@ Flatiron Health

"Code to Change the World" Dev StackUp
@ Stack Exchange
Alex Lo - 2016/2/17

TLDR

Flatiron Health is a new kind of business that needs to experiment quickly and safely. We are rapidly growing our product suite.

Our team's mission is to help application teams move quickly. Here are some tools / practices that have helped (dev-ops-y)

This guy?

Engineering Manager of Developer Infrastructure

@ Flatiron Health since Jan 2015

Agile since 2005, 3+ years in AWS envs

“Infrastructure is now where agility comes from”

@alexlo03 / alo@flatiron.com

Wedgetail Airborne Radar

Faberge Big Egg Hunt NYC April 2014

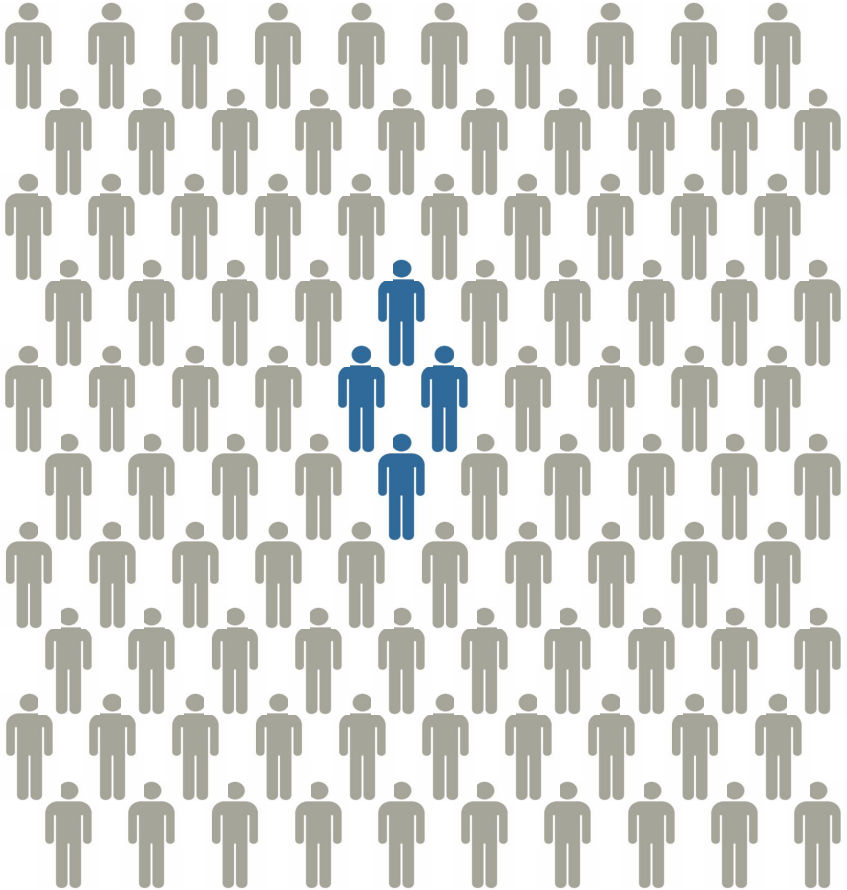
FLATIRON

Mission: To serve cancer patients and our customers by dramatically improving treatment and accelerating research.

Flatiron Health organizes the world's oncology information and makes it useful for patients, physicians, life science companies and researchers.

Today, our software connects community practices and cancer centers on a common technology infrastructure to address key healthcare challenges. Our goal is to power a national benchmarking and research network to transform how cancer care is delivered.

The Problem



4% of adult cancer patients are currently enrolled in clinical trials

Our objective is to unlock real world clinical data for the remaining 96% of cancer patients isolated within fragmented medical record systems

Poorly Resized Marketing Slides

4:3 -> 16:9: Pretty Much Intractable

Introducing Flatiron Health

Our Mission:

To serve cancer patients and our customers by dramatically improving treatment and accelerating research

Key stats:

