Scalable System Operations

♥ Joshua Hoffman

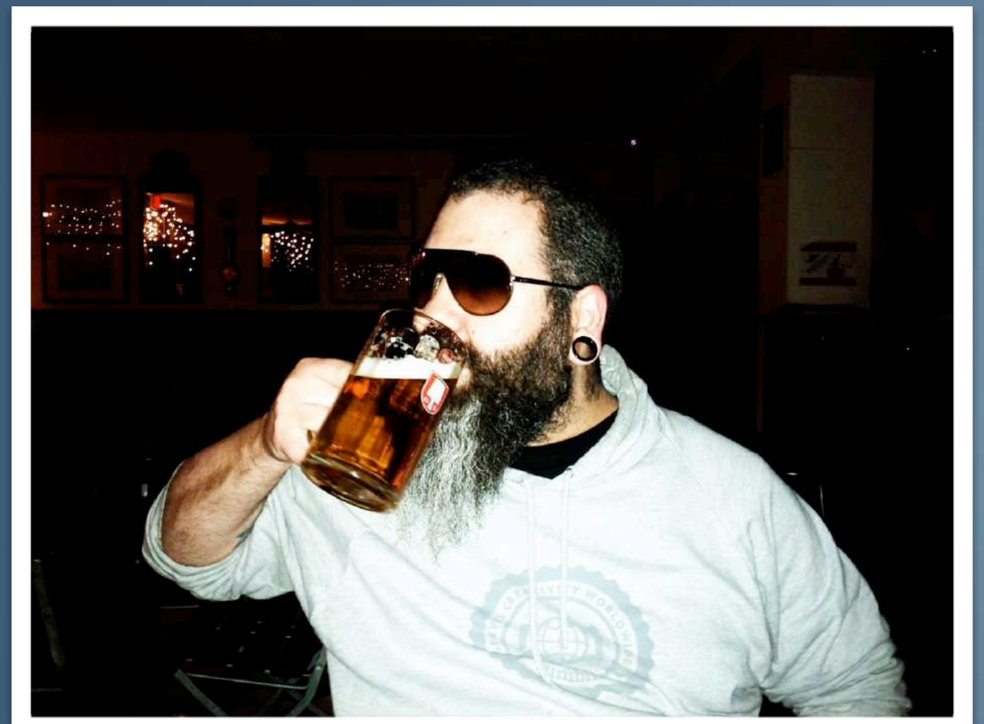
tumblr.

About This Talk

- Set of principles
- Operations Engineering
- Tumblr project
- Server management
- Massively automated
- Software
- Techniques
- Example code
- Best practices
- Open source

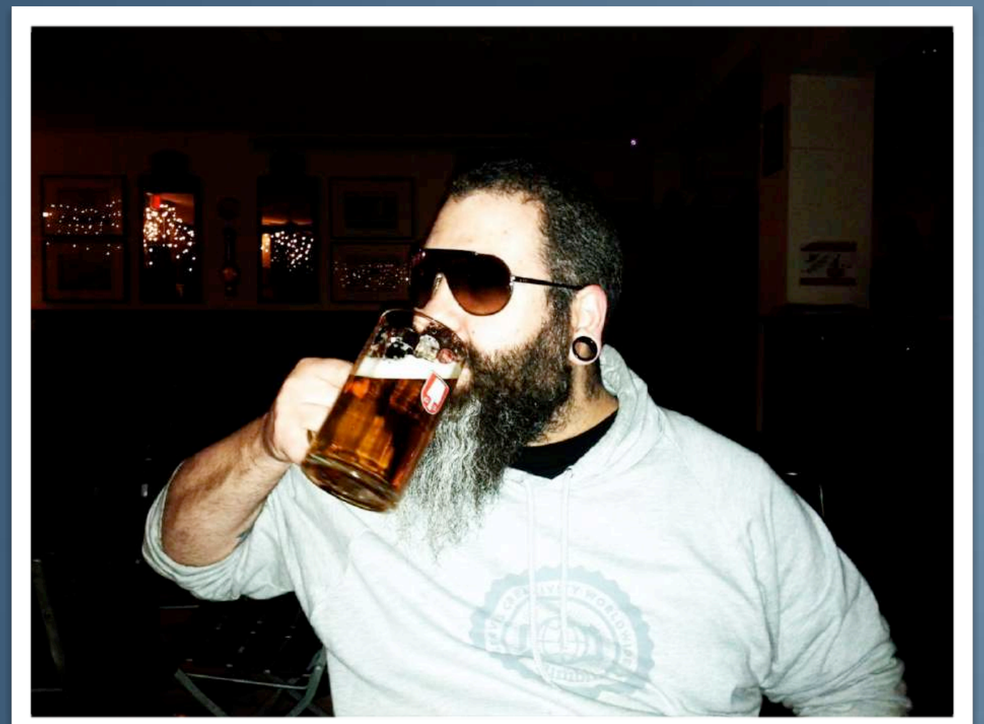
About Me

About Me



- 1995: CompUSA
  Intro to The Internet
- 2000: Guru Labs
  Sun, Cisco, Red Hat
- 2002: Red Hat
  Sys admin courseware

About Me

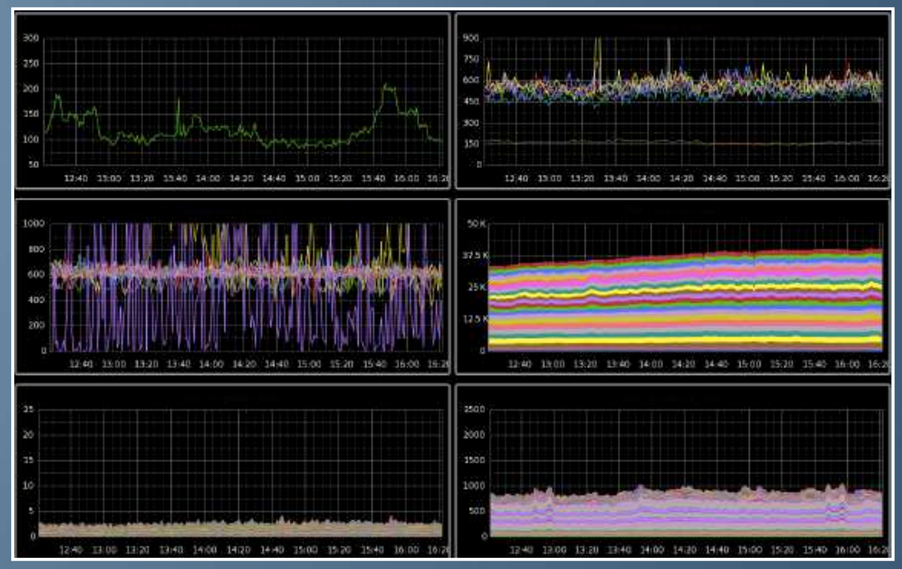


- 2004: Fortress Systems
  Anti-spam/malware
- 2005: Red Hat
  Virtualization cert
  Remote learning
  Defined “cloud”
- 2011: Tumblr
  Lead Systems Eng

