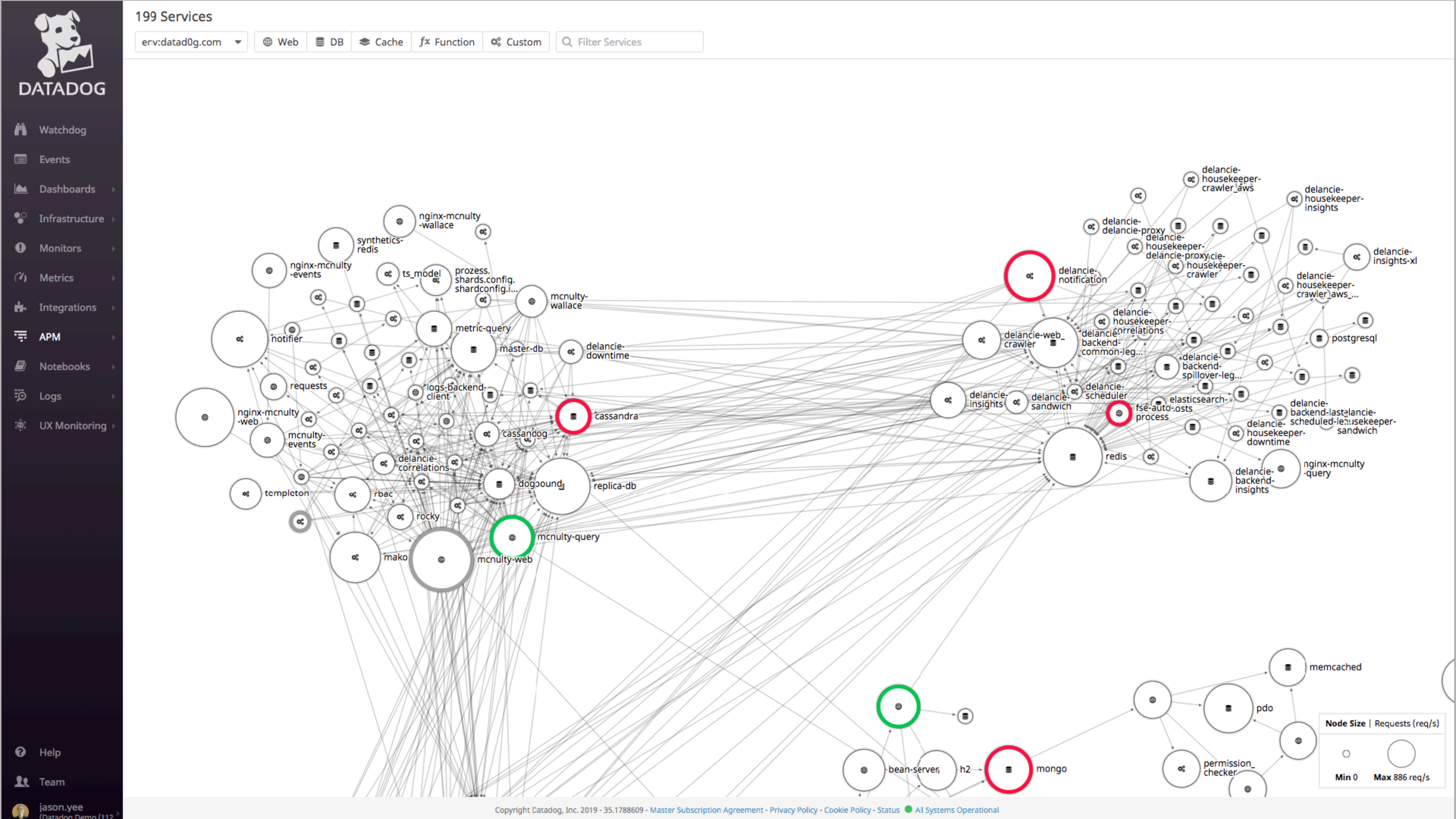
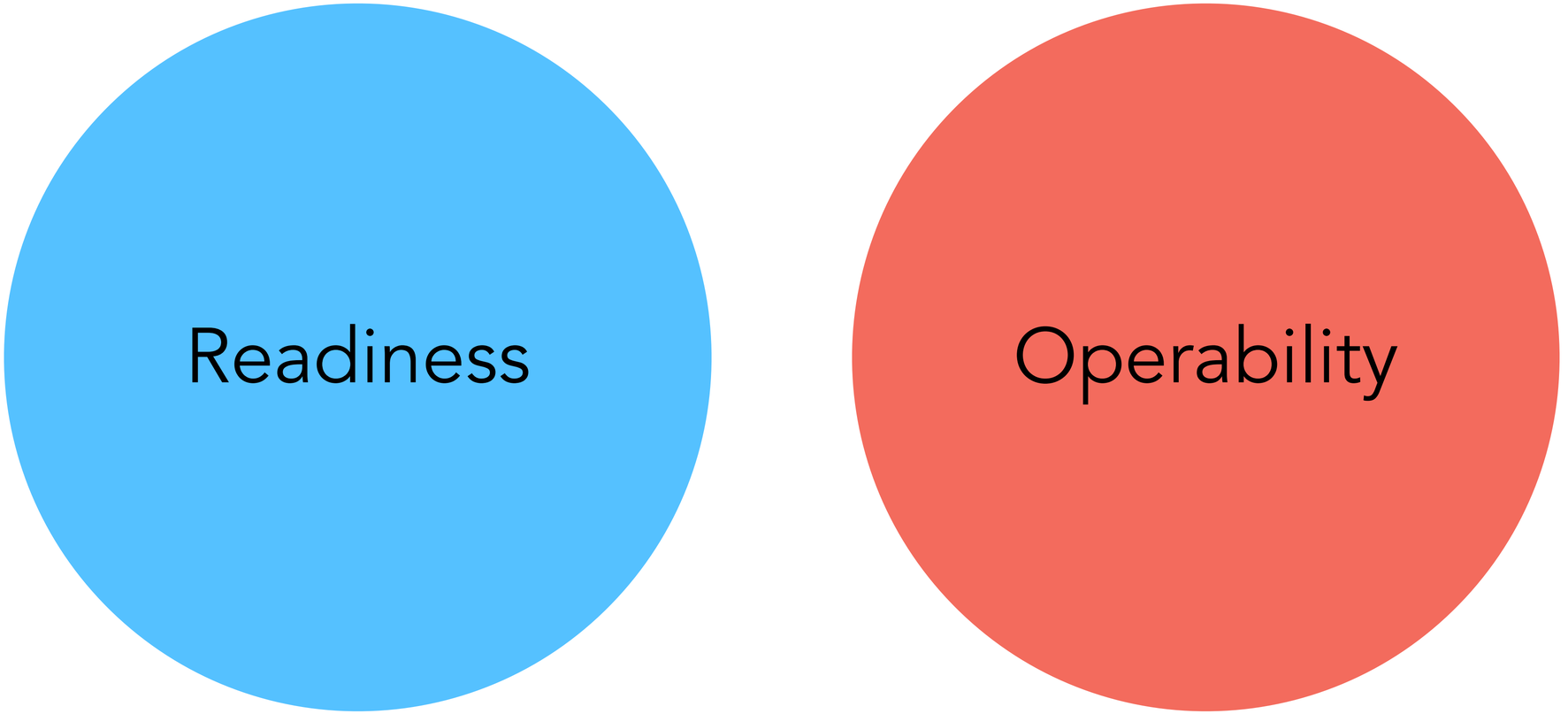
CHAOS!

