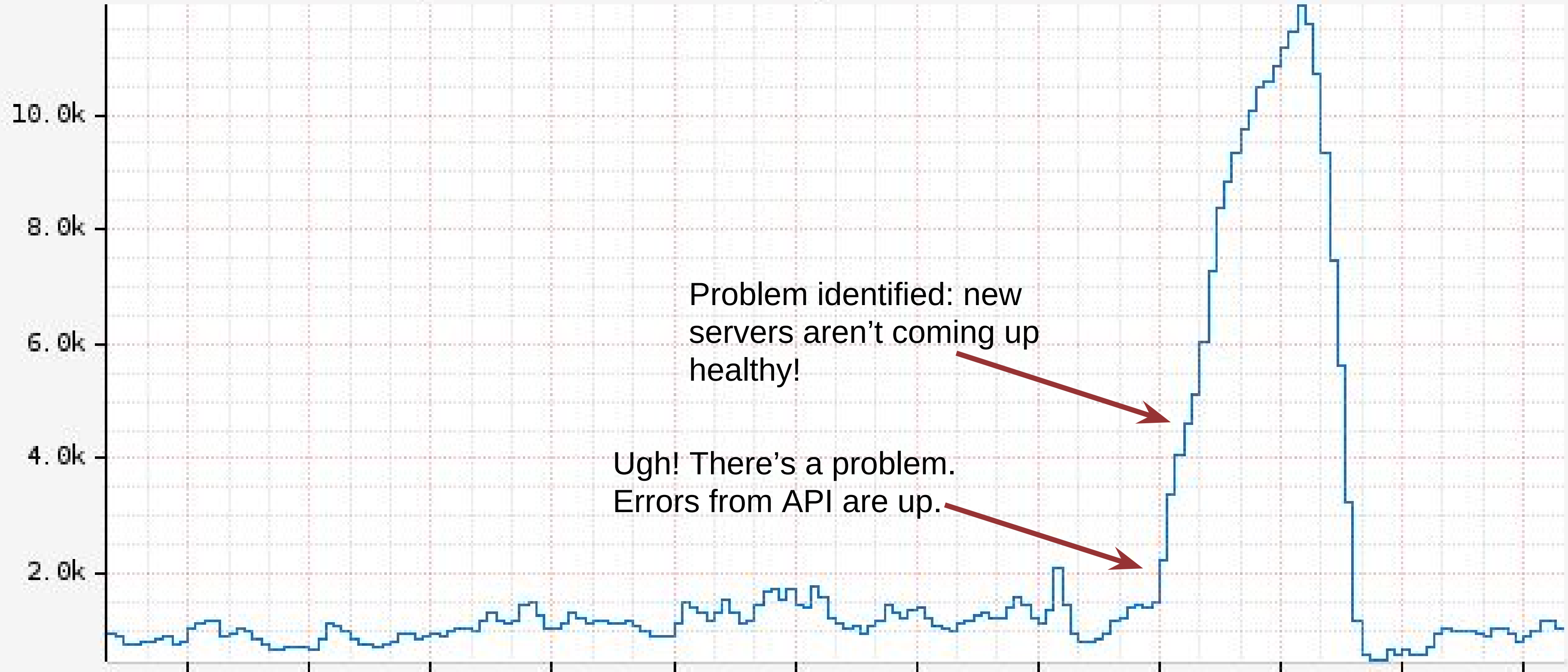
The Netflix API Platform for Server-Side Scripting

Katharina Probst

Engineering Manager, API at Netflix

https://www.linkedin.com/in/katharinaprobst, @probst_kathrin

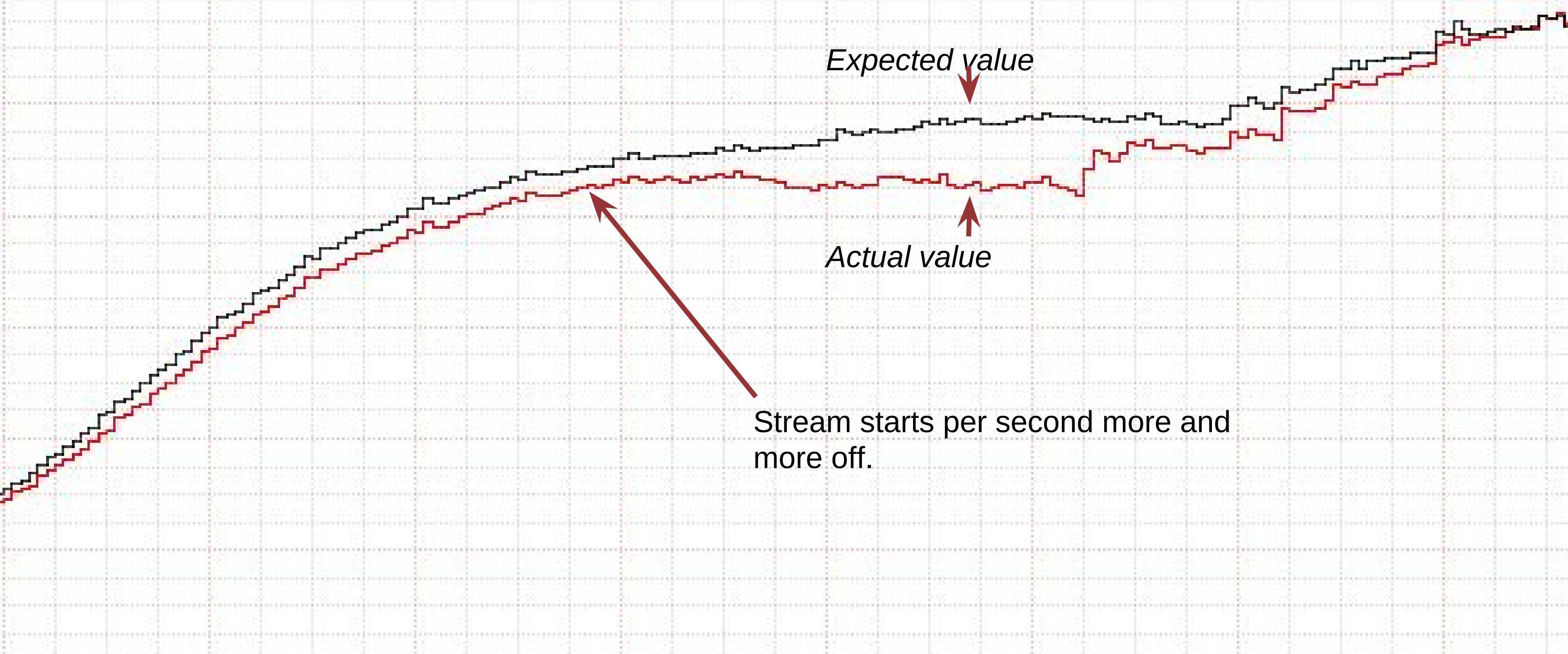
The risks of modifying running production servers



Problem identified: new servers aren’t coming up healthy!

Ugh! There’s a problem. Errors from API are up.

Escalating customer impact

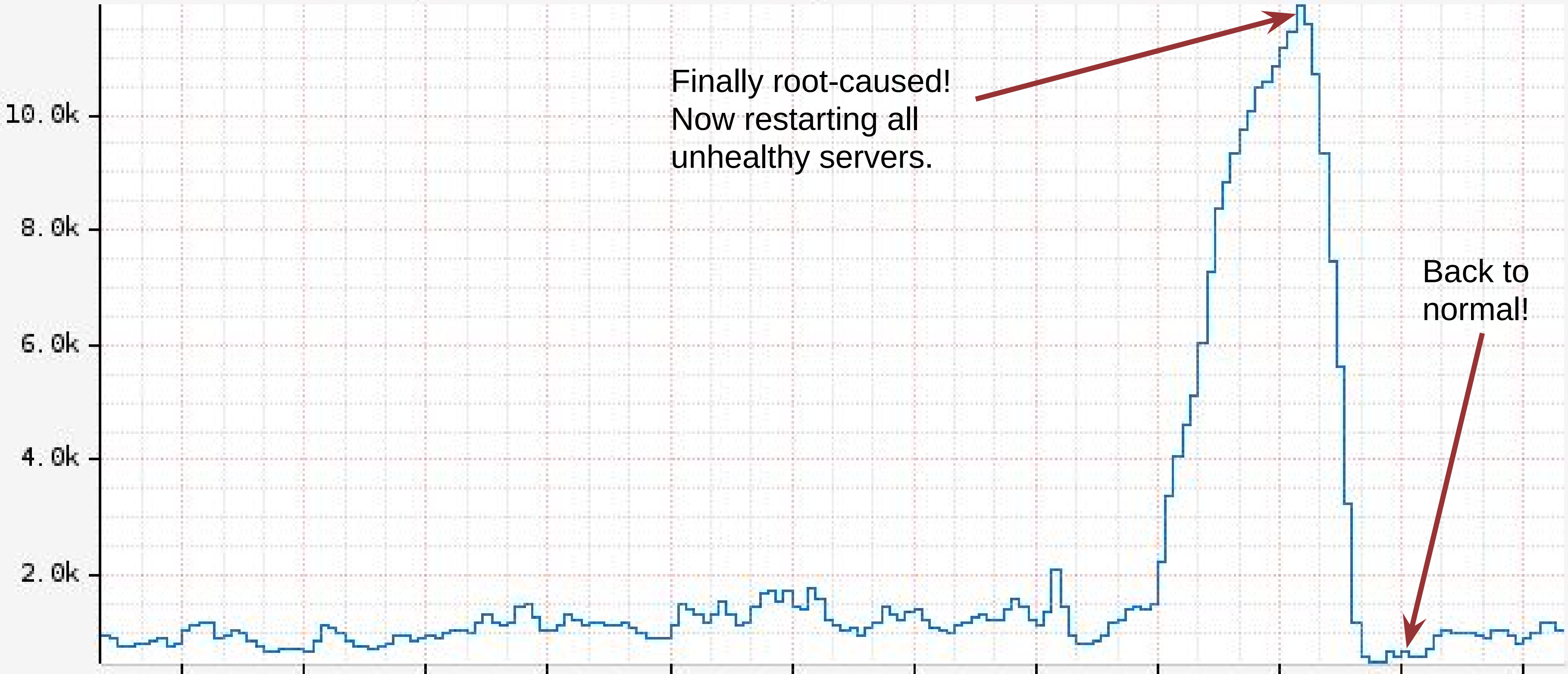


Expected value

Actual value

Stream starts per second more and more off.

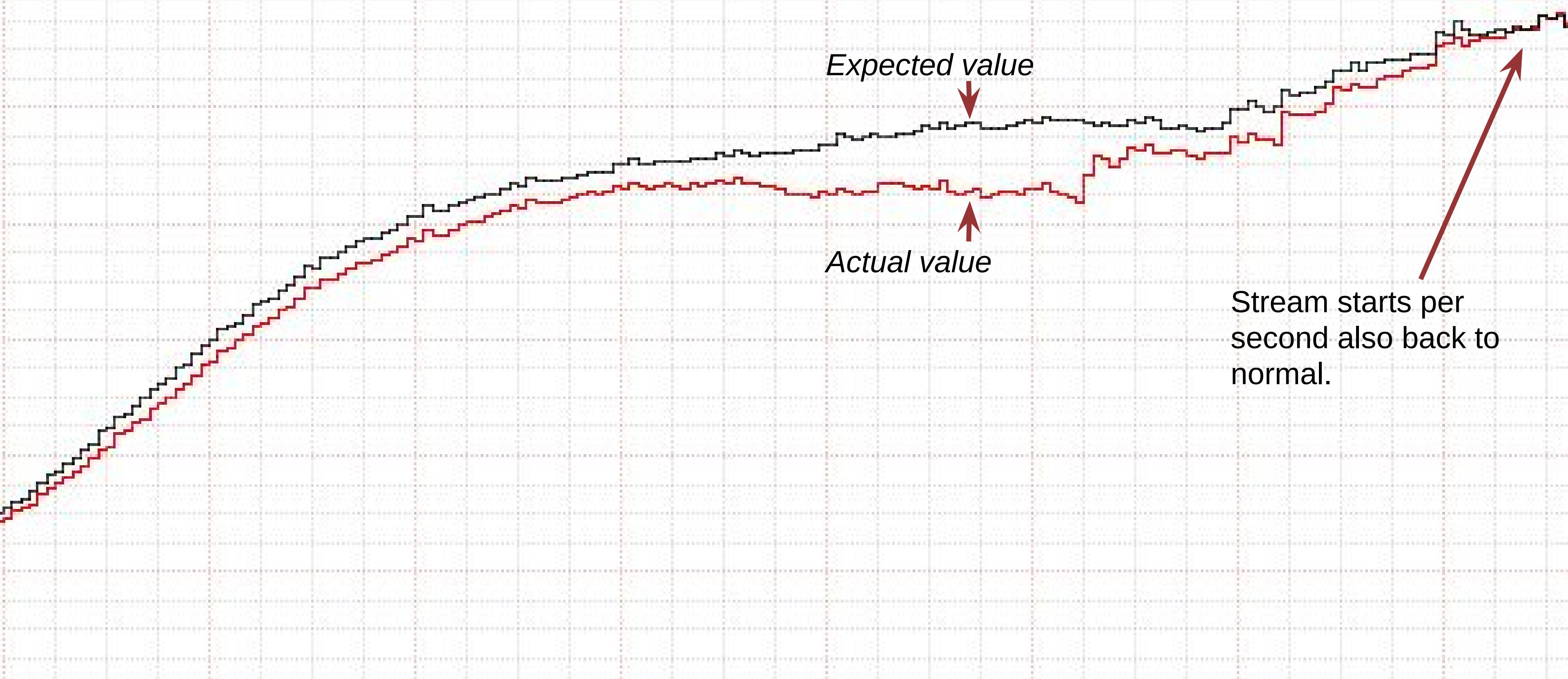
Resolving the issue



Finally root-caused!
Now restarting all
unhealthy servers.

