

---

# Opscode Chef 活用セミナー

リリース *nii-0.1.Dec*

Jay Hotta

2012 年 12 月 19 日

---

# Contents

---

<b>1</b>	<b>はじめに</b>	<b>2</b>
<b>2</b>	<b>研修について</b>	<b>3</b>
2.1	トレーニング時間と注意事項	3
2.2	本日のインストラクタ	3
2.3	トレーニングの目標	4
2.4	Chef を学習する方法	4
<b>3</b>	<b>Agenda</b>	<b>7</b>
3.1	研修のアジェンダ	7
<b>4</b>	<b>Server Configuration Tool の必要性</b>	<b>8</b>
4.1	仮想化, クラウドで何が変わった?	8
4.2	生き残り、成長する方法	8
4.3	問題解決への取り組み	9
4.4	自動化の中身を見てみると	9
<b>5</b>	<b>Opscode Chef について</b>	<b>10</b>
5.1	Chef Server と Client & Node の関係	10
5.2	Chef の利用メリット	11
5.3	重要コンセプトと単語	11
5.4	それで！ Chef を一言で？!	15
<b>6</b>	<b>ハンズオン</b>	<b>17</b>
6.1	Hosted Chef Server へのユーザー登録	17
6.2	Workstation の設定	22
6.3	knife-euca のインストールと設定	25
6.4	Chef Node を操作	27
6.5	recipe の基本	29
6.6	recipe で motd を操作してみる	30
6.7	Attributes の検索の結果を motd に反映する recipe の作成	32
6.8	Data bag を使った Attribute の設定	34
<b>7</b>	<b>まとめ</b>	<b>38</b>

8 資料作成責任者	39
8.1 注意書き . . . . .	39

---

# はじめに

---

このドキュメントは、Hosted Chef Server(Opscode Chef SaaS) と knife plugin を利用し、パブリッククラウドにインスタンスの起動し、Chef Server 管理下への登録し、Cookbook & Recipe にて管理する基本的な手順を説明したものです。

---

# 研修について

---

## 2.1 トレーニング時間と注意事項

- 時間 : 13:00 ~ 17:00
- 休憩 : 随時、顔を見て判断していきます。
- トイレ/喫煙 : 休憩中にお願いします。(やむを得ない場合は静かに退出して下さい。)
- 質問事項 : 随時質問してください。Q&A タイムまで待つ必要は有りません。

---

ノート:

本日の資料は、unix 系の操作コマンドベースに記述しています。

従って参加者は、windowsPC のソフトで unix のコマンドと同等の操作を理解している必要が有ります。

---

## 2.2 本日のインストラクタ

講師: 堀田直孝

略歴: 学生時代を海外で過ごし、卒業後大手自動車会社、食品流通業、新規事業企画会社、CMS(Plone) 開発会社を経て、クリエーションライン株式会社に至る。

1997 に世界で最初のインターネットによる日食中継に参加し、これから起きる世界の変革をいち早く体験する。以降、IT 業界でノンキャリアとして下積み生活を過ごす。しかしながら 2000 年代には Python 関連の書籍 2 冊に執筆参加、世界で唯一の FFmpeg 本の執筆にも参加する。

仮想化や Cloud と一通り新しいものに興味を持ち、AWS の region が日本にできる前から JAWS のコアなメンバーとして活動していた。IaaS が普及するにつれて、それが常に供給者側の理論で議論されていることに疑問を持つようになり、ユーザーの視点で Cloud の本当の意義を考えるようになる。この頃から、積極的に海外の Cloud 系イベントに参加するようになり、DevOps や Chef 等の意義を直接利用者から聞くようになる。

2012 年には GMO と協同で DevOps Day Tokyo イベントを開催し、ユーザー会として Chef の無料ハンズオンを開催する等、「ユーザー視点で Cloud を有効活用し、ビジネスの効率化を向上

する」ためにはどうしたらいいかを、企業と一緒に考えるためのコンサルタント活動をしている。

## 2.3 トレーニングの目標

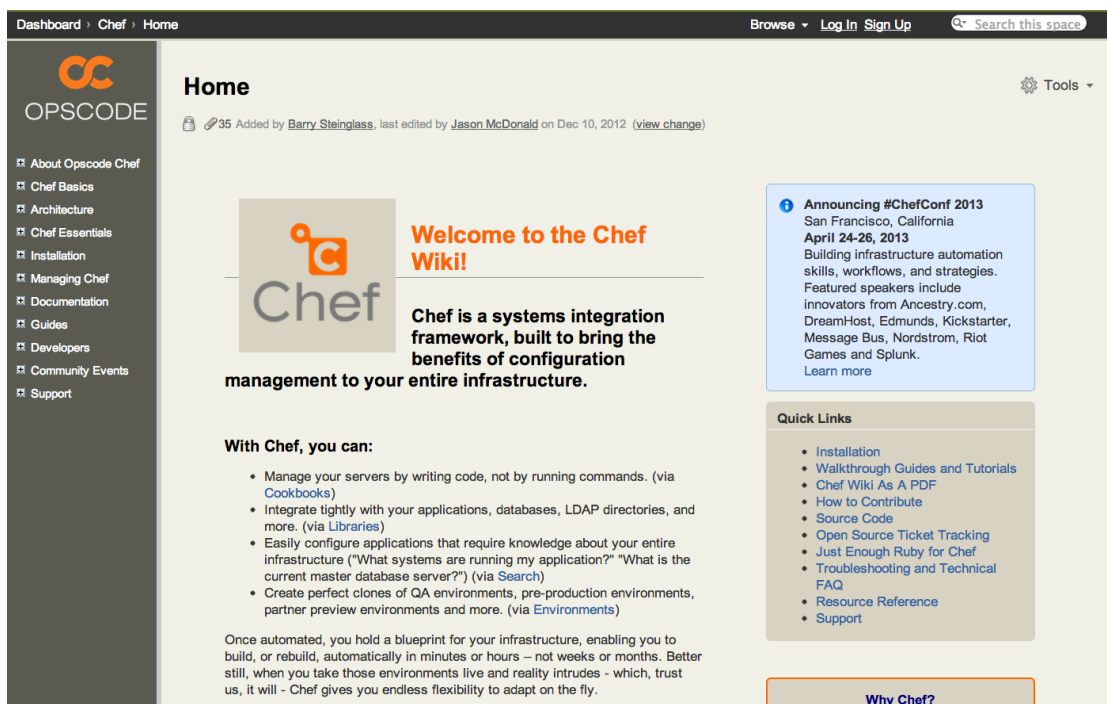
- 学術クラウドに対して Chef を使ってインスタンスの設定を試みる。
- 作業を通して、下記のような点を体感する。
  - cookbook を書くことにより、インフラ設定時の共通作業を自動化できる。
  - Chef フレームワークの構造と動作原理を理解する。
  - Chef フレームワークに含まれているツール及び関連したツールに触れてみる。
  - 今、抱えている問題を Chef フレームワークの活用で、どのように解決できるか感じとる。

## 2.4 Chef を学習する方法

### 重要

- とにかく、「触ること」、「試すこと」
- 80 %は、「慣れ」。このハンズオン習得
- 20%は、自分で色々試すして確認

### 2.4.1 更に！



- Opscode の wiki で必要な情報を得られる場所を理解する。

( <http://wiki.opscode.com/display/chef/Home> )

**Opscode**  
RULE THE CLOUD

Products | Cookbooks | Wiki | Community | Support | Search | Docs »

## Chef Documentation

**Note**

This page is a work in progress because Opscode is in the process of moving content from [wiki.opscode.com](http://wiki.opscode.com) to here. This set of topics is a collection of all documentation for Chef, including Hosted Chef, Private Chef, and Open Source Chef.

Chef is a systems and cloud infrastructure automation framework that makes it easy to deploy servers and applications to any physical, virtual, or cloud location, no matter the size of the infrastructure. Chef relies on abstract definitions (known as cookbooks and recipes) that are written in Ruby and are managed like source code. Each definition describes how a specific part of your infrastructure should be built and managed. Chef then applies those definitions to servers and applications, as specified, resulting in a fully automated infrastructure. When a new server comes online, the only thing that Chef needs to know is which cookbooks and recipes to apply.