Breaking your systems to make them unbreakable


DATADOG
Watchdog
Events
Dashboards
Infrastructure
Monitors
Metrics
Integrations
APM
Notebooks
Logs
UX Monitoring
Help
Team
jason.yee
199 Services
erv:datad0g.com
Web
DB
Cache
Function
Custom
Filter Services
nginx-mcnulty-wallace
synthetics-redis
nginx-mcnulty-events
ts_model
prozess.shards.config.shardconfig.i...
mcnulty-wallace
notifier
metric-query
master-db
delancie-downtime
requests
logs-backend-client
nginx-mcnulty-web
mcnulty-events
cassandra
cassandog
delancie-correlations
templeton
rbac
dogpound
replica-db
rocky
mako
mcnulty-web
mcnulty-query
delancie-housekeeper-crawler_aws
delancie-housekeeper-insights
delancie-delancie-proxy
delancie-housekeeper-delancie-proxy
delancie-housekeeper-crawler
delancie-insights-xl
delancie-notification
delancie-housekeeper-crawler_aws_...
delancie-housekeeper-correlations
delancie-web_crawler
delancie-backend-common-leg...
postgresql
delancie-backend-spillover-leg...
delancie-insights
delancie-sandwich
delancie-scheduler
elasticsearch-osts
fse-auto-process
delancie-backend-last-scheduled-leg...
delancie-housekeeper-sandwich
delancie-housekeeper-downtime
redis
nginx-mcnulty-query
delancie-backend-insights
bean-server
h2
mongo
memcached
pdo
permission_checker
Node Size | Requests (req/s)
Min 0
Max 886 req/s
Copyright Datadog, Inc. 2019 - 35.1788609 - Master Subscription Agreement - Privacy Policy - Cookie Policy - Status
All Systems Operational


Readiness
Operability

Failures in complex systems
require multiple contributing causes,
each necessary but only jointly sufficient

John Allspaw (paraphrazing Richard Cook)
http://j.mp/no-root-cause

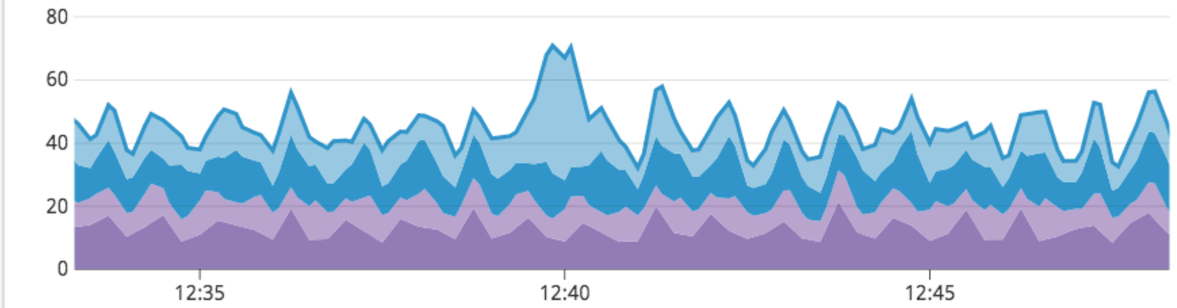
Chaos Demo

Edit Widgets

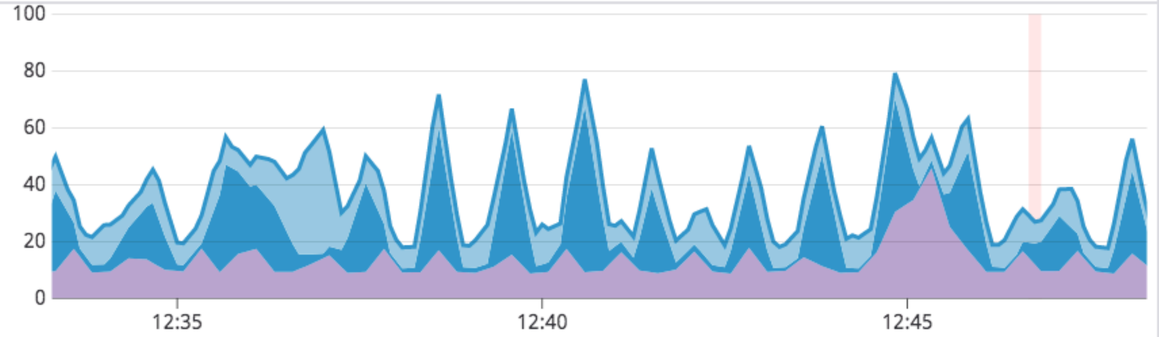
15 m Oct 15, 12:33 pm - Oct 15, 12:48 pm

