseline
...
Profile Summary: 52 successful controls, 1 control failure,
1 control skipped
Test Summary: 124 successful, 1 failure, 1 skipped
$
```

Error 2: auditd log setting `package-08`

```
control 'package-08' do
  impact 1.0
  title 'Install auditd'
  desc 'auditd provides extended logging capacities on recent
distribution'
...
  describe auditd_conf do
...
  its('max_log_file_action') { should cmp 'keep_logs' }
...
  end
end
```

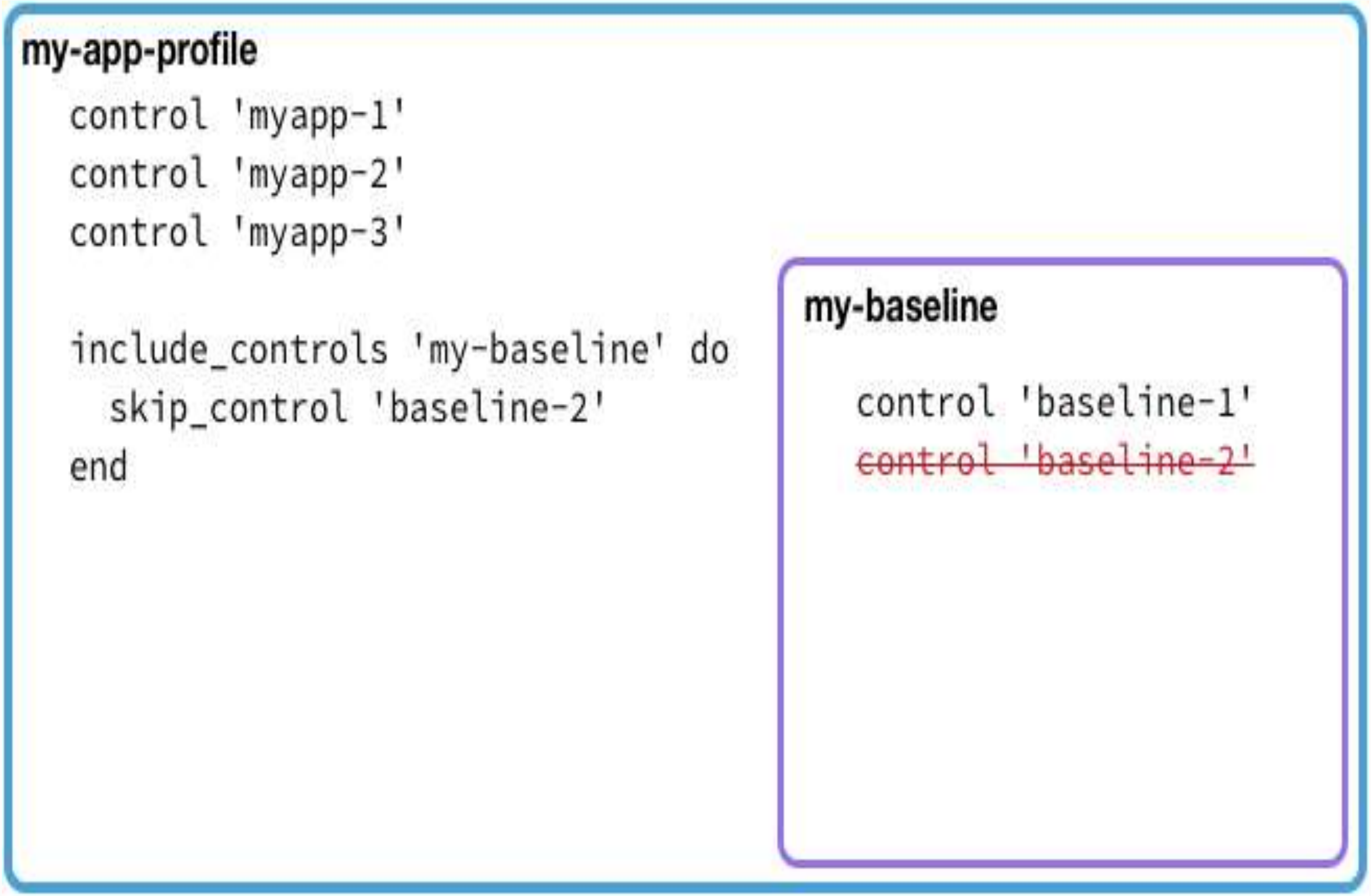
Maybe We're Ok with the Current Setting

- Large InSpec profiles contain lots of rules
- You may not want or need all of them for your infrastructure
- You can pick and choose which ones you want using your profile
- Let's ignore the auditd log file setting for now

Building New Profiles

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

Select the Controls You Want



my-app-profile

```
control 'myapp-1'
control 'myapp-2'
control 'myapp-3'

include_controls 'my-baseline' do
  skip_control 'baseline-2'
end
```

my-baseline

`control 'baseline-1'`

~~control 'baseline-2'~~

Including Profiles

```
$ vi my_hardening/inspec.yml
name: my_hardening
title: InSpec Profile
...
version: 0.1.0
depends:
  - name: linux-baseline
    git: https://github.com/dev-sec/linux-baseline
```

Skipping Individual Controls

```
$ rm -f my_hardening/controls/example.rb

$ vi my_hardening/controls/default.rb
include_controls 'linux-baseline' do
  skip_control ‘package-08'
end
```

Rerun the InSpec Profile

```
$ sudo inspec exec my_hardening/
...

Profile Summary: 52 successful controls, 0 control
failures, 1 control skipped
Test Summary: 113 successful, 0 failures, 1 skipped
```

Other Stuff – Test Kitchen

- InSpec also runs as a test suite in Test Kitchen
- Test Kitchen is a tool for your team to create fast-feedback loops for development
- Add InSpec tests to TK so that any change can also be certified with the security profile before it is pushed to source code repository
- More info at http://kitchen.ci/

Resources

- https://inspec.io
- https://dev-sec.io
- https://github.com/chef-training/workshops/
- http://www.anniehedgie.com/inspec-basics-1
- Windows and InSpec: http://datatomix.com/?p=236
- https://blog.chef.io/category/inspec/
- We're hiring! Work on InSpec in Belfast! https://chef.io/careers

CHEF™

INSPEC
BY CHEF