Search

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

CHEF™

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 Array and Ruby Hash dynamically
- Test that your load balancer is still balancing traffic

CHEF™

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.

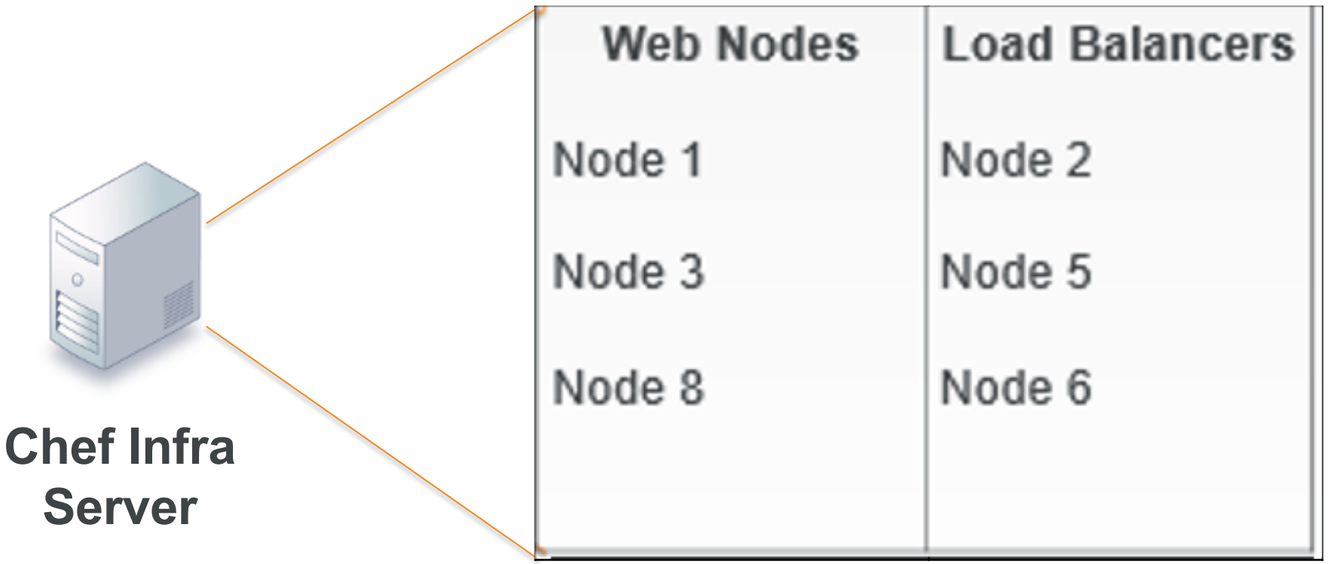
That seems inefficient to have to update a cookbook recipe.

CHEF™

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.