- **Getting Started:** An Overview of Chef | Just Enough Ruby for Chef | Install Chef on a Workstation
- **References:** About Resources and Providers | About Knife | About Knife Plug-ins | About Lightweight Resources | Cookbooks Site API | Chef Server API | the chef-client `executable` | About Configuration Files | Recipe DSL
- **Nodes:** Nodes | About Ohai | About Exception and Report Handlers | The Chef Run | chef-client
- **Workstations:** Workstations | About the Repository | Knife
- **The Chef Server:** Chef Server | About Node Objects | About Roles | About Data Bags | About Environments | About Search
- **Cookbooks:** About Cookbooks | Directory Structure | About Attribute Files | About Definitions | About Files | About Libraries | About Lightweight Resources and Providers | About Cookbook Metadata | About Recipes | About Resources and Providers | About Templates | About Versions
- **Manage Chef:** Manage the Hosted Chef Server

Products | Cookbooks | Wiki | Community | Support | Search | Docs »

**Chef**

**Next topic**

An Overview of Chef

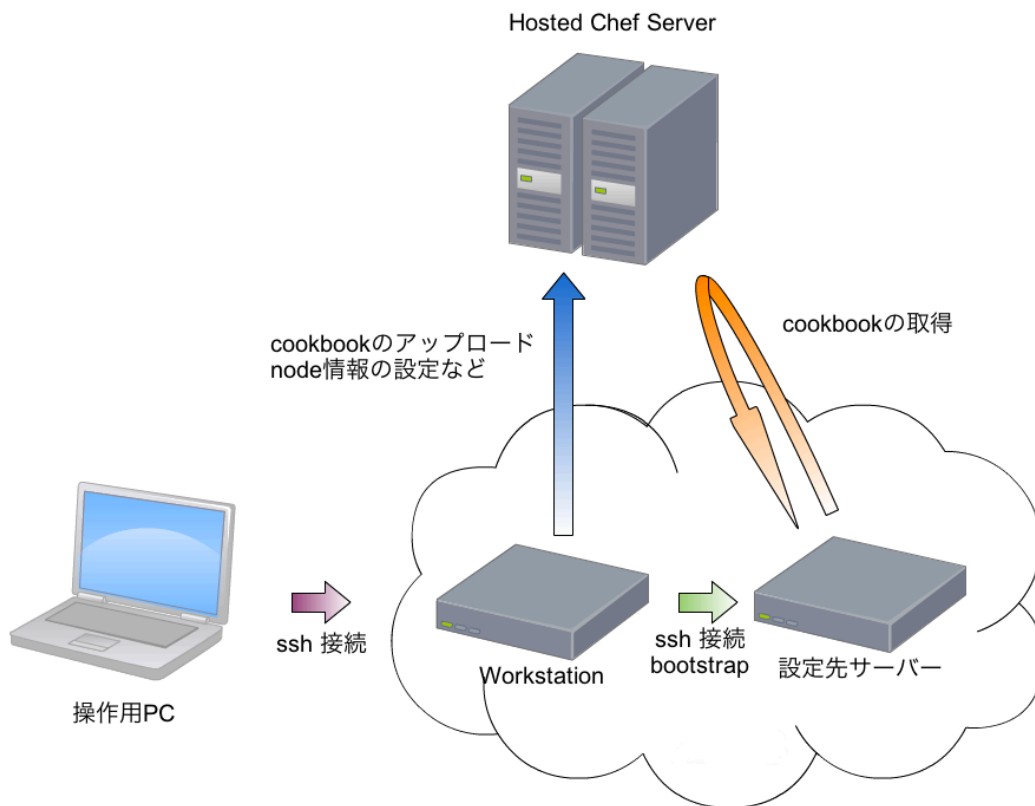
next

This work is licensed under a Creative Commons Attribution 3.0 Unported License.

- 最近、Opscode の docs サイトができた。

( <http://docs.opscode.com> )

## 2.4.2 Chef を実行する環境構成



クラウド上に、「Chefのワークステーション」と「実際に設定する node」を別々に起動します。Workstationとして設定したインスタンスに ssh でアクセスし全ての操作を行っていくことにします。



---

# Agenda

---

## 3.1 研修のアジェンダ

- 仮想化やクラウド化で何が変わった？
- 問題解決への取り組み
- 自動化の中身を見てみると
- **\*\*Chef Server と Client & Node の関係\***
- **Opscode Chef** 利用のメリット
- 重要コンセプトと単語
- それで！ **Chef** を一言で？!
- **Hosted Chef Server** へのユーザー登録
- **Workstation** の設定
- **knife-euca** のインストールと設定
- **Chef Node** を操作
- **recipe** の基本
- **recipe** で **motd** を操作してみる
- **Attributes** の検索の結果を **motd** に反映する **recipe** の作成
- **Data bag** を使った **Attribute** の設定
- **node object** の内容の検索
- **node object** の編集と **run\_list** の調整
- べき等性の確認
- **Q&A time**

---

# Server Configuration Toolの必要性

---

## 4.1 仮想化, クラウドで何が変わった？

### Pro

- 計算能力の調達が楽になった
- 1 台のマシンの利用効率が上がった

### Con

- より迅速な対応が求められるようになった
- 仕様が異なり、逆に手間が増えた
- 仕様が異なり、移行が難しくなった

### 重要

- クラウドコンピューティングは供給側のご都合的理論ベース
- 受給者視点での、クラウド利用効率化を見つけ出すことが必須

## 4.2 生き残り、成長する方法

### 複雑性を管理する

- 複雑かつ巨大なインフラストラクチャを一度に構築し、再構築にも対応できるようにする
- 例外や予期せぬイベントをモニタリングする

### 変化を加速させる

- アプリケーションやインフラストラクチャに対するアップデートを週毎または月毎ではなく、時間単位で実行できるようにする
- 「火事場対応」を止め、事前に備える
- 手動の作業プロセスを最小化し、それに費やす時間を減らす

