

Operational Buddhism

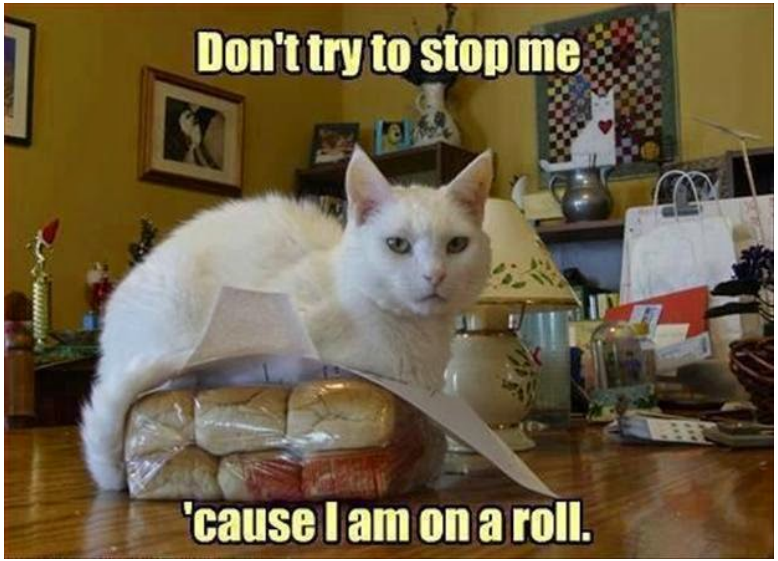
Building Reliable Services From Unreliable Components

Ernie Souhrada
Database Engineer / Bit Wrangler, Pinterest
SRECon 2016 - 07 April 2016

Agenda

My god, it's full of cats!

- Introductions
  - Who Am I, Why Am I Here?
  - Pinterest Infrastructure
- Operational Materialism
- The Rise of Utility Computing
- Operational Buddhism
  - Four Noble Truths
  - The Pinterest Way
  - A PaaS Not Taken
- Q&A, Credits



Who Am I, Why Am I Here?

Turning technical skill into cat food since 1996

Who am I?

- Database Engineer at Pinterest (January 2015)
  - One of two people solely responsible for hundreds of TB of MySQL data
  - Also loosely affiliated with HBase and Core SRE teams
- Previously: Percona, Sun, assorted random small companies
- Jack of many trades, master of some

Why am I here?

- Interested in almost EVERYTHING (not just tech)
- SRE is a very cross-functional discipline



Pinterest Infrastructure

The view from ceiling cat's perspective

- Pinterest is 100% hosted in the AWS cloud.
- Petabytes of data spread across MySQL, HBase, S3, Redis, etc.
- Tens of thousands of servers running at any given time
- Hundreds of unique services interacting with each other
- We make heavy use of some AWS offerings:
  - EC2
  - S3
  - Route53
  - Redshift
- Others, not so much (or at all):
  - RDS
  - CloudFront
  - ElastiCache
  - ElasticBeanstalk



A Trip Back In Time: Operational Materialism

Or, how did we ever manage without "the cloud" ?

Consider the world before the rise of AWS, Google Cloud, Microsoft Azure.

Need computing power or an Internet presence? Not many options:

- Use a managed-services provider.
- Build it yourself.

In the end, these are basically the same.



Operational Materialism

It's still servers all the way down.



Someone still has to deal with the vendors, buy the hardware, rack it, stack it, configure it, and make sure it all stays up and functional within agreed-upon SLAs.



Operational Materialism

The Four Sorrows

1. **Individual servers matter.**
2. **Failure is expensive, so it must be prevented.**
3. **Capacity planning can make or break you.**
4. **Sometimes your destiny is still outside your control.**

Operational Materialism

#1: Individual servers matter.

A server dies… now what?

- Hot/warm stand-by
- Spare parts / DIY
- Roll the trucks!

- Did you remember to buy the extended warranty / gold service plan?



Operational Materialism

#2: Failure is expensive.

