Running with Scissors

Ruby Service

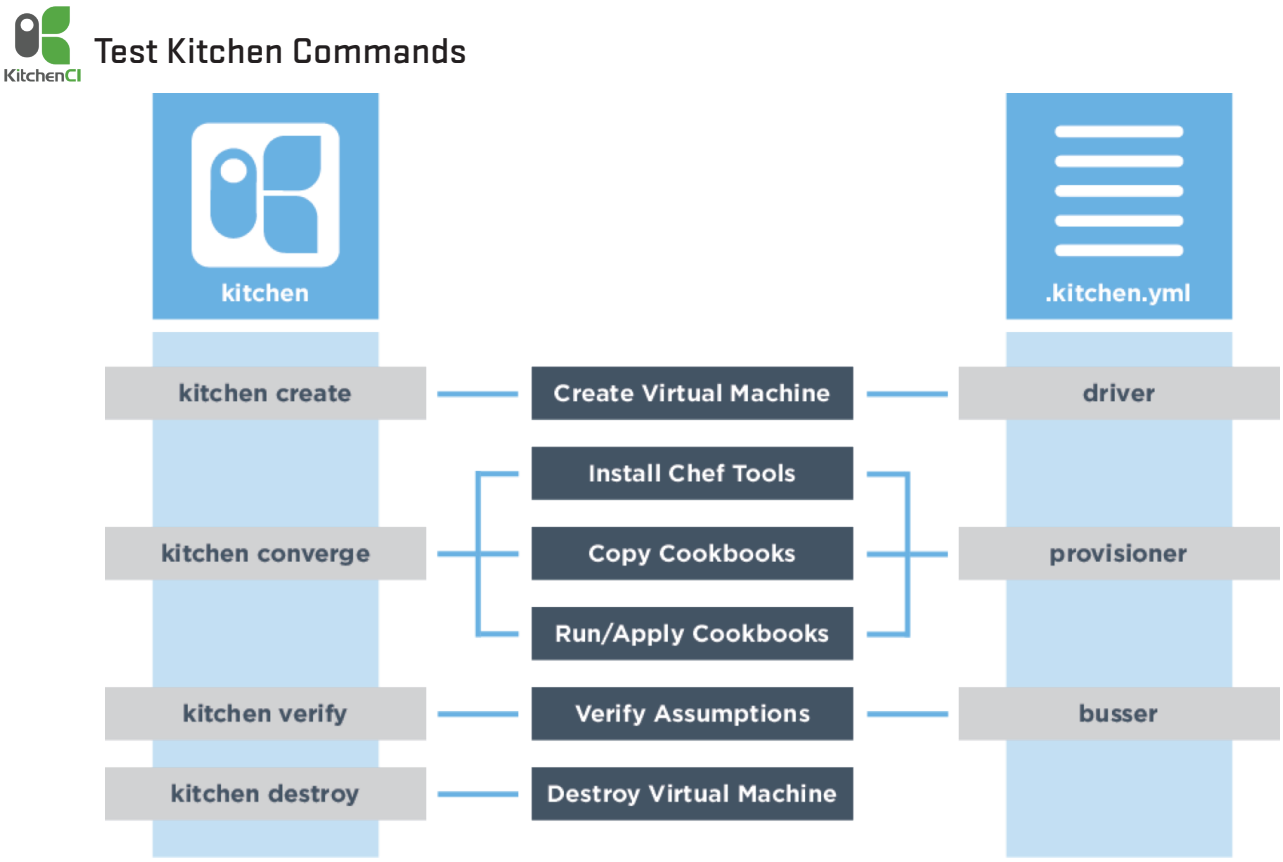
~/cookbooks/ruby_service/recipes/default.rb

```
     0: #
     1: # Cookbook:: ruby_service
     2: # Recipe:: default
     3: #
     4: # Copyright:: 2018, The Authors, All Rights Reserved.
     5:
     6: directory '/srv'
     7:
     8: package 'unzip'
     9:
    10: cookbook_file '/srv/specter.zip' do
    11:   source 'specter.zip'
    12:   notifies :run, 'execute[extract_site]', :immediately
    13: end
    14:
    15: execute 'extract_site' do
    16:   cwd '/srv'
    17:   command 'unzip specter.zip'
    18:   not_if { File.exist?('/srv/specter') }
    19:   action :nothing
    20: end
    21:
    22: package %w[git-core zlib zlib-devel gcc-c++ patch ⬀
readline readline-devel libyaml-devel libffi-devel ⬀
openssl-devel make bzip2 autoconf automake libtool bison curl ⬀
sqlite-devel]
    23:
    24: remote_file '/tmp/ruby-2.5.1.tar.gz' do
    25:   source ⬀
'https://cache.ruby-lang.org/pub/ruby/2.5/ruby-2.5.1.tar.gz'
    26: end
    27:
    28: execute 'extract ruby' do
    29:   cwd '/tmp'
    30:   command 'tar -xvf ruby-2.5.1.tar.gz'
    31:   not_if { File.exist?('/tmp/ruby-2.5.1') }
    32: end
    33:
    34: execute 'configure, make and install ruby' do
    35:   cwd '/tmp/ruby-2.5.1'
    36:   command './configure && make && make install'
    37:   not_if { File.exist?('/usr/local/bin/ruby') }
    38: end
    39:
    40: execute 'install bundler' do
    41:   command '/usr/local/bin/gem install bundler'
    42:   not_if { File.exist?('/usr/local/bin/bundle') }
    43: end
    44:
    45: execute 'bundle install' do
    46:   cwd '/srv/specter'
```