The Problem

Deploying new servers is very repetitive and slow.
(and we hate that)

The Job We Don't Want



The Solution

Automation

Automation

- [x] Install OS
- [x] Configure OS
- [x] Install software
- [x] Configure software
- [ ] Add to DNS
- [ ] Add to monitoring
- [ ] Add to trending

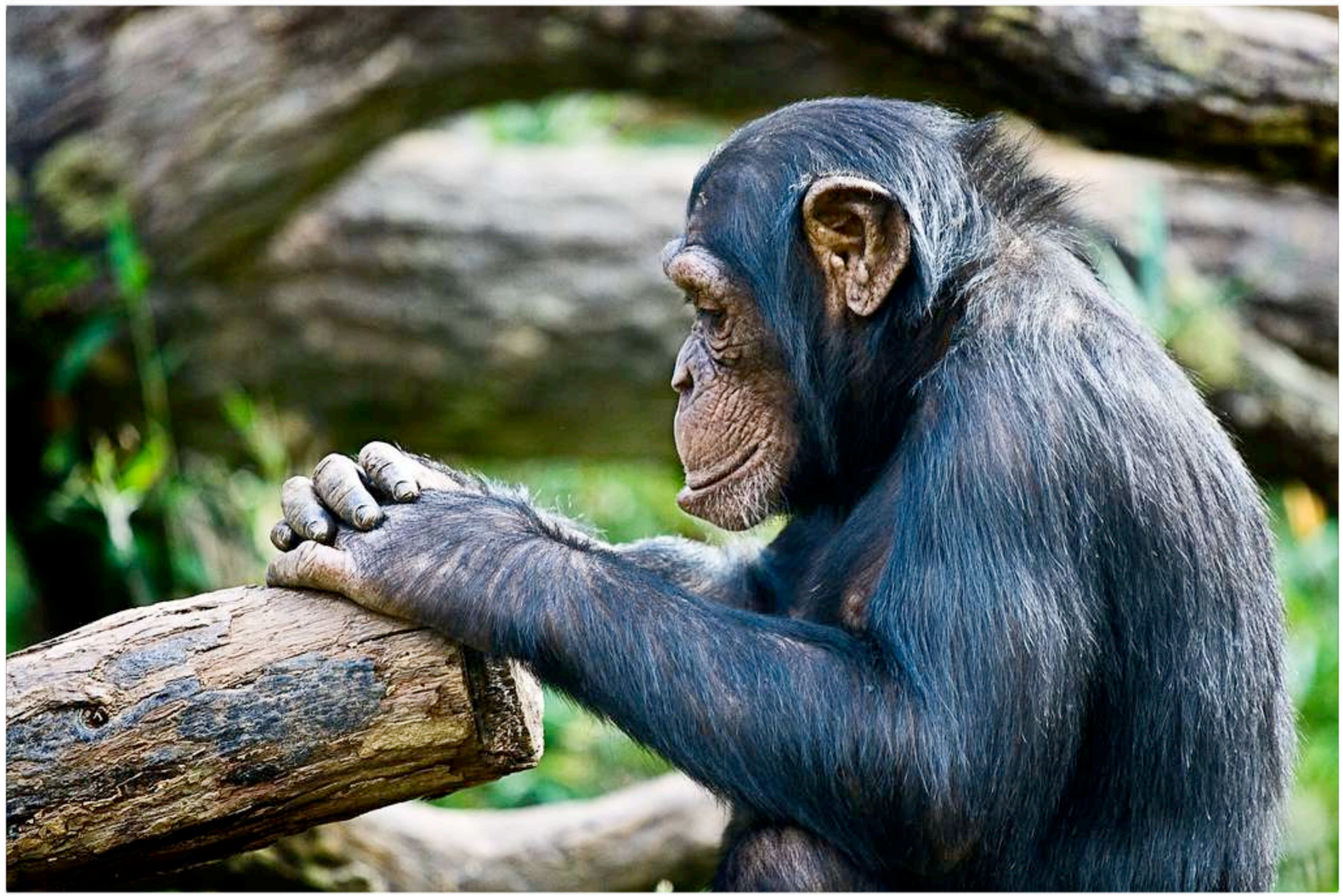
- [ ] Firmware
- [ ] Configure BIOS
- [ ] Set up BMC
- [ ] Inventory
- [ ] Stress testing
- [ ] Network config

