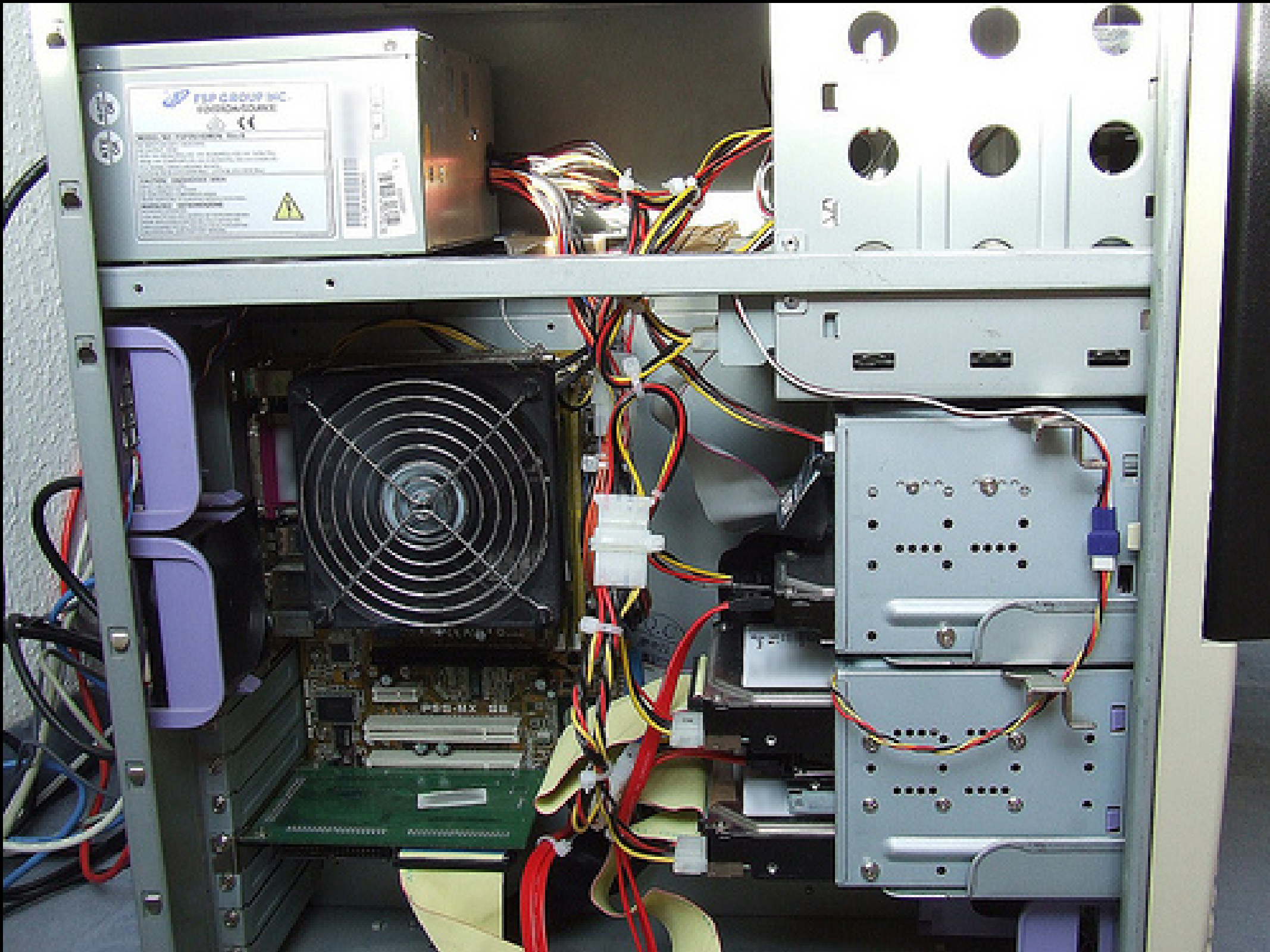
Scalable, Good, Cheap

a tale of sexiness, puppets, shell scripts, and python

From this...



...to this!

Get your infrastructure started right!

(not just preparing for incident and rapid event response)

Who we are?

Avleen Vig (@avleen)

- Senior Systems Engineer at Etsy
- Good at: Scaling frontends, python
- Previous companies: WooMe, Google, Earthlink

Marc Cluet (@lynxman)

- Senior Systems Engineer at WooMe
- Good at: Backend scaling, bash/python, languages
- Previous companies: RTFX, Tiscali, World Online

Overview

- Workflow
- Why planning for scaling is important
- How do you choose your software
- Setting up your infrastructure
- Managing your infrastructure

The background

- Larger startup, $32m in funding
- 6 million+ active users
- Dozens of developers
- 6 systems administrators
- 4 DBAs
- 10+ code releases every day
- Geographically distributed employees
  - Brooklyn HQ
  - Satellites in Berlin, San Francisco
  - Small number of remote employees