Founded: 2012
Fundraising to date: $313MM

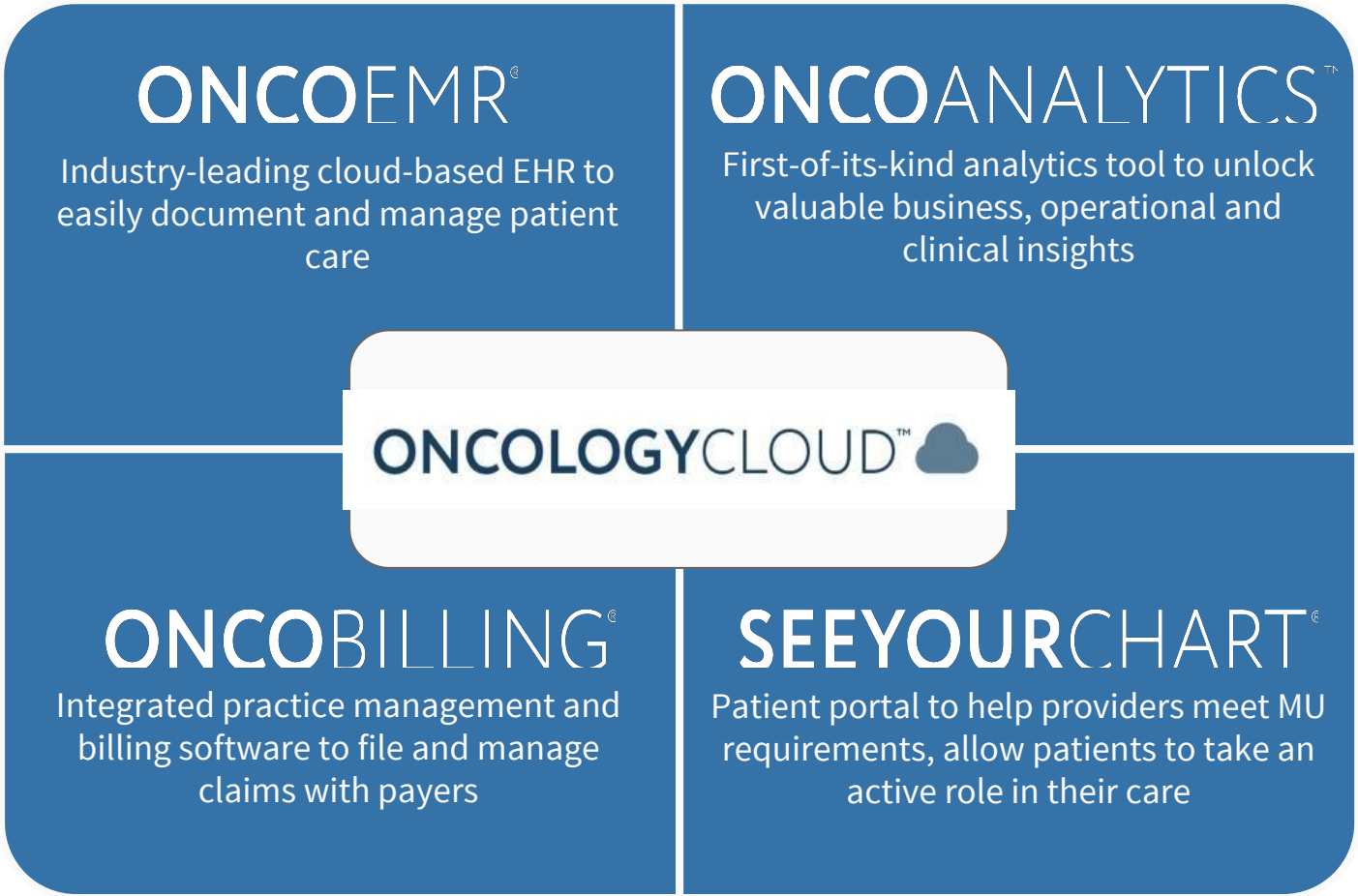
250+ employees including:

14	Medical oncologists and oncology nurses
6	Practice administrators
3	Clinical oncology pharmacists
80+	Engineers

We come from:

Duke Clinical Research Institute
Google
McKESSON
The US Oncology Network
ALTOS SOLUTIONS
Genentech
ONYX PHARMACEUTICALS
Microsoft
Memorial Sloan Kettering Cancer Center
HUMEDICA An Optum Company
imshealth INTELLIGENCE APPLIED
FLORIDA CANCER SPECIALISTS & Research Institute
GE Healthcare
Cleveland Clinic

