Add Continuous Compliance to Your CI/CD Pipelines

November 4, 2019

trility

Overview

Software development teams have long been able to take advantage of unit, integration, and functional testing as an integral part of a robust, test and behavior-driven development environment. Infrastructure as Code (IaC) provides new capabilities for DevOps teams to utilize new frameworks to build ephemeral environments with integrated compliance testing before, during, and after deployment.

Problem Statement:

I need to ensure compliance in my cloud environments.

Actively Monitoring Environments

Provider Solutions

- AWS Config

Cost structure, near real time, 100% customizable

- AWS Inspector

Access to EC2 instances, software vulnerabilities

Provider Solutions

- Azure Advisor

Cost, security, and configuration optimization

- AWS Trusted Advisor

External Monitoring

- CloudCheckr
- Prisma Cloud
- DivvyCloud
- Qualys
- Rapid7

Problem solved....ish

Monitoring Active Environments

Tools are validating the configuration of **active** environments.

- Vulnerabilities are found
- Tickets for the backlog
- Prioritized based on risk
- More often than not, left exposed for a period of time.

Refined Problem Statement:

I need to ensure compliance in my cloud environments **before** they are created.

Test-Driven Development (TDD)

- Add a test
- Run test to see the test fail
- Write code to satisfy the conditions of the test
- Run test to see the test pass
- Refactor
- Repeat as necessary

Test-Driven Infrastructure

If your infrastructure is in code, why is development of your IaC environments any different than the process developers use to write applications?

Infrastructure as Code Toolbox

Kitchen

https://kitchen.ci

Kitchen provides a test harness to execute infrastructure code on one or more platforms in isolation.

LocalStack

https://localstack.cloud

LocalStack provides a test framework on your local machine providing the same functionality and APIs as the real AWS cloud environment.

Clever Thinking

- Start with the mentality tests are required.
- Utilize orchestration like Jenkins or CircleCI to build private, ephemeral environments to test and validate changes

trility

Amazon Machine Image Example

Let’s walk through an example

- AMI Build Process
- ChefSpec Unit Tests
- Kitchen for Integration and Functional Testing
- Jenkins Pipeline

Amazon Machine Image Example

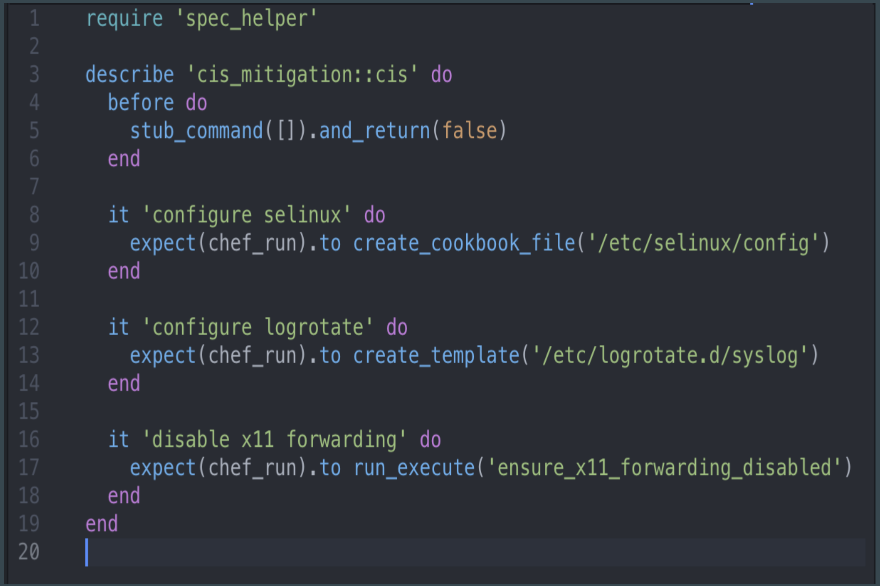
Components of the Example:

- Chef Cookbook - cis_mitigation
  - ChefSpec Unit Testing
  - Kitchen Configuration
- Packer Machine Definition
- Chef Cookbook - ami_builder
  - ChefSpec Unit Testing
  - Kitchen Configuration