Back to
normal!

Resolving the issue



Expected value

Actual value

Stream starts per second also back to normal.

The Netflix API

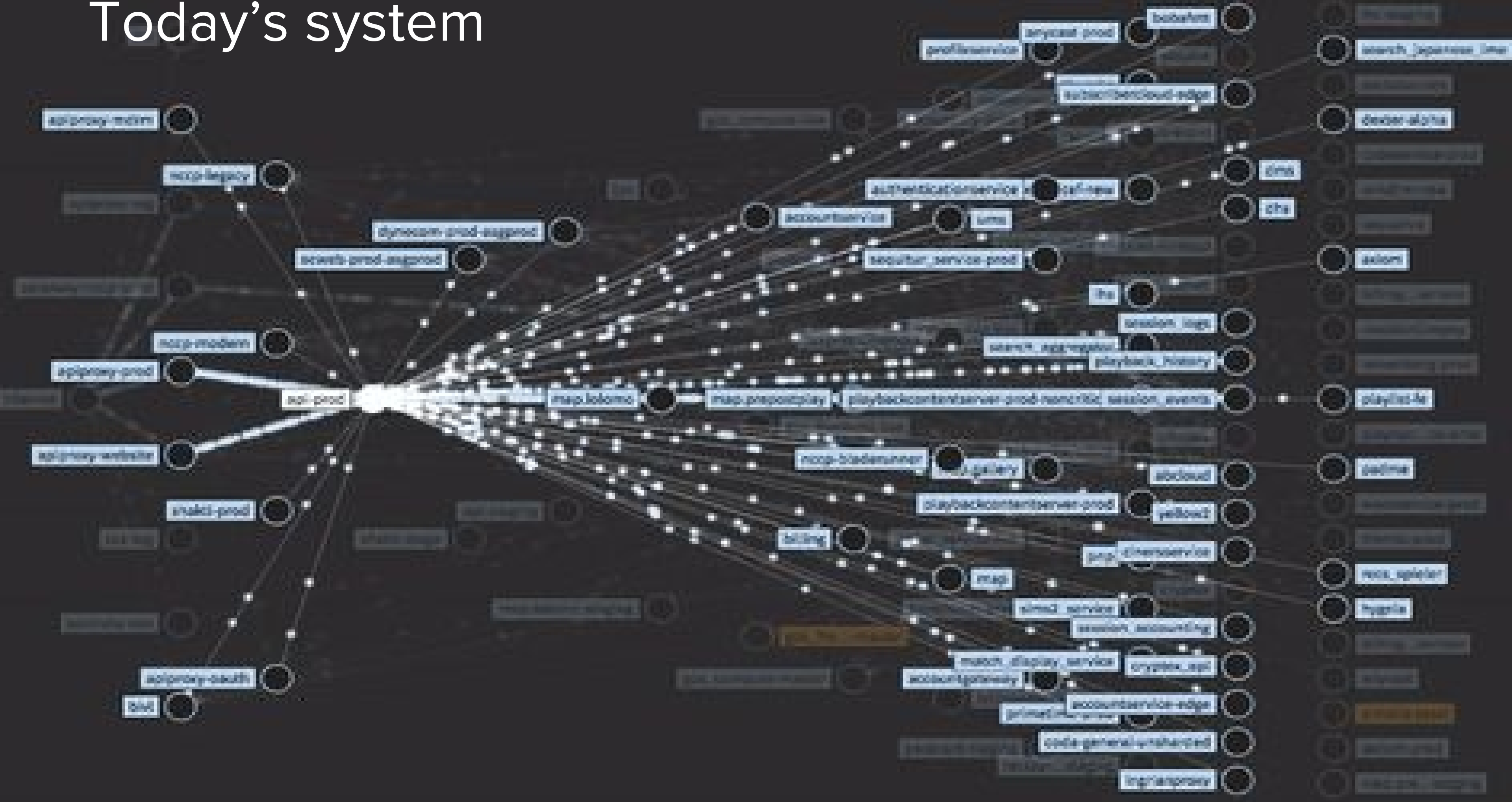
Access to Netflix mid-tier services

Non-member experience

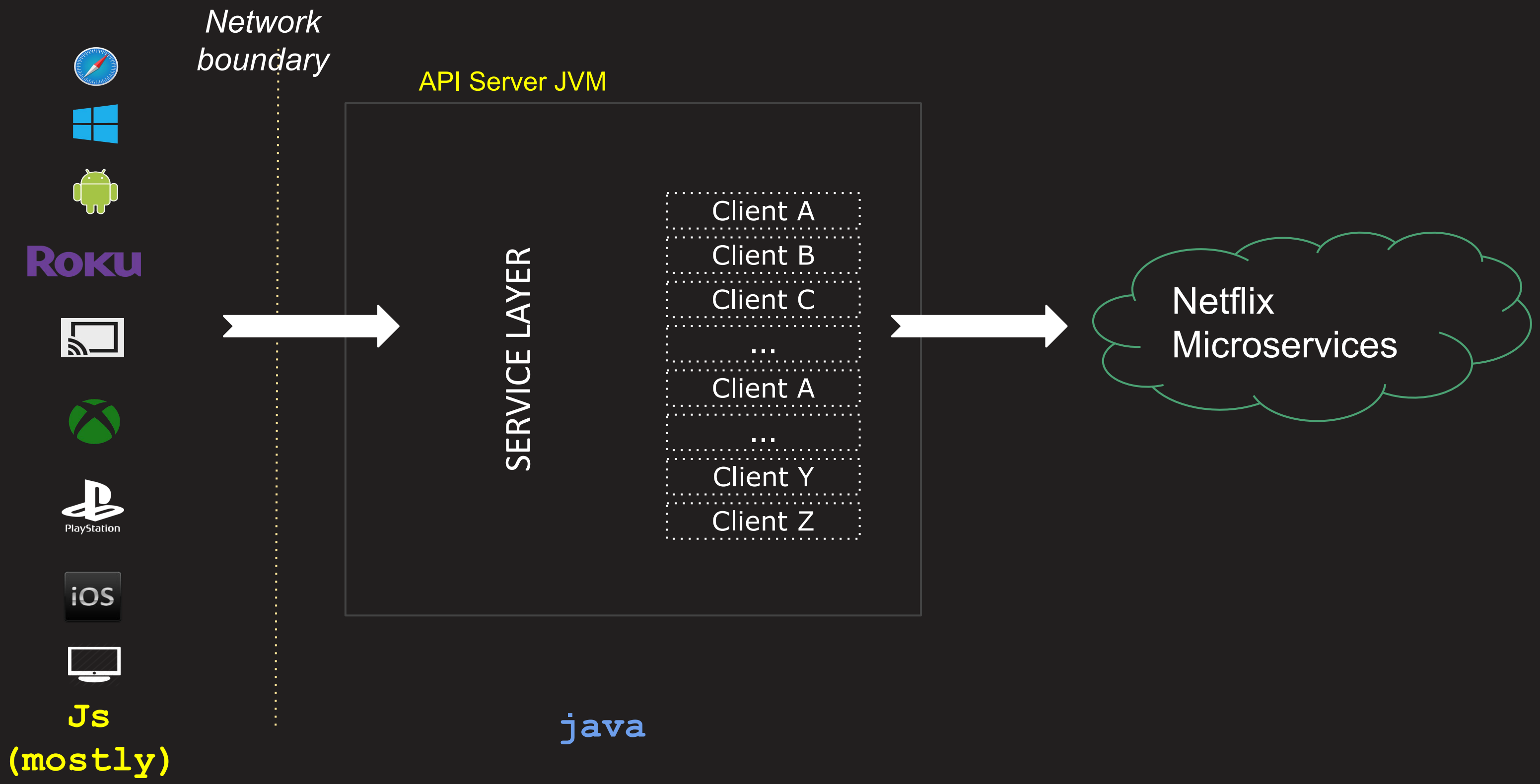
Discovery

Playback

Today’s system



Today’s system (simplified)



What we need

Great tooling

Low error rate

Great developer experience

High throughput

Resiliency

Reliability

Great documentation

Flexible APIs

Velocity

Limited device-server chattiness

Low latency

Insights into operational health

What we need

Great tooling

Low error rate

Great developer experience

High throughput

Resiliency

Reliability

Great documentation

Flexible APIs

Velocity

Limited device-server chattiness

Insights into operational health

Low latency

What we need

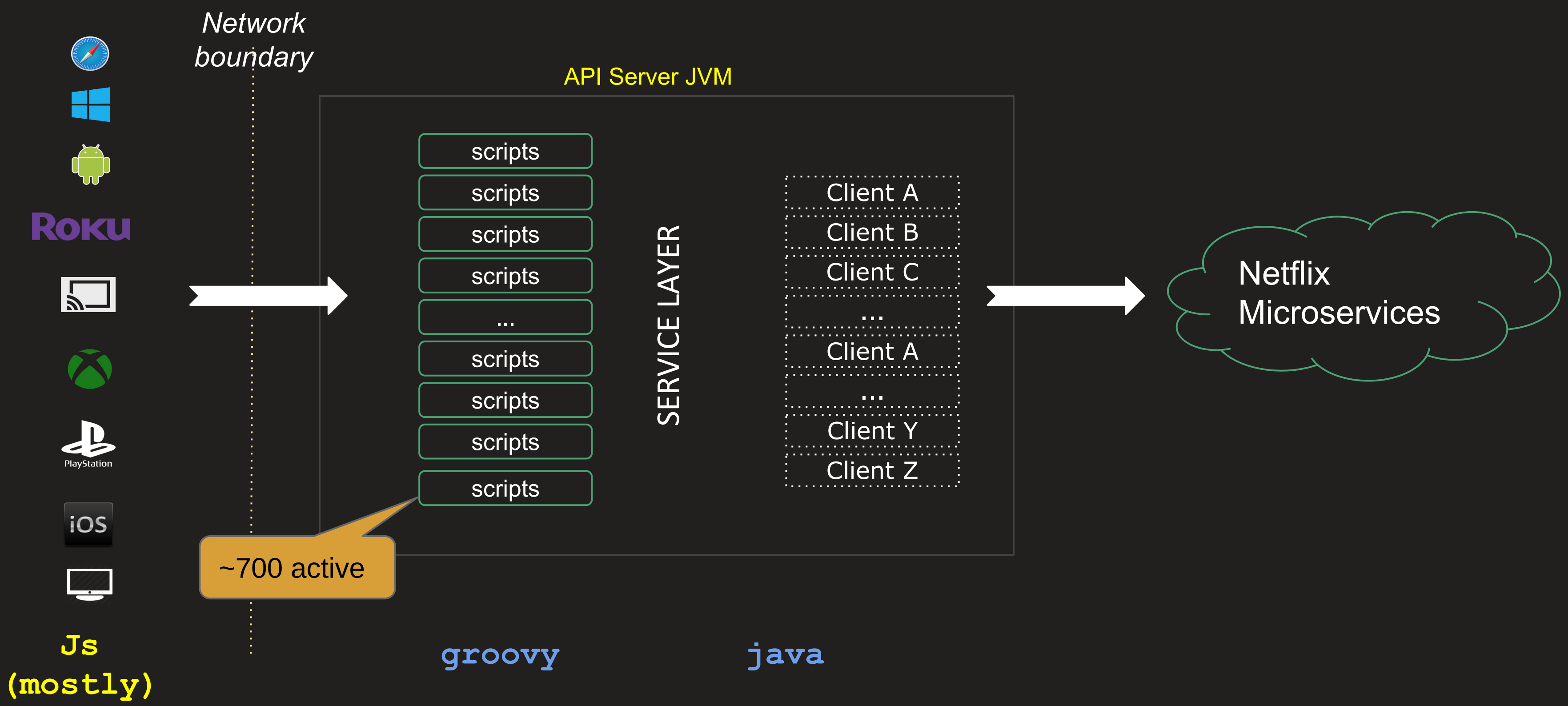
Flexible APIs

Velocity

Resiliency

Great developer experience

Today's architecture



Flexibility for devices

[Script 1]

```
    [...]
    Device1VideoCommon.formatKidsSeason (apiRequest,[...], imageUrl)
    [...]
```

[Script 2]

```
    [...]
    Device2Common.formatAllSeasons([...])
    [...]
```

[Script 3] // for A/B test

```
    [...]
    dataPublishingService.getShowFeedbackBuilder(user, video)
    [...]
```

