skillspeed

for the serious learner

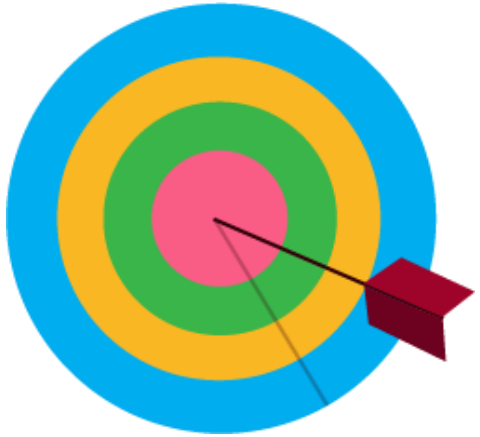
DevOps

Understanding Chef

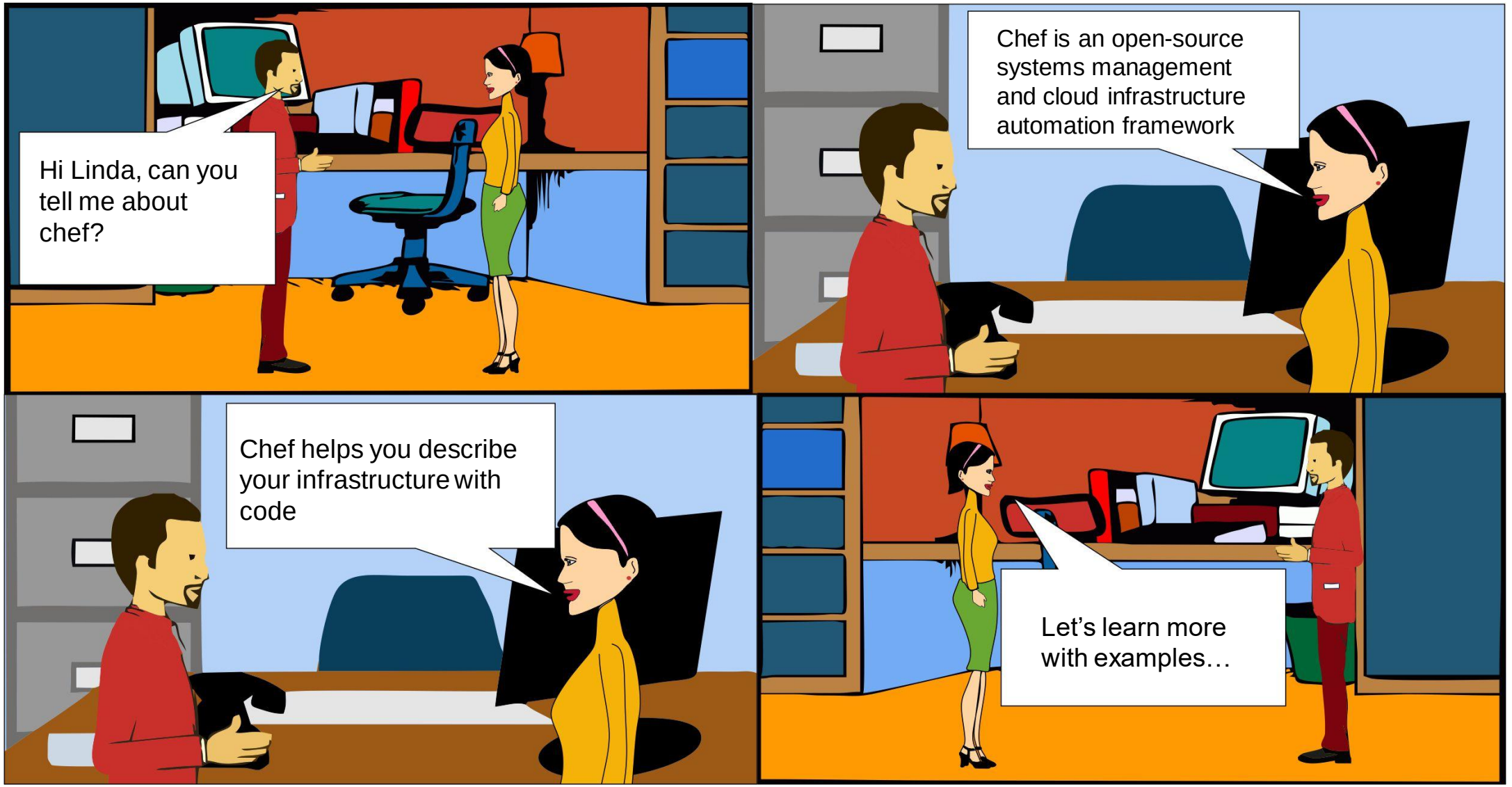
Objectives

This session will help you to understand:

- Configuration Management
- Chef Basics
  - What is Chef?
  - Why Chef?
  - History of Chef
  - Salient Features of Chef
  - Main components of Chef
- Chef Architecture, Chef Features
- Cookbooks, Recipes, Attributes, Files, Templates and Installation



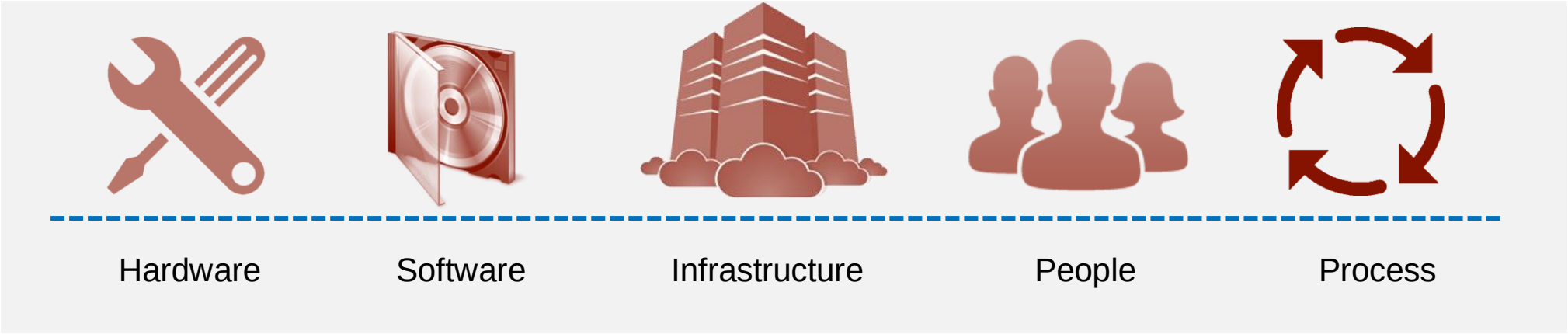
Introduction to Chef



Configuration Management

What Is Configuration Management?