### 生産性を向上する

- 日常タスクに費やす時間を減らし、不具合発生時に修復に要する時間を最小化する

- ユーザが安心して利用できる環境を用意する
- 人が戦うのではなく、ツールが人の代わりに戦うようにする

## 4.3 問題解決への取り組み

### 重要

インフラのコード化による、徹底した自動化

- マシンと人間の役割分担
- 設定作業の品質安定化
- 設定の抽象化

## 4.4 自動化の中身を見てみると

### 環境構築の自動化

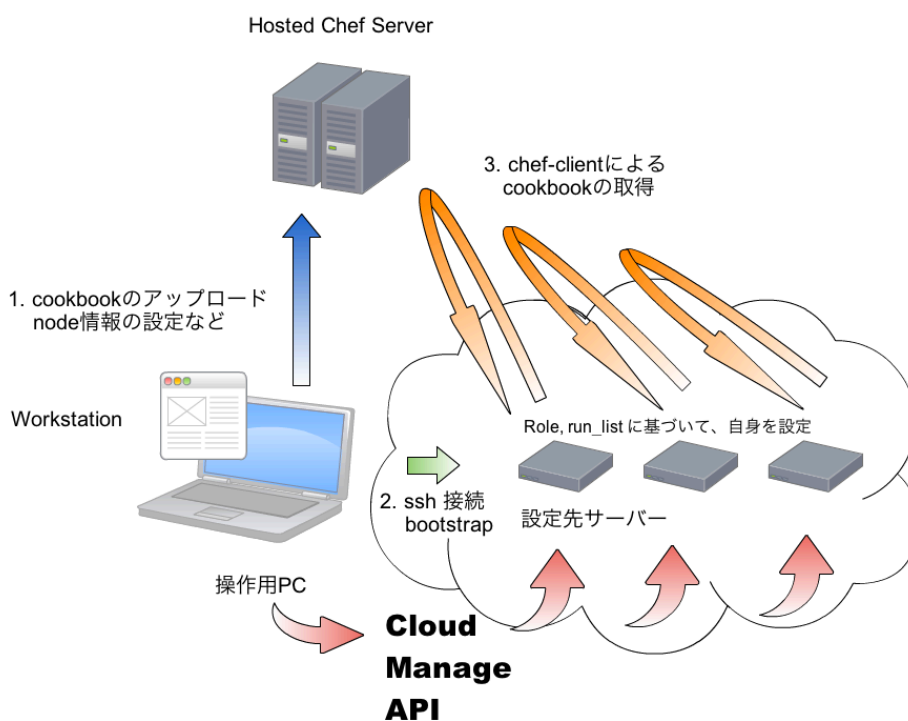
- 数行のコマンドで全体システム構築を自動化

### 運用環境の自動化

- 障害対応の自動化
- リソース状況の変化に関わる自動化 (Auto Scale)
- 環境移行の自動化
- ルーティンワークの自動化

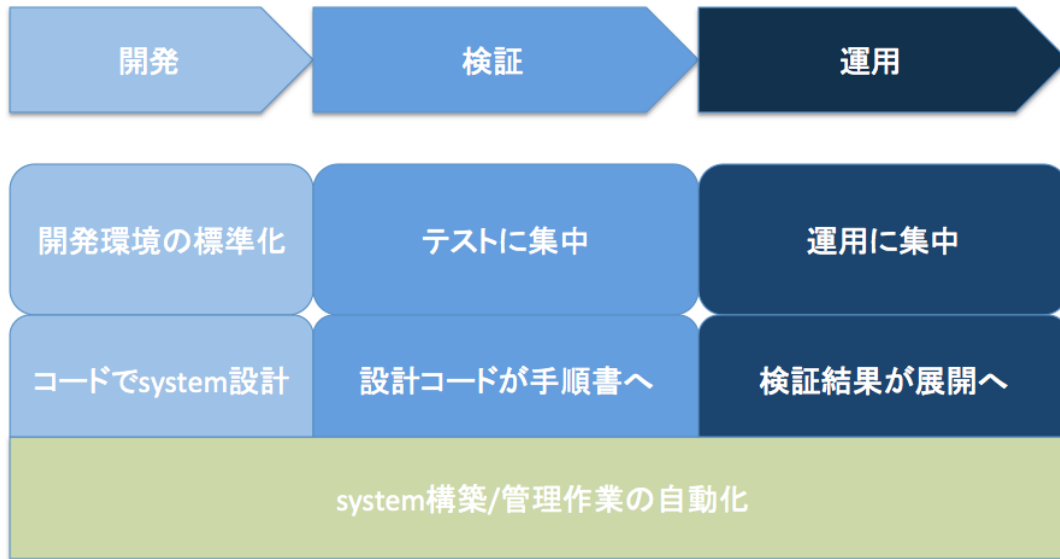
# Opscode Chef について

## 5.1 Chef Server と Client & Node の関係



- clients = nodes + workstaritons
- workstation には、knife とその機能拡張の rubyCLI がインストールされる

## 5.2 Chef の利用メリット



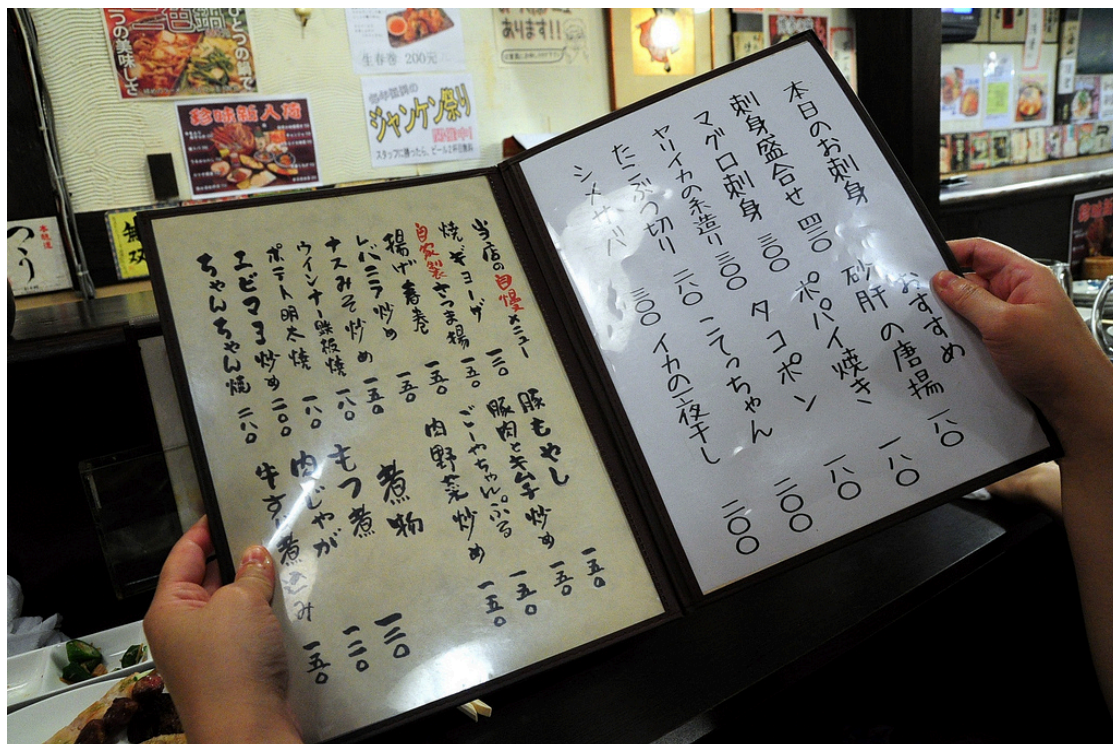
開発、検証、運用の各段階で:

- 同じ cookbook を利用することによって、環境構築作業の品質が安定する。
- 本来の業務に時間をかけることができ、各業務の品質が向上する。

## 5.3 重要コンセプトと単語

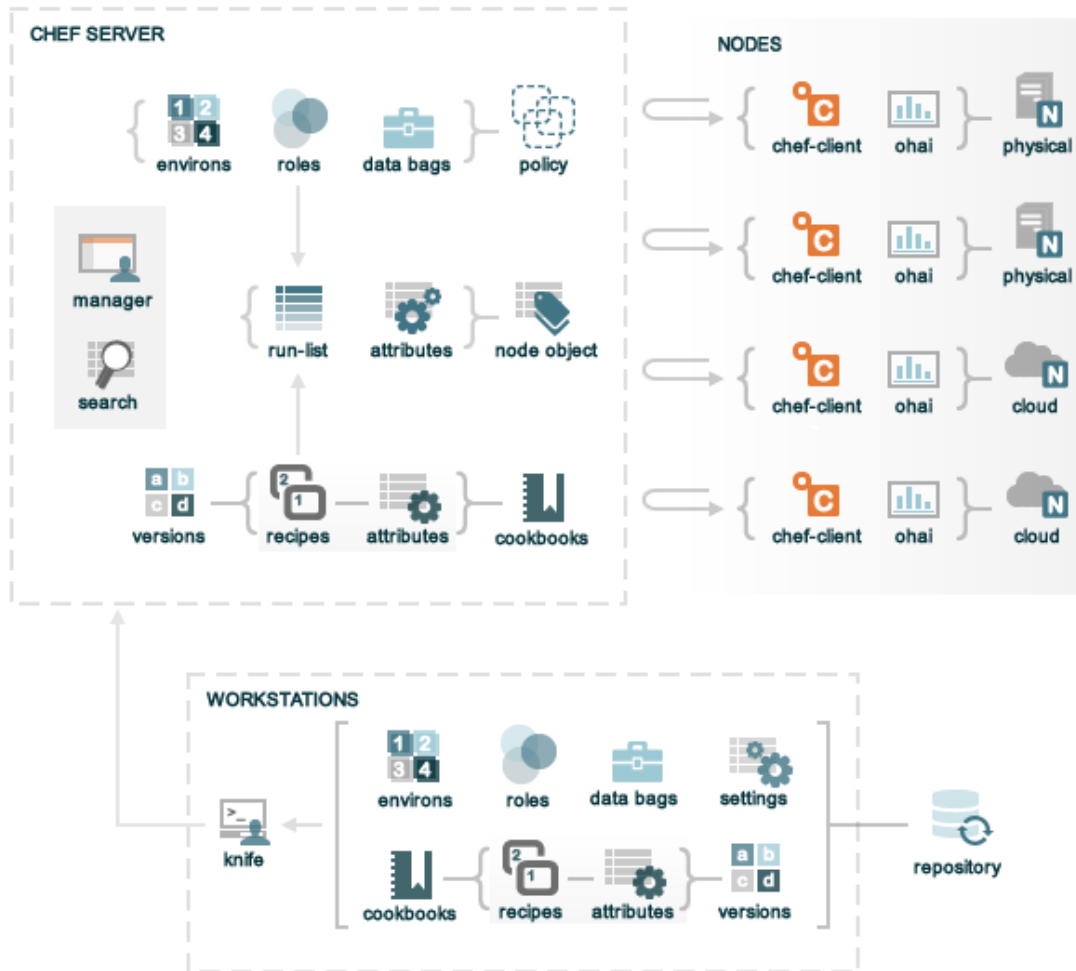
重要コンセプト:

