LINE TODAY - 微服務架構支撐千萬活躍用戶的影音內容平臺

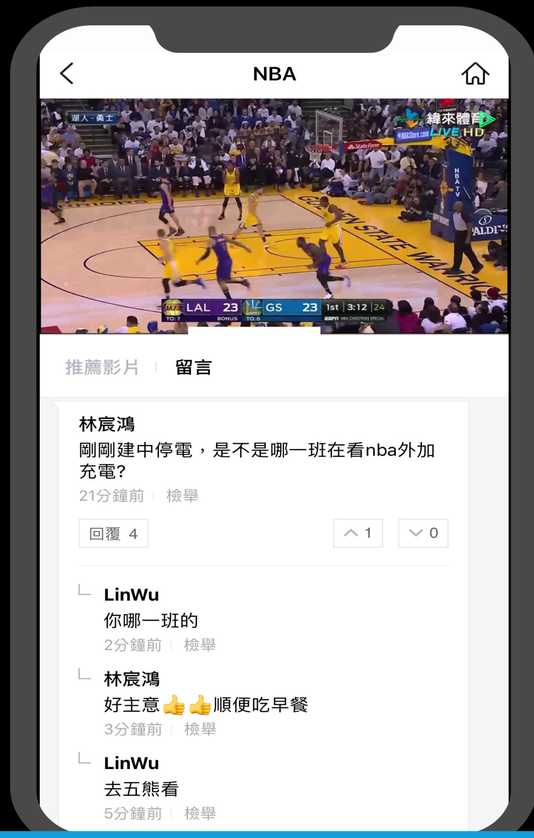
Libra Huang - LINE TODAY

10/18/2022

Agenda

- LINE TODAY and its architecture
- How miniservices and K8S changes our development and operation
- Lessons learnt

LINE TODAY Stays With You Today



個人化訊息
8:00
財經天氣

12:30
國際/國內/娛樂
新聞內容

生活內容
15:00
話題/投票/電影

18:00
TODAY看世界
官方帳號

直播
20:00
棒球/NBA/演唱會

Taiwan Thailand Hong Kong

18M MAU

LINE TODAY Taiwan

LINE TODAY Architecture

CDN

Object Storage

Cache Server

NUXTJS

Frontend Server

In Mem Cach e

spring boot

Backend Server

APACHE HBASE

mongoDB

redis

elastic

kafka

Data Warehouse

kubernetes

spring boot

Mini Service

spring boot

Feeding Service

React

Internal CMS

Vue.js

External CMS

python

Report Service

Third Party API

Content Provider

Internal Editor

External Editor

ML Data Analysis

LINE TODAY Architecture

CDN 96%

4%

Object Storage

Cache Server

NUXTJS

Frontend Server

In Mem Cache

spring boot

Backend Server

APACHE HBASE

mongoDB

redis

elastic

kafka

Data Warehouse

spring boot

Mini Service

Third Party API

spring boot

Feeding Service

Content Provider

React

Internal CMS

Internal Editor

Vue.js

External CMS

External Editor

python

Report Service

ML Data Analysis

LINE TODAY Architecture

CDN

Object Storage

Cache Server

NUXTJS

Frontend Server

In Mem Cache

spring boot

Backend Server

APACHE HBASE

mongoDB

redis

elastic

kafka

Data Warehouse

spring boot

Mini Service

Third Party API

spring boot

Feeding Service

Content Provider

React

Internal CMS

Internal Editor

Vue.js

External CMS

External Editor

python

Report Service

ML Data Analysis

LINE TODAY Architecture

CDN

Object Storage

Cache Server

NUXTJS

Frontend Server

In Mem Cache

spring boot

Backend Server

APACHE HBASE

mongoDB

redis

elastic

kafka

Data Warehouse

spring boot

Mini Service

Third Party API

spring boot

Feeding Service

Content Provider

React

Internal CMS

Internal Editor

Vue.js

External CMS

External Editor

python

Report Service

ML Data Analysis

LINE TODAY Architecture

CDN

Object Storage

Cache Server

NUXTJS

Frontend Server

In Mem Cache

spring boot

Backend Server

APACHE HBASE

mongoDB

redis

elastic

kafka

Data Warehouse

spring boot

Mini Service

Third Party API

spring boot

Feeding Service

Content Provider

React

Internal CMS

Internal Editor

Vue.js

External CMS

External Editor

python

Report Service

ML Data Analysis

LINE TODAY Architecture

CDN

Object Storage

Cache Server

NUXTJS

Frontend Server

In Mem Cache

spring boot

Backend Server

APACHE HBASE

mongoDB

redis

elastic

kafka

Data Warehouse

spring boot

Mini Service

Third Party API

spring boot

Feeding Service

Content Provider

React

Internal CMS

Internal Editor

Vue.js

External CMS

External Editor

python

Report Service

ML Data Analysis

LINE TODAY Architecture

CDN

Object Storage

Cache Server

NUXTJS

Frontend Server

In Mem Cache

spring boot

Backend Server

APACHE HBASE

mongoDB

redis

elastic

kafka

Data Warehouse

spring boot

Mini Service

Third Party API

spring boot

Feeding Service

Content Provider

React

Internal CMS

Internal Editor

Vue.js

External CMS

External Editor

python

Report Service

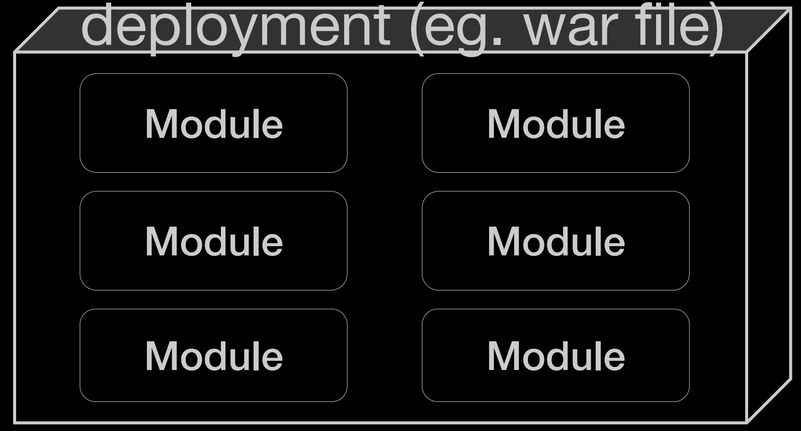
ML Data Analysis

Agenda

- LINE TODAY and its architecture
- How miniservices and K8S changes our development and operation
  - Refactor to mini services
- Lessons learnt

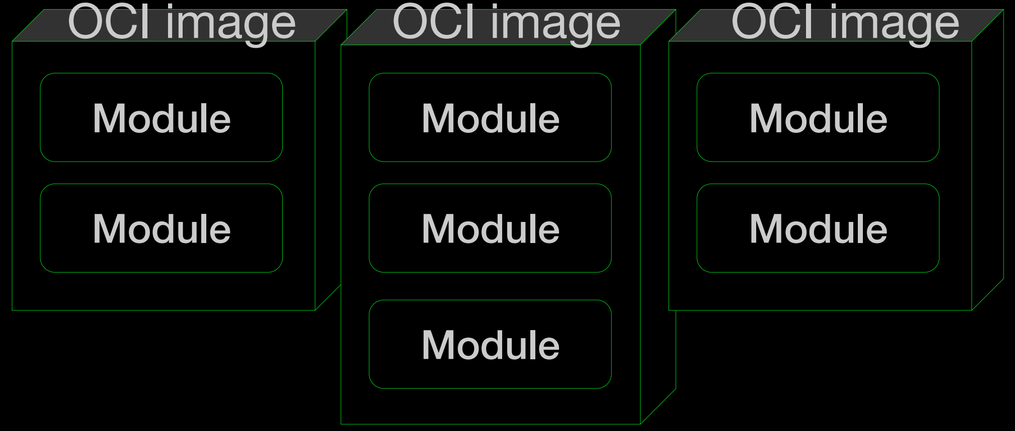
How to improve development and deployment efficiency?

