Search

Update a Cookbook to Dynamically Use Nodes with the company_web policy name

Objectives

After completing this module, you should be able to

- Describe the query syntax used in search
- Build a search into your recipe code
- Create a Ruby String and Array dynamically
- Test that your load balancer is still balancing traffic

CONCEPT

Search

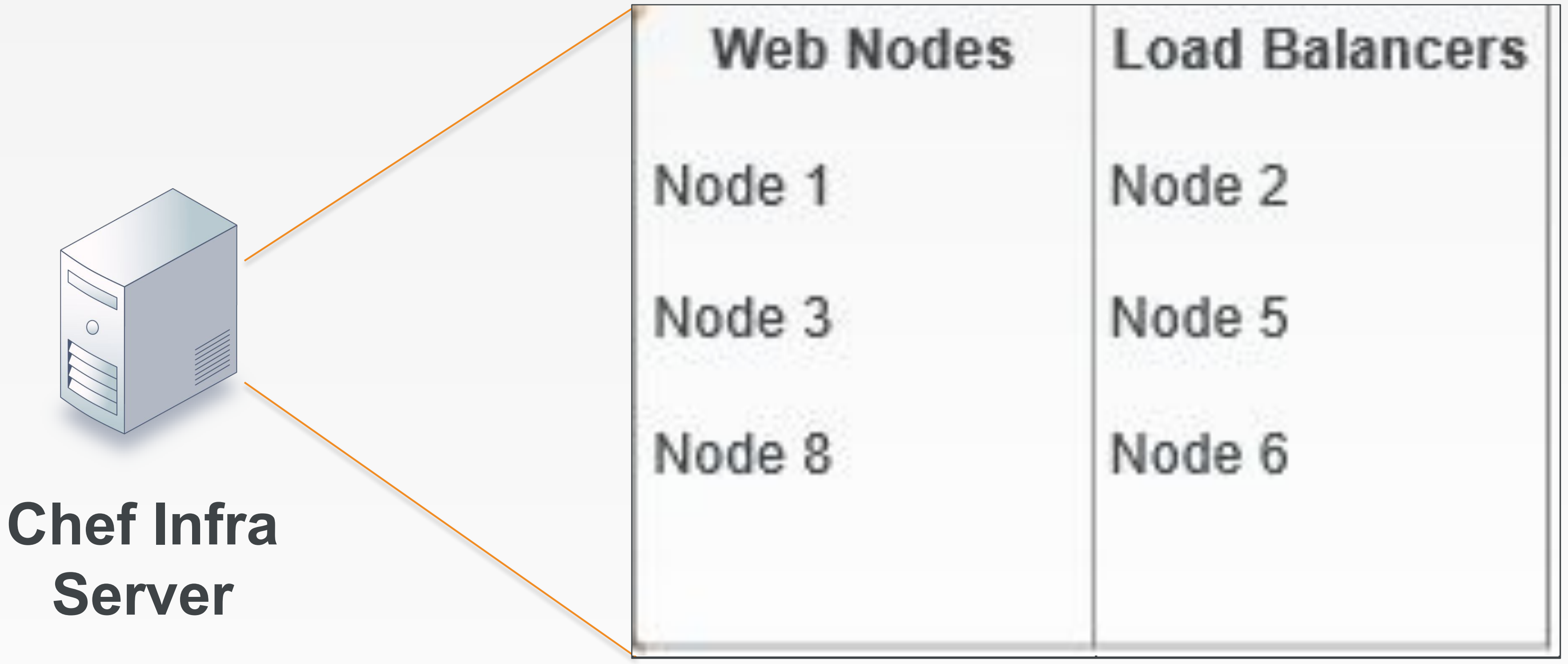
To add new servers as load balancer members, we would need to bootstrap a new web server and then update our load balancer's myhaproxy cookbook recipe.

That seems inefficient to have to update a cookbook recipe manually.

The Chef Infra Server and Search

Chef Infra Server maintains a representation of all the nodes within our infrastructure that can be searched on.

Search is a service discovery tool that allows us to query the Chef Infra Server.



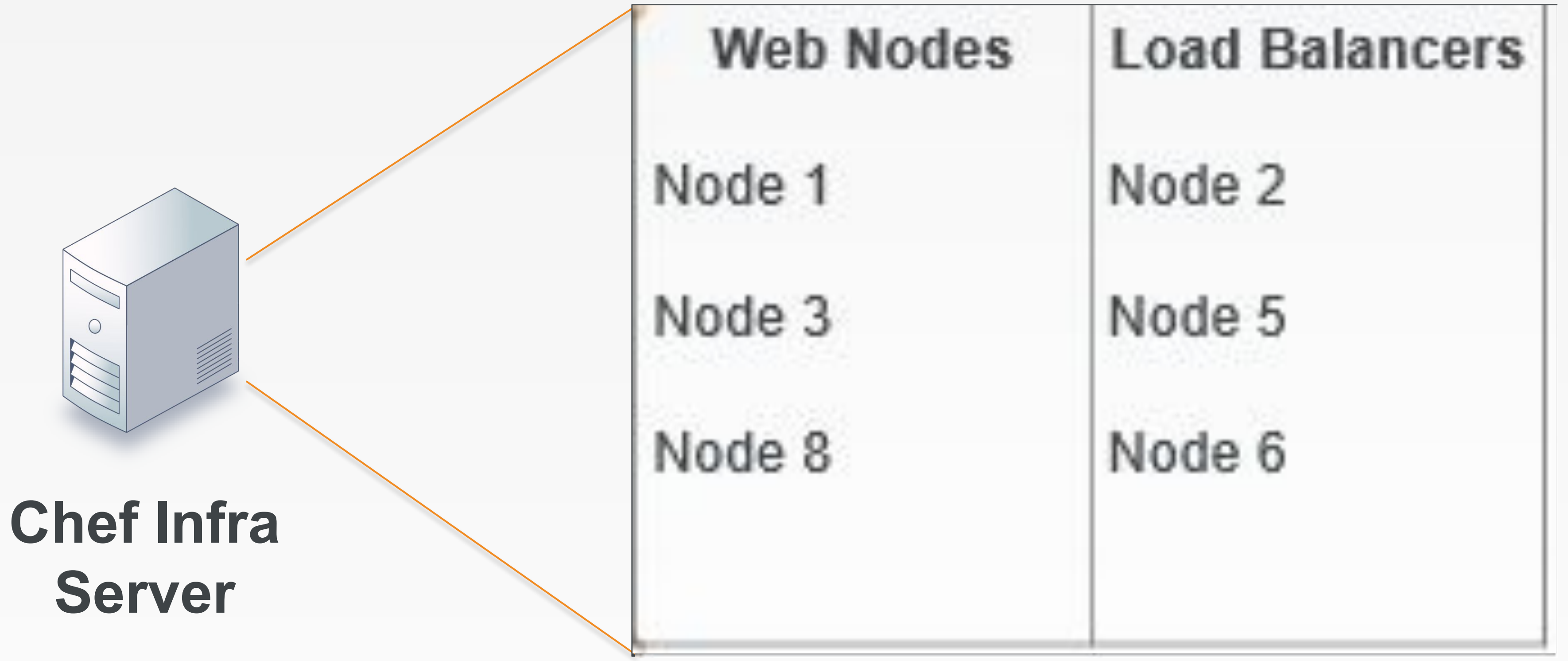
Web Nodes	Load Balancers
Node 1	Node 2
Node 3	Node 5
Node 8	Node 6

https://docs.chef.io/chef_search.html

https://docs.chef.io/chef_search.html#search-indexes

The Chef Infra Server and Search

We can ask the Chef Infra Server to return all the nodes or a subset of nodes based on the query syntax that we provide it through `knife search` or within our recipes through `search`.



Web Nodes	Load Balancers
Node 1	Node 2
Node 3	Node 5
Node 8	Node 6

Search Syntax

A search query is comprised of two parts: the key and the search pattern. A search query has the following syntax:

key:search_pattern

...where key is a field name that is found in the JSON description of an indexable object on the Chef Infra Server and search_pattern defines what will be searched for

Search Criteria

We may use wildcards within search so a search criteria that we could use is: "*:*"

However, querying and returning every node is not what we need to solve our current problem.

Scenario: We want only to return a subset of our nodes… only the nodes that are web servers.