The background

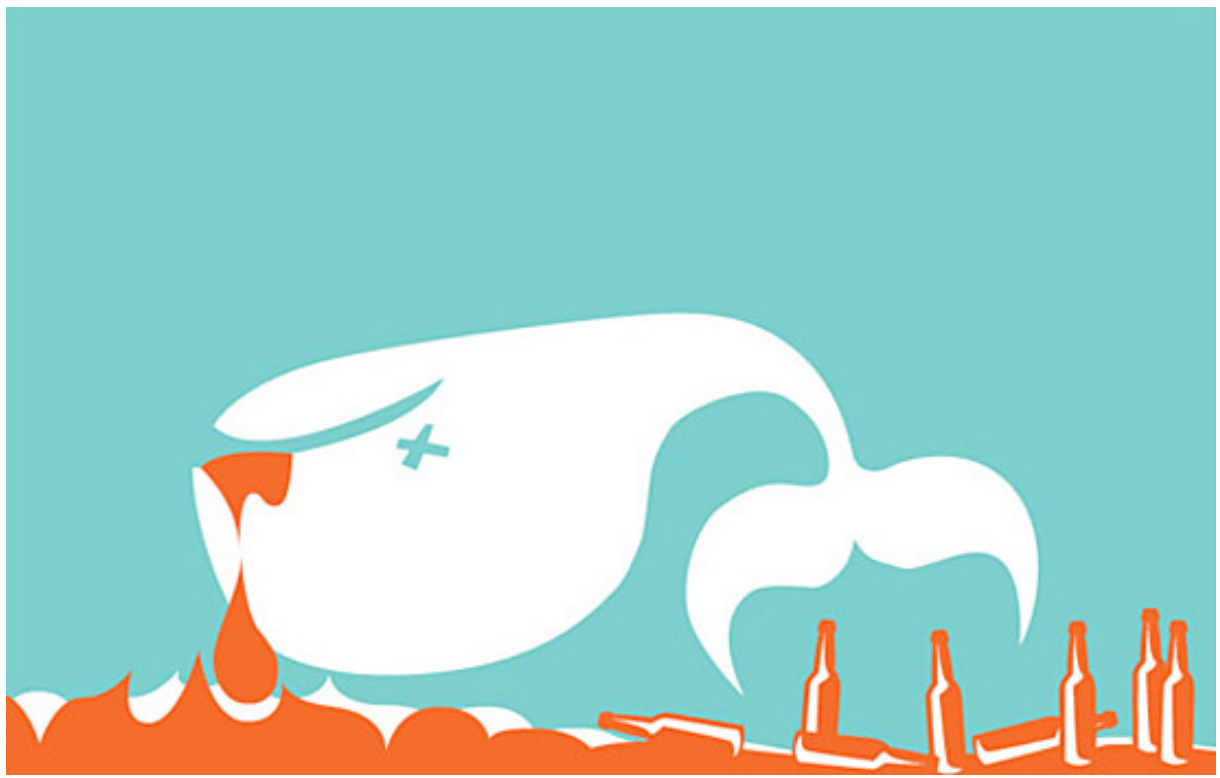
- Small, funded start up
- 6 python developers
- 2 front end developers
- 3 systems administrators
- 1 DBA (moustache included)
- Multiple code releases every day
- Geographically distributed employees
  - Berlin, Copenhagen, Leeds, London, Los Angeles, Oakland, Paris, Portland, Zagreb

Workflow

- Ticket systems
  - Ticket, or it didn't happen!
- Documentation
  - Wikis are good
- Don't Repeat Yourself
  - If you keep doing the same thing manually, automate
- Version control everything
  - All of your scripts
  - All of your configurations

Workflow

- Everything will change
- Technical debt vs Premature optimisation
  - If you try to be too accurate too early, you'll fail



Team integration

- Be sure to hire the right people
  - Beer recruitment interview
- Encourage speed
  - Release soon and release often
- Embrace mistakes as part of your day to day
  - Learn to work with it
- Ask for peer reviews for important components
  - Helps sanity checking your logic
- Developers, Sysadmins, DBAs, *one team*

Team communication

- Team communication is the most critical factor
- Make sure everyone is in the loop
- Useful applications
  - IRC
  - Skype
  - email
  - shout!
- Don't be afraid to use the phone to avoid miscommunication

Layering! Not just for haircuts.

Separate your systems

- Front end
- Application
- Database
- Caching

Choosing your software

- What does your software need to do?
  - FastCGI / HTTP proxy? Use nginx
  - PHP processing? Use apache
- What expertise do you already have?
  - Stick to what you're 100% good at
- Don't rewrite everything
  - If it does 70% of what you need it's good for you

Release management

- Fast and furious
- Automate, automate, automate
- Script your deploys and rollbacks
- Continuous deployment
- MTTR vs MTBF

MTTR vs MTBF

Maintainability

Versus

MTTR Optimized

MTBF Optimized

More info here: http://ti.arc.nasa.gov/projects/ishem/Papers/ONeill_Maintainability.doc