Search... $Role * $env * $az *

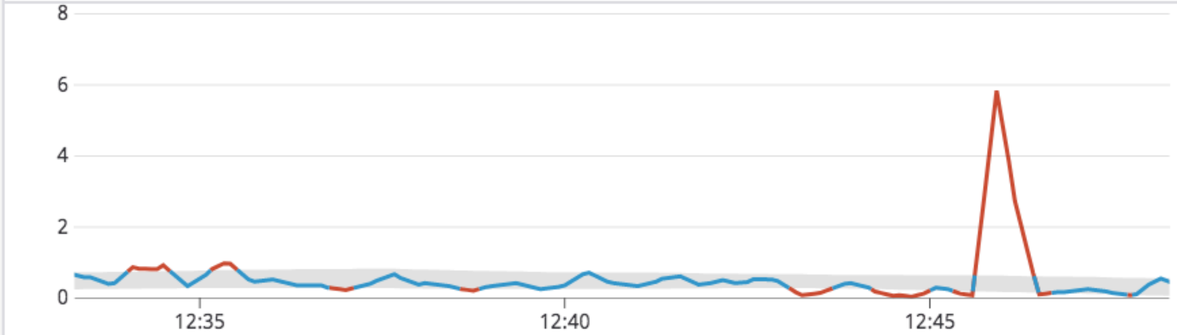
Avg of system.cpu.user over name:cassandra-watchdog by host



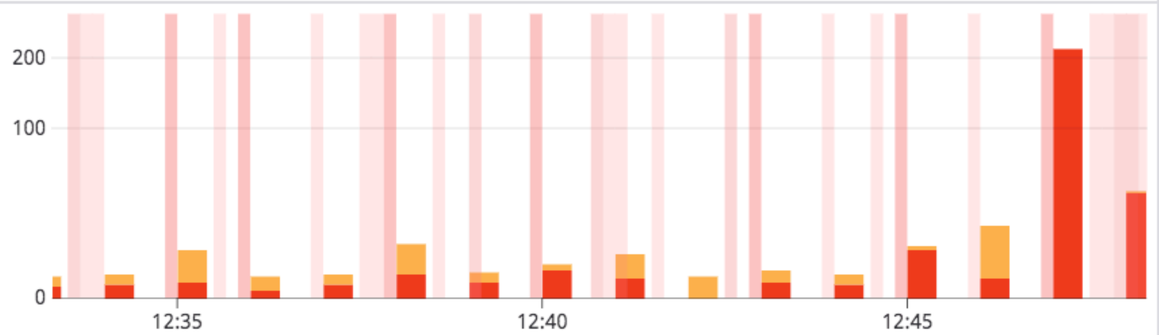
Avg of system.cpu.user over name:trace-cassandra by host



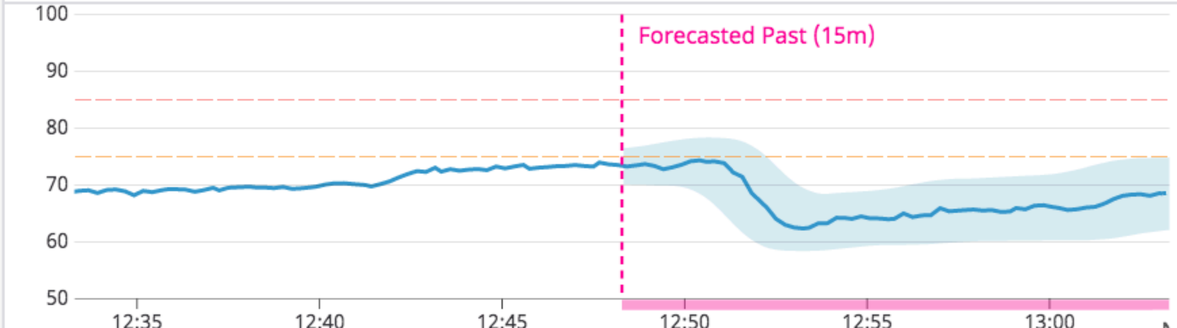
Avg of system.cpu.system over role:zookeeper



