THE SHAPE OF DATA

What kind of monitoring data should your service send?

July 24, 2024

LANDSCAPE DOCUMENTATION

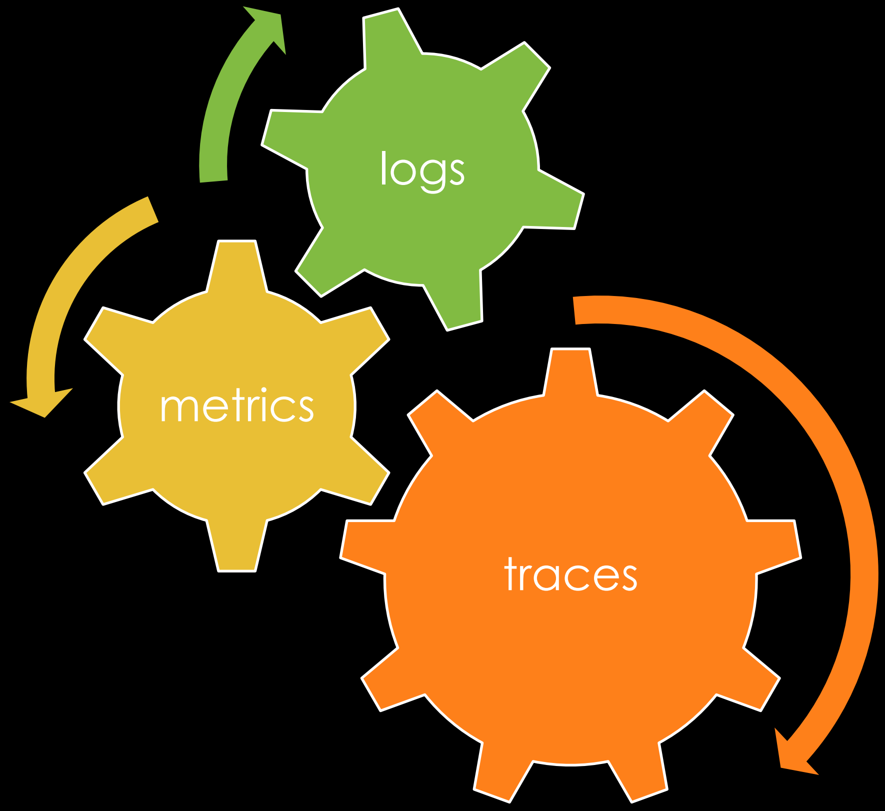
https://landscape.fnal.gov/docs/

Connection details and examples
All the information and links that are in this talk.

SSO Required

MONITORING

- Monitoring or *observability* is a broad subject, which involves the collection, storing, and querying/visualizing of three main types of data:
  - logs
  - metrics
  - traces
- These work together to provide the developer and operator with a complete picture and support rapid responses



LOGGING

- Logging is the simplest form of monitoring, which begins the moment a programmer writes their first “Hello, world!” program. Logs describe in detail what an application is doing (or not doing sometimes).
- In monitoring terms, most logs are really <u>events</u>:
  - **what** happened
  - **when** it happened
  - **why** it happened
  - **who/what** caused the event
  - **what type** of event/severity
- **DO NOT LOG SECRETS OR SENSITIVE INFORMATION**
  - “but I just want to quickly see…” **NO**

STRUCTURED LOGGING

```
time="2021-03-09T21:55:27Z" level=info msg="handled request"
duration=453.906118ms length=0 method=GET origin=71.57.54.226
path=/job/42595777.0@jobsub02.fnal.gov/ traceid=296731105f77ab6c
```

- Use a structured logging library, e.g. structlog for Python and logrus for Go
- Use a standard, easily parsed format like logfmt or json. Custom parsers (e.g. grok) are fragile.
- Use ISO8601/RFC3339 timestamps, with timezone (or UTC)
- Try to keep related info in a single log event, and/or provide some way to connect events (e.g. a unique trace ID)
- Exception: debug/trace logs, can become unwieldy and are only useful for someone looking at them in context. Generally, should be disabled in production.

COLLECTING LOGS

- Application should write logs to stdout/stderr or a file. Collect logs with **filebeat**, **promtail**, or a docker plugin.
  - Send logs to **Kafka** if they were collected by filebeat (or logstash), are well-structured, and contain fields that you'll want to do analytics on. These will be put into **Elasticsearch.**
  - Send logs to **Loki** if they are mainly for troubleshooting. These will be immediately viewable in **Grafana**.
- Sometimes you may want to publish key events separate from other logs, you can send these to **Kafka**, **RabbitMQ** (via AMQP or STOMP protocols), or the **Ingest** service.

METRICS

- While logs describe the details of what an application is doing, metrics provide aggregate high-level insight into the process, collected at some regular frequency to allow for trending and anomaly identification
- **Internal** metrics are collected within the process and expose the internal state.
- **External**, or blackbox, metrics are collected by some other service, and reflect the status from the perspective of a user or client
- Metrics may be published via **push**, where the service sends the metrics directly to the monitoring service, e.g. **Graphite**, **Statsd**
- … or by **pull**, where the monitoring service “scrapes” the metrics, typically over HTTP, e.g. **Prometheus**

PROMETHEUS

- Prometheus is a service for collecting, storing, and querying system and service metrics. It has a standard exposition format (which as evolved into the standard OpenMetrics) and a rich ecosystem of libraries for publishing metrics.
  - Libraries typically include HTTP server, if service does not already have one or wants to publish metrics on separate interface or port
- Some cases where the pull-based model used by Prometheus is not appropriate, e.g. short-lived processes, services behind firewall
  - push metrics to the Prometheus Pushgateway
  - Collect metrics with local Prometheus and remote_write to Mimir
  - Push metrics in document format to **Kafka** or **Ingest**

TRACING

- Distributed tracing provides a way to connect events across services, processes, or threads.
- It helps operators and developers determine *where* and *why* failures occur, current process states, and identify potential bottlenecks.
- A trace is typically instantiated by a single end-user request, and is then propagated through the services associated with servicing that request.
    - If relevant logs include the Trace ID, Grafana can correlate them.
    - Your service should always look for a parent trace ID in however it gets called (there are standards, and libraries implement them)
- Landscape runs the Jaeger service behind an OpenTelemetry collector to collect OTLP traces over HTTP and makes them viewable in the **Jaeger** frontend and in **Grafana**.
    - OTLP also has some support for metrics and logs, they will be routed to Mimir and Loki, respectively.