~/cookbooks/ruby_service/recipes/default.rb

```
    47:   command '/usr/local/bin/bundle install'
    48:   not_if { File.exist?('/srv/specter/Gemfile.lock') }
    49: end
    50:
    51: execute 'database migration' do
    52:   cwd '/srv/specter'
    53:   command 'rake migrate'
    54: end
    55:
    56: template '/etc/systemd/system/specter.service' do
    57:   source 'service.erb'
    58:   variables({ service_name: 'specter', ⬀
working_directory: '/srv/specter' })
    59: end
    60:
    61: template '/srv/specter/start.sh' do
    62:   source 'start.sh.erb'
    63:   mode '0774'
    64: end
    65:
    66: template '/srv/specter/stop.sh' do
    67:   source 'stop.sh.erb'
    68:   mode '0774'
    69: end
    70:
    71: service 'specter' do
    72:   action :start
    73: end
```

Running with Scissors



Within the directory of the cookbook:

```
> kitchen list
```

Display the instances (platforms * suites)

```
> kitchen converge [instance|regex]
```

Create the instance, install Chef, and apply the recipe.

```
> kitchen verify [instance|regex]
```

Verify the instance with the tests defined for the suite.

```
> kitchen destroy [instance|regex]
```

Destroy the instance.

```
> kitchen login [instance|regex]
```

Login to the instance.

Running with Scissors

Cookbook Attributes

https://docs.chef.io/attributes.html

Create the attributes directory and attribute file

```
> chef generate attribute FILENAME
```

The attribute is defined at particular precedence level in the attributes file and then it is retrieved from the node object.

~/cookbooks/ruby_service/attributes/default.rb

```
default['ruby_service']['site_path'] = '/srv/specter'
```

The attribute is defined on the node object and may be retrieved in the recipe.

~/cookbooks/ruby_service/recipes/default.rb

```
site_path = node['ruby_service']['site_path']

execute 'extract_site' do
  cwd '/srv'
  command 'unzip specter.zip'
  not_if { File.exist?(site_path) }
  action :nothing
end
```

Cookbook Recipe and include_recipe

Create the recipe with test files

https://docs.chef.io/dsl_recipe.html#include-recipes

```
> chef generate recipe FILENAME
```

A recipe can include one (or more) recipes located in cookbooks by using the include_recipe method.

~/cookbooks/ruby_service/recipes/default.rb

```
include_recipe 'ruby_service::install'
include_recipe 'ruby_service::configuration'
include_recipe 'ruby_service::service'
```

Running with Scissors

Web Service Implementation

Create new scans	POST	http://localhost:8000/scans
Read all scans or a scan	GET	http://localhost:8000/scans
Update any existing scans	PUT	http://localhost:8000/scan/{id}
Delete any existing scans	DELETE	http://localhost:8000/scan/{id}