What we need

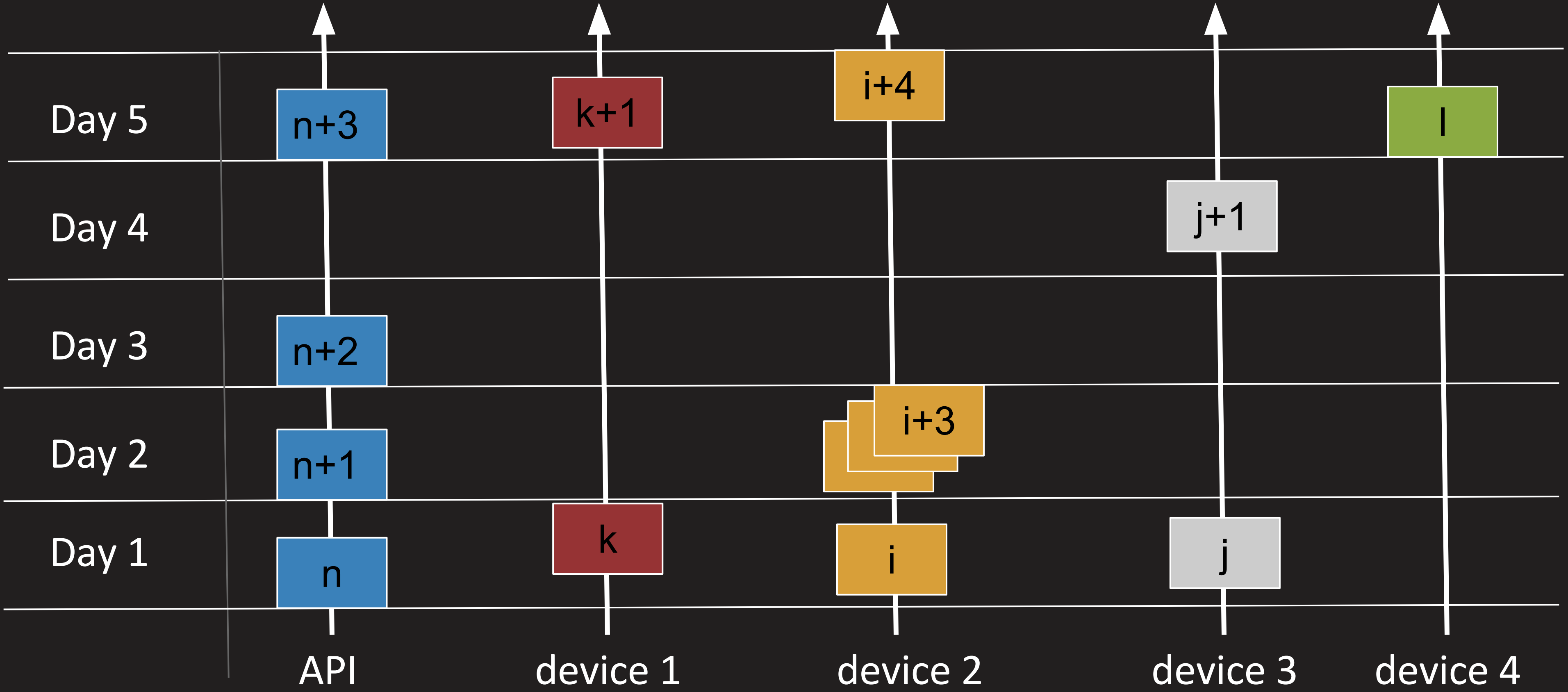
Flexible APIs

Velocity

Resiliency

Great developer experience

Developer Velocity:
Decoupled deployments of versions



	API	device 1	device 2	device 3	device 4
Day 5	n+3	k+1	i+4		l
Day 4				j+1	
Day 3	n+2				
Day 2	n+1		i+3		
Day 1	n	k	i	j	

What we need

Flexible APIs

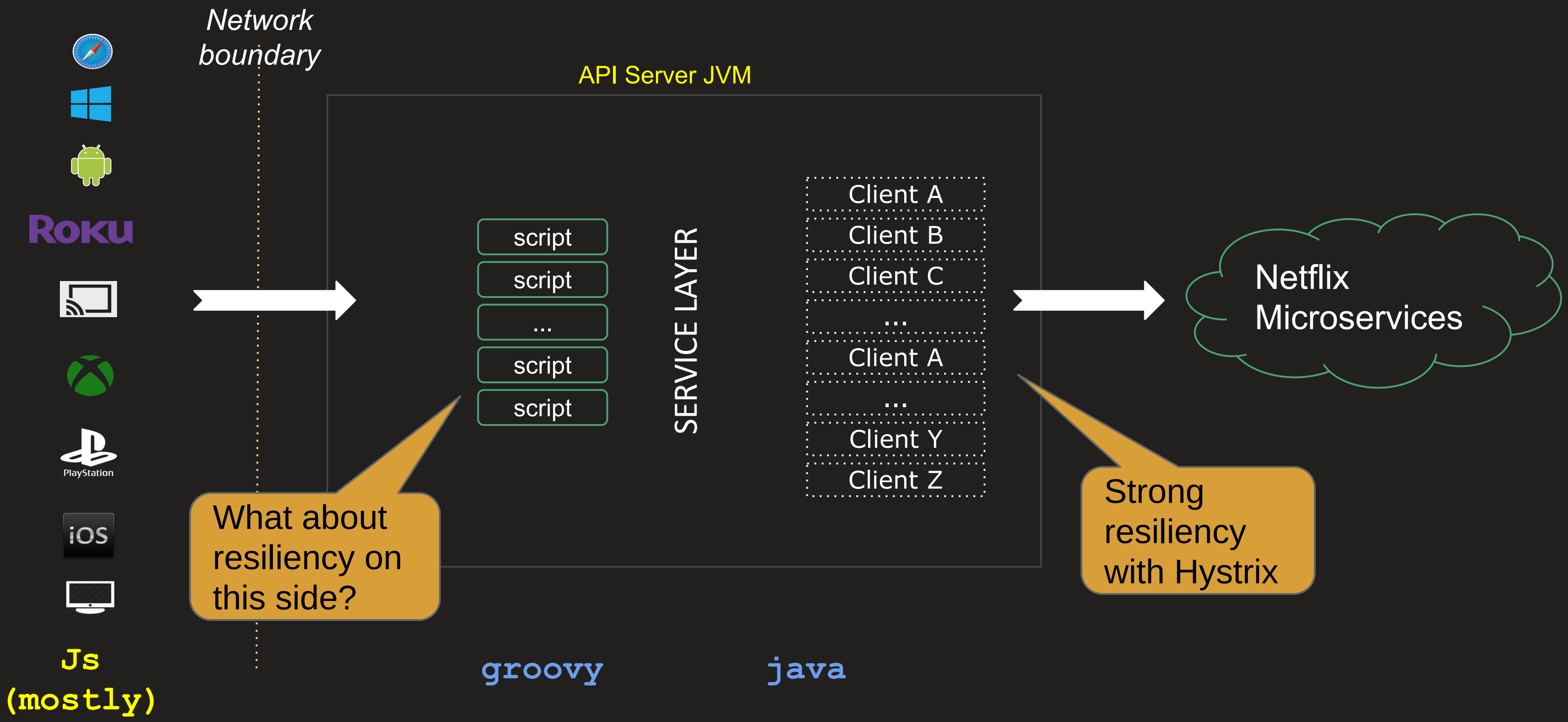
Velocity

Resiliency

Great developer experience

Challenge #1: Resiliency

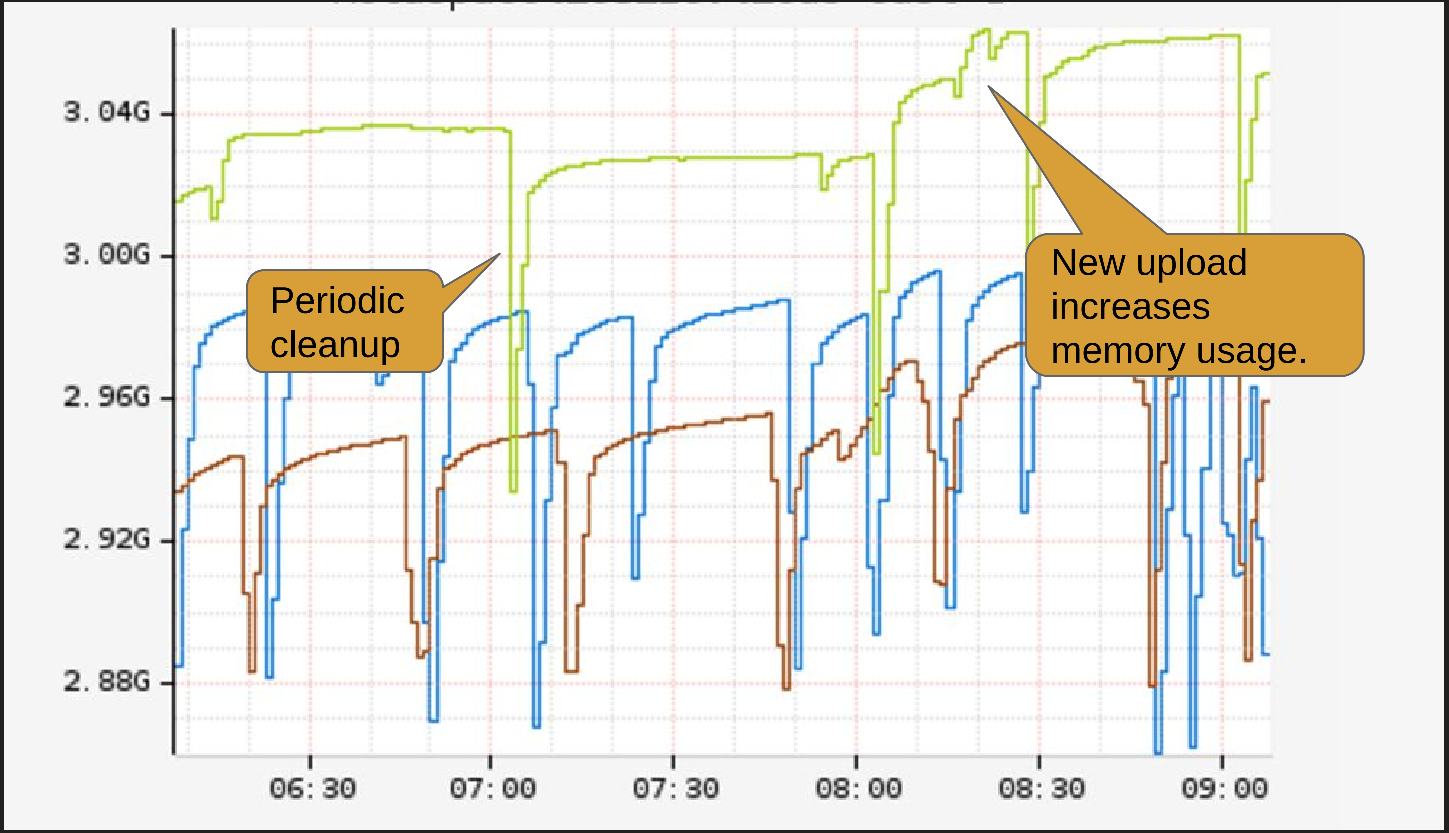
Resiliency in today’s system



What about resiliency on this side?

Strong resiliency with Hystrix

Example: memory usage



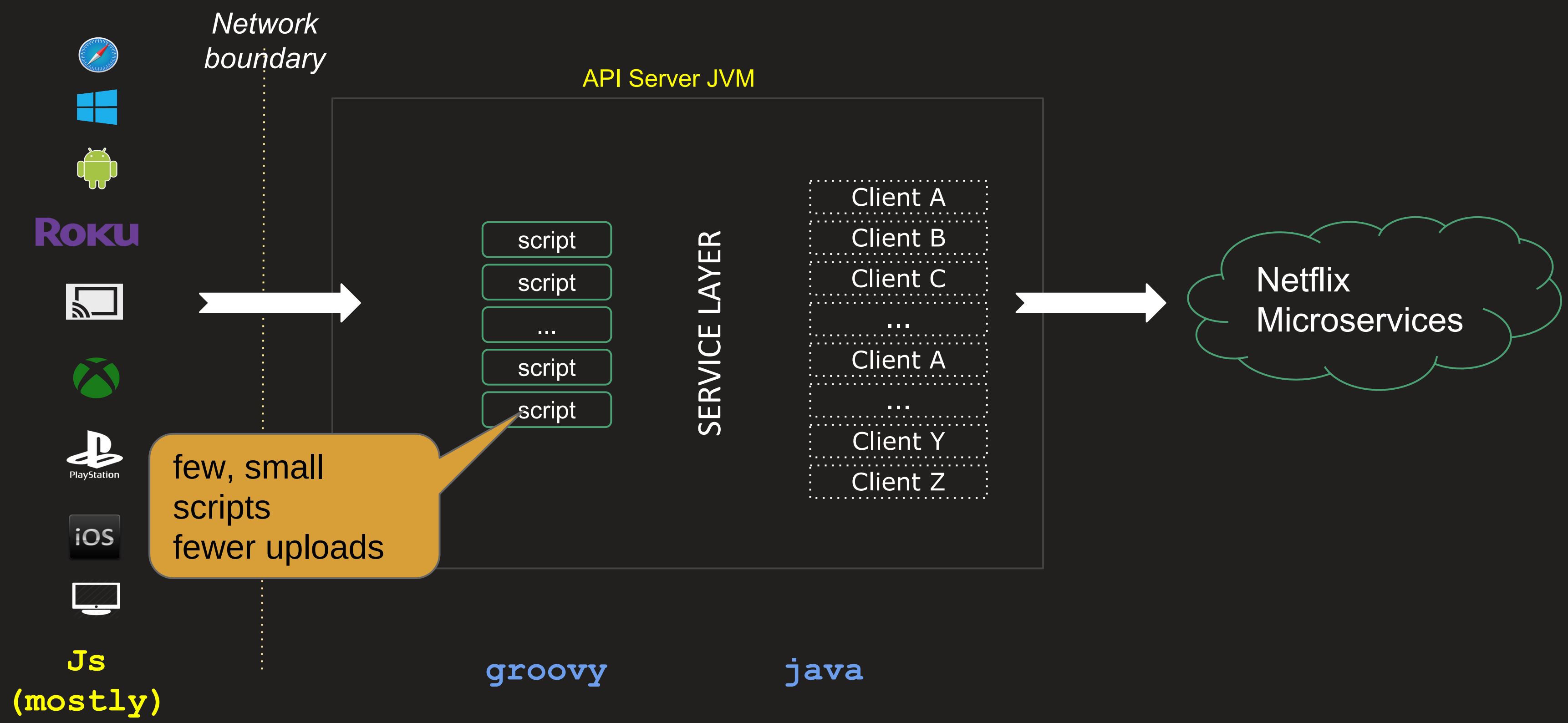
Periodic cleanup

New upload increases memory usage.