Logging

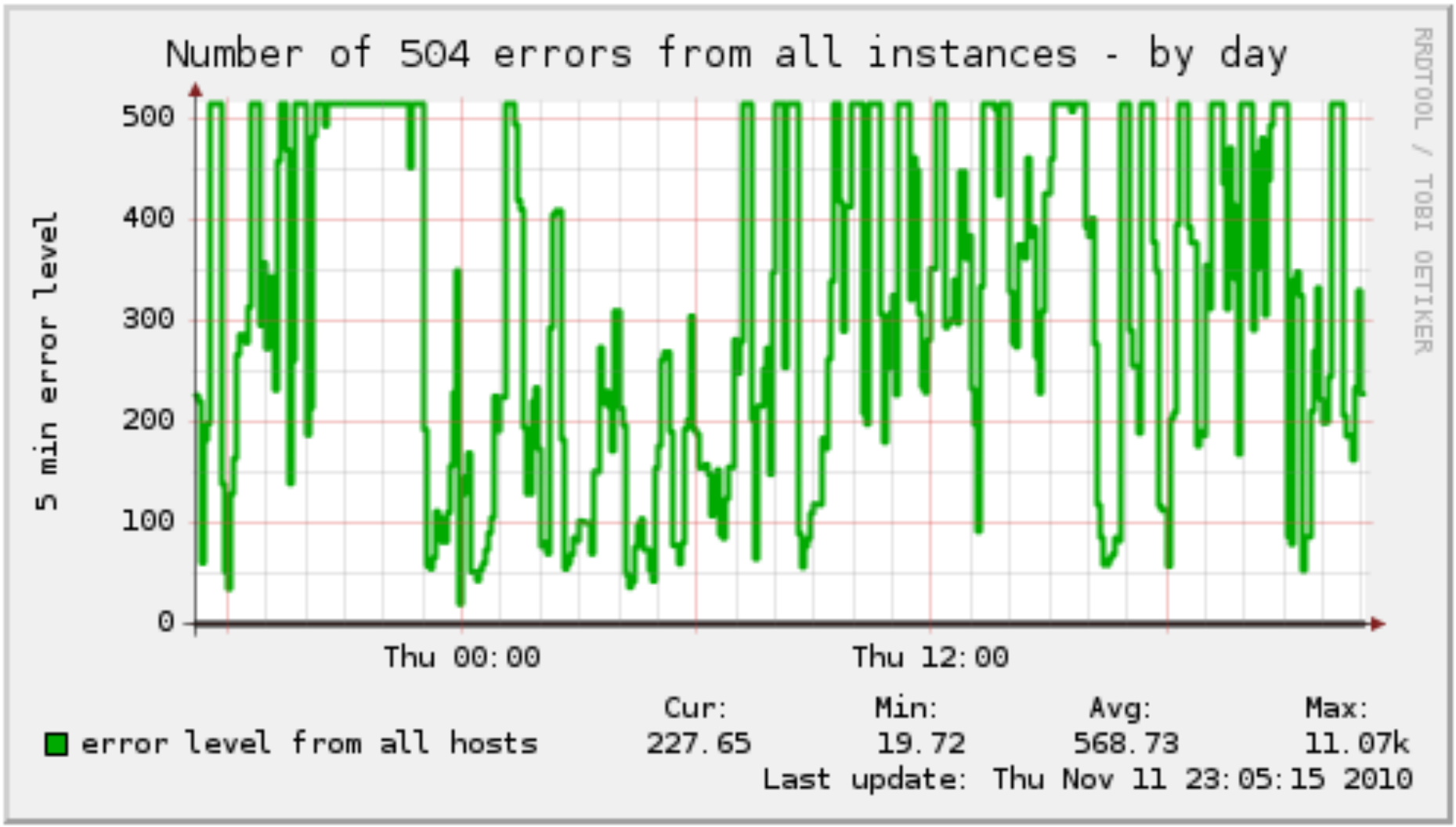
- Centralize your logging
  - syslog-ng
- Parsing web logs - the secret troubleshooting weapon
  - SQL
  - Splunk

Web logs in a database!

```
CREATE TABLE access (
    ip inet,
    hostname text,
    username text,
    date timestamp without time zone,
    method text,
    path text,
    protocol text,
    status integer,
    size integer,
    referrer text,
    useragent text,
    clienttime double precision,
    backendtime double precision,
    backendip inet,
    backendport integer,
    backendstatus integer,
    ssl_cipher text,
    ssl_protocol text,
    scheme text
);
```

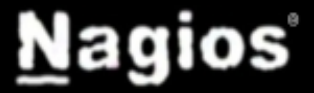
Web logs in a database!

:: *Number of 504 errors from all instances*



Monitoring

- Alerting vs Trend analysis



General

- Home
- Documentation

Monitoring

- Tactical Overview
- Service Detail
- Host Detail
- Hostgroup Overview
- Hostgroup Summary
- Hostgroup Grid
- Servicegroup Overview
- Servicegroup Summary
- Servicegroup Grid
- Status Map
- 3-D Status Map
- Service Problems
- Host Problems
- Network Outages

Show Host:

- Comments
- Downtime
- Process Info
- Performance Info
- Scheduling Queue

Reporting

- Trends
- Availability
- Alert Histogram
- Alert History
- Alert Summary
- Notifications
- Event Log

Configuration

- View Config

Tactical Monitoring Overview

Last Updated: Thu Nov 11 16:44:22 PST 2010
Updated every 90 seconds
Nagios® 2.11 - www.nagios.org
Logged in as *avleen*

Monitoring Performance

Service Check Execution Time:	0.01 / 10.12 / 0.416 sec
Service Check Latency:	0.00 / 0.36 / 0.144 sec
Host Check Execution Time:	0.01 / 5.04 / 0.247 sec
Host Check Latency:	0.00 / 0.00 / 0.000 sec
# Active Host / Service Checks:	81 / 1524
# Passive Host / Service Checks:	0 / 0

Network Outages

0 Outages

Network Health

Host Health:	
Service Health:	

Hosts

0 Down	0 Unreachable	81 Up	0 Pending

Services

6 Critical	6 Warning	0 Unknown	1512 Ok	0 Pending
1 Unhandled Problems 5 Acknowledged	6 Unhandled Problems			

Monitoring Features

Flap Detection	Notifications	Event Handlers	Active Checks	Passive Checks
Disabled	Enabled	Enabled	Enabled	Enabled
N/A	1 Service Disabled All Hosts Enabled	All Services Enabled All Hosts Enabled	All Services Enabled All Hosts Enabled	All Services Enabled All Hosts Enabled