Problem | Small change requires entire system rebuild and deployment



Solutions | - Break down coarse-grained deployments into functionally cohesive mini services

- Move to Kubernetes



Migrate to mini services and Kubernetes

CD

Web Server (VM)

API Server (VM)

Kubernetes

Cache Server

Frontend Server

Ingress Controller

Article Service

Interaction Service

Subscription Service

Mini Services

logs
metrics
tracing

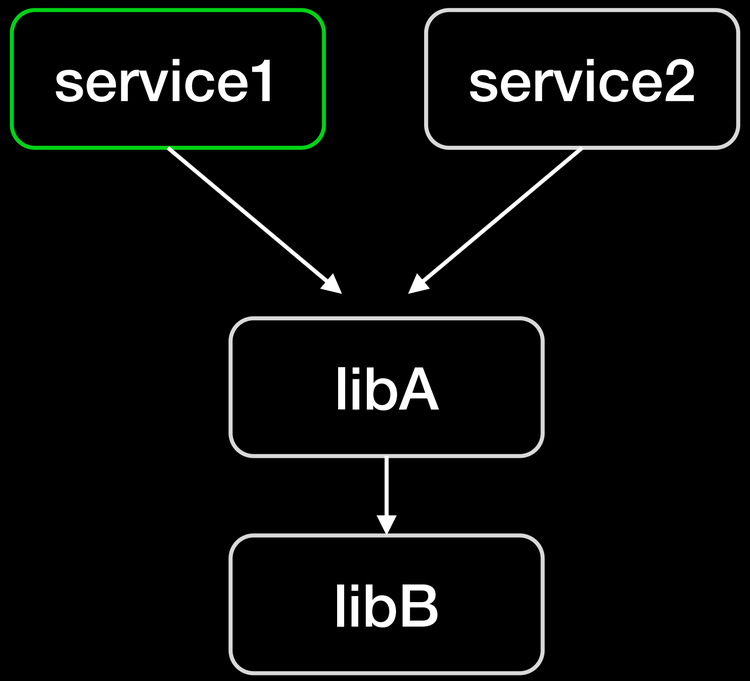
Observability

Agenda

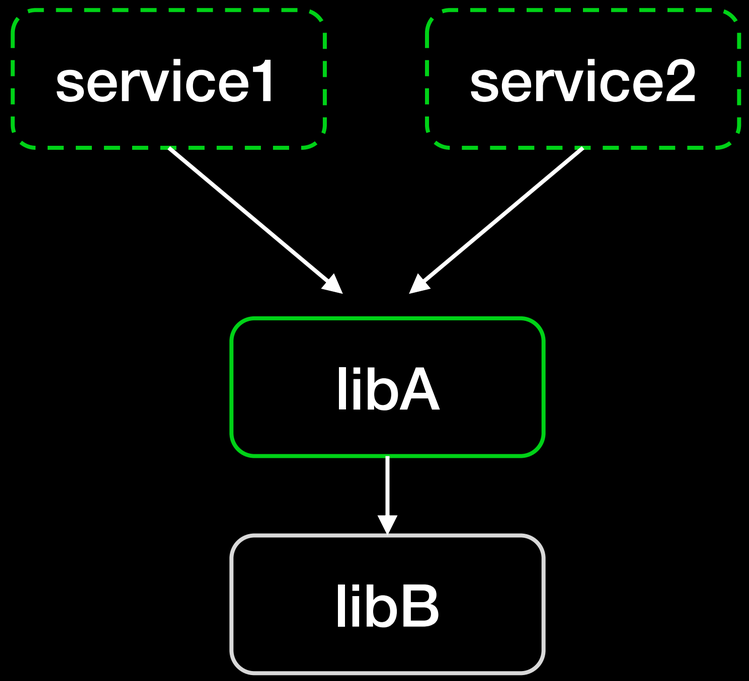
- LINE TODAY and its architecture
- How miniservices and K8S changes our development and operation
  - Refactor to mini services
  - Refine CI/CD
- Lessons learnt