- **べき等性**
- 大雑把に言って、ある操作を 1 回行っても複数回行っても結果が同じという概念



「注文した Beer が未だ届いていません」と店員さんに伝えた直後に、伝えた店員によって Beer が 1 杯届き。更に、しばらくすると別の店員が Beer をもう 1 杯持ってくる。こんな店には、べき等性がない。

### 5.3.1 全体像



### 5.3.2 Recipe

- 実際の設定手順が記載されたファイル
- このファイルに記述された順番通りに実行される
- Ruby の internal DSL で記述されている
- Ruby シンタックスで記述されている
- この recipe は他の recipe から参照できる
- node.object を通して、設定ターゲット node の attributes(情報)を読み込むことができる

### 5.3.3 Resource

- Chef のアクションを実行するユニット

```
package "debian-archive-keyring" do
  action :install
  options "--force-yes"
end

cookbook_file "/tmp/testfile" do
  source "testfile" # this is the value that would be inferred from the path parameter
  mode "0644"
end
```

- node の状態が別々にテストされ、アクションが実行される単位
- resource に含まれる項目

<b>Cook-book File</b>	Cron	De-ploy	Directory	Env	Erlang Call	<b>Exe-cute</b>
File	Git	Group	HTTP Request	Ifconfig	Link	Log
Mdadm	Mount	Ohai	<b>Package</b>	PowerShell Script	Remote Directory	Remote File
Route	Ruby Block	SCM	Script	Service	Subversion	<b>Tem-plate</b>
User	Opscode Cookbook LWRPs					

### 5.3.4 Cookbook

- recipe をパッケージ化したもので、apache2 のように複数の recipe を含む場合もある
- 一般的に一つのソフト (apache,mysql 等) や一つの機能をパッケージ化している

### 5.3.5 Roles

- Node を設定する際に、類似している要素 (feature) のグループ
- Role は、cookbook や recipe と Attribute の集合
- Node は複数の Role を持つことができる
- Chef-client の実行時に複数の Role と recipe は展開され、一つの run\_list にマージされる

### 5.3.6 Run List

- Node で実行される recipe のリスト
- knife, Role, 等の場所で設定することができる

```
run_list "recipe[apache2]", "recipe[apache2::mod_ssl]", "role[monitor]"
```

### 5.3.7 Attribute

- Chef Server の保存された Node を設定するための変数 data
- Chef Server 上で検索できる



- Chef Server を使った Node の運用時には、状況に応じて動的に変更できる

attribute を cookbook の中で設定する場合は、cookbook ディレクトリの中に attribute というディレクトリを作成し、値を記述したファイルを作成する。

例として cookbooks/apache2/attributes/default.rb に下記のような attribute を設定し、それを http.conf を生成するための template と合成し、状況に応じて最終設定ファイルを出力する。

```
default["apache"]["dir"]           = "/etc/apache2"
default["apache"]["listen_ports"] = [ "80", "443" ]
```

Ohai により自動収集される attribute:

- IP address
- hostname
- CPU, HDD, memory, partition
- インストールした kernel modules
- 導入されてプログラミング言語, version
- その他、大量の情報

### 5.3.8 Node Object

- Chef Server に保存されている、各 Node に関連する情報レコード
- それぞれの node に関わる情報は、node.object を介して参照することができる
- node.object に含まれる項目を、Attribute と呼ぶ
- この Attribute の参照とテンプレートへの書き込みが、Chef の柔軟な設定能力を強化している

### 5.3.9 Node

- Chef Client が実行されている場所
- Attribute と run list という要素を保持したレコード行
- Recipes と [Roles] を適用する先

## 5.4 それで！ Chef を一言で？！

重要：

開発者およびシステムエンジニアが、インフラストラクチャを、継続的に「定義、構築、管理」するための自動化プラットフォーム。

- インフラストラクチャを記述するための新たな手法
- 一般的なプログラム同様にモジュール化構造を採用することによって、インフラの高度な再利用が可能
- 検索可能なインフラおよび、その設計図を同時に実現

### 5.4.1 噛み砕いていうと:

---

重要:

1. モデルを基にして
  2. サーバーの設定を自動化し
  3. その状態を、保持し続ける
  4. フレームワーク
- 

### 5.4.2 間違った理解:

警告:

- (X) 既に数世代のエンジニアに渡って、コンソール経由で管理してきたシステムの復元や再現もできる
- (X) モデルで定義されているファイル以外も監視 / 管理している
- (X) サーバーしか設定できない
- (X) chef-solo だけで充分

---

# ハンズオン

---

## 6.1 Hosted Chef Server へのユーザー登録

以降の手順に基づいて、下記のファイルを取得します。

- Chef server への 認証ファイル (pem ファイル 2 種類)
- knife を使うための 環境設定ファイル (knife.rb)


### 6.1.1 Hosted Chef を利用するユーザーを登録

---

ノート: [Hosted Chef Sign UP Opscode](#) サイトの一番上の分の黒い帯の中にあります。

---

Opscode のサイトから [Hosted Chef](#) のユーザー登録画面にたどり着くには複数の方法がありますが、下記のページから登録を開始できます。



OPSCODE  
RULE THE CLOUD

[Customer Login](#) | [Sign Up](#) | [Account Management](#) | [Recover Password](#)

[Products & Services](#) | [Solutions](#) | [Customers](#) | [Support](#) | [About](#) | [Newsroom](#) | [Community](#)

[Home](#) | [Products](#) | [Hosted Chef](#) | [Private Chef](#) | [Chef](#)

### Plans & Pricing

	Launch	Standard	Premium
Monthly Fees	\$120	\$300	\$600
Nodes	20	50	100
Users	10	20	50
Standard Support	Included	Included	Included
Onsite Training	Not available	Not available	Available

[Buy Now >](#)
[Buy Now >](#)
[Buy Now >](#)

[Need more nodes? Contact us about Hosted Chef for the Enterprise](#)

**Hosted Chef is free for 5 nodes or less!**

Questions about Hosted Chef pricing and support? Check the [FAQ](#)

[Free Trial >](#)

### What you get


- Faster**  
Spend your time fixing your problems, not setting up your automation.
- Bigger**  
Opscode has your capacity needs covered - from 5 to 50,000 servers.
- Smarter**  
Infrastructure awareness provides automatic discovery of thousands of valuable data-points.
- Safer**  
Centrally managed role-based access controls and data-encryption.
- Real**  
You hold the blueprint for your infrastructure - rebuild it where, and when, you need.
- Supported**  
Managed API updates make upgrades a breeze. An industry-leading SLA and 24x7x365 support options give you the confidence to run mission-critical systems.

“About you” の部分に必要な事項を記入して、 **agree** にチェックマークを入れ、”Next>” をクリックします。

## Hosted Chef

Instant access to a highly available, dynamically scalable, fully managed and supported automation environment

Opscode's Hosted Chef combines the freedom and flexibility of Chef with the reliability and speed of Opscode. No matter how complex the realities of your business, Hosted Chef makes it easy to deploy servers and scale applications throughout your entire infrastructure. Because it combines the fundamental elements of configuration management and service oriented architectures with the full power of Ruby, Hosted Chef makes it easy to create a fully automated infrastructure.

[Download the fact sheet](#) 

### Plans & Pricing

	Trial
Monthly Fees	\$0
Nodes	5
Users	5
Standard Support	Included
Onsite Training	Not available

Free Trial >

### About you

1 2 3

First Name \*

Middle Name

Last Name \*

Username \*

Email Address \*

Password \*


Confirm Password \*


Organization Name \*

Organization Short Name \*

Phone Number \*

Company \*

Country \* 

State \* 

☐ I agree to the [Terms of Service](#), [Opscode Platform Customer Agreement](#), and [Opscode Service Level Agreement](#).