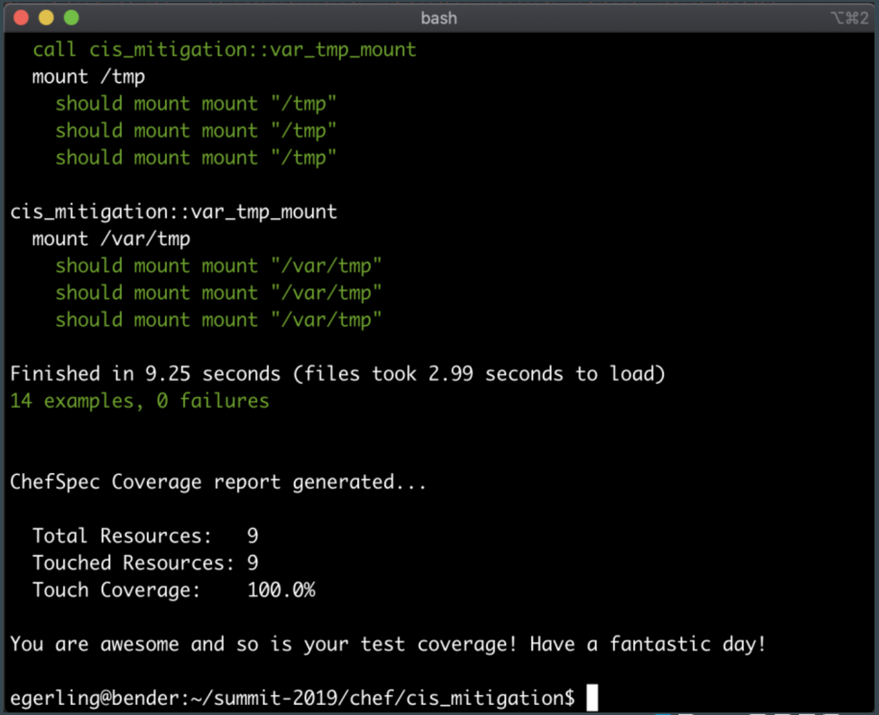
WAIT!! We don’t use Chef, we use <insert name of tool>!

ChefSpec Example - Chef Cookbook cis_mitigation

ChefSpec Tests



```ruby
require 'spec_helper'

describe 'cis_mitigation::cis' do
  before do
    stub_command([]).and_return(false)
  end

  it 'configure selinux' do
    expect(chef_run).to create_cookbook_file('/etc/selinux/config')
  end

  it 'configure logrotate' do
    expect(chef_run).to create_template('/etc/logrotate.d/syslog')
  end

  it 'disable x11 forwarding' do
    expect(chef_run).to run_execute('ensure_x11_forwarding_disabled')
  end
end
```



```
  call cis_mitigation::var_tmp_mount
  mount /tmp
    should mount mount "/tmp"
    should mount mount "/tmp"
    should mount mount "/tmp"

cis_mitigation::var_tmp_mount
  mount /var/tmp
    should mount mount "/var/tmp"
    should mount mount "/var/tmp"
    should mount mount "/var/tmp"

Finished in 9.25 seconds (files took 2.99 seconds to load)
14 examples, 0 failures


ChefSpec Coverage report generated...

  Total Resources:   9
  Touched Resources: 9
  Touch Coverage:    100.0%

You are awesome and so is your test coverage! Have a fantastic day!

egerling@bender:~/summit-2019/chef/cis_mitigation$
```

