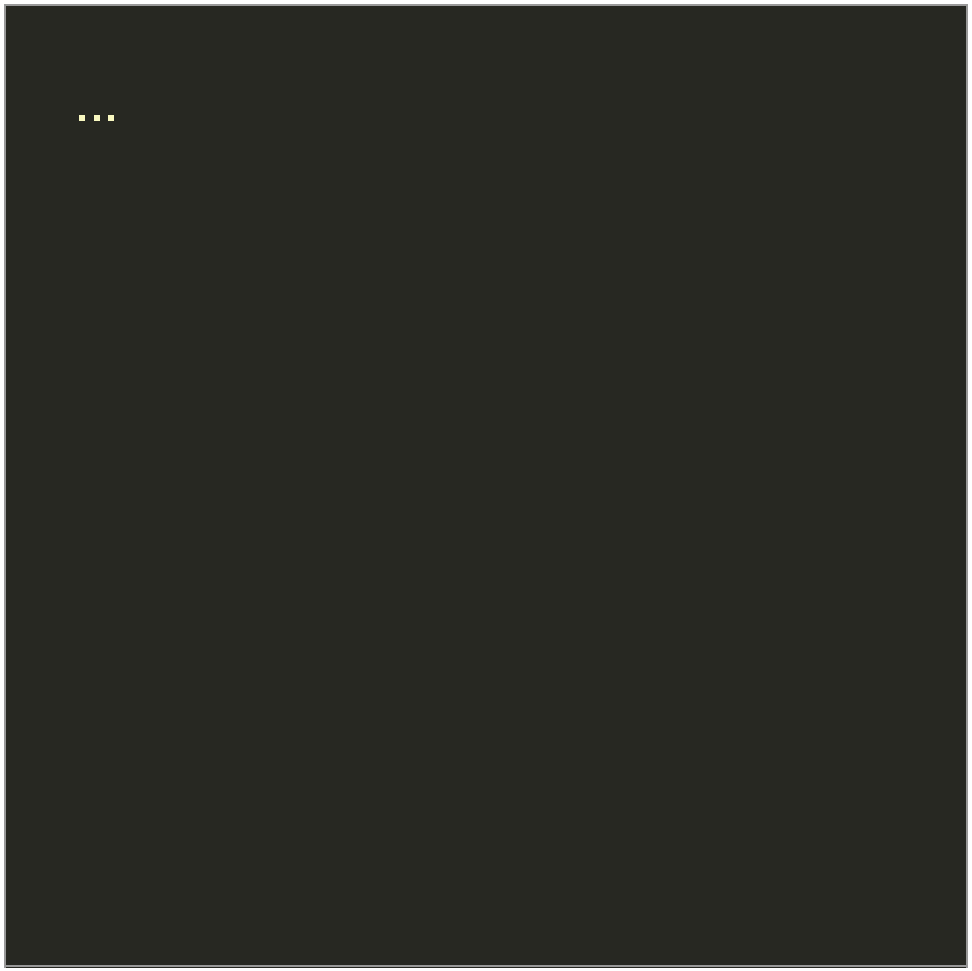
Doing the Mundane a Million Times a Minute

Mark Chadwick



Balancing ⇒

You Have Servers

You Have Requests

Requests Need To Get To Servers

(Quickly...)

Balancing ⇒

Take One: DNS Load Balancing

Balancing ⇒

Take One: DNS Load Balancing

Fault Tolerance

Distributed Request Balancing

Global Load Distribution


Meanwhile

Balancing ⇒

Take Two: Software

Balancing ⇒

Take Two: Software

Apache

(mod_proxy)

Nginx

(proxy_pass)

Balancing ⇒
Take Two: Software

Fewer DNS Entries

Faster Configuration Changes

Backend Isolation


Meanwhile

Balancing ⇒
Take Three: HAProxy + DNS

http://haproxy.1wt.eu

Fast as all get-out

Highly Configurable

Handles Hugely Concurrent Loads

splice()

