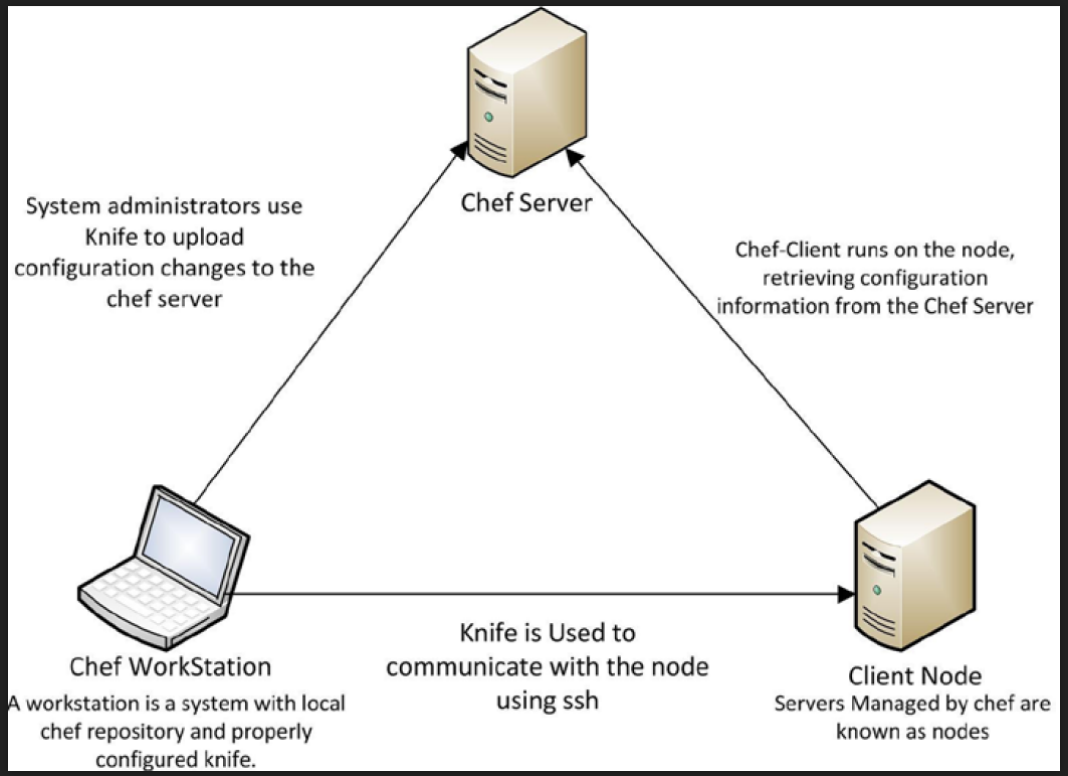
Chef Mount Everest Consulting

Overview

- Chef is a powerful automation platform that transforms complex infrastructure into code
- Chef was initially written in Ruby, but the latest version is a mixture of Erlang and Ruby.
- A single chef server can handle up to 10,000 nodes.
- Anything that can run the chef-client can be managed by Chef
- Chef can manage physical machines, virtual machines, containers, or cloud-based instances.
- Chef can run in client/server mode or in a standalone configuration named "chef-solo".
- The chef-client is an agent that runs on a node and performs the actual tasks that configure it.
- The Chef server is the central repository for all configuration data.
- The chef-client and Chef Server communicate with each other. For secure communications, they use a combination of public and private keys that ensures the Chef server responds only to requests made by the chef-client.