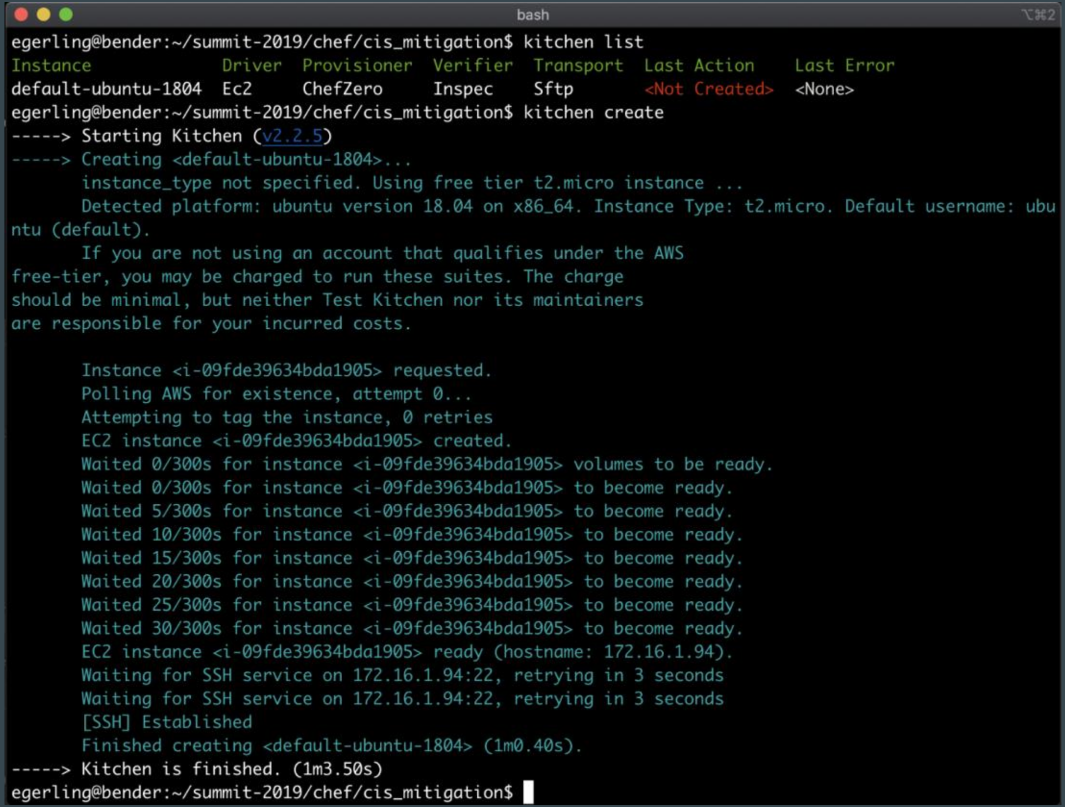
Kitchen Example - Chef Cookbook cis_mitigation

kitchen.yml

```
---
 driver:
   name: ec2
   aws_ssh_key_id: chef-test-kitchen-20191020
   security_group_ids: ["sg-asdf1234"]
   region: us-west-2
   subnet_id: subnet-1234asdf
   associate_public_ip: false
   interface: private
   tags:
     Name: test-kitchen-cis-mitigation
     Owner: "DevOps Group"
 provisioner:
   name: chef_zero
 transport:
   ssh_key: ~/.ssh/chef-test-kitchen-20191020
   connection_timeout: 10
   connection_retries: 5
   name: sftp
```

```
platforms:
  - name: ubuntu-18.04
    driver:
      transport:
        username: ubuntu
      image_search:
        owner-id: "099720109477"
        name: "*ubuntu-bionic-18.04-amd64-server*"
      block_device_mappings:
        - device_name: /dev/sda1
          ebs:
            volume_type: gp2
            volume_size: 20
            delete_on_termination: true
suites:
  - name: default
    provisioner:
      policyfile: policyfiles/kitchen.rb
verifier:
  name: inspec
```