Our Core Software Platform



Deep Engineering, Clinical and Oncology Business Expertise

ONCOLOGYCLOUD™

ONCOEMR®	ONCOANALYTICS™	ONCOBILLING®	SEEYOURCHART®	OncoTrials

Value-based Care Initiative

OCM Reporting	Integrated Treatment Pathways & Content	Value-based Analytics	Patient Engagement Services	Clinical Transformation Assistance	Alternative Payment Model Design

Clinical Research Opportunities

Strategic Partnerships

VARIAN medical systems

NCCN National Comprehensive Cancer Network®

FOUNDATION MEDICINE®

via oncology pathways™

GUARDANT HEALTH

VECTOR ONCOLOGY

230

Cancer Clinics

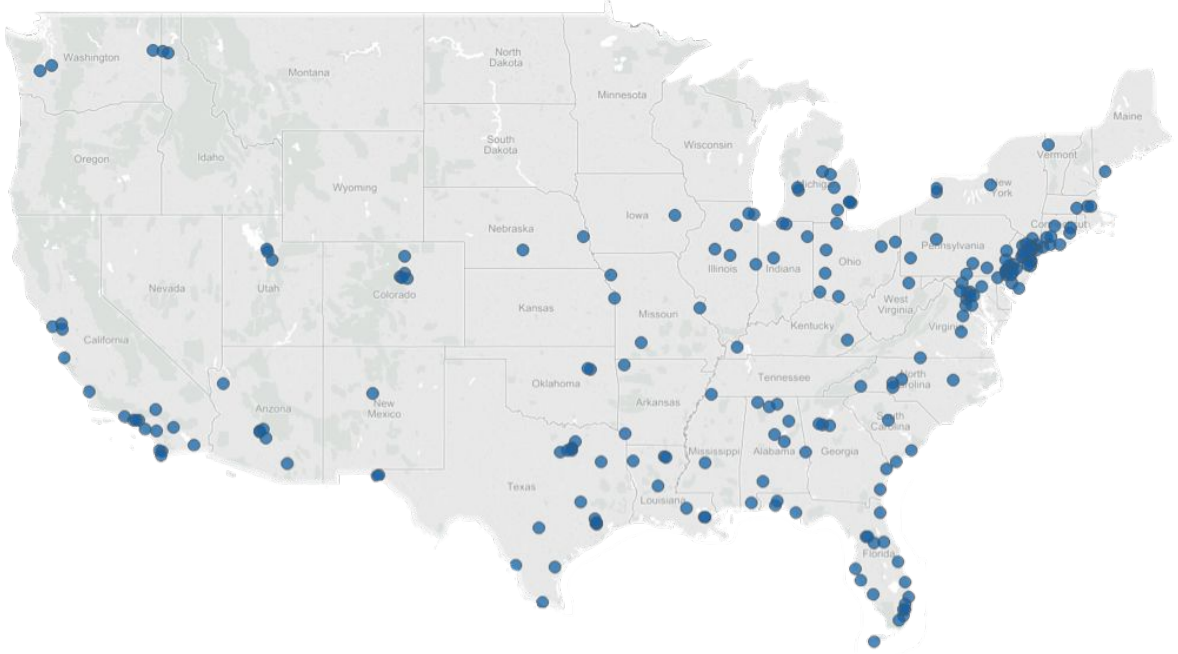
2,000

Clinicians

1,000,000

Active Cancer Patients

Flatiron Provider Network



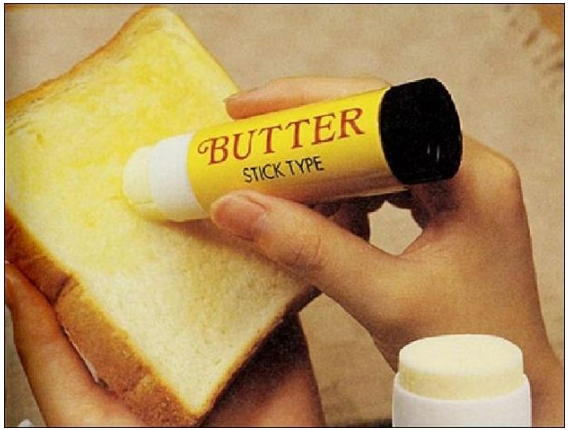
- Represents largest real-world oncology data source