With respect to IT, configuration management covers the set of engineering practices for managing the following entities involved in delivering software applications to consumers:



Why You Need a Configuration Management Tool to Automate IT?

There are a number of reasons why automated configuration management tools play a vital role in managing complex enterprise infrastructures

Here are four of the most popular reasons:

- Consistency
- Efficient Change Management
- Simplicity in Rebuild
- Visibility

Introduction to Chef

- Chef is an open-source systems management and cloud infrastructure automation framework created by Opscode
- Chef helps you describe your infrastructure with code
- Since your infrastructure is managed with code, it can be automated, tested and reproduced with ease
- Chef runs in two modes: Client/Server and standalone configuration
- It is written in Ruby and Erlang

Versions of Chef

Chef is available in three different versions:

- Private chef – An enterprise version which supports multi-tenancy and runs in-house behind a firewall
- Hosted Chef – A SaaS managed cloud service hosted by Opscode
- Open source Chef – A free download that requires that each instance of Chef to be configured and managed locally

Why Chef?

There are numerous reasons to use Chef:

- Chef immensely decreases the amount of documentation you ought to write
- Chef doesn’t change the scalability of the Bash
- Chef comes up with terrific technical approaches
- Chef is a tool that grows with you
- Chef can help you in holding back and stop you from reinventing the wheel

History of Chef

In February 2013, Opscode released version 11 of Chef

Name changed from "marionette", to “Chef”

They founded a new company with Barry Steinglass, Nathen Haneysmith, and Joshua Timberman to turn Chef into a product

Adam showed Chef to Jesse Robbins who saw its potential after running operations at Amazon

Created by Adam Jacob as a tool for his consulting company

Salient Features of Chef

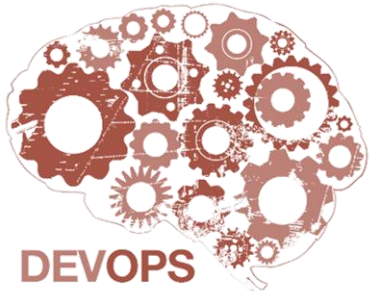
- Chef has different flavours of automated solutions for current IT operations
- Enables highly scalable, secure and a fault-tolerant automation capability features of infrastructure
- Flexible in nature
- Has a very strong community
- Can quickly handle all types of traditional dependencies and manual processes of the entire network
- Well suited for cloud instances

Salient Features of Chef (Cont'd)

Open Source

Easy to Learn

Cloud Friendly

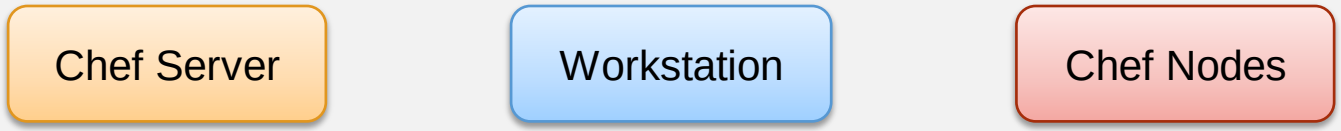


DevOps Friendly

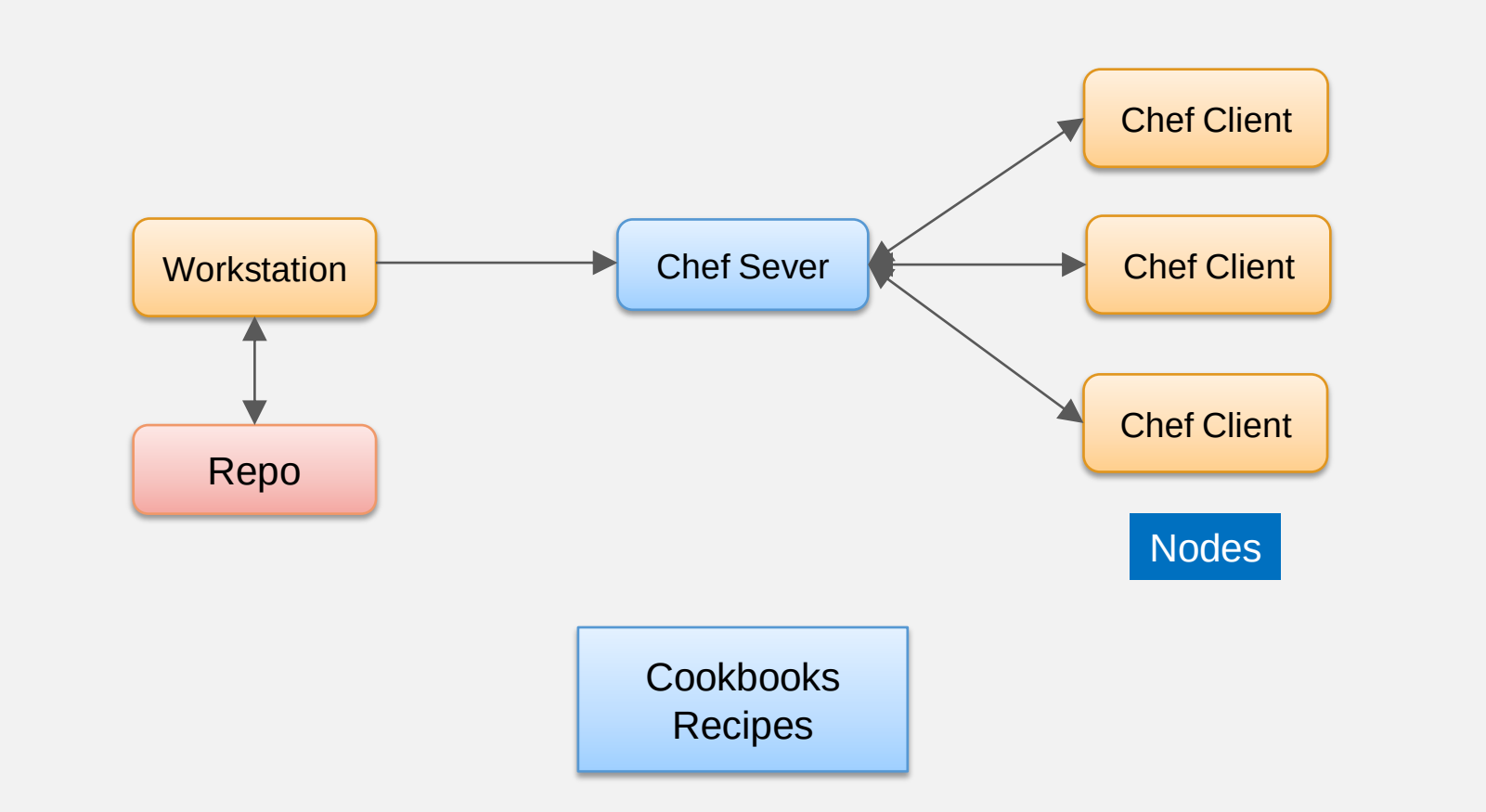
Strong Community Support

Chef Components

Chef environment is typically made up of three components



Chef Components (cont'd)



