Rebuilding a Linux Infrastructure in under an Hour...

Brought to you by Chef.

Starting Point

A business that requires constant change with rapid turn-around

A strong desire to automate the entire Linux Infrastructure

A git Repository

A Backup System

Knowledgeable Developers and Engineers

Laying the Groundwork

1. Server Infrastructure
2. Backups
3. A software development methodology supporting rapid change
4. An automated system to quickly apply change

The Cloud

- Long Term Backups at a very reasonable price
- As much Infrastructure as the business' budget supports

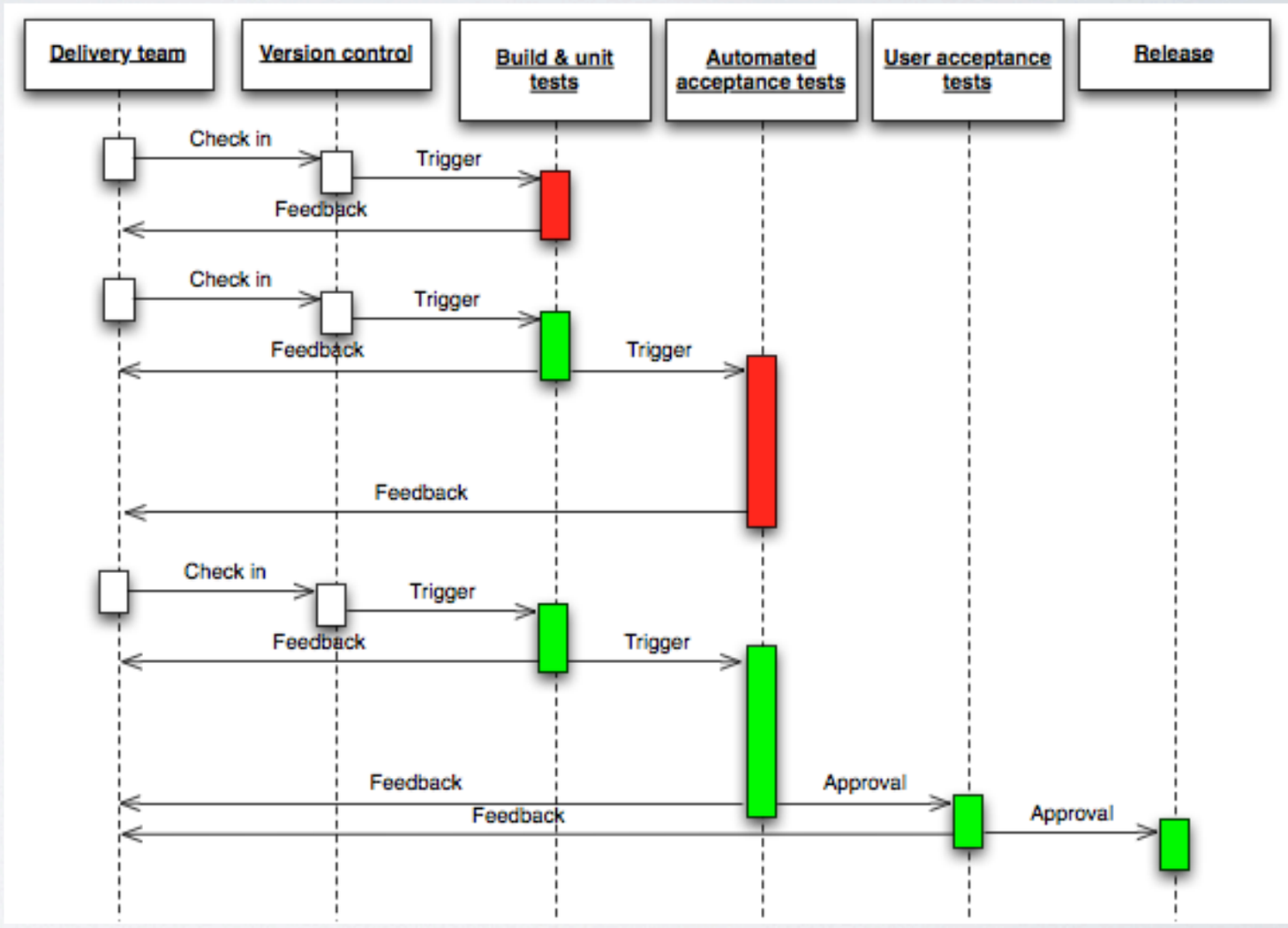
Amazon EC2, EBS snapshots and Glacier

Continuous Deployment

- Multiple small deployments decreases the risk compared to large incremental (waterfall type) deployments.
- Developer committing to mainline automatically triggers QA, and deployment to production.
- Flickr has successfully implemented a DevOps model that deploys code to Production more than 10 times a day!