INSPEC Resources

https://www.inspec.io/docs/reference/resources/

file resource

```
describe file('/usr/local/bin/ruby') do
  it { should exist }
  it { should be_executable }
end
```

```
describe file('/srv/specter') do
  it { should be_directory }
  its('owner') { should eq 'root' }
end
```

command resource

```
describe command('/usr/local/bin/ruby') do
  it { should exist }
end

describe command('/usr/local/bin/ruby -v') do
  its('stdout') { should match('2.5.1') }
end
```

```
describe command('curl http://localhost:8000') do
  its('stdout') { should match('welcome friend') }
end
```

host resource

```
describe host('localhost', port: 8000, protocol: 'tcp') do
  it { should be_reachable }
  it { should be_resolvable }
end
```

port resource

```
describe port(8000) do
  it { should be_listening }
  its('processes') { should include 'ruby'}
end
```

http resource

```
describe http('http://localhost:8000', enable_remote_worker: true) do
  its('status') { should eq 200 }
end
```

service resource

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

json resource

```
describe host('localhost', port: 8000, protocol: 'tcp') do
  it { should be_reachable }
  it { should be_resolvable }
end
```

Node Service

~/cookbooks/node_service/recipes/default.rb

```
 0: #
 1: # Cookbook:: node_service
 2: # Recipe:: default
 3: #
 4: # Copyright:: 2018, The Authors, All Rights Reserved.
 5:
 6: directory '/srv'
 7:
 8: package 'unzip'
 9:
10: cookbook_file '/srv/lich.zip' do
11:   source 'lich.zip'
12:   notifies :run, 'execute[extract_site]', :immediately
13: end
14:
15: execute 'extract_site' do
16:   cwd '/srv'
17:   command 'unzip lich.zip'
18:   not_if { File.exist?('/srv/lich') }
19:   action :nothing
20: end
21:
22: package %w[gcc gcc-c++ sqlite-devel]
23:
24: remote_file '/tmp/node-v10.1.0-linux-x64.tar.gz' do
25:   source ⬀
'http://nodejs.org/dist/v10.1.0/node-v10.1.0-linux-x64.tar.gz'
26: end
27:
28: execute 'extract node' do
29:   cwd '/tmp'
30:   command 'tar --strip-components 1 -xzvf ⬀
node-v10.1.0-linux-x64.tar.gz -C /usr/local'
31:   not_if { File.exist?('/usr/local/bin/node') }
32: end
33:
34: execute 'install dependencies' do
35:   cwd '/srv/lich'
36:   command 'npm install'
37: end
38:
39: execute 'migrate databate' do
40:   cwd '/srv/lich'
41:   command 'bin/migrate'
42: end
43:
44: template '/etc/systemd/system/lich.service' do
45:   source 'service.erb'
46:   variables({ service_name: 'lich', working_directory: ⬀
'/srv/lich' })
47: end
```

~/cookbooks/node_service/recipes/default.rb

```
48:
49: template '/srv/lich/start.sh' do
50:   source 'start.sh.erb'
51:   mode '0774'
52: end
53:
54: template '/srv/lich/stop.sh' do
55:   source 'stop.sh.erb'
56:   mode '0774'
57: end
58:
59: service 'lich' do
60:   action :start
61: end
```

Running with Scissors

INSPEC Commands

https://www.inspec.io/docs/reference/cli/

InSpec controls can be defined in their own artifact. This enables them to be reused across cookbook test suites and used to scan your target infrastructure.

```
> inspec init profile PROFILENAME
```

Inspec can execute scans against remote targets with profiles.

```
> inspec exec PROFILENAME -t PROTOCOL://USER:PASSWORD@HOST:PORT
```

Inspec profiles can be archived for easy transport.

```
> inspec arhive PROFILENAME
```

INSPEC Controls

https://www.inspec.io/docs/reference/dsl_inspec/

```
control 'sshd-8' do
  impact 0.6
  title 'Server: Configure the service port'
  desc '
    Always specify which port the SSH server should listen to.
    Prevent unexpected settings.
    '
  tag 'ssh','sshd','openssh-server'
  tag cce: 'CCE-27072-8'
  ref 'NSA-RH6-STIG - Section 3.5.2.1', url:
'https://www.nsa.gov/ia/_files/os/redhat/rhel5-guide-i731.pdf'

  describe sshd_config do
    its('Port') { should eq('22') }
  end
end
```

where

- `'sshd-8'` is the name of the control
- `impact`, `title`, and `desc` define metadata that fully describes the importance of the control, its purpose, with a succinct and complete description
- `impact` is an float that measures the importance of the compliance results and must be a value between `0.0` and `1.0`. The value ranges are:
  - `0.0 to <0.4` these are controls with minor criticality
  - `0.4 to <0.7` these are controls with major criticality
  - `0.7 to 1.0` these are critical controls
- `tag` is optional meta-information with with key or key-value pairs
- `ref` is a reference to an external document
- `describe` is a block that contains at least one test. A `control` block must contain at least one `describe` block, but may contain as many as required
- `sshd_config` is an InSpec resource. For the full list of InSpec resources, see InSpec resource documentation
- `its('Port')` is the matcher; `{ should eq('22') }` is the test. A `describe` block must contain at least one matcher, but may contain as many as required

