

The Machines are Alive

Engine Yard Cloud



On-demand deployment
and management for
Ruby on Rails applications

deployment vs. surfing

Jon Crosby

<http://joncrosby.me>

(we're hiring)

<http://www.engineyard.com/about/careers>

cloud computing definition

instant provisioning

cost effective

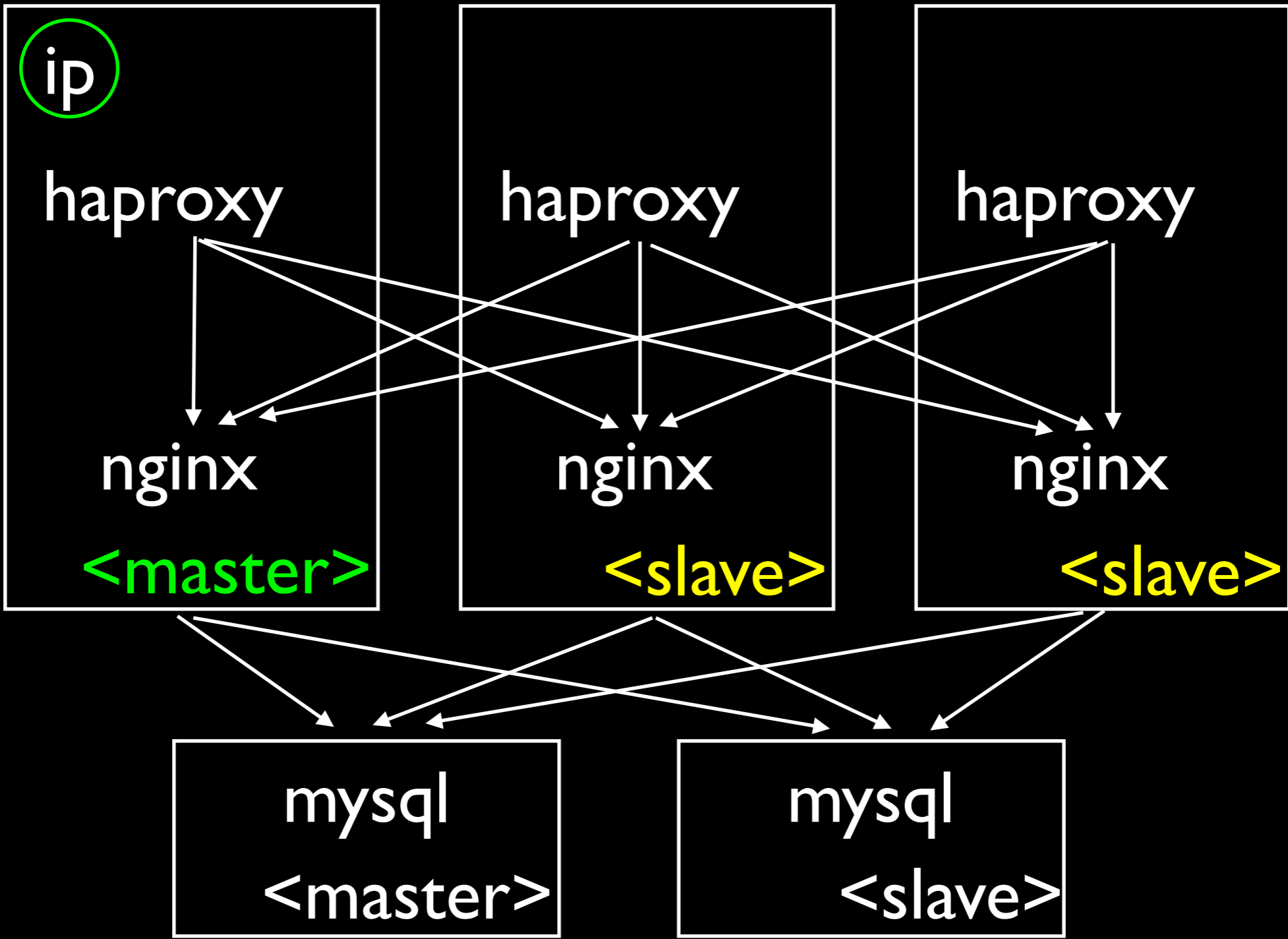
repeatable

infrastructure described in code

elastic / scalable

git url -> cluster

the internets



A Story
(of not scaling)