Monitoring

- Alerting vs Trend analysis
  - Nagios is great for raising alerts on problems

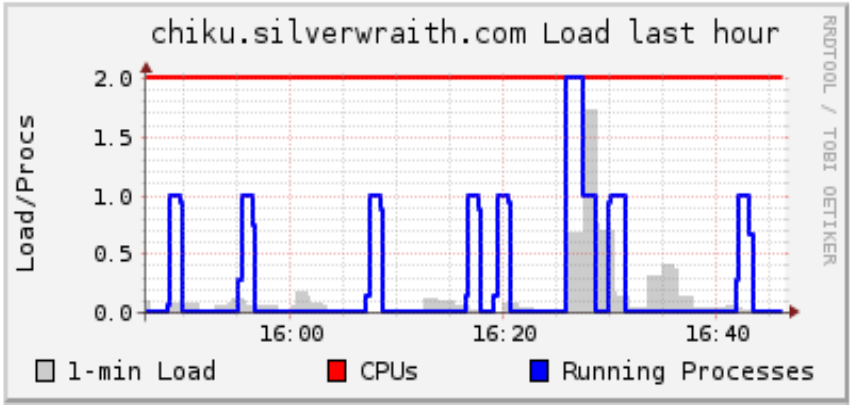
This host is up and running.

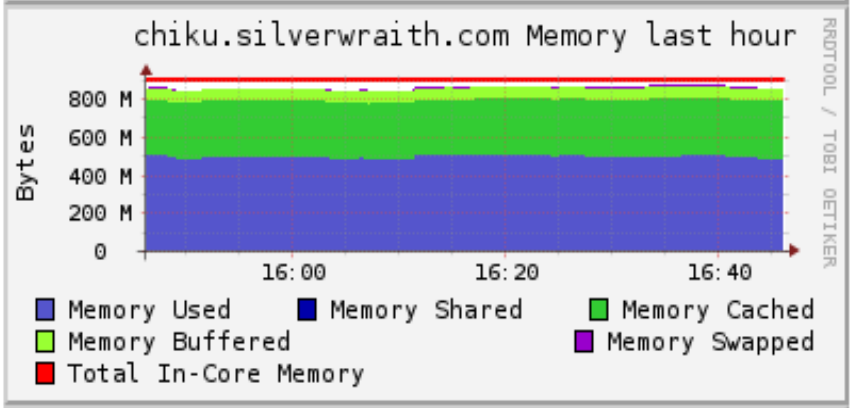
Time and String Metrics

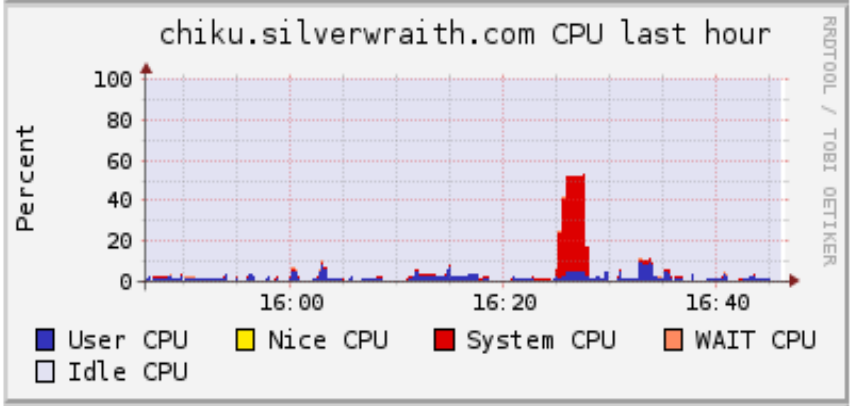
Last Boot Time	Tue, 31 Aug 2010 16:36:59 -0700
Gexec Status	OFF
Gmond Started	Mon, 06 Sep 2010 12:16:42 -0700
Last Reported	0 days, 0:00:03
Machine Type	x86
mysql_version	5.1.47-rel11.2-log
Operating System	Linux
Operating System Release	2.6.26-2-xen-686
Uptime	72 days, 1:08:55

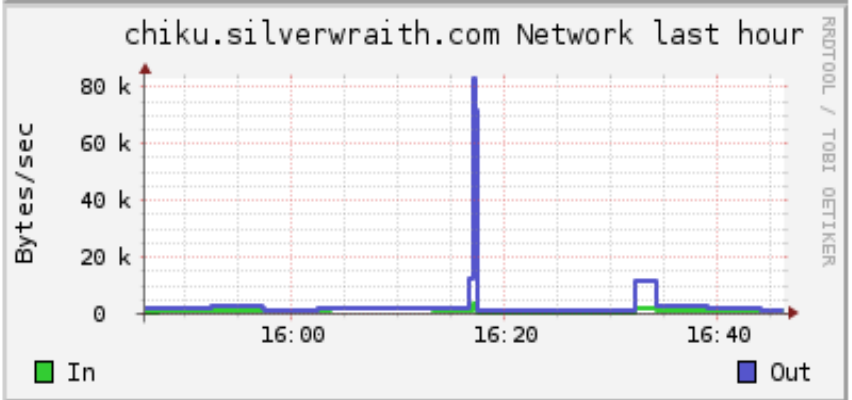
Constant Metrics

CPU Count	2 CPUs
CPU Speed	2795 MHz
Memory Total	917700 KB
Swap Space Total	0 KB









Monitoring

- Alerting vs Trend analysis
  - Nagios is great for raising alerts on problems
  - Ganglia is great at long term trend analysis
  - Know when something is out of the "ordinary"

Monitoring

- Alerting vs Trend analysis
  - Nagios is great for raising alerts on problems
  - Ganglia is great at long term trend analysis
  - Know when something is out of the "ordinary"
- What should you monitor?
  - Anything which breaks once
  - Customer facing services

