Infrastructure as Codeを学ぶ実践的ハンズオン

～大規模インフラのノウハウと春の彩りを添えて～

pyama86 / GMO Pepabo, Inc.

2018.04.08 インフラハンズオン

ホスティング事業部 チーフテクニカルリード

山下 和彦 @pyama86

ペパボ福岡

Infrastructure as Code

Infrastructure as Code

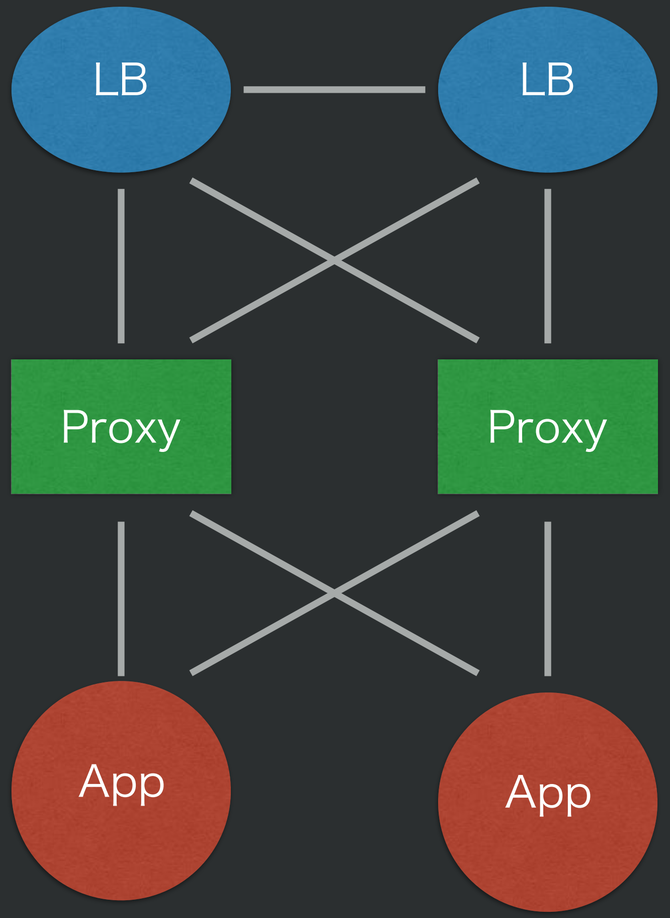
- ソフトウェア開発のノウハウをインフラへ
  - バージョン管理
  - テスト駆動開発
  - 継続的インテグレーション
  - 継続的デリバリー

Infrastructure as Code

- ソフトウェア開発のノウハウをインフラへ
  - バージョン管理
  - テスト駆動開発
  - 継続的インテグレーション
  - 継続的デリバリー

実践的ハンズオン

ハンズオンのゴール



冗長化されたWebシステムを
Infrastructure as Codeを体感しながら
開発する

ハンズオンで使用するツール・ミドルウェア

プロダクト名	役割
vagrant	VMマネージメント
itamae	プロビジョニング
Sererspec	テスト
keepalived	ロードバランシング
nginx	WEBプロキシ
httpd	アプリケーション

vagrant

- Rubyの記法でVirtualBoxなどのVMを管理できるソフトウェア
- HashiCorp(https://www.hashicorp.com/)

https://www.vagrantup.com/

itamae

- Preferred Networks の @ryot_a_rai が作製したOSS
- Light chef
- シンプルで使いやすく、mruby版はmitamae

https://github.com/itamae-kitchen/itamae

Serverspec

- @mizzy が作製したOSS
- RSpecの記法でサーバ環境をテストできる

http://serverspec.org/

ハンズオン環境

- https://github.com/pepabo/infrasturucture_as_code_hands_on

```
├── Gemfile // 利用gemの定義
├── Gemfile.lock // gemのバージョン固定ファイル
├── README.md
├── Vagrantfile // VMの定義
├── bootstrap.rb // Itamaeの実行スクリプト
├── cookbooks // プロビジョニングスクリプトを配置するディレクトリ
    └── www
        ├── default.rb
        ├── files
        └── templates
```