add a node

Stewart Smalley



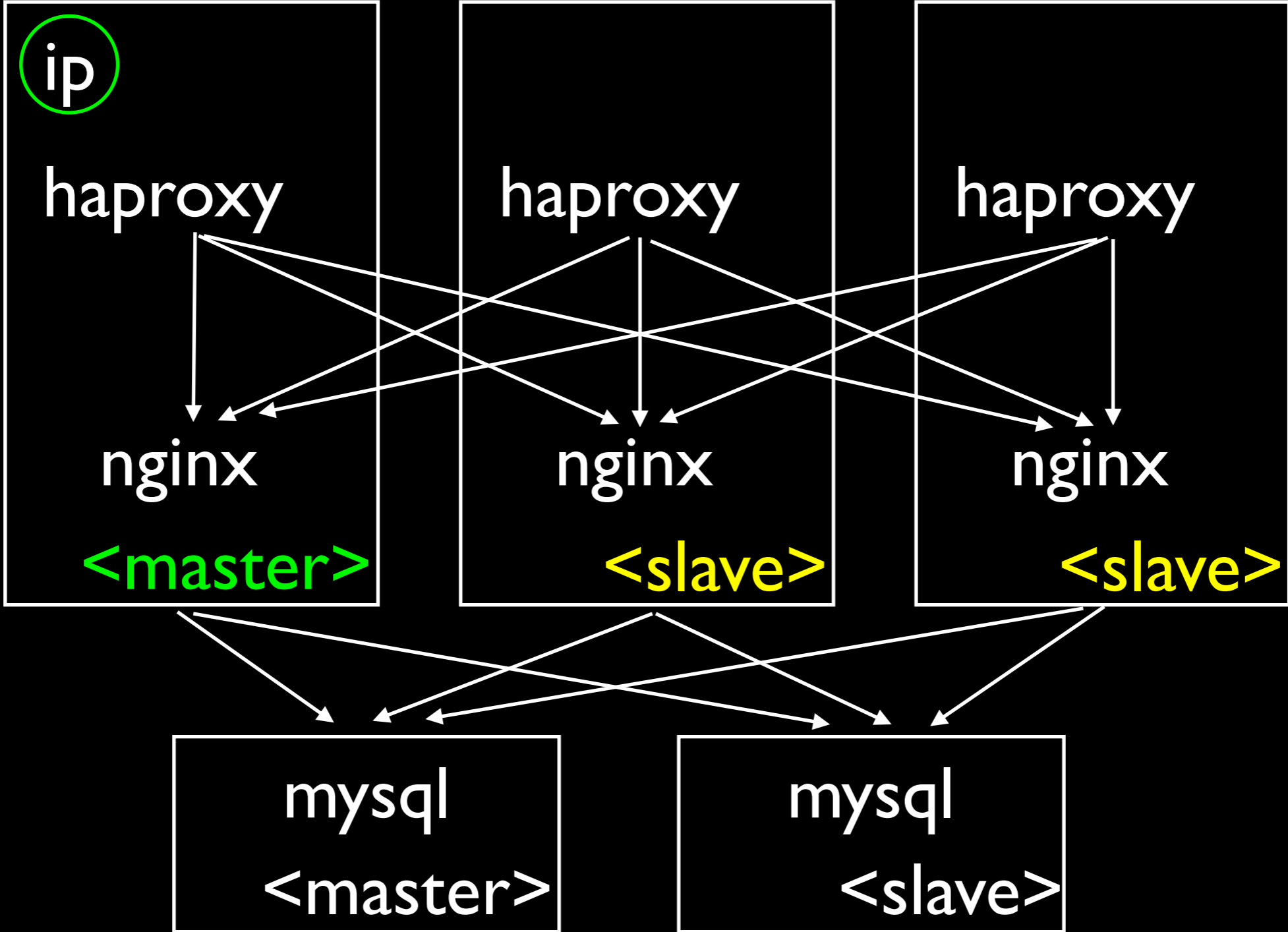
no server is special

remove a node

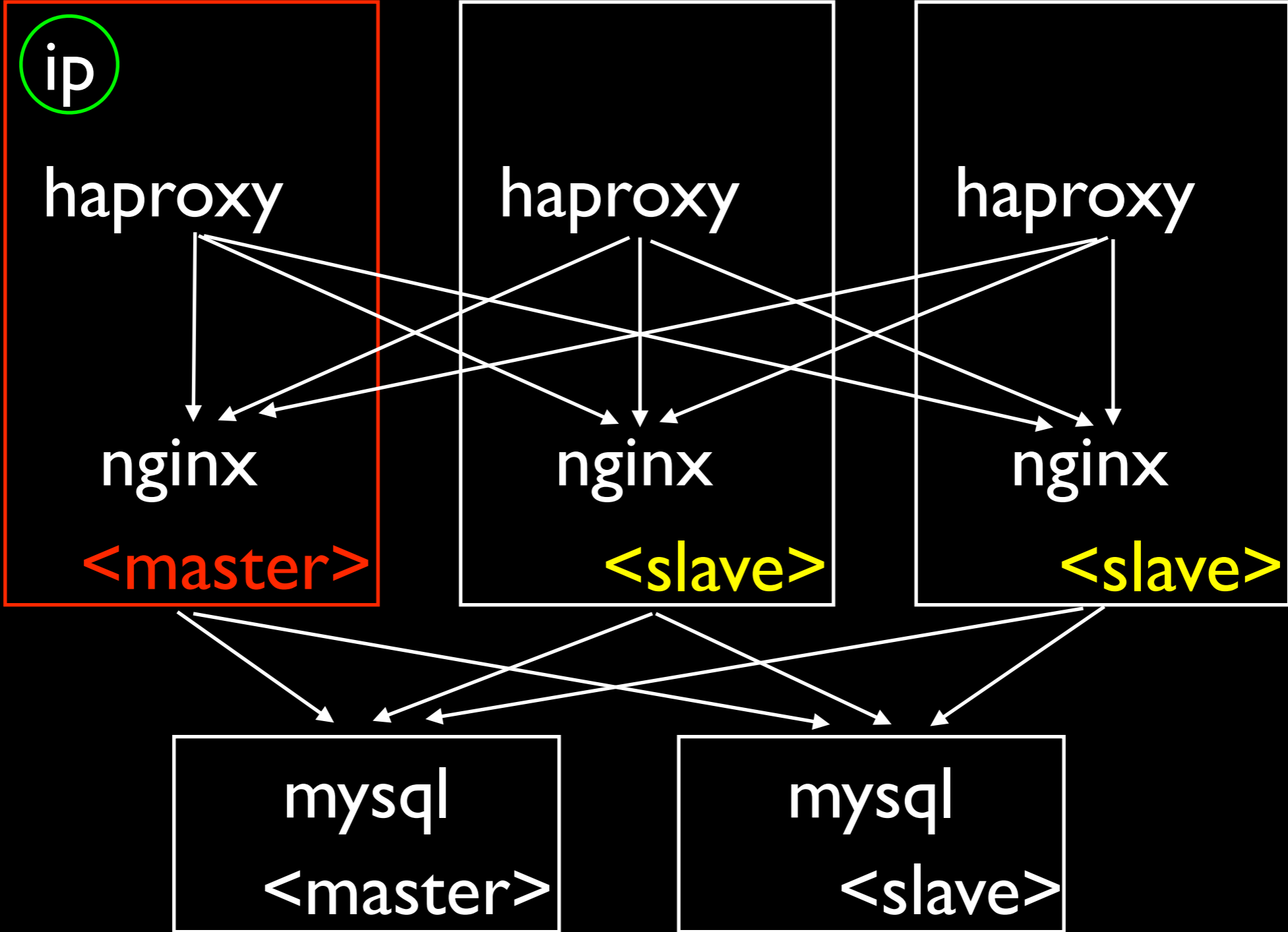
self healing

“The Machines are Alive”