A new kind of business

We're not a consumer app - we serve multiple audiences, with multiple requests

Our technology, processes and infrastructure have to support experimentation

We're rapidly growing across the country, and eventually internationally



Google Images: Invention

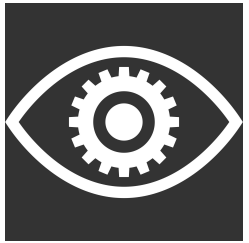
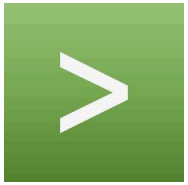
Developer Infrastructure

~~Move fast and break things~~

Move fast with stable infra

- Configuration Management
- Netsec (Connectivity + Segregation)
- Cloud Provider Controls
- Encrypted Volumes
- Secure Shared Storage
- Compute Resource Allocation and Inventory
- Application Stats + Monitoring
- On Call Management and Notifications
- AuthN/AuthZ
- Best Practices (SSL / ELB configs)
- … and friends





Challenges

- Charter: speed up development
- Systems had “grown”
- HIPAA / strong security concerns
- Were hosted in a cloud provider everyone hated

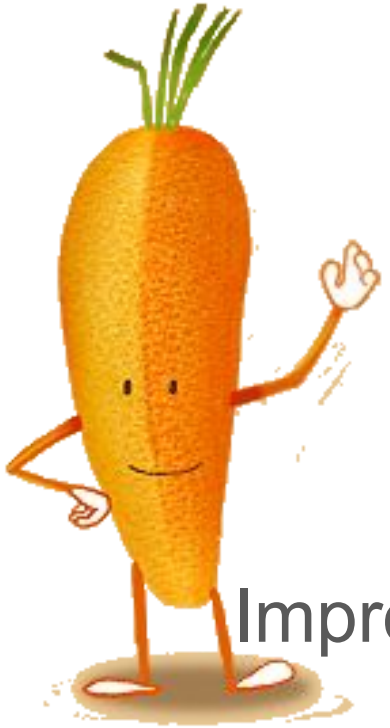
Our plan to speed up development

- Migrate to a platform that allows more automation (Cloud X to AWS)
- Make configuration drift a thing of the past
- Make infrastructure workflows easier in AWS (carrot)
  - Example: create self-configuring hosts
- Allow developer self service via infrastructure as code
  - Example: network configuration
- Culture of visibility
  - Chat integrated alerts
  - Jenkins as watchdog
  - Chatops

Drift Makes Life Hard

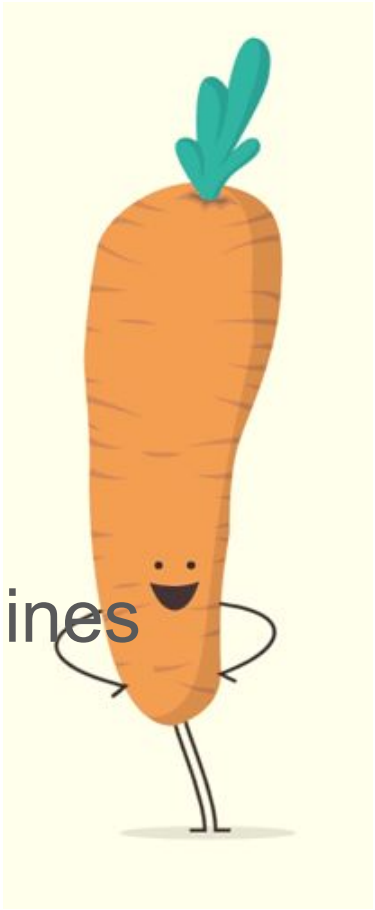
“Can I run Chef client?”
“Uh, maybe?” ❌

- When things are growing, a hybrid of configuration management and manual approaches
- Ideally running CM should always be safe
- Reconciliation is time consuming when there is drift
- Implications:
  - Run CM often
  - Make the creation of drift visible when it cannot automatically be corrected
- Applies to both host and cloud configuration “Why is this port open??”

