Infrastructure as Code using Chef

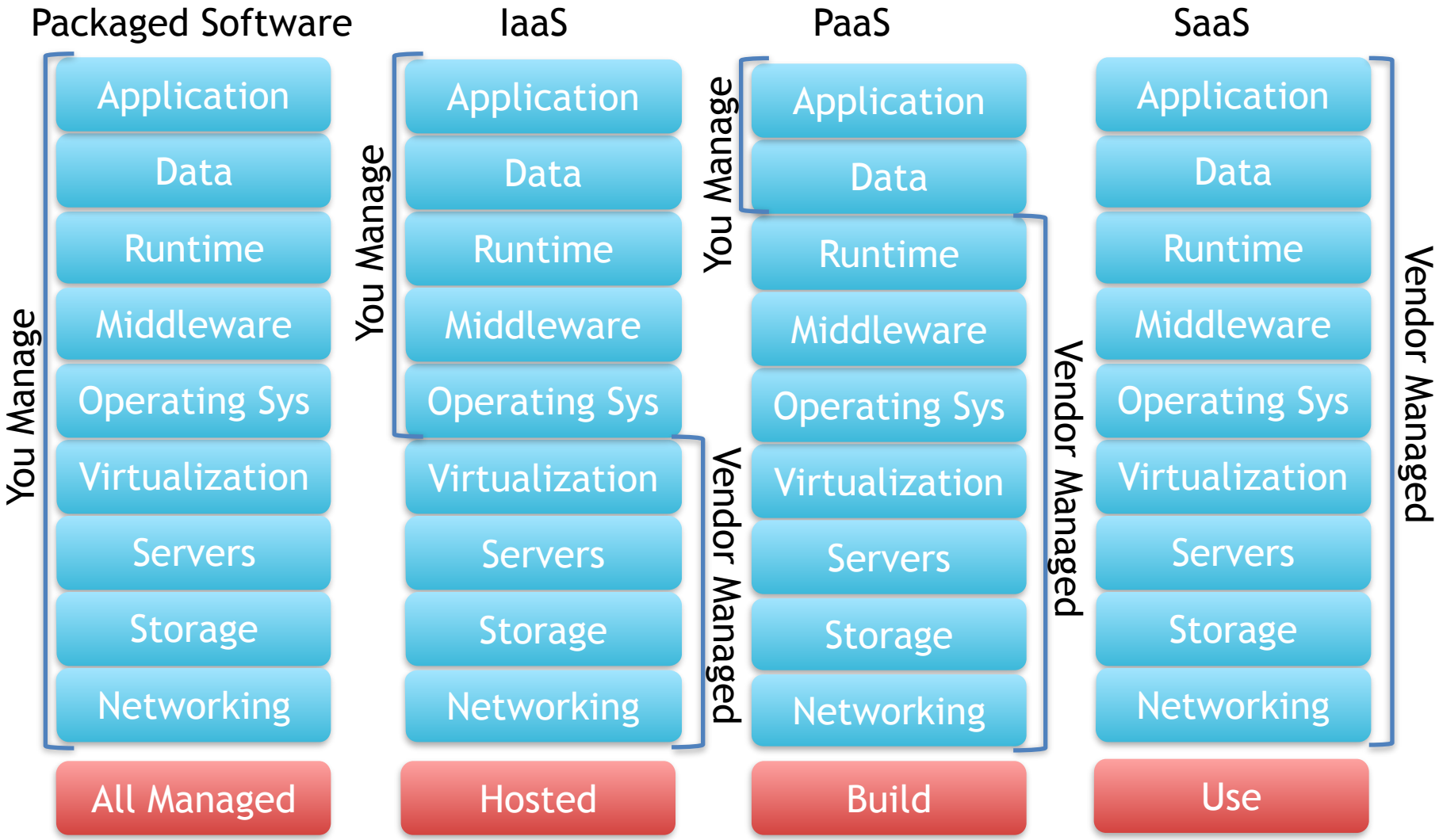
Prepared for Washington Metro CM Working Group