Build and test what was changed

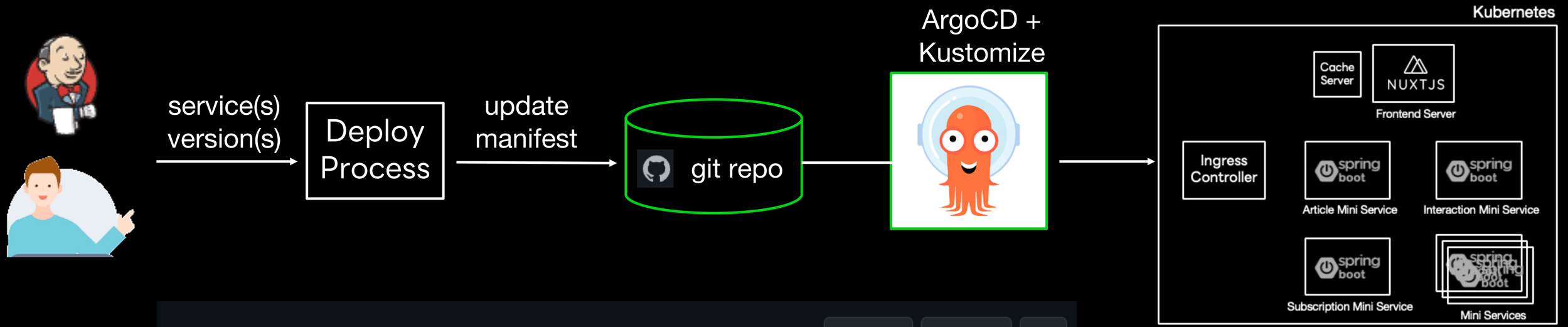
Changing service1
=> rebuild service1

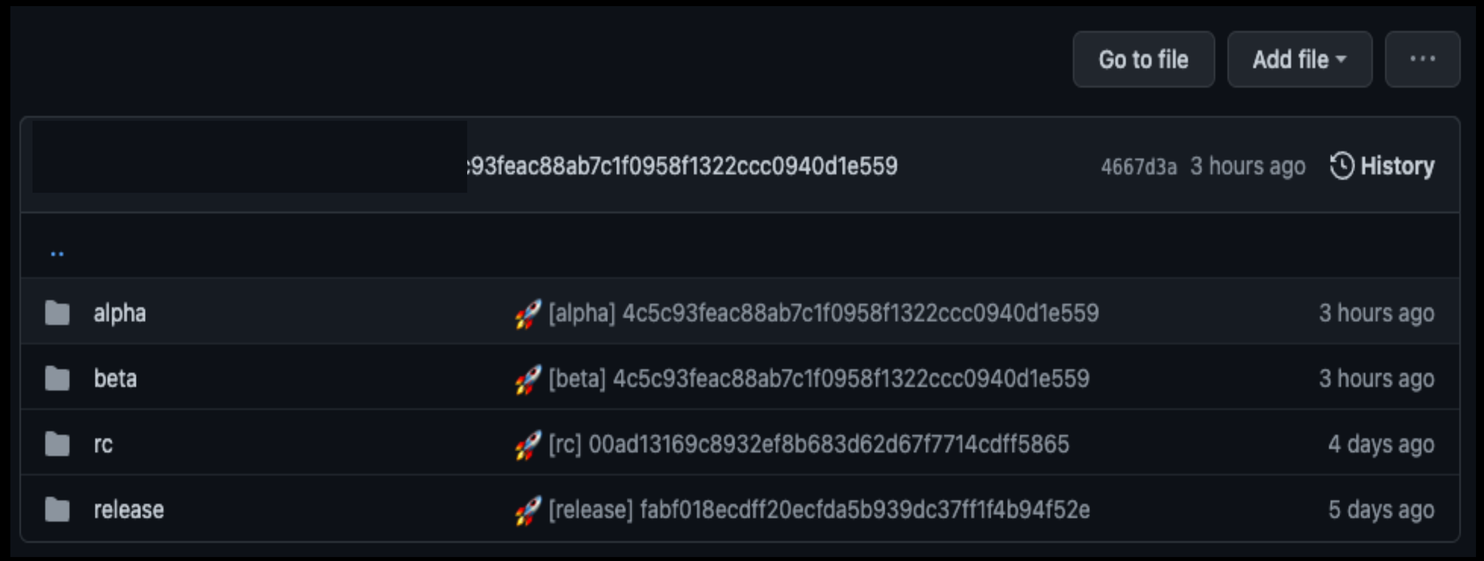


Changing libA => rebuild
libA, service1, service2



Manage deployment via GitOps





Agenda

- LINE TODAY and its architecture
- How miniservices and K8S changes our development and operation
  - Refactor to mini services
  - Refine CI/CD
  - Integrate with observability
- Lessons learnt

Observability improves operation efficiencies

CD

Web Server (VM)

API Server (VM)