Server XYZ is down → the site is down.

- What is this costing you in terms of lost business? (The Lamborghini Factor)
- Cost of employee time to recover?
- What about the cost to your reputation?

Not a situation we want to be in.



Operational Materialism

#2: Therefore, it must be avoided or prevented.

Upgrade ALL THE THINGS!

- Server grade hardware
- Spare parts, spare servers
- Cluster and scale up and out
- Multiple network paths
- Redundant generators
- Backup data centers
- And so on….

- Don’t forget the extra humans!

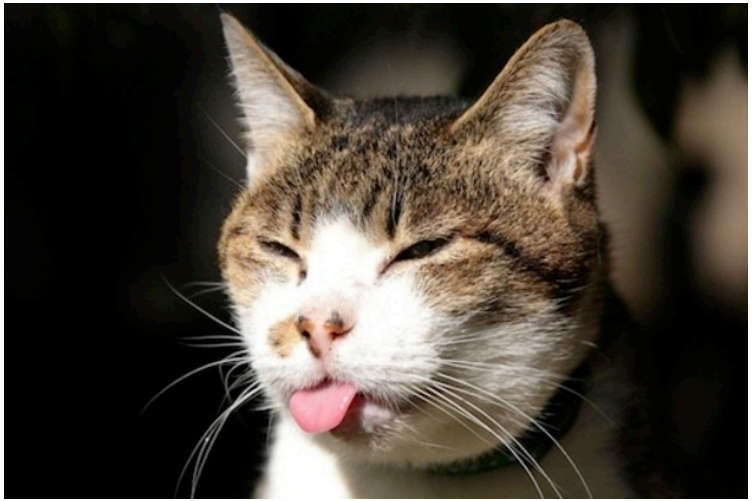


Operational Materialism

Want another nine? Add another zero.

Mo' Problems → Mo' Money → Different Problems

- Failure prevention infrastructure: complexity increases
- Server grade hardware: cost increases
- More humans: every type of pain imaginable increases



Efficiency cat is not pleased with this situation.

Operational Materialism

#3: Capacity planning can make or break you.



Got capacity?

- Not enough → performance sucks
- Too much → wasted resources

The problem of TIME ….

Operational Materialism

#4: Sometimes your destiny is still outside your control.



Even the best capacity planning models can break down due to unforeseen circumstances.

- Natural disasters
- Supply chain disruptions
- Legal conflicts
- The “Slashdot Effect”

Operational Materialism

All good things must come to an end?

- Not all doom and gloom – stuff was built, services were operated.
- Organizations managed their infrastructure this way for years.
- Many still do.

- But sometimes the landscape changes in a fundamental way.

The Rise of Utility Computing

Forecast calls for clouds.







The Rise of Utility Computing

Free your mind, and your servers will follow.

Some promises[1] of utility computing:

- Unlimited, on-demand capacity
- No massive up-front capital expenditures
- Democratization of computing technology
- Focus on building products, not running servers
- Experimentation is easy; failure inexpensive

[1] Some of these promises are more easily kept than others.