Demo: View Information for All Nodes

```
> knife search node *:*
```

```
Node Name:    apache_web
Policy Name:  company_web
Policy Group: prod
FQDN:         ip-172-31-57-169.ec2.internal
IP:           34.196.104.17
Run List:     recipe[company_web::default]
Recipes:      company_web::default, apache::default, apache::server
Platform:     centos 7.6.1810
Tags:

Node Name:    iis_web
Policy Name:  company_web
Policy Group: prod
FQDN:         WIN-DQFQCUFHDCP.ec2.internal
IP:           34.195.38.226
Run List:     recipe[company_web::default]
Recipes:      company_web::default, myiis::default, myiis::server
Platform:     windows 6.3.9600
Tags:

Node Name:    lb
Policy Name:  myhaproxy
Policy Group: prod
FQDN:         ip-172-31-22-163.ec2.internal
IP:           34.196.50.77
Run List:     recipe[myhaproxy::default]
Recipes:      myhaproxy::default, haproxy::manual, haproxy::install_package
Platform:     centos 7.6.1810
```

Demo: View Information for All Server Nodes

```
> knife search node policy_name:company_web
```

```
Node Name:   apache_web
Policy Name:  company_web
Policy Group: prod
FQDN:        ip-172-31-57-169.ec2.internal
IP:          34.196.104.17
Run List:    recipe[company_web::default]
Recipes:     company_web::default, apache::default, apache::server
Platform:    centos 7.6.1810
Tags:

Node Name:   iis_web
Policy Name:  company_web
Policy Group: prod
FQDN:        WIN-DQFQCUFHDCP.ec2.internal
IP:          34.195.38.226
Run List:    recipe[company_web::default]
Recipes:     company_web::default, myiis::default, myiis::server
Platform:    windows 6.3.9600
Tags:
```

CHEF

Demo: Return Public Hostname and IP for Servers

```
> knife search node policy_name:company_web -a cloud
```

```
apache_web:
  cloud:
    local_hostname:    ip-172-31-57-169.ec2.internal
    local_ipv4:        172.31.57.169
    local_ipv4_addrs:  172.31.57.169
    provider:          ec2
    public_hostname:   ec2-34-196-104-17.compute-1.amazonaws.com
    public_ipv4:       34.196.104.17
    public_ipv4_addrs: 34.196.104.17

iis_web:
  cloud:
    local_hostname:    ip-172-31-62-51.ec2.internal
    local_ipv4:        172.31.62.51
    local_ipv4_addrs:  172.31.62.51
    provider:          ec2
    public_hostname:   ec2-34-195-38-226.compute-1.amazonaws.com
    public_ipv4:       34.195.38.226
    public_ipv4_addrs: 34.195.38.226
```

Demo: Return Public Hostname for Servers

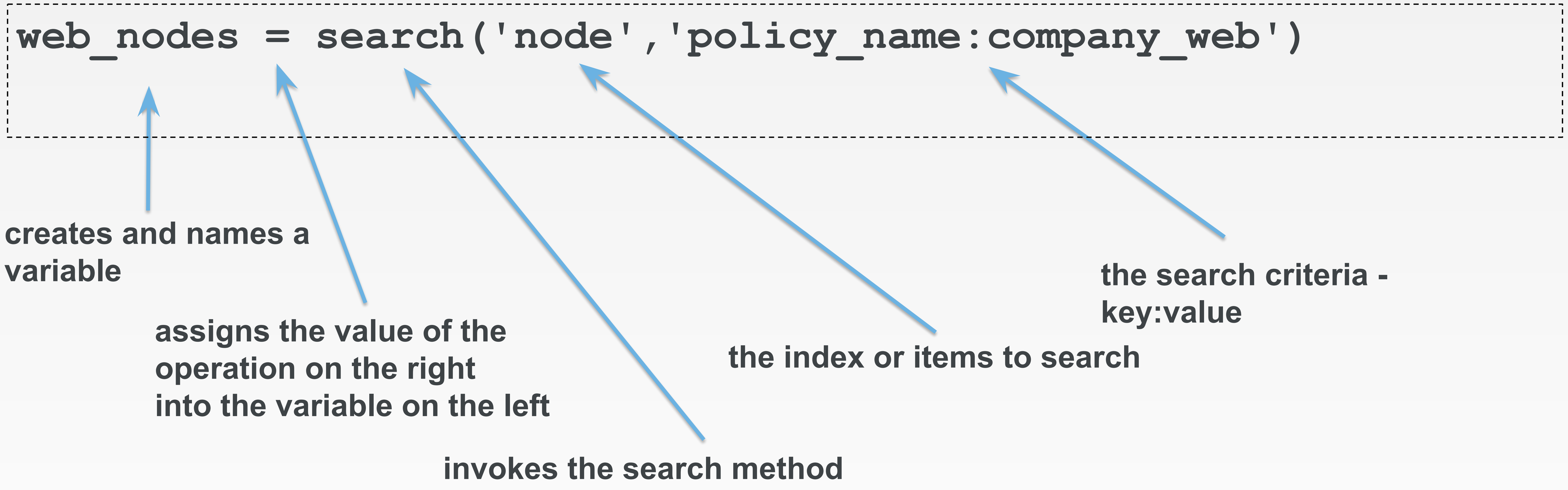
```
> knife search node policy_name:company_web -a cloud.public_hostname
```

```
2 items found

apache_web:
  cloud.public_hostname: ec2-34-196-104-17.compute-1.amazonaws.com

iis_web:
  cloud.public_hostname: ec2-34-195-38-226.compute-1.amazonaws.com
```

Search Syntax within a Recipe



```
web_nodes = search('node','policy_name:company_web')
```

creates and names a variable
assigns the value of the operation on the right into the variable on the left
invokes the search method
the index or items to search
the search criteria - key:value

The search syntax within a recipe differs from the search syntax when using `knife search` from the command line.

Hard Coding Example

```
haproxy_install 'package'

haproxy_frontend 'http-in' do
  bind '*:80'
  default_backend 'server_backend'
end

haproxy_backend 'server_backend' do
  server [
  'ec2-54-175-46-24.compute-1.amazonaws.com 54.175.46.24:80 maxconn 32',
  'ec2-34-196-10-17.compute-1.amazonaws.com 34.196.10.17:80 maxconn 32'
  ]
end

haproxy_service 'haproxy'
```

EXERCISE

GL: Dynamic Web Load Balancer

Every time we create a web node we need to update our load balancer (myhaproxy) cookbook. That doesn't feel right!

Objective:

- ❑ Update the myhaproxy cookbook to dynamically use nodes with the company_web policy_name.
- ❑ Update the major version of the myhaproxy cookbook
- ❑ Upload the Cookbook
- ❑ Run chef-client on the load balancer node
- ❑ Verify the load balancer node relays requests to both web nodes

GL: Remove the Hard-Coded Members

```
~/chef-repo/cookbooks/myhaproxy/recipes/default.rb
```

```
haproxy_install 'package'

haproxy_frontend 'http-in' do
  bind '*:80'
  default_backend 'server_backend'
end

haproxy_backend 'server_backend' do
  server [
  'ec2-54-175-46-24.compute-1.amazonaws.com 54.175.46.24:80 maxconn 32',
  'ec2-34-196-10-17.compute-1.amazonaws.com 34.196.10.17:80 maxconn 32'
  ]
end

haproxy_service 'haproxy'
```

GL: Add the Search Criteria

~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
...

haproxy_install 'package'

haproxy_frontend 'http-in' do
  bind '*:80'
  default_backend 'server_backend'
end

web_nodes = search('node','policy_name:company_web')
```

Note: We will provide the final recipe in a moment.

GL: Create an Array to Store the Converted Members

```
~/chef-repo/cookbooks/myhaproxy/recipes/default.rb
```

```
...
web_nodes = search('node','policy_name:company_web')

server_array = []


haproxy_backend 'server_backend' do
  server server_array
end

haproxy_service 'haproxy'
```

GL: Create an Array to Store the Converted Members