Kubernetes

Cache Server

NUXTJS

Frontend Server

Ingress Controller

Article Mini Service

Interaction Mini Service

Subscription Mini Service

Mini Services

logs
metrics
tracing

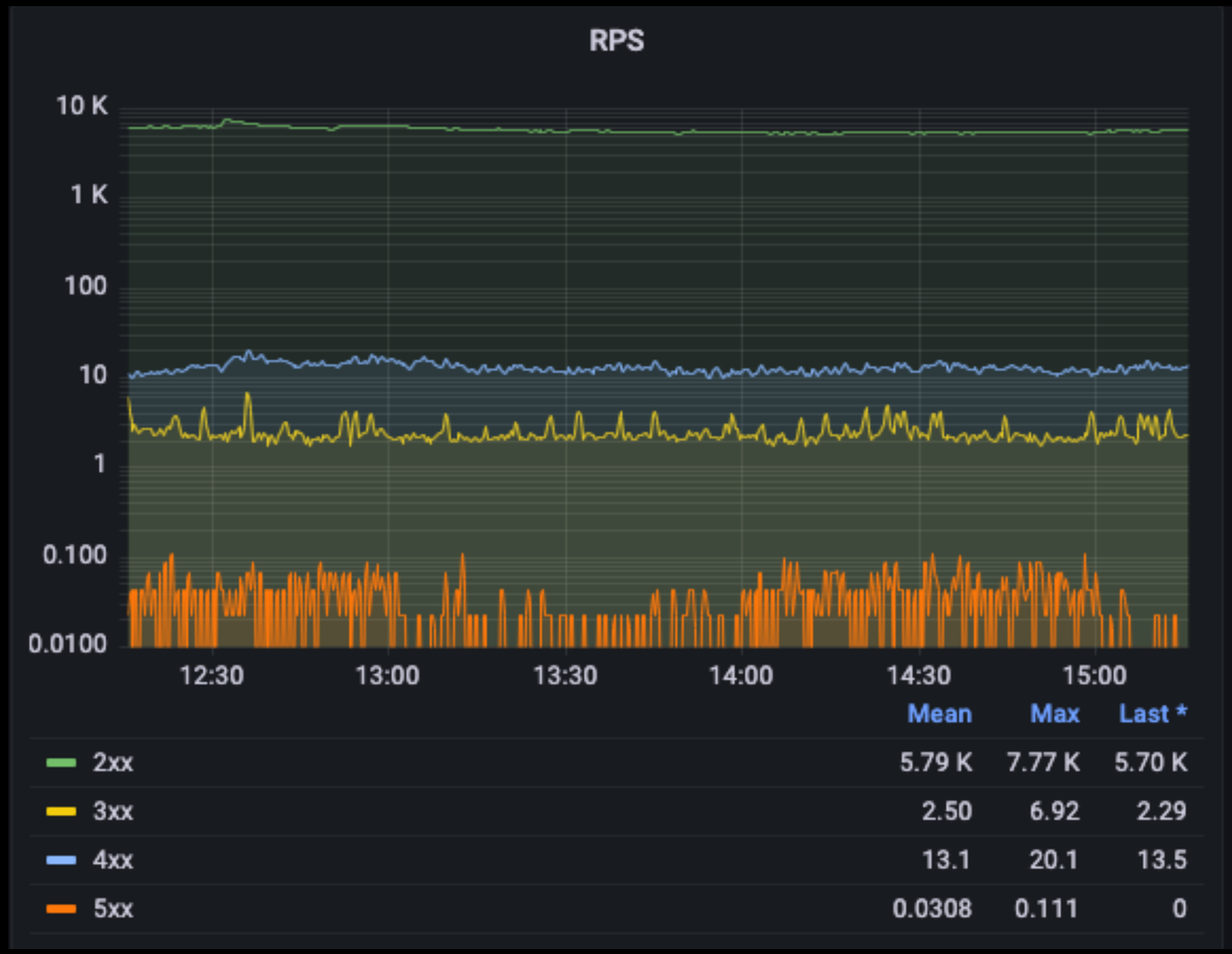
Observability

Grafana

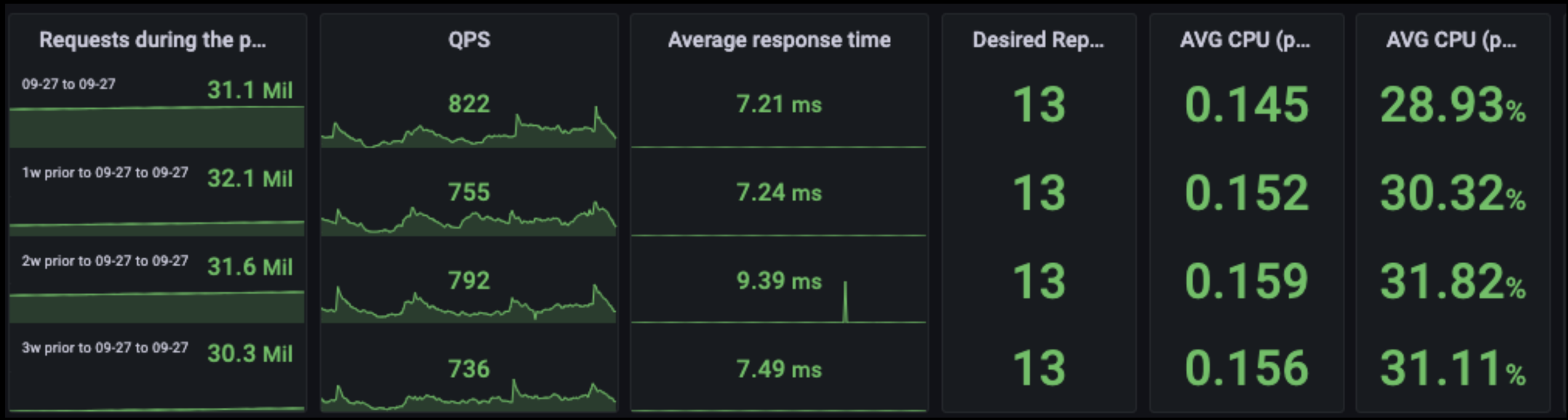
Grafana loki

Thanos

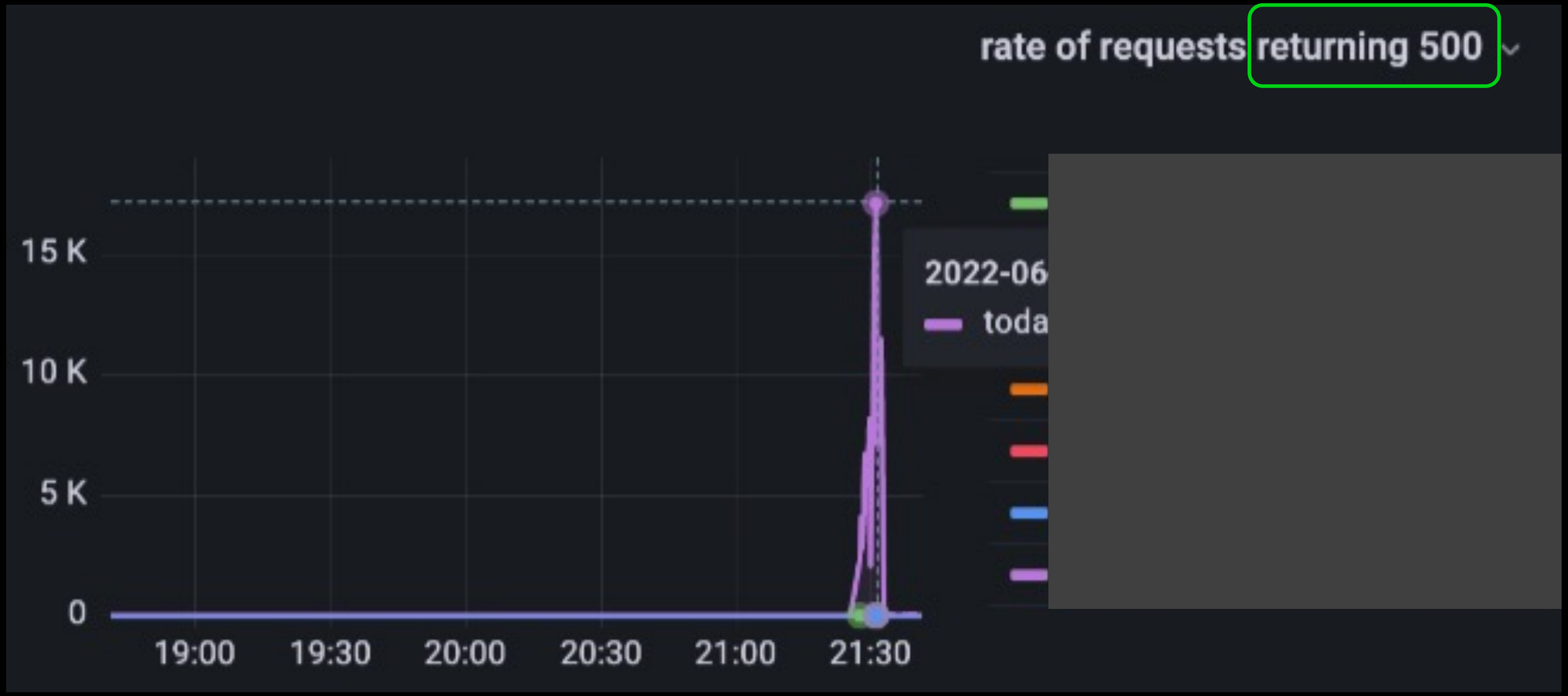
Observability - service RPS / latency metrics