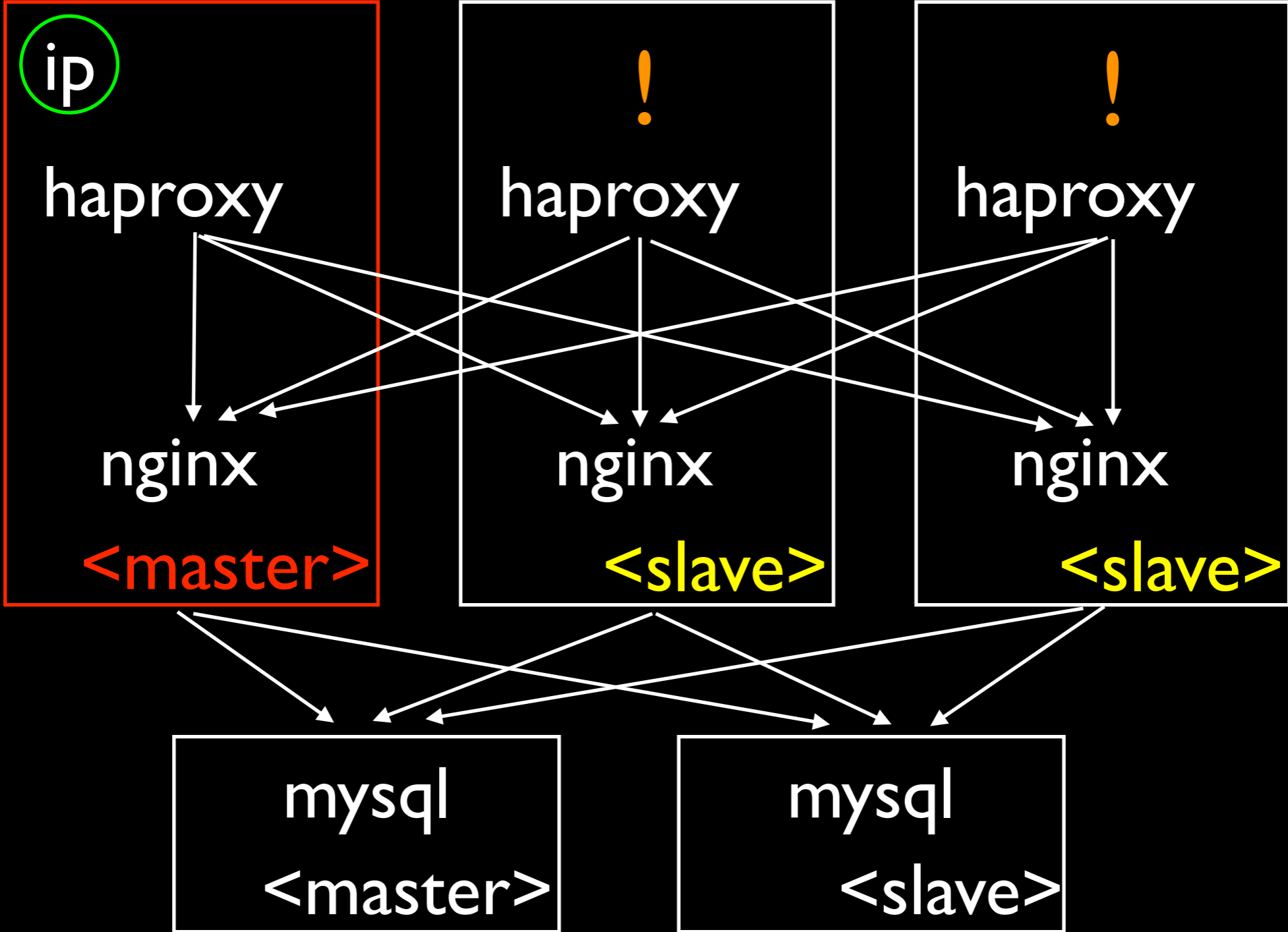
the internets

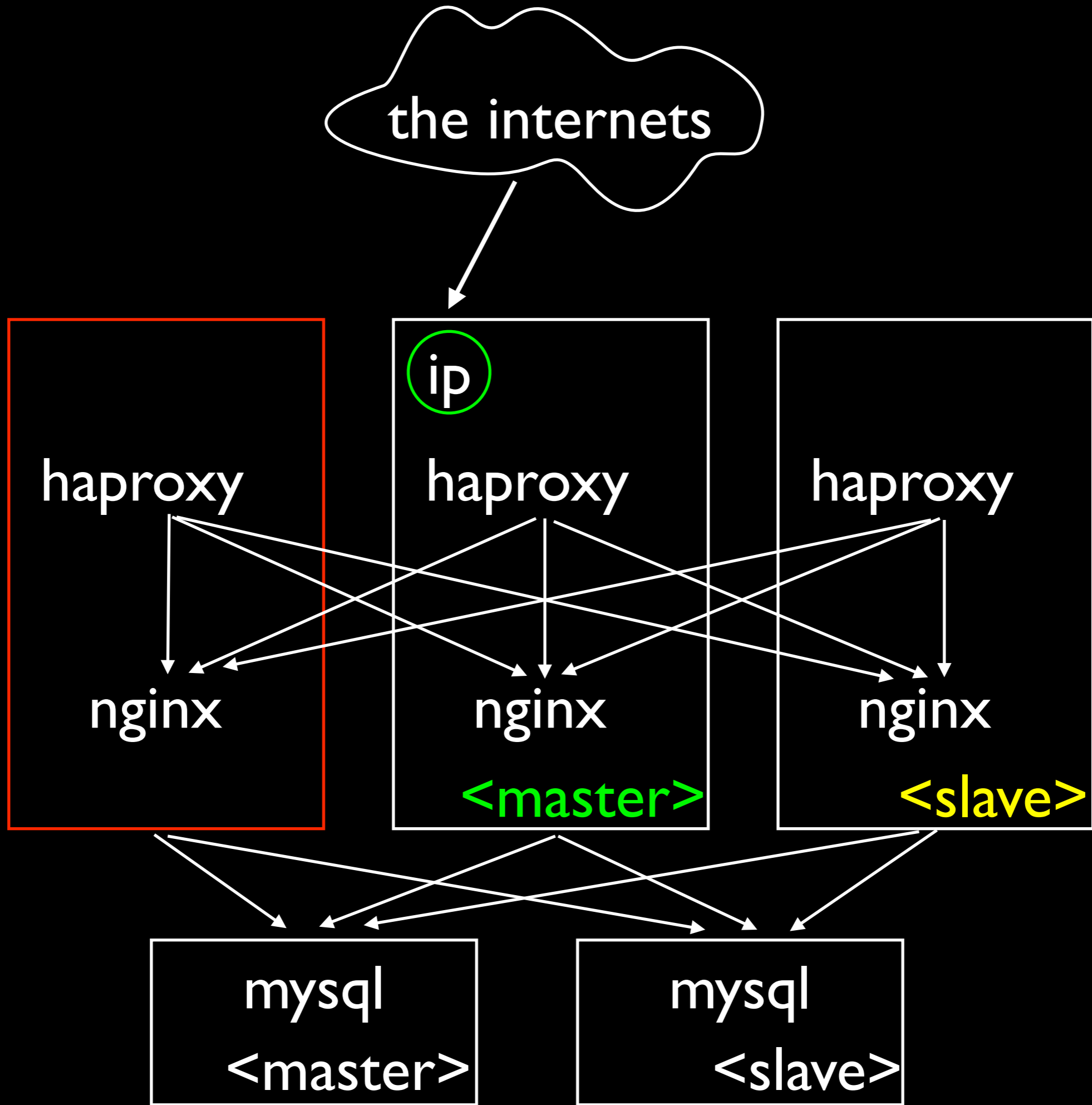


the internets

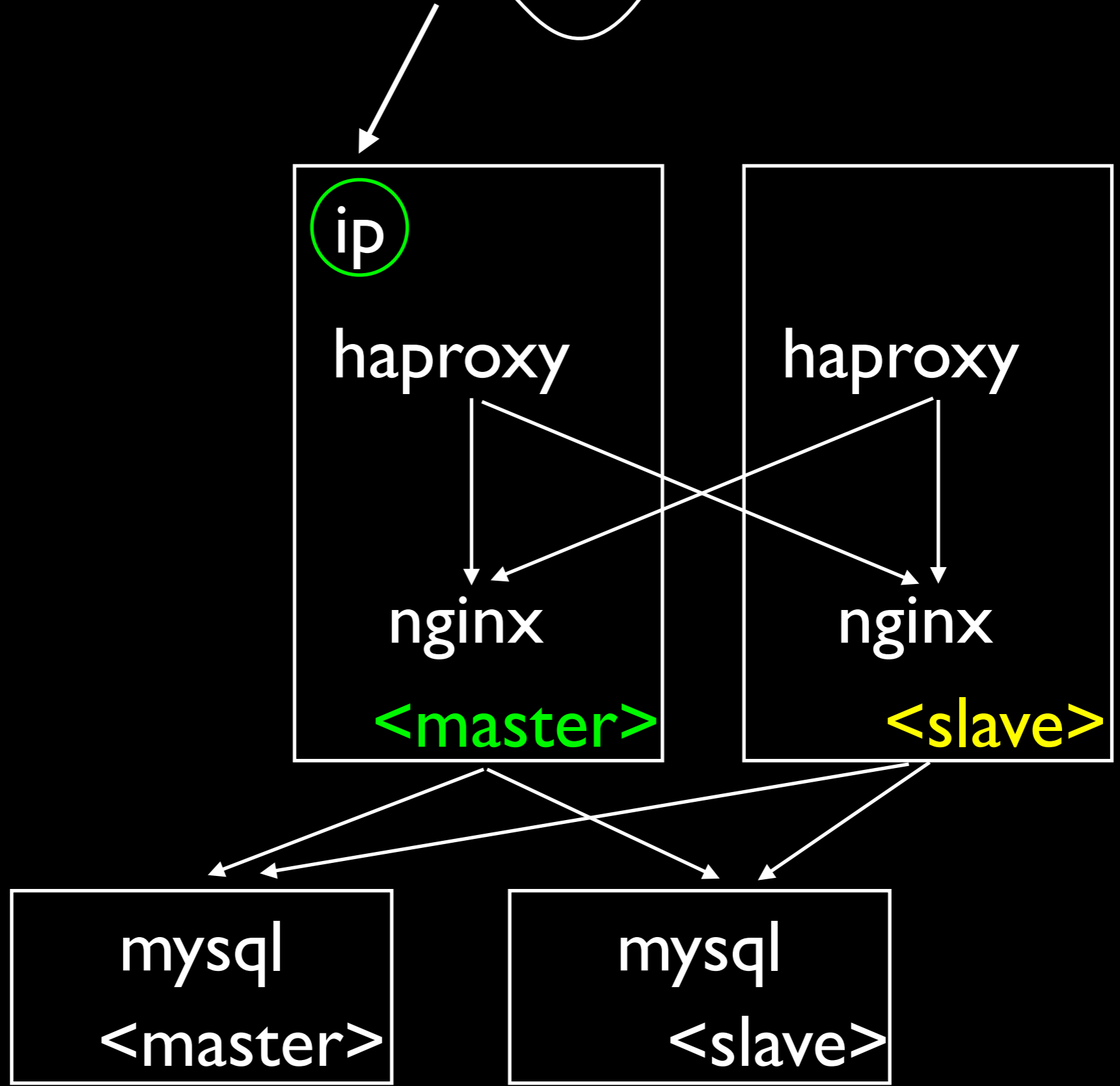