SUMMARY

- Collect and publish general service logs to Loki or Elasticsearch (through Kafka)
- Collect and publish internal service and process metrics with Prometheus/OpenTelemetry library.
- Publish key event data, possibly with metrics, in JSON to Kafka. Set up digest processes to enrich and/or summarize the data if necessary.
- For distributed systems, use OpenTelemetry to publish tracing data. Accept a parent trace ID in however your service gets called and pass one to whatever you call.
- Monitor user-facing services with Blackbox monitoring.
- Set up alerts in Grafana to notify operations of unexpected conditions.

DEMO TIME!

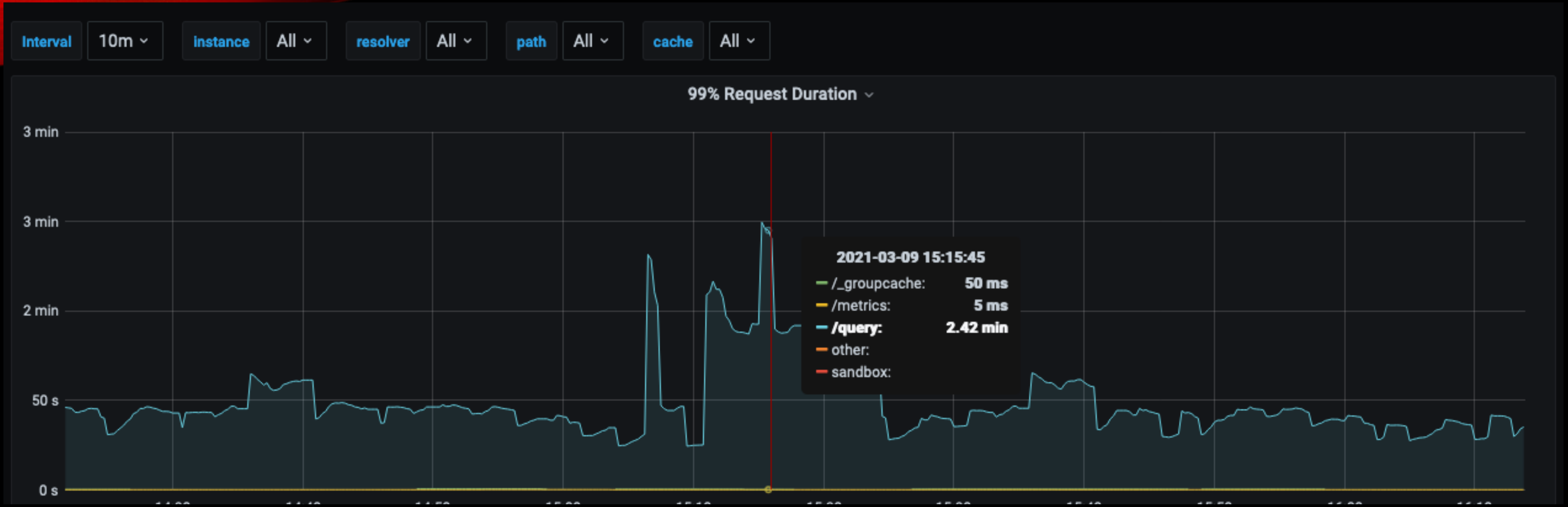
Metrics, Logging and Tracing with Prometheus, Loki, and Jaeger

Case Study: **Lens** API

LENS METRICS

- HTTP request metrics published with Prometheus Go library: https://landscape.fnal.gov/lens/metrics

```
http_request_duration_seconds_bucket{path="/query",le="0.005"} 50948
http_request_duration_seconds_bucket{path="/query",le="0.01"} 51110
http_request_duration_seconds_bucket{path="/query",le="0.025"} 51327
http_request_duration_seconds_bucket{path="/query",le="0.05"} 55991
http_request_duration_seconds_bucket{path="/query",le="0.1"} 58578
http_request_duration_seconds_bucket{path="/query",le="0.25"} 64500
http_request_duration_seconds_bucket{path="/query",le="0.5"} 99190
http_request_duration_seconds_bucket{path="/query",le="1"} 101526
http_request_duration_seconds_bucket{path="/query",le="2.5"} 102930
http_request_duration_seconds_bucket{path="/query",le="5"} 103725
http_request_duration_seconds_bucket{path="/query",le="10"} 104539
http_request_duration_seconds_bucket{path="/query",le="25"} 105082
http_request_duration_seconds_bucket{path="/query",le="50"} 105128
http_request_duration_seconds_bucket{path="/query",le="100"} 105164
http_request_duration_seconds_bucket{path="/query",le="250"} 105164
http_request_duration_seconds_bucket{path="/query",le="+Inf"} 105164
http_request_duration_seconds_sum{path="/query"} 35873.17846150313
http_request_duration_seconds_count{path="/query"} 105164
```