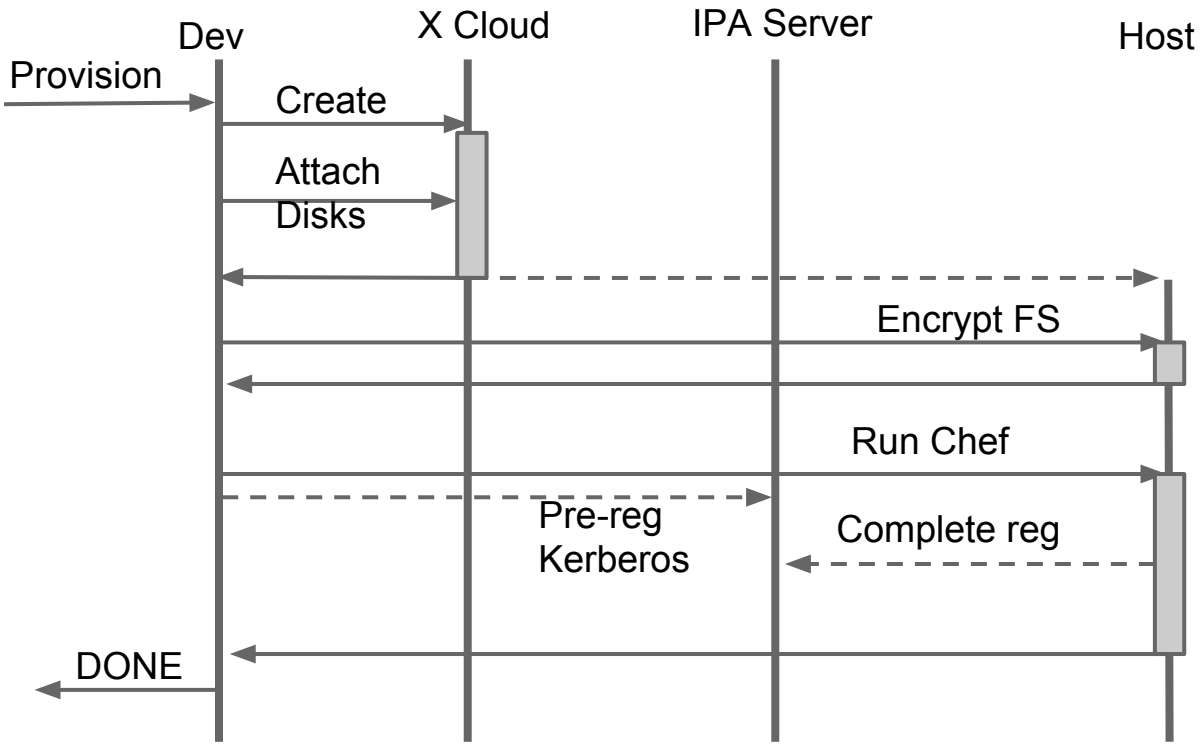


Carrot

Improving Creating New Virtual Machines

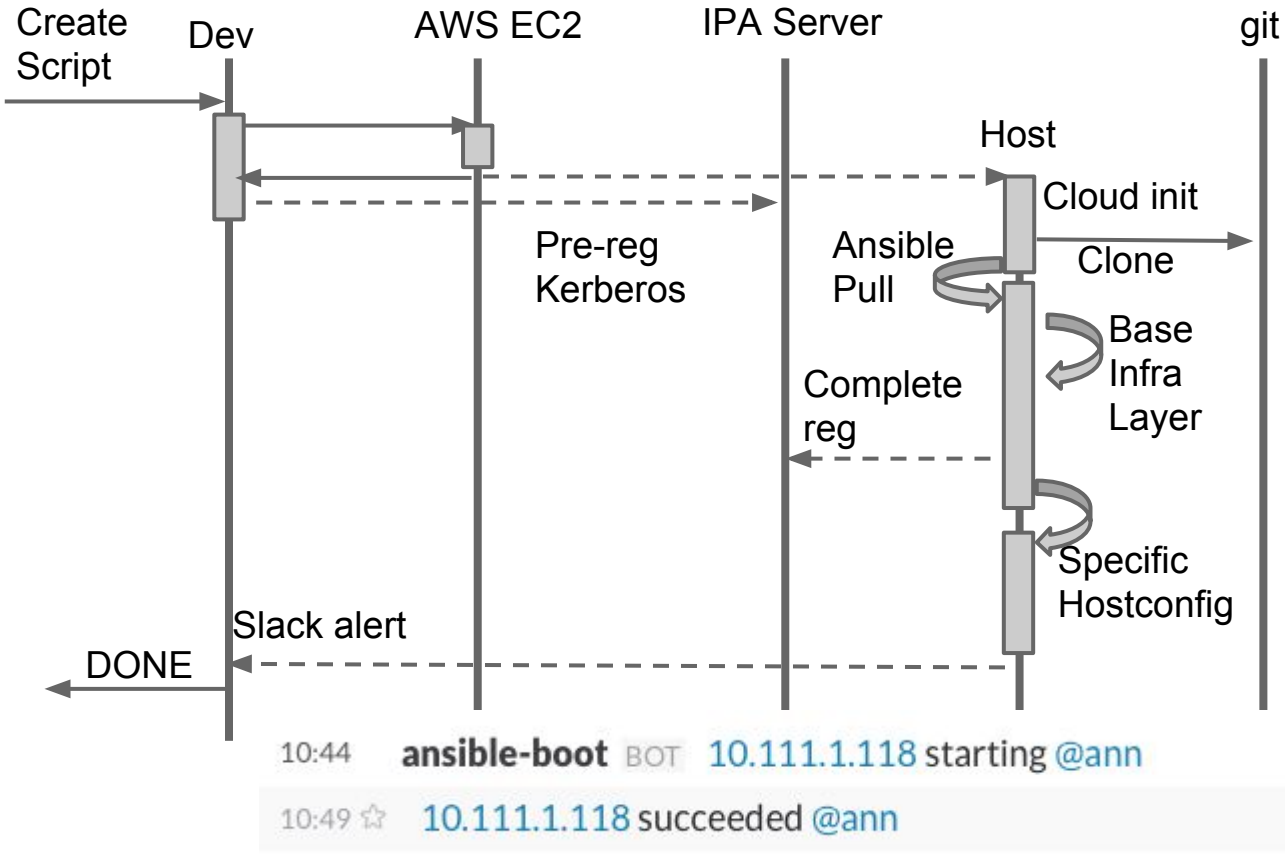


Previous Workflow



Not pictured: multiple user prompts, chances for user error, some other wrinkles and edge cases

Enter Ansible



After the script launch, no human touch points until machine announces it is ready

Chatops

In our old cloud host, bringing up new servers was time consuming, therefore engineers avoided doing it

14:03 **Alex Lo** [redacted]

14:04 [redacted]

14:04 doc ec2 add host name=jenkins-worker-e-01 role=jenkins-worker

14:04 **Dr. Hippo** BOT alo: This action requires secondary authentication.

14:04 alo: Sending approval request to Android (XXX-XXX-8814)...

14:04 alo: Authenticated successfully!