Kitchen Example - Chef Cookbook cis_mitigation

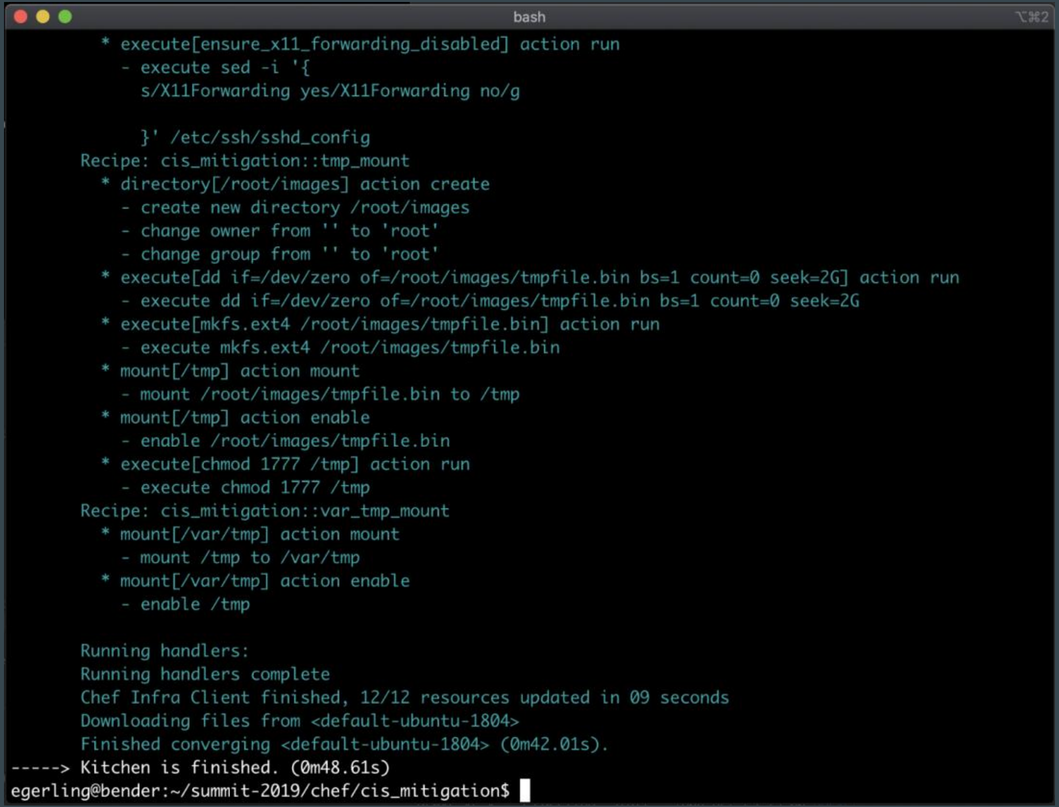


```
egerling@bender:~/summit-2019/chef/cis_mitigation$ kitchen list
Instance             Driver  Provisioner  Verifier  Transport  Last Action    Last Error
default-ubuntu-1804  Ec2     ChefZero     Inspec    Sftp       <Not Created>  <None>
egerling@bender:~/summit-2019/chef/cis_mitigation$ kitchen create
-----> Starting Kitchen (v2.2.5)
-----> Creating <default-ubuntu-1804>...
       instance_type not specified. Using free tier t2.micro instance ...
       Detected platform: ubuntu version 18.04 on x86_64. Instance Type: t2.micro. Default username: ubu
ntu (default).
       If you are not using an account that qualifies under the AWS
free-tier, you may be charged to run these suites. The charge
should be minimal, but neither Test Kitchen nor its maintainers
are responsible for your incurred costs.

       Instance <i-09fde39634bda1905> requested.
       Polling AWS for existence, attempt 0...
       Attempting to tag the instance, 0 retries
       EC2 instance <i-09fde39634bda1905> created.
       Waited 0/300s for instance <i-09fde39634bda1905> volumes to be ready.
       Waited 0/300s for instance <i-09fde39634bda1905> to become ready.
       Waited 5/300s for instance <i-09fde39634bda1905> to become ready.
       Waited 10/300s for instance <i-09fde39634bda1905> to become ready.
       Waited 15/300s for instance <i-09fde39634bda1905> to become ready.
       Waited 20/300s for instance <i-09fde39634bda1905> to become ready.
       Waited 25/300s for instance <i-09fde39634bda1905> to become ready.
       Waited 30/300s for instance <i-09fde39634bda1905> to become ready.
       EC2 instance <i-09fde39634bda1905> ready (hostname: 172.16.1.94).
       Waiting for SSH service on 172.16.1.94:22, retrying in 3 seconds
       Waiting for SSH service on 172.16.1.94:22, retrying in 3 seconds
       [SSH] Established
       Finished creating <default-ubuntu-1804> (1m0.40s).
-----> Kitchen is finished. (1m3.50s)
egerling@bender:~/summit-2019/chef/cis_mitigation$
```