ハンズオン環境

- https://github.com/pepabo/infrasturucture_as_code_hands_on

```
├── nodes // ロール毎のアトリビュートファイルを配置するディレクトリ
│   └── www.yaml
├── roles // ロール毎のプロビジョニングスクリプトを配置するディレクトリ
│   └── www
│       └── default.rb
├── spec // Serverpecのスクリプトを配置するディレクトリ
│   ├── spec_helper.rb
│   └── www
│       └── httpd_spec.rb
└── vagrant_properties.yml // vagrantの構成ファイル
```

進め方

- WWWロール作る
- PROXYロール作る
- LBロール作る

簡単！！！？

WWWロールの要件

- phpinfoが表示できる
- PHP7が利用できる

TDD

・テストでまずあるべき状態を定義してから、開発を行う

```
$ vagrant up www-1
$ bin/rake spec:www-1
```

apache2のインストール

・cookbooks/www/apache.rbの作成

・cookbooks/www/default.rbからapache.rbを読み込む

・roles/www/default.rbからwwwのcookbookを読み込む

package

パッケージのインストールを行うリソース

```
package <name> do
  action :install
  version 1.0
end
```

https://github.com/itamae-kitchen/itamae/wiki/package-resource

cookbooks/www/apache.rb

・レシピはミドルウェア単位で分割し、再利用性を高める

```
%w(
  apache2
  php7.0
  libapache2-mod-php7.0
).each do |n|
  package n
end
```

デフォルトアクションが :install なので省略可能

cookbooks/www/default.rbの定義

```
# 先ほど定義したレシピを読み込む

include_recipe 'apache.rb'
```

default.rbは複数のレシピを束ねる役割に使う

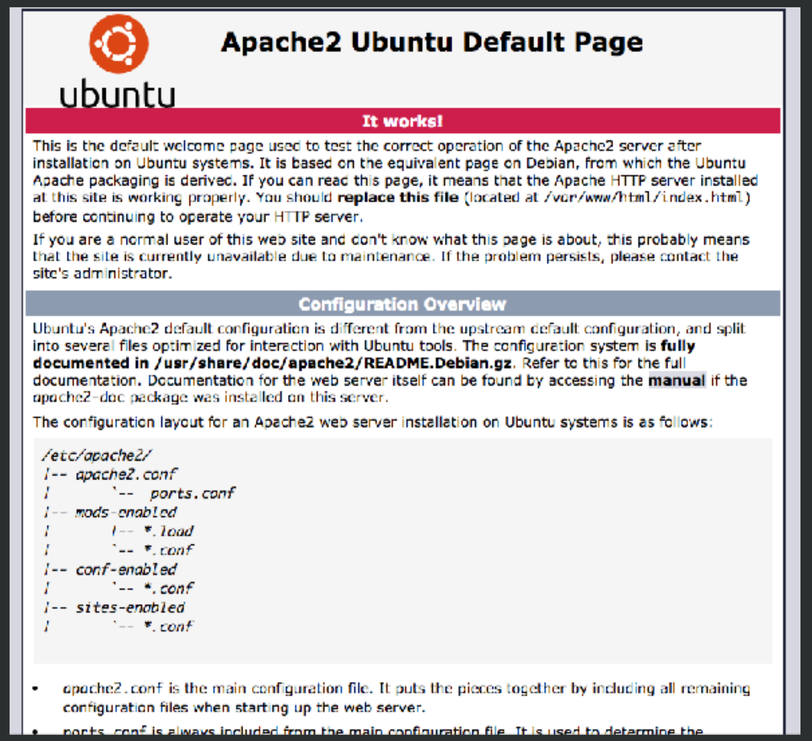
roles/www/default.rbの定義

```
# 先ほど定義したクックブックを読み込む
include_cookbook 'www'
```