The Rise of Utility Computing

TANSTAAFL

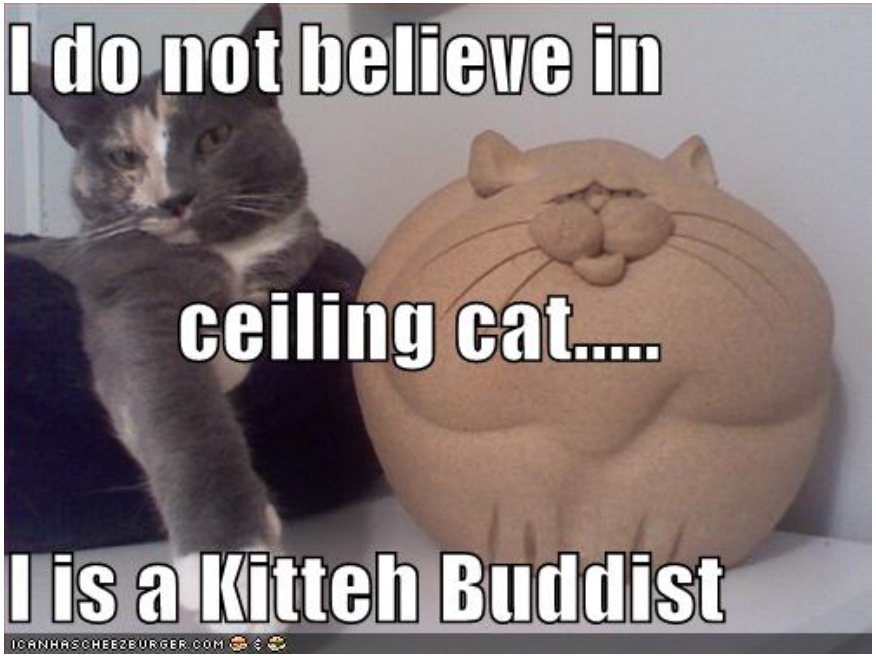
But there are tradeoffs….

- Reduced architectural flexibility
- Black box infrastructure
- Unpredictable performance
- Strong potential for vendor lock-in
- New challenges to cost containment

Operational Materialism doesn't work here. We need a new mindset.

Operational Buddhism

No servers. No attachment. No suffering. No problem.



Operational Buddhism

Four Noble Truths

1. **Cloud servers can, and do, fail at any time, for any reason.**

2. **Trying to prevent this server failure is an endless source of suffering for SREs and DBAs alike.**

3. **Accepting the impermanence of our servers, we should design systems that are failure-resilient, not failure-resistant.**

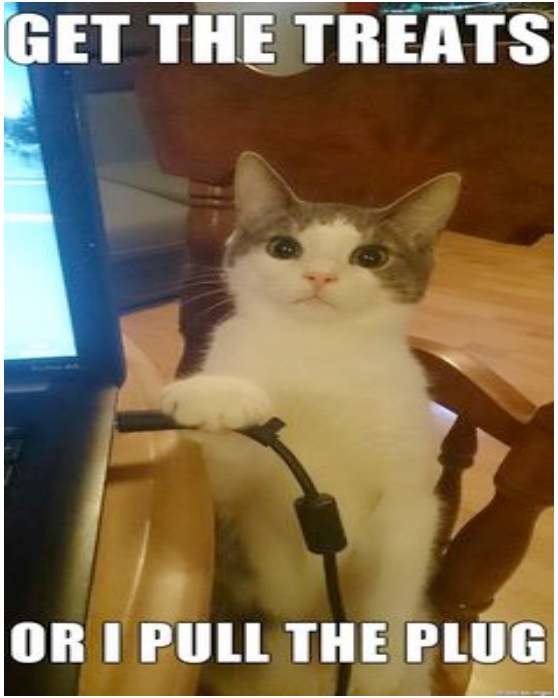
4. **We can break the cycle of suffering and create a better experience for end users, internal customers, and colleagues.**

Operational Buddhism

#1: Failure happens.

Cloud-based servers can fail at any time, for any reason.

- Underlying physical hardware problems
- Oversubscription / “noisy neighbors”
- Hypervisor bugs
- Cascading failures from elsewhere in the cloud



Operational Buddhism

#2: Attachment to servers leads to suffering.

Trying to prevent individual server failure is an unending source of suffering for SREs and DBAs alike.