Chef Workstation

Workstation is the development machine from where users run all configuration related tasks which includes creation of cookbooks and recipes, updating chef-repo, interacting with the single Chef server and many more

Workstation is the place where user will spend most of their time with Chef and will do most of their work that includes:

- Development of the cookbooks and the recipes
- Retaining the chef-repo synchronization with version source control
- Making the use of knife to upload items from the chef-repo to the Chef server
- Configuring organizational policies i.e. defining roles as well as environments and ensuring that critical data is stored in data bags
- Communicating with the nodes whenever needed, such as carrying out a bootstrap operation

Chef Server

- The Chef server is the brains of the operation which acts as a hub to store configuration data that is available to every node and also administer access rights
- It preserves all the information regarding your infrastructure
- All the client nodes are registered within the server
- Clients communicates with the server in order to get the correct configuration details from the server such as recipes, templates, file distributions and apply it to the nodes
- This type of flexible and scalable approach distributes the configuration effort throughout the organization

Knife

- knife is a command-line tool that provides an interface between a local chef-repo and the Chef server
- All knife subcommands share a set of common options and usage patterns
- knife settings are stored in the knife.rb file

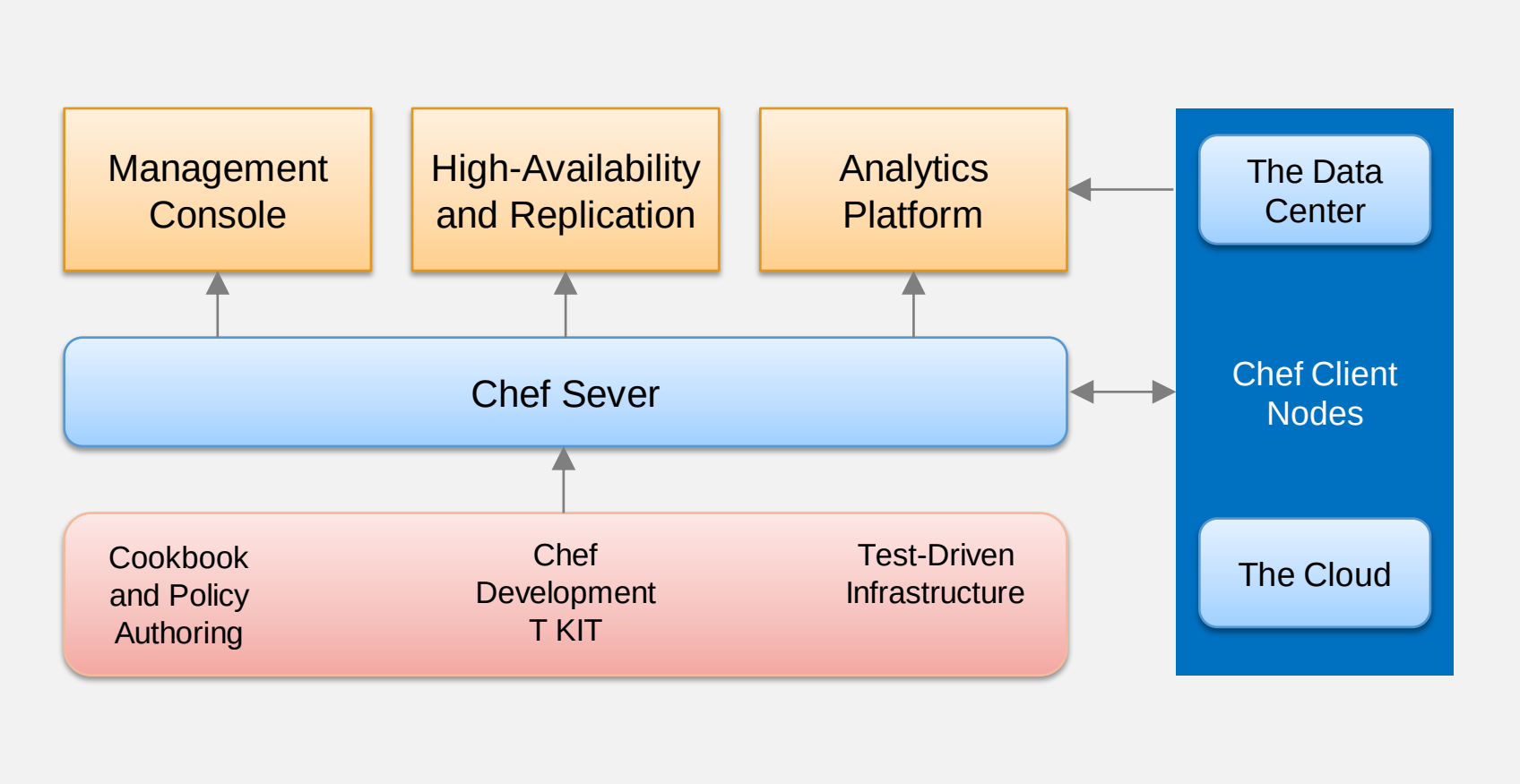
Some of knife built-in subcommands are:

- knife bootstrap
- knife client
- knife configure
- knife cookbook
- knife cookbook site (which uses the Cookbooks Site API)
- knife data bag
- knife delete
- knife deps
- knife diff
- knife download…

Chef Nodes

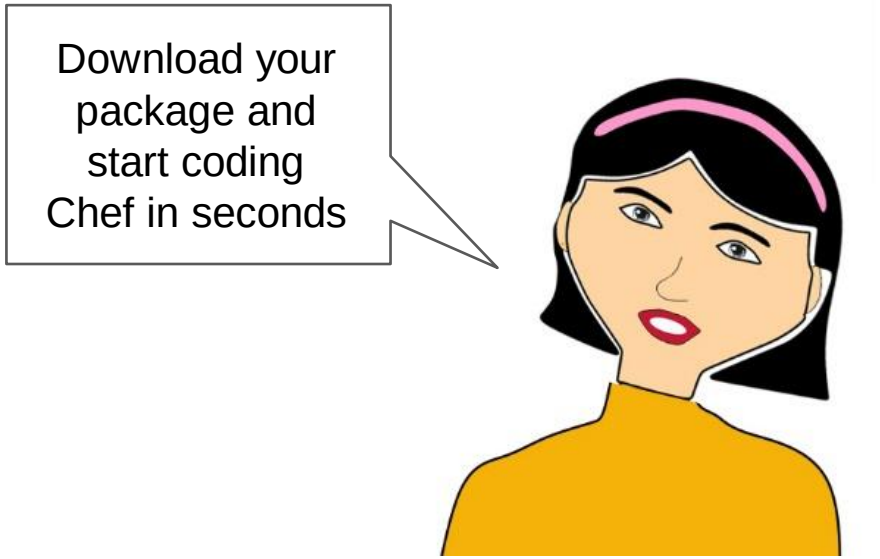
- A node can be a physical, virtual or cloud machine that contain the chef-client which performs all the infrastructure automation
- Nodes are the computers that we manage using chef and it act as server in our infrastructure