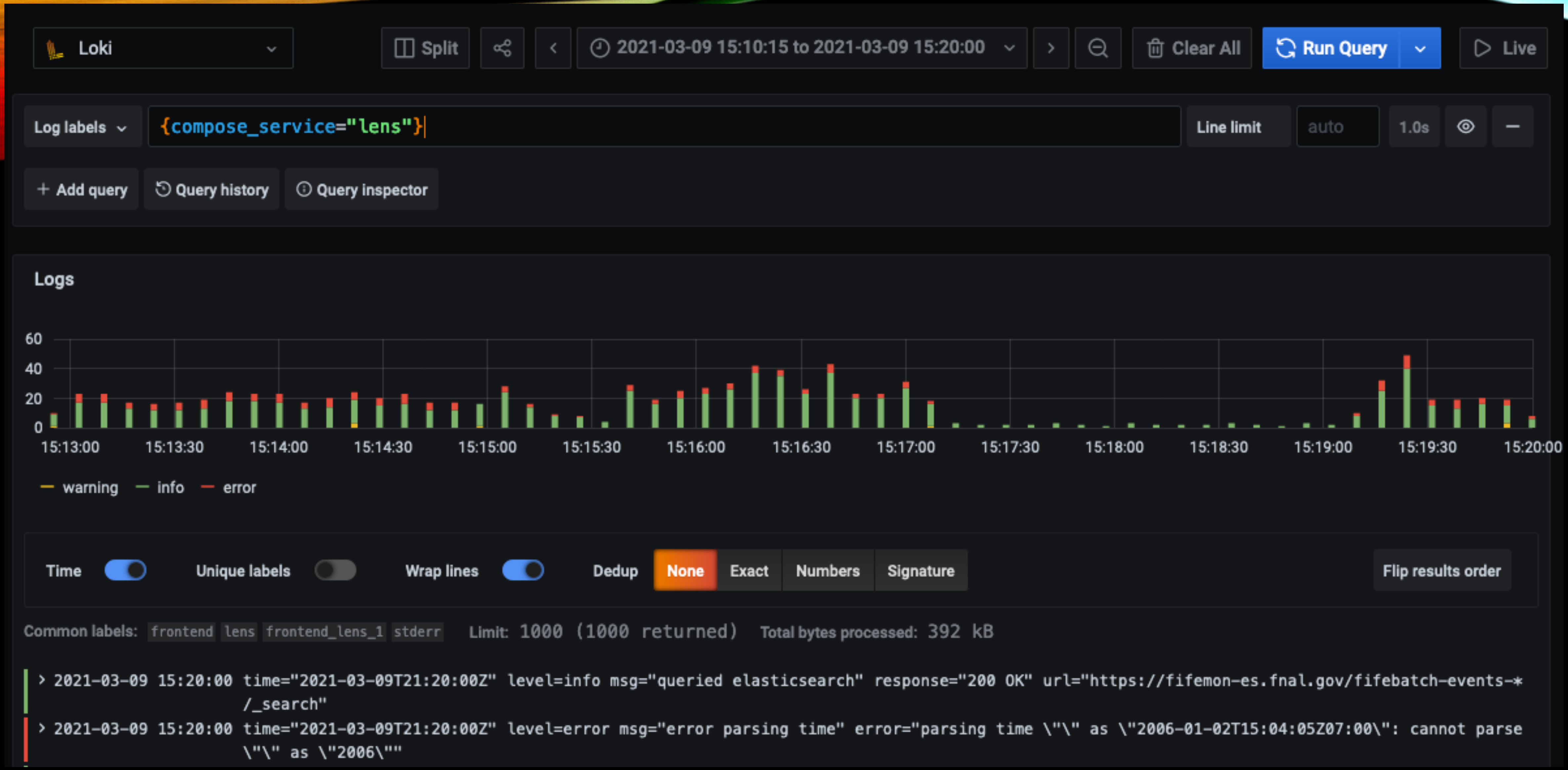
Interval 10m
instance All
resolver All
path All
cache All

2021-03-09 15:15:45
/_groupcache: 50 ms
/metrics: 5 ms
/query: 2.42 min
other:
sandbox:

ALERT! 99TH PERCENTILE THRESHOLD EXCEEDED (GRAFANA)

(not a real alert, we don't guarantee quite that good of a response)

```
histogram_quantile(0.99,rate(http_request_duration_seconds_bucket{job="lens",instance=~"$instance"}[$interval]))
```