< Back

Next >

## 6.1.2 Hosted Chef Server に登録したユーザー認証用の pem ファイルを取得

**Account Management**

[My Profile](#) [Change Password](#) [View/Edit Plan](#) [Billing](#)

### Change Your Password

Current Password: \*

New Password: \*

Confirm Password: \*

**Change Password**

**Public Profile**

View your user profile on the Opscode Community Site.

[View Profile](#)

**Need Help?**

If you have questions or you're stuck, we're here to help.

[Get Help](#)

### Reset User Key

If you've lost your private key, or would like to replace it, click the button below.

When you get a new key, **your old key will stop working.**

This private key **replaces your old key.**

**We do not keep a copy** so please store it somewhere safe.

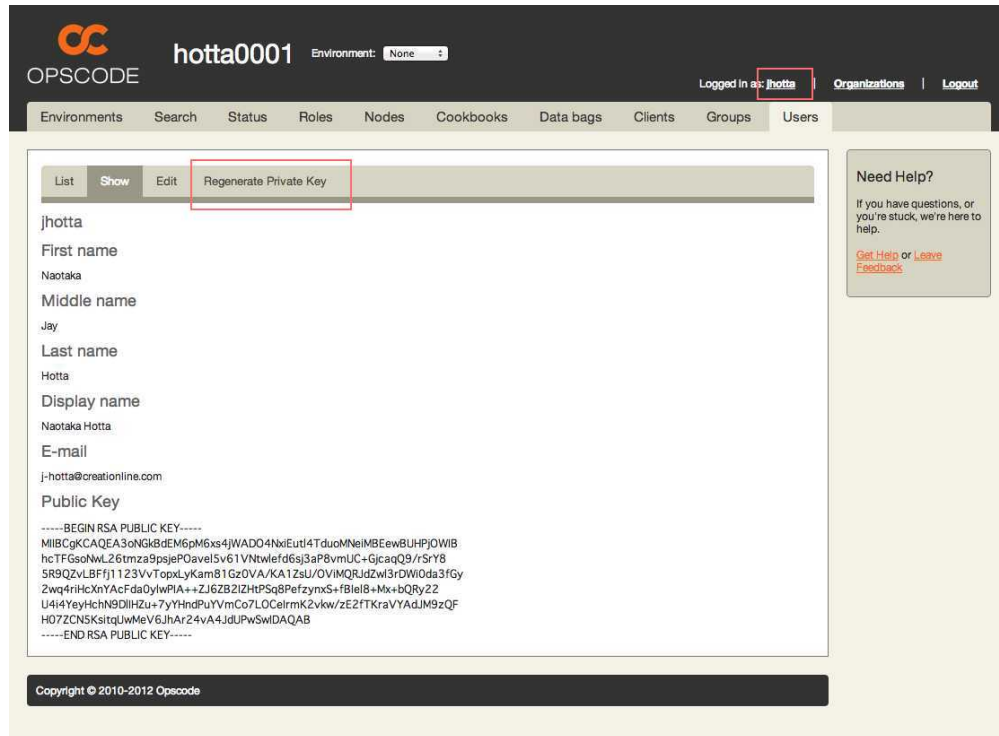
**Get a new key**

---

ノート: 画面右上の黒い帯の中の "logged in as: " と表示されている部分をクリックし Account Management 画面が表示されたら、"Change Password" をクリックします。

---

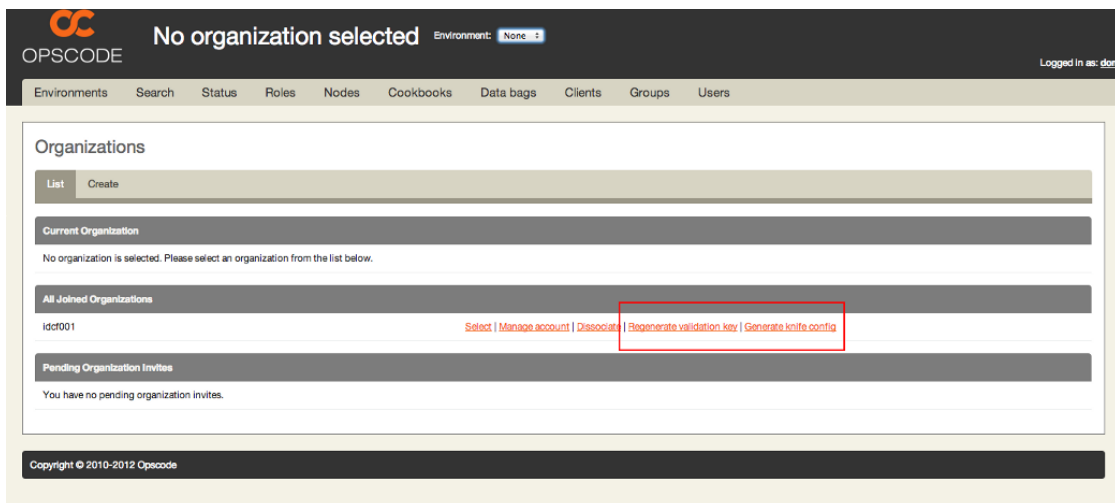
ノート: 上記のタイミングで pem ファイルを取得できなかった場合は、下記の手順で新しく pem ファイルをジェネレートします。



### 6.1.3 Organization(組織)の認証用の pem ファイルと knife の環境設定用ファイルを取得

右上の Organization(組織) をクリックします。その後、赤枠の部分にある項目をクリックしファイルをダウンロードします。

- Regenerate validation key
- Generate knife config



## 6.2 Workstation の設定

警告: 使用するソフトウェアの操作方法に置き換えて作業を進めてください。ssh に関しては、tera term scp に関しては、win scp

### 6.2.1 起動したインスタンスに ssh で接続

クライアントソフトから、先に起動したインスタンスの IP アドレスを確認しておきます。

---

ノート:

- security group は、ossforum-chef を利用します。
- 

確認した IP アドレスに対し、ssh で接続します。Unix 系の OS の場合、下記のようになります。

```
$ ssh ubuntu@[インスタンス IP]
```

---

ノート: password は、ossforum に設定されています。

---

下記のような内容が表示されるはずです。

```
Welcome to Ubuntu 12.04.1 LTS (GNU/Linux 3.2.0-32-virtual x86_64)

* Documentation:  https://help.ubuntu.com/

System information as of Tue Oct 23 03:28:23 UTC 2012

System load:  0.0                Processes:            59
Usage of /:   10.5% of 7.87GB     Users logged in:     0
Memory usage: 3%                IP address for eth0: 10.162.15.62
Swap usage:   0%

Graph this data and manage this system at https://landscape.canonical.com/

0 packages can be updated.
0 updates are security updates.

Get cloud support with Ubuntu Advantage Cloud Guest
http://www.ubuntu.com/business/services/cloud
```

### 6.2.2 Chef 関連のソフトウェアを Omnibus installer でインストール

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

下記のような出力が出れば、インストール完了です。

```
Setting up chef (10.16.0-1.ubuntu.11.04) ...
Thank you for installing Chef!
```



---

ノート:

- **Omnibus Installer** では、Ubuntu 以外の OS にもインストールすることができます。
  - **Omnibus Installer** でインストールした場合は、Chef 関連の全てのソフトウェアは/opt/chef にまとめてインストールされます。
  - Chef 関連のソフトウェアが不要になった場合、/opt/chef 以下を削除してください。
- 

### 6.2.3 Chef 操作のために必要なファイルを保存しておくディレクトリを作成

---

ノート: 既に git がインストールされているインスタンスを利用する場合は、下記の apt-get による git のインストールは不要です。不明の場合、そのまま実行しても問題ありません。

```
$ sudo apt-get install git-core
```

---

```
$ sudo apt-get install git-core
$ git clone https://github.com/opscode/chef-repo.git
```

下記のように、ファイル構成のクローニングできれば完成です。

```
Cloning into 'chef-repo'...
remote: Counting objects: 199, done.
remote: Compressing objects: 100% (117/117), done.
remote: Total 199 (delta 72), reused 160 (delta 49)
Receiving objects: 100% (199/199), 30.34 KiB, done.
Resolving deltas: 100% (72/72), done.
```

下記のようなディレクトリが存在していることを確認してください。

```
.
+-- chef-repo
|   +-- certificates
|   |   +-- README.md
|   +-- cheffignore
|   +-- config
|   |   +-- rake.rb
|   +-- cookbooks
|   |   +-- README.md
|   +-- data_bags
|   |   +-- README.md
|   +-- environments
|   |   +-- README.md
|   +-- Rakefile
|   +-- README.md
|   +-- roles
|       +-- README.md
```