Rust Service

~/cookbooks/rust_service/recipes/default.rb

```
     0: #
     1: # Cookbook:: rust_service
     2: # Recipe:: default
     3: #
     4: # Copyright:: 2018, The Authors, All Rights Reserved.
     5:
     6: directory '/srv'
     7:
     8: package 'unzip'
     9:
    10: cookbook_file '/srv/wraith.zip' do
    11:   source 'wraith.zip'
    12:   notifies :run, 'execute[extract_site]', :immediately
    13: end
    14:
    15: execute 'extract_site' do
    16:   cwd '/srv'
    17:   command 'unzip wraith.zip'
    18:   not_if { File.exist?('/srv/wraith') }
    19:   action :nothing
    20: end
    21:
    22: execute 'install rustup default nightly' do
    23:   command 'curl https://sh.rustup.rs -sSf | sh -s -- ⤤
--default-toolchain nightly -y'
    24:   not_if { File.exist?('~/.rustup') }
    25: end
    26:
    27: # execute 'install working rust nightly build' do
    28: #   cwd '/srv/wraith'
    29: #   command '/root/.cargo/bin/rustup install ⤤
nightly-2018-04-19'
    30: # end
    31:
    32: execute 'yum groupinstall "Development Tools" -y'
    33:
    34: # package [ 'sqlite-devel' , 'mariadb-devel', ⤤
'postgresql-devel' ]
    35: package [ 'sqlite-devel' ]
    36:
    37: execute 'install dependencies' do
    38:   cwd '/srv/wraith'
    39:   command '/root/.cargo/bin/cargo install --force'
    40: end
    41:
    42: execute 'install diesel-cli' do
    43:   cwd '/srv/wraith'
    44:   command '/root/.cargo/bin/cargo install diesel_cli ⤤
--version 1.2.0 --no-default-features --features "sqlite"'
    45:   not_if { File.exist?('/root/.cargo/bin/diesel') }
    46: end
```

~/cookbooks/rust_service/recipes/default.rb

```
    47:
    48: execute 'database setup' do
    49:   cwd '/srv/wraith'
    50:   command '/root/.cargo/bin/diesel setup && ⤤
/root/.cargo/bin/diesel migration run'
    51: end
    52:
    53: execute 'build' do
    54:   cwd '/srv/wraith'
    55:   command '/root/.cargo/bin/cargo build'
    56: end
    57:
    58: template '/etc/systemd/system/wraith.service' do
    59:   source 'service.erb'
    60: end
    61:
    62: template '/srv/wraith/start.sh' do
    63:   source 'start.sh.erb'
    64:   mode '0777'
    65: end
    66:
    67: template '/srv/wraith/stop.sh' do
    68:   source 'stop.sh.erb'
    69:   mode '0777'
    70: end
    71:
    72: service 'wraith' do
    73:   action :start
    74: end
```

Running with Scissors

shell_out

https://github.com/chef/mixlib-shellout

```
Chef.event_handler do
  on :run_completed do

    scan_path = "#{Chef::Config[:cookbook_path]}/audit/inspec-*.json"
    headers = '"Content-Type: application/json"'
    url = 'http://localhost:8000/scans'

    upload = Mixlib::ShellOut.new "curl -d @$(ls #{scan_path}) -H #{headers} #{url}"
    upload.run_command

  end
end
```

Ruby / Chef

```
Chef.event_handler do
  on :run_completed do

    scan_path = "#{Chef::Config[:cookbook_path]}/audit/inspec-*.json"
    connection = Chef::HTTP.new('http://localhost:8000')
    headers = { 'Content-Type' => 'application/json' }
    Dir[scan_path].each do |scan_file|
      data = File.read(scan_file)
      connection.post("/scans", data.to_s, headers)
    end

  end
end
```

Test Kitchen Configuration with a new Suite

```
suites:
  # existing default suite
  - name: audit
    run_list:
      - recipe[node_service::default]
      - recipe[azrael::default]
    verifier:
      inspec_tests:
        - test/integration/audit
    attributes:
```