Kitchen Commands

- kitchen list
  - Lists all of the test suites available for each platform.
- kitchen create
  - Create the test instance using the kitchen driver

Kitchen Example - Chef Cookbook cis_mitigation



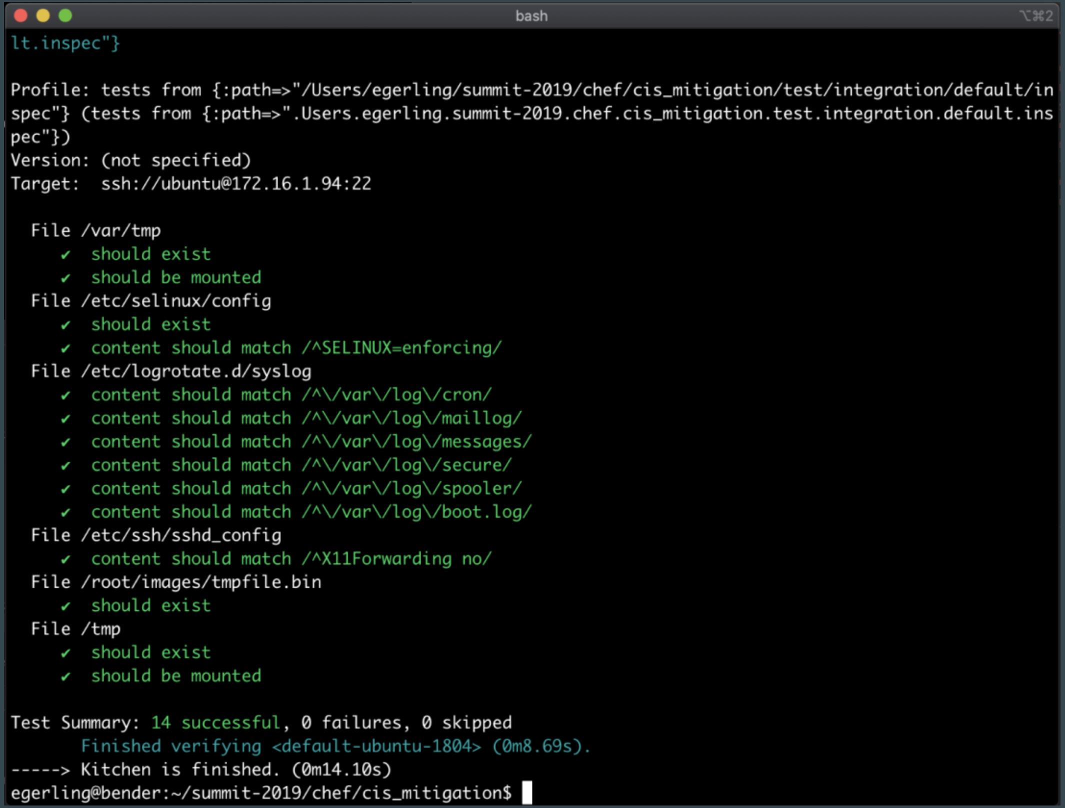
bash

```
          * execute[ensure_x11_forwarding_disabled] action run
            - execute sed -i '{
              s/X11Forwarding yes/X11Forwarding no/g

              }' /etc/ssh/sshd_config
        Recipe: cis_mitigation::tmp_mount
          * directory[/root/images] action create
            - create new directory /root/images
            - change owner from '' to 'root'
            - change group from '' to 'root'
          * execute[dd if=/dev/zero of=/root/images/tmpfile.bin bs=1 count=0 seek=2G] action run
            - execute dd if=/dev/zero of=/root/images/tmpfile.bin bs=1 count=0 seek=2G
          * execute[mkfs.ext4 /root/images/tmpfile.bin] action run
            - execute mkfs.ext4 /root/images/tmpfile.bin
          * mount[/tmp] action mount
            - mount /root/images/tmpfile.bin to /tmp
          * mount[/tmp] action enable
            - enable /root/images/tmpfile.bin
          * execute[chmod 1777 /tmp] action run
            - execute chmod 1777 /tmp
        Recipe: cis_mitigation::var_tmp_mount
          * mount[/var/tmp] action mount
            - mount /tmp to /var/tmp
          * mount[/var/tmp] action enable
            - enable /tmp

        Running handlers:
        Running handlers complete
        Chef Infra Client finished, 12/12 resources updated in 09 seconds
        Downloading files from <default-ubuntu-1804>
        Finished converging <default-ubuntu-1804> (0m42.01s).
-----> Kitchen is finished. (0m48.61s)
egerling@bender:~/summit-2019/chef/cis_mitigation$ █
```