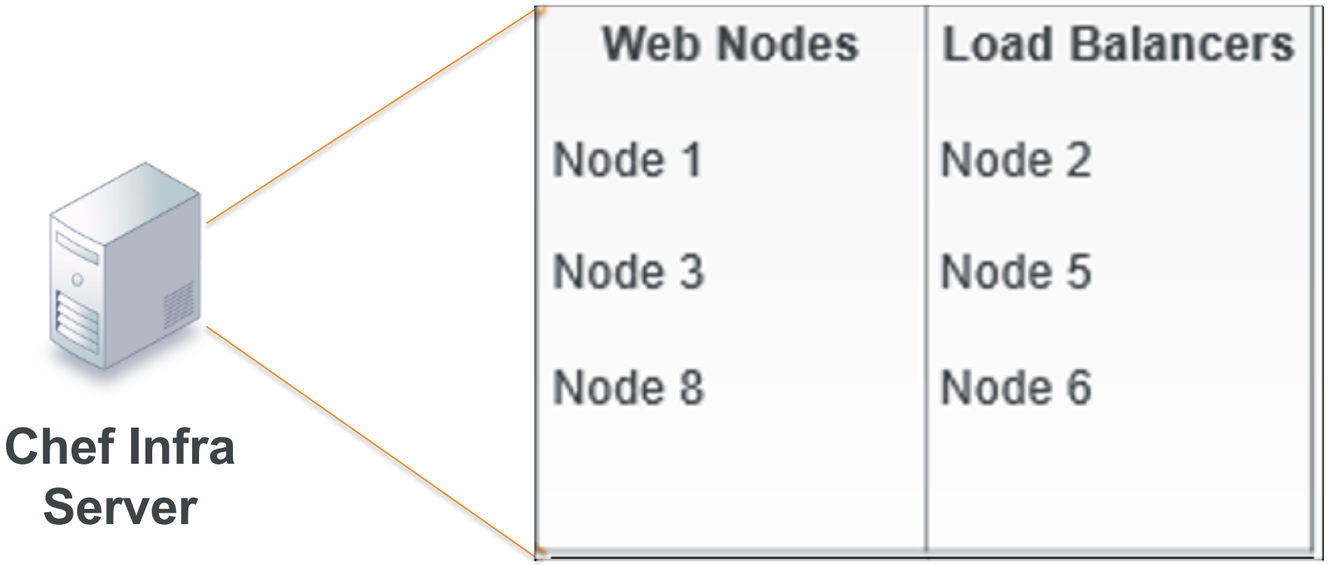


https://docs.chef.io/chef_search.html

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`.



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,

CHEF™

Search Criteria - Query

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.

CHEF

Demo: View Information for All Nodes

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

```
3 items found

Node Name:   node1
Policy Name:  company_web
Policy Group: prod
FQDN:        ip-172-31-18-33.us-east-2.compute.internal
IP:          3.16.131.207
Run List:    recipe[company_web::default]
Recipes:     company_web::default, apache::default, apache::server
Platform:    centos 7.8.2003
Tags:

Node Name:   node3
Policy Name:  company_web
Policy Group: prod
FQDN:        ip-172-31-24-5.us-east-2.compute.internal
IP:          52.15.221.52
Run List:    recipe[company_web::default]
Recipes:     company_web::default, apache::default, apache::server
Platform:    centos 7.8.2003
Tags:

Node Name:   node2
Policy Name:  haproxy
Policy Group: prod
FQDN:        ip-172-31-19-183.us-east-2.compute.internal
IP:          18.220.243.173
Run List:    recipe[haproxy::default]
Recipes:     haproxy::default
Platform:    centos 7.8.2003
```

Demo: Filter by one query

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

```
2 items found

Node Name:   node1
Policy Name:  company_web
Policy Group: prod
FQDN:        ip-172-31-18-33.us-east-2.compute.internal
IP:          3.16.131.207
Run List:    recipe[company_web::default]
Recipes:     company_web::default, apache::default, apache::server
Platform:    centos 7.8.2003
Tags:

Node Name:   node3
Policy Name:  company_web
Policy Group: prod
FQDN:        ip-172-31-24-5.us-east-2.compute.internal
IP:          52.15.221.52
Run List:    recipe[company_web::default]
Recipes:     company_web::default, apache::default, apache::server
Platform:    centos 7.8.2003
Tags:
```

CHEF

Demo: Filter by multiple query

```
> knife search node "policy_name:company_web AND policy_group:prod"
```

```
2 items found

Node Name:   node1
Policy Name:  company_web
Policy Group: prod
FQDN:        ip-172-31-18-33.us-east-2.compute.internal
IP:          3.16.131.207
Run List:    recipe[company_web::default]
Recipes:     company_web::default, apache::default, apache::server
Platform:    centos 7.8.2003
Tags:

Node Name:   node3
Policy Name:  company_web
Policy Group: prod
FQDN:        ip-172-31-24-5.us-east-2.compute.internal
IP:          52.15.221.52
Run List:    recipe[company_web::default]
Recipes:     company_web::default, apache::default, apache::server
Platform:    centos 7.8.2003
```

CHEF

Demo: Filter by multiple query

```
> knife search node "policy_name:company_web AND
policy_group:dev"
```

```
0 items found
```

CHEF

Search Criteria Attributes

- Passing attributes
- Multiple attributes
- Nested attributes

Demo: Attributes

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

```
2 items found

node1:
  cloud:
    local_hostname:    ip-172-31-18-33.us-east-2.compute.internal
    local_ipv4:        172.31.18.33
    local_ipv4_addrs:  172.31.18.33
    provider:          ec2
    public_hostname:   ec2-3-16-131-207.us-east-2.compute.amazonaws.com
    public_ipv4:       3.16.131.207
    public_ipv4_addrs: 3.16.131.207

node3:
  cloud:
    local_hostname:    ip-172-31-24-5.us-east-2.compute.internal
    local_ipv4:        172.31.24.5
    local_ipv4_addrs:  172.31.24.5
    provider:          ec2
    public_hostname:   ec2-52-15-221-52.us-east-2.compute.amazonaws.com
    public_ipv4:       52.15.221.52
    public_ipv4_addrs: 52.15.221.52
```