Continuous Deployment

- Easily accommodates all types of change
- Complete Infrastructure wide changes lightning fast
- Q/A must be built into the process



http://en.wikipedia.org/wiki/Continuous_deployment

Automation is key to an effective C.D. implementation

- Reliability of Software Stack
- Consistency on all nodes
- Accepts change over time

An automated System to rapidly deploy changes

- Complete Automation of the entire Software Stack
- Easy Deployment of new server nodes
  - Disaster Recovery
  - Scalability -- dealing with spikes of traffic in Real-Time
- The Infrastructure exists as ‘code’
  - Documentation included, by adding in-line comments and defining server ‘Roles’

Infrastructure Management tools

Puppet

Opscode Chef

CF Engine

Salt

Custom Ruby/Python/Bash scripts

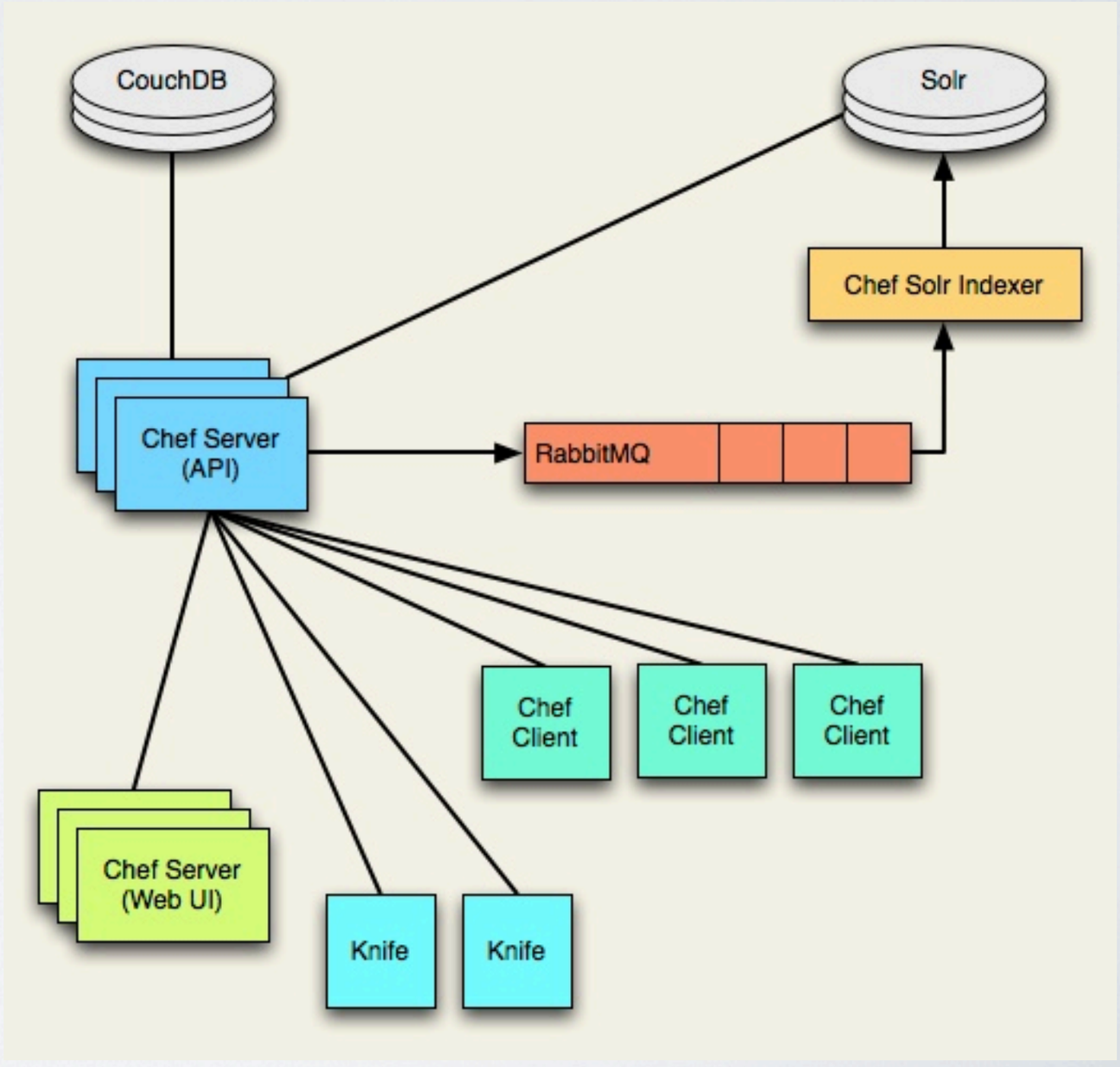
Each has it's positives and Negatives, today we will focus on
Opscode Chef