14:04 Launching your instance...

14:04 ...and blastoff! Your host "jenkins-worker-e-01" is being provisioned.
IP Address: `10.110.0.132`
Instance ID: `i-`[redacted]

"doc"

14:06 **ansible-boot** BOT 10.110.0.159 starting @alo

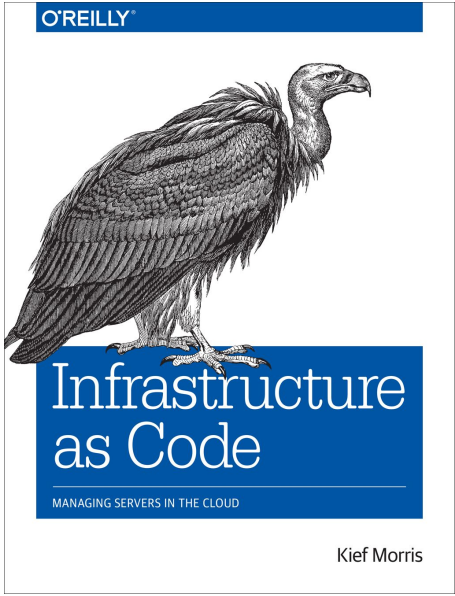
14:07 10.110.0.132 starting @alo

14:34 **ansible-boot** BOT 10.110.0.132 succeeded @alo

14:35 10.110.0.159 succeeded @alo

Self Serve Infrastructure

- “Infrastructure as Code” is not a new idea
  - Make “what is” transparent
  - Make changes auditable, historical
  - Allow change proposals via Pull Requests
  - Assert that infrastructure is currently complying with what we think it should be
    - Avoid configuration management “drift”
- We’ve found Ansible + Jenkins to work well for us