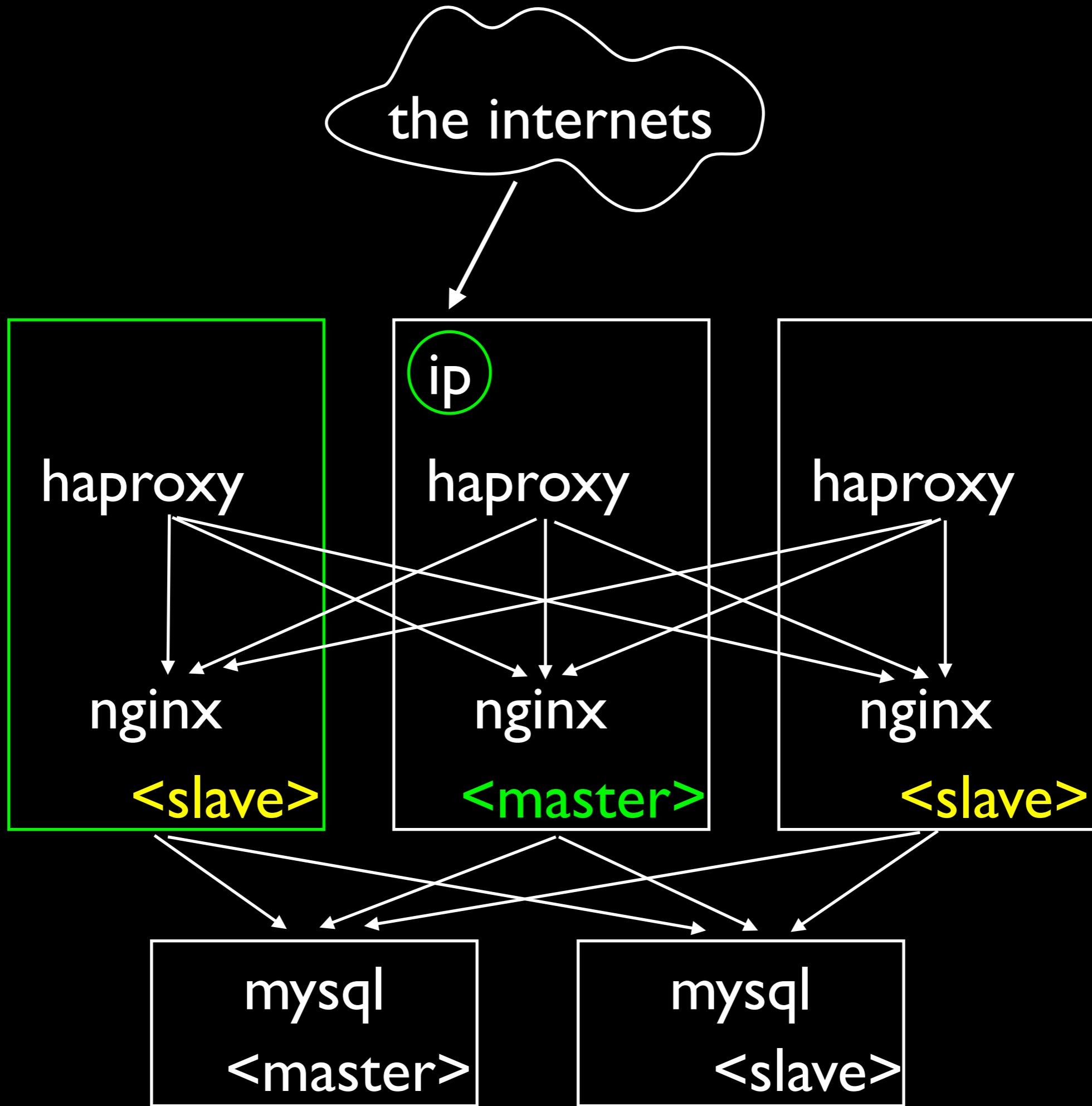
the internets





the internets





deploying updates

deploy from the web

deploy hooks

```
deploy/  
|-- after_restart.rb  
|-- before_migrate.rb  
|-- before_restart.rb  
`-- before_symlink.rb
```

```
run "echo $RAILS_ENV >> #{release_path}/hello.log"
```