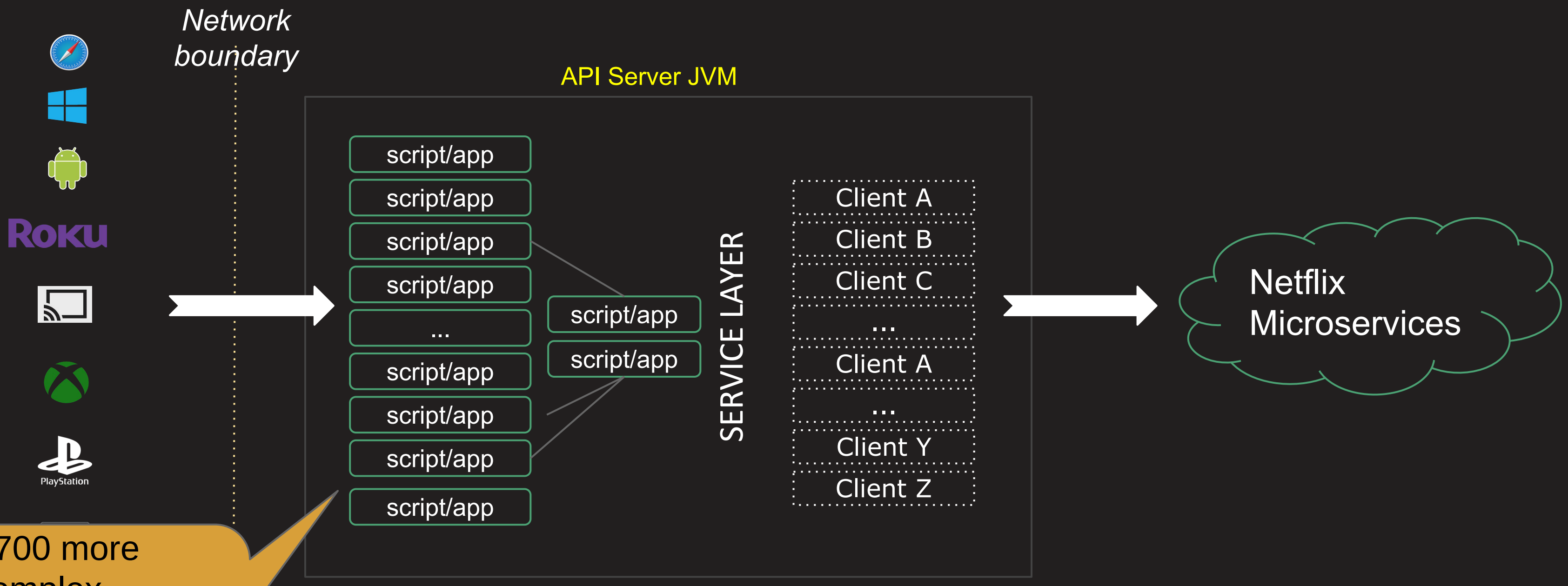
Uploading Scripts to Production Servers at any time

Freedom & Responsibility

1-2 years ago


Network boundary
API Server JVM
Roku
PlayStation
iOS
script
script
...
script
script
SERVICE LAYER
Client A
Client B
Client C
...
Client A
...
Client Y
Client Z
Netflix Microservices
few, small scripts fewer uploads
Js (mostly)
groovy
java

Today

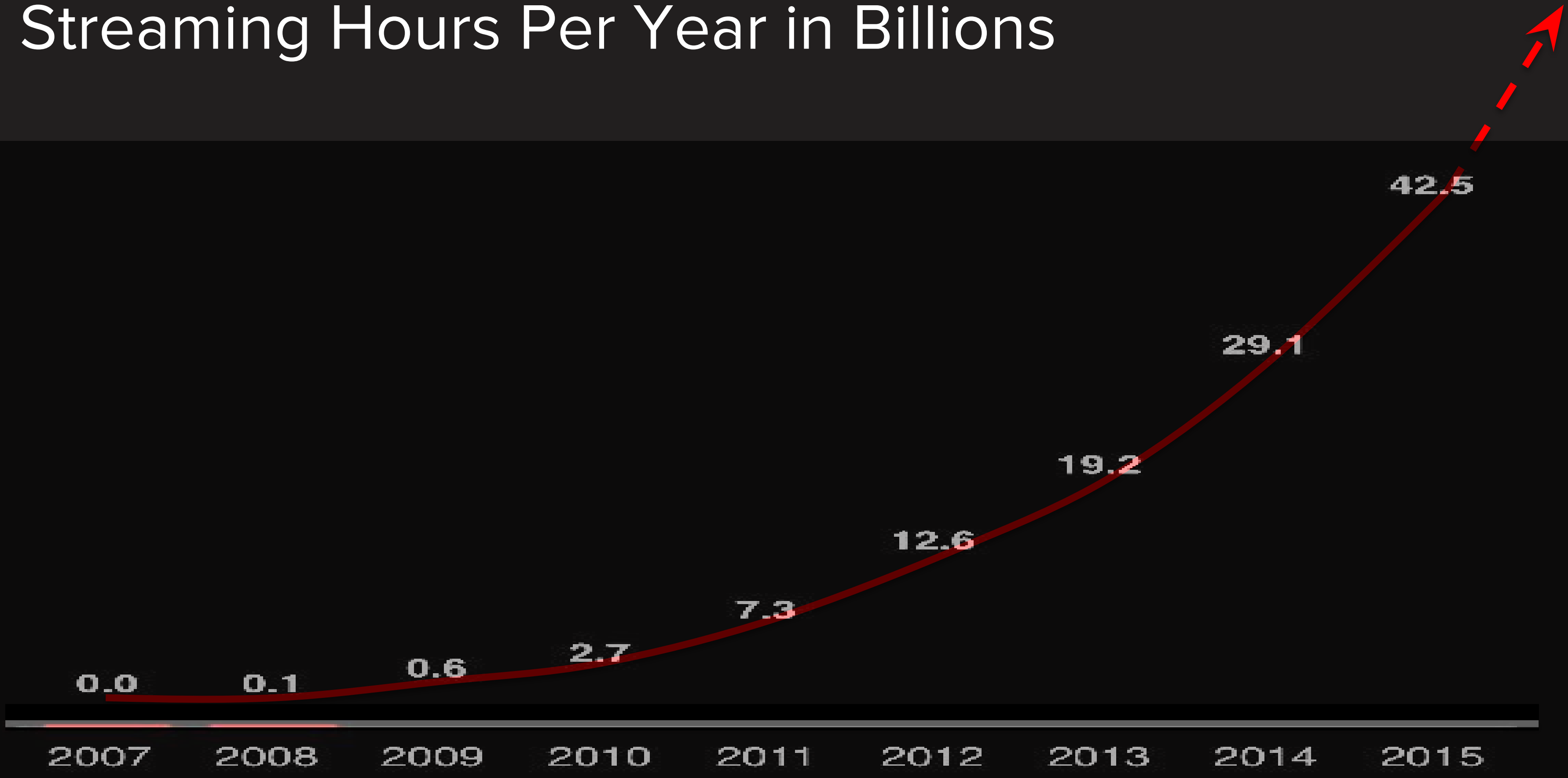


~700 more complex scripts/apps, 10-50 uploads per day

groovy

java

Streaming Hours Per Year in Billions



Changing risk profile

Growing number and complexity of scripts (scripts → apps)

Growing number of users/streams (→ RPS)

But also:

Growing number of devices

Growing number of A/B tests

Growing number of languages

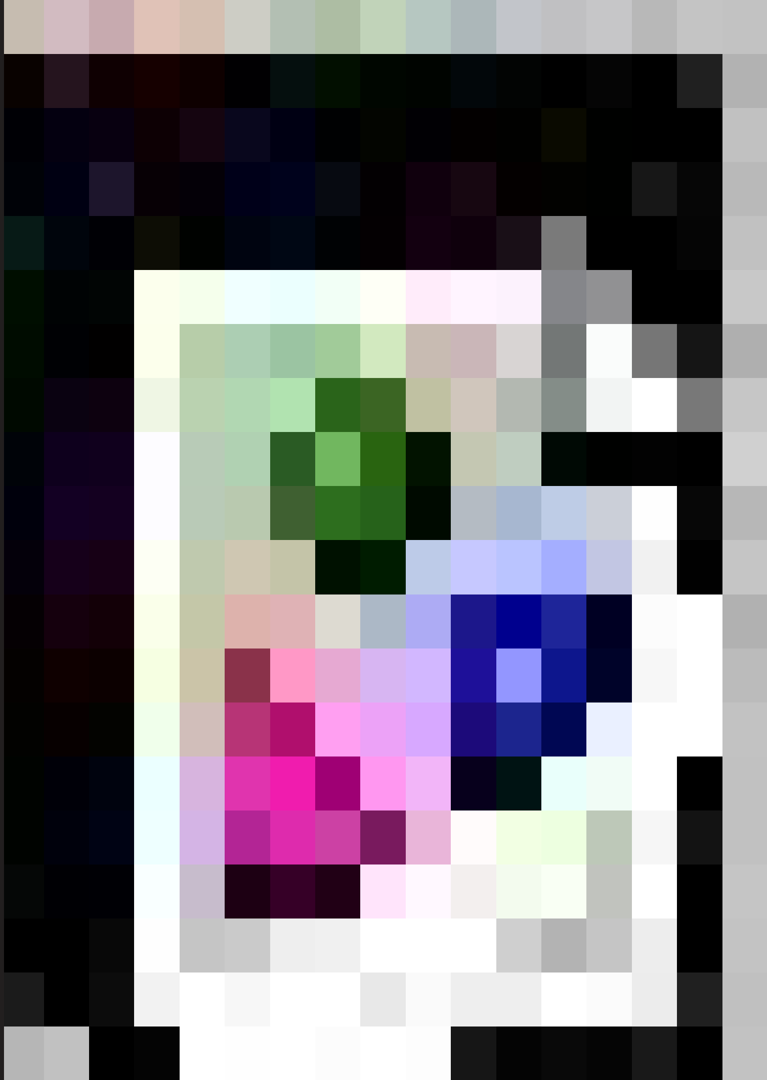
Lack of process isolation is a growing risk.

Possible mitigations

Split scripts across multiple farms

Bigger servers

Publish scripts rarely, slowly, only if they fit into memory



Velocity vs. Resiliency

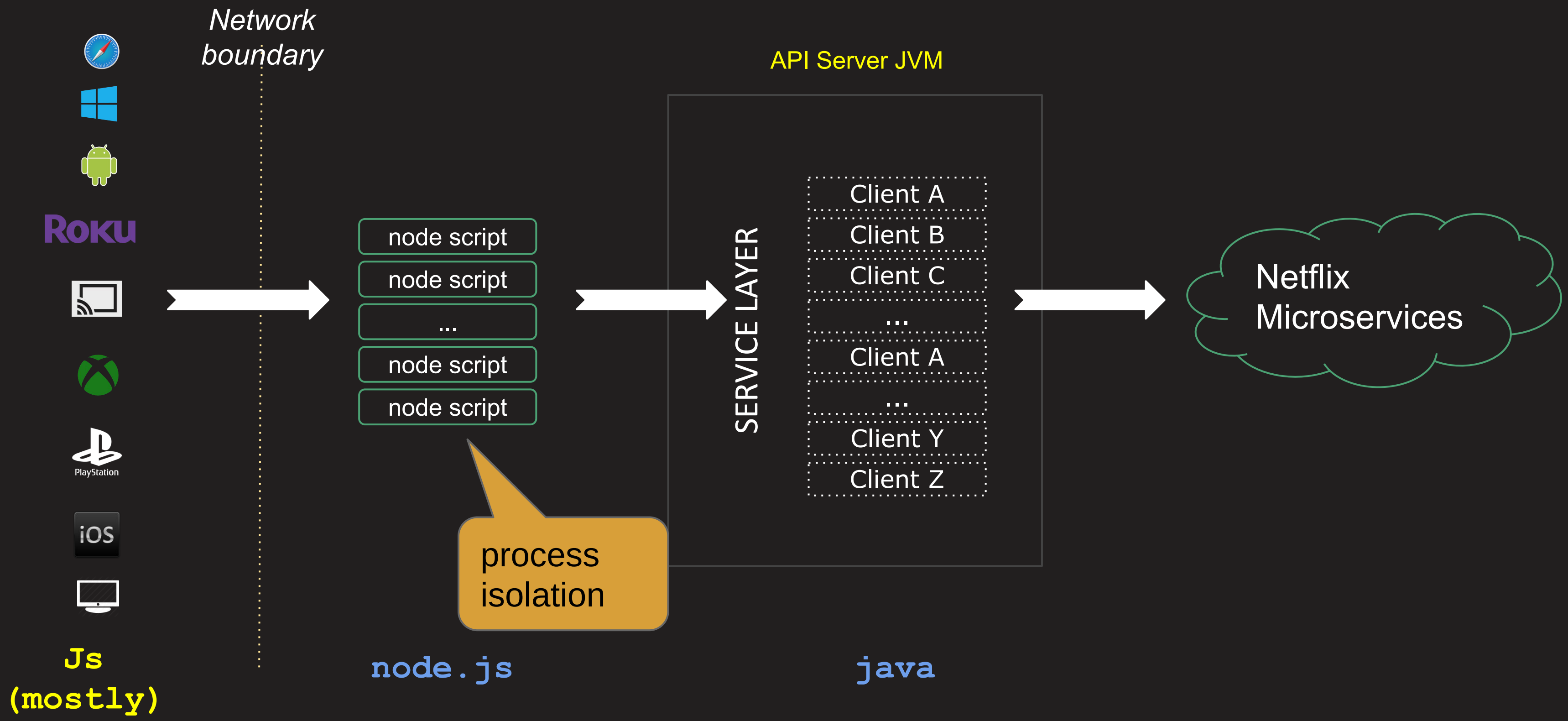
How to break out of the trade-off?