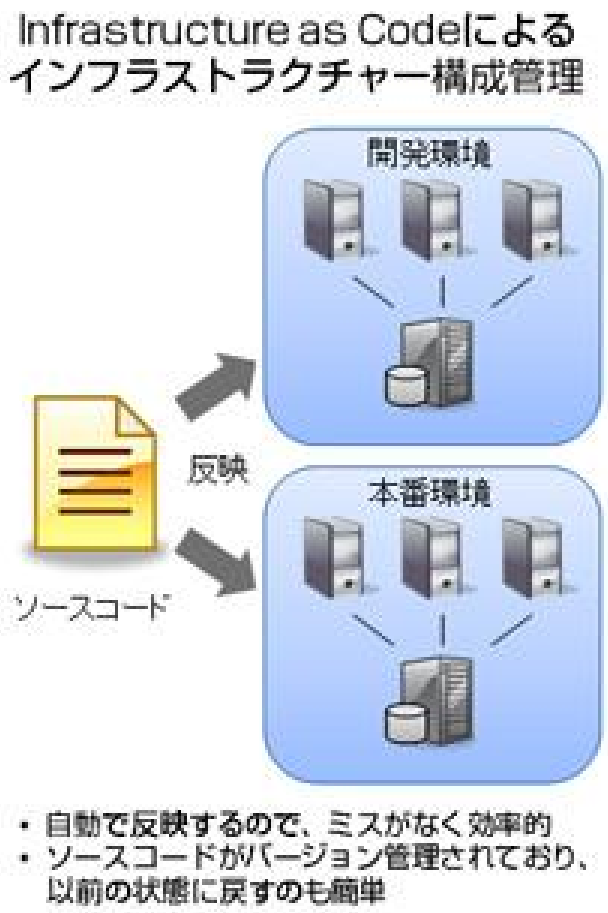
Infrastructure as Code

Cloud Configuration

- Network Security Groups
- ELBs
- S3 Buckets
- IAM Policies

Machine Configuration

- VM definitions (EC2 size, etc)
- Software configuration (nginx, etc)



Security Group Example

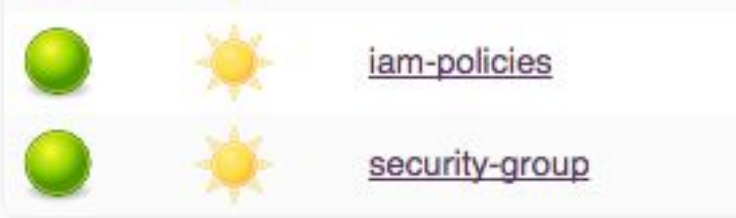
Infrastructure as Code
Security Groups

```
- name: jenkins security group
  ec2_group:
   name: jenkins
   description: jenkins
   vpc_id: "{{ vpc_id }}"
   region: us-east-1
   rules:
     - proto: tcp
       from_port: 8080
       to_port: 8080
       group_id: "{{ sg_jenkins_elb.group_id }}"
    rules_egress:
     - proto: all
       cidr_ip: 0.0.0.0/0
```

“Don’t trust and verify” > “Trust but verify”

Jenkins as the enforcer

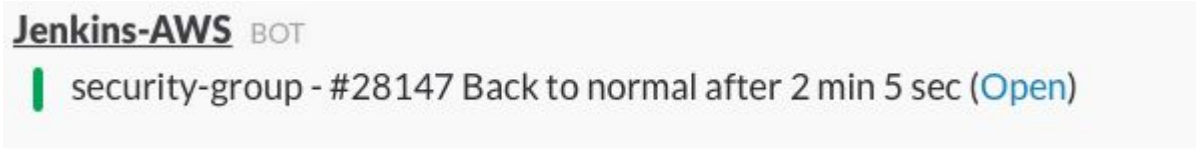
Allow read access to infrastructure and code, can tell us when things are awry



Security Group Workflow

1. Engineer proposes Security Group changes in code diff
2. Security and/or our team approves after review
3. Engineer merges to master
4. Drift detected (code is ahead of cloud conf)

   Mechanism: `ansible-playbook --check` - anything updated?