capistrano

deploy from git

cloning

murphy

monitoring and alerting

secret sauce

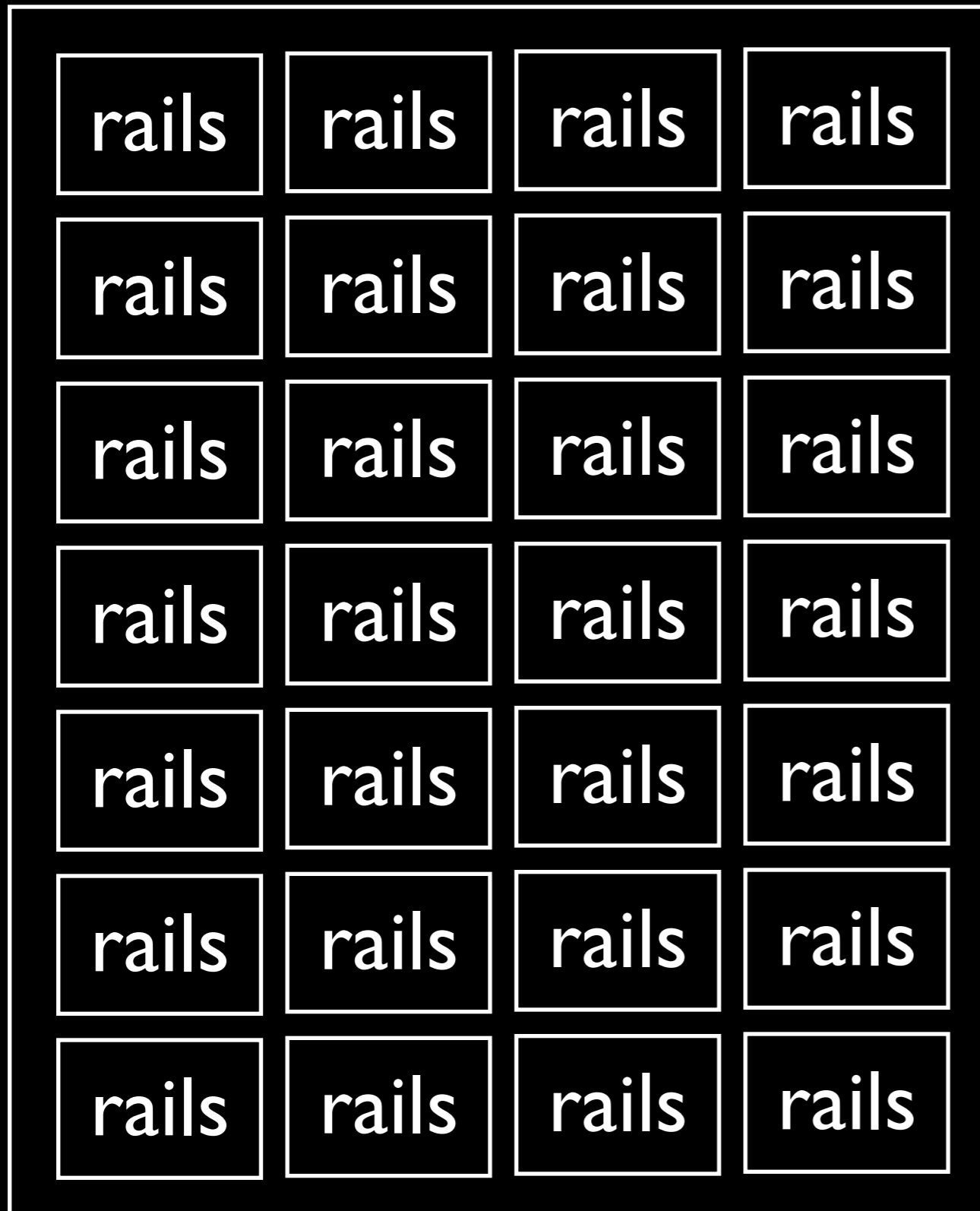
<http://cloudscaling.com/blog/cloud-applications/the-secret-sauce-problem>

credit: Randy Bias

deployment platform

customizable

proprietary platforms



elsewhere



extensible platforms

rails	rails	rails	rails
rails	rails	rails	rails
rails	rails	rails	rails
rails	rails	rails	rails
rails	rails	rails	rails
rails	rails	rails	rails
rails	rails	rails	rails
rails	rails	rails	rails



customization with chef

systems integration framework

node

roles

cookbooks

recipes

resources

files, packages, services, etc.

chef example

```
if ['solo', 'app', 'app_master'].include?(node[:instance_role])
```

```
if ['solo', 'app', 'app_master'].include?(node[:instance_role])
```

/etc/chef/dna.json

```
{  
  "instance_role" : "app_master",  
  "applications" : {  
    "railsapp" : {  
      "repository_name" : "git://github.com...",  
      "services" : [  
        {  
          "resource" : "mongrel",  
          "mongrel_base_port" : 5000,  
          "mongrel_mem_limit" : 150,  
          "mongrel_instance_count" : 5  
        }  
      ]  
    }  
  }  
}
```