The Goal Is Clear

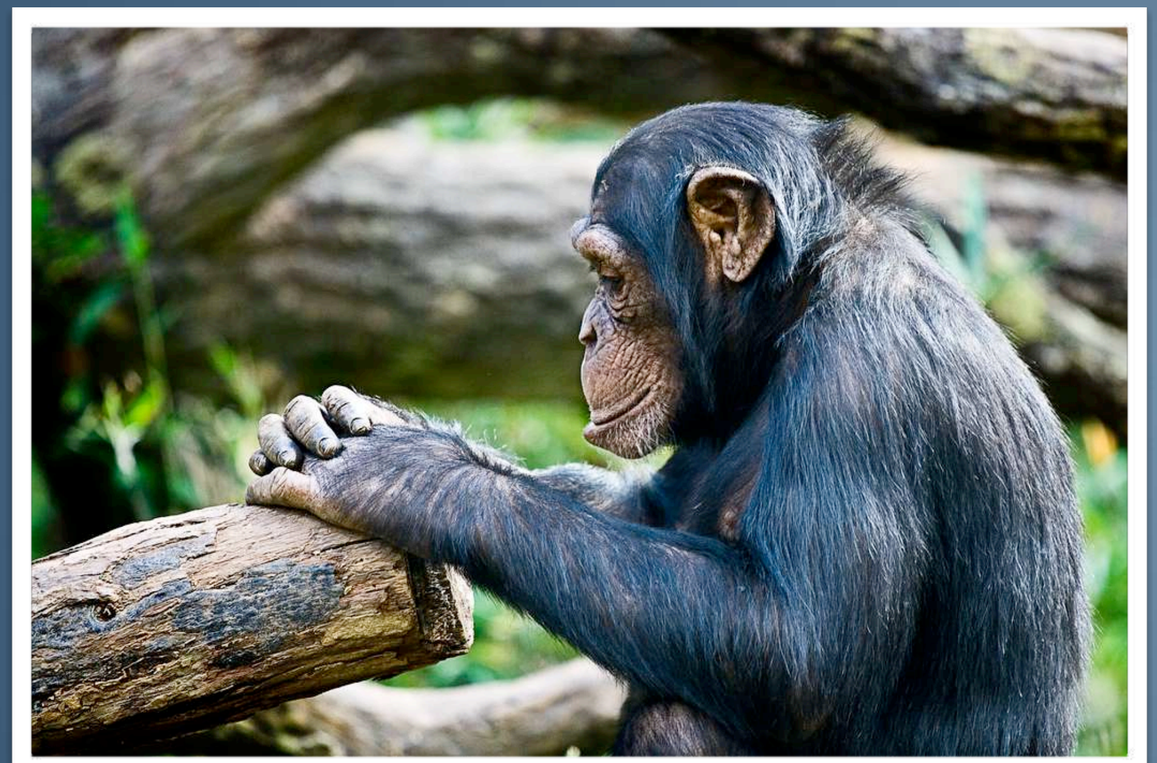
Automation

tumblr.

Time To Strategize



Time To Strategize



Use open source?
Which?
Buy software?
Which?
Write software?
Mix and match?

The Choice Principle

The time to make a decision is a function of the possible choices.

Rapid Software Research

Rapid Software Research

1. Define
2. Gather
3. Disqualify
4. Rank

Rank

Rank

- Modularity
- Compliance
- Novelty
- Disruption

My Requirements

- Asset inventory
- State management
- Robust API
- Event triggers

tumblr.

My Requirements

- Modular
- Flexible
- Extensible
- Fast

My Requirements

Manage physical hardware
as easily as virtual machines.

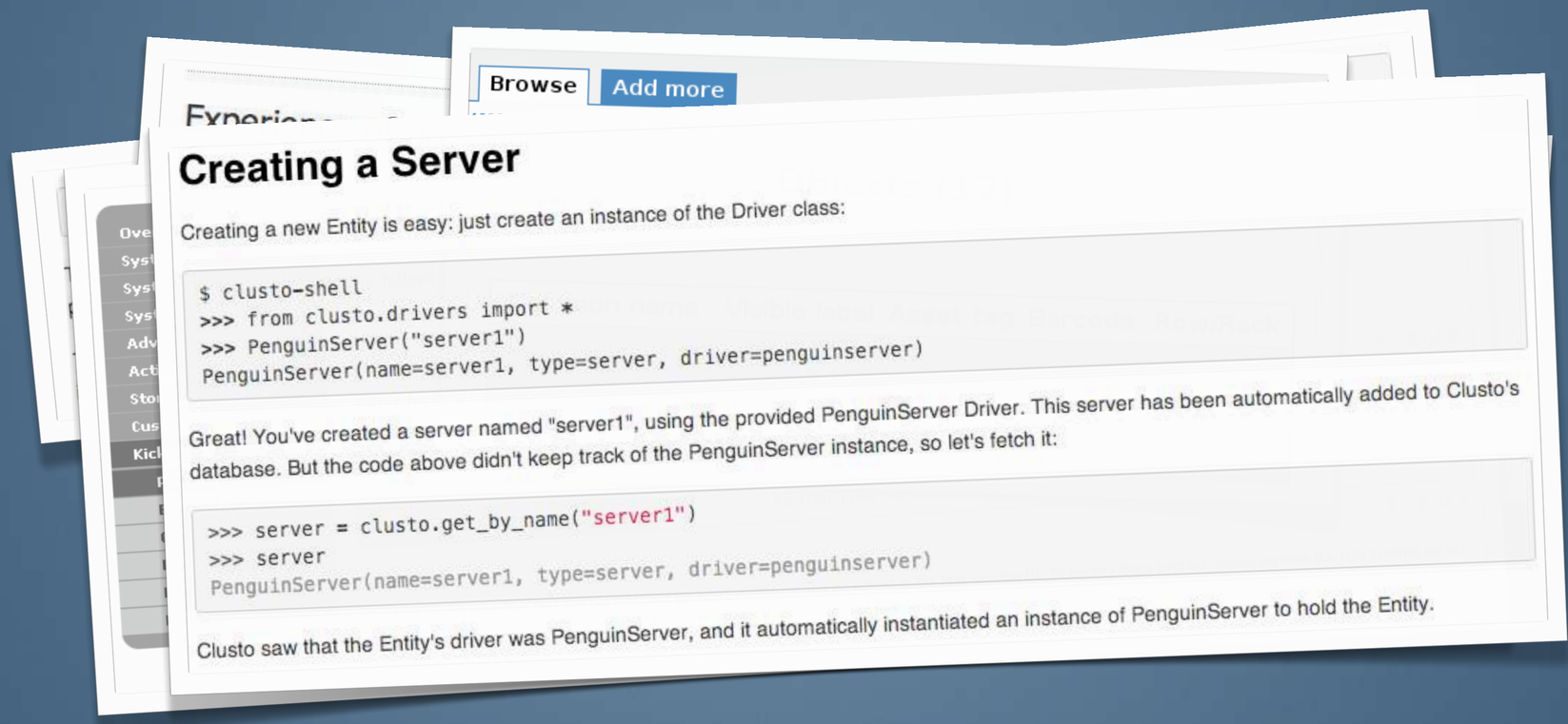
tumblr.

The Usual Suspects

- Cobbler
- Foreman
- Satellite
- Orchestra
- Racktables
- Clusto

tumblr.

But Wait!


Browse
Add more
Creating a Server
Creating a new Entity is easy: just create an instance of the Driver class:
$ clusto-shell
>>> from clusto.drivers import *
>>> PenguinServer("server1")
PenguinServer(name=server1, type=server, driver=penguinserver)
Great! You've created a server named "server1", using the provided PenguinServer Driver. This server has been automatically added to Clusto's database. But the code above didn't keep track of the PenguinServer instance, so let's fetch it:
>>> server = clusto.get_by_name("server1")
>>> server
PenguinServer(name=server1, type=server, driver=penguinserver)
Clusto saw that the Entity's driver was PenguinServer, and it automatically instantiated an instance of PenguinServer to hold the Entity.