```
~/chef-repo/cookbooks/myhaproxy/recipes/default.rb
```

```
...
web_nodes = search('node','policy_name:company_web')

server_array = []

web_nodes.each do |one_node|
  one_server = ""
   # TODO: Populate the array with each webserver's hostname and ipaddress
end

haproxy_backend 'server_backend' do
  server server_array
end

haproxy_service 'haproxy'
```

GL: Create an Array to Store the Converted Members

~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
...
web_nodes = search('node','policy_name:company_web')

server_array = []

web_nodes.each do |one_node|

  one_server = "#{one_node['cloud']['public_hostname']} #{one_node['cloud']['public_ipv4']}:80
maxconn 32"
  server_array.push(one_server)

end

haproxy_backend 'server_backend' do
  server server_array
end

haproxy_service 'haproxy'
```

GL: The Final Recipe

~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
haproxy_install 'package'

haproxy_frontend 'http-in' do
  bind '*:80'
  default_backend 'server_backend'
end

web_nodes = search('node','policy_name:company_web')

server_array = []

web_nodes.each do |one_node|
  one_server = "#{one_node['cloud']['public_hostname']} #{one_node['cloud']['public_ipv4']}:80 maxconn
32"
  server_array.push(one_server)
end

haproxy_backend 'server_backend' do
  server server_array
end

haproxy_service 'haproxy'
```

GL: Dynamic Web Load Balancer

Every time we create a web node we need to update our load balancer (myhaproxy) cookbook. That doesn't feel right!

Objective:

✔ Update the myhaproxy cookbook to dynamically use nodes with the company_web policy_name.
❑ Update the major version of the myhaproxy cookbook
❑ Upload the Cookbook
❑ Run chef-client on the load balancer node
❑ Verify the load balancer node relays requests to both web nodes

GL: Updating the Cookbook's Version Number

`~/chef-repo/cookbooks/myhaproxy/metadata.rb`

```ruby
name 'myhaproxy'
maintainer 'The Authors'
maintainer_email 'you@example.com'
license 'All Rights Reserved'
description 'Installs/Configures myhaproxy'
long_description 'Installs/Configures myhaproxy'
version '1.0.0'
chef_version '>= 15.0' if respond_to?(:chef_version)

depends 'haproxy', '~> 8.3.0'
```

Dynamic Web Load Balancer

Every time we create a web node we need to update our load balancer (myhaproxy) cookbook. That doesn't feel right!

Objective:

✔ Update the myhaproxy cookbook to dynamically use nodes with the company_web policy_name.

✔ Update the major version of the myhaproxy cookbook

❑ Update and push the Policyfile

❑ Run chef-client on the load balancer node

❑ Verify the load balancer node relays requests to both web nodes

GL: Ensure You are in the chef-repo

```
$ cd ~/chef-repo
```

GL: Update the Policyfile

```
$ chef update policyfiles/myhaproxy.rb
```

```
Attributes already up to date
Building policy myhaproxy
Expanded run list: recipe[myhaproxy::default]
Caching Cookbooks...
Installing myhaproxy >= 0.0.0 from path
Using      haproxy         8.3.0
Using      build-essential 8.2.1
Using      yum-epel        4.1.4
Using      seven_zip       4.2.2
Using      mingw           2.1.3
Using      windows         6.0.1

Lockfile written to
/Users/sdelfante/chef-repo/policyfiles/myhaproxy.lock.json
Policy revision id:
```

GL: Push the myhaproxy.lock.json to Chef Infra Server

```
$ chef push prod policyfiles/myhaproxy.lock.json
```

```
Uploading policy myhaproxy (08c39ccc8f) to policy group prod
Uploading policy myhaproxy (c0cb162cdb) to policy group prod
Using    build-essential 8.2.1 (4b9d5c72)
Using    haproxy         8.3.0 (1a4f7607)
Using    mingw           2.1.3 (9f5d572c)
Using    myhaproxy       1.0.0 (1a9d7377)
Using    seven_zip       4.2.2 (0e1fed3b)
Using    windows         6.0.1 (042f3380)
Using    yum-epel        4.1.4 (187c02d6)
```

GL: Converging the Load Balancer Node

```
$ knife ssh 'name:lb' -x chef -P PASSWORD 'sudo chef-client'
```

```
ec2-35-170-33-199.compute-1.amazonaws.com Starting Chef Infra Client, version 17.3.48
ec2-35-170-33-199.compute-1.amazonaws.com Using policy 'myhaproxy' at revision
'9da82055ea2c960cd680d131de8e92ba773703538575e4e429a52ca13f01fbba'
ec2-35-170-33-199.compute-1.amazonaws.com resolving cookbooks for run list:
["myhaproxy::default@1.0.0 (e9a41c2)"]
ec2-35-170-33-199.compute-1.amazonaws.com Synchronizing Cookbooks:
ec2-35-170-33-199.compute-1.amazonaws.com   - build-essential (8.2.1)
ec2-35-170-33-199.compute-1.amazonaws.com   - haproxy (8.3.0)
ec2-35-170-33-199.compute-1.amazonaws.com   - mingw (2.1.3)
ec2-35-170-33-199.compute-1.amazonaws.com   - seven_zip (4.2.2)
ec2-35-170-33-199.compute-1.amazonaws.com   - windows (6.0.1)
ec2-35-170-33-199.compute-1.amazonaws.com   - yum-epel (4.1.4)
ec2-35-170-33-199.compute-1.amazonaws.com   - myhaproxy (1.0.0)
...
...
ec2-35-170-33-199.compute-1.amazonaws.com Chef Infra Client finished, 6/26 resources
updated in 05 seconds
```

Dynamic Web Load Balancer

Every time we create a web node we need to update our load balancer (myhaproxy) cookbook. That doesn't feel right!

Objective:

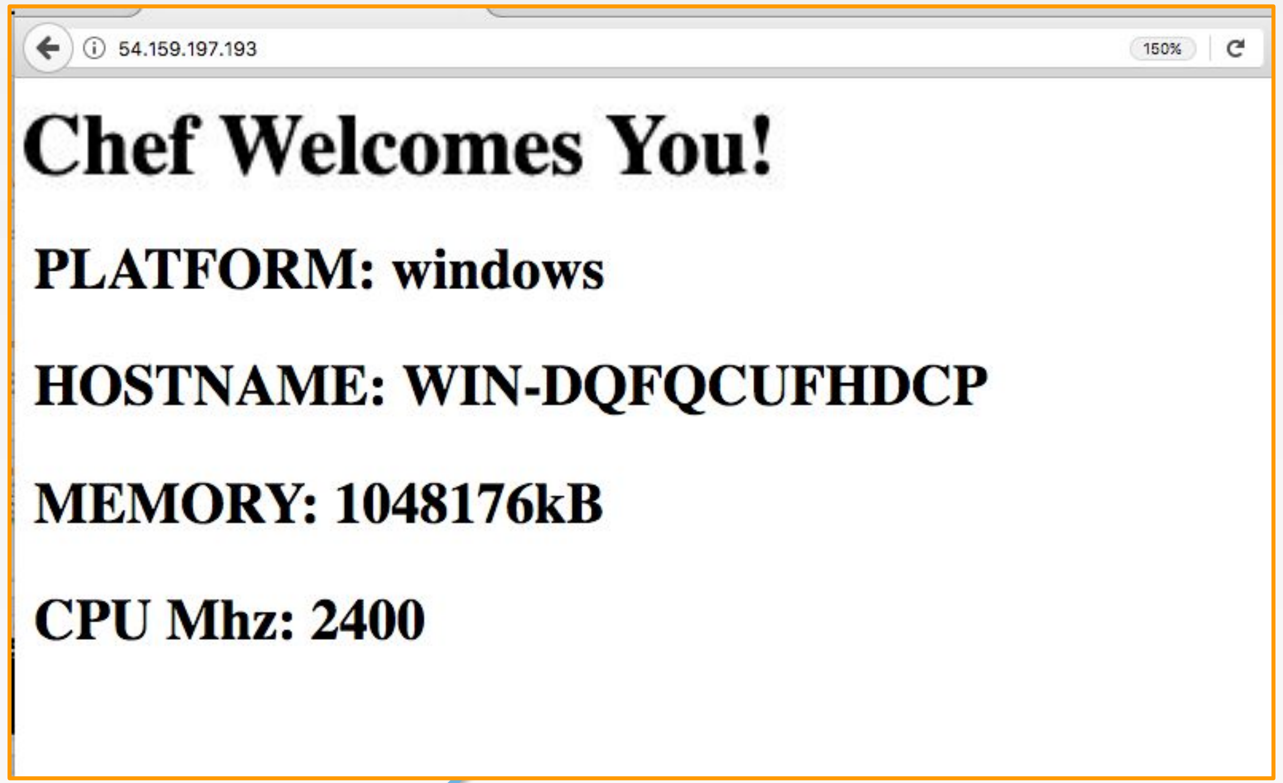
✔ Update the myhaproxy cookbook to dynamically use nodes with the company_web policy_name.

✔ Update the major version of the myhaproxy cookbook

✔ Update and push the Policyfile

✔ Run chef-client on the load balancer node

❑ Verify the load balancer node relays requests to both web nodes