Balancing ⇒

Take Three: HAProxy + DNS

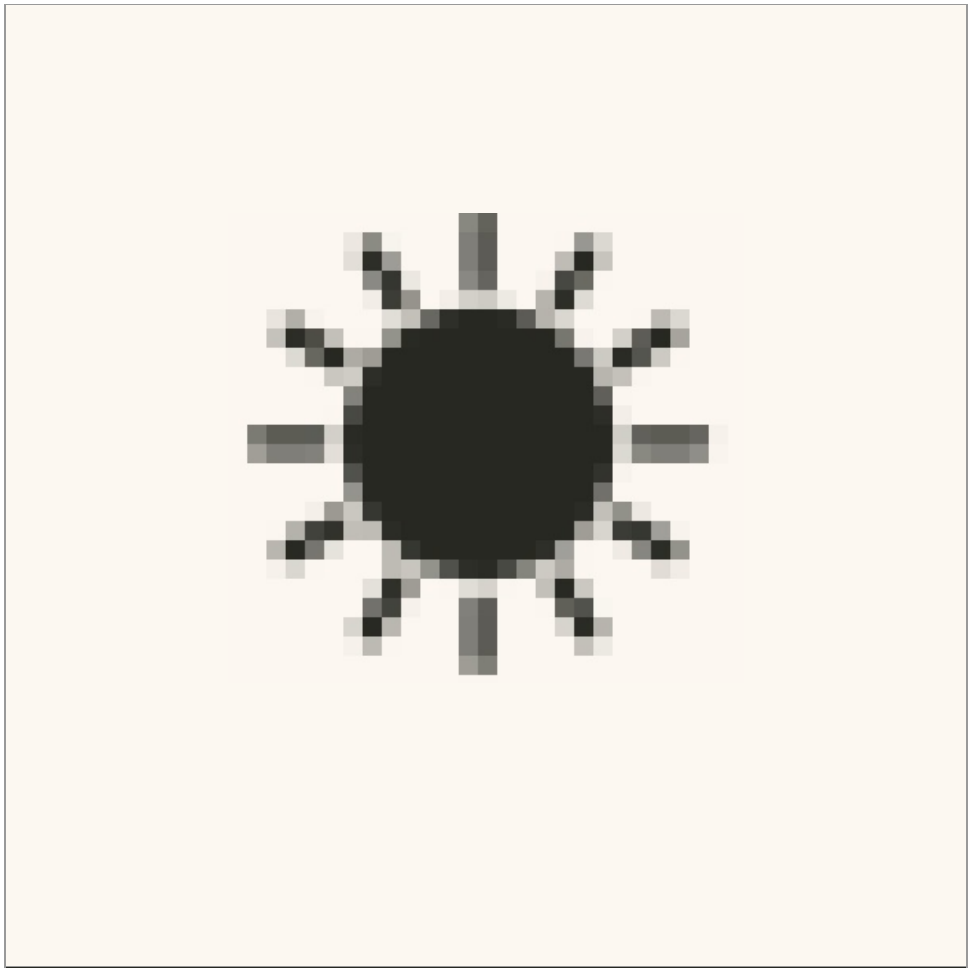
net.ipv4.tcp_max_syn_backlog

net.ipv4.tcp_max_tw_buckets

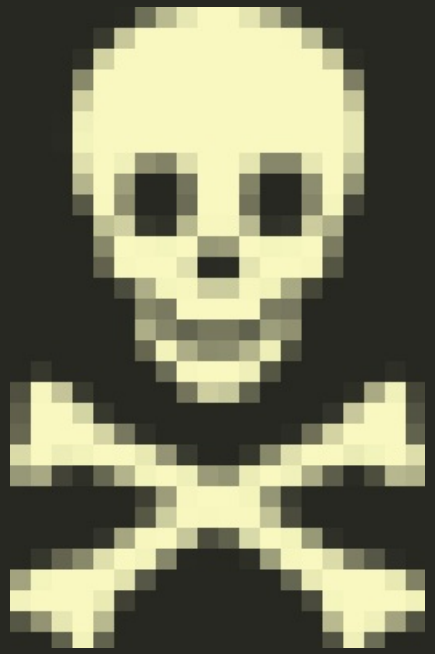
net.ipv4.tcp_netdev_max_backlo

net.ipv4.tcp_tw_recycle

~50k req/sec



Error Detection ⇒



Error Detection ⇒

Machines Fail

Code Fails

Infrastructure Fails

(You need to know about it)