Data Entry

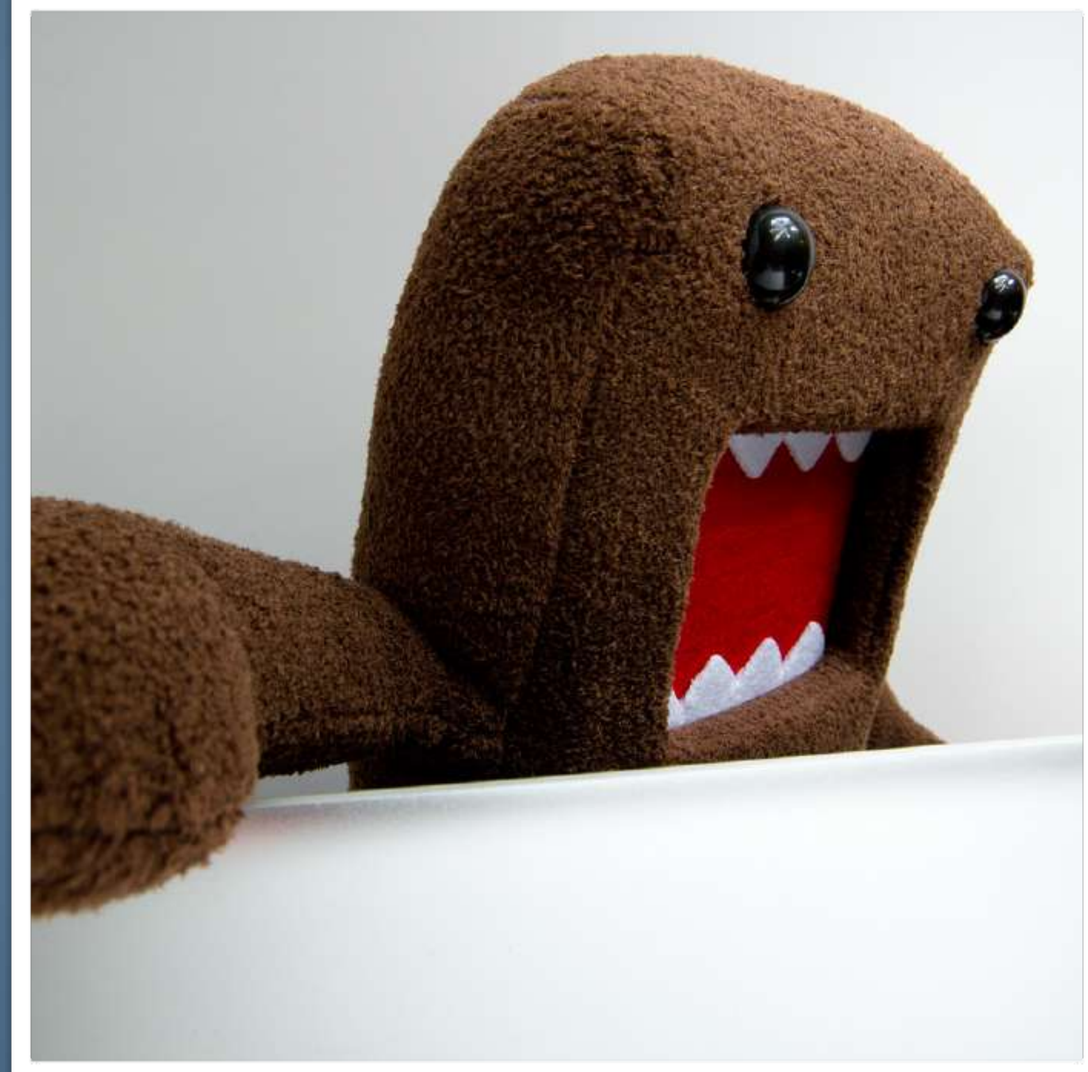
"Just import the data supplied by the hardware vendor..."

Missing Requirements

- ☐ Firmware
- ☐ Configure BIOS
- ☐ Set up BMC
- ☐ Inventory
- ☐ Stress testing
- ☐ Network config
- ☐ Add to monitoring
- ☐ Add to trending

tumblr.

We have to write software!



We have to write software!

- Delivery Schedule
- Scope Creep
- Maintenance
- Documentation

Tumblr Management Stack

The Glue Principle

Unix Rule of Parsimony:
Write a big program only when it is clear by demonstration that nothing else will do.

The Standards Principle

The nice thing about standards is that you have so many to choose from.