```
54.159.197.193
```

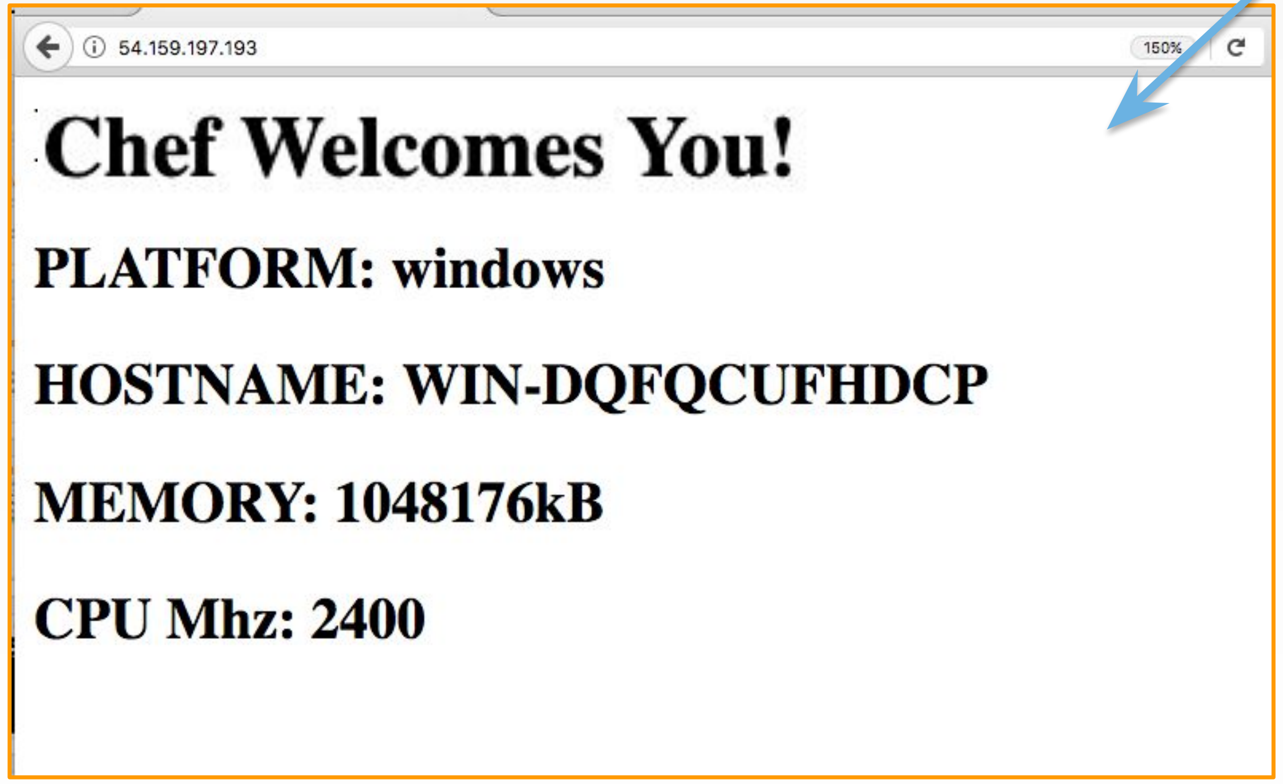
Chef Welcomes You!

PLATFORM: windows

HOSTNAME: WIN-DQFQCUFHDCP

MEMORY: 1048176kB

CPU Mhz: 2400



```
54.159.197.193
```

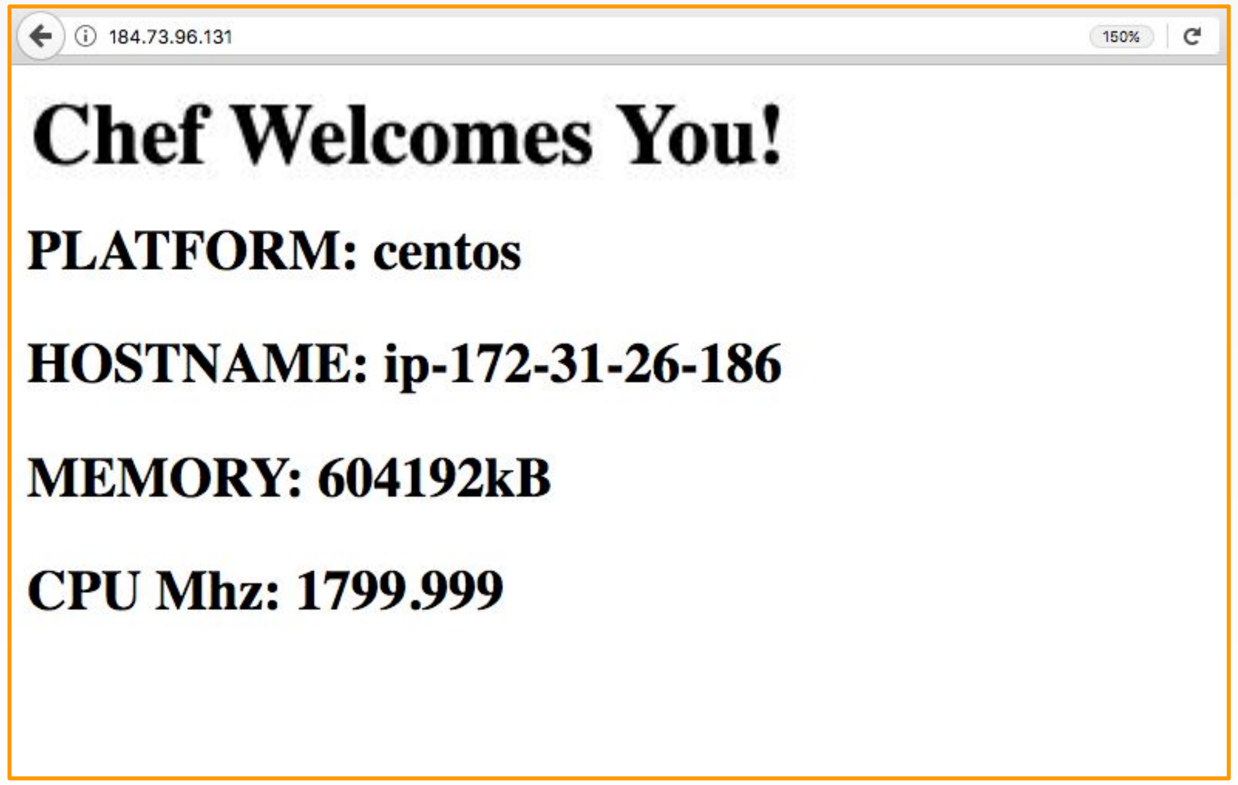
Chef Welcomes You!

PLATFORM: windows

HOSTNAME: WIN-DQFQCUFHDCP

MEMORY: 1048176kB

CPU Mhz: 2400



