Really large scale systems configuration

Config Management @ Facebook
Phil Dibowitz
Who am I?

Configuration Management Experience
- Authored Provision

Scale Experience
- Ticketmaster, Google, Facebook

Passionate about scaling configuration management
Scaling
Scaling Configuration Management

How many homogeneous systems can you maintain?
How many heterogeneous systems can you maintain?
How many people are needed?
Can you safely delegate delta configuration?
The Goal

http://www.prathiphwani.org/sports/goal-2012/the-beautiful-game-is-back/
The Goal

- 4 people
- Tens of thousands of heterogeneous systems
- Service owners own/adjust relevant settings
What did we need?
1. Basic Scalable Building Blocks
## 1. Basic Scalable Build Blocks

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Distributed!</strong></td>
<td>Everything on the client (duh!)</td>
</tr>
<tr>
<td><strong>Deterministic!</strong></td>
<td>The system you want on every run</td>
</tr>
<tr>
<td><strong>Idempotent!</strong></td>
<td>Only the necessary changes</td>
</tr>
<tr>
<td><strong>Extensible!</strong></td>
<td>Tied into internal systems</td>
</tr>
<tr>
<td><strong>Flexible!</strong></td>
<td>No dictated workflow</td>
</tr>
</tbody>
</table>
2. Configuration as Data
2. Configuration as Data

I want
• shared mem
• VIP
• core files somewhere else
• service running
• less/more/no nscd caching
2. Configuration as Data

Service Owners don’t know:

- How to configure VIPs
- Optimal sysctl settings
- Network settings
- Authentication settings
3. Flexibility
3. Flexibility

- Adapt to our workflow
- Super-fast prototyping
- Internal assumptions can be changed - easily
- Extend in new ways - easily
3. Flexibility - Example

- Template /etc/sysctl.conf
- Build a hash of default sysctls
- Provide these defaults early in “run”
- Let any engineer munge the bits they want
- /etc/sysctl.conf template interpolated “after”
Picking a tool
Why Chef?

Easier to see from a problem with Chef
Problem (for us): node.save()

- Scale issues with 15k-node clusters
- Standard solution sucks (disable ohai plugins)
Solution (for us): whitelist_node_attrs

- New cookbook re-opens Chef::Node.save
- Deletes non-white-listed attrs before saving
- Have as much data as you want during the run
- We send < 1kb back to the server!

Code available:
https://github.com/opscode-cookbooks/whitelist-node-attrs
class Chef

class Node
  alias_method :old_save, :save
  # Overwrite chef’s node.save to whitelist. doesn’t get “later” than this
  def save
    Chef::Log.info("Whitelisting node attributes")
    whitelist = self[:whitelist].to_hash
    self.default_attrs = Whitelist.filter(self.default_attrs, whitelist)
    self.normal_attrs = Whitelist.filter(self.normal_attrs, whitelist)
    self.override_attrs = Whitelist.filter(self.override_attrs, whitelist)
    self.automatic_attrs = Whitelist.filter(self.override_attrs, whitelist)
    old_save
  end
end
end
Chef: whitelist_node_attrs

Well... that’s flexible!

(we did this a few times)
Our desired workflow
Our Desired Workflow

• Provide API for anyone, anywhere to extend configs by munging data structures
• Engineers don’t need to know what they’re building on, just what they want to change
• Engineers can change their systems without fear of changing anything else
• Testing should be easy
• And...
Something Different

Moving Idempotency “up”

http://www.flickr.com/photos/esi_design/4548531839/sizes/l/in/photostream/
Moving Idempotency Up

- Idempotent records can get stale
- Remove cron/sysctl/user/etc. resource
- Resulting entry **not** removed => stale entries

- Idempotent systems control a set of configs
- Remove cron/sysctl/user/etc.
- No longer rendered in config
Idempotent Records vs. Systems

This is a pain:

cron 'tmp_cleaner' do
  minute '5'
  command '/usr/local/bin/tmp_cleaner'
end

cron 'tmp_cleaner' do
  minute '5'
  command '/usr/local/bin/tmp_cleaner'
  action :delete
end

# delete after 3/1/13
user 'coolsoftd' do
  uid 512
  home '/var/coolsoftd'
end

# delete after 3/1/13
user 'coolsoftd' do
  uid 512
  home '/var/coolsoftd'
  action :delete
end
Idempotent Records vs. Systems

This is better:

```ruby
1  cron 'tmp_cleaner' do
2    minute '5'
3    command '/usr/local/bin/tmp_cleaner'
4  end

5  user 'coolsoftd' do
6    uid 512
7    home '/var/coolsoftd'
8  end
```
Case Study
Case Study: sysctl

