If you're involved in your organization's compliance or security process, you'll want to learn about InSpec.

**Habitat** is an open-source project by Chef that moves an *application's configuration, management, and behavior to the application itself, not the infrastructure that the application runs on.*

Check out these modules to learn how Habitat provides simple and consistent ways to package both legacy and modern applications. See how Habitat Builder enables you to automatically rebuild and repackage your apps and their dependencies when you push a change up to GitHub.

**Infrastructure as Code**

**Chef code is made up of resources** which each define a single configuration task that must be performed.

Resource Example

package 'nginx' do

action :install

end

**Chef resources** are declarative by design, which means you tell Chef what action to take, and Chef dynamically takes platform-appropriate action automatically. Chef resources are also designed to be run continuously. Chef’s test-and-repair behavior means that configurations are only applied to systems that have diverged from their desired state. On properly configured systems, Chef simply confirms that configurations are up to date. Otherwise, Chef converges each system with their desired state, allowing you to apply configuration updates continuously, with confidence that Chef will only take action when it needs to.

**Chef Cookbooks**

**Chef code is collected into purpose-specific artifacts called cookbooks**. Each **cookbook defines a configuration scenario and all the necessary components to support that scenario. Most cookbooks consist of:**

**Recipes:** Ordered lists of Chef resources that describe an end-to-end configuration task

**Attributes:** User-defined variables that can be used to alter the behavior of a Chef recipe without needing to alter its underlying code

**Files & Templates:** Any static files or templates required for configuration tasks

**metadata.rb:** A metadata file used to ensure that each cookbook is correctly deployed to each node, providing support for cookbook versioning and dependency management

Includes Everything in the **Chef Development Kit**

In addition to chef-run and InSpec, Chef Workstation includes all of the powerful cookbook creation, testing, and dependency management tools built into the **ChefDK. This includes**:

**chef-client:** an agent that configures your nodes

**Ohai:** a tool for collecting system profiling data

**chef and knife:** command-line utilities for generating Chef cookbooks and managing node configurations respectively

**Test Kitchen:** a testing harness for rapid validation of Chef code

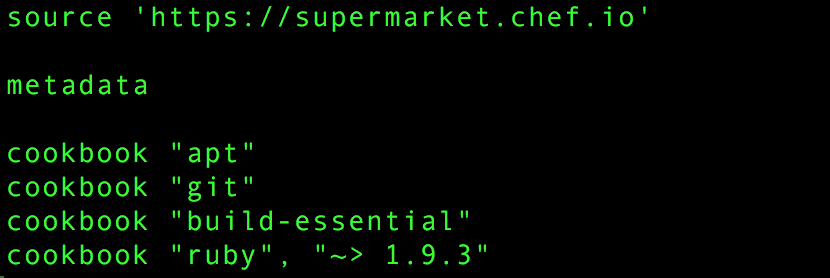
**InSpec:** Chef’s open source security & compliance automation framework

**Test Kitchen:** a testing harness for rapid validation of Chef code

**Berkshelf** is a dependency manager for Chef cookbooks. With it, you can easily depend on community cookbooks and have them safely included in your workflow. You can also ensure that your CI systems reproducibly select the same cookbook versions, and can upload and bundle cookbook dependencies without needing a locally maintained copy. Berkshelf is included in the Chef Development Kit.

**Berksfile**

**Berksfile** is the most crucial component of Berkshelf! It’s just like metadata for Chef. However, the usage is somewhat different. Berksfile makes it simple to download a dependency cookbook from chef supermarket (or other places) and upload to the cookbook repository on the server. Here is the usual content inside it:



**It involves mainly three settings:**

**source :** Place from where these cookbooks and dependencies will be fetched if those are not available locally with Berkshelf.

**metadata :** This points to Berkshelf itself saying that for every dependency just look at me and I will look into metadata.rb placed just along with me. It’s just like gemspec for Gemfile.

**cookbook :** List of all the cookbooks/dependencies required.

**A Policyfile** is an optional way to manage role, environment, and community cookbook data with a single document that is uploaded to the Chef server. The file is associated with a group of nodes, cookbooks, and settings. When these nodes perform a Chef client run, they utilize recipes specified in the Policyfile run-list.

**Consider the following before using Policyfiles:**

**Policyfile is not supported as part of a Chef Automate workflow**

**Policyfile is intended to be used with Chef server 12.3 or above, and Chef client 12.8 or above**