

# NETFLIX

## Netflix RaaS: Reliability as a Service

Coburn Watson



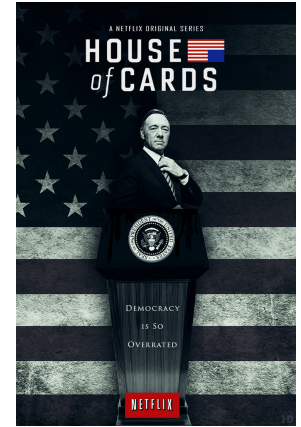
# @coburnw

- Cloud Performance, Reliability (and Capacity) @Netflix
  - Reactive and Proactive consultation
  - Build innovative performance analysis tooling
    - [Microscope on Microservices Netflix Tech Blog](#)
  - Drive operational best practice adoption
  - Optimize usage of AWS cloud
  - Optimize Netflix Linux BaseAMI and middleware configurations
  - Production On-Call resources as needed
- Hiring!
- Handing out Simian Army stickers after the talk

# Netflix, Inc.

- Early to the cloud
  - tens of thousands of AWS instances, 3 regions
  - autoscale ~3k instances a day (per region)
  - failover traffic between regions
- [Freedom & Responsibility Culture](#)
- 34% of US Internet Traffic at Night\*
  - Build and Maintain own CDN
- > 2B hours of streaming per month
  - 55M+ subscribers
  - Strong focus on originals