このようにすると、一つのロールを複数の
クックブックを組み合わせて構築できる

Let’s Provision!!!1

```
$ vagrant provision www-1
```



http://www-1.hands-on.pbdev/

ubuntuのデフォルトページが見えればOK

phpinfoを表示する

・spec/www/phpinfo_spec.rb

```
describe file('/var/www/html/index.php') do
  it { should be_file }
  it { should be_mode 755 }
  it { should be_owned_by 'root' }
  it { should be_grouped_into 'root' }
end
```

http://serverspec.org/resource_types.html#file

phpinfoを表示する

・cookbooks/www/phpinfo.rb

```
remote_file '/var/www/html/index.php' do
  owner 'root'
  group 'root'
  mode '755'
end
```

https://github.com/itamae-kitchen/itamae/wiki/remote_file-resource

phpinfoを表示する

・cookbooks/www/files/index.php

```
<?php
    echo phpinfo();
```

※一般的にはプロビジョニングツールで

コンテンツ配置はやらない

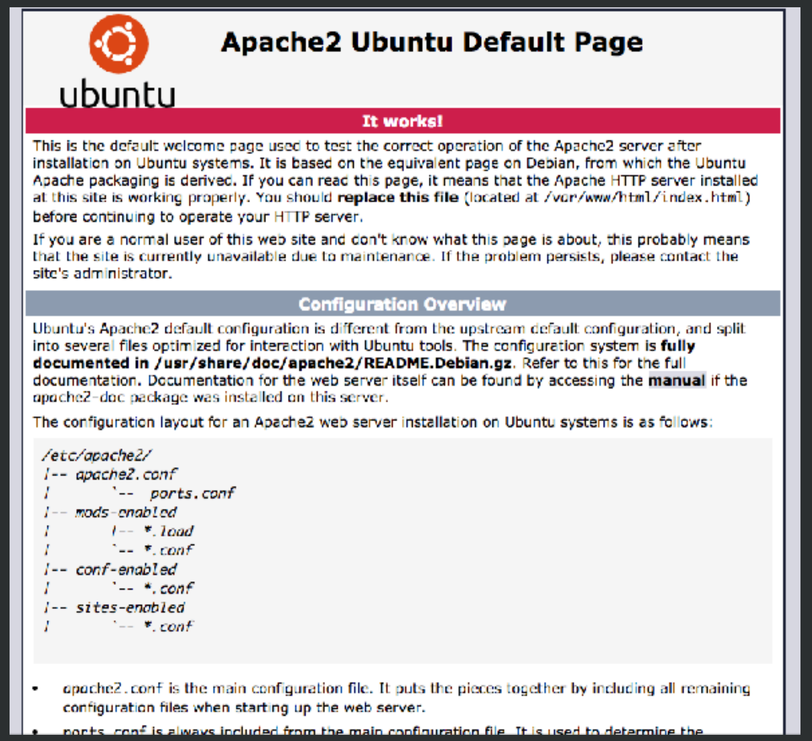
cookbooks/www/default.rbの定義

```
# 先ほど定義したレシピを読み込む

include_recipe ‘apache.rb'
include_recipe ‘phpinfo.rb’
```

default.rbは複数のレシピを束ねる役割に使う

Let’s Provision!!!1

```
$ vagrant provision www-1
```



http://www-1.hands-on.pbdev/

phpinfo見えない・・・

sshログイン

```
$ vagrant ssh www-1
ubuntu@www-1:~$ ls -ltr /var/www/html
total 16
-rw-r--r-- 1 root   root 11321 Apr  3 07:13 index.html
-rwxr-xr-x 1 ubuntu root    26 Apr  3 07:29 index.php
```

index.htmlを削除する必要がある

index.htmlの削除

- spec/www/phpinfo_spec.rb

```
describe file('/var/www/html/index.html') do
  it { should_not exist }
end
```