Monitoring

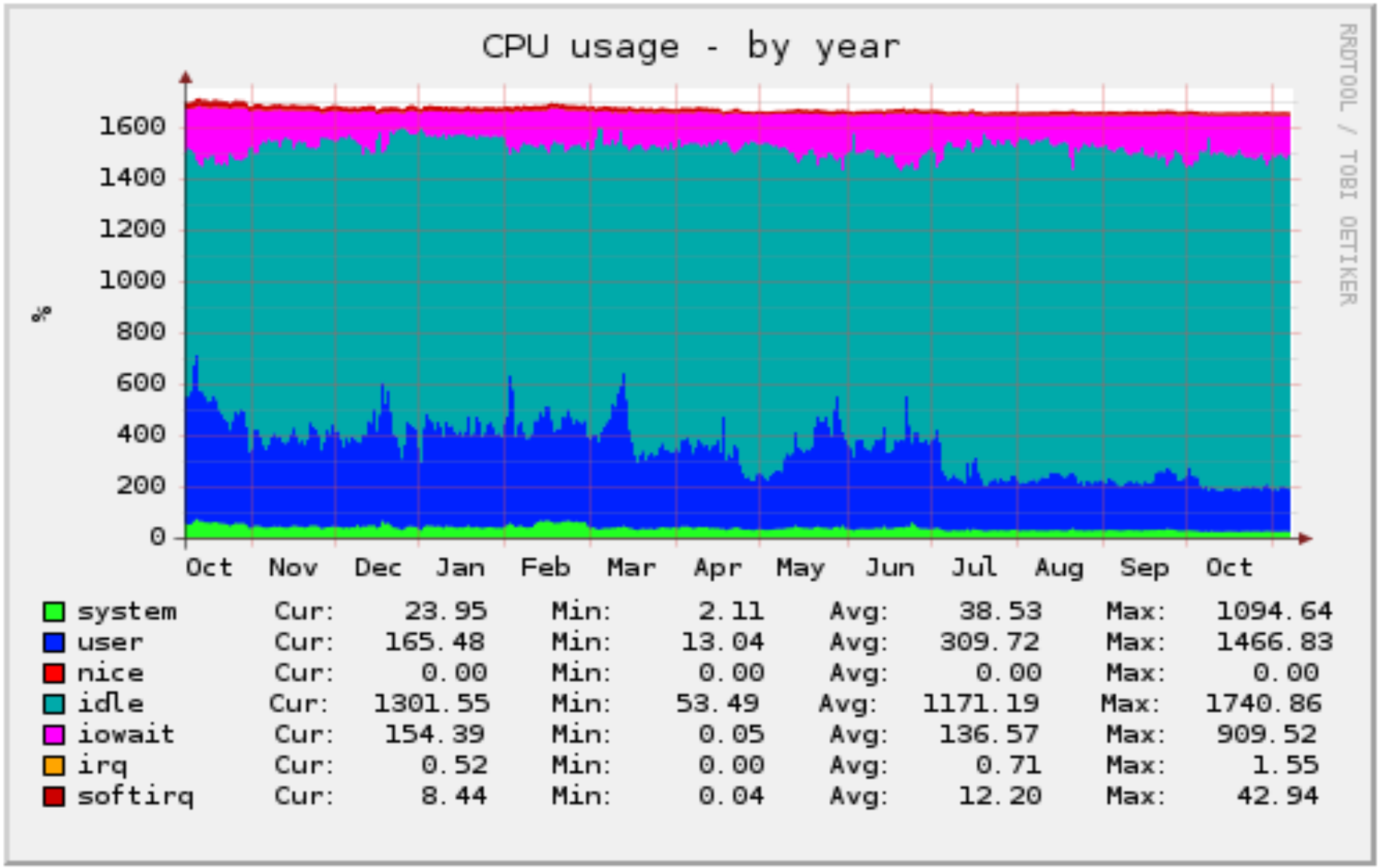
- Alerting vs Trend analysis
  - Nagios is great for raising alerts on problems
  - Ganglia is great at long term trend analysis
  - Know when something is out of the "ordinary"
- What should you graph?
  - Everything! If it moves, graph it.
  - Customer facing rates and statistics

Monitoring

Get statistics from your logs:

- PostgreSQL: pgfouine
- MySQL: mk-query-digest
- Web servers: webalizer, awstats, urchin
- Custom applications: Do it yourself! Integrate with Ganglia

Monitoring


CPU usage - by year
RRDTOOL / TOBI OETIKER
%
1600
1400
1200
1000
800
600
400
200
0
Oct Nov Dec Jan Feb Mar Apr May Jun Jul Aug Sep Oct
system Cur: 23.95 Min: 2.11 Avg: 38.53 Max: 1094.64
user Cur: 165.48 Min: 13.04 Avg: 309.72 Max: 1466.83
nice Cur: 0.00 Min: 0.00 Avg: 0.00 Max: 0.00
idle Cur: 1301.55 Min: 53.49 Avg: 1171.19 Max: 1740.86
iowait Cur: 154.39 Min: 0.05 Avg: 136.57 Max: 909.52
irq Cur: 0.52 Min: 0.00 Avg: 0.71 Max: 1.55
softirq Cur: 8.44 Min: 0.04 Avg: 12.20 Max: 42.94