September 1, 2015

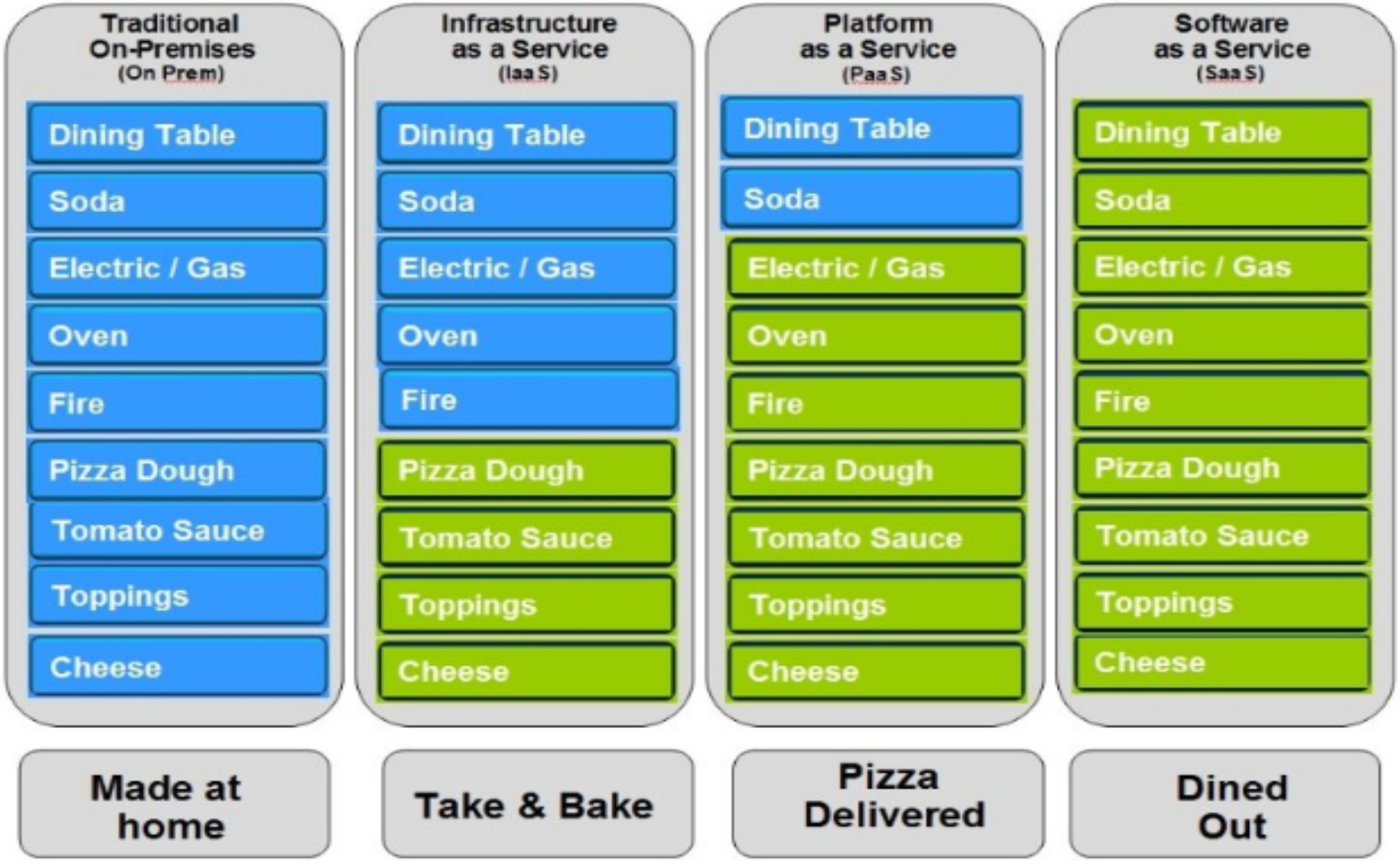
Infrastructure as Code:
What are the Objectives?