index.htmlの削除

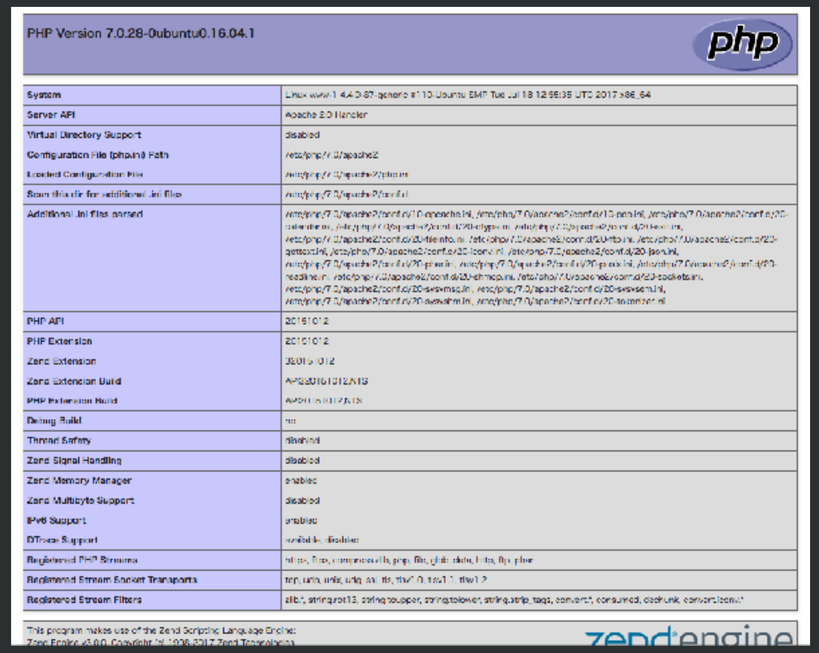
- cookbooks/www/phpinfo.rb

```
file '/var/www/html/index.html' do
  action :delete
end
```

https://github.com/itamae-kitchen/itamae/wiki/file-resource

Let’s Provision!!!1

```
$ vagrant provision www-1
$ bin/rake spec:www-1
```



http://www-1.hands-on.pbdev/

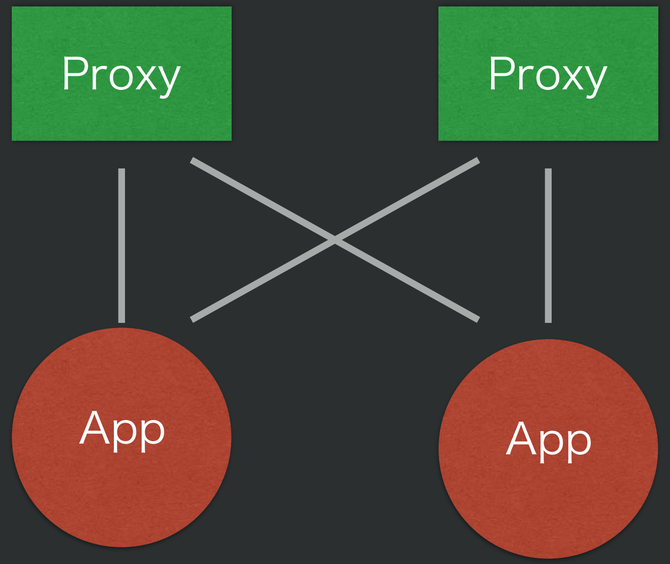
YY!!!!1

記法チェックとcommit

```
$ bin/rake rubocop -a
$ git add .
$ git commit -m “wwwの構築”
```