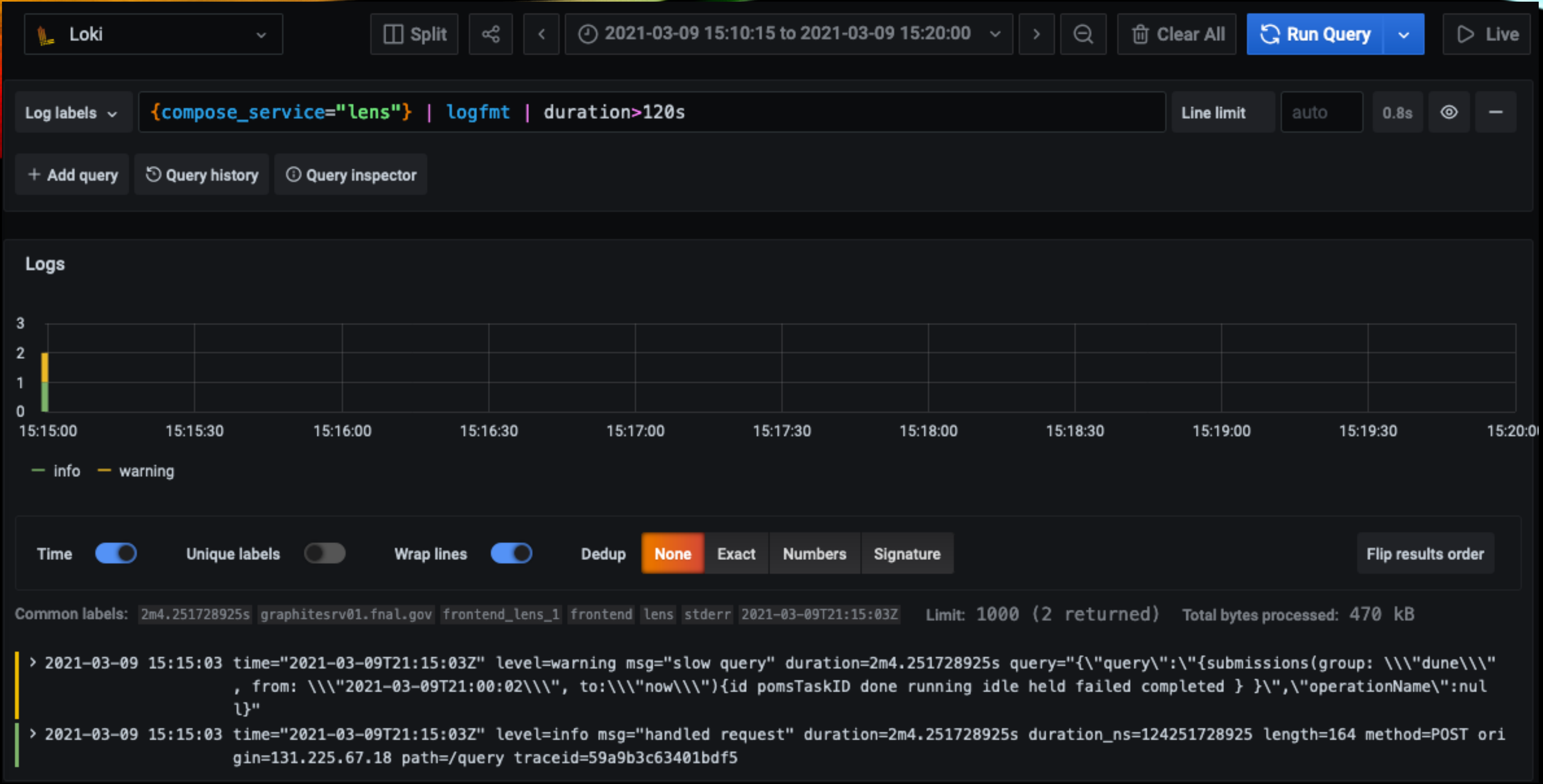


```
{compose_service="lens"}
```

Common labels: frontend lens frontend_lens_1 stderr Limit: 1000 (1000 returned) Total bytes processed: 392 kB

```
> 2021-03-09 15:20:00 time="2021-03-09T21:20:00Z" level=info msg="queried elasticsearch" response="200 OK" url="https://fifemon-es.fnal.gov/fifebatch-events-*
                      /_search"
> 2021-03-09 15:20:00 time="2021-03-09T21:20:00Z" level=error msg="error parsing time" error="parsing time \"\" as \"2006-01-02T15:04:05Z07:00\": cannot parse
                      \"\" as \"2006\""
```

LET'S GO TO THE LOGS (LOKI)

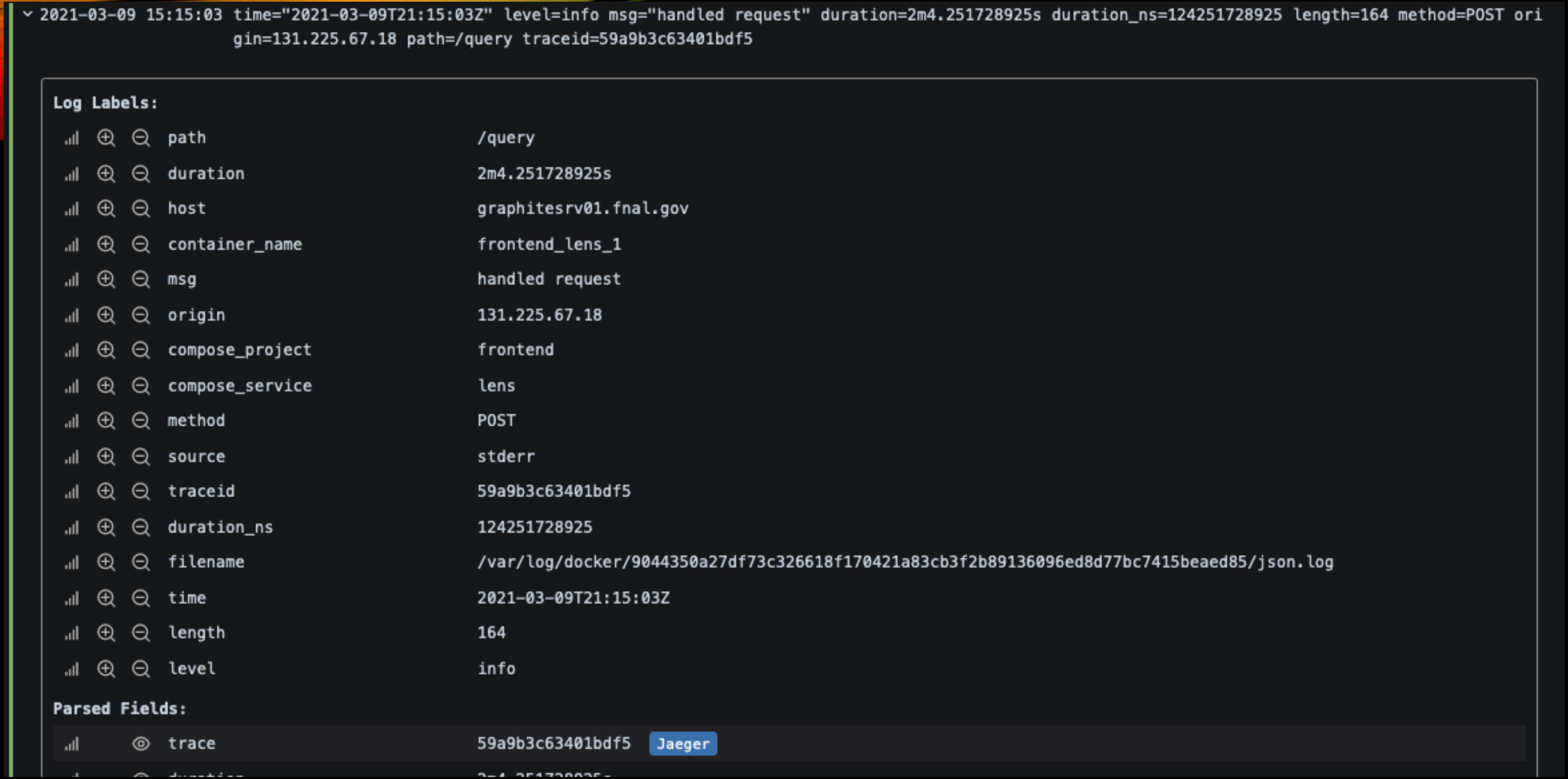


```
{compose_service="lens"} | logfmt | duration>120s
```