\* 6% of traffic is upload



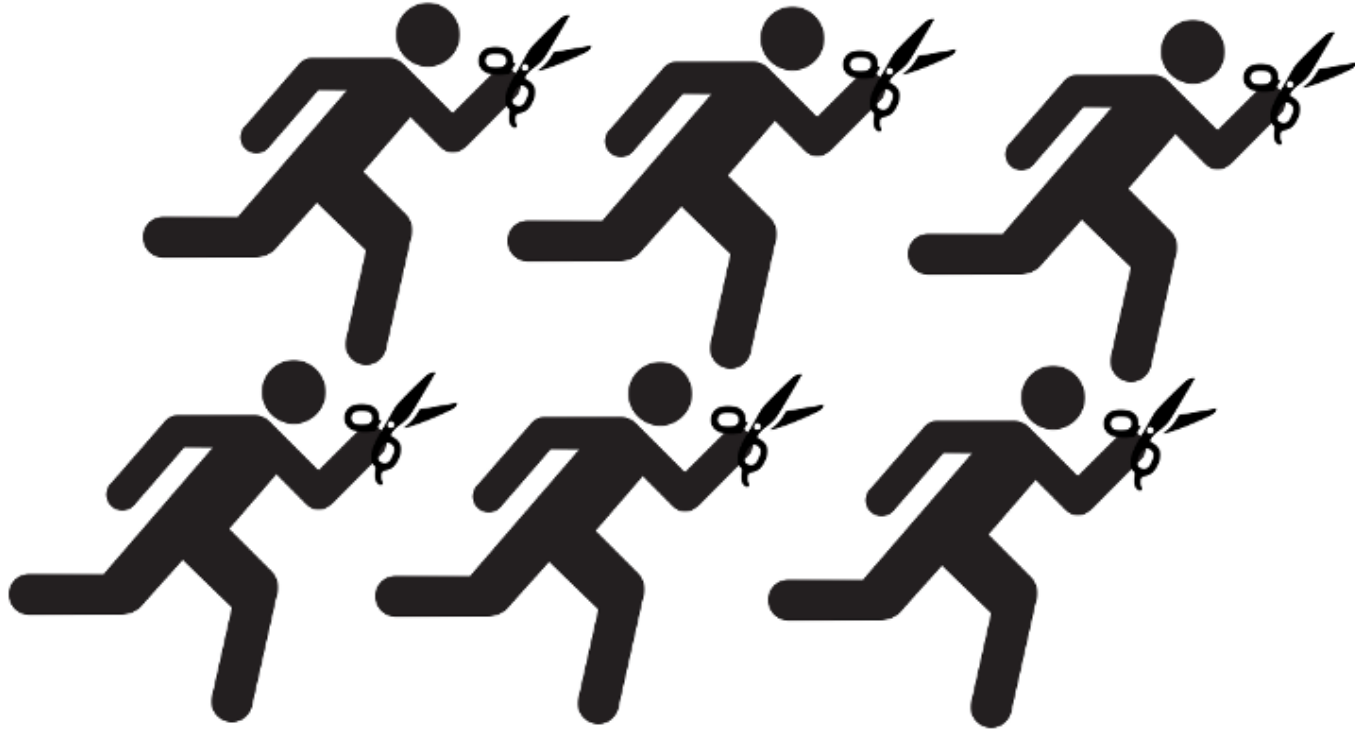
**NETFLIX**

# The Challenge

- Finding the right balance
  - Dev vs. Ops responsibilities for a team
  - Shared versus distributed model
- Further Complications
  - large microservice-based architecture (many moving parts)
  - numerous, often complex, frameworks used across microservices
  - asymmetric distribution of operational complexity across teams



# *A Primitive DevOps Model*



# Let me just push to Production..

## *(Implicit)* Guidance

- Don't break production!
  - Catch "bad" code early
  - Stagger global pushes
  - Rollback quickly if needed
  - Create and maintain effective alerts
  - Tune circuits (isolate failure)
  - Ensure sufficient capacity
    - but don't use too much...
  - Remove misbehaving nodes



# A high bar...



- Requires broad and deep expertise
- Complicates hiring strategy
- Dilutes operational focus of team
- Introduces manual error
- Exacerbates configuration drift
- Encourages development of operational point solutions
- possibly impact engineer morale

# One Possible Solution



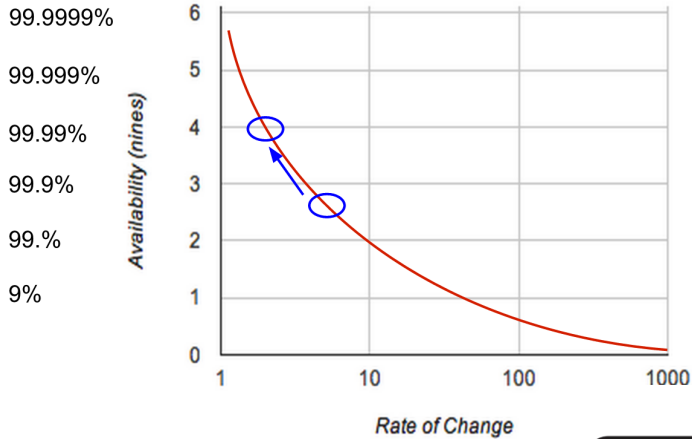
By Frits Ahlefeldt

**NETFLIX**

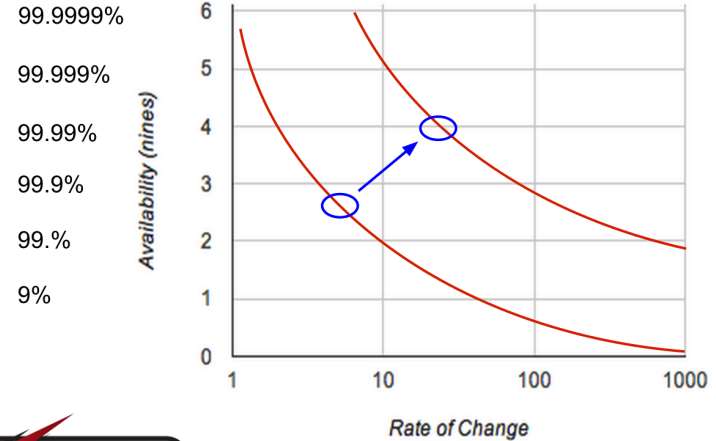


# Operations Engineering

Availability vs. Rate of Change



Availability vs. Rate of Change



NETFLIX

# Operations Engineering

## 5 Primary Teams

- Engineering Tools
- Insight Engineering
- Crisis Response Engineering
- Traffic & Chaos
- Cloud Network Engineering
- Performance and Reliability Engineering