PROXYロールの要件

・nginxを利用したHTTPプロキシができる



nginxをインストールする

- spec/proxy/nginx_spec.rb

```
require 'spec_helper'
%w(
  nginx
).each do |n|
  describe package(n) do
    it { should be_installed }
  end
end

describe service('nginx') do
  it { should be_enabled }
  it { should be_running }
end

describe port(80) do
  it { should be_listening }
end
```

cookbook、roleの雛形を作る

```
$ bin/itamae generate cookbook proxy
$ bin/itamae generate role proxy
$ echo ‘role: proxy’ > nodes/proxy.yaml
$ vagrant up proxy-1
$ bin/rake spec:proxy-1
```

nginxのインストール

- cookbooks/proxy/nginx.rb

```
package 'nginx'

service ‘nginx’ do
  action %w(enable start)
end
```

nginxのインストール

- cookbooks/proxy/default.rb

```
include_recipe 'nginx.rb'
```

- roles/proxy/default.rb

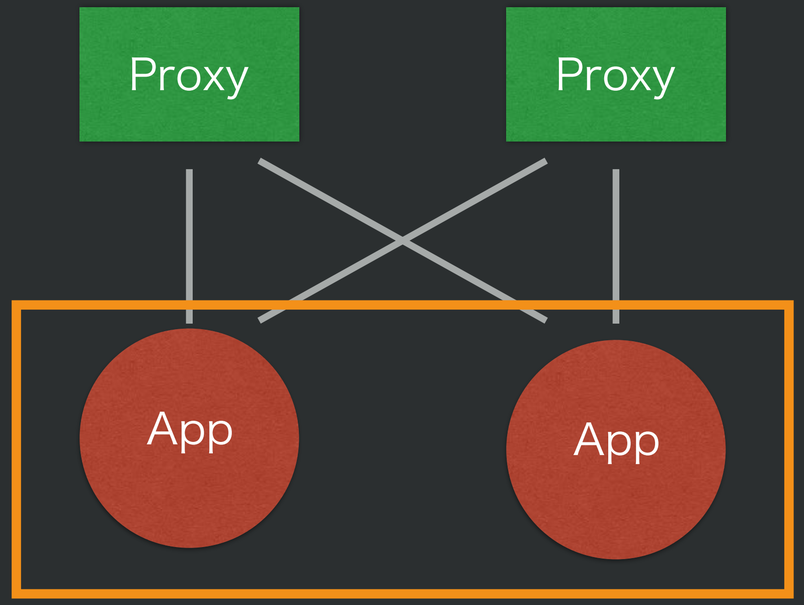
```
include_cookbook 'proxy'
```

nginxの設定をする

- proxyサーバの中を覗く

```
% vagrant ssh proxy-1
ubuntu@proxy-1:~$ sudo su -
root@proxy-1:~# cd /etc/nginx/
root@proxy-1:/etc/nginx# ls -ltr
root@proxy-1:/etc/nginx# more nginx.conf
…
include /etc/nginx/conf.d/*.conf; # nginxのconfigにはinclude機構がある

…
```

upstreamの定義を行う



proxyサーバから見て、プロキシ先の

サーバをupstreamと定義

upstreamの定義を行う

- spec/proxy/nginx_spec.rb

```
describe file('/etc/nginx/conf.d/www.conf') do
  its(:content) { should match /server 172.18.1.31/ }
  its(:content) { should match /server 172.18.1.32/ }
end

describe file('/etc/nginx/sites-enabled') do
  it { should_not exist }
end

describe file('/etc/nginx/sites-available') do
  it { should_not exist }
end
```

templateを利用する

```
template '/etc/nginx/conf.d/www.conf' do
  owner 'root'
  group 'root'
  notifies :restart, 'service[nginx]'
end
```

https://github.com/itamae-kitchen/itamae/wiki/template-resource

templateを利用する

cookbooks/proxy/templates/www.conf.erb

```
upstream app {
  <% node['proxy']['app']['servers'].each do |s| %>
  server <%= s %>;
  <% end %>
}

server {
  listen       80;
  server_name  localhost;
  location / {
    proxy_pass http://app/;
  }
}
```