Kitchen Commands

- kitchen converge
    - Apply the Chef run list to the test instance

Kitchen Example - Chef Cookbook cis_mitigation

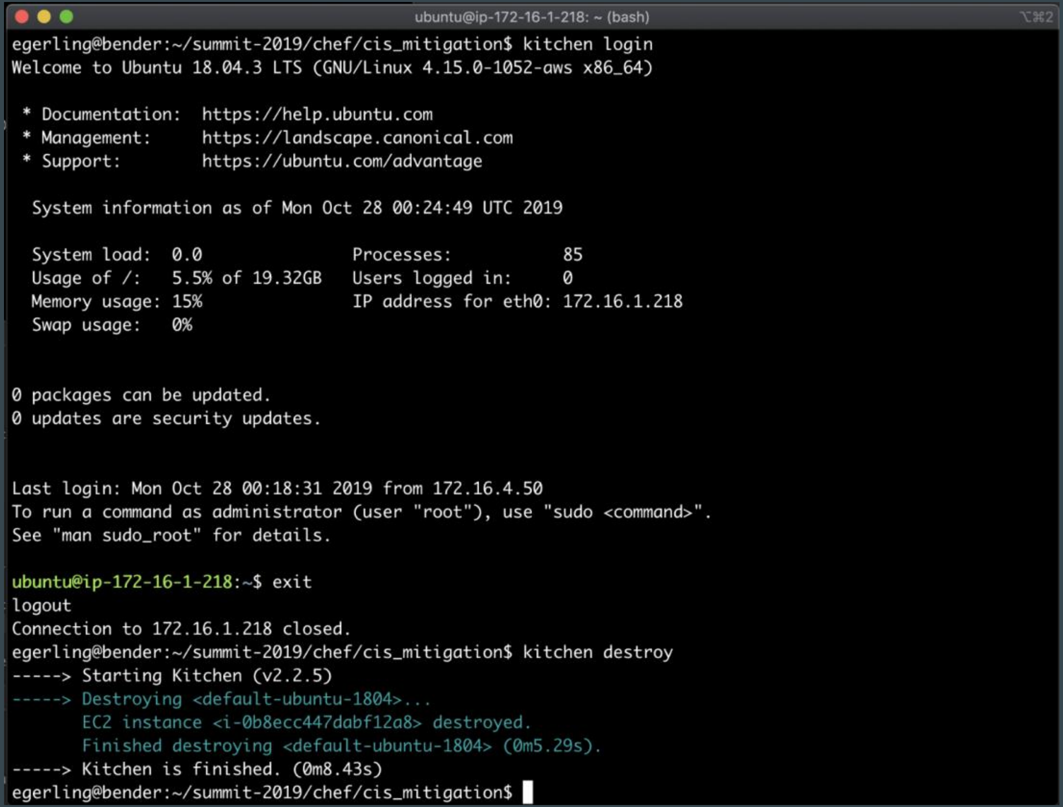


```
lt.inspec"}

Profile: tests from {:path=>"/Users/egerling/summit-2019/chef/cis_mitigation/test/integration/default/in
spec"} (tests from {:path=>".Users.egerling.summit-2019.chef.cis_mitigation.test.integration.default.ins
pec"})
Version: (not specified)
Target:  ssh://ubuntu@172.16.1.94:22

  File /var/tmp
     ✔  should exist
     ✔  should be mounted
  File /etc/selinux/config
     ✔  should exist
     ✔  content should match /^SELINUX=enforcing/
  File /etc/logrotate.d/syslog
     ✔  content should match /^\/var\/log\/cron/
     ✔  content should match /^\/var\/log\/maillog/
     ✔  content should match /^\/var\/log\/messages/
     ✔  content should match /^\/var\/log\/secure/
     ✔  content should match /^\/var\/log\/spooler/
     ✔  content should match /^\/var\/log\/boot.log/
  File /etc/ssh/sshd_config
     ✔  content should match /^X11Forwarding no/
  File /root/images/tmpfile.bin
     ✔  should exist
  File /tmp
     ✔  should exist
     ✔  should be mounted

Test Summary: 14 successful, 0 failures, 0 skipped
       Finished verifying <default-ubuntu-1804> (0m8.69s).
-----> Kitchen is finished. (0m14.10s)
egerling@bender:~/summit-2019/chef/cis_mitigation$
```