Chef Architecture



The Chef Development Kit

The Chef Development Kit (ChefDK) brings the best-of-breed development tools built by the awesome Chef community to your workstation with just a few clicks



Chef Analytics

- The Enterprise Chef Analytics platform provides real-time visibility into what's happening on your Chef server
- It's the latest in a growing suite of features aimed at providing you with a comprehensive view of your Chef-managed infrastructure

Chef Management Console

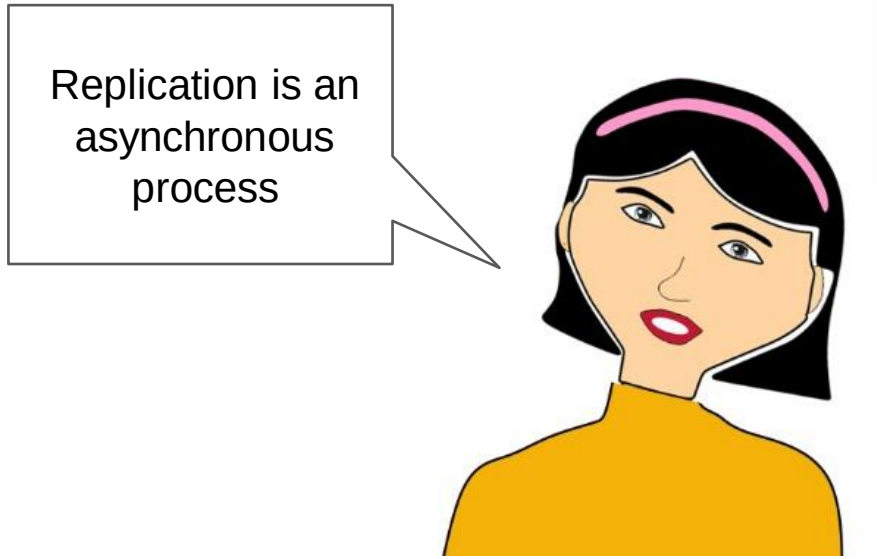
Use the web-based management console to control access via role-based access control (RBAC), edit and delete nodes, and reset private keys. Keep up to date with what's happening during Chef client runs

Chef Features

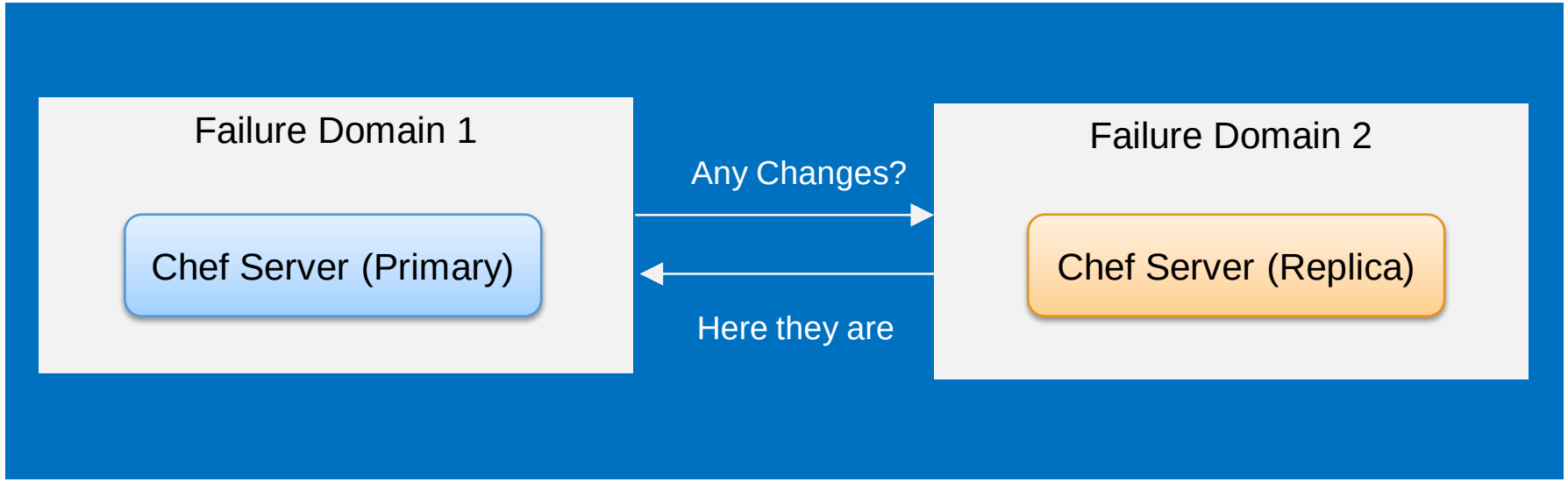
- High availability (HA) is a premium feature of Chef that protects you if your Chef server fails
- Just as you run more than one web server or database to protect yourself against a single point of failure, you also want to make sure that the critical data on your Chef server is always available

Chef Features (Cont'd)

Replication is a premium feature of Chef that ensures consistency between Chef servers located in different failure domains



Chef Features (Cont'd)



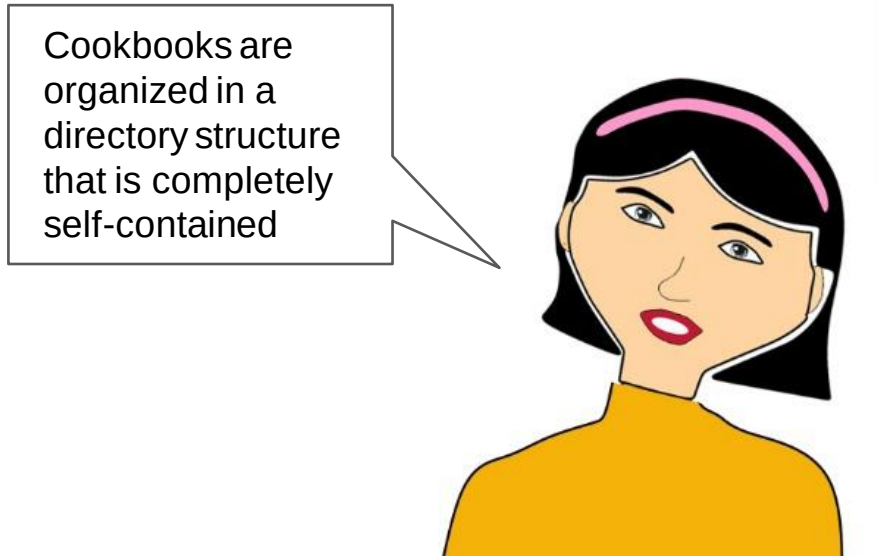
The replica polls the primary to see if there are any changes to the organizational data. If there are, the primary sends them to the replica