## 6.2.4 Chef Server に Workstation として登録するために必要なファイルを設置するディレクトリを作成

```
$ cd chef-repo
$ mkdir .chef
```

ここまできたら、EC2 のインスタンスからログアウトします。

```
$ exit
```

## 6.2.5 インスタンスに対して Workstation として必要なファイルを転送

Hosted Chef のユーザー登録時に取得した、認証用 pem ファイルと knife の環境設定ファイルをインスタンスに転送します。

- ユーザーの認証用 pem ファイル
- Organization(組織) の認証用 pem ファイル
- Organization(組織) の knife.rb

```
$ scp -i hosted_chef_user.pem \
    ubuntu@[インスタンス IP アドレス]:chef-repo/.chef/

$ scp -i organization-validator.pem \
    ubuntu@[インスタンス IP アドレス]:chef-repo/.chef/

$ scp -i knife.rb \
    ubuntu@[インスタンス IP アドレス]:chef-repo/.chef/
```

---

ノート: winscp を利用して、同様の内容を得るための操作をしてください。

---

Workstation として動作しているかどうか確認します。先ほどと同じように ssh でインスタンスにアクセスして下記のコマンドを実行します。

```
$ cd chef-repo
$ knife client list
```

knife.rb のファイルに記載されている "validation\_client\_name" の値が表示されていることを確認します。

## 6.2.6 Site 専用の cookbooks の設置ディレクトリの設定

Workstation に ssh でアクセスし、knife.rb を編集します。

```
$ ssh root@[workstation のインスタンス IP]
$ vi ~/chef-repo/.chef/knife.rb
```

cookbook\_path に site 専用の cookbook の設置場所を追記します。

```
cookbook_path ["#{current_dir}/../cookbooks", "#{current_dir}/../site-cookbooks"]
```

## 6.3 knife-euca のインストールと設定

### 6.3.1 gem を使って、knife-euca をインストール

---

ノート: Omnibus Installer を使った Chef のインストールでは、全てのスクリプトは /opt/chef 以下に配置されます。

---

Ruby の gem 形式で配布されている、knife の拡張プラグインをコンパイルするためのヘッダーファイルとビルド環境をインストールします。

```
$ sudo apt-get install libxml2-dev libxslt-dev build-essential
```

準備が整ったら、/opt/chef/embedded/bin/gem を使って、プラグインをインストールします。

```
$ sudo /opt/chef/embedded/bin/gem install knife-eucalyptus \
  --no-ri \
  --no-rdoc
```

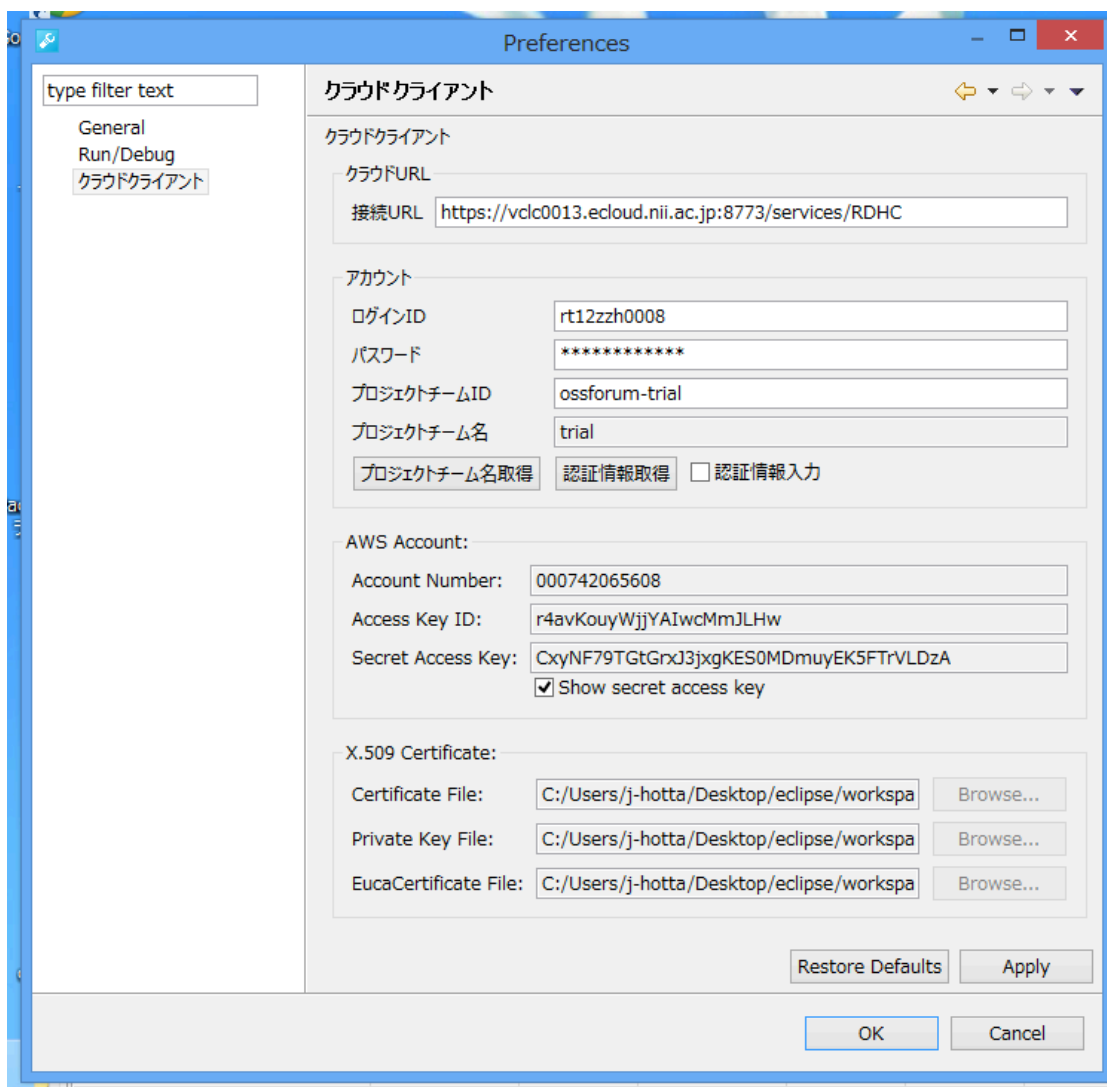
下記のようなコメントが出力されれば、インストールは完了です。

```
Gem files will remain installed in /opt/chef/embedded/lib/ruby/gems/1.9.1/gems/nokogiri-1.5.5 fo
Results logged to /opt/chef/embedded/lib/ruby/gems/1.9.1/gems/nokogiri-1.5.5/ext/nokogiri/gem_ma
```

### 6.3.2 クラウドを使うための設定を knife.rb に追記

---

ノート: クラウドクライアントの [設定]>[クラウドクライアント] を選択し、情報を確認してください。



Workstation に ssh でアクセスし、knife.rb を編集します。

```
$ ssh root@[workstation のインスタンス IP]
$ vi chef-repo/.chef/knife.rb
```

3 項目を追記します。

```
# for NII Eucalyptus

knife[:euca_access_key_id]      = "r4avKouyWjjYAIwcMmJLHw"

knife[:euca_secret_access_key] = "CxyNF79TGtGrxJ3jxgKES0MDmuyEK5FTrVLDzA"

knife[:euca_api_endpoint] = "http://vclc0013.ecloud.nii.ac.jp:8773/services/Eucalyptus"
```

ノート:

クラウドクライアントから下記うつ際の注意点！

<https://> -> <http://> (オレオレ認証証明書への対策)

RDHC -> Eucalyptus (独自認証用のサイトをバイパスする対策)

knife-eucalyptus が正しく設定できているか確認してみます。

```
$ ssh root@[workstation のインスタンス IP]
$ cd ~/chef-repo
$ knife euca flavor list
$ knife euca server list
```