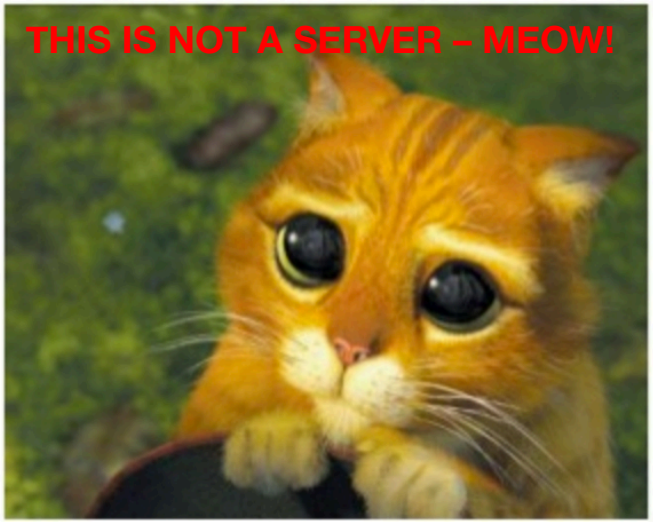
No control over physical infrastructure.
No visibility into physical infrastructure.
No guarantees are possible.

VMs fail at a much higher rate than server-grade bare metal hardware.

Operational Buddhism

#3: Be failure-resilient, not failure-resistant.

Accept the impermanence of individual servers, and in doing so, design systems that are failure-resilient, not failure-resistant.



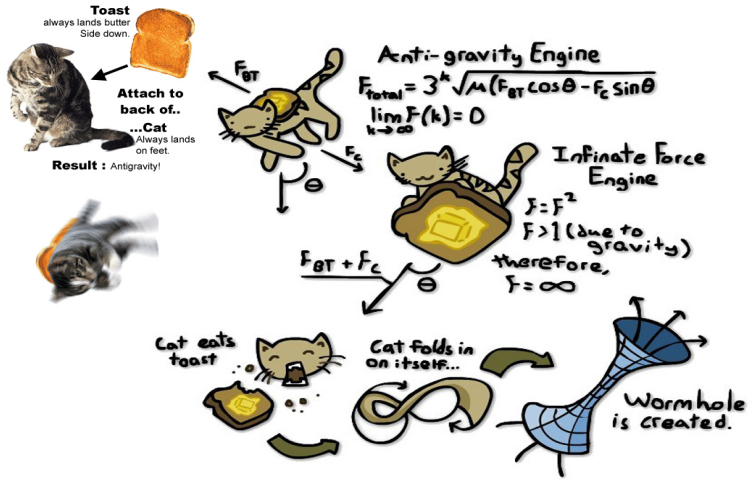


Operational Buddhism

#4: The cycle can be broken.

We can escape the cycle of suffering and create a better experience for our internal customers, end users, and colleagues.

The best infrastructure is invisible to those that rely on it or build on top of it; things JUST WORK.

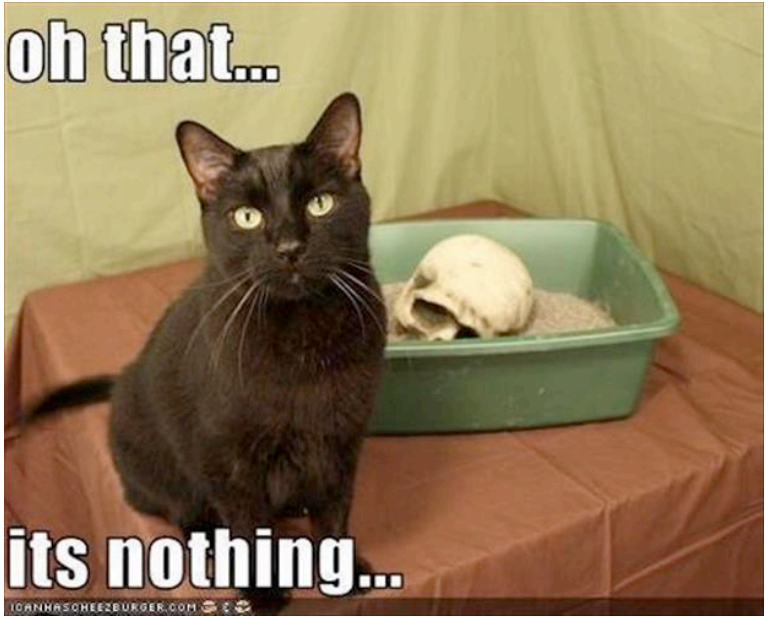


Operational Buddhism

The Pinterest Way

Servers can die at any time for any reason.