Error Detection ⇒
Logging: Take 1

Log to Disk

SSH ain't broken

Error Detection ⇒
Logging: Take 1

Log to Disk

SSH ain't broken

Untenable after a just a handful of hosts

Error Detection ⇒

Logging: Take 2

Log to a centralized host

Now all the logs are together

Error Detection ⇒
Logging: Take 2

Log to a centralized host

Now all the logs are together

Log Host gets overwhelmed with hundreds of machines

Error Detection ⇒
Logging: Take 3

Log to a distributed set of hosts

Files spread across a distributed file system

Error Detection ⇒
Logging: Take 3

Log to a distributed set of hosts

Files spread across a distributed file system

Finally Works!

Error Detection ⇒
Logging: Take 3

Log to a distributed set of hosts

Files spread across a distributed file system

Finally Works!

Way too much information

Error Detection ⇒
Logging: Take 4

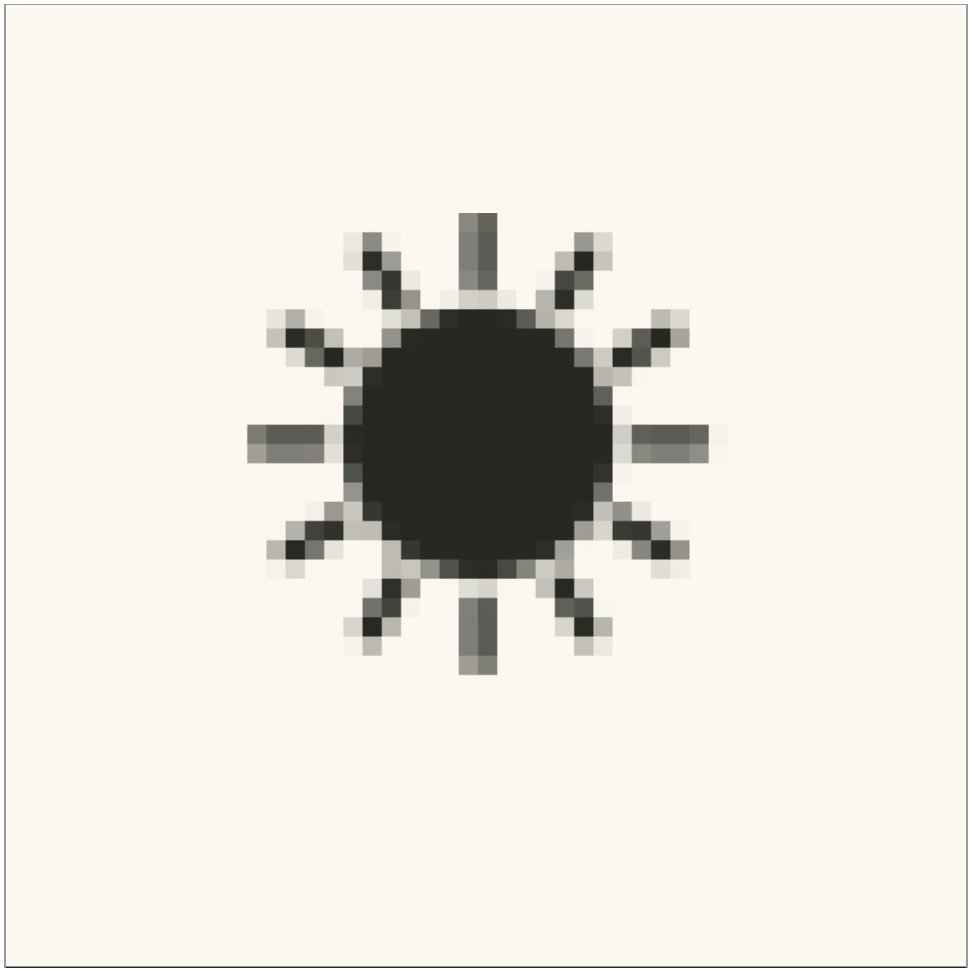
Screw Logging (j/k...sorta)