...

```
run_for_app("myapp") do |app_name, data|
```

```
directory "/var/run/sphinx" do
  owner node[:owner_name]
  group node[:owner_name]
  mode 0755
end
```

```
directory "/var/log/engineyard/sphinx/#{app_name}" do
  recursive true
  owner node[:owner_name]
  group node[:owner_name]
  mode 0755
end
```

```
remote_file "/etc/logrotate.d/sphinx" do
  owner "root"
  group "root"
  mode 0755
  source "sphinx.logrotate"
  action :create
end
```

```
template "/etc/monit.d/sphinx.#{app_name}.monitrc" do
  source "sphinx.monitrc.erb"
  owner node[:owner_name]
  group node[:owner_name]
  mode 0644
  variables({
    :app_name => app_name,
    :user => node[:owner_name]
  })
end
```

```
template "/data/#{app_name}/shared/config/sphinx.yml" do
  owner node[:owner_name]
  group node[:owner_name]
  mode 0644
  source "sphinx.yml.erb"
  variables({
    :app_name => app_name,
    :user => node[:owner_name]
  })
end
```

chef testing

test code

test infrastructure

?

appears tedious without clear benefit

```
echo "x" > file
```

```
assert(file)
```

```
assert(file.cat()) == "x"
```

chef == declarative

mock out unix?

assert **critical** parts

new parts **evolve** with confidence

```
|-- Rakefile
`-- cookbooks
  |-- couchdb
  | + files
  | | |-- recipes
  | | | `-- default.rb
  | | `-- templates
  | |   `-- default
  | |     `-- couch.ini.erb
  |-- postgres
  | |-- recipes
  | | `-- default.rb
  | `-- templates
  |   `-- default
  |     `-- database.yml.erb
```

```
|-- Rakefile
`-- cookbooks
  |-- couchdb
  | + files
  | |-- recipes
  | | `-- default.rb
  | `-- templates
  |   `-- default
  |       `-- couch.ini.erb
  |-- postgres
  | |-- recipes
  | | `-- default.rb
  | `-- templates
  |   `-- default
  |       `-- database.yml.erb
```

```
|-- Rakefile
`-- cookbooks
  |-- couchdb
  | + files
  | | |-- recipes
  | | | `-- default.rb
  | | `-- templates
  | |   |-- default
  | |   | `-- couch.ini.erb
  |-- postgres
  | |-- recipes
  | | `-- default.rb
  | `-- templates
  |   |-- default
  |   | `-- database.yml.erb
```

```
|-- Rakefile
`-- cookbooks
  |-- couchdb
  | + files
  |   |-- recipes
  |   | `-- default.rb
  |   `-- templates
  |       `-- default
  |           `-- couch.ini.erb
|-- postgres
  |-- recipes
  |   `-- default.rb
  `-- templates
      `-- default
          `-- database.yml.erb
```

```
|-- Rakefile
`-- cookbooks
  |-- couchdb
  | + files
  | | |-- recipes
  | | | `-- default.rb
  | | `-- templates
  | |   `-- default
  | |     `-- couch.ini.erb
  |-- postgres
  | |-- recipes
  | | `-- default.rb
  | `-- templates
  |   `-- default
  |     `-- database.yml.erb
```

testing chef with cucumber

**BDD framework with plain text
descriptions**

Feature: Nginx Server

In order to serve static HTTP requests

As a Cloud customer

I want to ensure nginx is properly configured

Scenario: Nginx service is running

Given chef has finished running

Then the nginx service has been started

And the nginx service has a pid

...

Scenario Outline: filesystem setup

Then the <path> <type> exists

And the owner of <path> is the user <user>

And the group of <path> is the group of the <user> user

And the permissions for <path> are <permissions>

...

Scenario Outline: filesystem setup

Then the **<path>** **<type>** exists

And the owner of **<path>** is the user **<user>**

And the group of **<path>** is the group of the **<user>** user

And the permissions for **<path>** are **<permissions>**

...

Scenarios: configuration

type	permissions	user	path
file	0644	customer's	/data/nginx/nginx.conf
directory	0775	customer's	/data/nginx/ssl
directory	0755	customer's	/data/nginx/common
file	0644	customer's	/data/nginx/common/proxy.conf
file	0644	customer's	/data/nginx/common/servers.conf
directory	0755	customer's	/data/nginx/servers
file	0644	customer's	/data/nginx/servers/default.conf
file	0644	customer's	/data/nginx/common/fcgi.conf
directory	0775	customer's	/var/log/engineyard/nginx
file	0755	root	/etc/init.d/nginx
file	0755	customer's	/data/nginx/mime.types
file	0755	customer's	/data/nginx/koi-utf
file	0755	customer's	/data/nginx/koi-win
file	0600	root	/etc/conf.d/nginx
file	0755	root	/etc/logrotate.d/nginx

Scenarios: configuration

type	permissions	user	path
file	0644	customer's	/data/nginx/nginx.conf
directory	0775	customer's	/data/nginx/ssl
directory	0755	customer's	/data/nginx/common
file	0644	customer's	/data/nginx/common/proxy.conf
file	0644	customer's	/data/nginx/common/servers.conf
directory	0755	customer's	/data/nginx/servers
file	0644	customer's	/data/nginx/servers/default.conf
file	0644	customer's	/data/nginx/common/fcgi.conf
directory	0775	customer's	/var/log/engineyard/nginx
file	0755	root	/etc/init.d/nginx
file	0755	customer's	/data/nginx/mime.types
file	0755	customer's	/data/nginx/koi-utf
file	0755	customer's	/data/nginx/koi-win
file	0600	root	/etc/conf.d/nginx
file	0755	root	/etc/logrotate.d/nginx