- Automated replacement
  - AWS Auto-Scaling Groups
  - Teletraan[1]: Deployment and auto-scaling platform
  - Morpheus[2]: Automated remediation framework
  - Destroy all humans!
- Configuration management tools
  - We are a Puppet shop.
  - You should use whatever works for you.



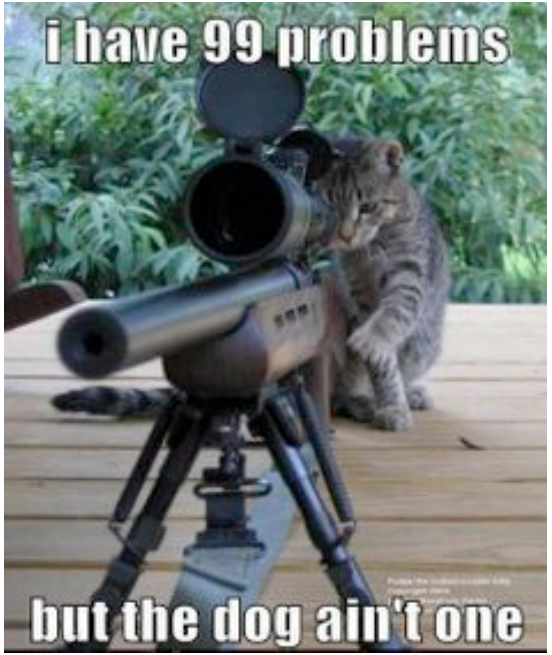
[1] https://github.com/pinterest/teletraan
[2] Not open-sourced… yet. Later this year, perhaps.

Operational Buddhism

The Pinterest Way

Trying to prevent server failure leads only to suffering.

- Don’t do it.
- Don’t even try.
- Shoot them in the head and move on.
- It’s not always necessary (or even possible) to know why a server went AWOL.



Operational Buddhism

The Pinterest Way

Design systems that are failure-resilient; avoid operational anti-patterns.

- Retry logic with back-off can be useful.
- Hardcoded hostnames are the devil.
- KingPin[1, 2]:
  - ZooKeeper-based service discovery and runtime configuration management tools
  - Convergence across ~20K hosts in under 10sec



[1] https://engineering.pinterest.com/blog/open-sourcing-kingpin-building-blocks-scaling-pinterest
[2] https://github.com/pinterest/kingpin

Operational Buddhism

The Pinterest Way

Break the cycle of suffering; create a better experience for all involved.

- Solid infrastructure is just the beginning.
- Automate yourself out of shit work;
  free up cycles for more interesting challenges.
- Developer buy-in is critical all the way up the stack.

Servers may come and go, but uptime is forever.



Operational Buddhism

A PaaS Not Taken

- Infrastructure-as-a-Service (IaaS) vs. Platform-as-a-Service (PaaS)
  - No right or wrong answer – both are manifestations of Operational Buddhism
  - At Pinterest, we lean heavily towards IaaS.
    - Just give us servers; we'll handle the rest.
    - We want the flexibility & control; we have the resources to manage it.
  - PaaS shifts more of the burden to the provider.
    - Additional abstraction comes with additional costs & restrictions.
    - Being tied too heavily to one vendor is always dangerous.

Questions? Answers! Credits!

email: esouhrada@pinterest.com | twitter: @denshikarasu | pinterest engineering blog: https://engineering.pinterest.com



Cat memes are from all over the Internet, primarily:

- *icanhazcheezburger.com*
- *lolcatz.com*
- *http.cat*

We are hiring!

https://careers.pinterest.com

But most importantly…

As much as I'd like to say I thought of it first, the term "Operational Buddhism" was coined by one of my Pinterest colleagues, Danilo Stefanovic.