**NETFLIX**

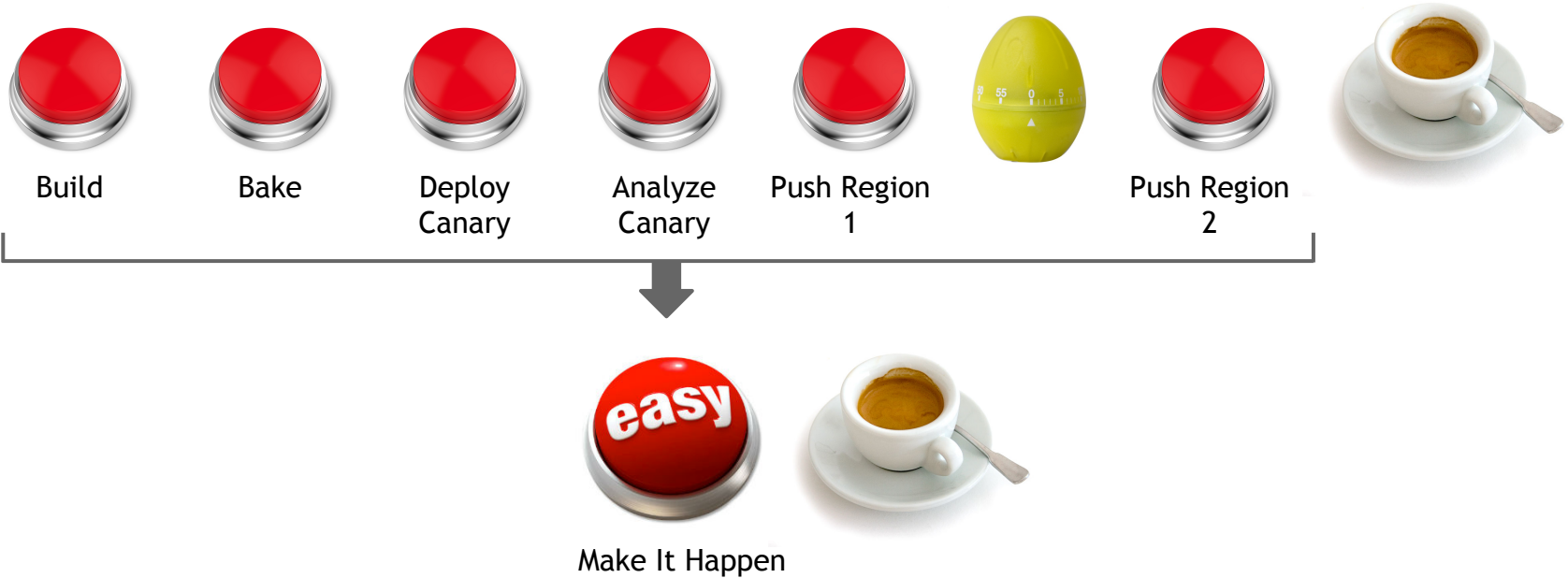
# Strategy

- Lower operational barriers for getting code into production
  - Eliminate undifferentiated heavy lifting
- Create effective feedback loops (context)
  - Performance and reliability
  - Code Quality
  - Efficiency and Cost



**NETFLIX**

# Strategies Applied



**NETFLIX**

# Continuous Integration

- One button staggered global deploy (or scheduled)
- Integrates with Automated Canary Analysis Framework

The screenshot displays a dashboard for pipeline management. At the top, there are navigation tabs: DELIVERY (selected), CLUSTERS, LOAD BALANCERS, SECURITY GROUPS, TASKS, and CONFIG. Below the tabs, there are filters for 'Group by Pipeline' and 'Pipeline Status' with options: Not Started, Running, Completed, Failed, Canceled, and Suspended. A 'Show 5 per group' dropdown is also present. A 'Configure Pipelines' link is visible on the right.

The main content area shows a pipeline named 'Deploy PROD' with a 'RUN NEW EXECUTION' button. Below this, a table lists pipeline runs with columns for Trigger, Status, and task progress bars.