Kitchen Commands

- kitchen verify
  - Run the Inspec tests on the test instance

Kitchen Example - Chef Cookbook cis_mitigation



```
egerling@bender:~/summit-2019/chef/cis_mitigation$ kitchen login
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-1052-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Mon Oct 28 00:24:49 UTC 2019

  System load:  0.0               Processes:           85
  Usage of /:   5.5% of 19.32GB   Users logged in:     0
  Memory usage: 15%               IP address for eth0: 172.16.1.218
  Swap usage:   0%


0 packages can be updated.
0 updates are security updates.


Last login: Mon Oct 28 00:18:31 2019 from 172.16.4.50
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-16-1-218:~$ exit
logout
Connection to 172.16.1.218 closed.
egerling@bender:~/summit-2019/chef/cis_mitigation$ kitchen destroy
-----> Starting Kitchen (v2.2.5)
-----> Destroying <default-ubuntu-1804>...
       EC2 instance <i-0b8ecc447dabf12a8> destroyed.
       Finished destroying <default-ubuntu-1804> (0m5.29s).
-----> Kitchen is finished. (0m8.43s)
egerling@bender:~/summit-2019/chef/cis_mitigation$
```

Kitchen Commands

- kitchen login
  - Connect to the test instance and allow manual inspection of the instance
- kitchen destroy
  - Destroy the test instance

Kitchen Example - Inspec Tests

Example Inspec Test

```
# CIS 1.6.1 Configure SELinux
describe file('/etc/selinux/config') do
  it { should exist }
  its('content') { should match(/^SELINUX=enforcing/) }
end


# CIS 4.3 Ensure logrotate is configured
describe file('/etc/logrotate.d/syslog') do
  its('content') { should match %r{^\/var\/log\/cron} }
end
```

Example Inspec Test

```
# Test for Amazon SSM Agent installed and active
describe package('amazon-ssm-agent') do
  it { should be_installed }
end

if os[:family] == 'debian' || (os[:family] == 'redhat' &&
os[:release].to_i == 7)
  describe systemd_service('amazon-ssm-agent') do
    it { should be_installed }
    it { should be_enabled }
  end
end
```

Kitchen Example - Inspec Tests

Example Inspec Test

```
# CIS 1.1.9 Ensure noexec option set on /var/tmp partition
if os[:family] == 'ubuntu'
  describe command('/tmp/testfile.sh') do
    its('exit_status') { should eq 3 }
  end
end

if os[:family] == 'redhat'
  describe command('/tmp/testfile.sh') do
    its('exit_status') { should eq 1 }
  end
end
```

Example Inspec Test

```
describe command('/usr/local/bin/aws --version') do
  its('exit_status') { should eq 0 }
end

if os[:family] == 'redhat' && os[:release].to_i == 6
  describe upstart_service('amazon-ssm-agent') do
    it { should be_installed }
    it { should be_enabled }
    it { should be_running }
  end
end
```

Kitchen Building Blocks

Now we have the general building blocks, how can we use them?