Just Count Stuff

Error Detection ⇒
Counting Stuff

Graphite

http://graphite.wikidot.com



Error Detection ⇒
System Monitoring: Take 1

Zenoss

http://zenoss.com/

Error Detection ⇒

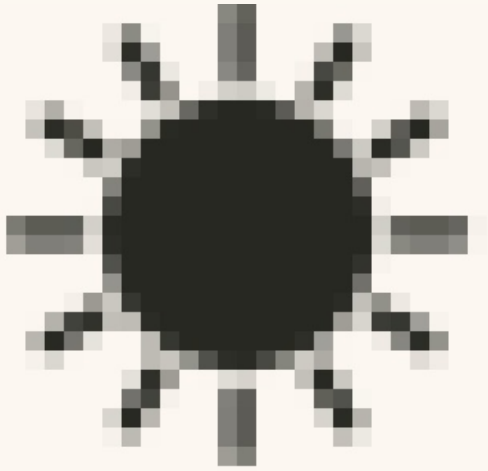
System Monitoring: Take 1

Zenoss

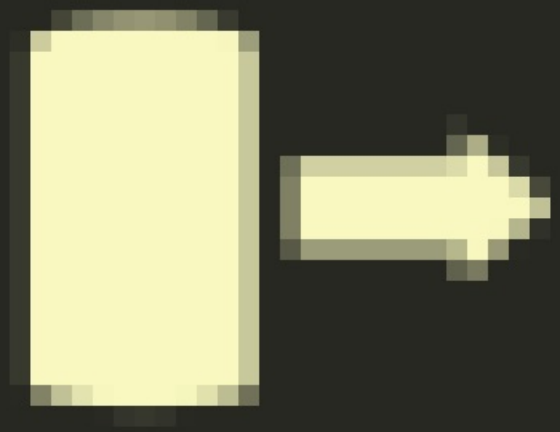
http://zenoss.com/

Can get pret-ty toasty with hundreds of machines

Scale Vertically!
Good Enough!



Moving Data ⇒



Moving Data ⇒

Your application emits records

Something needs to consume them

You need to store them somewhere

Moving Data ⇒
Take One

Your transaction records are published...

Subscribers want to process them

PubSub!


Meanwhile

Moving Data ⇒
Take Two

If the middleman is getting killed...

And we know who's going to process the data anyway...

RPC!


Meanwhile

Moving Data ⇒
Take Three

Network calls per-message will fail

So we need to spool somewhere

Disk!