- **Lower Total Cost of Ownership through:**
  - ✓ Highly automated and scalable compute resources,
  - ✓ Self-provisioned cloud storage and network capability
  - ✓ Elastic compute services based on events or schedules
  - ✓ Pay for what you use improves Demand, Capacity and Availability management
- **Improve control over Infrastructure:**
  - **"Enable the reconstruction of the business from nothing but a source code repository, an application data backup, and bare metal resources"**
  - *Jesse Robins*
- **Improve Quality and Turn-around of Environment Provisioning for Software Development**
- **Reduce or transfer hardware asset management risk to third-party**
- **DTSTTCPW - Do The Simplest Thing That Could Possibly Work**
- **Critical to Success of DevOps !!**

Service Layers and Risk



Pizza as a Service



Chef Offerings – 30,000 mile view

- Programmatically provision and configure infrastructure components based on version controlled code base

- Ability to reconstruct infrastructure and software services from code repository, data backup, and compute resources

Managing Complexity

Overview of Chef Model

- Organization – Independent tenants of Chef Enterprise Server
- Environment - Groupings of attributes to model a workflow (e.g. dev, test, prod).
- Role – Used to model the types of servers in your infrastructure
- Node – Belongs to one organization, one environment, has a run-list and zero or more roles
- Resource – an infrastructure configuration item and its desired state
- Recipes – Configuration specifications describing the resources and their desired state
- Cookbooks – Collection of related recipes and supporting files managed together for consistency and to enhance re-use
- Run list – Ordered list of policies a Node should follow to converge to desired state Stored in Chef Server, retrieved during chef-client run on node
- Data Bags – data sets that are available to all nodes. Can be encrypted

Overview of Chef Server

Let’s see all that in the Hosted Chef Server:

https://manage.chef.io/login

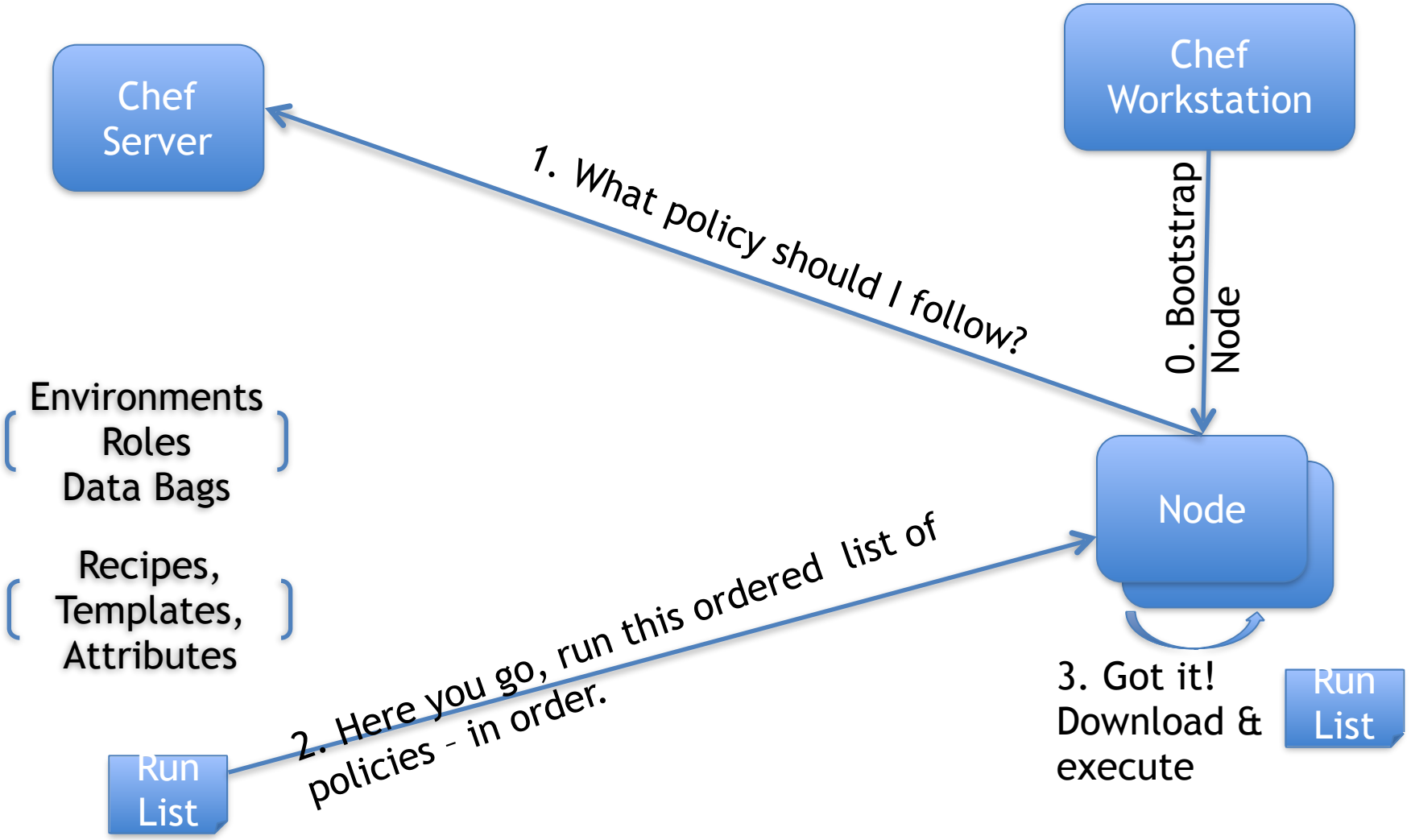
You can use the SaaS Hosted Chef Server or install enterprise chef server within your own infrastructure