Moving toward our ideal API:
What will change

Scripts will run in containers

Scripts will call API remotely

The (near) future



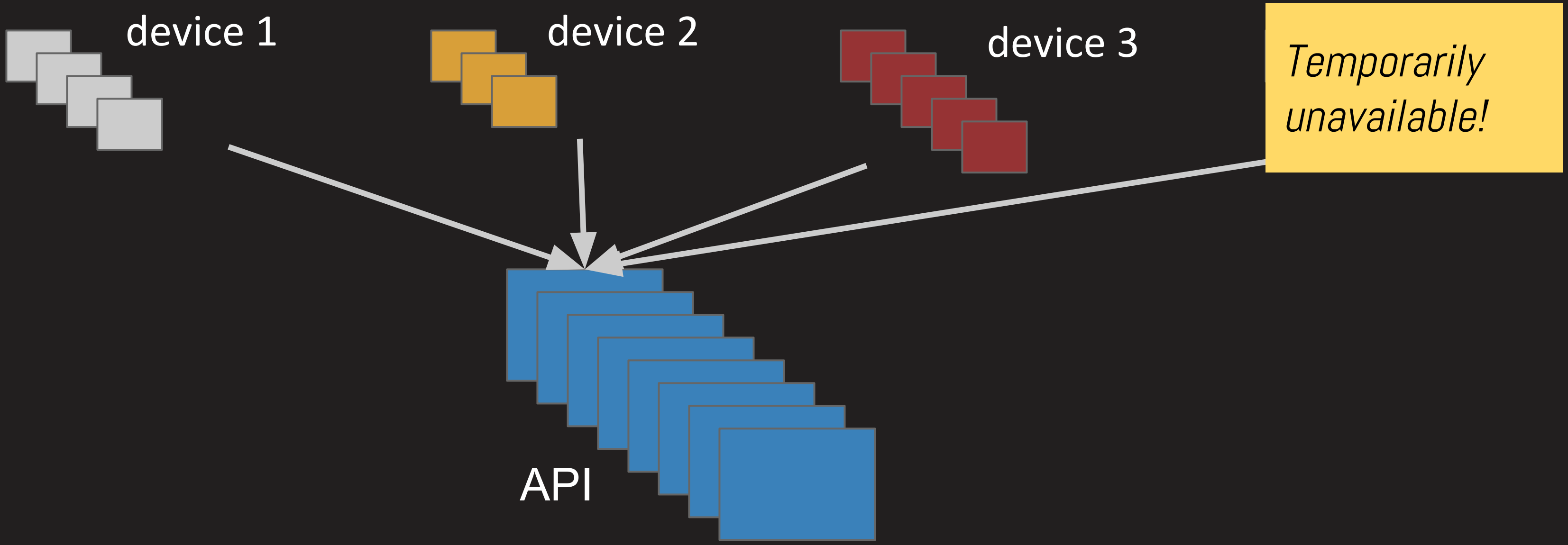
Why containers?

Process isolation

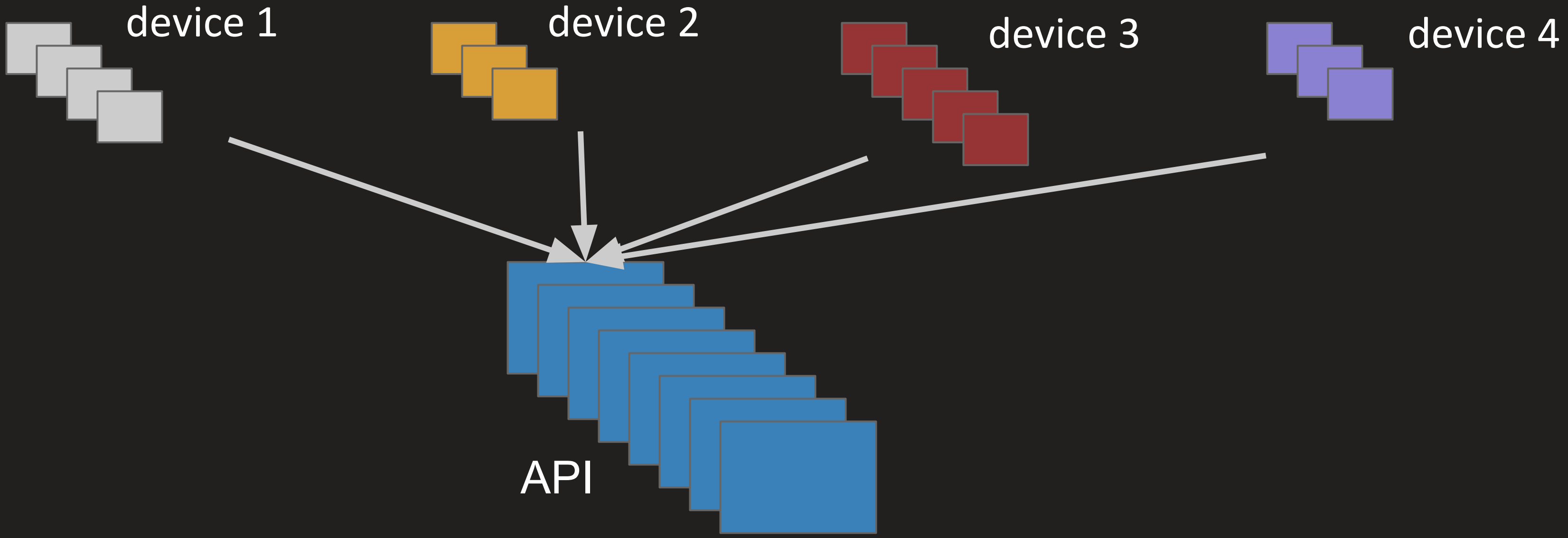
Fast startup

Consistent developer experience across environments

Isolated failures: scripts don’t affect each other (usually)



Independent autoscaling



Fast startup

New API server: minutes

New container: seconds

Fast rollouts, fast rollback, faster MTTR

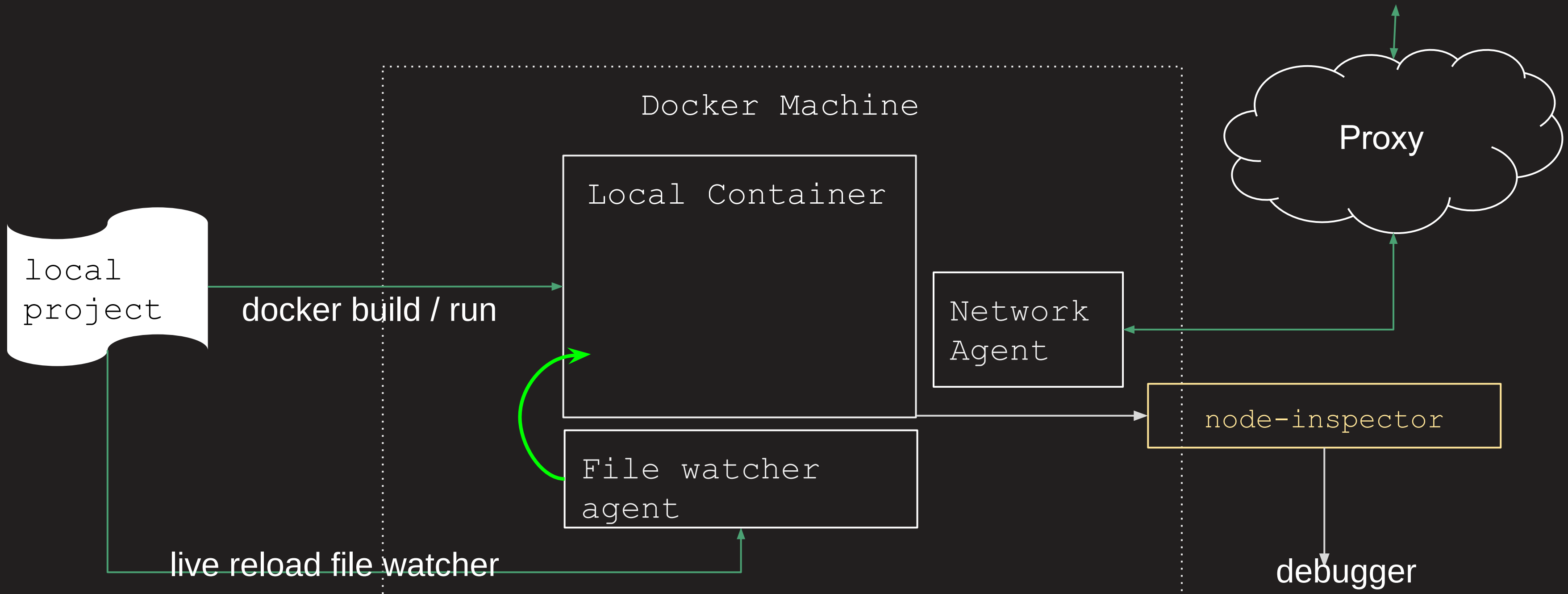
Challenge #2: Great developer experience

Step-through-debugging (today)

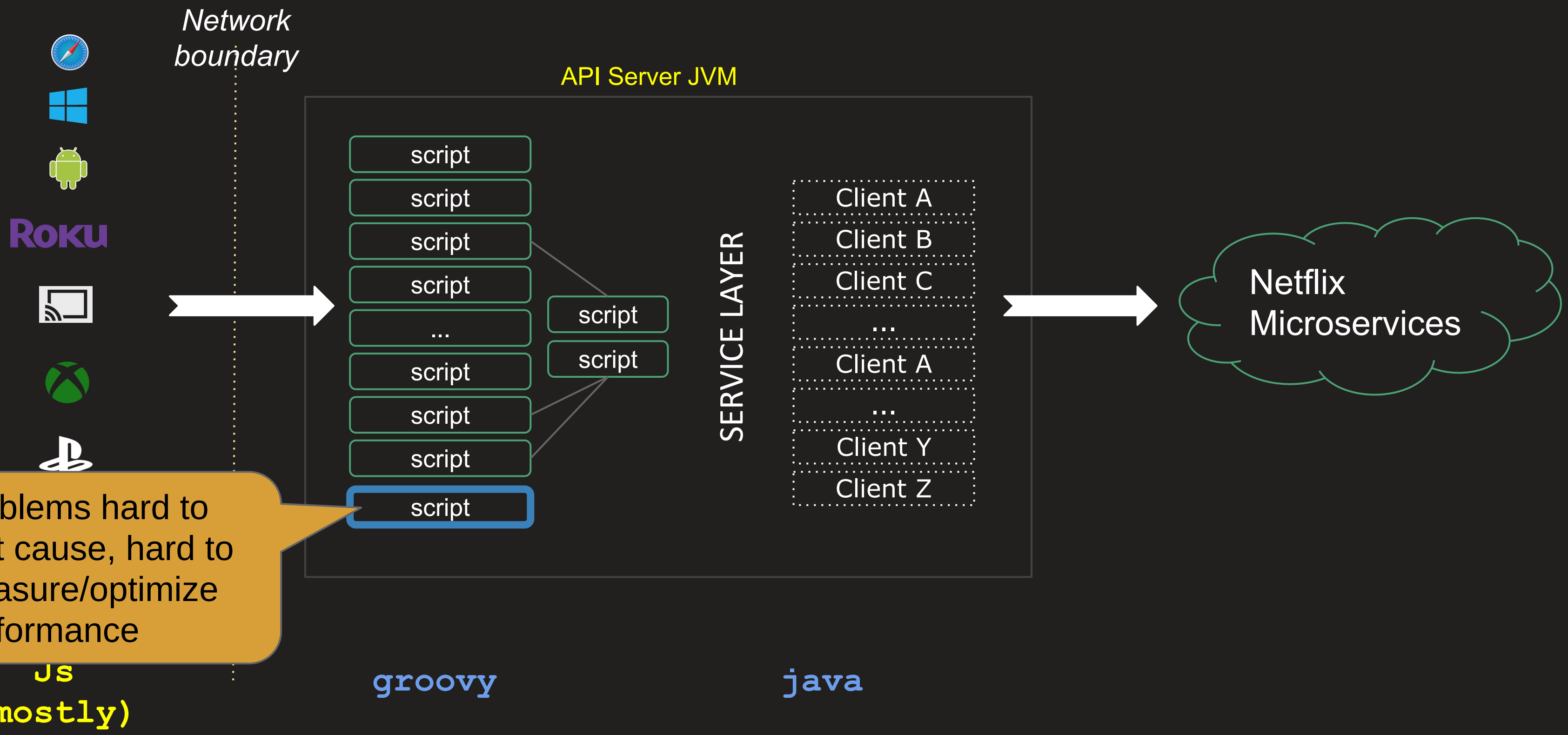
Requires cumbersome local setup

Many rely on more rudimentary debugging

Local development (future)



Run-time debugging/optimization (today)