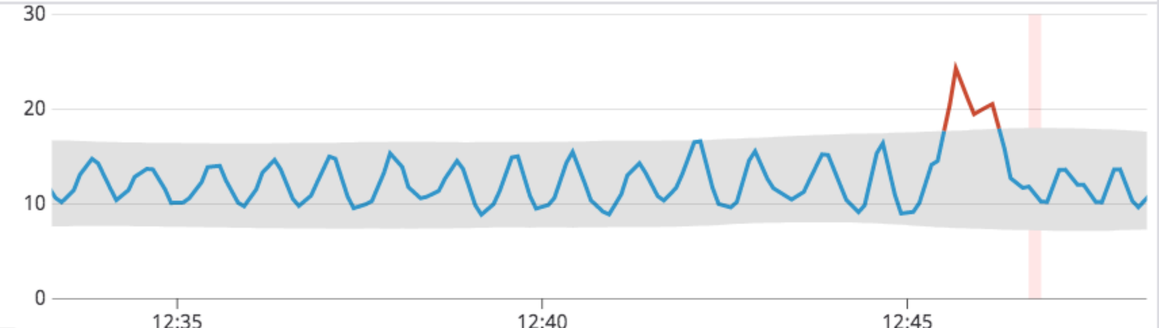
ELB 4xx & 5xx

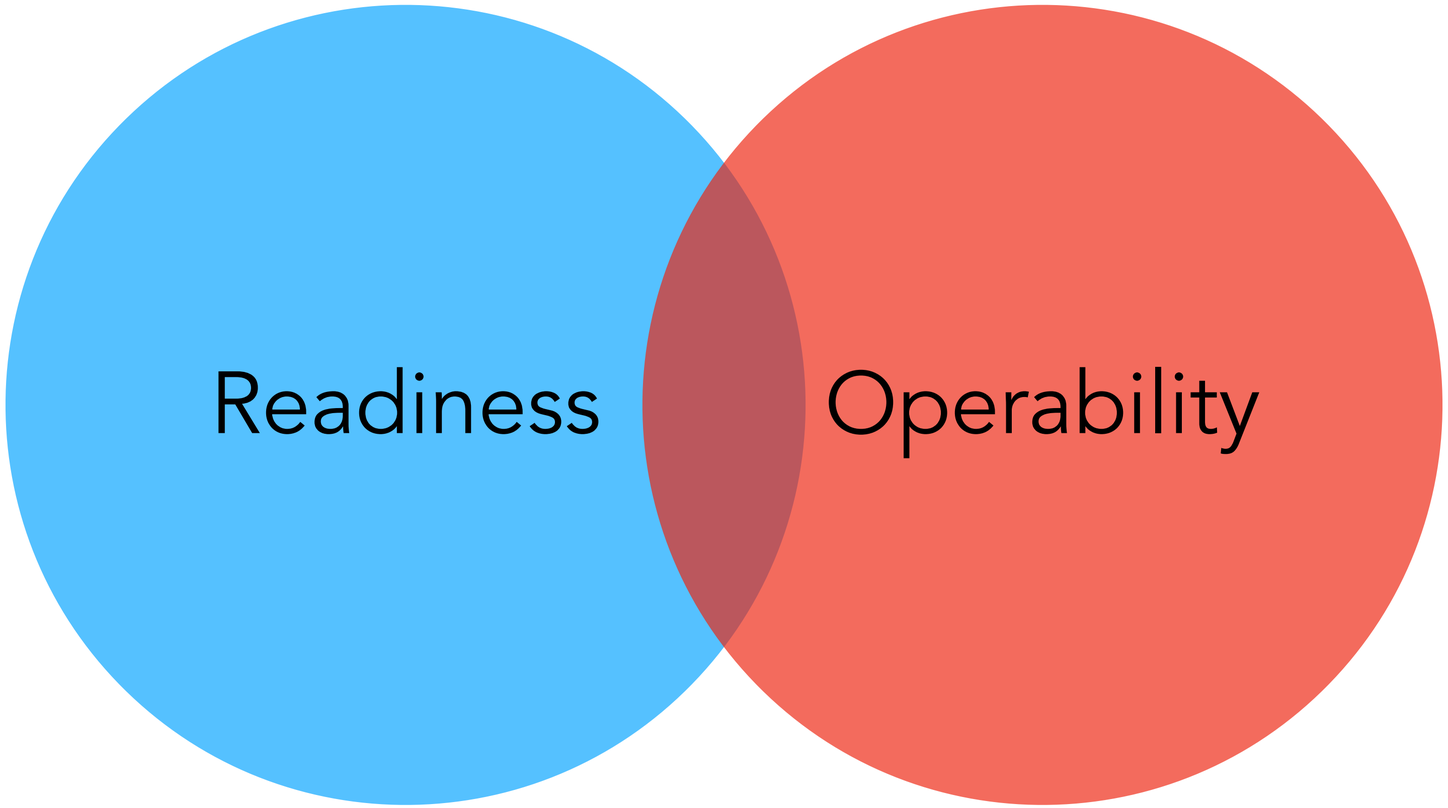


Disk Used



Avg of system.cpu.user over name:cassandra-containers




Readiness
Operability

JASON YEE

- Technical Evangelist
- Conference Organizer
  (DevOpsDays, DeliveryConf)
- Travel Hacker
- Whiskey Hunter
- Pokemon Trainer

Tw: @gitbisect
Em: jyee@datadoghq.com

DELIVERYCONF

Seattle, WA | Jan 21 & 22, 2020

- Learning from today—Shaping tomorrow
- Deep technical talks
- Engaging discussions

Use the code “EVENT” to get 10% off at deliveryconf.com

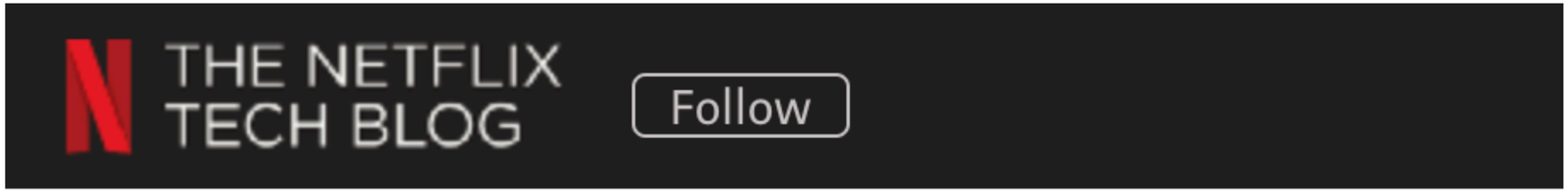
DATADOG

SaaS-based observability platform:

- Metrics
- Traces (APM)
- Logs
- Synthetics

Tw: @datadoghq

We're hiring:
jobs.datadoghq.com



3. The best way to avoid failure is to fail constantly.

http://bit.ly/netflix-5-things

Chaos Monkey

“By running Chaos Monkey in the middle of a business day, in a carefully monitored environment with engineers standing by to address any problems, we can still learn the lessons about the weaknesses of our system, and build automatic recovery mechanisms to deal with them.