@app @app_master @nginx_passenger

Feature: Nginx Server

In order to serve static HTTP requests

As a Cloud customer

I want to ensure nginx is properly configured

Scenario: Nginx service is running

Given chef has finished running

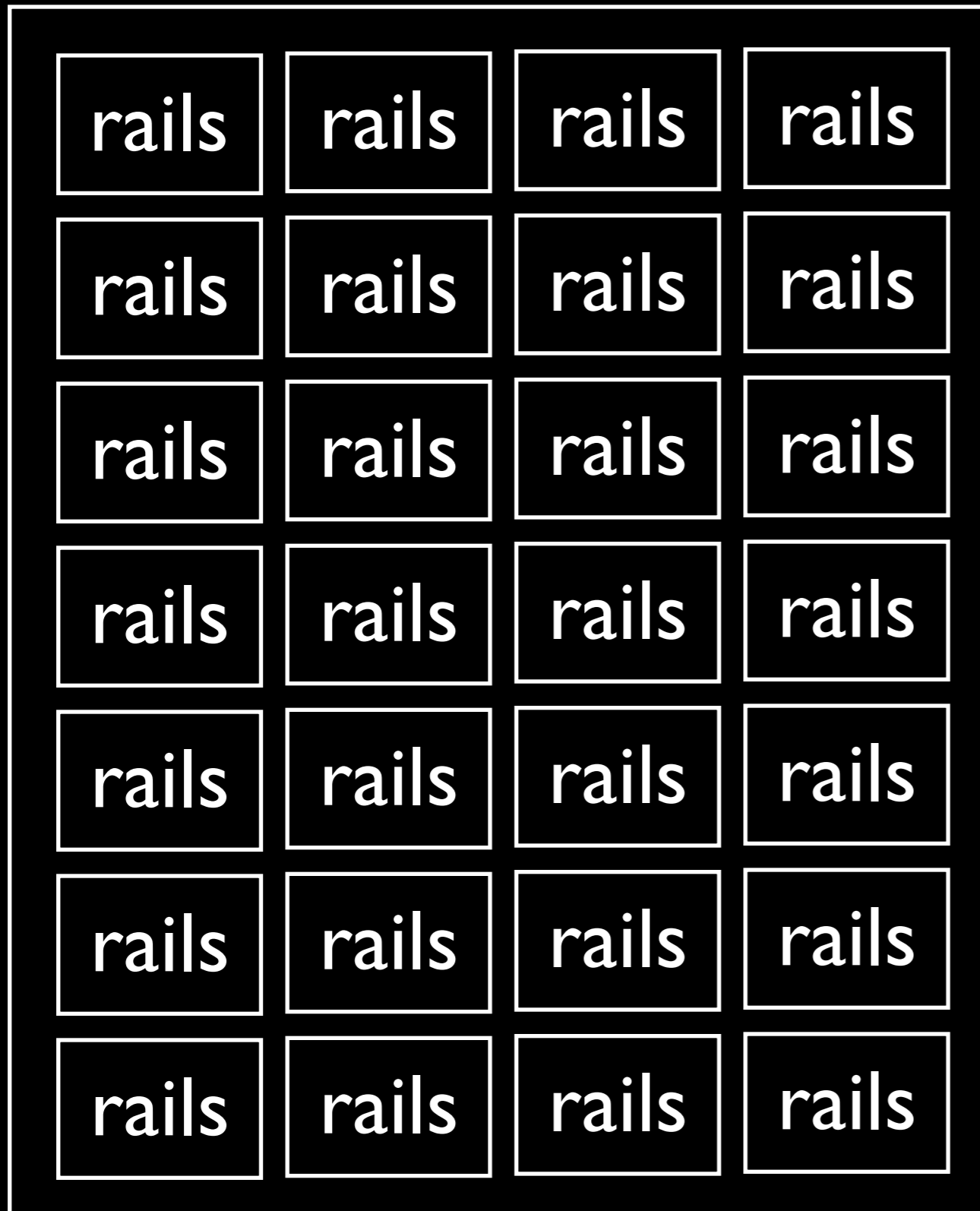
Then the nginx service has been started

And the nginx service has a pid

...

chef + monitoring

proprietary platforms



elsewhere



extensible platforms

rails	rails	rails	rails
rails	rails	rails	rails
rails	rails	rails	rails
rails	rails	rails	rails
rails	rails	rails	rails
rails	rails	rails	rails
rails	rails	rails	rails
rails	rails	rails	rails



`/etc/ey-alerts.json`

```
{  
  "skip" : ["mysqld"],  
  "check" : ["ttsrv"]  
}
```

growing community of recipes

open!

http://github.com/37signals/37s_cookbooks

<http://github.com/opscode/cookbooks/>

<http://github.com/engineyard/ey-cloud-recipes>

<http://github.com/ezmobius/ey-nosql>

scales with your needs

world class support

(new: two tiers, optional)

review

cloud computing platform

focused on **ruby**

configurable

open way

chef

scale up

scale out

on demand

your assignment:

find the secret **sauce**

build an **amazing** app

meaningful

worth using

don't make a clone of facebook

don't make a social
network for turtles

don't listen to social
marketing gurus with
great haircuts following
over 1000 people on
twitter

being awesome

part of your job

git url

one server

anything you can imagine

no excuse not to do so

get started

cook the **sauce**

make your users **happy**

done

q/a