- fb_sysctl/attributes/default.rb
- Provides defaults looking at hw, kernel, etc.
- fb_sysctl/recipes/default.rb
- Defines a template
- fb_sysctl/templates/default/sysctl.conf.erb
- 3-line template
Case Study: sysctl

Template:

```ruby
# Generated by Chef, do not edit directly!
<%- node['fb']['fb_sysctl'].to_hash.keys.sort.each do |key| %>
<%= key %> = <%= node['fb']['fb_sysctl'][key] %>
<%- end %>
```

Result:

```ini
# Generated by Chef, do not edit directly!
...
net.ipv6.conf.eth0.accept_ra = 1
net.ipv6.conf.eth0.accept_ra_pinfo = 0
net.ipv6.conf.eth0.autoconf = 0
...
```
In the cookbook for the DB servers:

```ruby
node.default['fb']['fb_sysctl']['kernel.shmmax'] = 19541180416
node.default['fb']['fb_sysctl']['kernel.shmall'] = 5432001
```
Case Study: sysctl

How does this help us scale?

- Significantly better heterogenous scale
- Fewer people need to manage configs
- Delegation is simple
Other Examples

Want IPv6?

node.default['fb']['fb_networking']['want_ipv6'] = true

Want to know what kind of network?

node.is_layer3?()

New cronjob?

node.default['fb']['fb_cron']['jobs']['myjob'] = {
    'time' => '*/15 * * * *',
    'command' => 'thing',
    'user' => 'myservice',
}
Other Examples: DSR

Internet

LB

Web  Web  Web
node.add_dsr_vip('10.1.1.2')
Our Chef Infrastructure
Our Chef Infrastructure

OSC and OPC
Our Chef Infrastructure - Customizations

• Stateless Chef Servers
  • No search
  • No databags
• Separate Failure Domains
• Tiered Model
Assumptions

• Server is basically stateless
  • Node data not persistent
  • No databags
  • grocery_delivery keeps roles/cookbooks in sync
• Chef only knows about the cluster it is in
Implementation Details

• Persistent data needs to come from FB systems of record (SOR)
• Ohai is tied into necessary SORs
• Runlist is forced on every run
Implementation Details: Client

- Report Handlers feed data into monitoring:
  - Last exception seen
  - Success/Failure of run
  - Number of resources
  - Time to run
  - Time since last run
  - Other system info
Implementation Details: Server

- chef-server-stats gathers data for monitoring:
  - Stats (postgres, authz [opc], etc.)
  - Errors (nginx, erchef, etc.)
  - More...

- chef-server-stats & grocery_delivery open-sourced:
  - https://github.com/facebook/chef-utils
But does it scale?
Scale

- Cluster size ~10k+ nodes
- 15 minute convergence (14 min splay)
- grocery_delivery runs every minute
- Lots of clusters
Let’s throw more than a cluster at a Chef instance!
Scale - OSS Chef

![Chart showing Chef Server vs Nodes with percentage changes]

- 80% increase
- 67% increase
- 82% increase
- 63% decrease
- 9k value
Scale - Erchef (OPC)

Pre-erchef vs Post-erchef
The graph shows the scale of Erchef (OPC) with different metrics and annotations:

- **Standby Backend**: Initially at 4k, then increases to 50% and 85%, reaching 79%.
- **Active BE::system.chefbe_num_nodes**: Starts at 4k, goes down to 2k, then increases to 7k.
- **Active BE::system.cpu-idle**: Remains constant at 8k.
- **Standby BE::system.cpu-idle**: Remains constant at 80k.

Key events:
- **Start upgrade**: Indicated by a downward arrow and a label.
- **Add nodes**: Indicated by an upward arrow and a label.
I DON'T ALWAYS TEST MY CODE

BUT WHEN I DO IT'S IN PRODUCTION
Testing: Approach

Create user and org

$ chef_test init

Sync your repo to org, test on a server

$ chef_test test -s <server>

Run Chef on test server

server# chef-client

Fix bugs, re-sync

$ vim … ; chef_test upload
Lessons

FAIL HARDER

THINK WRONG
Lessons

• Idempotent systems > idempotent records
• Delegating delta config == easier heterogeneity
• Full programming languages > restrictive DSLs
• Scale is more than just a number of clients
• Easy abstractions are critical
• Testing against real systems is useful and necessary
Summary

So how about those types of scale?
Summary

How many homogeneous systems can you maintain?  >> 17k
How many heterogeneous systems can you maintain?  > 17k
How many people are needed?  ~4
Can you safely delegate delta configuration?  Yes
Thanks

- Opscode
- Adam Jacob, Chris Brown, Steven Danna & the erchef team
- Andrew Crump
- foodcritic rules!
- Everyone I work with
- Aaron, Bethanye, David, KC, Larry, Pedro, Tyler