Demo: Nested Attributes

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

```
2 items found

node1:
  cloud.public_hostname: ec2-3-16-131-207.us-east-
2.compute.amazonaws.com

node3:
  cloud.public_hostname: ec2-52-15-221-52.us-east-
2.compute.amazonaws.com
```

CHEF

Demo: Multiple Attributes

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

```
2 items found

node1:
  cloud.local_hostname:  ip-172-31-18-33.us-east-2.compute.internal
  cloud.public_hostname: ec2-3-16-131-207.us-east-
2.compute.amazonaws.com

node3:
  cloud.local_hostname:  ip-172-31-24-5.us-east-2.compute.internal
  cloud.public_hostname: ec2-52-15-221-52.us-east-
2.compute.amazonaws.com
```

CHEF

Search Criteria – Matching Patterns

- Exact Matching
- Not Matching
- Wildcard Matching
- Range Matching

Demo: Exact Matching

```
> knife search node name:node1
```

```
1 items found

Node Name:   node1
Policy Name:  company_web
Policy Group: prod
FQDN:        ip-172-31-18-33.us-east-2.compute.internal
IP:          3.16.131.207
Run List:    recipe[company_web::default]
Recipes:     company_web::default, apache::default, apache::server
Platform:    centos 7.8.2003
Tags:
```

CHEF

Demo: Not Matching

```
> knife search node 'NOT name:node1' -i
```

```
2 items found

node3
node2
```

Demo: Exact Matching on nested field

```
> knife search node 'cloud_public_ipv4:3.16.131.207'
```

```
1 items found

Node Name:   node1
Policy Name:  company_web
Policy Group: prod
FQDN:        ip-172-31-18-33.us-east-2.compute.internal
IP:          3.16.131.207
Run List:    recipe[company_web::default]
Recipes:     company_web::default, apache::default, apache::server
Platform:    centos 7.8.2003
Tags:
```

. Changes to _ in query

CHEF

Demo: Wildcard Matching

```
> knife search node name:node* -i
```

```
3 items found

node1
node3
node2
```

Returns only id

* matches any characters any times
? matches any characters one time

Demo: Range Matching

```
> knife search node "name:[node1 TO node2]" -i
```

```
2 items found

node1
node2
```

Range query
name:[node3 TO node1] – no result

Internally, this Solr gets executed

https://api.chef.io/organizations/aspechef23/search/node?q=name:%5Bnode1%20TO%20node2%5D&start=0&rows=1000