- Integration Tests
- Functional Tests

Kitchen Building Blocks

Start with the Chef Cookbooks

- ChefSpec Unit Tests
- Kitchen Integration Tests

Kitchen Building Blocks

AMI Builder

- Packer Machine Definition
- AMI Builder Cookbook
  - ChefSpec Unit Tests
  - Kitchen Integration Tests
- AMI Functional Tests

Wrap It Up With Jenkins

Using a Jenkins Pipeline script, automate the entire process checking for failure along the way.

Jenkins Pipeline Example

```
stage("Lint") {
  sh "foodcritic -f correctness ami/cookbooks/ami-builder"
  dir ('ami/cookbooks/ami-builder') {
    sh "rubocop -r cookstyle ami/cookbooks/ami-builder"
  }
}
stage("Spec") {
  dir ('ami/cookbooks/ami-builder') {
    sh "chef exec rspec"
  }
}
stage("Integration Kitchen-Create") {
  dir ('ami/cookbooks/ami-builder') {
    sh "kitchen create integration-pre-packer-ubuntu-1804"
  }
}
```

```
stage("Integration Kitchen-Converge") {
  dir ('ami/cookbooks/ami-builder') {
    sh "kitchen converge int-pre-packer-ubuntu-1804"
  }
}
stage("Integration Kitchen-Verify") {
  dir ('ami/cookbooks/ami-builder') {
    sh "kitchen verify integration-pre-packer-ubuntu-1804"
  }
}
stage("Integration Kitchen-Destroy") {
  dir ('ami/cookbooks/ami-builder') {
    sh "kitchen destroy integration-pre-packer-ubuntu-1804"
  }
}
```

```
stage("Berks Vendor Cookbooks") {
  dir ('ami') {
    sh "rm -rf berks_vendor/"
```

Jenkins Pipeline Example

```
stage("Berks Vendor Cookbooks") {
  dir ('ami') {
    sh "rm -rf berks_vendor/"
    sh "mkdir -p berks_vendor"
    sh """
      for DIR in `ls cookbooks/`; do
        berks vendor berks_vendor -b \
          cookbooks/\$DIR/Berksfile
      done
    """

  }
}
        stage("Functional Kitchen-Create") {
          dir ('ami/cookbooks/ami-builder') {
            sh "sed -i 's/===ubuntu-1804ImageId===/${amiId}/g'
.kitchen.yml"
            sh "kitchen create functional-post-packer-ubuntu-
1804"
          }
```

```
def amiId = ""
stage("Run Packer") {
  dir ('ami') {
    sh """
    { packer build -machine-readable -var
branch=${env.BRANCH} base_ami.json; echo "\$?"; } | tee
packer_build_output.txt
    tail -n 1 output.txt > exit_code.txt
    grep "artifact,0,id" output.txt | cut -d":" -f 2 > ami_id.txt
    """

    packerExitCode = readFile "exit_code.txt"
    if (packerExitCode.trim() != "0") {
      error("Packer failed")
    }
  }
  amiId = readFile "ami_id.txt"
}
```

Jenkins Pipeline Example

```
stage("Functional Kitchen-Create") {
  dir ('ami/cookbooks/ami-builder') {
    sh "sed -i 's/===ImageId===/${amiId}/g' .kitchen.yml"
    sh "kitchen create functional-post-packer-ubuntu-1804"
  }
}
stage("Functional Kitchen-Verify") {
  dir ('ami/cookbooks/ami-builder') {
    sh "kitchen verify functional-post-packer-ubuntu-1804"
  }
}
stage("Functional Kitchen-Destroy") {
  dir ('ami/cookbooks/ami-builder') {
    sh "kitchen destroy functional-post-packer-ubuntu-1804"
  }
}
```

Additional Steps

We have a Jenkins Pipeline to build and test an AMI. What's next?

- Rapid7
- Qualys
- Encryption
- Sharing With Accounts

Thank you!

Eric Gerling, CTO, Trility Consulting
eric@trility.io

https://github.com/Trility/cloud-devops-security-summit-2019