```
184.73.96.131
```

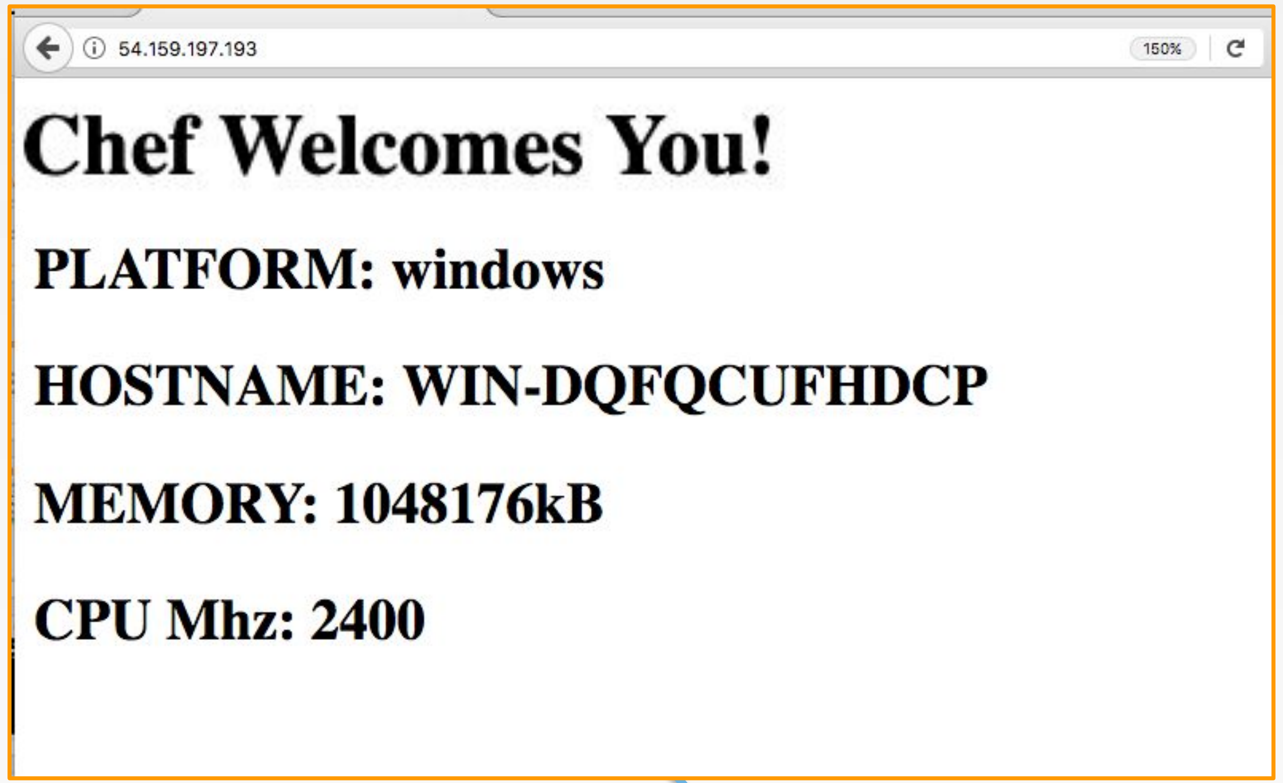
Chef Welcomes You!

PLATFORM: centos

HOSTNAME: ip-172-31-26-186

MEMORY: 604192kB

CPU Mhz: 1799.999


54.159.197.193

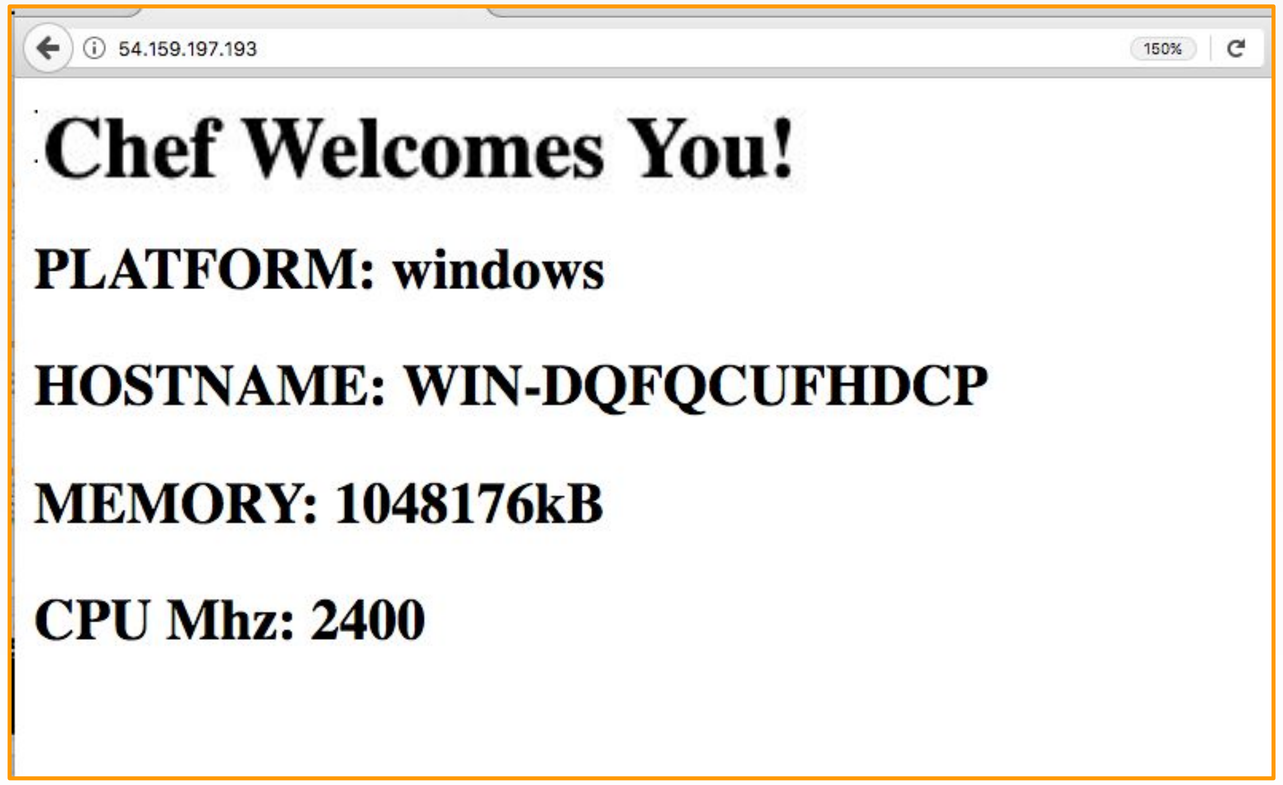
Chef Welcomes You!

PLATFORM: windows

HOSTNAME: WIN-DQFQCUFHDCP

MEMORY: 1048176kB

CPU Mhz: 2400


54.159.197.193

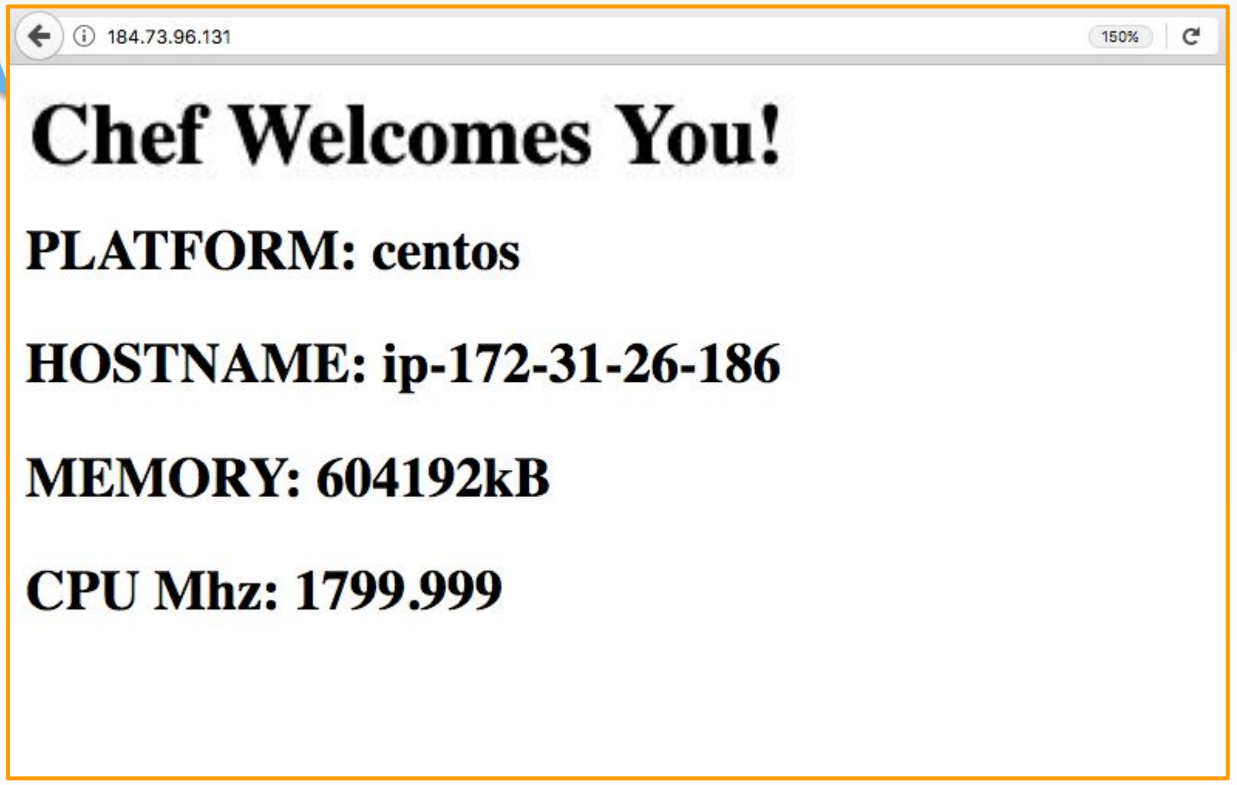
Chef Welcomes You!

PLATFORM: windows

HOSTNAME: WIN-DQFQCUFHDCP

MEMORY: 1048176kB