Chef Cookbooks

- Cookbooks serve as the fundamental unit of configuration and policy details that Chef uses to bring a node into a specific state i.e. Chef uses cookbooks to perform work and make sure things are as they should be on the node
- Cookbooks are usually used to handle one specific service, application, or functionality
- Cookbooks are basically packages for infrastructure choices

Chef Cookbooks (cont'd)

- Cookbooks are created on the workstation and then uploaded to a Chef server
- From there, 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 you set for it
- In this way, the configuration details that you write in your cookbook are applied to the nodes you want to adhere to the scenario described in the cookbook



Chef Recipes

- A recipe is the main workhorse of the cookbook
- A cookbook can contain more than one recipe, or depend on outside recipes
- Recipes are used to declare the state of different resources
- A recipe is a list related resources that tell Chef how the system should look if it implements the recipe
- When Chef runs the recipe, it checks each resource for compliance to the declared state
- If the system matches, it moves on to the next resource, otherwise, it attempts to move the resource into the given state

Chef Resources

Resources can be of many different types. Some common ones are:

- Package: Used to manage packages on a node
- Service: Used to manage services on a node
- User: Manage users on the node
- Group: Manage groups
- Template: Manage files with embedded ruby templates
- Cookbook file: Transfer files from the files subdirectory in the cookbook to a location on the node
- File: Manage contents of a file on node
- Directory: Manage directories on node
- Execute: Execute a command on the node
- Cron: Edit an existing cron file on the node

Chef Attributes

- Attributes in Chef are basically settings
- There are several different kinds of attributes that can be applied, each with a different level of precedence over the final settings that a node operates under
- When creating a cookbook, attributes can be set for service in the attributes subdirectory of cookbook. Then these values can be referenced in other parts of cookbook

Chef Files

- The files subdirectory within the cookbook contains any static files that we will be placing on the nodes that use the cookbook
- For instance, if any simple configuration files that we are not likely to modify can be placed, in their entirety, in the files subdirectory
- A recipe can then declare a resource that moves the files from that directory into their final location on the node

Chef Templates

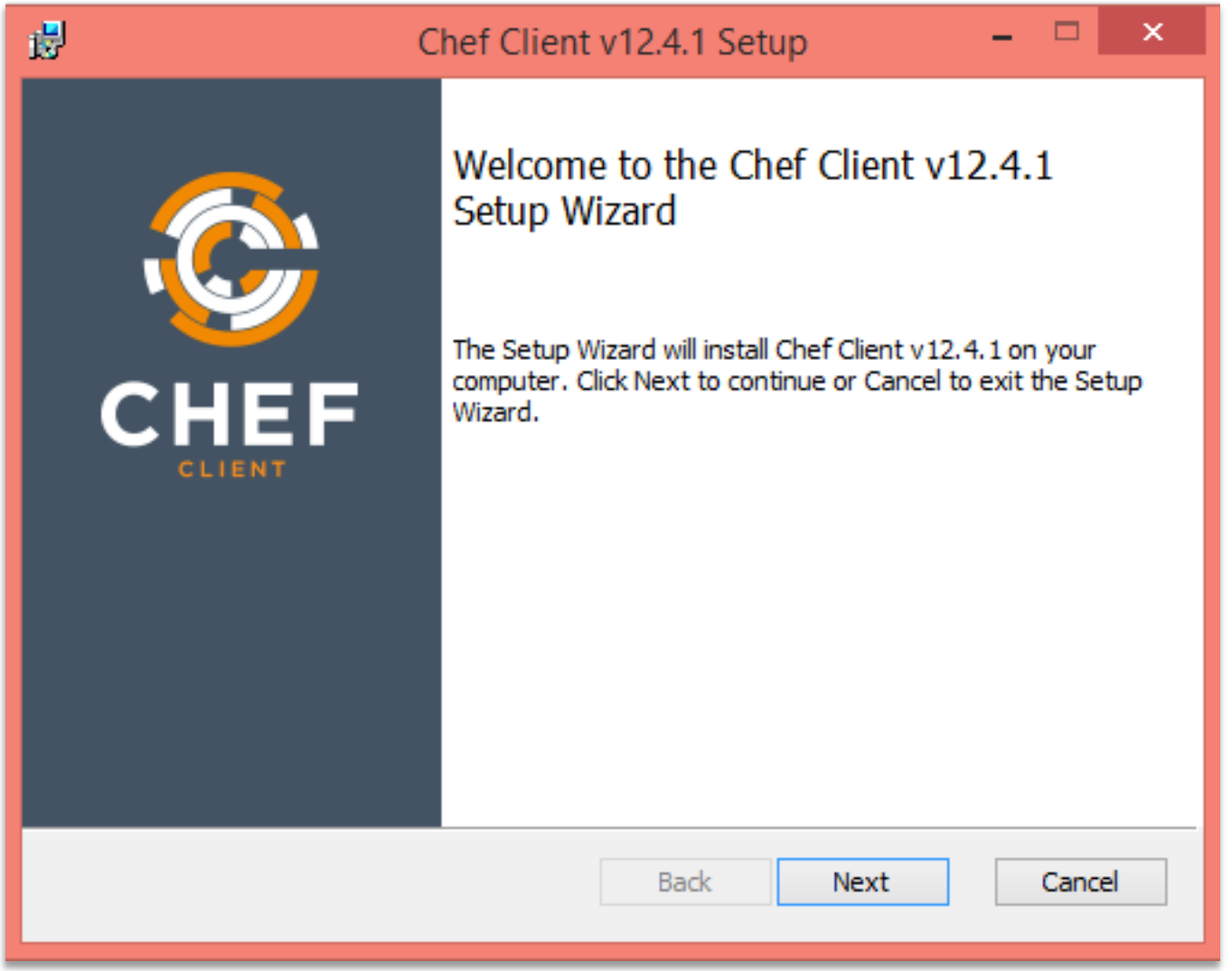
- Templates are similar to files, but they are not static
- Template files end with the .erb extension, meaning that they contain embedded Ruby
- These are mainly used to substitute attribute values into the file to create the final file version that will be placed on the node