-Andrew Tanenbaum

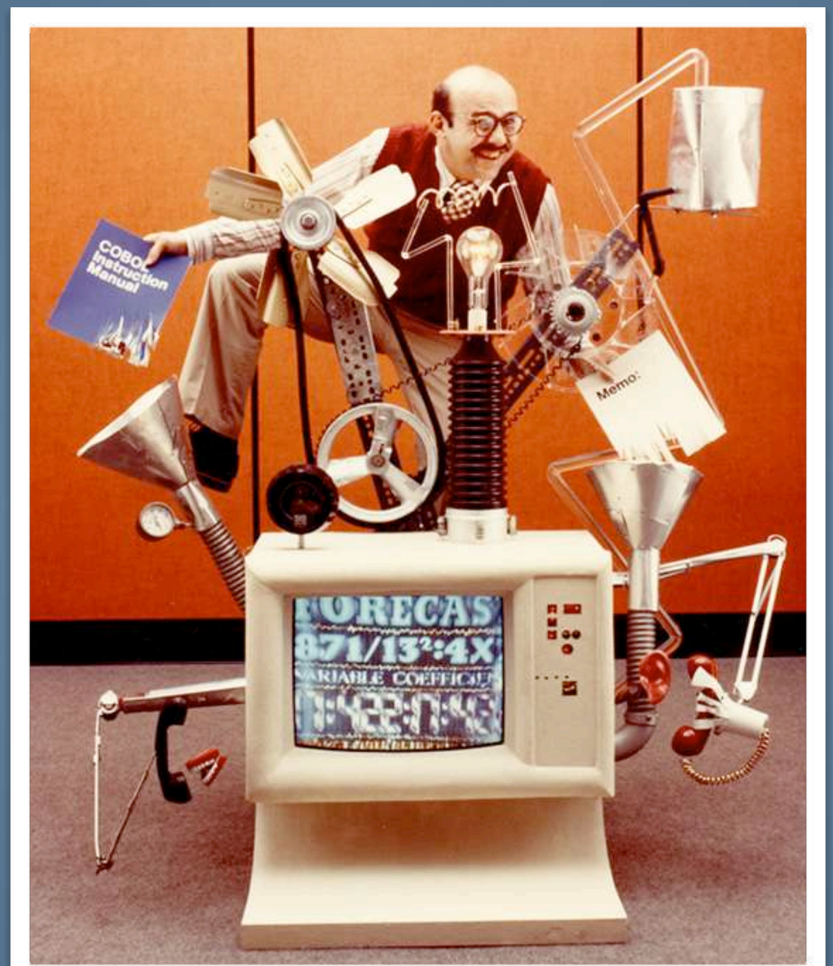
The Simplicity Principle

Unix Rule of Simplicity:
Design for simplicity; add complexity only where you must.

The 3:00 AM Principle

It must be obvious to someone woken up from a sound sleep at 3:00 am.

The Don’t Break The OS Principle



The software should NOT prevent the OS from working as expected.

The Amnesia Principle

Given enough time, you WILL forget why you did that.

Tumblr Management Stack

- iPXE
- Invisible Touch
- Collins
- Phil
- Kickstart
- Puppet

Why not pxelinux?

Why not pxelinux?

- TFTP
- Flat files

iPXE

iPXE

- HTTP, FTP, iSCSI
- Scriptable
- Variables
- Dynamic

ISC DHCP For iPXE

```
# subnet for the provisioning vlan
subnet <%= subnet %> netmask <%= netmask %> {
    option domain-name           "<%= option_domain_name %>";
    option routers               <%= option_routers %>;
    option domain-name-servers <%= option_dns_servers.map{|i| "#{i}"}.join(", ") -%>;
    option subnet-mask           <%= option_subnet_mask %>;
    default-lease-time           21600;
    max-lease-time               43200;
    range                        <%= range_start %> <%= range_end %>;
    # If a pxe request comes in from ipxe send the config url
    if exists user-class and option user-class = "iPXE" {
        filename "<%= ipxe_config_url %>"; # http://foo.example.com/ipxe/${net0/mac}
    # For all other pxe requests send ipxe
    } else {
        next-server <%= next_server %>; # tftp server
        filename "<%= filename %>";     # path to ipxe binary on tftp server
    }
 }
}
```

tumblr.

Fedora LiveCD Tools

```
lang en_US.UTF-8
keyboard us
timezone US/Eastern
auth --useshadow --enablemd5
selinux --enforcing
firewall --disabled
repo --name=centos   --baseurl=http://127.0.0.1/pub/repo/centos/os/6.2
repo --name=infra    --baseurl=http://127.0.0.1/pub/repo/infra/6.2
repo --name=epel     --baseurl=http://127.0.0.1/repo/epel/6/x86_64/

%packages --excludedocs
@core
dracut
dracut-kernel
device-mapper
device-mapper-event
%end
```

tumblr.

Invisible Touch Kickstart

```
# Invisible Touch Live OS image
%include centos-6.2-livecd-minimal.ks
%packages --excludedocs
it
%end
%post
cat > /etc/issue <<EoF
Invisible Touch Live OS v0.0.4
Kernel \r
EoF
# set ipmi to start at boot up
/sbin/chkconfig ipmi on
# configure rsyslog
cat >> /etc/rsyslog.conf <<EoF
# invisible touch
local0.*                                                /var/log/it.log
local0.*                                                /dev/tty7
EoF
%end
```

Invisible Touch Utilities

- lshw
- lldpd
- Breakin
- ipmitool
- Bash scripts

tumblr.

lshw

```
<node id="disk:1" claimed="true" class="disk" handle="SCSI:04:00:01:00">
  <description>ATA Disk</description>
  <product>ST91000640NS</product>
  <vendor>Seagate</vendor>
  <physid>0.1.0</physid>
  <businfo>scsi@4:0.1.0</businfo>
  <logicalname>/dev/sdf</logicalname>
  <dev>8:80</dev>
  <version>n/a</version>
  <serial>9XG0ETB8</serial>
  <size units="bytes">1000204886016</size>
  <configuration>
    <setting id="ansiversion" value="5" />
    <setting id="signature" value="000e1763" />
  </configuration>
  <capabilities>
    <capability id="partitioned">Partitioned disk</capability>
    <capability id="partitioned:dos">MS-DOS partition table</capability>
  </capabilities>
</node>
```

lshw generates hardware info XML

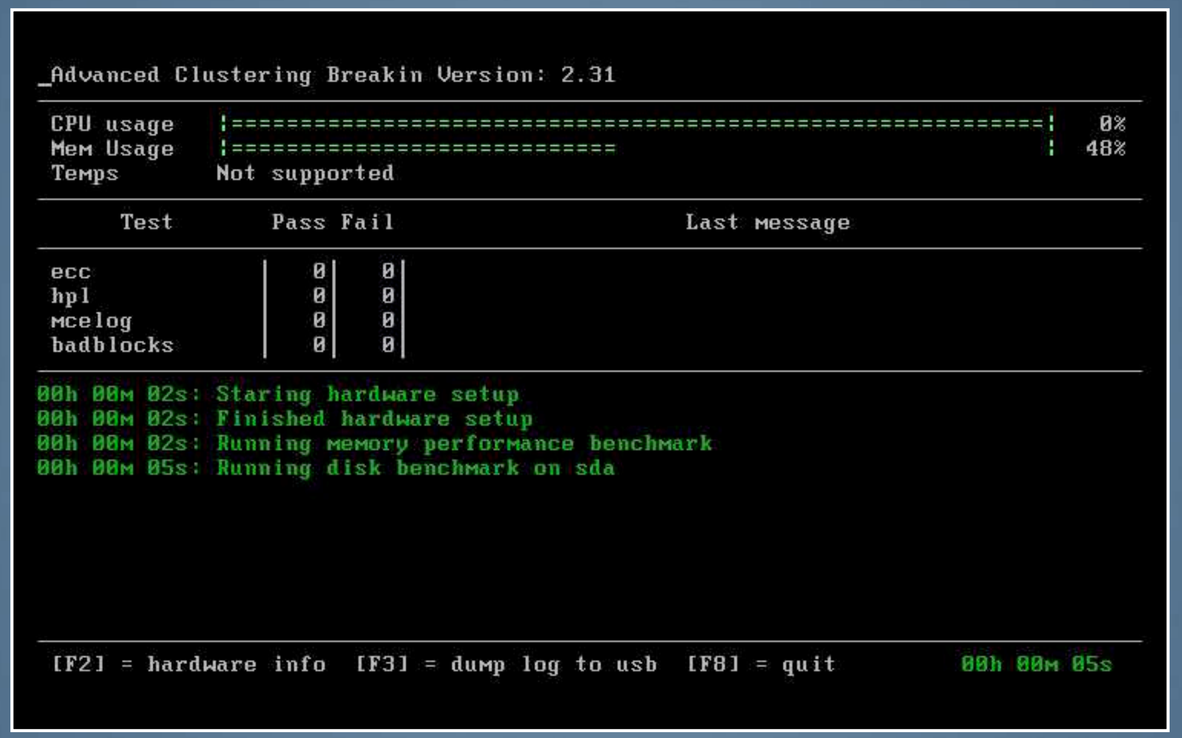
tumblr.

