Course Code: INFO3606
Course Title: Cloud Computing
Semester Project

Tevin Achong - 816000026, Jonathan Owen -

April 8, 2020

Contents

1 Chef

Chef is a configuration management tool that is used to streamline the task of configuring and maintaining a company's servers. Aditionally, Chef can integrate with various cloud-based platforms to automatically provision and configure new machines. It contains solutions for small and large scale systems, and features and pricing for each of the various ranges. Essentially, Chef ensures that the files and the software that users are expecting to be on a machine are actually present, configured correctly, and working as is expected. Performing these tasks for a single machine is fairly straightforward. However, as an organization's infrastructure scales up (more machines are introduced) it becomes increasingly difficult. This is the reason Chef was developed and is used.

1.1 Components

Chef operates with three core components: **Chef Server**, **Workstations**, and **Nodes**. These three components communicate in a linear fashion, as explained below.

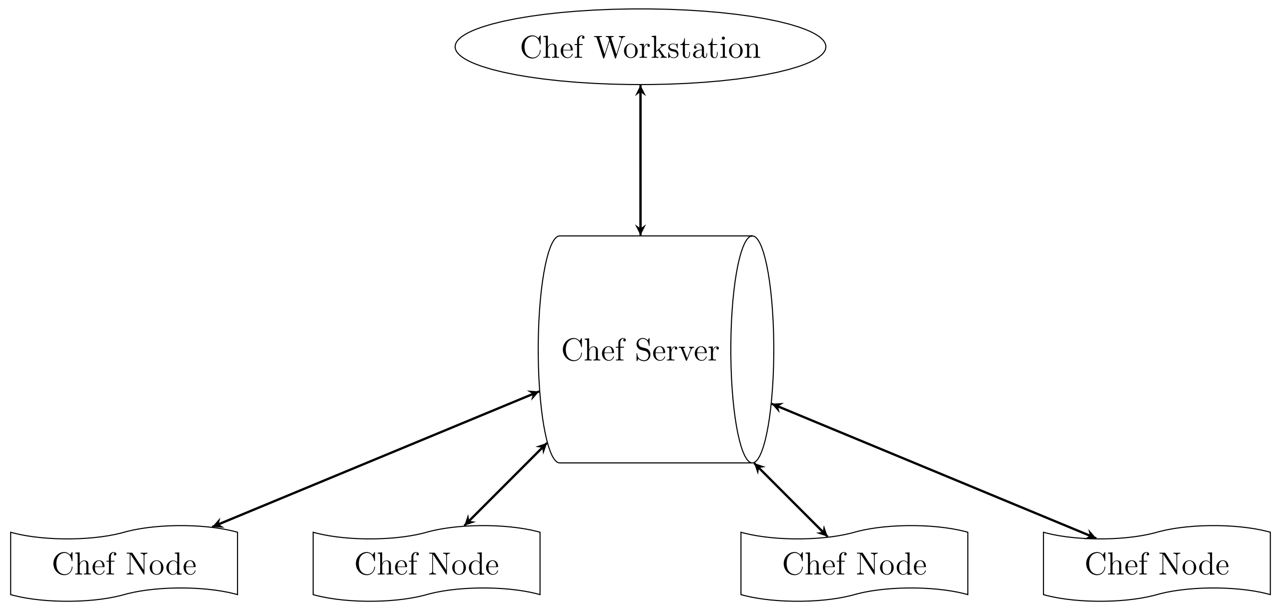
1. **Workstations:** All the configuration code is created, tested, and changed at workstations, which are personal computers or virtual servers. There can exist as many workstations as is necessary.

2. **Chef Server:** The Chef Server is the center of all of Chef's operations. It provides a communication pathway between the workstations (where infrastructure is coded) and the nodes (where the configurations are deployed by the Chef client). Any configuration files, metadata, cookbooks and other information created on workstations are stored on the Chef server. Aditionally, the Chef server contains information regarding the state of all nodes at the time of the last chef-client run. Any changes made to infrastructure code must pass through the Chef server in order to be applied to nodes. Before accepting changes, the Chef server authenticates all communication via its REST API using public key encryption.

3. **Nodes:** These are the servers that are managed by Chef i.e. the machines to which changes are being pushed. Nodes are generally a fleet of multiple machines that require the benefits of automation. Chef can manage nodes that are virtual servers, containers, network devices, and storage devices and a Chef client is installed on every node that is under management by Chef.

1.2 Architecture

In order to achieve its goals, Chef treats infrastructure as code. Instead of manually changing anything, the machine setup is described in a Chef *recipe*. *Cookbooks* store collections of recipes. Ideally, one cookbooks relates to a single task, but it can have numerous server configurations involved. The chef server stores each of the cookbooks and as a new chef client node checks in with the server, recipes are sent to tell the node how to configure itself. Afterwards, the client will occassionally check in with the server to see if anything needs to be changed. If something does need to be changed, then the client deals with it. Patches and updates can be applied to the entire infrastructure by changing the recipe; there is no need to interact with each machine individually. As such, Chef utilizes a three-tier client-server architecture. The working units (like cookbooks) are developed on the Chef workstation. From the command line utilities like Knife, they are uploaded to the Chef server and all the nodes which are present in the architecture are registered with the Chef server.



1.3 Features

Chef provides the following features:

- Automatic Backup
- Automatic Notifications
- Compliance Management
- Configuration Management
- Data Recovery
- Data Visualization
- Real Time Analytics
- Real Time Data
- Real Time Monitoring
- Real Time Notifications
- Real Time Reporting
- Real Time Updates
- Server Monitoring

1.4 Official Website

The official website for Chef is located at https://www.chef.io.

1.5 Latest Version

1.6 Advantages

1.7 Disadvantages

2 Ansible

2.1 Components

2.2 Architecture

2.3 Features

2.4 Official Website

2.5 Latest Version

2.6 Advantages

2.7 Disadvantages

3 Fuel

3.1 Components

3.2 Architecture

3.3 Features

3.4 Official Website

3.5 Latest Version

3.6 Advantages

3.7 Disadvantages

4 Puppet

4.1 Components

4.2 Architecture

4.3 Features

4.4 Official Website

4.5 Latest Version

4.6 Advantages

4.7 Disadvantages

5 Compass

5.1 Components

5.2 Architecture

5.3 Features

5.4 Official Website

5.5 Latest Version

5.6 Advantages

5.7 Disadvantages