Workstation として起動したインスタンス情報が表示されれば設定は完了しています。

```
ubuntu@localhost:~/chef-repo$ knife euca flavor list
ID            Architecture  RAM           Disk           Cores
c1.medium     32-bit       1740.8 MB     350 GB         5
c1.xlarge     64-bit       7168 MB      1690 GB        20
cc1.4xlarge   64-bit      23552 MB     1690 GB       33.5
cc2.8xlarge   64-bit      61952 MB     3370 GB        88
cg1.4xlarge   64-bit      22528 MB     1690 GB       33.5
hi1.4xlarge   64-bit      61952 MB     2048 GB        35
m1.large      64-bit       7680 MB      850 GB          4
m1.medium     32-bit       3750 MB      400 GB          2
m1.small      32-bit      1740.8 MB    160 GB           1
m1.xlarge     64-bit      15360 MB     1690 GB          8
m2.2xlarge    64-bit     35020.8 MB   850 GB          13
m2.4xlarge    64-bit     70041.6 MB  1690 GB          26
m2.xlarge     64-bit     17510.4 MB   420 GB          6.5
t1.micro      0-bit        613 MB       0 GB             2

ubuntu@localhost:~/chef-repo$

ubuntu@localhost:~/chef-repo$ knife euca server list
Instance ID   Public DNS Name  Flavor      Image          Security Groups  State
i-458D07D3    157.1.150.10    c1.medium   emi-19F41435   [sg-xxxxxx]      running
i-31BE06D9    157.1.150.14    c1.medium   emi-1D58145A   [sg-xxxxxx]      running
i-40BA07A6    157.1.150.13    c1.medium   emi-1C55144B   [sg-xxxxxx]      running
i-392407BF    157.1.150.11    m1.xlarge   emi-1C55144B   [sg-xxxxxx]      running
i-4028086C    157.1.150.15    c1.medium   emi-1C55144B   [sg-xxxxxx]      running

ubuntu@localhost:~/chef-repo$
```

## 6.4 Chef Node を操作

\*\* EUCA COMMANDS \*\*

- knife euca flavor list (options)
- knife euca image list (options)
- knife euca server create (options)
- knife euca server list (options)
- knife euca server delete SERVER\_ID [SERVER\_ID] (options)

### 6.4.1 クラウド内で起動できるインスタンスの情報を収集

```
$ cd /root/chef-repo
$ knife euca flavor list
```

### 6.4.2 edubase Cloud で起動したインスタンスに、Chef-client を組み込み、Chef Server に登録

```
$ knife cookbook site install chef-client
$ knife cookbook upload chef-client
```

実際に cookbook が登録できているか確認してみます。

```
$ knife cookbook list
```

cookbook の名前と、バージョンを確認します。(今回は、下記のように表示される)

```
chef-client 2.0.2
```

### 6.4.3 収集した情報に基づいてインスタンスを起動

```
$ knife euca server create \
  -r "recipe[chef-client]" \
  -I emi-1C55144B \
  --flavor c1.medium \
  -x ubuntu \
  -P ossforum \
  -Z clouster0 \
  -G ossforum-chef
  -N ossforum-user001 \
  -VV
```

それぞれの変数オプションの内容は以下の通り

knife euca server create (options)

-S	-ssh-key KEY	The Eucalyptus SSH key id
-i IDENTITY_FILE		The SSH identity file used for authentication
-r	-run-list RUN_LIST	Comma separated list of roles/recipes to apply
-I	-image IMAGE	The AMI for the server
-f	-flavor FLAVOR	The flavor of server (m1.small, m1.medium, etc)
-Z	-availability-zone ZONE	The Availability Zone
-G	-groups X,Y,Z	The security groups for this server; not allowed when
-x	-ssh-user USERNAME	The ssh username
-P	-ssh-password PASSWORD	The ssh password
-N	-node-name NAME	The Chef node name for your new node
-V	-verbose	More verbose output. Use twice for max verbosity
-d	-distro DISTRO	Bootstrap a distro using a template; default is 'chef-full'

---

ノート: 今回のコマンドは、Eucalyptus でインスタンス起動 + Chef client のインストール + node の登録を一気に実行してます。

---

上記のコマンドで chef-client インストールできない場合

```
$ knife bootstrap [IP アドレス] -N jay-server101 \  
-x ubuntu \  
-P ossforum \  
-r "recipe[chef-client]" \  
--sudo
```

---

ノート: knife に準備されている bootstrap スクリプト (-d で指定できるもの)

- archlinux-gems.erb
  - centos5-gems.erb
  - chef-full.erb
  - fedora13-gems.erb
  - ubuntu10.04-apt.erb
  - ubuntu10.04-gems.erb
  - ubuntu12.04-gems.erb
- 

インスタンスが既に起動している場合は、下記のコマンドで遠隔的に chef-client をインストールすることができます。

```
$ knife bootstrap IP_ADDRESS -x ubuntu \  
-P PASSWORD \  
--sudo  
  
$ knife bootstrap IP_ADDRESS -x ubuntu \  
-i ~/.ssh/id_rsa \  
--sudo
```

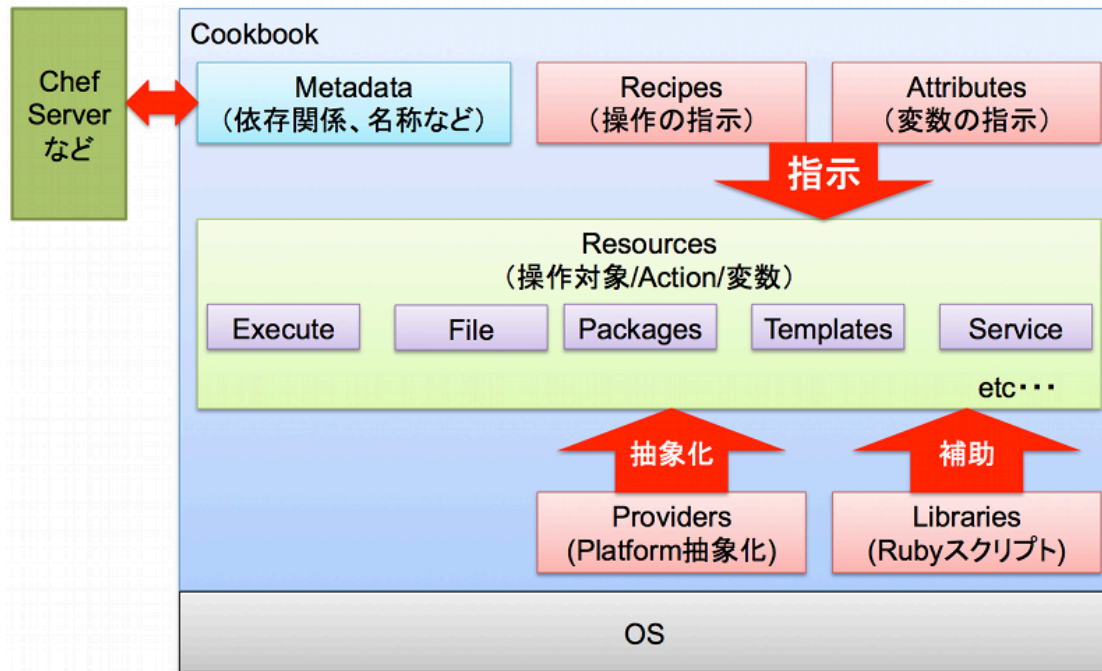
#### 6.4.4 インスタンスの削除と Node & Client の削除

```
$ knife euca server list  
  
$ knife euca server delete [削除したい instance id]  
  
$ knife node list  
  
$ knife node delete -y [削除した node]  
  
$ knife clinet list  
  
$ knife clinet delete -y [削除した clinet]
```

### 6.5 recipe の基本

- Recipe は、リソースを集めたもの

- Cookbooks は、recipes, templates, files, custom resources, etc
- Code のモジュール化と再利用



## 6.6 recipe で motd を操作してみる

### 6.6.1 motd の cookbook を site-cookbook ディレクトリに作成

ノート: 今回は、cookbook の名前を、j-motd と命名し作業を進めます。この名前は各自自由につけてください。

```
$ cd /root/chef-repo/
$ knife cookbook create j-motd -o site-cookbooks
```

j-motd cookbook のディレクトリに移動します。

```
$ cd ./site-cookbooks/j-motd
```

j-motd のディレクトリは、下記のような構成で作成されます。

```
.
+-- CHANGELOG.md
+-- README.md
+-- attributes
+-- definitions
+-- files
|   +-- default
|       +-- motd.tail
+-- libraries
+-- metadata.rb
+-- providers
+-- recipes
```





upload 出来ている cookbook が確認します。

```
$ knife cookbook list
```

出力結果：

```
chef-client 2.0.2
j-motd      0.1.0
```

先ほど、作成したインスタンスの cookbook を登録します。

```
$ knife node run_list add [node名] j-motd
```