lldpd

```
<interface label="Interface" name="eth0" via="LLDP" rid="1" age="0 day, 00:01:03">
 <chassis label="Chassis">
  <id label="ChassisID" type="mac">78:19:f7:88:60:c0</id>
  <name label="SysName">core01.dfw01</name>
  <descr label="SysDescr">Juniper Networks, Inc. ex4500-40f</descr>
  <capability label="Capability" type="Bridge" enabled="on" />
  <capability label="Capability" type="Router" enabled="on" />
 </chassis>
 <port label="Port">
  <id label="PortID" type="local">608</id>
  <descr label="PortDescr">ge-0/0/3.0</descr>
  <mfs label="MFS">1514</mfs>
  <auto-negotiation label="PMD autoneg" supported="no" enabled="yes">
   <advertised label="Adv" type="10Base-T" hd="no" fd="yes" />
   <current label="MAU oper type">unknown</current>
  </auto-negotiation>
 </port>
 <vlan label="VLAN" vlan-id="666" pvid="yes">DFW01-PROVISIONING</vlan>
 <lldp-med label="LLDP-MED">
  <device-type label="Device Type">Network Connectivity Device</device-type>
  <capability label="Capability" type="Capabilities" />
 </lldp-med>
</interface>
```

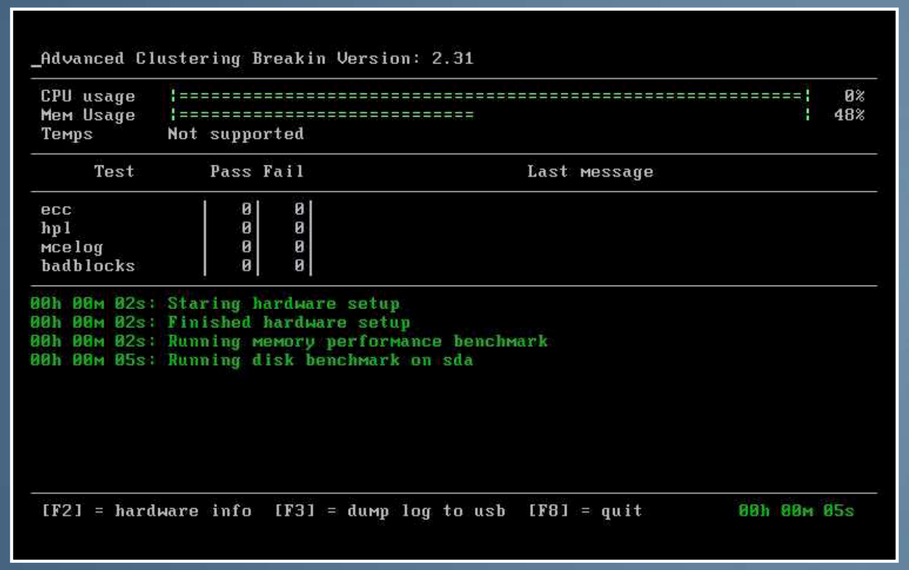
lldpctl outputs network info in XML

Breakin



Stress testing framework

Breakin



- Standard tools
- LINPACK
- Extensible
- Bash scripts

Invisible Touch

- ✓ Firmware
- ✓ Configure BIOS
- ✓ Set up BMC
- ✓ Inventory
- ✓ Stress testing
- ✓ Network config

Collins

- Asset management system in Scala
- REST API
- Client libraries in Ruby, Python and Bash
- Shell tool for scripting and automation
- Callback system for hooking into events
- Granular permissions model
- Flexible web and API based provisioning
- Remote power management
- IP Address allocation and management
- Distributed mode for spanning data centers

Collins Docs

Collins

Asset management for engineers

About

Collins started as a system to manage all of the physical servers, switches, racks, etc in Tumblr production environments. As we started to inventory hardware, IP addresses, software, and so on, we found the API and data gave us an excellent way to drive automation processes. Today Collins can do push button cluster (HBase, Hadoop, web, etc) deployment, drive configuration generation when hardware cluster topologies change, drive infrastructure updates when software configuration changes, and help manage software deploys.

Because of the loosely coupled design of Collins, consistently applied conventions are a system requirement. This document serves as a guide to those conventions as well as the basic core concepts of the collins system. If you're just interested in the basic howto or screenshots, click here.

Approach

Collins is extremely dumb. It knows about assets, their meta-data and asset logs. You can think of collins as a key/value store where each asset has its own set of key/value pairs. There are no relationships between assets other than the ones you, through convention, derive. The API makes it trivial to create and manage the tags (meta-data, key/value pairs) associated with an asset, and to query based on those tags.