So next time an instance fails at 3 am on a Sunday, we won’t even notice.”

–Netflix Technology Blog, 2011
http://bit.ly/netflix-chaos

“By running Chaos Monkey in the *middle of a business day*, in a carefully monitored environment with engineers standing by to address any problems, we can still learn the lessons about the weaknesses of our system, and build automatic recovery mechanisms to deal with them.

So next time an instance fails at 3 am on a Sunday, we won’t even notice.”

–Netflix Technology Blog, 2011
http://bit.ly/netflix-chaos

“By running Chaos Monkey in the *middle of a business day*, in a *carefully monitored environment* with engineers standing by to address any problems, we can still learn the lessons about the weaknesses of our system, and build automatic recovery mechanisms to deal with them.

So next time an instance fails at 3 am on a Sunday, we won’t even notice.”

–Netflix Technology Blog, 2011
http://bit.ly/netflix-chaos

"By running Chaos Monkey in the *middle of a business day*, in a *carefully monitored environment* with *engineers standing by* to address any problems, we can still learn the lessons about the weaknesses of our system, and build automatic recovery mechanisms to deal with them.

So next time an instance fails at 3 am on a Sunday, we won't even notice."

–Netflix Technology Blog, 2011
http://bit.ly/netflix-chaos

"By running Chaos Monkey in the *middle of a business day*, in a *carefully monitored environment* with *engineers standing by* to address any problems, we can still *learn the lessons* about the weaknesses of our system, and build automatic recovery mechanisms to deal with them.

So next time an instance fails at 3 am on a Sunday, we won't even notice."

–Netflix Technology Blog, 2011
http://bit.ly/netflix-chaos

Don’t be a jerk!

Game Days

@gitbisect

90 Minutes

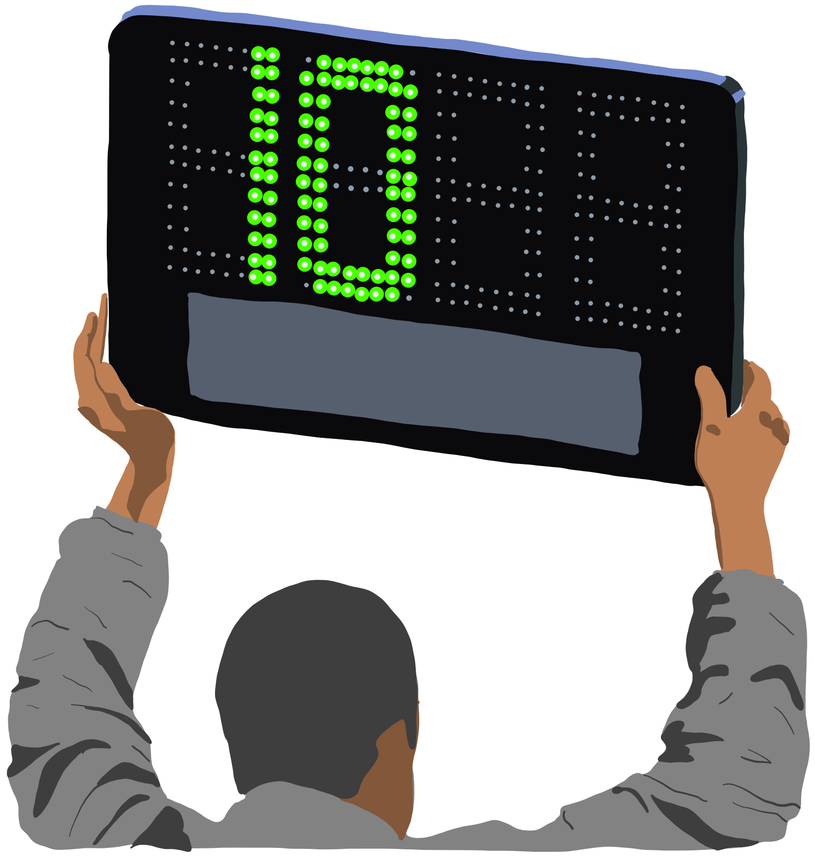
90 Minutes

30 minutes planning.

90 Minutes

30 minutes planning.

50 minutes playing.



90 Minutes

30 minutes planning.

50 minutes playing.

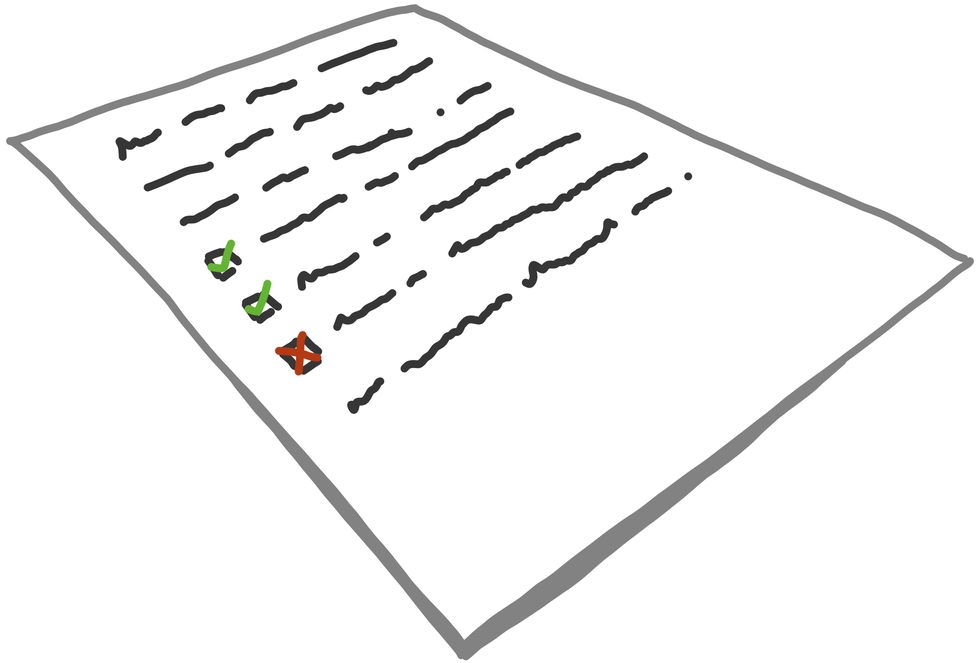
10 minutes reporting.