Trigger	Status	BAKE US-EAST-1	DEPLOY US-EAST-1	BAKE US-WEST-2	DEPLOY US-WEST-2	BAKE EU-WEST-1	DEPLOY EU-WEST-1
Manual Start 2015-02-20 11:55:34	COMPLETED 10 minutes	Completed	Completed	Completed	Completed	Completed	Completed
Manual Start 2015-02-19 15:48:40	COMPLETED 9 minutes	Completed	Completed	Completed	Completed	Completed	Completed

# Catch “bad” code early

- Automated Canary Analysis

**Canary Score: 83%** **MARGINAL**

Report Date: Mon Mar 09 2015 04:35:53 GMT-0700 (PDT)  
Region: eu-west-1.prod  
Canary: [REDACTED]  
Type: cluster  
Duration: PT36H  
Version: [REDACTED]

### Metric Analysis

Show  pass  low  high  nodata Filter

Group Name	metrics	Deviation	Score
untagged	64		83% <b>MARGINAL</b>

### Tag: untagged

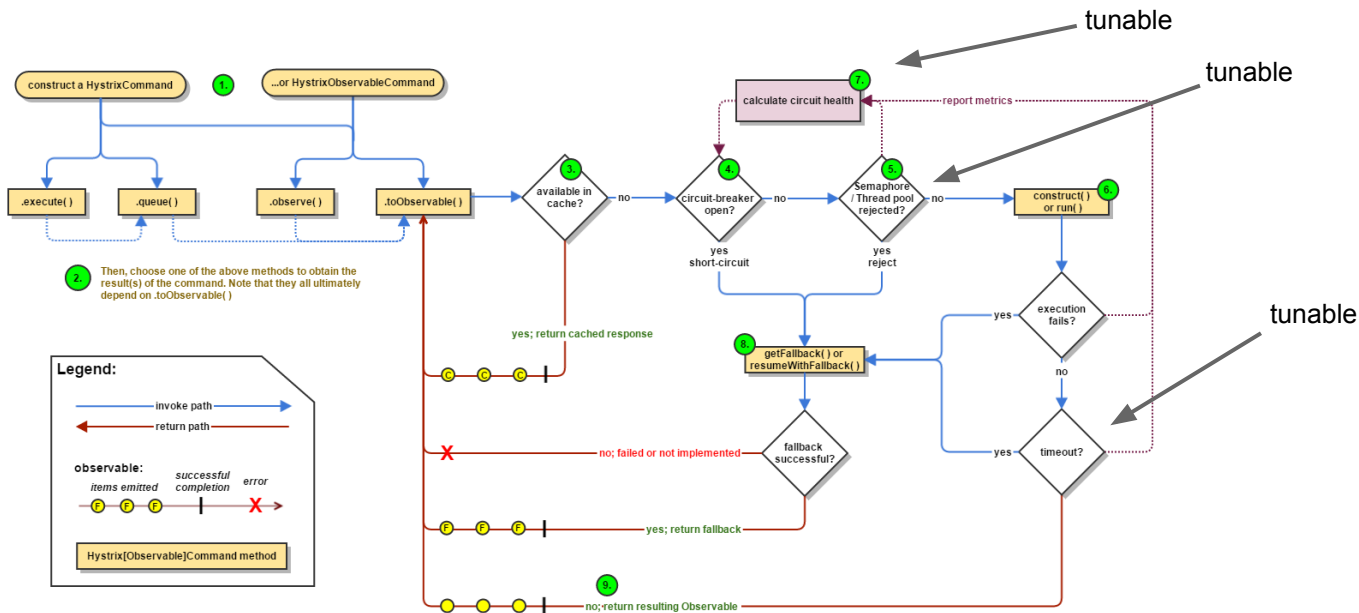
**Score: 83%** **MARGINAL**

### 64 Metrics

Group Name	Deviation	Status
EVCACHE_SUB-CheckedOperationTimeout-RATIO	+15.5%	PASS
MapGetTurboCharacterListServerCount_errorPercentage	0%	PASS
RequestStats_MapGetTopLevelTurboGenresServer_TotalTimeMillis90Percentile	+1.4%	PASS
EVCACHE_CINERS-CheckedOperationTimeout-RATIO	+24.4%	PASS
CMS_Old_Gen_Usage_used	+0.4%	PASS
EVCACHE_RECS_TARGETING-PROMOTION-LatencyGet	+12.8%	HIGH

# Tune Circuits

- Fault isolation frameworks can require significant babysitting
  - Netflix framework is [Hystrix](#)



# Tune Circuits, cont.

- Strategy
  - Consult experts users of Hystrix within Netflix
  - Identify possible models to apply
  - Train selected model against simulation framework
  - Validate model against production workload
  - Automate analysis through RTA framework
  - Configure RTA\* framework to send emails highlighting risk
- 90/10 rule
  - There are always corner cases, don't be discouraged
- Research took about a quarter