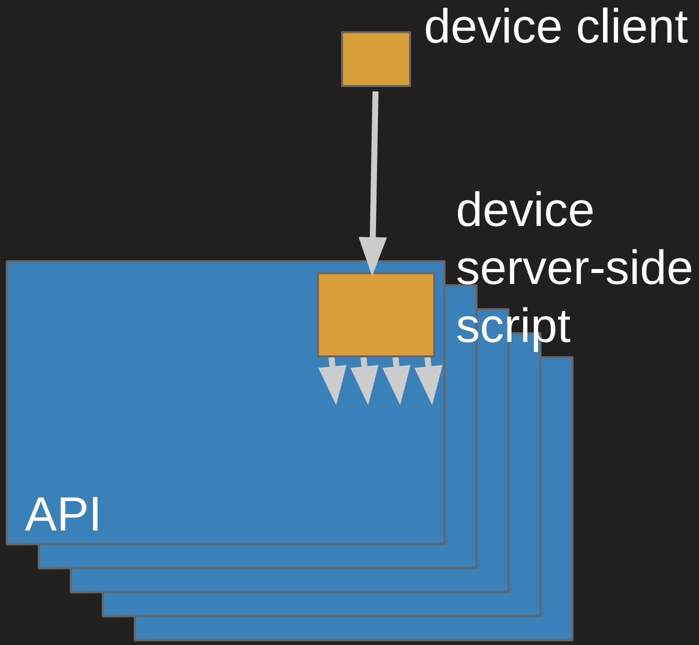
Problems hard to root cause, hard to measure/optimize performance

Script → API interaction (today)

Calls from script to API are method calls inside the JVM

Difficult to measure/optimize

Difficult to throttle

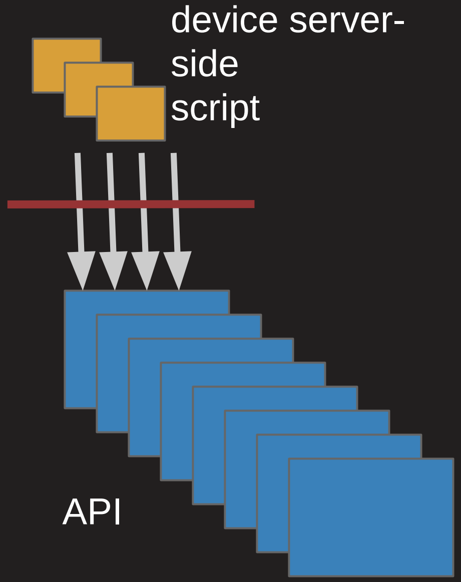


Script → API interaction (future)

Calls remotely

Easier to measure/optimize

Easier to throttle



Platform for device teams

Deploy script in seconds

UI/CLI for self-management

Default configuration

Lifecycle management:

dependency management, deployment pipelines, continuous integration, canaries, autoscaling

Tooling and insights:

alerting, logging, audits and analytics, dashboards

Default configuration

Lifecycle management:

dependency management, deployment pipelines, continuous integration, **canaries**, autoscaling

Tooling and insights:

alerting, logging, audits and analytics, dashboards

Default configuration

Lifecycle management:

dependency management, deployment pipelines, continuous integration, canaries, autoscaling

Tooling and insights:

alerting, **logging**, audits and analytics, dashboards

NeWT: **Ne**tflix **W**orkflow **T**oolkit (CLI)

docker

git

npm

YEOMAN

Spinnaker

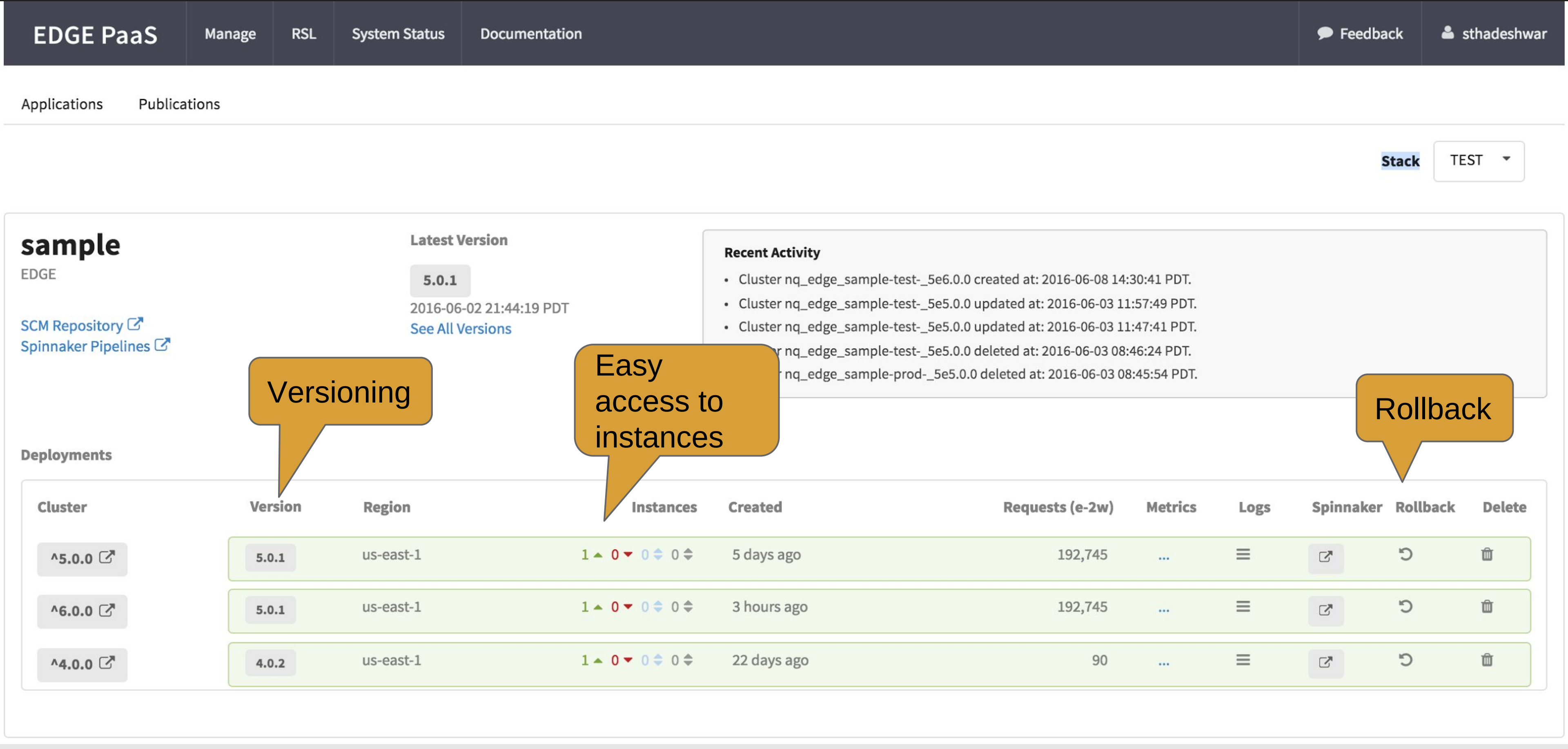
Titus

ATLAS

Homebrew

The missing package manager for OS X

Corresponding UI



EDGE PaaS | Manage | RSL | System Status | Documentation | Feedback | sthadeshwar

Applications   Publications

Stack   TEST

sample

EDGE

SCM Repository
Spinnaker Pipelines

Latest Version

5.0.1

2016-06-02 21:44:19 PDT

See All Versions

Recent Activity

- Cluster nq_edge_sample-test-_5e6.0.0 created at: 2016-06-08 14:30:41 PDT.
- Cluster nq_edge_sample-test-_5e5.0.0 updated at: 2016-06-03 11:57:49 PDT.
- Cluster nq_edge_sample-test-_5e5.0.0 updated at: 2016-06-03 11:47:41 PDT.
- r nq_edge_sample-test-_5e5.0.0 deleted at: 2016-06-03 08:46:24 PDT.
- r nq_edge_sample-prod-_5e5.0.0 deleted at: 2016-06-03 08:45:54 PDT.