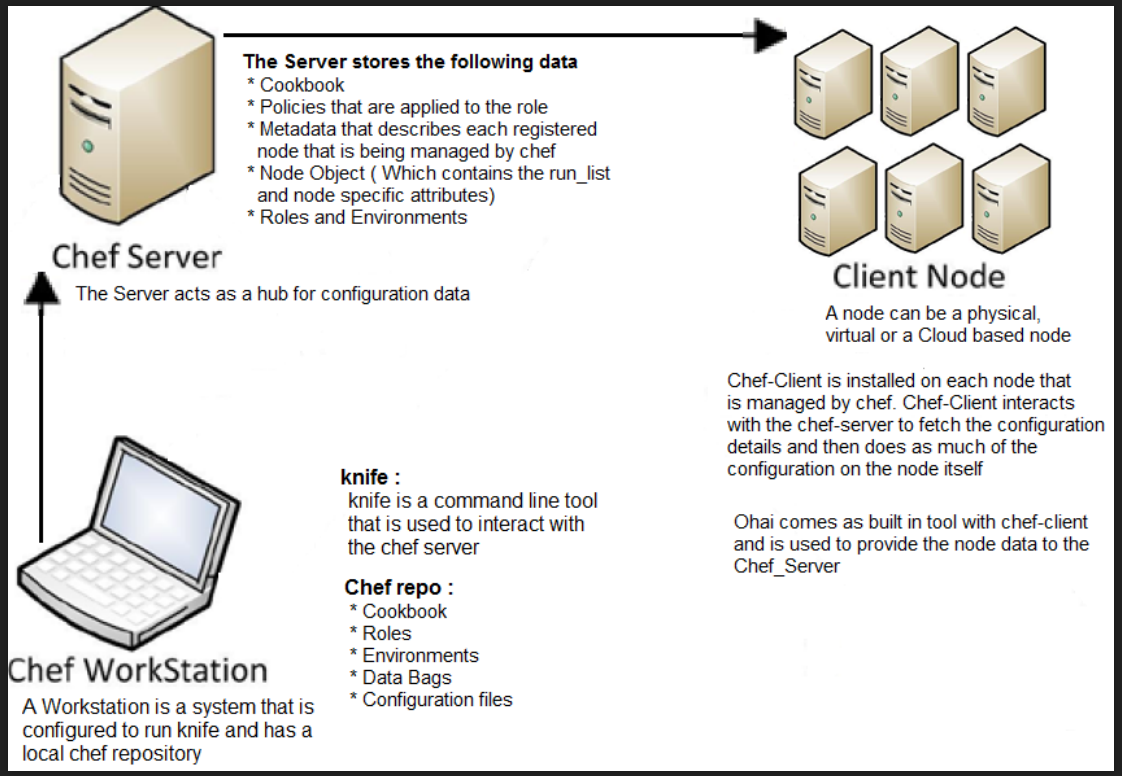
Architecture



Architecture

- **Chef Server -** This is the central location that stores configuration recipes, cookbooks, and node and workstation definitions
- There are three types of chef server
- **Enterprise Chef Server -** Enterprise chef is the paid version of the chef server which comes with two types of installations: one is on premise installation and the other is the hosted version.
- **Open source chef -** The open source chef has most of the feature of the enterprise version. This version of chef can be installed only in stand-alone mode.
- **Chef solo –** Chef Solo comes with the chef client package and is used to manage a node without any access to the server.
- **Chef Node –** Also known as chef client. Chef client are the deployment target that configured and managed by Chef.
- There two types of nodes that chef can manage
- **Cloud-based –** This type of node is basically hosted on any of the cloud providers (e.g., AWS or Windows Azure). There is a chef CLI known as knife which can be used to create instances on the cloud.
- **Physical -** It can be hardware or a virtual machine that exists in our own environment.
- **Chef Workstations -** Chef Workstations are where main development will takes place, all Chef Configuration files are created or edited. The configuration files are then pushed to the Chef server, where they will be available to deploy to any nodes.
- www.chef.io

Components


The Server stores the following data
* Cookbook
* Policies that are applied to the role
* Metadata that describes each registered node that is being managed by chef
* Node Object (Which contains the run_list and node specific attributes)
* Roles and Environments
Chef Server
The Server acts as a hub for configuration data
Client Node
A node can be a physical, virtual or a Cloud based node
Chef-Client is installed on each node that is managed by chef. Chef-Client interacts with the chef-server to fetch the configuration details and then does as much of the configuration on the node itself
Ohai comes as built in tool with chef-client and is used to provide the node data to the Chef_Server
knife :
knife is a command line tool that is used to interact with the chef server
Chef repo :
* Cookbook
* Roles
* Environments
* Data Bags
* Configuration files
Chef WorkStation
A Workstation is a system that is configured to run knife and has a local chef repository

Cookbook

- **Basic Cookbook Concepts –**
- Cookbook serves as the fundamental unit of configuration and policy details that Chef uses to bring a node into a specific state.
- Cookbooks are usually used to handle one specific service, application, or functionality.
- Cookbooks are created on the workstation and then uploaded to a Chef server.
- Once the Cookbook is uploaded in server then recipes and policies described within the cookbook can be assigned to nodes as part of the node's "run-list".
- A run-list is a sequential list of recipes and roles that are run on a node by chef-client in order to bring the node into compliance with the policy we set for it.
- Cookbook is organized into various directories; below section has detail descriptions about these directories.

Cookbook Structure


```
% tree cookbooks/smartmontools
cookbooks/smartmontools
├── README.md
├── attributes
├── definitions
├── files
│   └── default
├── libraries
├── metadata.rb
├── providers
├── recipes
│   └── default.rb
├── resources
└── templates
    └── default
```

Cookbook Structure

- Below is the description about few important directories
- **Attributes –** This directory contains attribute definitions that can be used to override or define settings for the nodes that will have this service.
- **Files –** This directory describes how chef should distribute files throughout the node on which this cookbook is deployed.
- **Recipes –** The directory contains the "recipes" that define how the service should be configured.
- **Templates –** This directory is used to provide more complex configuration management. We can provide entire configuration files that contain embedded Ruby commands.