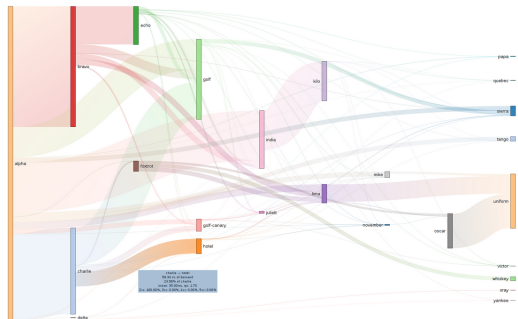


\*RTA is an analytics platform built internally

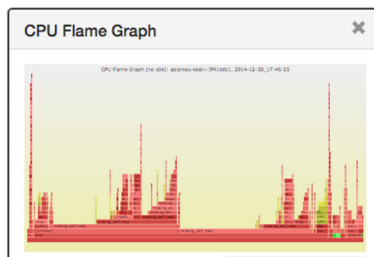
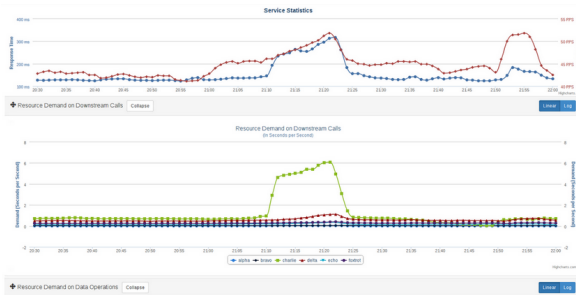


# Improved Observability

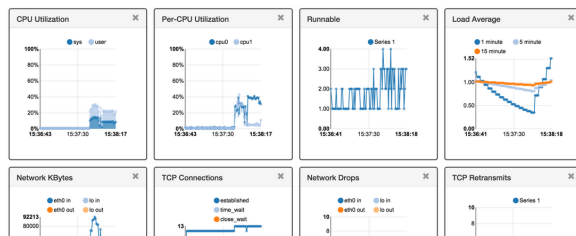
Self-service tooling based on that used by Performance & Reliability Team



(Macro)



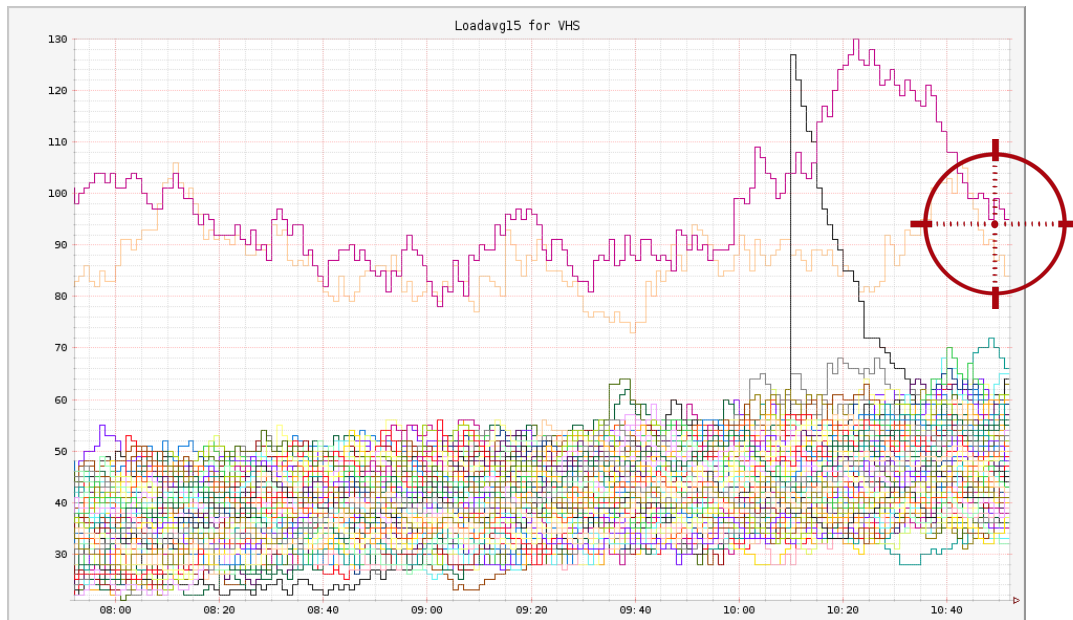
(Micro)