A Brief Interlude ⇒
Estimating

10,000 req/s

1k logging / req

~ 10MB/s

~ 0.8TB/day

A Brief Interlude ⇒
Estimating

DC Bandwidth

Disk bandwidth

A Brief Interlude ⇒
Estimating

When your traffic is constant, storage increases linearly

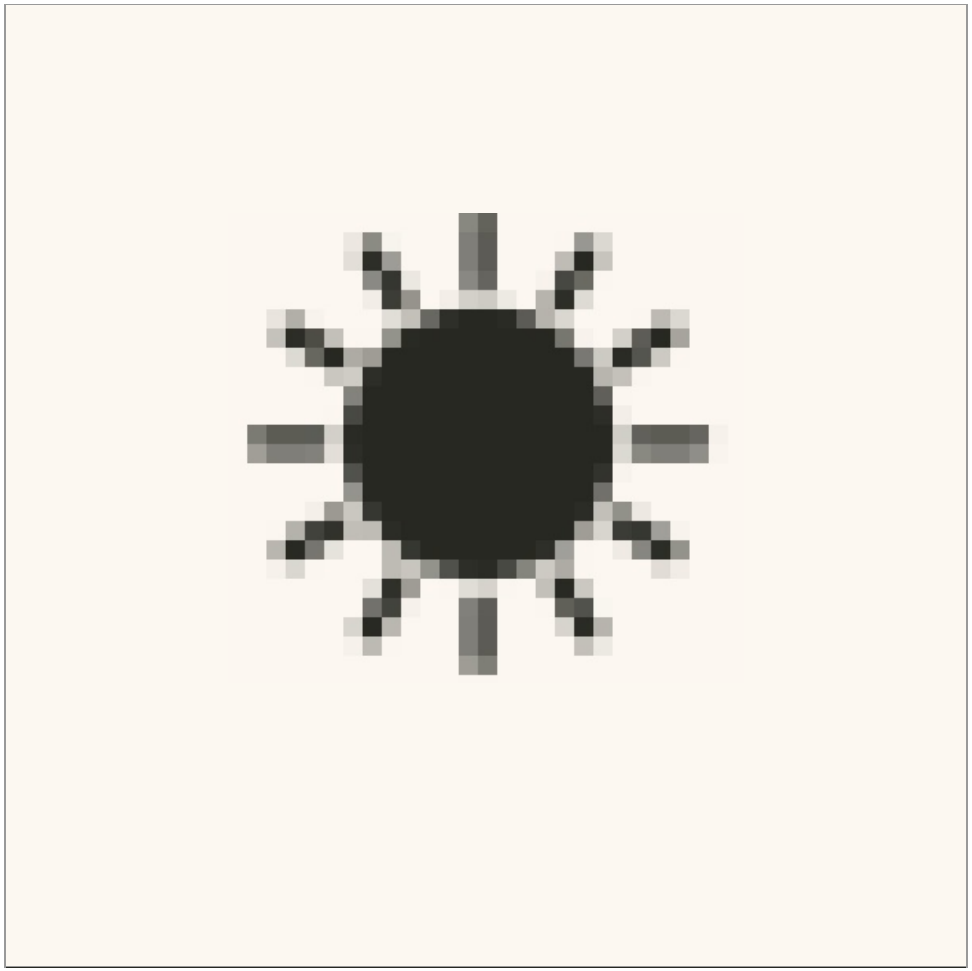
When your traffic increases linearly...

Moving Data ⇒
Take Three

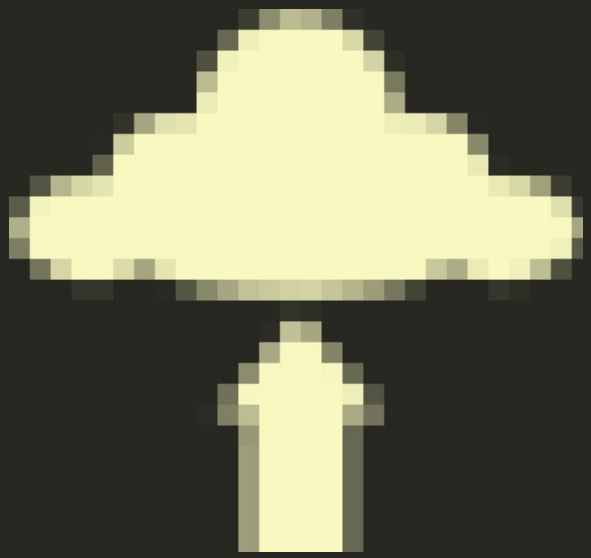
Compression

Yes

Please



Deploying



Deploying

Deploying to 100's of machines

Canary'ing code

Cascading failures during upgrades

Cross-version interactions

Global Deployments

Deploying ⇒
Take 1: Source Checkout

If each machine needs code...