Anatomy of Chef

http://wiki.opscode.com/display/chef/Architecture

Open Source Chef Server

- An Open Source Chef Server is comprised of the following components:
  - **Chef Server:** services HTTP API requests from the Web UI, nodes, and other clients (see above).
  - **Chef Server Web UI:** the web-based management console for the Chef Server. It manages your infrastructure by making API calls to Chef Server.
  - **CouchDB:** the primary data store for a Chef server.
  - **RabbitMQ:** stores and then forwards data from Chef Server to the Chef Solr Indexer. It acts as a buffer for cases when high write loads temporarily exceed the ability of the Chef Solr Indexer to update the search index.
  - **Chef Solr Indexer:** flattens and expands data to enhance searchability, then writes the data to Chef Solr.
  - **Chef Solr:** a thin wrapper around the Apache Solr search engine. Chef Solr allows you to find your way around your infrastructure by querying its metadata.



Standing on the shoulders of Giants

Opscode Cookbook Repo:

http://community.opscode.com/

Become a member

http://community.opscode.com/cookbooks/

workstation:~ user$ knife cookbook site download <*name*>

Knife

DevOps Engineer's tool of choice

workstn1:~ user$ knife ec2 server create -I <ami> -x <user>

workstn1:~ user$ knife bootstrap servername

workstn1:~ user$ knife node edit web-2

workstn1:~ user$ knife ssh name:web-2 "sudo chef-client"

Consistent deployments

"apt-get install" and "yum install" provide the similar end results

The chef "package" resource adds a layer of abstraction

```
package "apache2" do
  package_name node['apache']['package']
  action :install
end
```

```
template "apache2.conf" do
  case node['platform']
  when "redhat", "centos", "scientific", "fedora", "arch", "amazon"
    path "#{node['apache']['dir']}/conf/httpd.conf"
  when "debian","ubuntu"
    path "#{node['apache']['dir']}/apache2.conf"
  end
  source "apache2.conf.erb"
  owner "root"
  group node['apache']['root_group']
  mode 0644
  notifies :restart, resources(:service => "apache2")
end
```

The need for boundaries

- Keeping code consistent in git and Chef
  - No manual uploads with knife to chef, only automated uploads
  - script to clone git, identify differences and upload, if necessary
  - Or should we elect a Master Chef to control the uploads?
- Storing Passwords Securely
  - Encrypted Data Bags
  - Who keeps the keys?
  - What about logging in the recipe?
- Bringing it all together
  - Communication between groups
  - git was created to solve collaboration issues encountered when multiple developers in distant locations work together *(along with **many** other issues related to code repository control)*.

Rebuilding an Infrastructure

Opscode Chef, the Data and EC2... Nothing else matters.

EC2 can provide multiple 'warm' backup sites at minimal cost... Any number of nodes can be created quickly and easily. Setting up an Infrastructure is a complicated task, and traditionally it was extremely costly (time and hardware) to replicate an Enterprise class environment. EC2 provides the hardware on demand, Glacier provides the inexpensive, long-term backup solution.

Rebuilding in an organized fashion

- Deployment of the crucial VMs
  - Chef Server
    - CouchDB
    - RabbitMQ
  - Git repository
- Bootstrap all of the 'vanilla' VMs

Rebuilding in an organized fashion

- Edit the runlist of all nodes
- Run chef-client on the nodes
  - knife ssh role:MySql-Master "sudo chef-client"
  - knife ssh role:Web-Server "sudo chef-client"
  - knife ssh name:* "sudo chef-client"
- QA

The Final Cut-over

- DNS
  - This could take hours to propogate
- Load-Balancer or TCP Proxy in Data Center
  - Outage in the Data Center (Internet Connectivity, Power, etc.)
- Planning for this moment should happen months in advance!!!

Leveraging the cloud permanently

- DNS
  - Third Party Provider using TTLs that expire quickly
- Load-Balancer in the cloud
  - VPN connection from cloud provider to the DC
  - Issues with nodes in the DC triggers a fail-over to EC2
- Most cloud providers have multiple Data Centers
  - Use all available cloud Data Centers, even Amazon only claims 99.95% uptime

References

Opscode Chef

http://www.opscode.com/chef/

http://wiki.opscode.com/display/chef/Home

https://github.com/opscode/cookbooks

Amazon AWS

http://aws.amazon.com/

Wikipedia

http://en.wikipedia.org/wiki/Continuous_integration