ここでインスタンスに、ssh でログインしていきます。

```
$ ssh root@[インスタンス IP]
$ sudo chef-client
```

chef-client が実行されたことを確認できたら、一度 logout し、再度 login します。

```
$ ssh root@[インスタンス IP]
```

## 6.7 Attributes の検索の結果を motd に反映する recipe の作成

先ほど利用した j-motd にファイルを追加することで、改良を加えてきます。

```
$ cd /root/chef-repo
$ vi site-cookbooks/j-mod/recipe/use_attribute.rb
```

下記の内容を記述します。

```
# Cookbook Name:: j-motd
# Recipe:: use_attribute
#
# Copyright 2012, YOUR_COMPANY_NAME
#
# All rights reserved - Do Not Redistribute
#

template "/etc/motd.tail" do
  source "motd.tail.erb"
  mode "0644"
  variables({
    :x_men => node[:chef_handson]
  })
end
```

次に、`ruby` のテンプレートを記述します。

```
$ vi site-cookbooks/j-motd/templates/default/motd.tail.erb
```

以下の内容を記述していきます。

[illegible]

```
\_____/ |__| |_____/ \____| \_____/ |_____/ |_____|
```

Modified by jhotta@creationline. Inc.

```
YOU ARE LOGED IN AS: <%= node[:current_user] %>
This will be an extra attribute that show in the handson <%= @x_men %>
```

#### ノート:

- erb の中から、node.object に対して直接検索を書けることもきます。
- recipe の中で検索し、結果を variables で erb に引き渡すこともできます。

Chef Server に向けて、cookbook を再度 upload します。

```
$ knife cookbook upload j-motd
```

ここで、run\_list に変更を加えます。

```
$ knife node show jay-server211
$ knife node run_list remove jay-server211 "recipe[j-motd]"
$ knife node show jay-server211
$ knife node run_list add jay-server211 "recipe[j-motd::use_attribute]"
$ knife node show jay-server211
```

node.object を直接変更する場合は、下記の様に行います。

**警告:** node.object の手動での変更は、実際の運用では絶対にやめてください。

```
$ EDITOR=vi knife node edit [起動している node 名]
```

以下の様に、node.object を変更します。

```
{
  "name": "起動時につけた名前",
  "chef_environment": "_default",
  "normal": {
    "tags": [

    ],
    "chef_client": {
      "bin": "/usr/bin/chef-client"
    },
    "chef_handson": "表示したい名前"
  },
  "run_list": [
    "recipe[chef-client]",
    "recipe[j-motd::use_attribute]"
  ]
}
```

ターゲットの node で、chef-client を実行してみます。

```
$ ssh root@[インスタンス IP]
$ sudo chef-client
```

一度 exit し、node に再度 login すると下記の様に表示されます。

[illegible]

Modified by jhotta@creationline. Inc.

YOU ARE LOGED IN AS: ubuntu

This will be an extra attribute that show in the handson "表示したい名前"

## 6.8 Data bag を使った Attribute の設定

### 6.8.1 Data bag のカタログを作ります。

Chef server に handzon に関する Attribute を収納するためのカタログを作ります。

```
$ knife data bag create handzon
```

現在の中身を確認してみます。

```
$ knife data bag list
$ knife data bag show handzon
```

### 6.8.2 Data bag の中身のデータを作成します。

まずは、1つ目の json data を作成します。

```
$ cd ~/chef-repo/data_bag
$ mkdir default
$ vi data1.json
```

ファイルの中身は下記のようにになります。

```
{
  "id": "microsoft",
  "value": "Bill Gates"
}
```

2つ目の json data を作成します。

```
$ cp data1.json data2.json
$ vi data2.json
```

ファイルの中身は下記のようになります。

```
{
  "id": "apple",
  "value": "Steve Jobs"
}
```

Data bag に収納されている data を Chef server に登録します。

```
$ knife data bag from file handzon data_bags/default/data1.json
$ knife data bag from file handzon data_bags/default/data2.json
$ knife data bag show handzon
```

下記のように 2 項目も id が表示されます。

```
microsoft
apple
```

更に、Data bag に登録しているデータの中身を詳しく見ていきます。

```
$ knife data bag show handzon microsoft
```

下記の様に表示され内容が確認できます。

```
id:      microsoft
value:   Bill Gates
```

尚、console から data bag の中身を編集したい場合は下記のコマンドで編集します。

```
$ EDITOR=vi knife data bag edit handzon microsoft
```

---

ノート: Data bag に関するこれらの作業は、Gui を使っても同じことができます。

---

### 6.8.3 Data bag の中身を検索して、Apache のページの表示が変わる cookbook を作ります。

```
$ cd ~/chef-repo/
$ knife cookbook create mywebapp -o site-cookbooks
```

default.rb という recipe を編集します。

```
$ cd ~/chef-repo/site-cookbooks/mywebapp/recipes
$ vi default.rb
```

追記内容は、下記になります。

```
#
# Cookbook Name:: mywebapp
# Recipe:: default
#
# Copyright 2012, YOUR_COMPANY_NAME
#
# All rights reserved - Do Not Redistribute
#
```

```
w_stage = search('handzon', 'value:ap*').first

template "/var/www/index2.html" do
  source "index2.html.erb"
  mode "0644"
  variables({
    :x_men => data_bag_item('handzon', 'microsoft')['value'],
    :y_men => w_stage['value']
  })
end
```

index2.html.erb という template を追加します。

```
$ cd ~/chef-repo/site-cookbooks/mywebapp/templates/default
$ vi index2.html.erb
```

追記内容は、下記に成ります。

```
<html>
<head><title>Test Template</title>
<body>
<h1>
  YOU ARE LOGED IN AS: <%= node[:current_user] %>
</h1>
<h3>
  Usage of Data bag item search: <%= @x_men %>
</br>
  Usage of search of date bag: <%= @y_men %>
</h3>
</body>
</html>
```

Chef Server に向けて、cookbook を再度 upload します。

```
$ knife cookbook upload mywebapp
```

cookbook を取得し、Chef Server に upload します。

```
$ knife cookbook site install apache2
$ knife cookbook upload apache2
```

role を設定します。

```
$ cd ~/chef-repo/roles
$ vi mywebapp.rb
```

追記内容は、下記に成ります。

```
name "mywebapp"
description "my sample Web App"
run_list(
  "recipe[motd]",
  "recipe[apache2]",
  "recipe[mywebapp]"
)
```

role を Chef server に upload します。

```
$ knife role create mywebapp
```

role が登録されているか確認してみます。

```
$ knife role list
```

登録した role の内容を確認してみます。

```
$ knife role show mywebapp
```

ここで、run\_list の整理をしてみます。

```
$ knife node run_list remove "recipe[j-motd::use_attribute]"
$ knife node run_list add "role[mywebapp]"
```

ターゲットの node に、再度 login し、chef-client を実行してみます。

```
$ ssh root@[インスタンス IP]
$ sudo chef-client
```

一度 logout して、再度 login してみる。その後、browser を使って node の IP address の index2.html を表示してみる。

---

ノート: 早く終わっている場合は、index.html と index2.html を削除し、再度 chef-client を実行してみる。

```
$ cd /var/www/
$ sudo rm index.html
$ sudo rm index2.html
$ ls
$ sudo chef-client
$ ls
```

---

---

## まとめ

---



---

## 資料作成責任者

---

堀田直孝

email: [fukuzo@cybertron.co.jp](mailto:fukuzo@cybertron.co.jp)



### 8.1 注意書き

ハンズオン実施にあたり手順の検証は行っておりますが、全ての動作を保証するものではありません。ドキュメント内で紹介しているコマンドや各構成要素の仕様変更は、Opscode wiki のドキュメントを参考に、常に最新情報に読み替えてください。

またハンズオン参加者が、本ドキュメントに従って操作をしたことにより不利益 / 損害等にあったとしても、Japan Chef User Group は一切保証しません。各参加者の責任において、パブリッククラウド及び Chef の操作を実行してください。