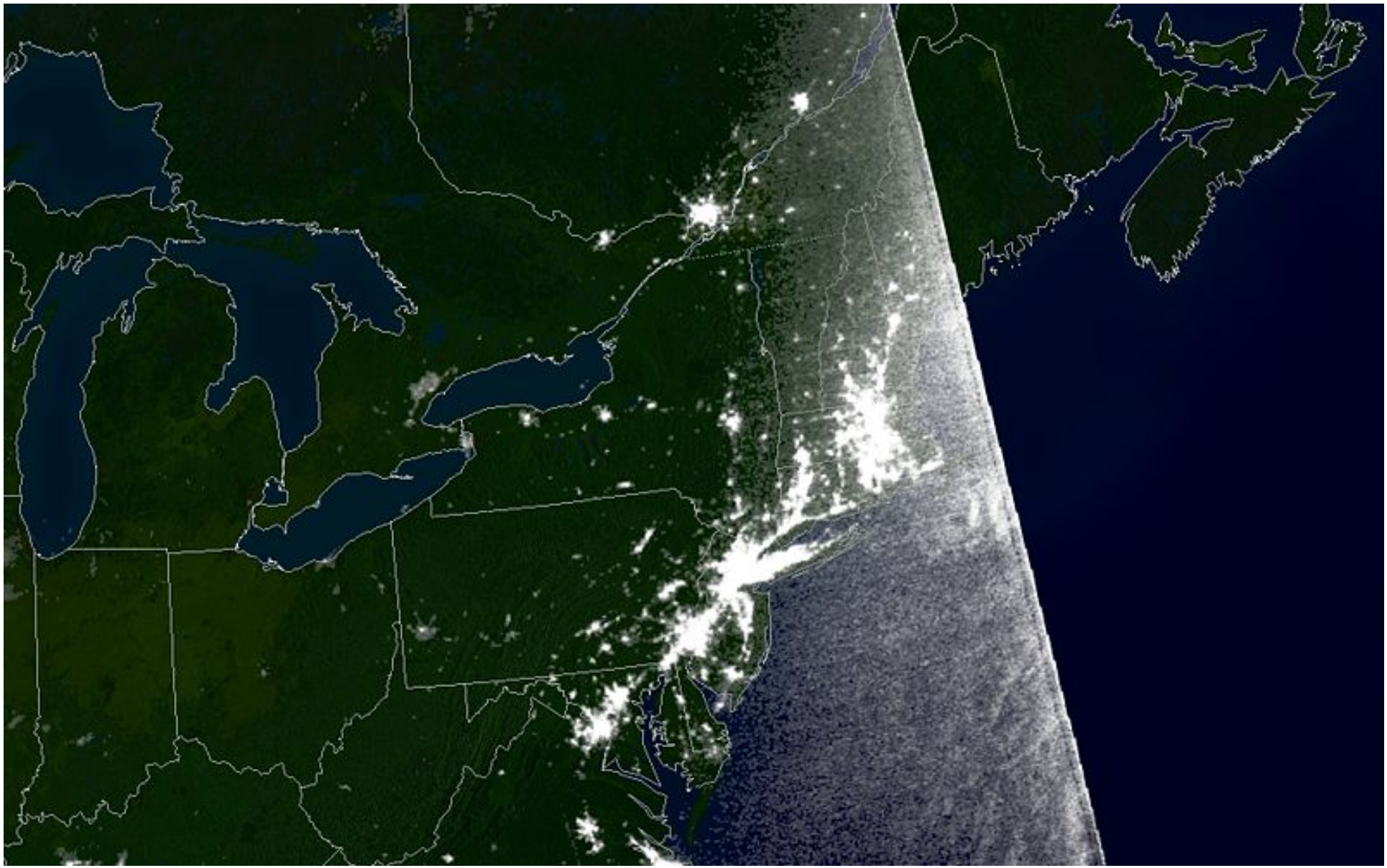
Caching

- Caches are disposable

Caching

- Caches are disposable
- But what about the thundering herd?