Versioning

Easy access to instances

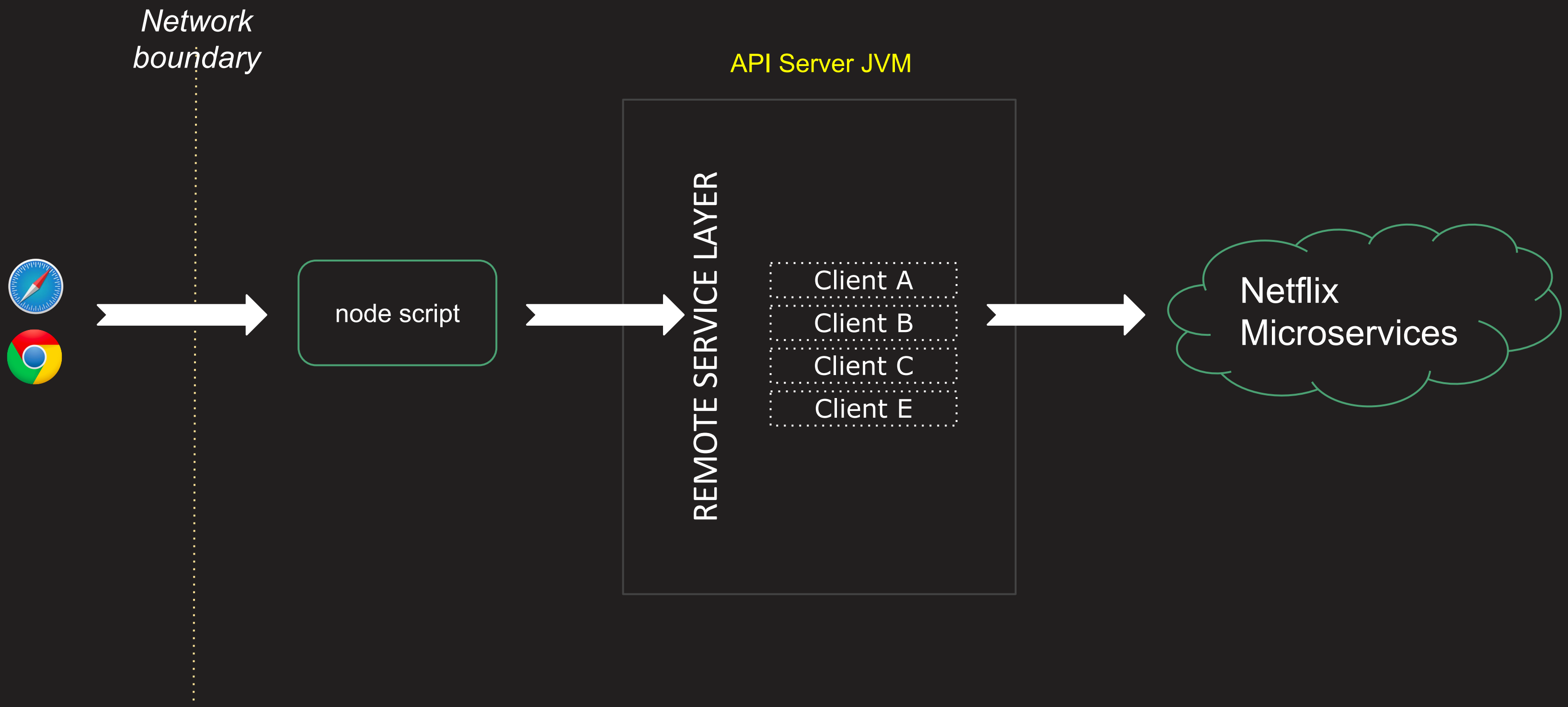
Rollback

Deployments

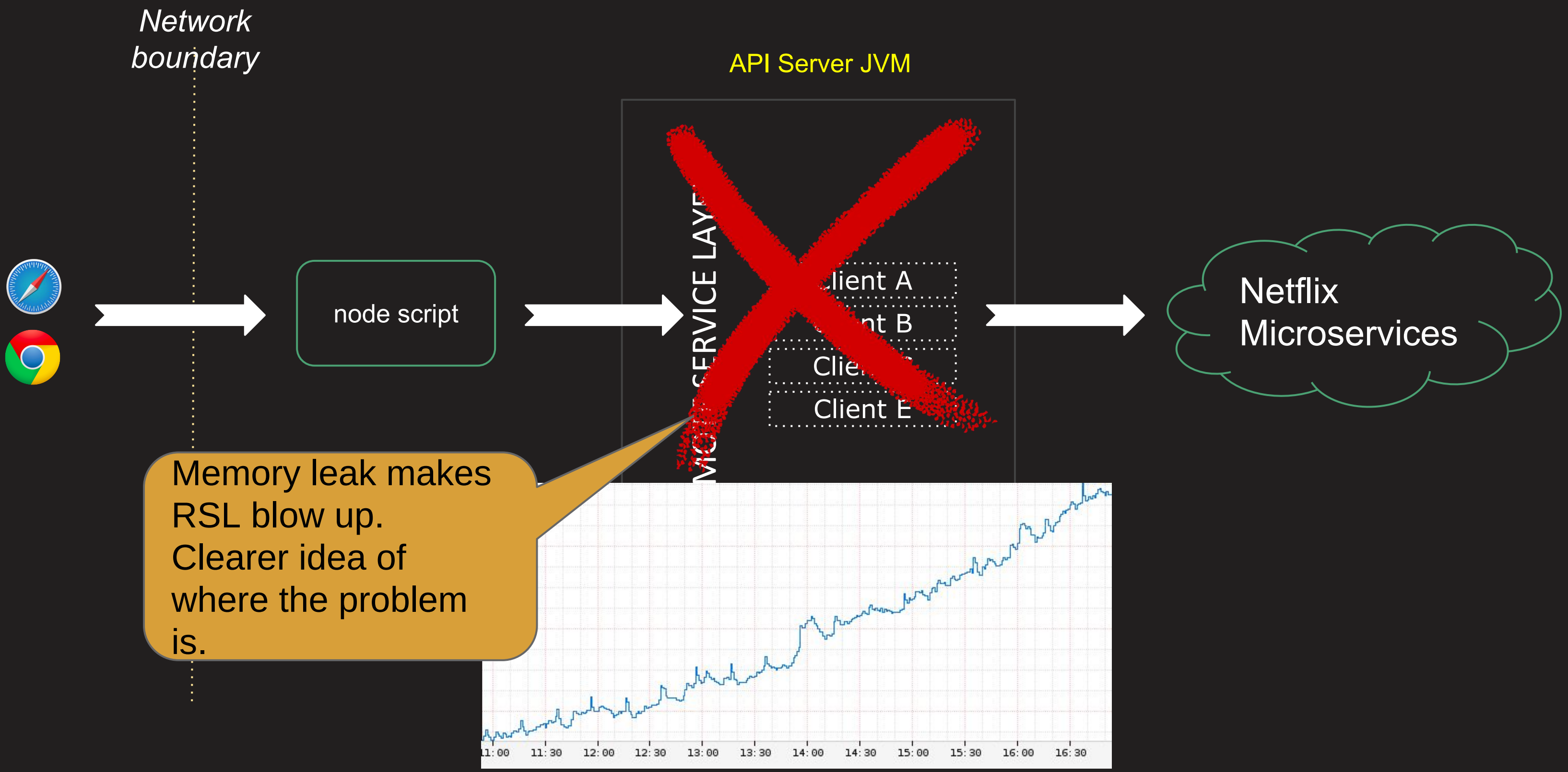
Cluster	Version	Region	Instances	Created	Requests (e-2w)	Metrics	Logs	Spinnaker	Rollback	Delete
^5.0.0	5.0.1	us-east-1	1 ▲ 0 ▼ 0 ⬍ 0 ⬍	5 days ago	192,745	...	≡	↗	↺	🗑
^6.0.0	5.0.1	us-east-1	1 ▲ 0 ▼ 0 ⬍ 0 ⬍	3 hours ago	192,745	...	≡	↗	↺	🗑
^4.0.0	4.0.2	us-east-1	1 ▲ 0 ▼ 0 ⬍ 0 ⬍	22 days ago	90	...	≡	↗	↺	🗑

Initial impressions

First end-to-end implementation and shadow traffic

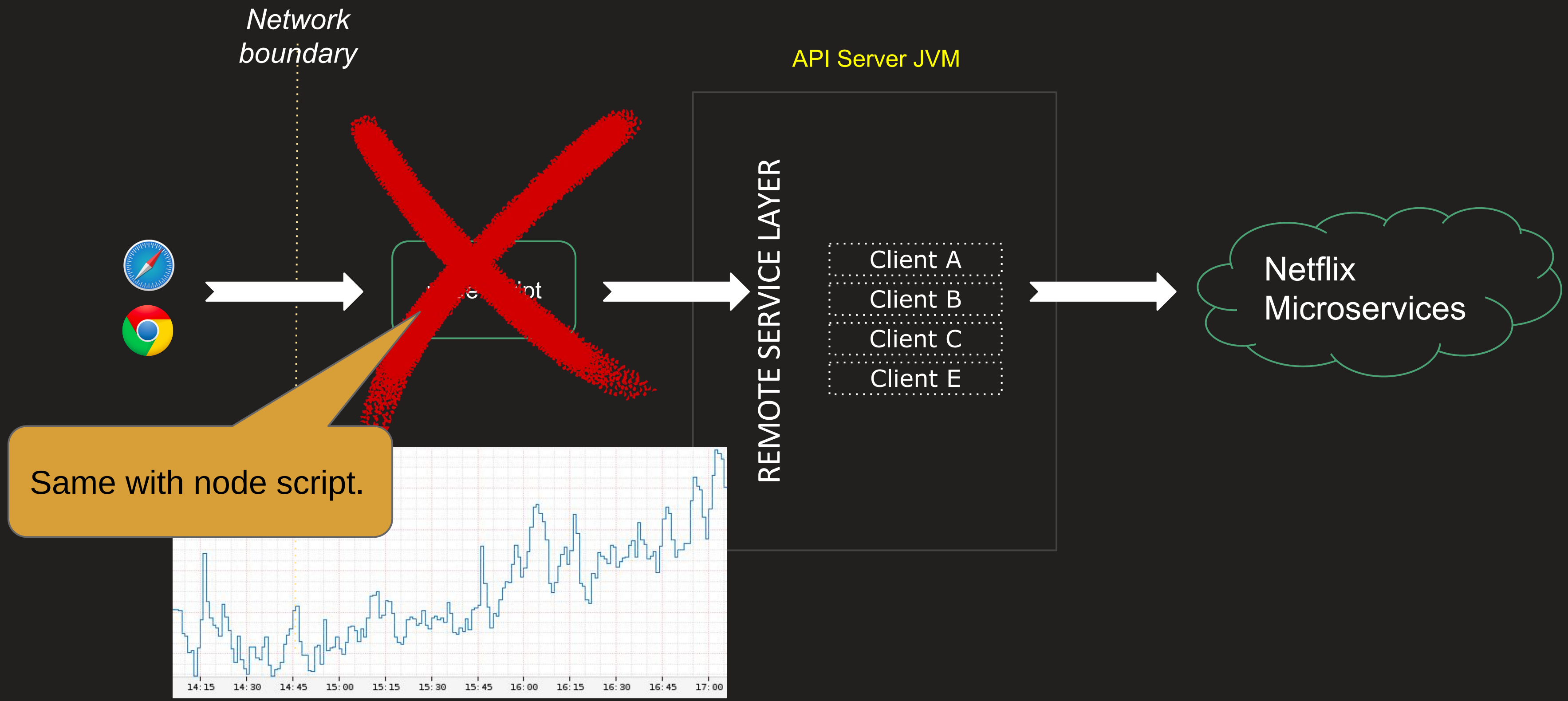


Problem isolation (ex: memory leak)



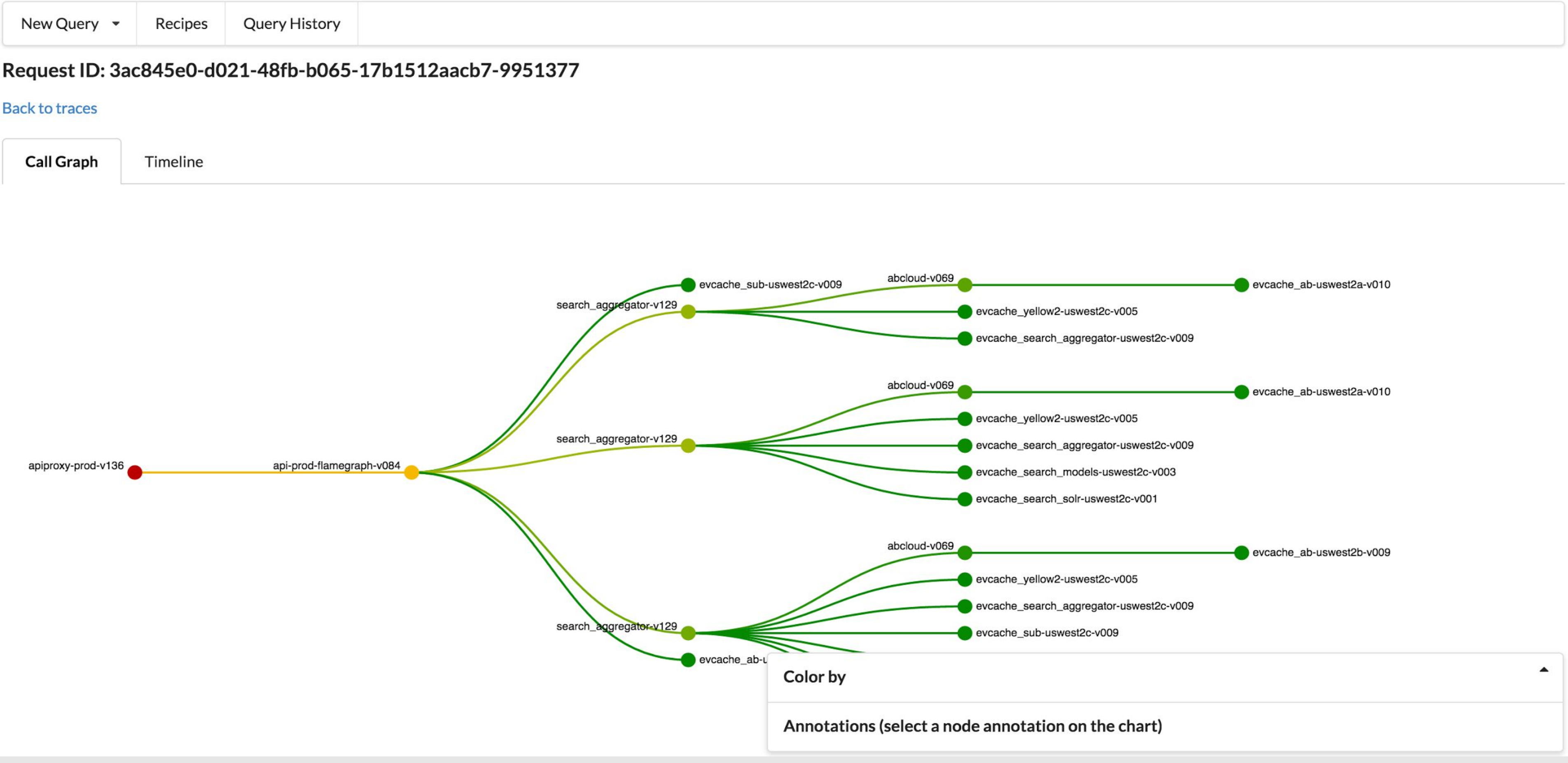
Memory leak makes RSL blow up. Clearer idea of where the problem is.

Problem isolation (ex: memory leak)



Same with node script.

Request tracing: clearer picture of fan-out

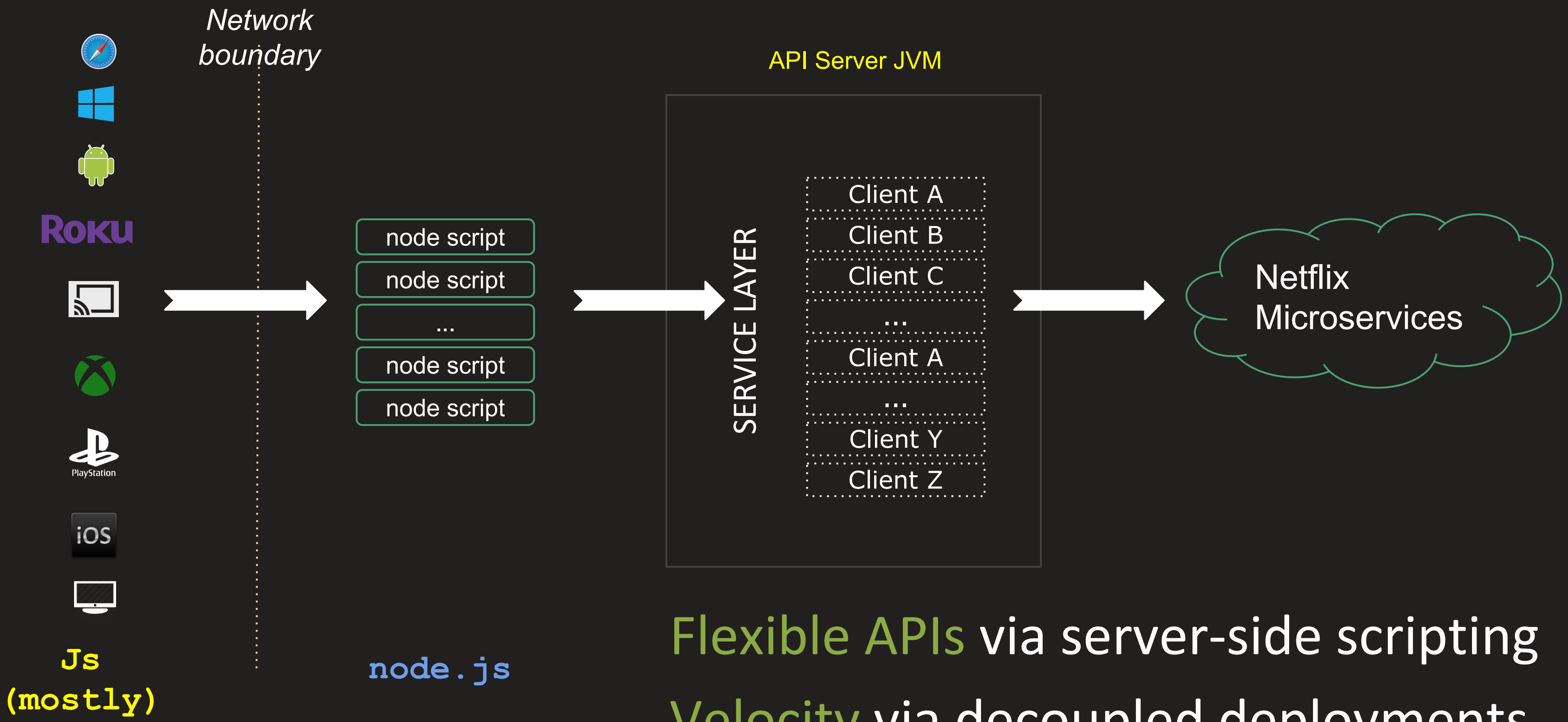


New Query
Recipes
Query History

Request ID: 3ac845e0-d021-48fb-b065-17b1512aacb7-9951377

Back to traces
Call Graph
Timeline
apiproxy-prod-v136
api-prod-flamegraph-v084
evcache_sub-uswest2c-v009
search_aggregator-v129
abcloud-v069
evcache_ab-uswest2a-v010
evcache_yellow2-uswest2c-v005
evcache_search_aggregator-uswest2c-v009
search_aggregator-v129
abcloud-v069
evcache_ab-uswest2a-v010
evcache_yellow2-uswest2c-v005
evcache_search_aggregator-uswest2c-v009
evcache_search_models-uswest2c-v003
evcache_search_solr-uswest2c-v001
search_aggregator-v129
abcloud-v069
evcache_ab-uswest2b-v009
evcache_yellow2-uswest2c-v005
evcache_search_aggregator-uswest2c-v009
evcache_sub-uswest2c-v009
evcache_ab-u
Color by
Annotations (select a node annotation on the chart)

How not to compromise what we’re good at



Flexible APIs via server-side scripting

Velocity via decoupled deployments

What we need

Low error rate

Great developer experience

Great tooling

High throughput

Resiliency

Reliability

Great documentation

Flexible APIs

Velocity

Insights into operational health

Low latency

Limited device-server chattiness

“Progress is impossible without change, and those who cannot change their minds cannot change anything.”