Before

- Schedule it.

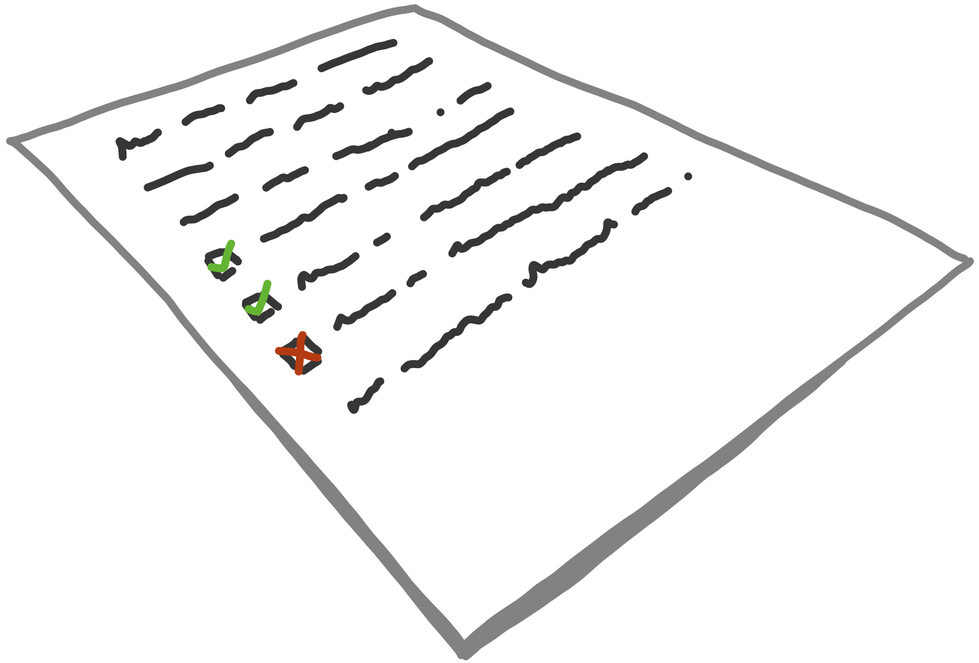
Before

- Schedule it.
- Pick tests. Start easy.



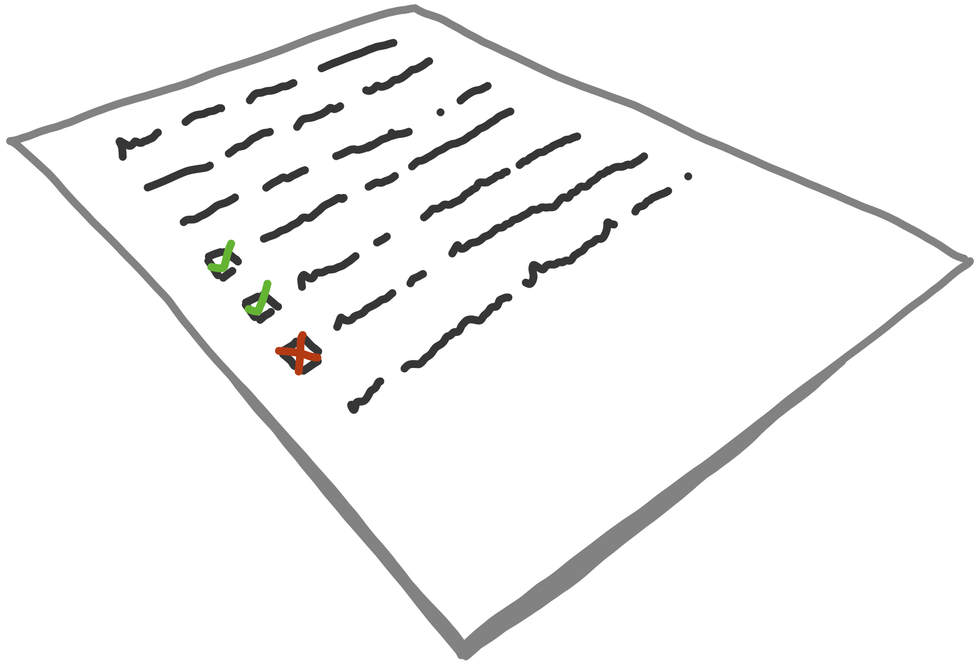
Before

- Schedule it.
- Pick tests. Start easy.
- Write down what you expect to happen.