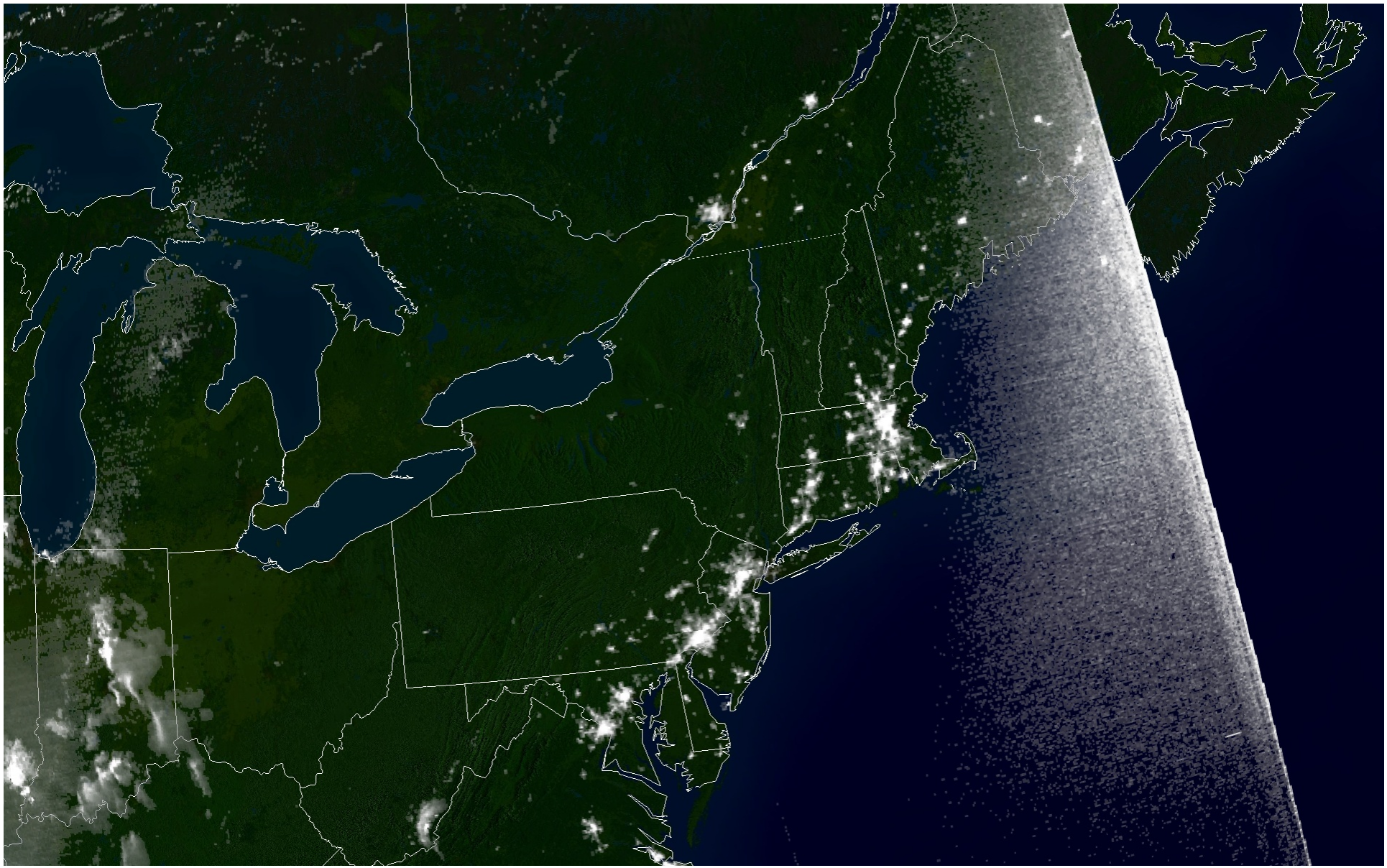
The importance of scaling



The importance of scaling

- August 2003 Northeastern US and Canada blackout
  - Caused by poor process execution
  - Lack of good monitoring
  - Poor scaling

The importance of scaling



The importance of scaling

- Massive destruction avoided!
  - 256 power stations automatically shut down
  - 85% after disconnecting from the grid
  - Power lost but plants saved!

Caching

- Caches are disposable
- But what about the thundering herd?
  - Increase backend capacity along with cache capacity
  - Plan for cache failure
  - Reduce demand when cache fails

Caching

- Find out how your caching software works
  - Memcache + peep!
  - Is it better with lots of keys and small objects?
  - Or fewer keys and large objects?
  - How is memory allocated?

Caching

- Caches are disposable
  - Solved!
- But what about the thundering herd?
  - Solved!
- Now we get into database scaling!
  - Over to Marc...

Databases

Databases...

or how to live and die dangerously

Databases

SQL or NoSQL?

Databases

- SQL
  - Gives you transactional consistency
  - Good known system
  - Hard to scale

- NoSQL
  - Transactionally consistent "eventually"
  - New cool system
  - Easy to scale

Databases

- SQL
  - Gives you transactional consistency
  - Good known system
  - Hard to scale

- NoSQL
  - Transactionally consistent "eventually"
  - New cool system
  - Easy to scale

You may end up using BOTH!