Update code on each machine!
Restart.

Doesn't scale past a few nodes

Deploying ⇒
Take 2: Single Deploy Host

If we automate how the code is deployed...

Write scripts to push code from a single host!

Deployments take hours

Inconsistent/Dead Machines

Deploying ⇒
Take 3: Deployment System

We'll use a deployment system!

Chef, CFEngine, Puppet, et all.

Deploy machines melt

Deploying ⇒

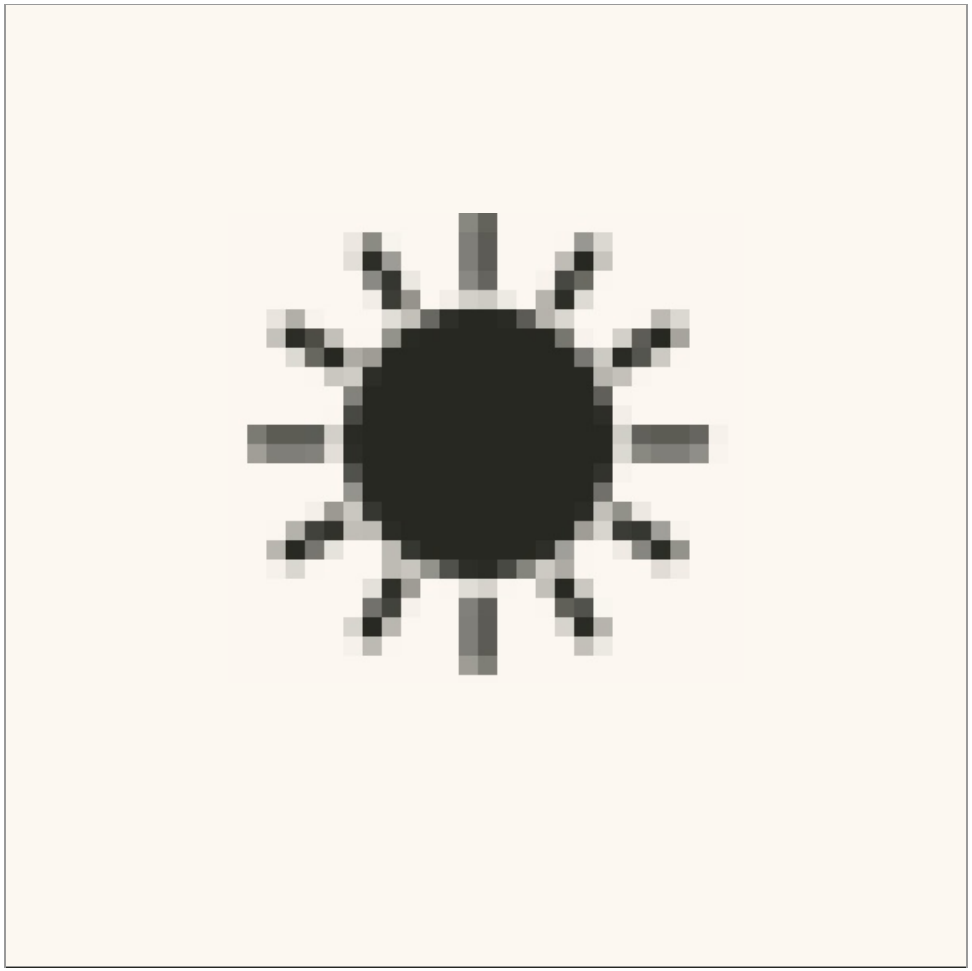
Take 4: Distributed Deployment System

- Distribute Chef Components

Deploying ⇒

Take 4: Distributed Deployment System

- Distribute Chef Components
- CouchDB
- RabbitMQ
- Solr Indexer
- Web UI



Summary ⇒

It's easy to outgrow components of your stack

Component failures may be quite subtle

Iterative improvements are often test best you can do

Questions?

Thanks!