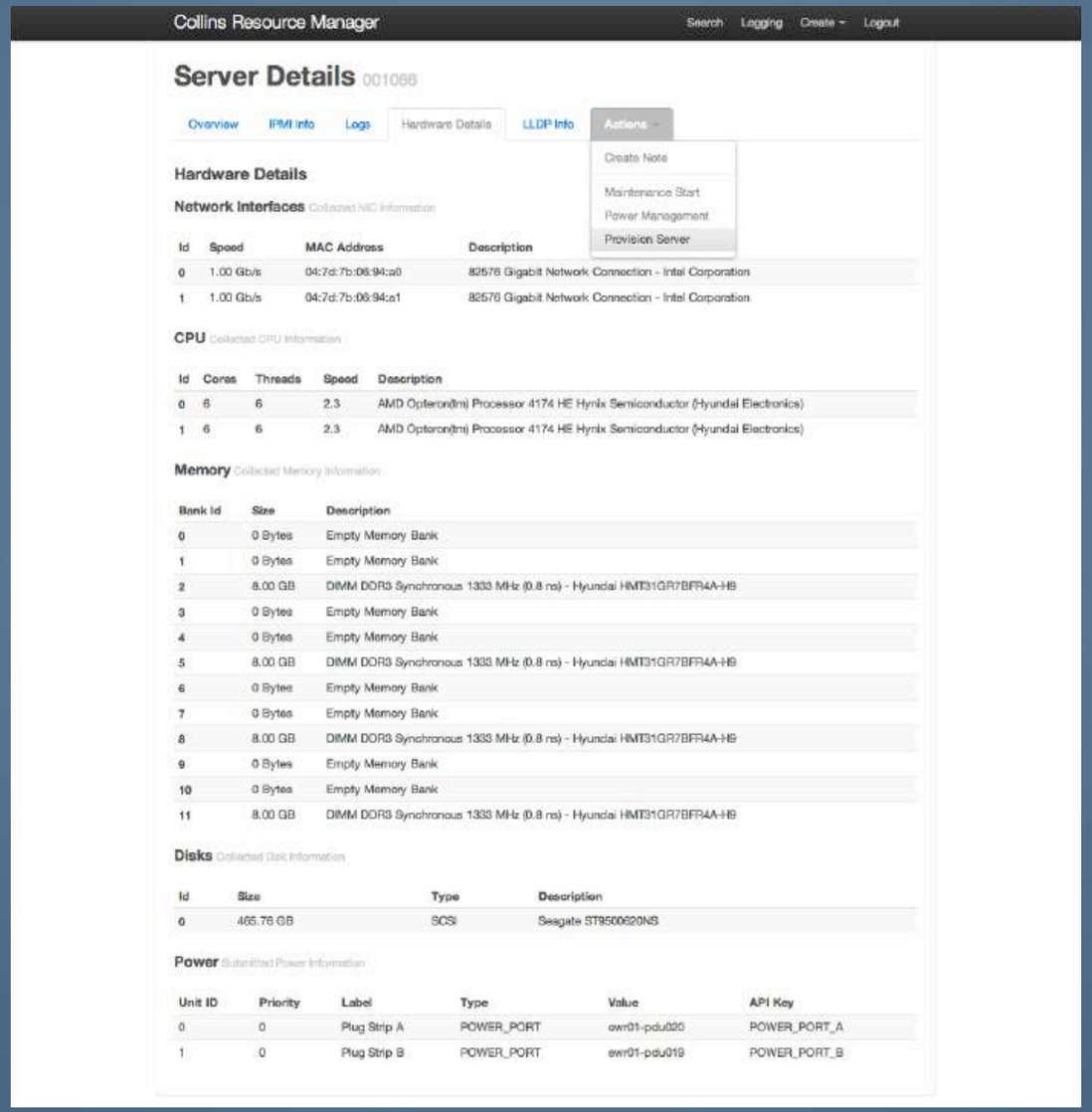
Collins is intentionally dumb. It worries about basic authentication, clean API interactions, and data persistence. If you start thinking, "Hey, I should build X into Collins", you probably shouldn't. Collins supports both a plugins architecture (for things that actually in some way change the behavior of collins) as well as a very usable API (including clients in Python, Ruby and Bash). Nearly everything you might want to do can be accomplished via the API and anything that can't is doable as a plugin.

Pages

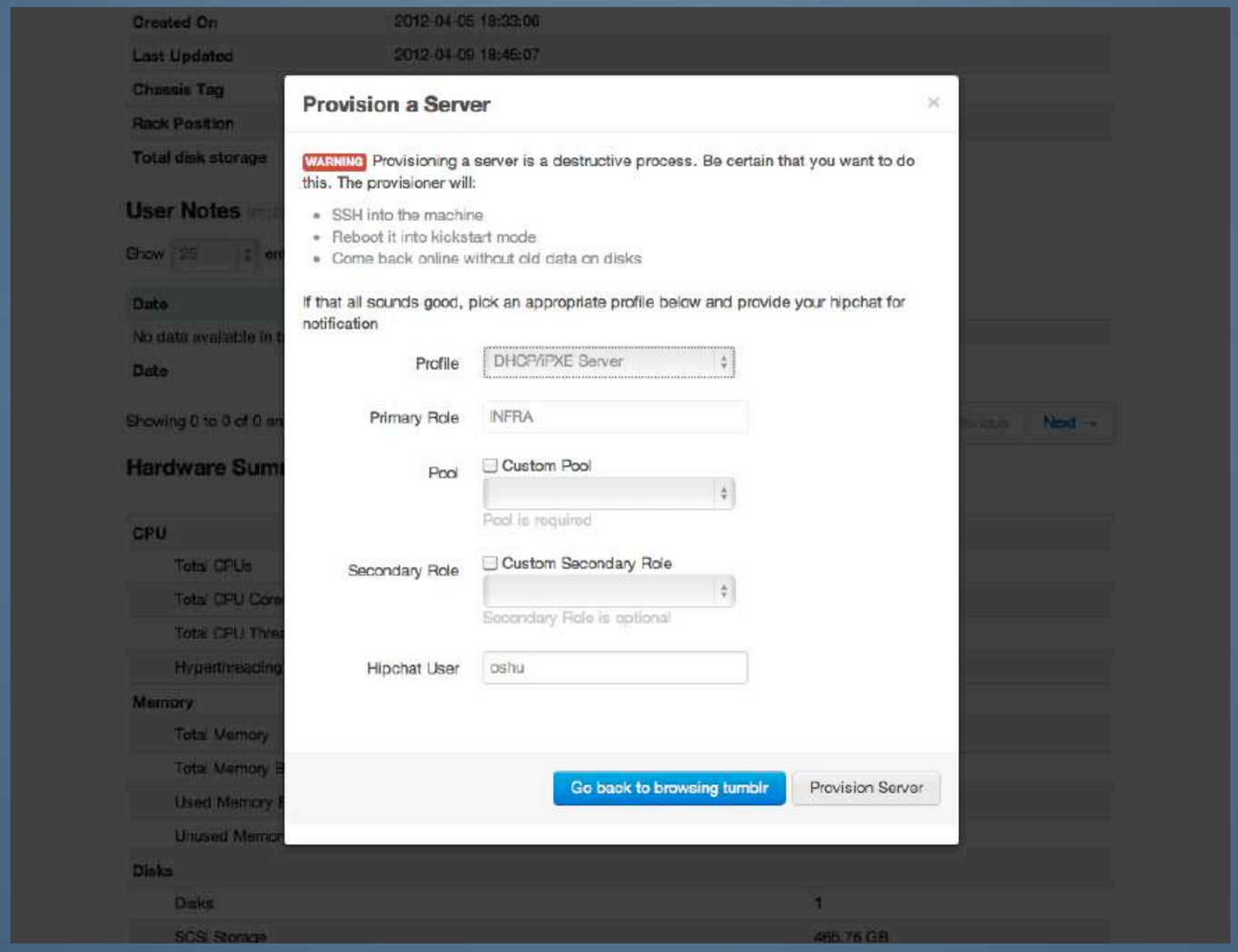
tumblr.