Common labels: 2m4.251728925s graphitesrv01.fnal.gov frontend_lens_1 frontend lens stderr 2021-03-09T21:15:03Z Limit: 1000 (2 returned) Total bytes processed: 470 kB

```
> 2021-03-09 15:15:03 time="2021-03-09T21:15:03Z" level=warning msg="slow query" duration=2m4.251728925s query="{\"query\":\"{submissions(group: \\\"dune\\\"
                      , from: \\\"2021-03-09T21:00:02\\\", to:\\\"now\\\"){id pomsTaskID done running idle held failed completed } }\",\"operationName\":nul
                      l}"
> 2021-03-09 15:15:03 time="2021-03-09T21:15:03Z" level=info msg="handled request" duration=2m4.251728925s duration_ns=124251728925 length=164 method=POST ori
                      gin=131.225.67.18 path=/query traceid=59a9b3c63401bdf5
```

FIND THE SLOW QUERY



```
2021-03-09 15:15:03 time="2021-03-09T21:15:03Z" level=info msg="handled request" duration=2m4.251728925s duration_ns=124251728925 length=164 method=POST origin=131.225.67.18 path=/query traceid=59a9b3c63401bdf5
```

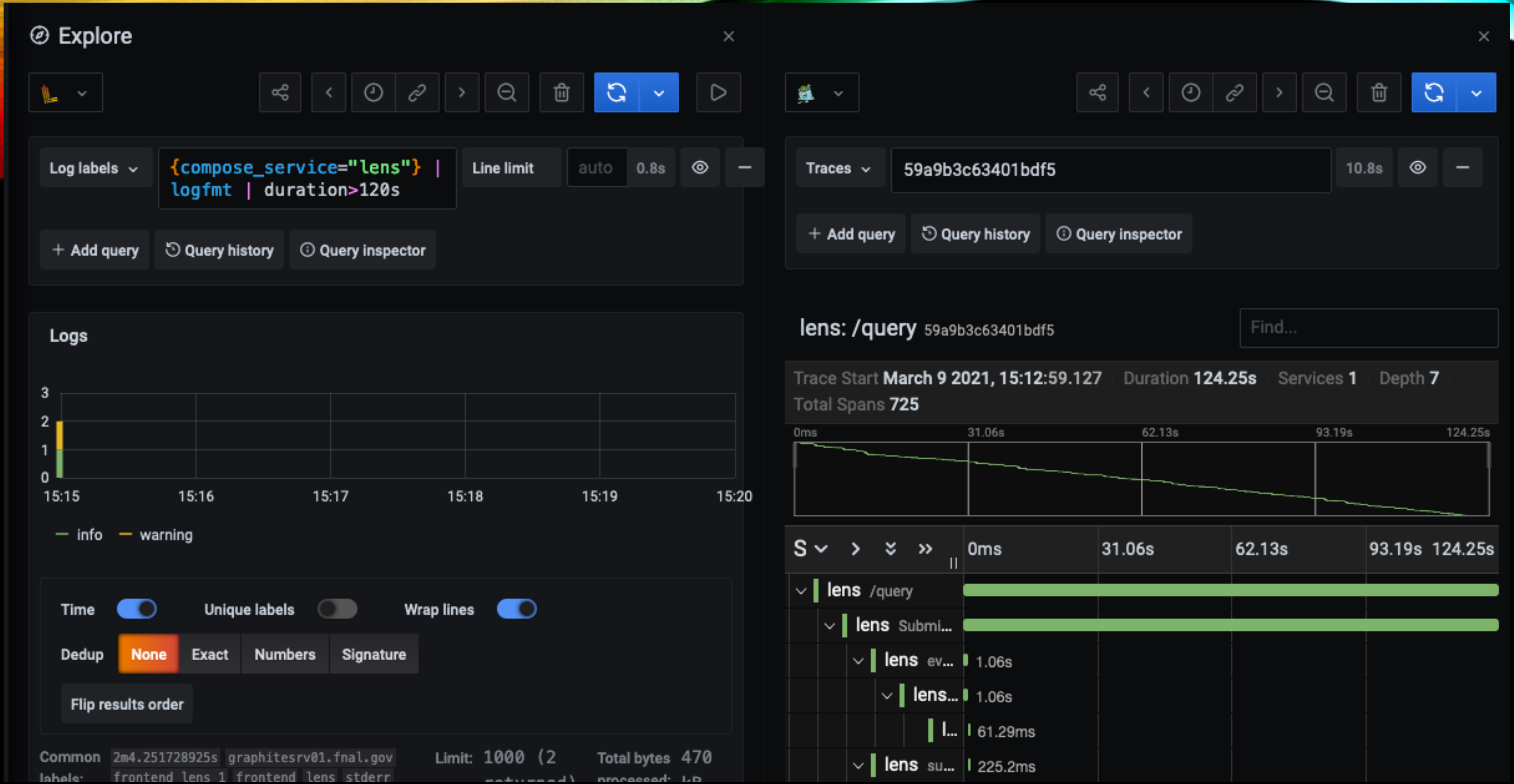
Log Labels:

path	/query
duration	2m4.251728925s
host	graphitesrv01.fnal.gov
container_name	frontend_lens_1
msg	handled request
origin	131.225.67.18
compose_project	frontend
compose_service	lens
method	POST
source	stderr
traceid	59a9b3c63401bdf5
duration_ns	124251728925
filename	/var/log/docker/9044350a27df73c326618f170421a83cb3f2b89136096ed8d77bc7415beaed85/json.log
time	2021-03-09T21:15:03Z
length	164
level	info

Parsed Fields:

trace	59a9b3c63401bdf5 Jaeger

OH LOOK, A TRACE ID!



```
{compose_service="lens"} | logfmt | duration>120s
```

Time ● Unique labels ○ Wrap lines ●

Dedup [✓ None] Exact Numbers Signature

Flip results order

Common labels: 2m4.251728925s graphitesrv01.fnal.gov frontend_lens_1 frontend lens stderr

Limit: 1000 (2 returned)

Total bytes processed: 470 kB

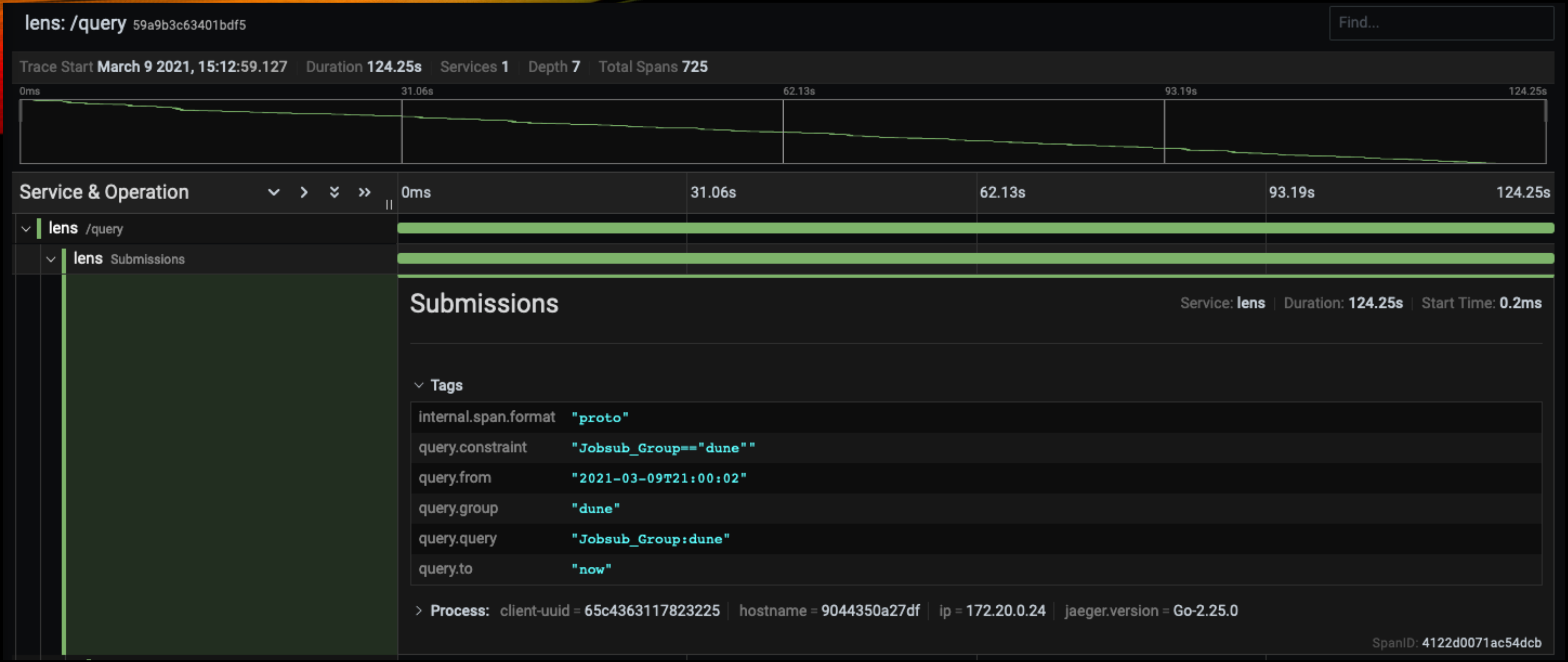
```
59a9b3c63401bdf5
```

lens: /query 59a9b3c63401bdf5

Trace Start **March 9 2021, 15:12:59.127** Duration **124.25s** Services **1** Depth **7**
Total Spans **725**

```
lens /query
  lens Submi...
    lens ev...      1.06s
      lens...       1.06s
        l...        61.29ms
    lens su...      225.2ms
```