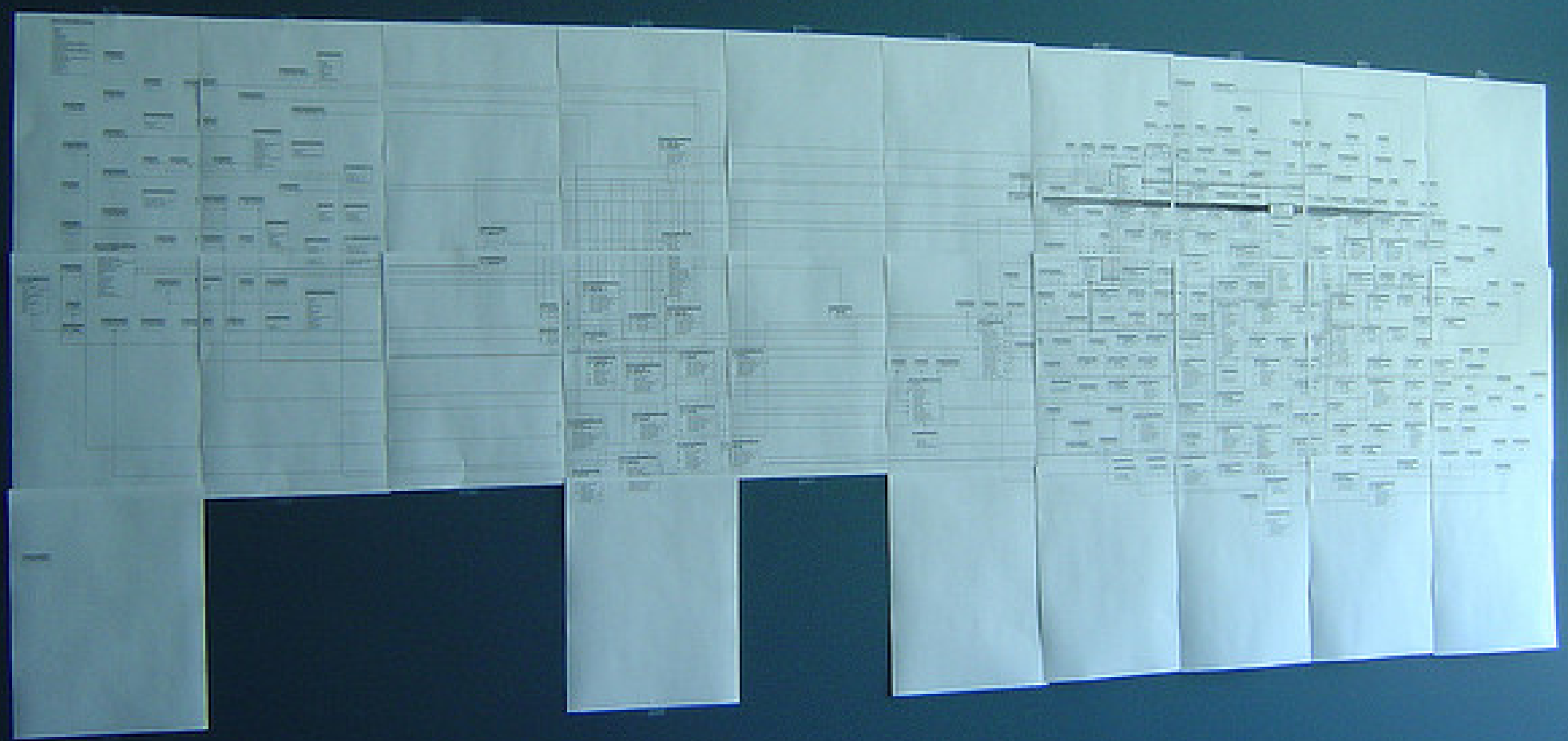
Databases

- Be smart about your table design

Databases

- Be smart about your table design
  - Keep it simple but modular to avoid surprises

You need to design your database right!



Databases

- Be smart about your table design
  - Keep it simple but modular to avoid surprises
  - Don't abuse many-to-many tables, they will just give you hell

Databases

- Be smart about your table design
  - Keep it simple but modular to avoid surprises
  - Don't abuse many-to-many tables, they will just give you hell

- **YOU WILL GET IT WRONG**
  - You'll need to redesign parts of your DB semi-regularly
  - Be prepared for the unexpected

Databases

The read dilemma

- As the tables grow so do read times and memory. Several options:
  - Check your slow query log, tune indexes
  - Partition to read smaller numbers of rows
  - Master / Slave, but this adds replication lag!

Databases

The read dilemma

- As the tables grow so do read times and memory.
  Several options:
    - Check your slow query log, tune indexes
        - Single most common problem with slow queries and capacity
        - Be careful about foreign keys

Databases

The read dilemma

- As the tables grow so do read times and memory. Several options:
  - Check your slow query log, tune indexes
  - Partition to read smaller numbers of rows
    - By range (date, id)
    - By hash (usernames)
    - By anything you can imagine!

Databases

The write conundrum

- As the database grows so do writes
- Writes are bound by disk I/O
  - RAID1+0 helps
- Don't shoot yourself in the foot!
  - Don't try to solve this early
  - Have monitoring ready to foresee this issue
  - Bring pizza

Databases

Divide writes!

- Remember about modular? This is it

Databases

How to give a consistent view to the servers?

Use a query director!

- pgbouncer on Postgres
- gizzard on MySQL

Web frontend

- Hardware load balancers - Good but expensive!
- Software load balancers - Good and cheap! (more pizza)
  - Web server frontends
    - nginx, lighttpd, apache
  - Reverse proxies
    - varnish, squid
  - Kernel stuff
    - Linux ipvs

Web frontend

Which way should I go?

- Web servers as load balancers
  - Gives you nice add on features
  - You can offload some process in the frontend
  - Buffering problems

- Reverse proxies
  - Caching stuff is good
  - Fast reaction time
  - No buffering problems

Web frontend

Divide your web clusters!

- You can send different requests to different clusters
- You can use an API call to connect between them

Configuration management

- Be ready to mass scale
  - Keep all your machines in line
- Automated server installs
  - Use it to install new software
  - Also to rapidly deploy new versions

Writing tools

- If you do something more than 2 times it's worth scripting
- Write small tools when you need them
- Stick to one or two languages
  - And be good at them

Writing tools

- Even better
- Have your scripts repo in a cvs and push it everywhere

```
changeset:   3646:455b84f75c21
user:        mcluet
date:        Mon Oct 25 10:14:56 2010 -0700
summary:     Marc hates typos
```

Backups

- It's important to have backups

Backups

- It's important to have backups
- It's even more important to exercise them!
  - Having backups without testing recovery is like having no backups

Backups

- It's important to have backups
- It's even more important to exercise them!
  - Having backups without testing recovery is like having no backups
- How can we exercise backups for cheap?

Backups

- It's important to have backups
- It's even more important to exercise them!
  - Having backups without testing recovery is like having no backups
- How can we exercise backups for cheap?
  - Cloud computing!

Cloud computing

- Cloud computing help us recreate our platform on the cloud
- Giving us a more than credible recovery scenario
- Also very useful to spawn more instances if we run into problems

Interesting things to read

Wikipedia
- http://en.wikipedia.org/wiki/DevOps

Web Operations and Capacity Planning
- http://kitchensoap.com

High scalability (if you get there)
- http://highscalability.com/

If you really fancy databases, explain extended
- http://explainextended.com/

Questions?

Work at Etsy!
http://etsy.com/jobs

Etsy

@avleen

Work at WooMe!
http://bit.ly/work4woome

@lynxman