Collins Search

Collins Asset Details



Collins Provisioning



Phil

- iPXE dispatcher
- Kickstart generator
- Light Ruby app
- Collins API client

tumblr.

Server Intake Workflow

1. Rack and stack
2. Power on
3. Enter physical data

tumblr.

Server Intake Process

1. Server boots iPXE via DHCP/PXE
2. iPXE gets config from Phil
3. Phil sends Invisible Touch
4. IT updates firmware (if needed)
5. IT configures BIOS
6. IT configures BMC
7. IT uploads inventory data to Collins
8. IT starts stress tests
9. IT powers down server

Provisioning Workflow

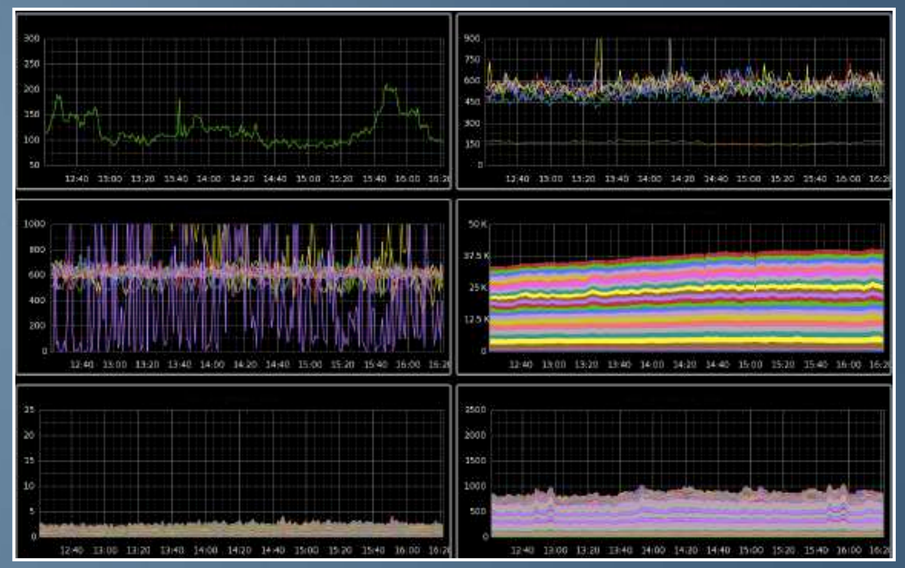
1. Search Collins
2. Choose Profile, Role, Pool
3. Click button

tumblr.

Provisioning Process

1. Server boots iPXE via DHCP/PXE
2. iPXE gets config from Phil
3. Phil sends install image
4. Install image gets Kickstart from Phil
5. Install runs Puppet in %post
6. End of %post calls back to Collins
7. Collins triggers vlan update
8. Collins triggers monitoring/trending
9. Added to production if “all green”

Result



Fast, scalable, no hassle provisioning!

Hurdles

Hurdles

- PXE kickstart w/ multiple NICs
- Network set up in %post
- Virident SSD set up in %post

PXE Kickstart / Multiple NICs

Phil iPXE config

```
initrd <%= os_install_url %>/images/initrd.img
kernel <%= os_install_url %>/images/vmlinuz ip=dhcp ksdevice=${mac}
```

Phil kickstart snippet

```
# network
network --bootproto=dhcp
```

tumblr.

%post Network Set Up

Phil kickstart snippet

```
# Bond Interface: <%= bond.name %>

cat > /etc/sysconfig/network-scripts/ifcfg-<%= bond.name %> <<EoF
DEVICE=<%= bond.name %>
BONDING_OPTS="<%= bond.options %>"
BOOTPROTO=static
IPADDR=<%= bond.address %>
NETMASK=<%= bond.netmask %>
GATEWAY=<%= bond.gateway %>
EoF
```

%post Viridٍent SSD Set Up

```
# Start the virident daemon
/etc/init.d/vgcd start
# create a device node
mknod /dev/vgca0 b 252 0
# create a mount point
mkdir -p /var/lib/mysql
# create partitions
parted -s /dev/vgca0 mklabel msdos
parted -s /dev/vgca0 unit s mkpart primary ext2 2048 100%
# make another device node
mknod /dev/vgca0p1 b 252 1
# make the filesystem
/sbin/mkfs.xfs -f -d su=64k,sw=3 -l size=32m,su=16k /dev/vgca0p1
# create fstab entry
echo "/dev/vgca0p1      /var/lib/mysql    xfs    noauto     0 0" >>/etc/fstab
# create virident config
cat > /etc/sysconfig/vgcd.conf << EoF
RESCAN_MD=1
RESCAN_LVM=1
MOUNT_POINTS="/var/lib/mysql"
RESCAN_MOUNT=1
EoF
# mount the virident
mount /var/lib/mysql
```

tumblr.

Lessons Learned

- Modularity is very important
- Hardware always has issues at scale
- Use modern Bash syntax
- 4 hour burn-in is not enough

tumblr.

Yes, we're hiring!

Joshua Hoffman
joshua@tumblr.com

tumblr.com/jobs

tumblr.

tumblr.