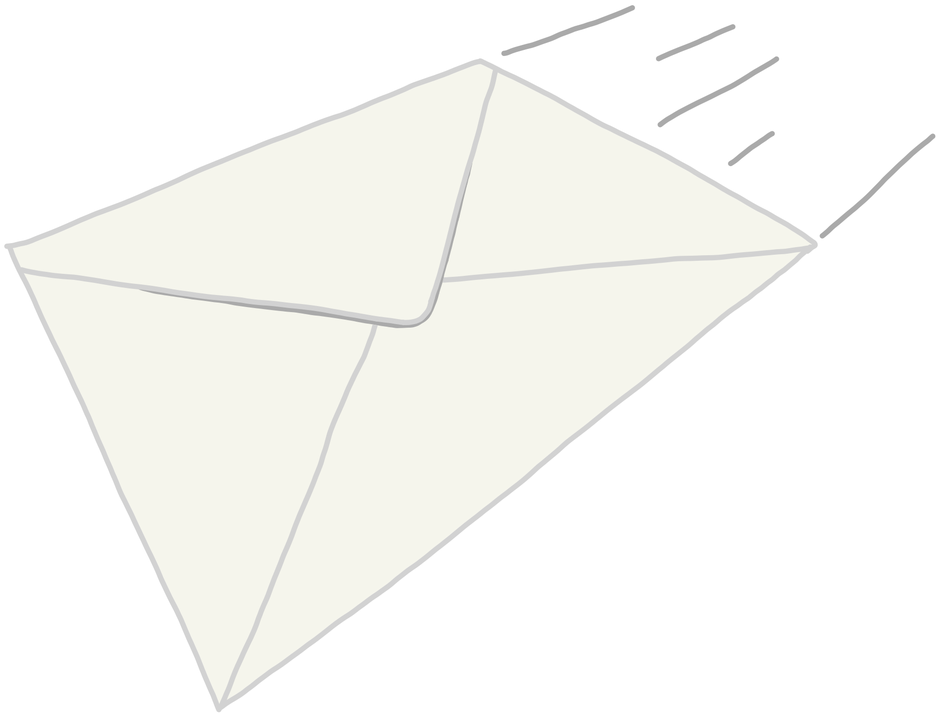
Before

- Schedule it.
- Pick tests. Start easy.
- Write down what you expect to happen.
- Write down your plan if things go wrong.



Before

- Schedule it.
- Pick tests. Start easy.
- Write down what you expect to happen.
- Write down your plan if things go wrong.
- Share your document!



During

- Staging -> Production (off-peak) -> Production (primetime)

During

- Staging -> Production (off-peak) -> Production (primetime)
- Announce start in group chat.

During

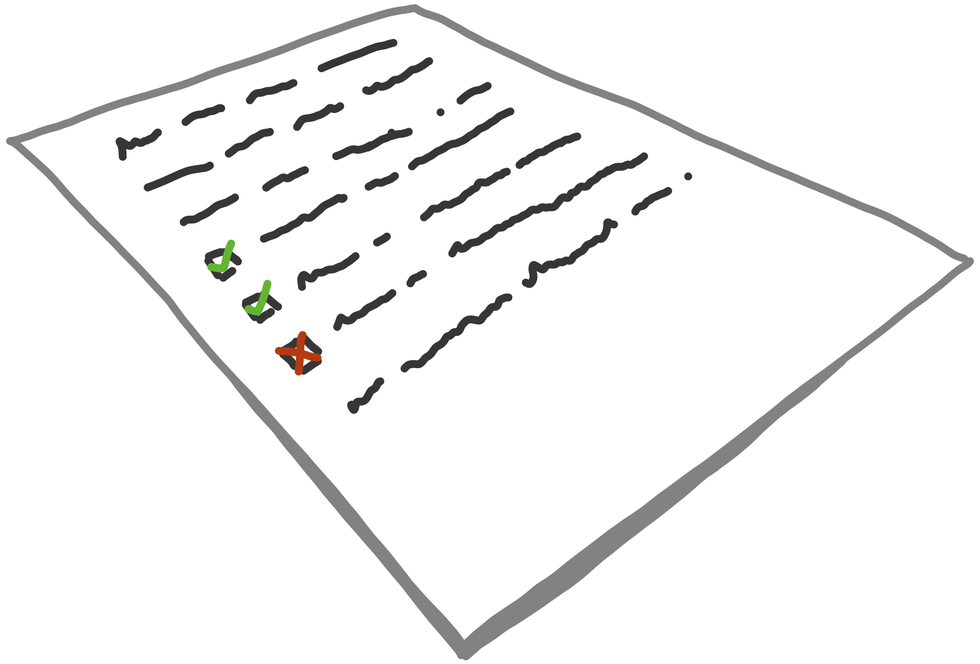
- Staging -> Production (off-peak) -> Production (primetime)
- Announce start in group chat.
- Maintain discussion in group chat.

During

- Staging -> Production (off-peak) -> Production (primetime)
- Announce start in group chat.
- Maintain discussion in group chat.
- Monitor for outages.

During

- Staging -> Production (off-peak) -> Production (primetime)
- Announce start in group chat.
- Maintain discussion in group chat.
- Monitor for outages.
- Run your test *and take notes!*