CPU Mhz: 2400


184.73.96.131

Chef Welcomes You!

PLATFORM: centos

HOSTNAME: ip-172-31-26-186

MEMORY: 604192kB

CPU Mhz: 1799.999

Let's Test that Our Code Really Works

To verify that our code is working, let's remove the load balancer configuration file, forcing our code to run.

GL: Delete the haproxy.cfg File

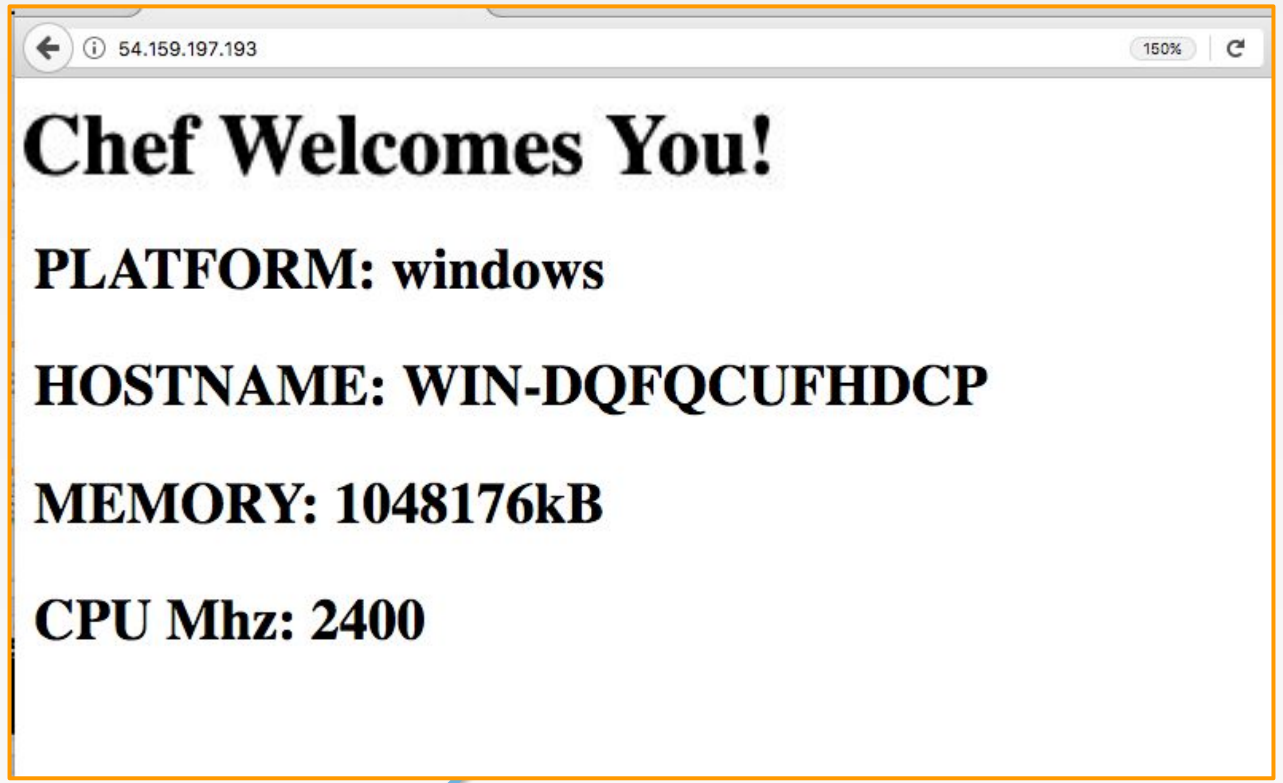
```
$ knife ssh 'name:lb' -x chef -P PASSWORD "sudo rm
/etc/haproxy/haproxy.cfg"
```

GL: Converge the Load Balancer