LOOK THROUGH THE TRACE (JAEGER)



lens: /query 59a9b3c63401bdf5

Trace Start **March 9 2021, 15:12:59.127** | Duration **124.25s** | Services **1** | Depth **7** | Total Spans **725**

Submissions

Service: **lens** | Duration: **124.25s** | Start Time: **0.2ms**

Tags

internal.span.format	"proto"
query.constraint	"Jobsub_Group=="dune""
query.from	"2021-03-09T21:00:02"
query.group	"dune"
query.query	"Jobsub_Group:dune"
query.to	"now"

Process: client-uuid = 65c4363117823225 | hostname = 9044350a27df | ip = 172.20.0.24 | jaeger.version = Go-2.25.0

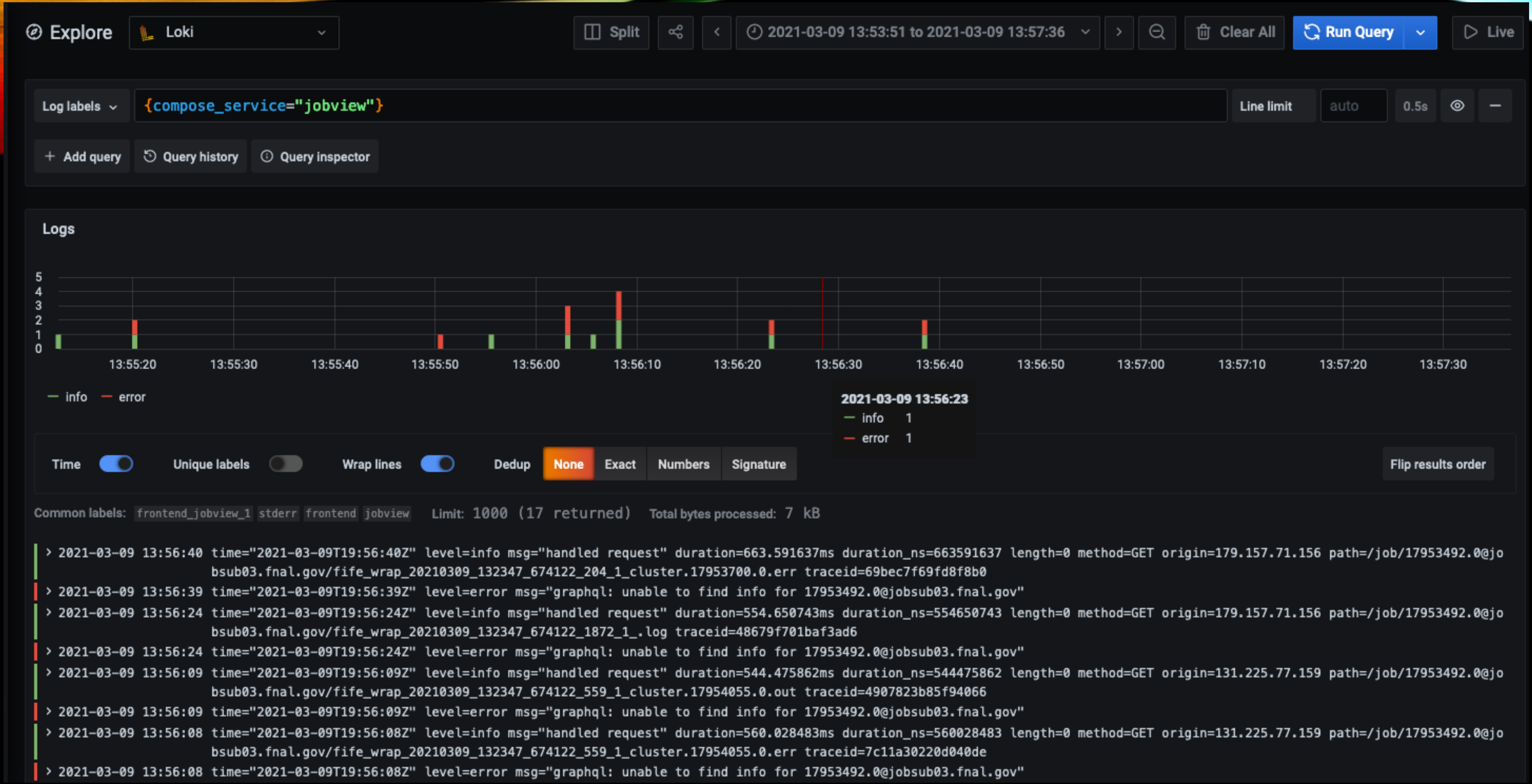
SpanID: 4122d0071ac54dcb

WOW THAT'S A **LOT** OF CALLS (A LOT OF JOBS)

DID SOMEONE SAY ENCORE?