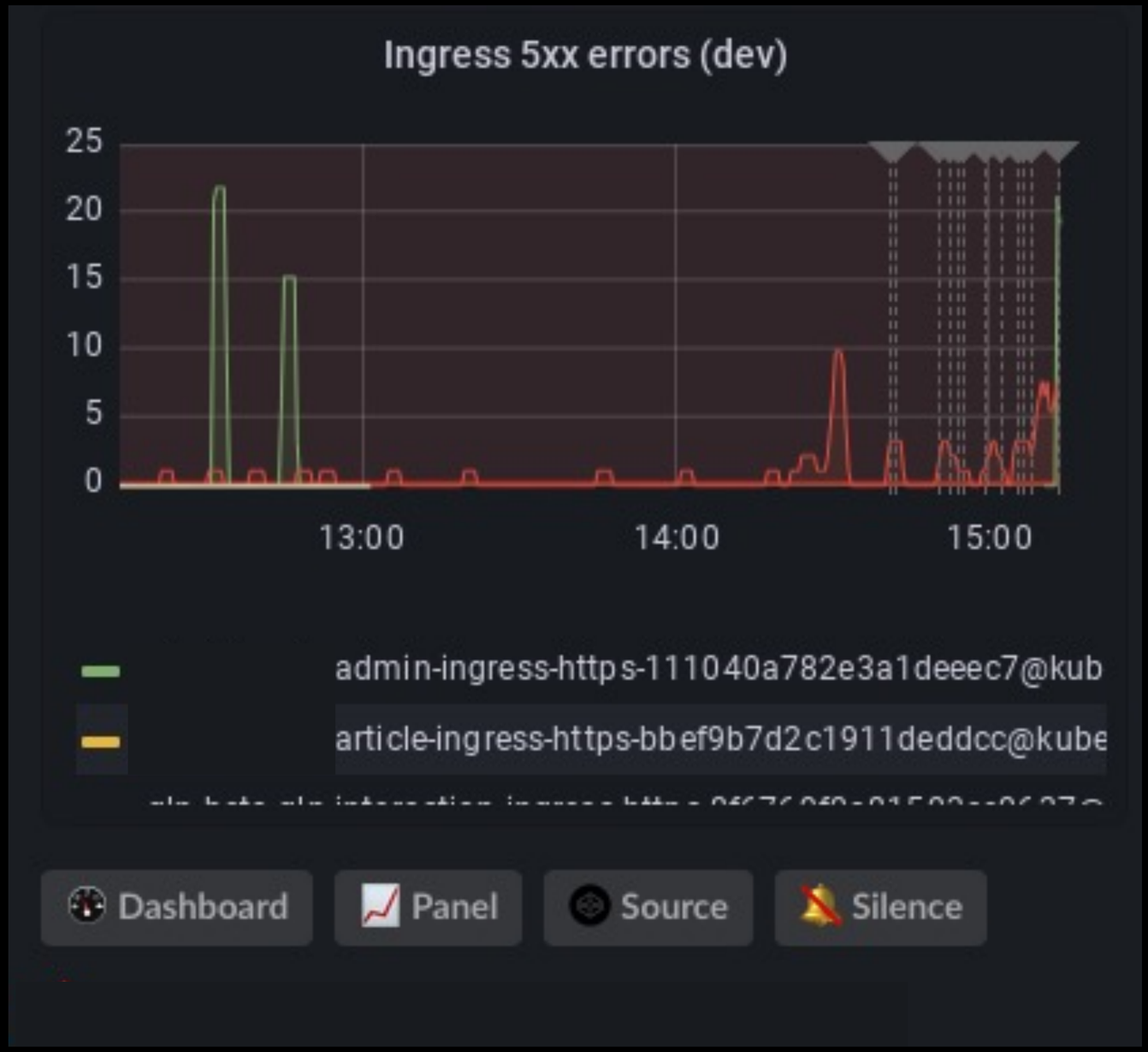
Observability - metrics week-over-week



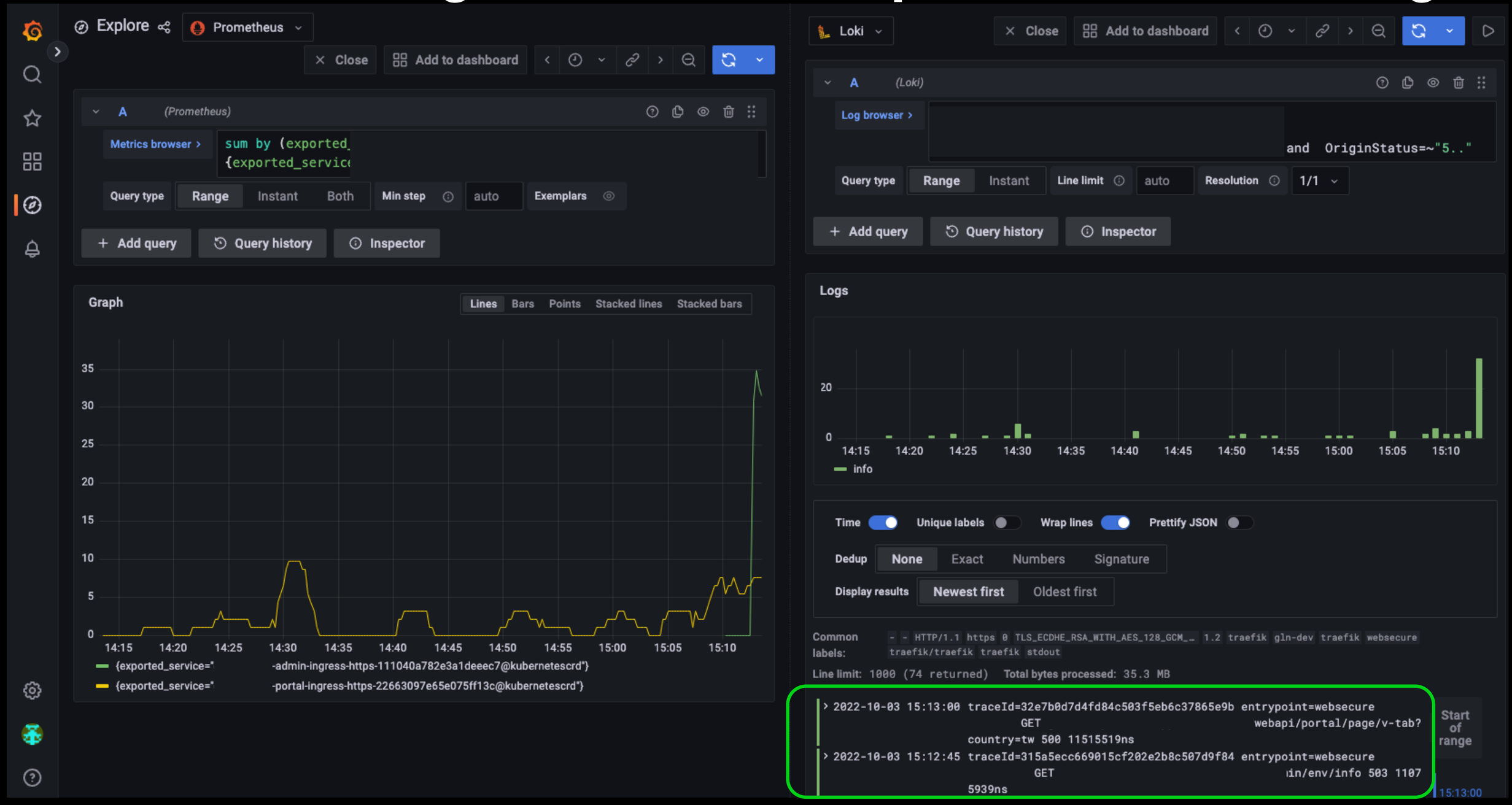
Observability - abnormal spikes



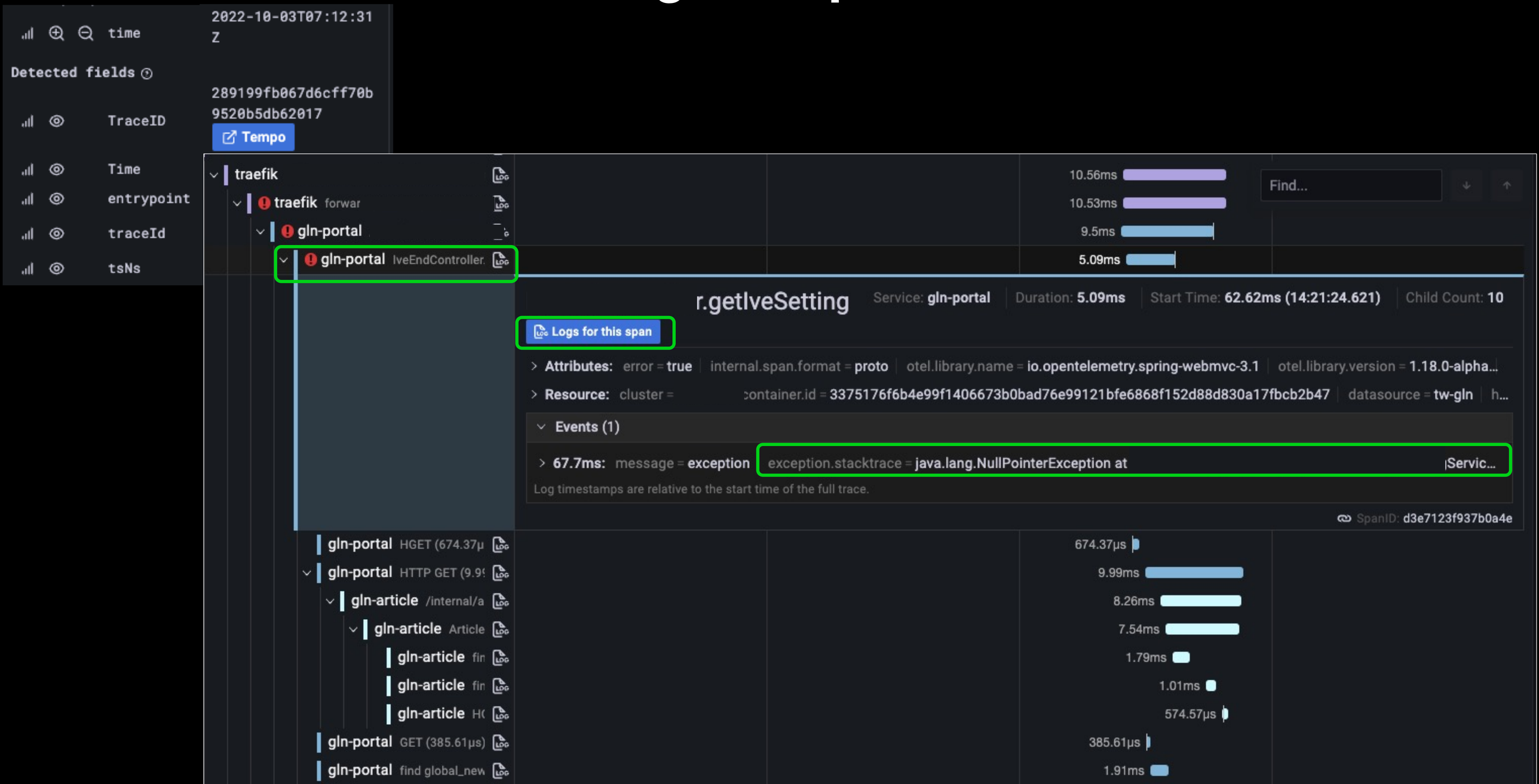
Troubleshooting - 1. receive alert from slack