Chef Installation

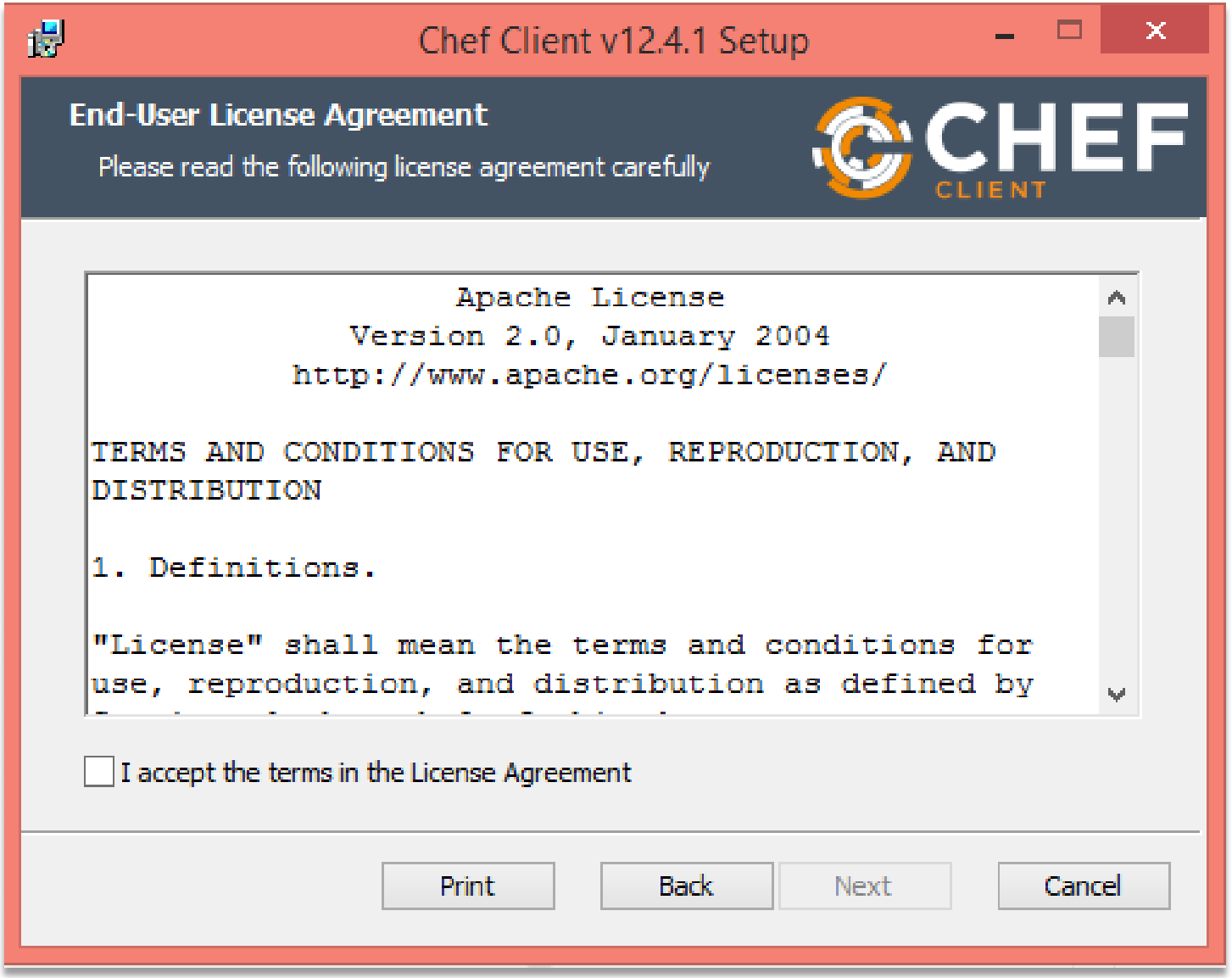
To install the chef-client on Microsoft Windows, do the following:

- Go to http://www.chef.io/chef/install
- Click the Chef Client tab
- Select Windows, a version, and an architecture
- Under Downloads, select the version of the chef-client to download, and then click the link that appears below to download the package
- Ensure that the MSI is on the target node
- Run the MSI package and use all the default options

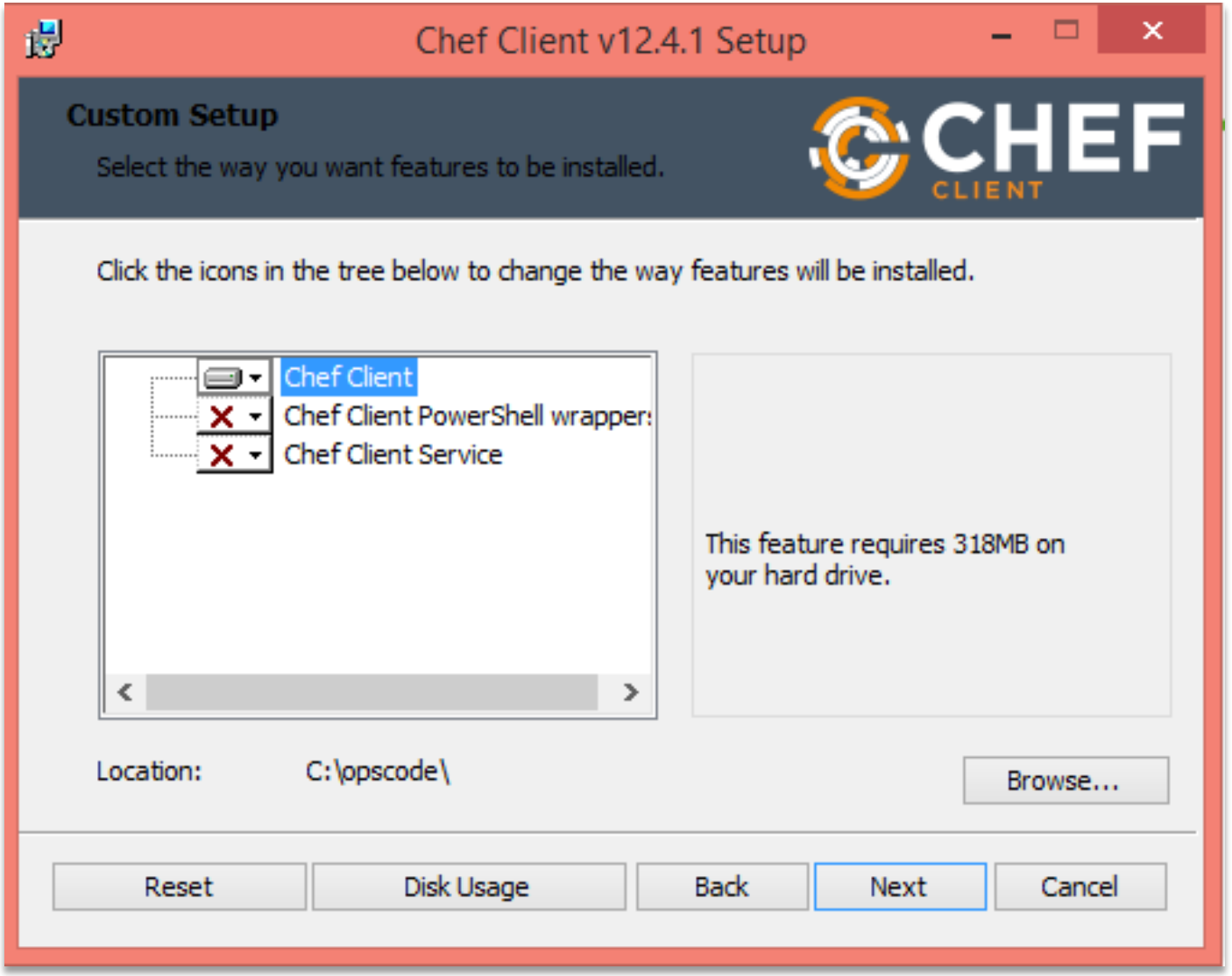
Chef Installation (cont'd)