5. Admin acts to run playbook
6. 

Base Software Configuration and Drift

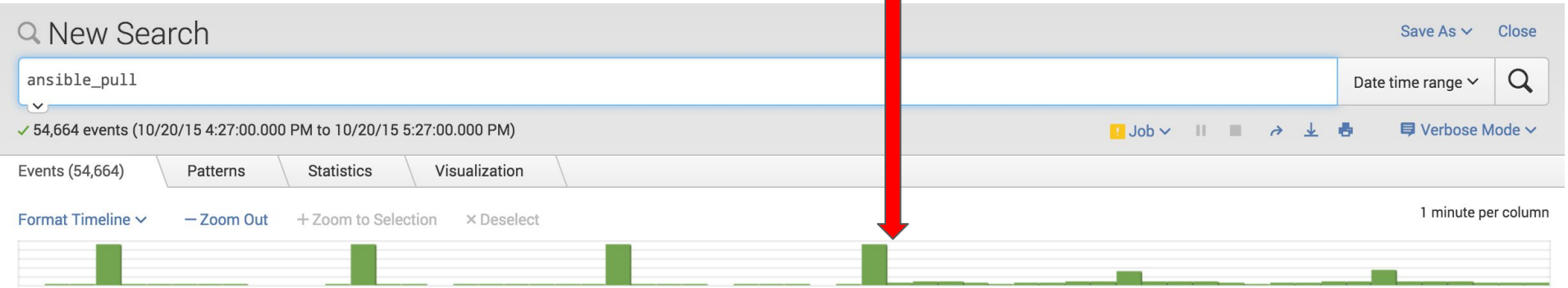
“One touch” hosts all come equipped with continuously running configuration management of “base” level concerns

Decouples our team from specifics of how teams would like to administer / evolve their hosts

Example: when we added splunk universal forwarder to all machines, we didn’t require action from anyone

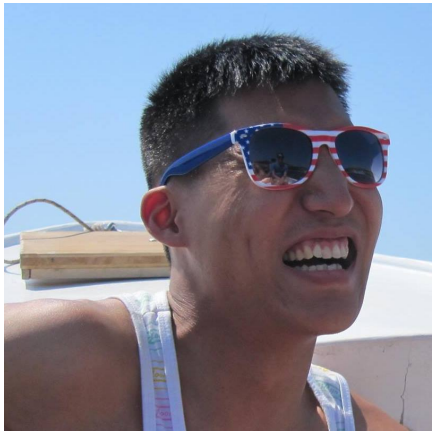
Thundering Herd

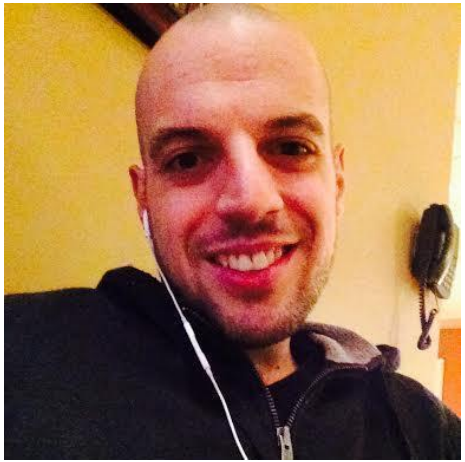
Thundering git herd (~ 150 hosts)




```
# run ansible-pull every ten minutes on the last ipv4 octet % 10
- name: ansible-pull cron
  cron:
   name: ansible-pull
   minute: "{{ ansible_eth0['ipv4']['address'].split('.')[3] | int % 10 }}-59/10"
   job: ...
```

Thanks to





Thanks + Ops Drinks

Thank you Stack Exchange and Dev StackUp

I’m organizing drinks with ops teams to cross pollinate ideas and experience

We are hiring

alo@flatiron.com