Troubleshooting - 2. link to alert panel and access logs

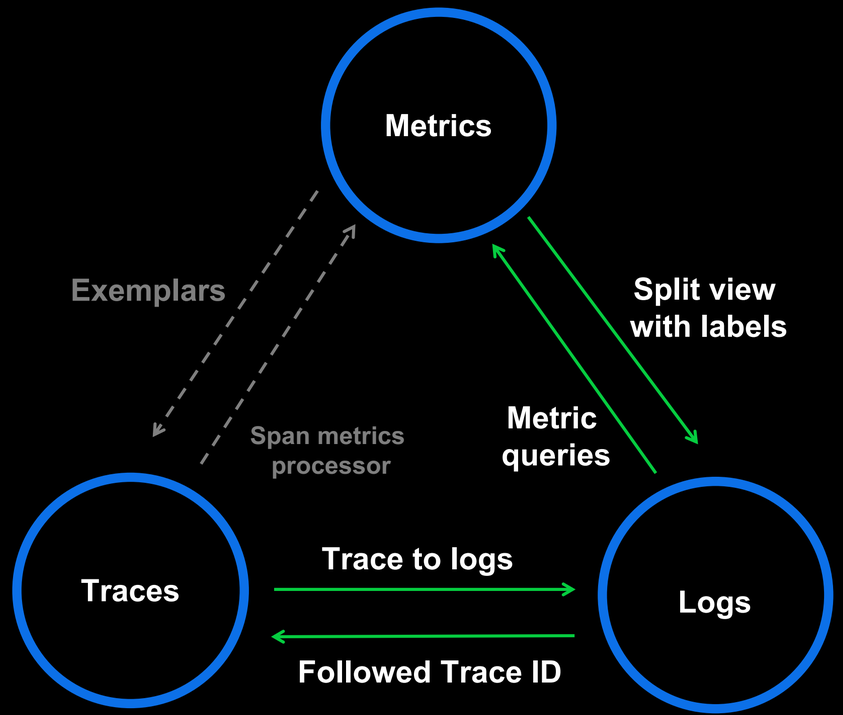


Troubleshooting - 3. open trace viewer



Troubleshooting via observability

Alerts	Metrics	Logs	Traces	Logs
Received alerts from slack	Check error source and time period	Inspect access logs	Open trace viewer	Jump to service logs of the trace



Agenda

- LINE TODAY and its architecture
- How miniservices and K8S changes our development and operation
  - Refactor to mini services
  - Refine CI/CD
  - Integrate with observability
  - Leverage K8S CronJob
- Lessons learnt

Run periodic simple tasks (java) on Kubernetes

- Requirements
  - run at the specific time / interval
  - simple
  - concurrency control
  - running history and logs
  - monitor
  - easy to run in local and test env
- Options
  - Spring @scheduled
  - Quartz
  - Spring Cloud Data Flow
  - AirFlow
  - K8S CronJob

Use K8S CronJob to run periodic tasks

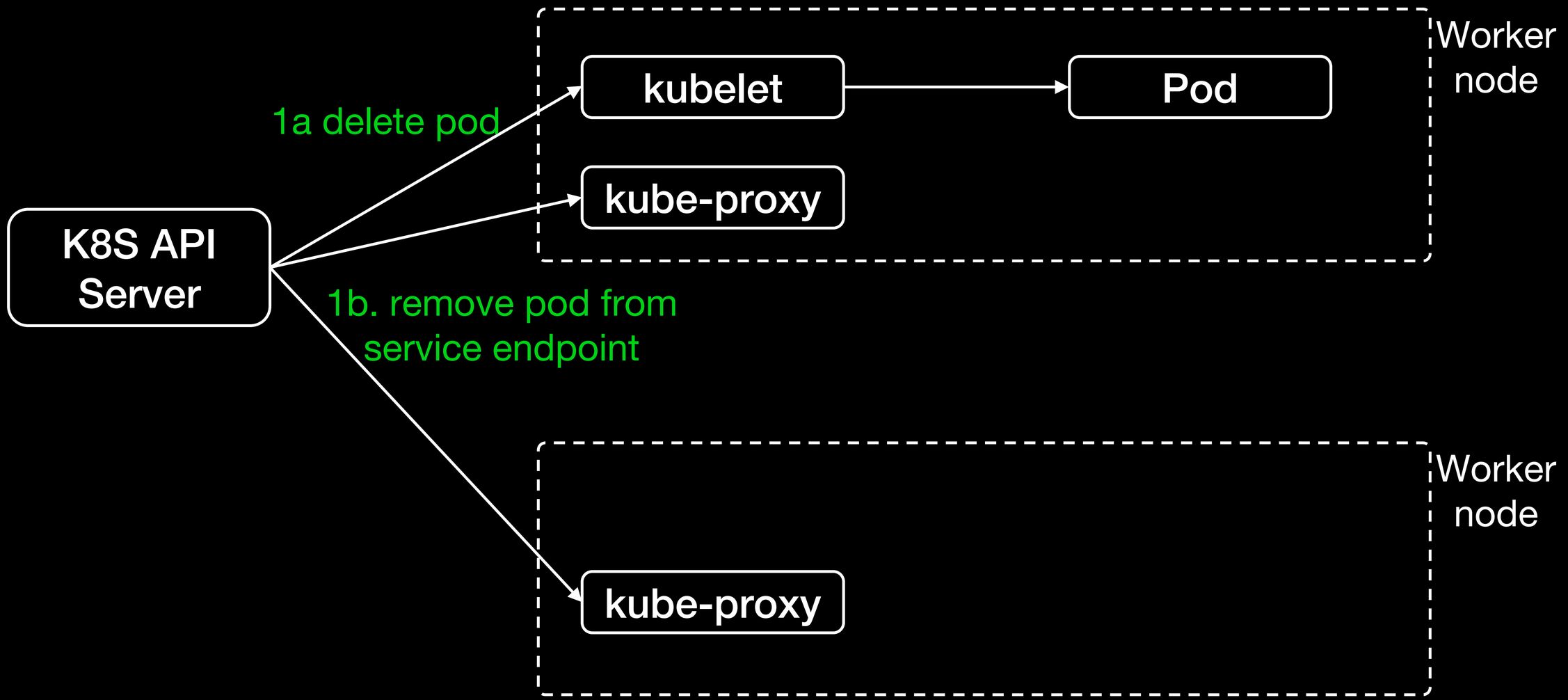
```yaml
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: helloworld
spec:
  schedule: "* * * * *"
  successfulJobsHistoryLimit: 100
  failedJobsHistoryLimit: 100
  concurrencyPolicy: Forbid
  suspend: false
  jobTemplate:
    spec:
      template:
        spec:
          containers:
            - name: helloworld
              image: docker-registry.linecorp.com/line-tai
```

K8S CronJob metrics

Agenda

- LINE TODAY and its architecture
- How miniservices and K8S changes our development and operation
- Lessons learnt

Issue - intermittent errors during rolling update



Solution - graceful shutdown

deployment manifest

```
lifecycle:
  preStop:
    exec:
      command:
      - sh
      - -c
      - sleep 10
```

spring boot application.yaml

```
server:
  shutdown: graceful
```

- Existing services allowed to complete
- No new requests permitted

Container killed (if running)

