Kokki: Configuration Management Framework

It means “cook” in Finnish

Samuel Stauffer

What is it?

- Infrastructure automation → configuration management
- Part of provisioning
- Library and simple command line tool
  - currently no client/server component
  - can use GitHub in place of client/server
- Alternative to Puppet/Chef/CFengine

Why not Puppet, Chef, or ...

- Python
- It's a framework/library rather than a complete solution
- Different needs

What can I use it for?

- Creating a template for configuring servers
  - Great for cloud based servers (EC2)
  - Mix in some ‘userscript’ and you have a new web instances in less than 5 minutes.
- Avoid having to log into servers except when absolutely necessary
  - Configuring servers by hand is tedious, not scalable, and more prone to mistakes. (fine for 1 server, not for 10)

Overview

- Environment: execution environment
  - Resource: describes a file, service, package, etc..
    - Provider: knows how to execute a resource
- Kitchen: container for cookbooks
  - Cookbook: container for recipes and libraries
    - Recipe: group of resource definitions
    - Library: utility methods, resources, and providers

Environment

- An environment holds and executes resources and actions.
- config: attribute dictionary of configuration values
- system: provides access to system information
    - machine architecture, os (linux, darwin, ...), flavor (redhat, ubuntu, debian, ...)

Resource

- A resource is a description of a small part of a configuration
- Examples: File, Mount, Package, Service, ...

```
File(“/etc/nginx/nginx.conf”,
     owner = “root”,
     mode = 0644,
     content = Template(“nginx/nginx.conf.j2”),
     notifies = [
         (“restart” env.resources[“Service”][“nginx”]),
     ])
```

Provider

- A provider is a concrete implementation of a resource.
- Responsible for checking current state of the resource and taking any steps necessary to make resource match. (e.g. compare the contents of a file and overwrite if it doesn't match)
- Each platform can have a provider (e.g. one provider knows how to install packages using 'apt' on Ubuntu)

Kitchen

- A Kitchen:
  - is a subclass of an Environment
  - provides a higher level of abstraction for describing roles and resources
  - contains cookbooks which contain recipes and libraries

Cookbook

- A cookbook groups recipes, libraries, and default configuration as a unit. For instance a package like “apache2” could have a cookbook.
- Each cookbook is required to contain a file metadata.py which holds a dictionary of all configuration values and their defaults.
- Subdirectories of a cookbook include: recipes, libraries, templates, and files.
- Multiple recipes can be used for different purposes: apache2.prefork, apache2.worker, etc..

Recipe

- A recipe is a "script" that defines resources.
- Each recipe receives a global 'env' which is the current Environment/Kitchen.
- A cookbook can have a 'default.py' recipe which is the one used if no recipe specified.
  - env.include_recipe("memcached")
    vs env.include_recipe("munin.node")

Demo

- http://github.com/samuel/kokki-pywebsf-demo

Current Limitations

- Currently only supports Debian/Ubuntu
  - easy to support other platforms, just isn’t done
- No client/server
- Small userbase
- No documentations

Roadmap

- Better documentation
- Client/server
- Cloud tools (automate EC2 instances)
- More platforms (though probably not Windows)
- Pick a different theme? (enough with the cooking/chefs)

Thanks! Questions?

- http://github.com/samuel/kokki
- http://github.com/samuel/kokki-pywebsf-demo
- https://convore.com/kokki/
- samuel@descolada.com