環境ごとのファイルはnodesで管理

nodes/proxy.yaml

```
proxy:
  app:
    servers:
      - 172.18.1.31
      - 172.18.1.32
```

複数環境の場合は、proxy-**production**.yamlや

proxy-**development**.yamlなどを作成する

課題

/etc/nginx/sites-enabled

/etc/nginx/sites-available

上記のディレクトリを

削除してください

正答例

```
%w(
  enabled
  available
).each do |n|
  directory "/etc/nginx/sites-#{n}" do
    action :delete
    notifies :restart, 'service[nginx]'
  end
end
```

記法チェックとcommit

```
$ bin/rake rubocop -a
$ git add .
$ git status
$ git commit -m “proxyの構築”
```

課題

/etc/nginx/conf.d/www.conf

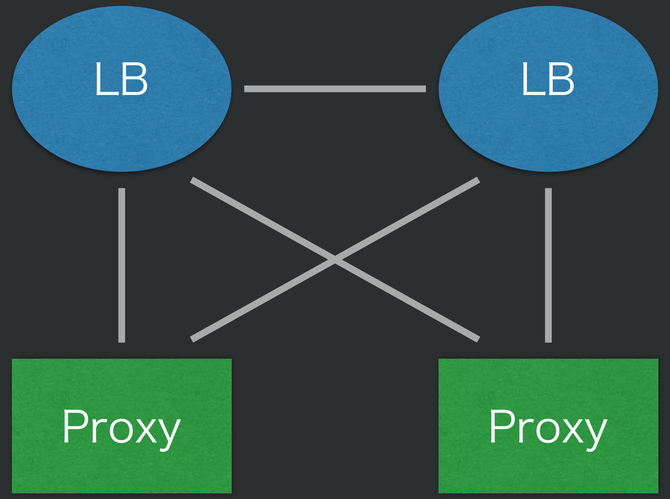
このレシピをtemplateの

variablesを利用してより

再利用性を高める

LBロールの要件

- VIPを管理できる
- VIPで受けたトラフィックをProxyサーバにバランシングできる



VIPとは

keepalivedをインストールする

- spec/lb/keepalived_spec.rb

```
require 'spec_helper'
%w(
  keepalived
).each do |n|
  describe package(n) do
    it { should be_installed }
  end
end

describe service('keepalived') do
  it { should be_enabled }
  it { should be_running }
end

describe port(80) do
  it { should be_listening }
end
```

リアルサーバもテスト

- spec/lb/keepalived_spec.rb

```
describe file(‘/etc/keepalived/keepalived.conf’) do
  its(:content) { should match /real_server 172.18.1.21 80/ }
  its(:content) { should match /real_server 172.18.1.22 80/ }
end
```

cookbook、roleの雛形を作る

```
$ bin/itamae generate cookbook lb
$ bin/itamae generate role lb
$ echo ‘role: lb’ > nodes/lb.yaml
$ vagrant up lb-1
$ bin/rake spec:lb-1
```

keepalivedのインストール

- cookbooks/lb/keepalived.rb

```
package 'keepalived'

service ‘keepalived’ do
  %w(enable start)
end

template '/etc/keepalived/keepalived.conf' do
  owner 'root'
  group 'root'
  notifies :restart, 'service[keepalived]'
end
```

keepalivedのインストール

- cookbooks/lb/default.rb

```
include_recipe ‘keepalived.rb'
```

- roles/lb/default.rb

```
include_cookbook ‘lb’
```

VIPの定義

- cookbooks/lb/templates/keepalived.conf.erb

```
vrrp_instance vrrp_int {
  interface <%= node['lb']['keepalived']['if'] %>
  virtual_router_id <%= node['lb']['keepalived']['router_id'] %>
  nopreempt
  state BACKUP
  priority 100
  advert_int 3
  garp_master_delay 5
  authentication {
    auth_type PASS
    auth_pass hands_on
  }
  virtual_ipaddress {
    <%= node['lb']['keepalived']['vip'] %>
  }
}
```