```
$ knife ssh 'name:lb' -x chef -P PASSWORD 'sudo chef-client'
```

```
ec2-34-196-50-77.compute-1.amazonaws.com Starting Chef Infra Client, version 17.3.48
...
ec2-34-196-50-77.compute-1.amazonaws.com resolving cookbooks for run list:
["myhaproxy::default@1.0.0 (e9a41c2)"]
ec2-34-196-50-77.compute-1.amazonaws.com Synchronizing Cookbooks:
...
ec2-34-196-50-77.compute-1.amazonaws.com * template[/etc/haproxy/haproxy.cfg] action
create
ec2-34-196-50-77.compute-1.amazonaws.com    - create new file /etc/haproxy/haproxy.cfg
ec2-34-196-50-77.compute-1.amazonaws.com    - update content in file
/etc/haproxy/haproxy.cfg from none to 4334de
ec2-34-196-50-77.compute-1.amazonaws.com    - suppressed sensitive resource
ec2-34-196-50-77.compute-1.amazonaws.com    - change mode from '' to '0644'
ec2-34-196-50-77.compute-1.amazonaws.com    - change owner from '' to 'haproxy'
ec2-34-196-50-77.compute-1.amazonaws.com    - change group from '' to 'haproxy'
...
```

CHEF


54.159.197.193 150%

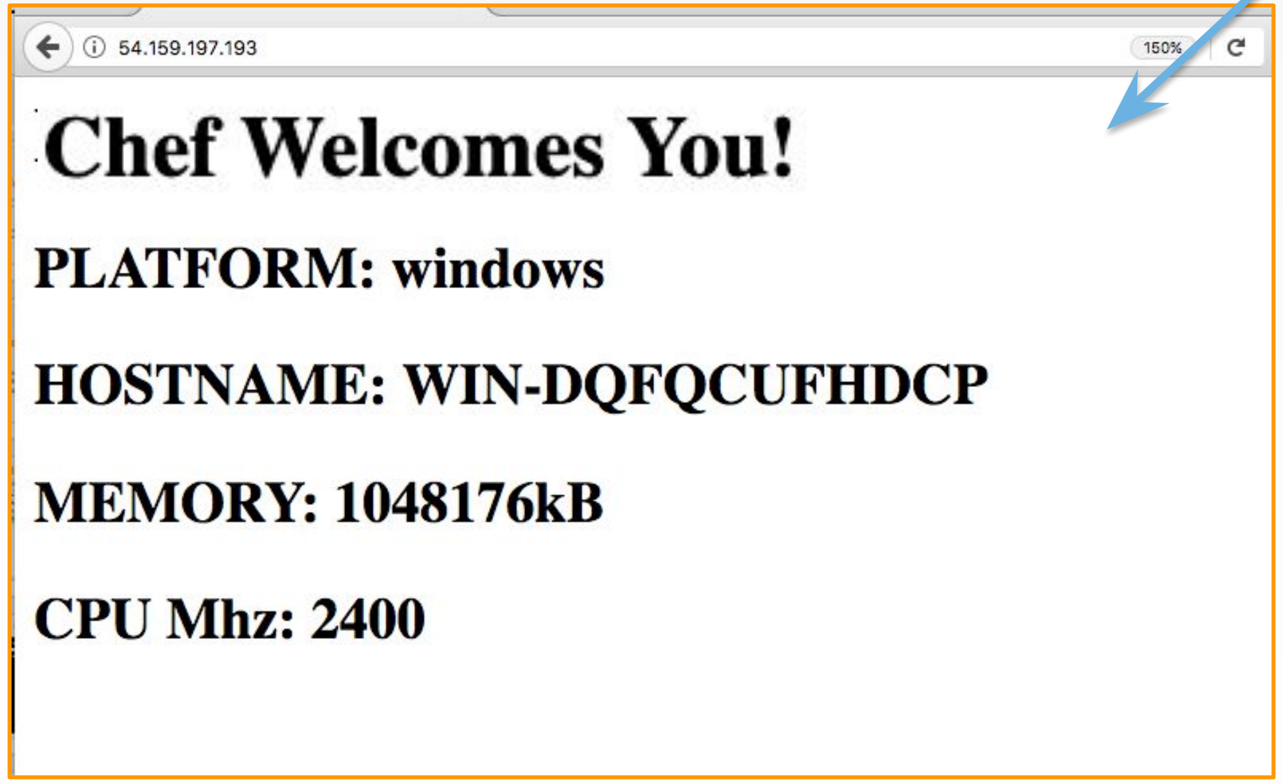
Chef Welcomes You!

PLATFORM: windows

HOSTNAME: WIN-DQFQCUFHDCP

MEMORY: 1048176kB

CPU Mhz: 2400


54.159.197.193 150%

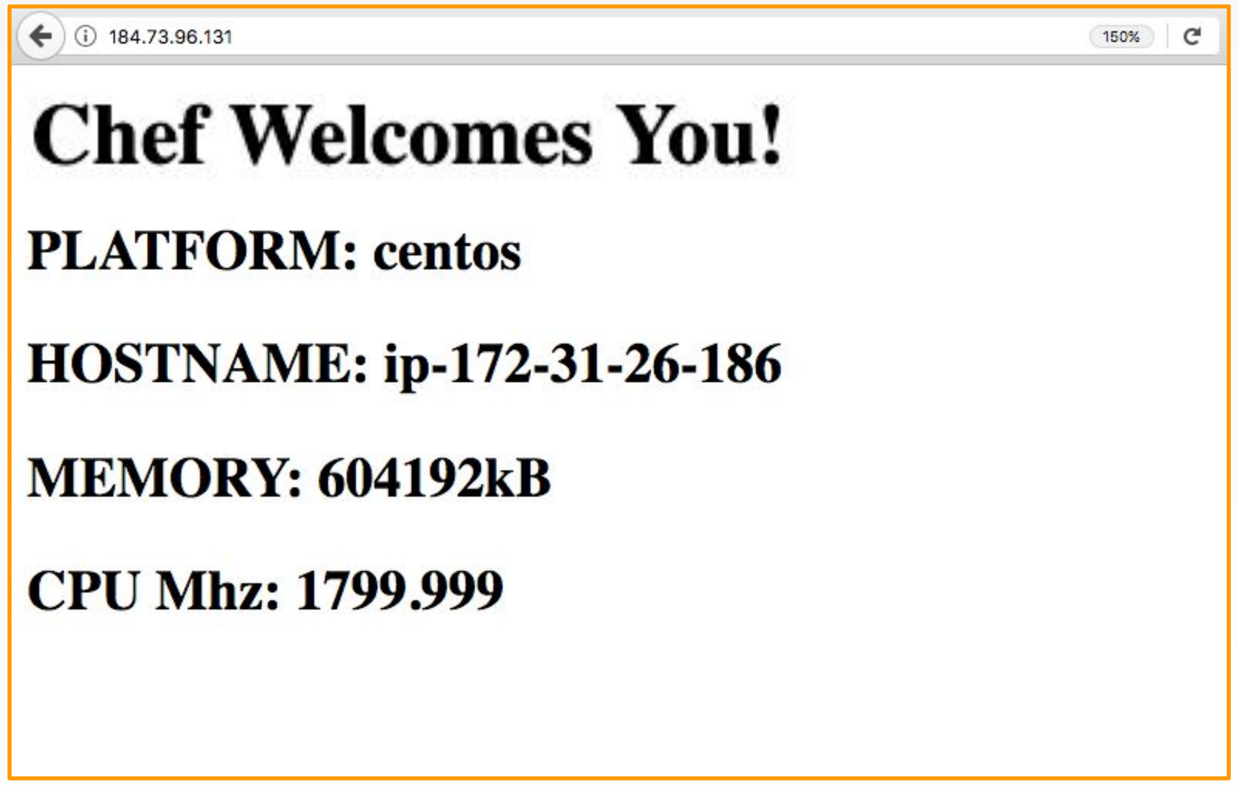
Chef Welcomes You!

PLATFORM: windows

HOSTNAME: WIN-DQFQCUFHDCP

MEMORY: 1048176kB

CPU Mhz: 2400


184.73.96.131 150%

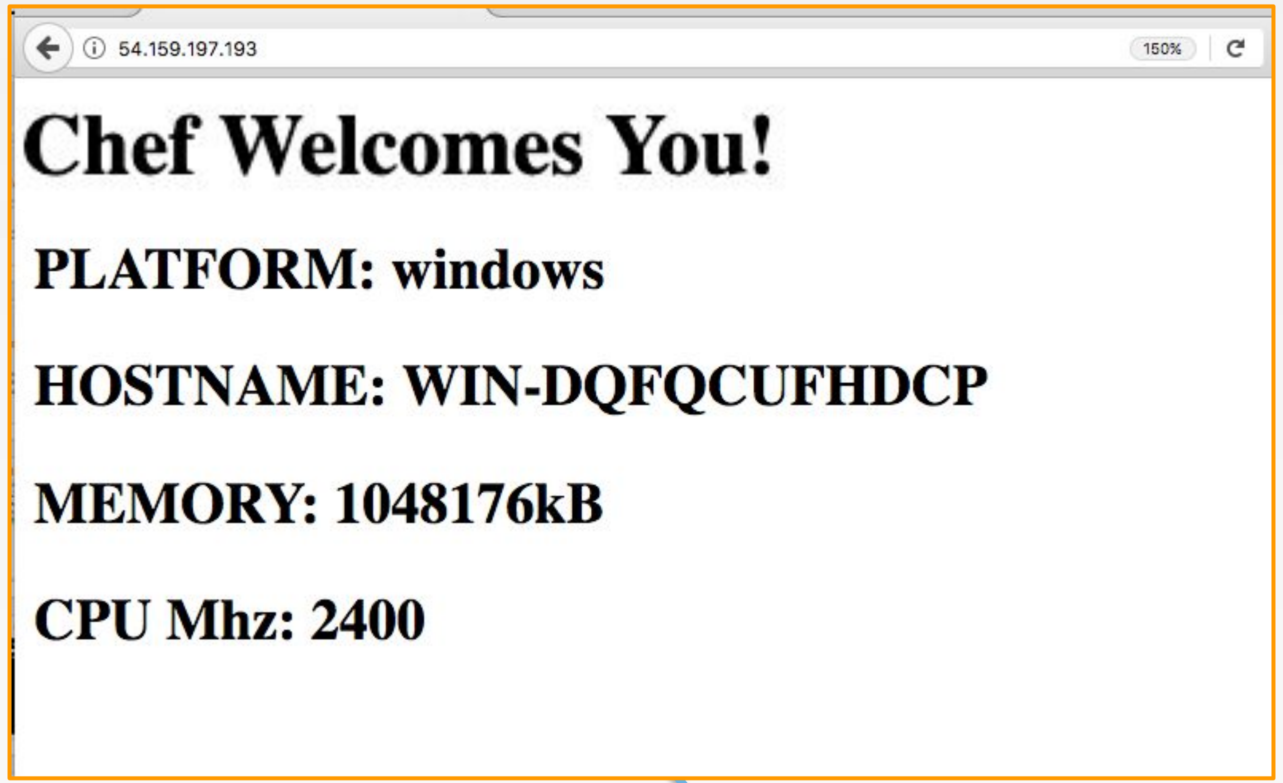
Chef Welcomes You!

PLATFORM: centos

HOSTNAME: ip-172-31-26-186

MEMORY: 604192kB

CPU Mhz: 1799.999



54.159.197.193

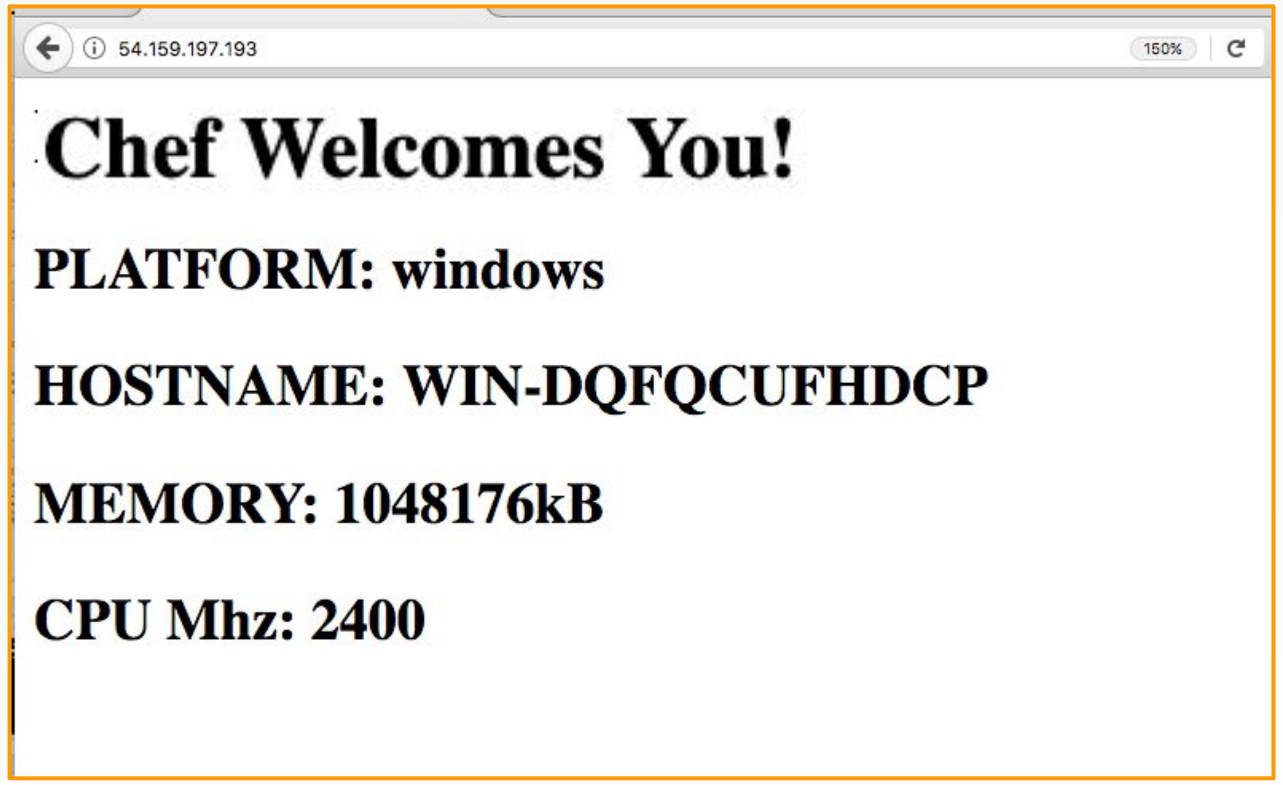
Chef Welcomes You!

PLATFORM: windows

HOSTNAME: WIN-DQFQCUFHDCP

MEMORY: 1048176kB