**NETFLIX**

# Eliminate Outliers

- *Two of these instances are not like the others, two of these just don't belong\*\*..*



\*\* paraphrased

# Another helping hand

All the monkeys and gorillas you could want..the Netflix Simian Army



- Automated (unplanned) Terminations
- Security Scans
- Cleanup of unused resources
- Injection of latency and failures
- Terminate an availability zone
- Fail a Region

*...all opt-in by default*

**NETFLIX**

# Efficiency & Cost

Distribute weekly cost report

Capacity/Zones

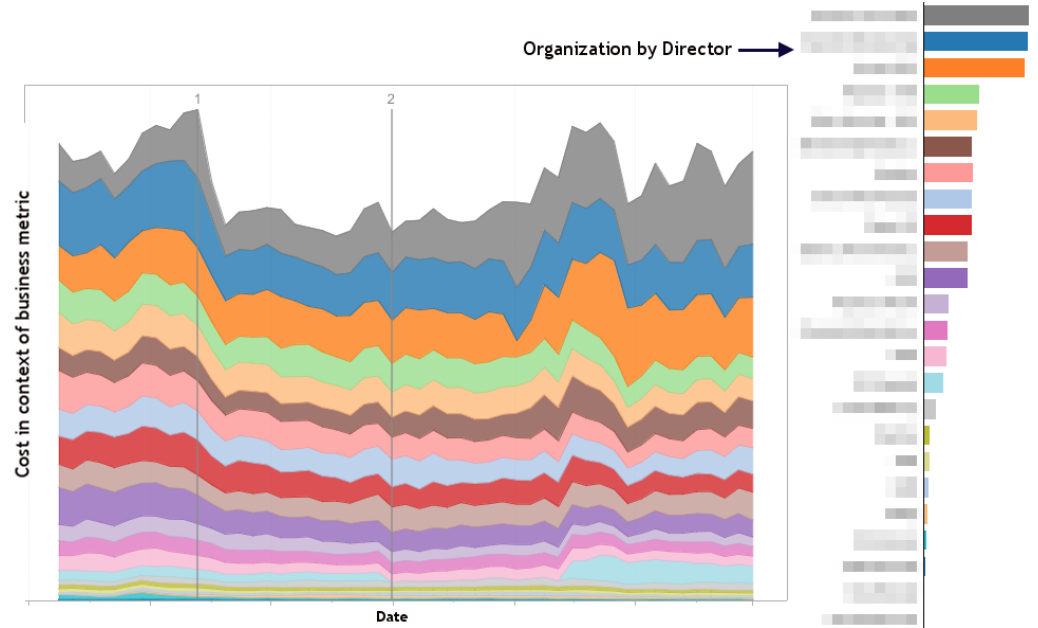
Sets min, max, and desired instance counts to the same value.

To allow true auto-scaling, use the [Advanced Mode](#).

Number of Instances

Availability Zones Automatic Availability Zone Balancing:

Freedom



Responsibility

NETFLIX

# Tool Transitions

- (Point) Solutions built out of necessity
  - broad value for long-term care and feeding
- Example: Scryer
  - Predictive autoscaling engine
  - Developed by Edge Platform..migrated to Performance and Reliability team



**NETFLIX**

# Measuring Success

- Reduction in incidents related to specific operational areas/tasks
- Instrument applications to quantify uptake rates
- Talk to your customers...often; solicit critical feedback
- Engineering teams providing resources for tool development/extension
  - vs building point solution w/in team



**NETFLIX**

# Onwards and upwards



- Evolve operational best practices program
  - Production Ready
  - Prioritize work to align with “critical” services; maximize leverage
- Create and extend tooling
  - Autoscaling RTA module, Java GC tuning, etc.
- Continue to strengthen Reliability Engineering base

# Q&A

- [cwatson@netflix.com](mailto:cwatson@netflix.com), @coburnw
- [Netflix Open Source Portal](#)