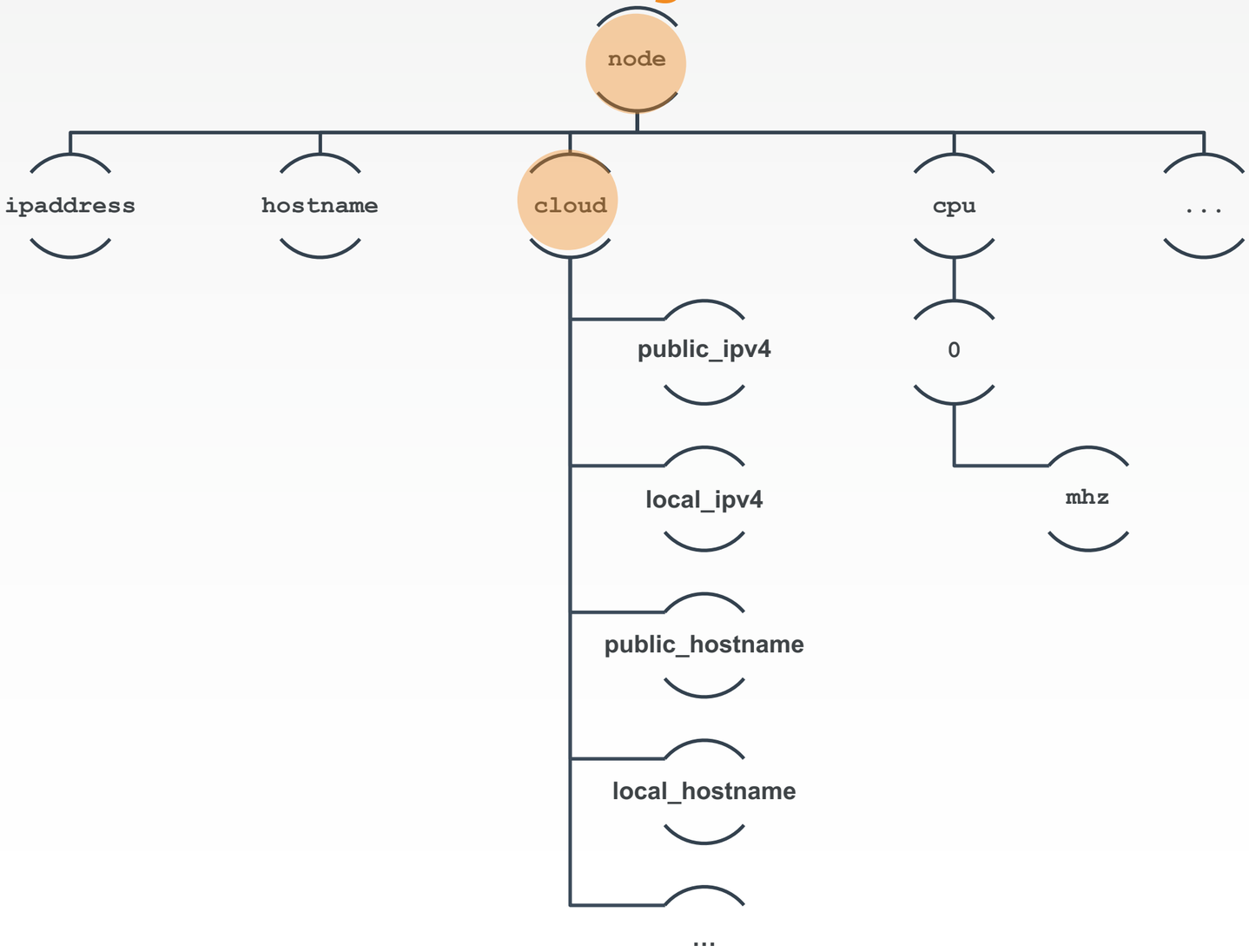
CHEF

DISCUSSION

Goal:
Use knife search to dynamically change haproxy servers

CHEF

EC2 and the Node Object



HAProxy Configuration Should Look Like This

```
...
backend app
  balance roundrobin
  server app0 <<node1 IP ADDRESS>>:80 weight 1 maxconn 100 check
  server app1 <<node2 IP ADDRESS>>:80 weight 1 maxconn 100 check
```

Values from

```
knife search node 'policy_name:company_web AND policy_group:prod' -
a cloud.public_ipv4
```


CHEF

Search for relevant nodes

```
$ knife search node 'policy_name:company_web AND policy_group:prod'
-a cloud.public_ipv4
```

```
2 items found


node1:
  cloud.public_ipv4: 3.16.131.207


node3:
  cloud.public_ipv4: 52.15.221.52
```

These IP Addresses are not accessible from the outside network

GL: Edit haproxy cookbook's default recipe

chef-repo/cookbooks/haproxy/recipes/default.rb

https://github.com/shekhar2010us/chef-server-example/blob/master/cookbooks/haproxy/recipes/default.rb

```
package 'haproxy'

allwebservers = search('node', 'policy_name:company_web AND policy_group:prod')

template '/etc/haproxy/haproxy.cfg' do
  source 'haproxy.cfg.erb'
  owner "root"
  group "root"
  mode 0644
  variables(
    :webservers => allwebservers
  )
  notifies :restart, "service[haproxy]"
end

service "haproxy" do
  supports :restart => true, :status => true, :reload => true
  action [:enable, :start]
end
```

GL: Configuring haproxy.cfg.erb

```
cookbooks/haproxy/templates/haproxy.cfg.erb
```

```
listen application 0.0.0.0:80
  balance roundrobin
  <% @webservers.each_with_index do |web, n| -%>
    server <%= "webserver#{n}" %> <%= web['cloud']['public_ipv4'] %>:80
weight 1 maxconn 100 check
  <% end -%>
```

Default

https://github.com/shekhar2010us/chef-server-example/blob/master/haproxy.cfg.original