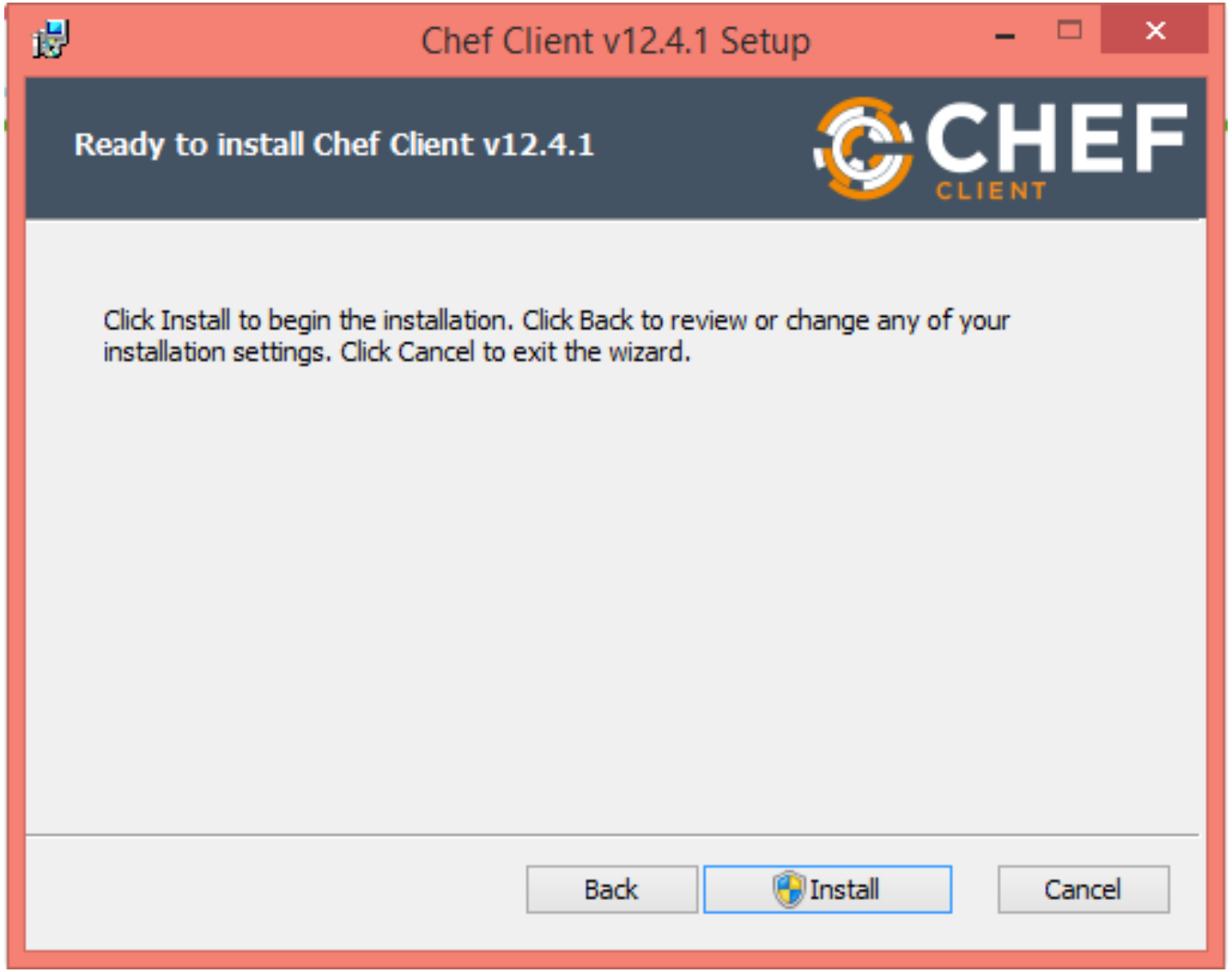
Chef Installation (cont'd)