Managing Complexity with Chef

1. Determine the desired state of your infrastructure
2. Identify the Resources required to meet that state
3. Gather the Resources into Recipes
4. Develop providers for target platforms, as necessary
5. Compose a Run List from Recipes and Roles
6. Apply a Run List to each Node in your Environment
7. Avoid Configuration Drive: Maintain updates to infrastructure through updates to Recipes

➢Hands-off infrastructure!

Converging a Node: How it Works



Chef Cookbook: Resources

Are **Typed**

```
template 'apache2-conf-charset' do
  path "#{node['apache']['dir']}/conf.d/
charset.conf"
  source 'charset.erb'
  owner 'root'
  group node['apache']['root_group']
  mode '0644'
  backup false
  notifies :restart, 'service[apache2]'
end

service 'apache2' do
  action :start
end
```

Chef Cookbook: Resources

Are Typed
Have a **Name**

```
template 'apache2-conf-charset' do
  path "#{node['apache']['dir']}/conf.d/
charset.conf"
  source 'charset.erb'
  owner 'root'
  group node['apache']['root_group']
  mode '0644'
  backup false
  notifies :restart, 'service[apache2]'
end

service 'apache2' do
  action :start
end
```

Chef Cookbook: Resources

Are Typed
Have a Name
May have **Parameters**

```
template 'apache2-conf-charset' do
  path "#{node['apache']['dir']}/conf.d/
charset.conf"
  source 'charset.erb'
  owner 'root'
  group node['apache']['root_group']
  mode '0644'
  backup false
  notifies :restart, 'service[apache2]’
end

service 'apache2' do
  action :start
end
```

Chef Cookbook: Resources

Are Typed
Have a Name
Have Parameters
Perform **Actions** to converge node to desired state

```
template 'apache2-conf-charset' do
  path "#{node['apache']['dir']}/conf.d/
charset.conf"
  source 'charset.erb'
  owner 'root'
  group node['apache']['root_group']
  mode '0644'
  backup false
  notifies :restart, 'service[apache2]'
end

service 'apache2' do
  action :start
end
```

Chef Cookbook: Resources

Are Typed
Have a Name
Have Parameters
Perform Actions to converge node to desired state
Can **Notify** other resources

```
template 'apache2-conf-charset' do
  path "#{node['apache']['dir']}/conf.d/
charset.conf"
  source 'charset.erb'
  owner 'root'
  group node['apache']['root_group']
  mode '0644'
  backup false
  notifies :restart, 'service[apache2]'
end

service 'apache2' do
  action :start
end
```

Chef Cookbook: Resources & Providers

Chef Resources are **Declarative** – they state *what* needs to be done, not *how*

Providers perform the actions, hiding implementation complexity from the recipes

Resources take **action** through **Providers**.
Chef determines the appropriate **Provider** to use based on the **Platform** on which the node is running.