Copy the URL from below the slide in your participant guide:

https://github.com/shekhar2010us/chef-server-example/blob/master/cookbooks/haproxy/templates/haproxy.cfg.erb

CHEF

Upgrade haproxy version in metadata

```
$ ~/chef-repo/cookbooks/haproxy/metadata.rb
```

```
Earlier it was
  version '0.2.0'
Make it
  version '0.3.0’
Or whatever version you have, upgrade by 1
```

Update Policyfile

```
$ chef update haproxy.rb
```

```
Attributes already up to date
Building policy haproxy
Expanded run list: recipe[haproxy::default]
Caching Cookbooks...
Installing haproxy >= 0.0.0 from path

Lockfile written to /home/centos/chef-repo/haproxy.lock.json
Policy revision id: 252dc5bd70caffae17b0d362f77b562a33822c0b521b017e2cbe7decf5b7bdcf
```

Push Policyfile to Server

```
$ chef push prod haproxy.lock.json
```

```
Uploading policy haproxy (252dc5bd70) to policy group prod
Uploaded haproxy 0.3.0 (b729718e)
```

Check the Node

```
$ knife node show node2
```

```
Node Name:   node2
Policy Name:  haproxy
Policy Group: prod
FQDN:        ip-172-31-19-183.us-east-2.compute.internal
IP:          18.220.243.173
Run List:    recipe[haproxy::default]
Recipes:     haproxy::default
Platform:    centos 7.8.2003
Tags:
```

Apply new policy to the node

```
$ knife ssh <haproxy node ip> -m -x centos -i <aws.pem path> "sudo chef-client"
```

```
18.220.243.173 Starting Chef Infra Client, version 15.10.12
18.220.243.173 Using policy 'haproxy' at revision
'252dc5bd70caffae17b0d362f77b562a33822c0b521b017e2cbe7decf5b7bdcf'
18.220.243.173 resolving cookbooks for run list: ["haproxy::default@0.3.0
(b729718)"]
18.220.243.173 Synchronizing Cookbooks:
18.220.243.173   - haproxy (0.3.0)
18.220.243.173 Installing Cookbook Gems:
18.220.243.173 Compiling Cookbooks...
18.220.243.173 Converging 3 resources
18.220.243.173 Recipe: haproxy::default
18.220.243.173   * yum_package[haproxy] action install (u
……….
```

Test load balancer

Load Balancer

My company Welcomes You!

PLATFORM: centos

HOSTNAME: ip-172-31-24-5

MEMORY: 1880312kB

CPU Mhz: 2300.202

My company Welcomes You!

PLATFORM: centos

HOSTNAME: ip-172-31-18-33

MEMORY: 1880312kB

CPU Mhz: 2300.117

CHEF

Efficient ssh commands

```
$ knife ssh <haproxy node ip> -m -x centos -i <aws.pem path> "sudo chef-client"
```

```
$ knife ssh "policy_name:haproxy AND policy_group:prod" -x centos -i <aws.pem path>
"sudo chef-client"
```

Other ssh commands

From workstation, cat haproxy cfg file from the node

```
$ knife ssh "policy_name:haproxy AND policy_group:prod" -x centos -i ~/aws.pem "cat /etc/haproxy/haproxy.cfg"
```

From workstation, delete haproxy cfg file from the node

```
$ knife ssh "policy_name:haproxy AND policy_group:prod" -x centos -i ~/aws.pem "sudo rm -rf /etc/haproxy/haproxy.cfg"
```

From workstation, check again haproxy cfg

```
$ knife ssh "policy_name:haproxy AND policy_group:prod" -x centos -i ~/aws.pem "cat /etc/haproxy/haproxy.cfg"
```

From workstation, execute chef-client

```
$ knife ssh "policy_name:haproxy AND policy_group:prod" -x centos -i ~/aws.pem "sudo chef-client"
```

Test Load Balancer Again

Load Balancer

My company Welcomes You!

PLATFORM: centos

HOSTNAME: ip-172-31-24-5

MEMORY: 1880312kB

CPU Mhz: 2300.202

My company Welcomes You!

PLATFORM: centos

HOSTNAME: ip-172-31-18-33

MEMORY: 1880312kB

CPU Mhz: 2300.117

CHEF

DISCUSSION

Q&A

What questions can we help you answer?

- Search
- Passing variables into templates

CHEF

CHEF™