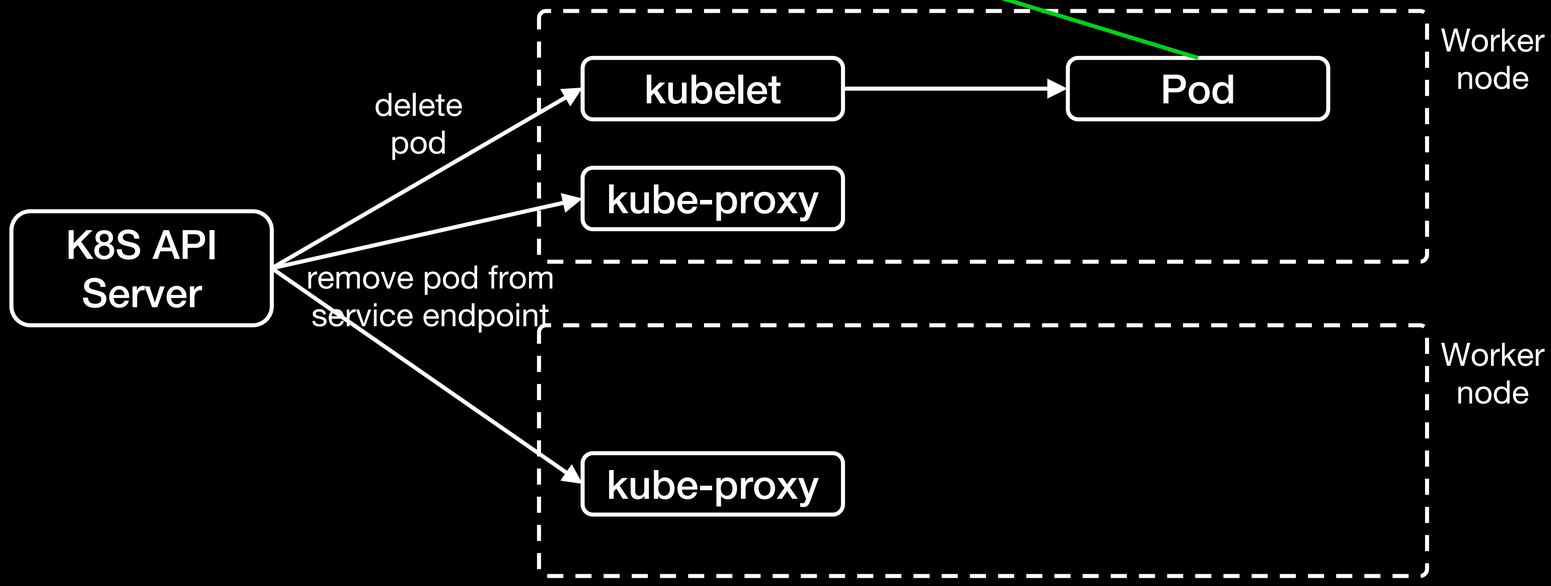
Main container process

Pre-stop hook

Container shutdown

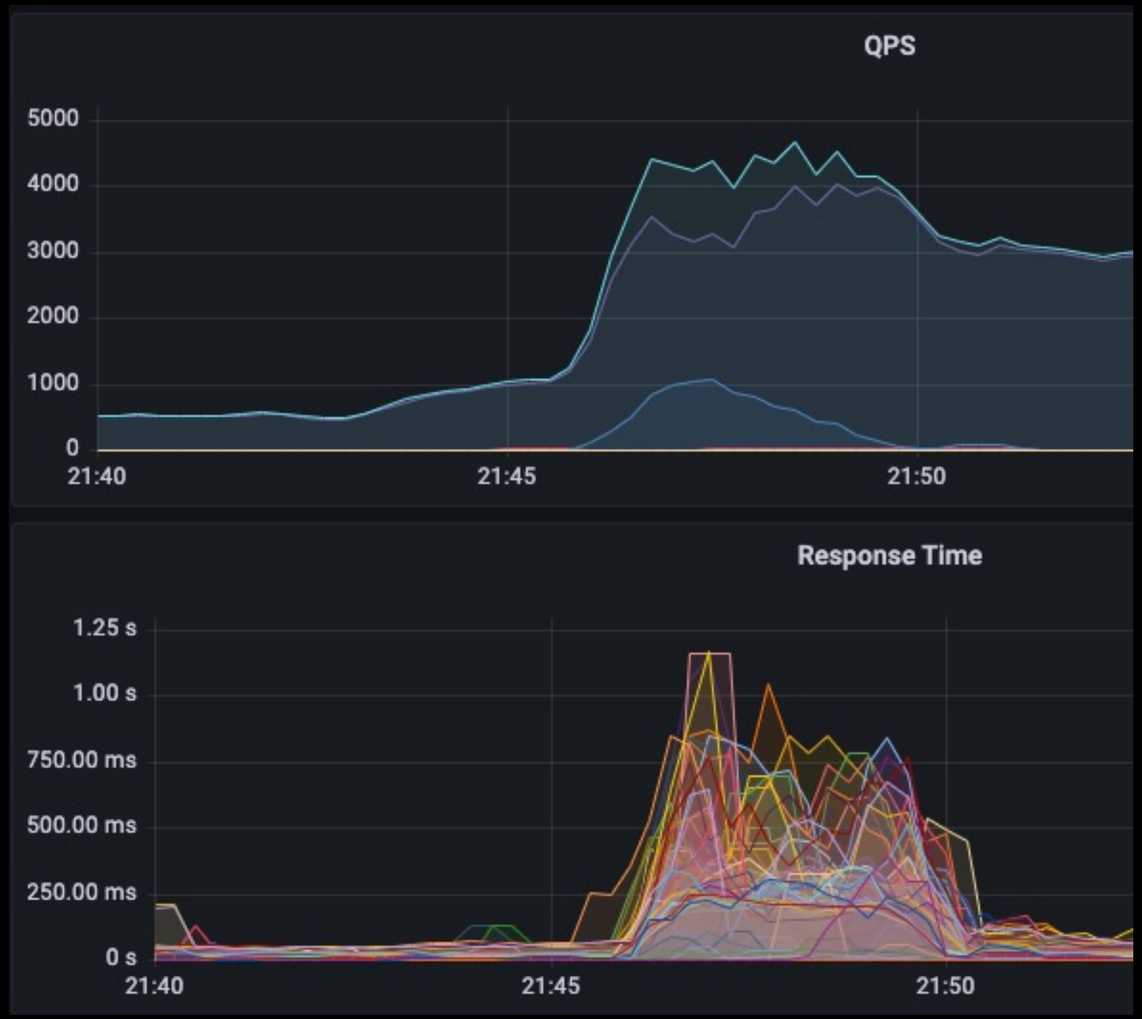
SIGTERM

SIGKILL



Issue - unpredictable request spikes

```yaml
readinessProbe:
  httpGet:
    port: 8080
    path: /health/readiness
  periodSeconds: 10
  timeoutSeconds: 1
  failureThreshold: 3
livenessProbe:
  httpGet:
    port: 8080
    path: /health/liveness
  periodSeconds: 20
  timeoutSeconds: 1
  failureThreshold: 3
```



- pod removed from endpoint at 30ish seconds
- fewer pods available to serve requests
- requeusts queue up
- pod restarted at 60ish seconds
- downward spiral

Options to handle request spikes - it depends

- Overprovision
  - $$$$
- Auto scaling
  - pod - 20+ seconds
  - node - ~5 minutes
  - serverless (lambda) - seconds
- Protection via ingress controller / api gateway
  - circuit breaker - 503
  - rate-limit - 429
- Improve design