CPU Mhz: 2400



54.159.197.193

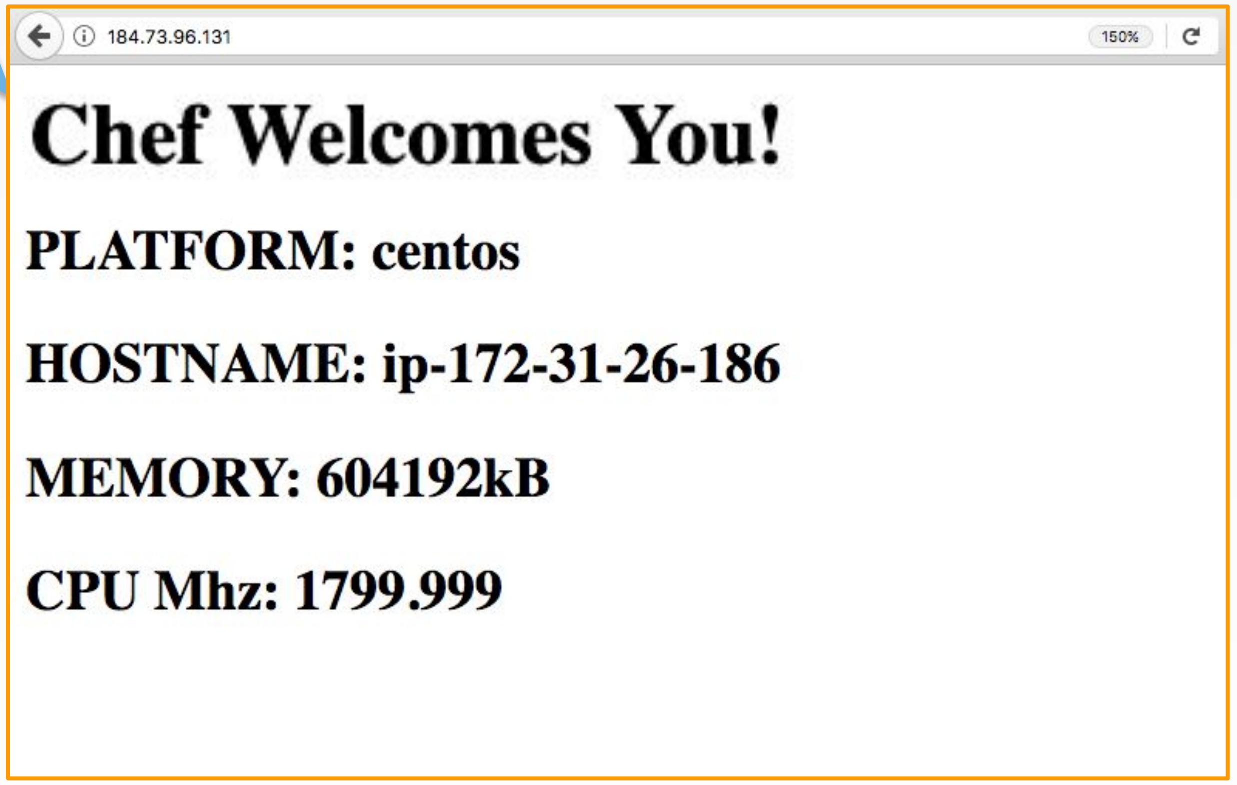
Chef Welcomes You!

PLATFORM: windows

HOSTNAME: WIN-DQFQCUFHDCP

MEMORY: 1048176kB

CPU Mhz: 2400



184.73.96.131

Chef Welcomes You!

PLATFORM: centos

HOSTNAME: ip-172-31-26-186

MEMORY: 604192kB

CPU Mhz: 1799.999

Dynamic Web Load Balancer

Every time we create a web node we need to update our load balancer (myhaproxy) cookbook. That doesn't feel right!

Objective:

✔ Update the myhaproxy cookbook to dynamically use nodes with the company_web policy_name.
✔ Update the major version of the myhaproxy cookbook
✔ Update and push the Policyfile
✔ Run chef-client on the load balancer node
✔ Verify the load balancer node relays requests to both web nodes

Review Questions

1. What is the great advantage of using the following dynamic search in the load balancer's default.rb?

```
web_nodes = search('node','policy_name:company_web')

server_array = []

web_nodes.each do |one_node|
  one_server = "#{one_node['cloud']['public_hostname']}
#{one_node['cloud']['public_ipv4']}:80 maxconn 32"
  server_array.push(one_server)
end
. . .
. . .
```

Review Questions

2. What is the key item that that tells the load balancer how to find the web servers it’s supposed to balance?

```
web_nodes = search('node','policy_name:company_web')

server_array = []

web_nodes.each do |one_node|
  one_server = "#{one_node['cloud']['public_hostname']}
#{one_node['cloud']['public_ipv4']}:80 maxconn 32"
  server_array.push(one_server)
end
...
```

DISCUSSION

Q&A

What questions can we help you answer?

CHEF™

CHEF™