VIPの定義

- cookbooks/lb/templates/keepalived.conf.erb

```
virtual_server <%= node['lb']['keepalived']['vip'] %> 80 {
  delay_loop 10
  lvs_sched  lc
  lvs_method NAT
  protocol TCP
  <% node['lb']['keepalived']['servers'].each do |s| %>
    real_server <%= s %> 80 {
    weight 1
    TCP_CHECK {
      connect_port 80
      connect_timeout 30
    }
  }
  <% end %>
}
```

YAMLでアトリビュートを定義する

- nodes/lb.yaml

```
lb:
  keepalived:
    vip: 172.18.1.10
    router_id: 100
    if: enp0s8
    servers:
      - 172.18.1.21
      - 172.18.1.22
```

Let’s Provision!!!1

```
$ vagrant provision lb-1
$ bin/rake spec:lb-1
```

サーバの中を見てみましょう

ipvsの状態を見る

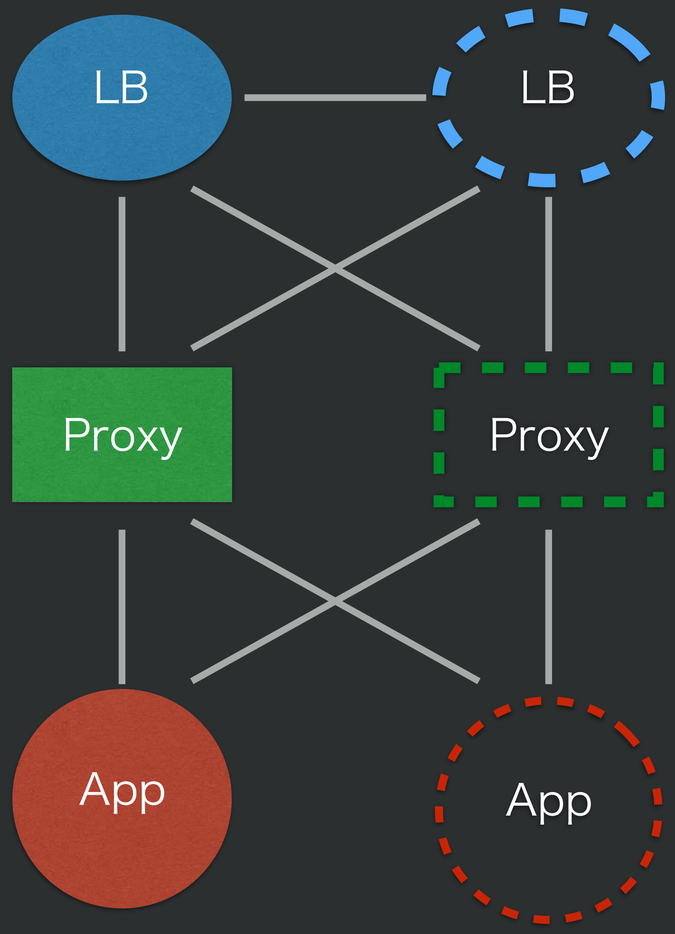
```
# vipを保持しているか

$ ip a
# real serverの状態を見る

$ ipvsadm -L -n
```

記法チェックとcommit

```
$ bin/rake rubocop -a
$ git commit -m “lbの構築”
```

冗長化

待機系を起動する




```
$ vagrant up
```

ipvsの状態を見る

```
# real serverの状態を見る

$ ipvsadm -L -n
```

おもむろにサーバを落とす

```
$ vagrant halt www-1
$ vagrant halt proxy-1
$ vagrant halt lb-1
```

サービスが無停止であること

まとめ

今日学んだこと

- Infrastructure as Codeはソフトウェア開発のノウハウをインフラの世界へ持ってきたもの
- インフラにおいてもテスト駆動開発
- 再利用しやすい粒度でレシピを管理
- インフラは楽しい！！！1