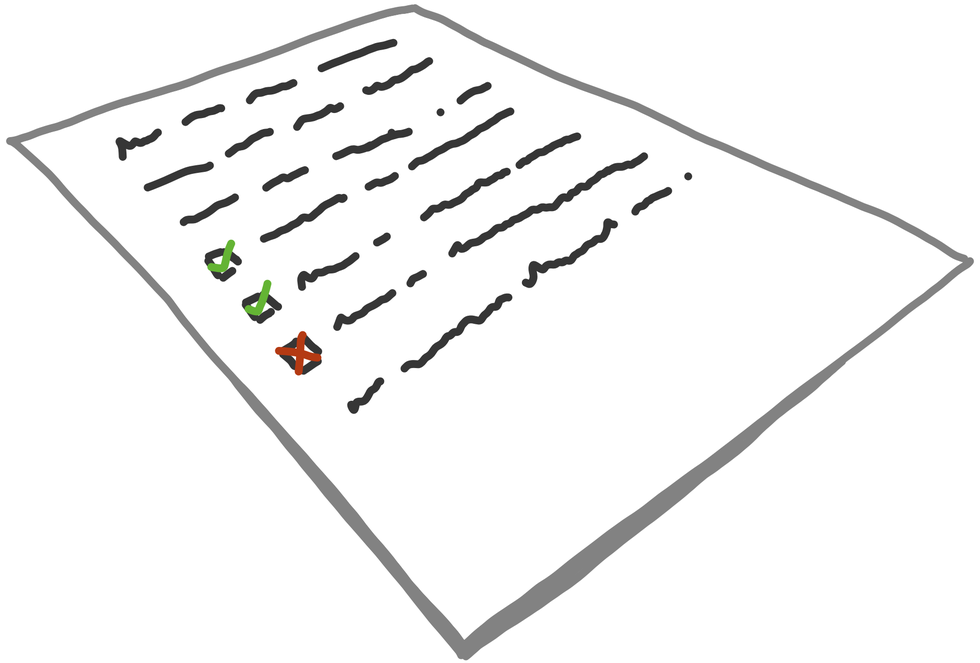


After

- Create cards/tickets to track issues that need work.

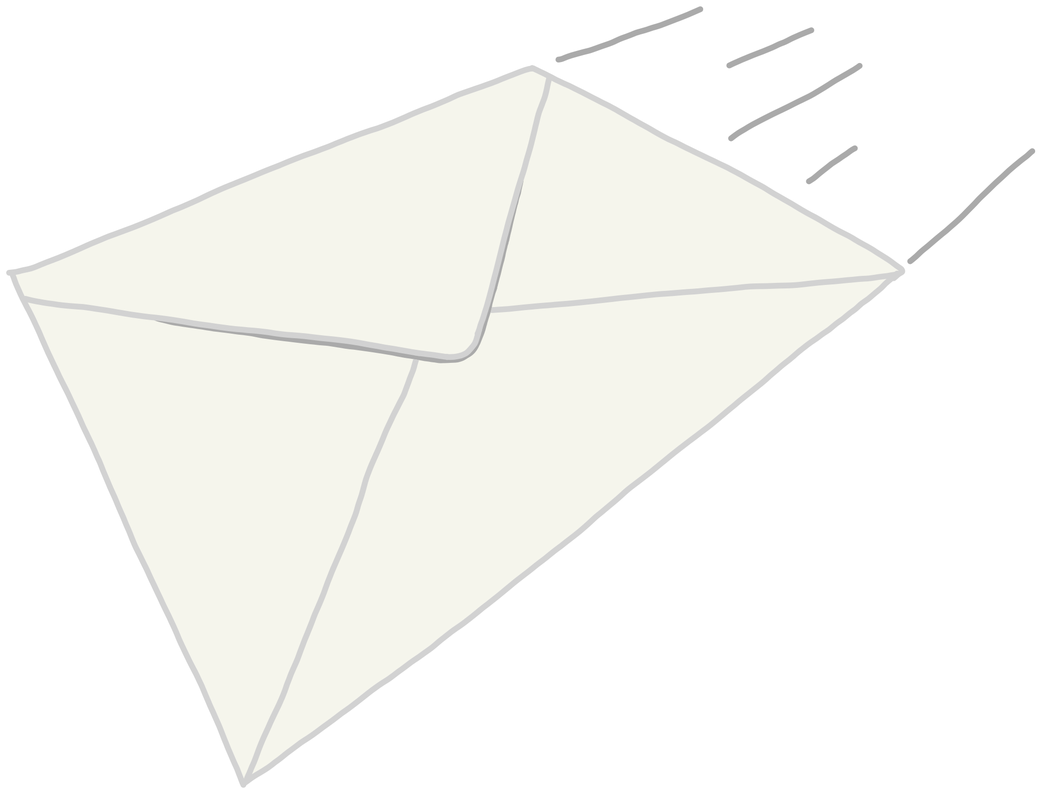
After

- Create cards/tickets to track issues that need work.
- Write a summary & key lessons.



After

- Create cards/tickets to track issues that need work.
- Write a summary & key lessons.
- Email to all of engineering.



After

- Create cards/tickets to track issues that need work.
- Write a summary & key lessons.
- Email to all of engineering.
- Celebrate!

Game Levels

0. Terminate the service. Block access to 1 dependency.

Game Levels

0. Terminate the service. Block access to 1 dependency.

1. Block access to all dependencies.

Game Levels

0. Terminate the service. Block access to 1 dependency.

1. Block access to all dependencies.

2. Terminate the host.

Game Levels

0. Terminate the service. Block access to 1 dependency.

1. Block access to all dependencies.

2. Terminate the host.

3. Degrade the environment.
   Monitor & alert on Latency, Errors, Traffic, & Saturation

Game Levels

0. Terminate the service. Block access to 1 dependency.

1. Block access to all dependencies.

2. Terminate the host.

3. Degrade the environment.

4. Spike traffic.

Game Levels

0. Terminate the service. Block access to 1 dependency.

1. Block access to all dependencies.

2. Terminate the host.

3. Degrade the environment.

4. Spike traffic.

5. Terminate the region/cloud.

Experiment time!

Review

Share!

Plan!

Have fun!

Additional Resources

- systemctl, kill, iptables

Additional Resources

- systemctl, kill, iptables
- Comcast - https://github.com/tylertreat/comcast

Additional Resources

- systemctl, kill, iptables
- Comcast - https://github.com/tylertreat/comcast
- Vegeta - https://github.com/tsenart/vegeta

Additional Resources

- systemctl, kill, iptables
- Comcast - https://github.com/tylertreat/comcast
- Vegeta - https://github.com/tsenart/vegeta
- stress-ng

Additional Resources

- Gremlin - https://www.gremlin.com/
- ChaosToolkit - https://chaostoolkit.org/

Questions? sli.do!

jason.yee@datadoghq.com
@gitbisect