- *George Bernard Shaw*

Other Netflix talks at QCon New York

SCHEDULING A FULLER HOUSE: CONTAINER MGMT @NETFLIX
Andrew Spyker &
Sharma Podila
Tue, 11:50 am

CHAOS KONG - ENDOWING NETFLIX WITH ANTIFRAGILITY
Luke Kosewski
Tue, 11:50 am

THE EVOLUTION OF THE JAVASCRIPT
Jafar Husain
Wed, 4:10 pm

ASYNC PROGRAMMING IN JS: THE END OF THE LOOP
Jafar Husain
Wed, 4:10 pm

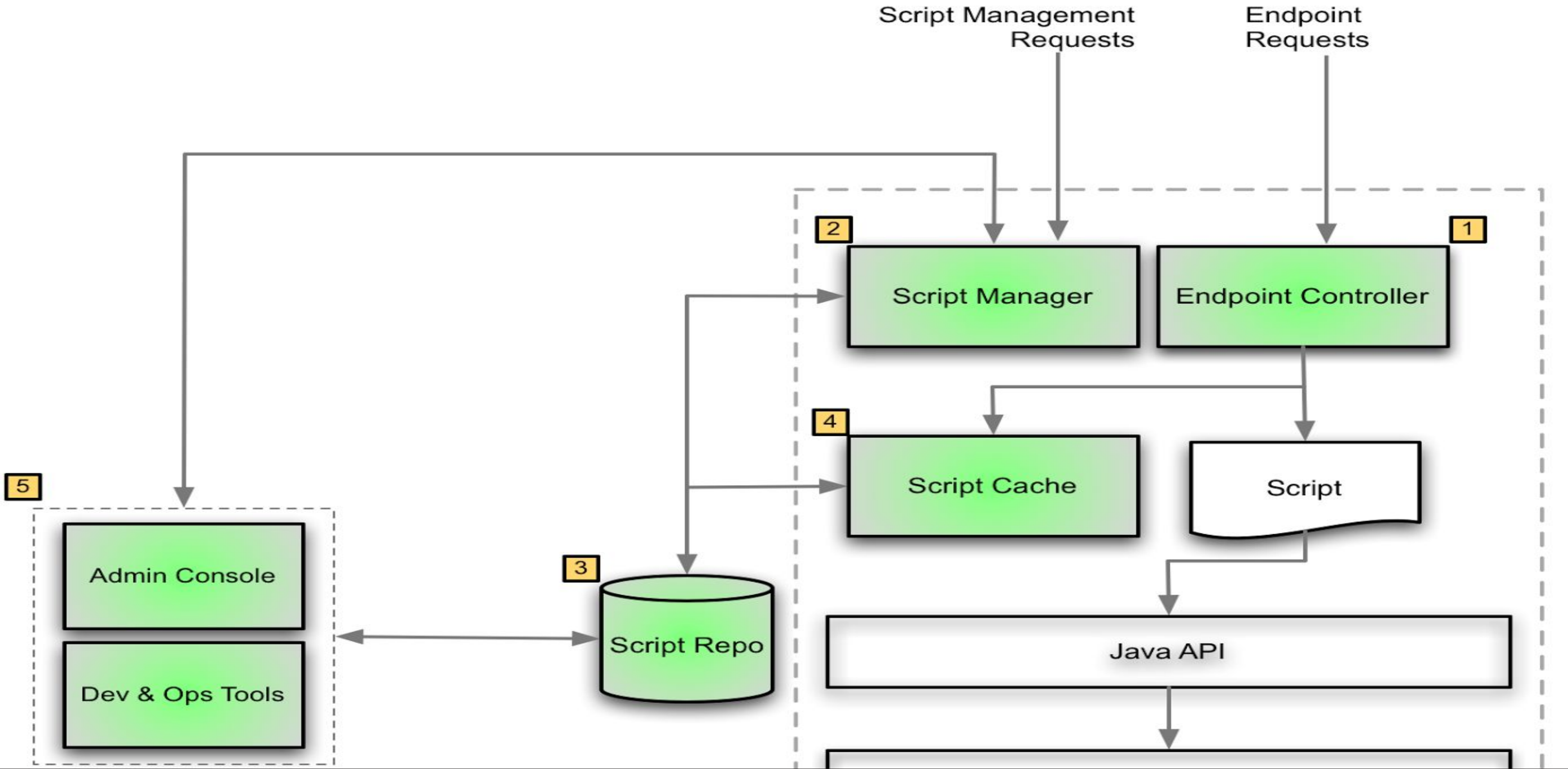
Thank you!

Katharina Probst

https://www.linkedin.com/in/katharinaprobst

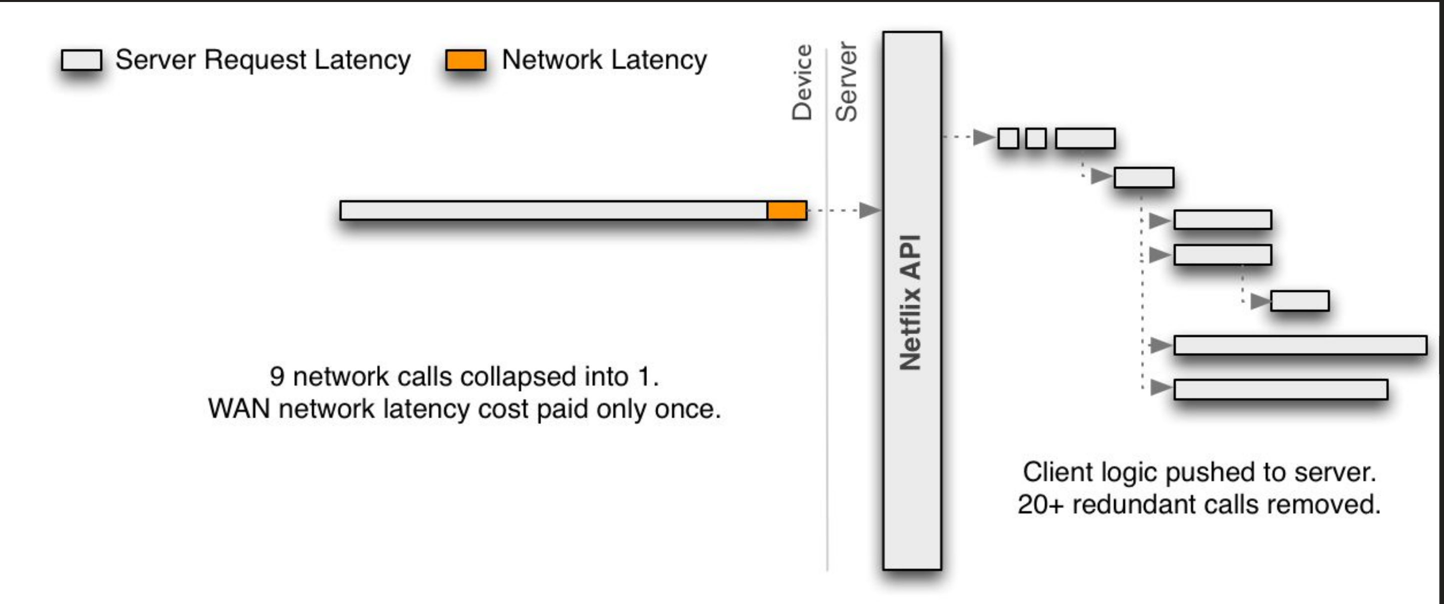
@probst_kathrin

Script management



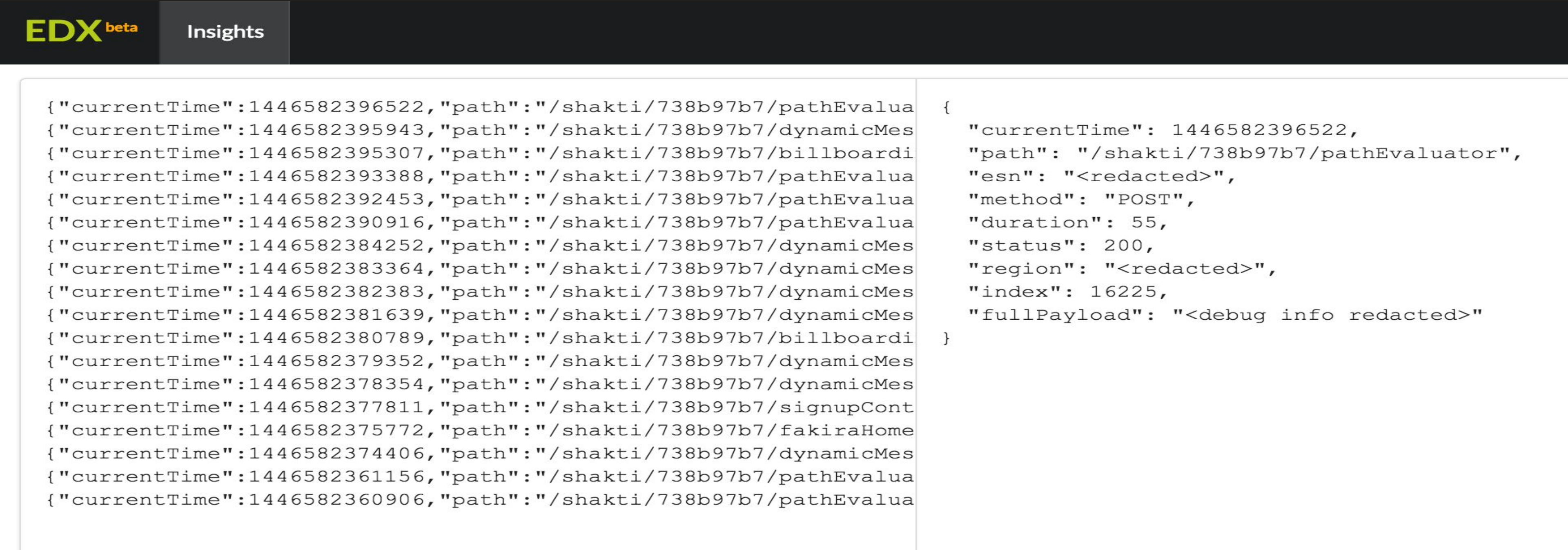
Outtakes

Limited device-server chattiness



Server Request Latency
Network Latency
Device
Server
Netflix API
9 network calls collapsed into 1.
WAN network latency cost paid only once.
Client logic pushed to server.
20+ redundant calls removed.

Operational insights



```
{"currentTime":1446582396522,"path":"/shakti/738b97b7/pathEvalua
{"currentTime":1446582395943,"path":"/shakti/738b97b7/dynamicMes
{"currentTime":1446582395307,"path":"/shakti/738b97b7/billboardi
{"currentTime":1446582393388,"path":"/shakti/738b97b7/pathEvalua
{"currentTime":1446582392453,"path":"/shakti/738b97b7/pathEvalua
{"currentTime":1446582390916,"path":"/shakti/738b97b7/pathEvalua
{"currentTime":1446582384252,"path":"/shakti/738b97b7/dynamicMes
{"currentTime":1446582383364,"path":"/shakti/738b97b7/dynamicMes
{"currentTime":1446582382383,"path":"/shakti/738b97b7/dynamicMes
{"currentTime":1446582381639,"path":"/shakti/738b97b7/dynamicMes
{"currentTime":1446582380789,"path":"/shakti/738b97b7/billboardi
{"currentTime":1446582379352,"path":"/shakti/738b97b7/dynamicMes
{"currentTime":1446582378354,"path":"/shakti/738b97b7/dynamicMes
{"currentTime":1446582377811,"path":"/shakti/738b97b7/signupCont
{"currentTime":1446582375772,"path":"/shakti/738b97b7/fakiraHome
{"currentTime":1446582374406,"path":"/shakti/738b97b7/dynamicMes
{"currentTime":1446582361156,"path":"/shakti/738b97b7/pathEvalua
{"currentTime":1446582360906,"path":"/shakti/738b97b7/pathEvalua
```

```
{
  "currentTime": 1446582396522,
  "path": "/shakti/738b97b7/pathEvaluator",
  "esn": "<redacted>",
  "method": "POST",
  "duration": 55,
  "status": 200,
  "region": "<redacted>",
  "index": 16225,
  "fullPayload": "<debug info redacted>"
}
```