Tools We'll Look at

Chef Standard Tools:

- Knife – configuration tool used for most tasks, such as managing nodes, environments, roles, etc.
- Chef-client – chef software used to converge a client to stated policy
- Chef-solo – stand-alone chef used for testing or deployments to hosts not connected to a chef server
- Chef-zero – in-memory chef server. Faster than chef-solo, preferred for testing
- Ohai – retrieves automatic and chef-created attributes of node.
- Fauxhai – ohai mock tool to trick chef into running on platforms other than host.
- Development Kit, includes:
  - chef – a new command-line tool
  - Berkshelf dependency manager
  - Test Kitchen for integration testing
  - ChefSpec – an RSpec extension for cookbook unit testing
  - Foodcritic – for static / lint analysis of cookbook
  - Rubocop – a ruby static code analyzer & formatter
  - Along with the standard chef tools (knife, chef-client, etc.)

Demo Time!

Live Demo of Chef, Virtual Box, Vagrant, test-kitchen, berksfile, AWS, chefdk and more.

Getting Started

Follow product guides for installation of each tool (see slide Set-up)

1. Download/install git, chef, chefDK, virtualBox, Vagrant.
2. Create a hosted chef account, download & install starter kit (with validation key).
3. Update your ~/.chef/knife.rb file with chef server URL
4. Test client/server connectivity
5. Develop & test your cookbooks
6. Install Knife plugins (e.g. knife ec2 for AWS) as needed

Set-up

VIrtualBox: https://www.virtualbox.org/wiki/Downloads
Vagrant: https://www.vagrantup.com/downloads.html
Vagrant boxes:https://atlas.hashicorp.com/boxes/search
Git: http://git-scm.com/download
Chef:
http://www.getchef.com/chef/install

```
curl -L http://www.getchef.com/chef/install.sh | sudo bash
```

Hosted Enterprise Chef: http://www.getchef.com
Chef Development Kithttps://downloads.chef.io/chef-dk/
Knife ec2 plugin: https://github.com/chef/knife-ec2
Rubymine: https://www.jetbrains.com/ruby/
Berkshelf: berkshelf.com

Documentation

Chef Training:

https://learn.chef.io/skills/fundamentals-series-week-1/

(Or see last slide!)

Blogs:

Pizza as a service:

- Original: https://www.linkedin.com/pulse/20140730172610-9679881-pizza-as-a-service

Good Follow-up:

- http://www.ektron.com/Blogs/Fred-Bals/Pizza-as-a-Service---On-Prem,-IaaS,-PaaS-and-SaaS-Explained-through-Pie-(not-Pi)/

DevOps:

https://dzone.com/articles/infrastructure-code-key-devops

Chef: https://www.chef.io/resources/

http://rspec.info

Books:

Test-Driven Infrastructure with Chef, 2nd Edition - http://shop.oreilly.com/product/0636920030973.do

Chef Infrastructure Automation Cookbook, Matthias Marchall:

https://www.goodreads.com/book/show/18430599-chef-infrastructure-automation-cookbook

Demo – the gory details

Knife:

- Knife help

Knife Searching for & downloading cookbooks from chef.io:

- Knife cookbook site search apache
- Knife cookbook site download apache

Chef Development Kit:

- Berks init
- Kitchen create (creates nodes for each platform defined in .kitchen.yml)
- Kitchen converge (bootstrap, run chef-client)
- Kitchen verify (run test suites)
- Kitchen list (list all nodes created for cookbook testing, with providers, etc).
- Foodcritic <file/dir>
- Rubocop <file/dir> #Note: don't use on .erb files!

Operations on Nodes:

- knife bootstrap 127.0.0.1:2222 -x vagrant -P vagrant --sudo -N "<name of node>"
- knife ec2 server create –r "role[webserver]" -I ami-673af20e -f t1.micro --ssh-key iic-keys --region us-east-1 --identity-file ~/.ssh/id_rsa --ssh-user ec2-user
- knife ssh name:i-83486b28 "sudo chef-client -o role[webserver]" -i ~/.ssh/id_rsa -x ec2-user

Questions / Comments

For training & additional support setting up Chef and/or AWS in your organization, contact the folks at Chef.io, Amazon Web Services, or:

Internet Informatics Corporation

1005 Charlton Place

Herndon VA

Contact: James Stallard

jamesstallard@yahoo.com

703-409-3552