Issue - job killed without error log

```
11:06:54.293717486Z
11:06:54.294852302Z   .   ____          _            __ _ _
11:06:54.294858393Z  /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
11:06:54.294881263Z ( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
11:06:54.294884615Z  \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
11:06:54.294887441Z   '  |____| .__|_| |_|_| |_\__, | / / / /
11:06:54.29489038Z  =========|_|==============|___/=/_/_/_/
11:06:54.375201034Z  :: Spring Boot ::        (v2.2.4.RELEASE)
11:06:54.375237075Z
11:08:21.1044027Z Killed
```

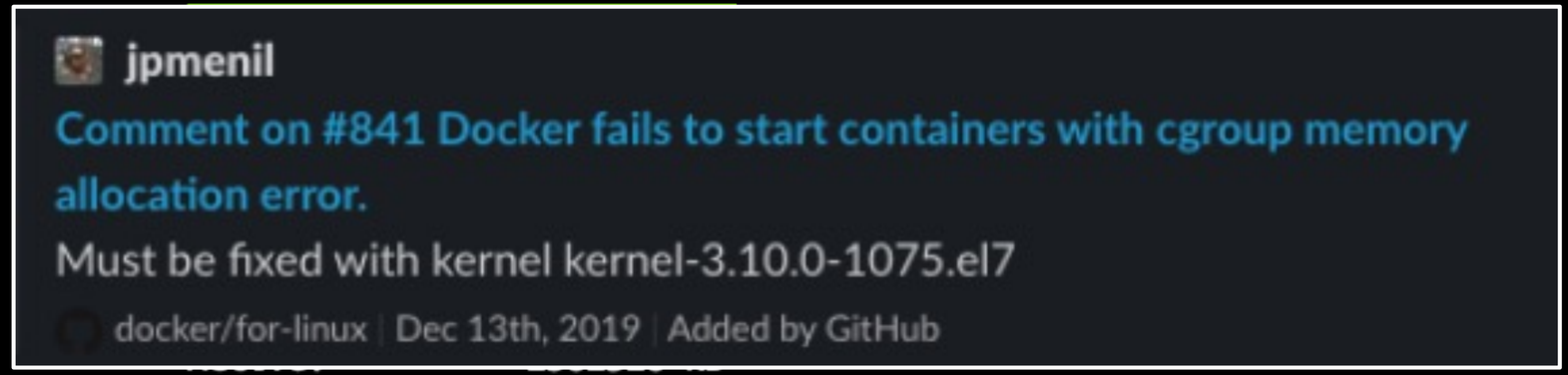
```
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource                                    Requests         Limits
  --------                                    --------         ------
  cpu                                         760m (21%)       2 (55%)
  memory                                      2708516Ki (39%)  4350371072 (61%)
  ephemeral-storage                           0 (0%)           0 (0%)
  hugepages-1Gi                               0 (0%)           0 (0%)
  hugepages-2Mi                               0 (0%)           0 (0%)
  attachable-volumes-csi-cinder.csi.openstack.org  0           0
Events:
  Type     Reason     Age                    From                    Message
  ----     ------     ----                   ----                    -------
  Warning  SystemOOM  4m1s (x4452 over 3d10h)  kubelet,          w8  System OOM encountered
```

Root cause: linux kernel memory leak

```
# cat /proc/meminfo
MemTotal:       8008704 kB
MemFree:         126904 kB
MemAvailable:   2402932 kB
Buffers:           60 kB
Cached:         12328 kB
SwapCached:         0 kB
Active:        900820 kB
Inactive:        9508 kB
Active(anon):   900432 kB
Inactive(anon):    336 kB
Active(file):      388 kB
Inactive(file):   9172 kB
Unevictable:        0 kB
Mlocked:            0 kB
SwapTotal:          0 kB
SwapFree:           0 kB
Dirty:              0 kB
Writeback:          0 kB
AnonPages:      898288 kB
Mapped:           3672 kB
Shmem:            2480 kB
Slab:          6750240 kB
SReclaimable:  2492932 kB
SUnreclaim:    4257308 kB
KernelStack:     16336 kB
PageTables:      11608 kB
```

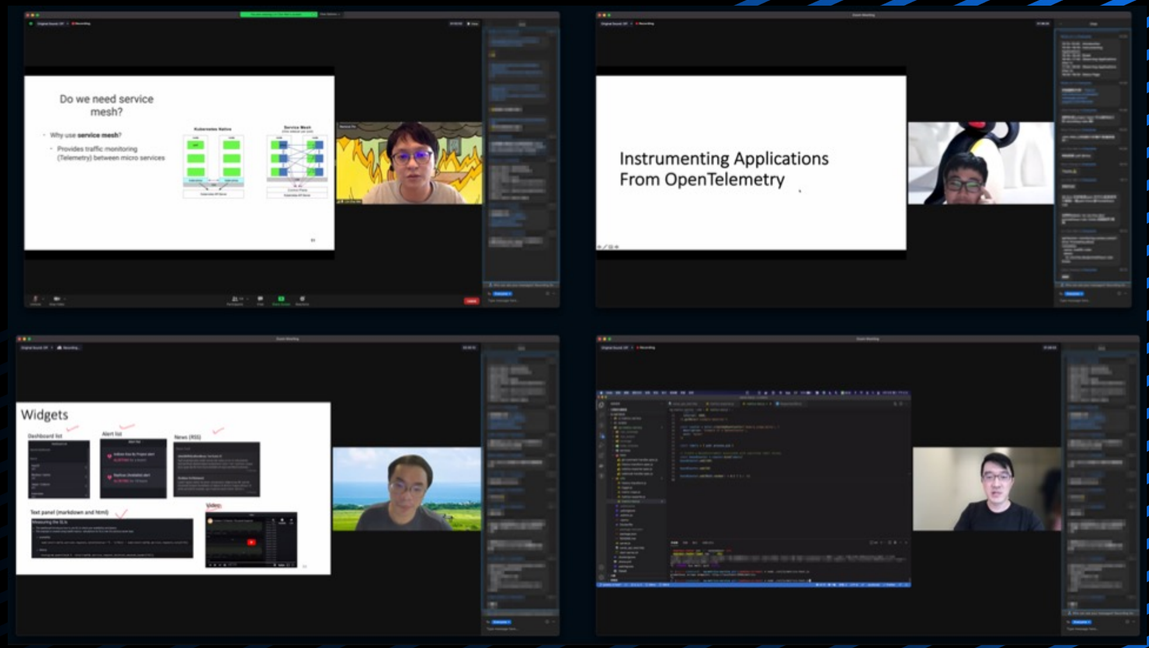
reboot

```
Inactive:        998908 kB
Active(anon):   1121036 kB
Inactive(anon):     340 kB
Active(file):    440280 kB
Inactive(file):  998568 kB
Unevictable:          0 kB
Mlocked:              0 kB
SwapTotal:            0 kB
SwapFree:             0 kB
Dirty:                8 kB
Writeback:            0 kB
AnonPages:      1119600 kB
Mapped:          218276 kB
Shmem:             1784 kB
Slab:            136192 kB
SReclaimable:     80400 kB
SUnreclaim:       55792 kB
KernelStack:      11520 kB
```



Summary

- LINE TODAY and its architecture
- Mini services and K8S helps dev / ops efficiency for large systems
  - Refactor to mini services
  - Refine CI/CD
  - Integrate with observability
  - Leverage K8S CronJob
- Build in-depth DevOps and K8S skills



Thank you