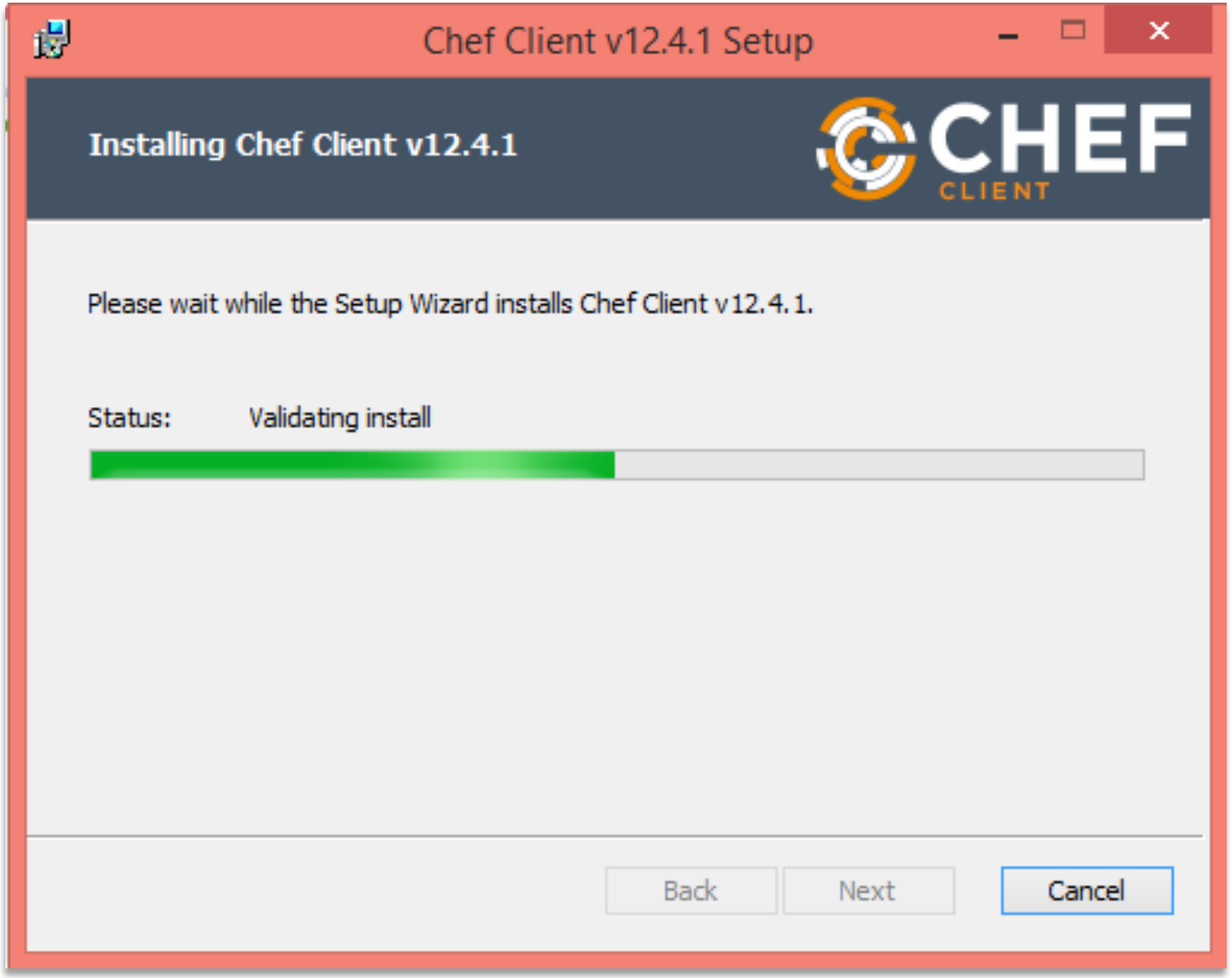
Chef Installation (cont'd)



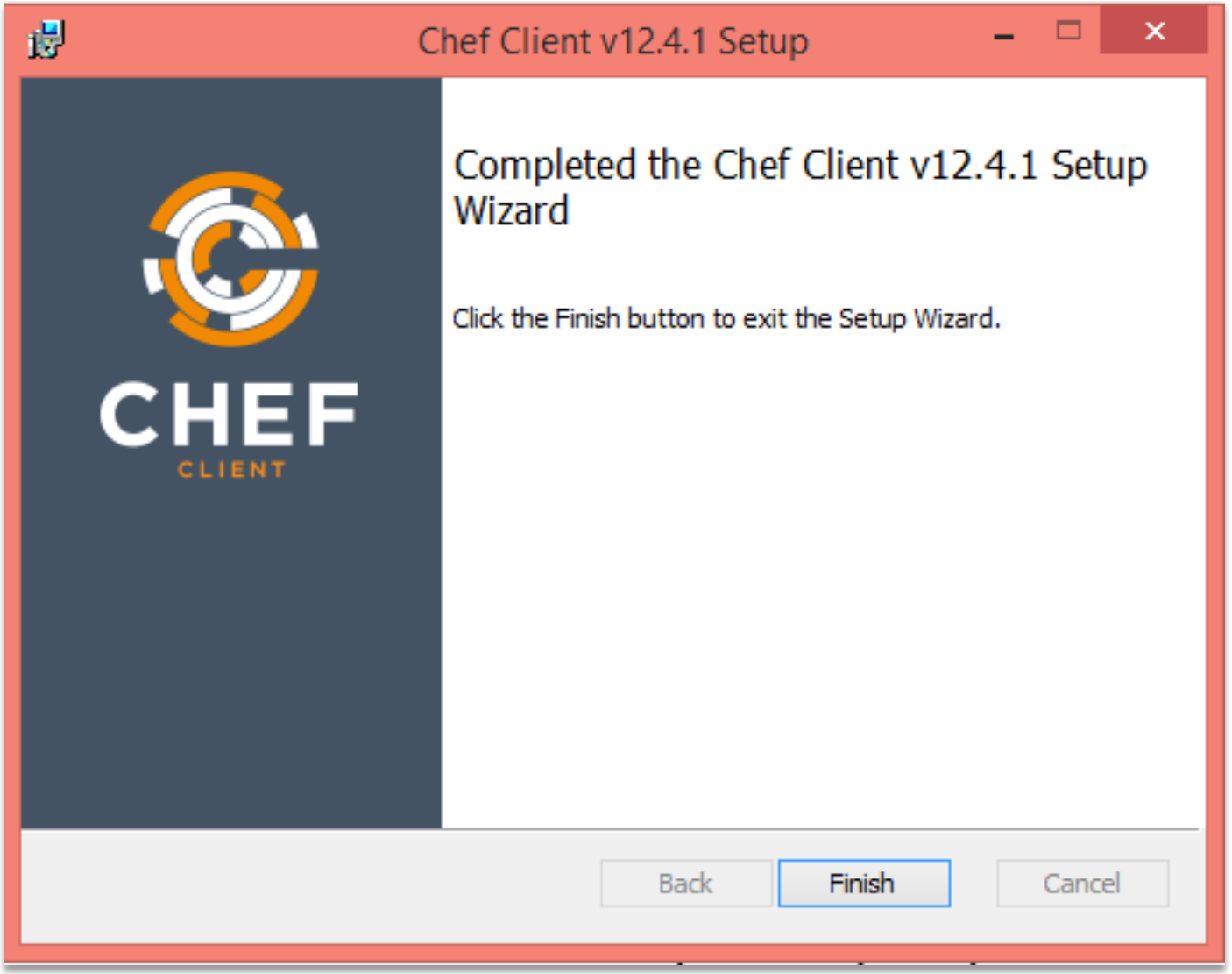
Chef Installation (cont'd)



Chef Installation (cont'd)



Chef Installation (cont'd)



Chef Installation (cont'd)

skillspeed
for the serious learner

Run as a Service

- To run the chef-client at periodic intervals (so that it can check in with the Chef server automatically), configure the chef-client to run as a service or as a scheduled task

- This can be done via the MSI, by selecting the Chef Client Service option on the Custom Setup page or by running the following command after the chef-client is installed:

```
$ chef-service-manager -a install
```

- And then start the chef-client as a service:

```
$ chef-service-manager -a start
```

- After the chef-client is configured to run as a service, the default file path is: c:\chef\chef-client.log

Some Users of Chef

Questions

skillspeed
for the serious learner