Chef Server

- Chef server is the central storage location of cookbook, roles and other configuration information's. As we have seen there are three types of Chef Server but here we will follow enterprise Chef server configuration steps which will be host in AWS.
  - **Pre-requisites**
- Before starting Chef Server configuration we need to make sure we have a Linux instance ready in AWS with admin privilege. Chef Server 12 onwards supports command line installation. We can only install Chef Server in Red Hat, CentOS 6 and Ubuntu.

Chef Server Installation

- **Configuration Chef Server**

- Follow the steps as per the order given below. Below steps is for Red Hat Linux.
- Download the package from http://downloads.chef.io/chef-server/.
- Upload the package in the machine that will be configured as Chef Server.
- [Here we are uploading package in /tmp directory]
- Install the Chef Server package on the server
- $ rpm -Uvh /tmp/chef-server-core-<version>.rpm
- Need to run below command to start all required services
- $ chef-server-ctl reconfigure Run the following command to create an administrator. The --filename option will save the RSA private key to a specified path.

- $ chef-server-ctl user-create user_name first_name last_name email password --filename FILE_NAME **(Run the following command to create an organization.)**
- $ chef-server-ctl org-create short_name full_organization_name --association_user user_name --filename FILE_NAME Now we can login below Chef Server URL
- https://10.X.X.X/signup

Chef Client/Workstation

- From Chef Workstation we can perform any kind of administration work. Workstation can be many flavors like Windows, Mac OS, Linux, and Unix. In order to configure any system as Chef Workstation we need to follow below steps irrespective to any OS flavor. In this document Workstation configuration in Windows has been described.
-
- Install Chef Client Package
- Copy Chef Starter kit
- Configure knife
- **Pre-requisites**
- Before starting Chef Client configuration in Windows system we need to make sure below points already has been taken care.
-
- Chef enterprise server set-up and configuration
- Chef starter-kit
- Required port open/Open VPN connection
- Administrative privilege on local system

Chef Workstation

- **Configuration Chef Workstation**
- Follow the steps as per the order given below.
  - **Install Chef Client Package**
- Login to below URL.
- URL - https://www.chef.io/download-chef-client/
-
- Download "msi" installer according to type of workstation.
-
- Once the Chef Client installer successfully downloaded, install it in local system.
  - **Copy Chef Starter Kit**
- Get the starter kit from appropriate source (responsible team/person who configured Chef Enterprise server)
- Un-zip the content of *StarterKit.zip* file to **%HOMEPATH%** path of current user
  - **Configure Knife**
- When we installed Chef Client package in our local system, its installed knife utility as well.
-
- **Checking knife version**
- cd %HOMEPATH%\chef-repo\.chef
- knife –version
- **List Chef Clients**
- Now we will check Chef Client by running below command if we will not see expected result then we need to configure knife in chef workstation
-
- knife client list

Chef Workstation

- **Note – If we will not observe expected result**
- **Install Certification**
- Run "knife ssl fetch" to install certification
- **Verify the certificate**
- Under "%HOMEPATH%\chef-repo\.chef" path a sub-directory will be created named "trusted_certs" under this certification file will be available.
-
- **Make entry in host file**
- In local system's host file need to make an entry of Chef Server Host name and IP address
-
- Example –
- 10.0.140.226 ip-10-0-140-226.ec2.internal
-
- **Change Chef Server URL**
- We need to change Chef Server URL in knife.rb file as per host entry.
- Sample entry is below, here we have to use chef server host name instead of IP address.
-
- chef_server_url https://ip-10-0-140-226.ec2.internal/organizations/acc
-
- **Knife SSL Check**
- Using "knife ssl check" subcommand we can verify the SSL configuration for the Chef server. When we run this command the certificate file located in "%HOMEPATH%\chef-repo\.chef\trusted_certs" path has been verified for valid properties.

Chef Workstation

- **Verify knife configuration**
- Finally it's time to verify knife configuration, two ways we can verify knife configuration.
-
- **Using "knife client list" command**
- As we have seen earlier this command lists the entire registered API client. This ensures that knife is configured to our workstation.
-
- **Using login to Chef Server URL**
- Open the below Chef Server URL in browser, if we can able see the login page we can assume knife has been configured.
-
- https://*<Chef Server IP Address>*/organizations/*<Organization Name>*
- OR
- https://*<Chef Server Host Name>*/organizations/*<Organization Name>*