Metrics, Logging and Tracing with Prometheus, Loki, and Jaeger

Case Study #2: **Jobview** job log viewer



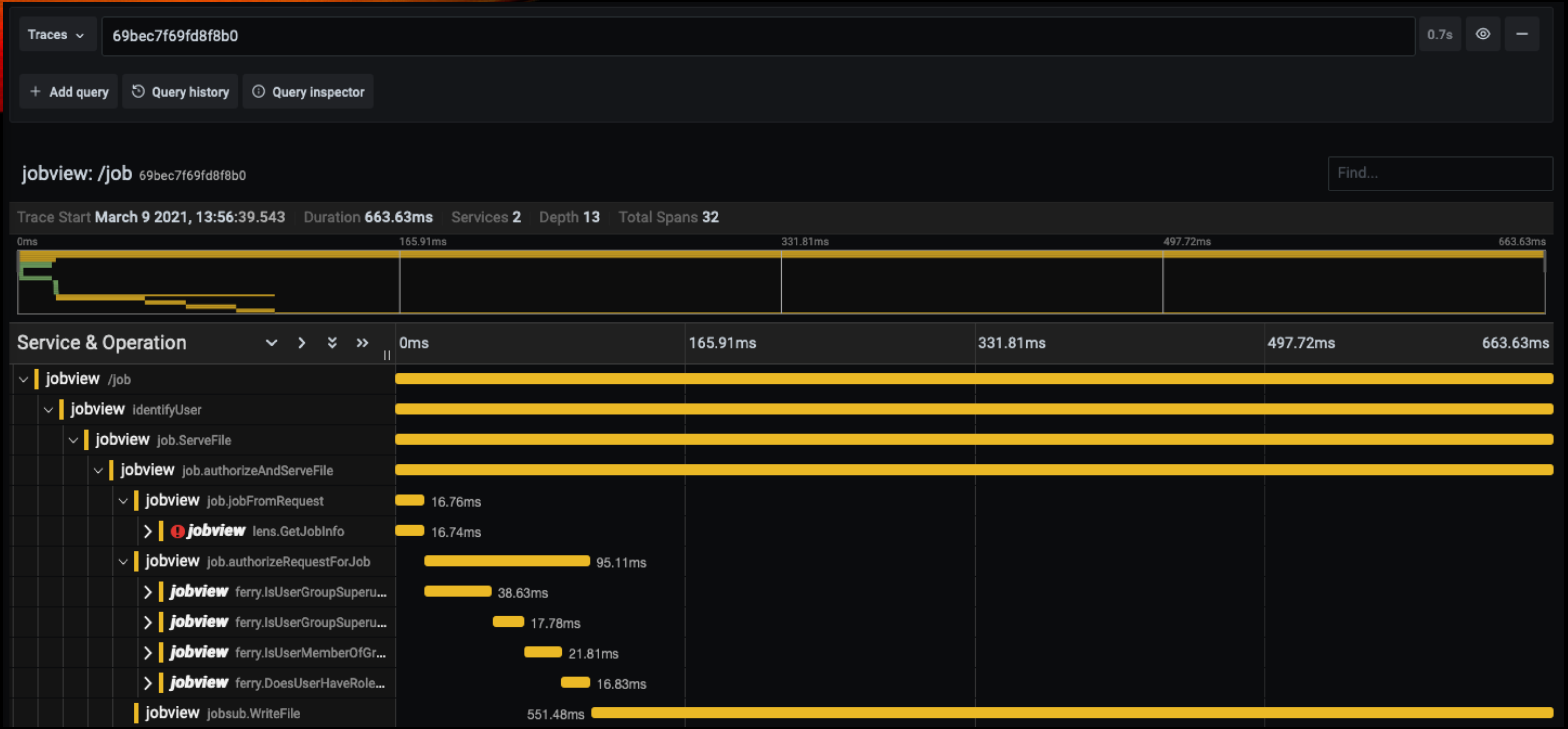
```
{compose_service="jobview"}
```

Time: ● on · Unique labels: ○ off · Wrap lines: ● on · Dedup: [✓ None] Exact Numbers Signature

Common labels: frontend_jobview_1 stderr frontend jobview   Limit: 1000 (17 returned)   Total bytes processed: 7 kB

```
> 2021-03-09 13:56:40 time="2021-03-09T19:56:40Z" level=info msg="handled request" duration=663.591637ms duration_ns=663591637 length=0 method=GET origin=179.157.71.156 path=/job/17953492.0@jo
                      bsub03.fnal.gov/fife_wrap_20210309_132347_674122_204_1_cluster.17953700.0.err traceid=69bec7f69fd8f8b0
> 2021-03-09 13:56:39 time="2021-03-09T19:56:39Z" level=error msg="graphql: unable to find info for 17953492.0@jobsub03.fnal.gov"
> 2021-03-09 13:56:24 time="2021-03-09T19:56:24Z" level=info msg="handled request" duration=554.650743ms duration_ns=554650743 length=0 method=GET origin=179.157.71.156 path=/job/17953492.0@jo
                      bsub03.fnal.gov/fife_wrap_20210309_132347_674122_1872_1_.log traceid=48679f701baf3ad6
> 2021-03-09 13:56:24 time="2021-03-09T19:56:24Z" level=error msg="graphql: unable to find info for 17953492.0@jobsub03.fnal.gov"
> 2021-03-09 13:56:09 time="2021-03-09T19:56:09Z" level=info msg="handled request" duration=544.475862ms duration_ns=544475862 length=0 method=GET origin=131.225.77.159 path=/job/17953492.0@jo
                      bsub03.fnal.gov/fife_wrap_20210309_132347_674122_559_1_cluster.17954055.0.out traceid=4907823b85f94066
> 2021-03-09 13:56:09 time="2021-03-09T19:56:09Z" level=error msg="graphql: unable to find info for 17953492.0@jobsub03.fnal.gov"
> 2021-03-09 13:56:08 time="2021-03-09T19:56:08Z" level=info msg="handled request" duration=560.028483ms duration_ns=560028483 length=0 method=GET origin=131.225.77.159 path=/job/17953492.0@jo
                      bsub03.fnal.gov/fife_wrap_20210309_132347_674122_559_1_cluster.17954055.0.err traceid=7c11a30220d040de
> 2021-03-09 13:56:08 time="2021-03-09T19:56:08Z" level=error msg="graphql: unable to find info for 17953492.0@jobsub03.fnal.gov"
```

WELL WE GOT TROUBLE (RIGHT HERE!)



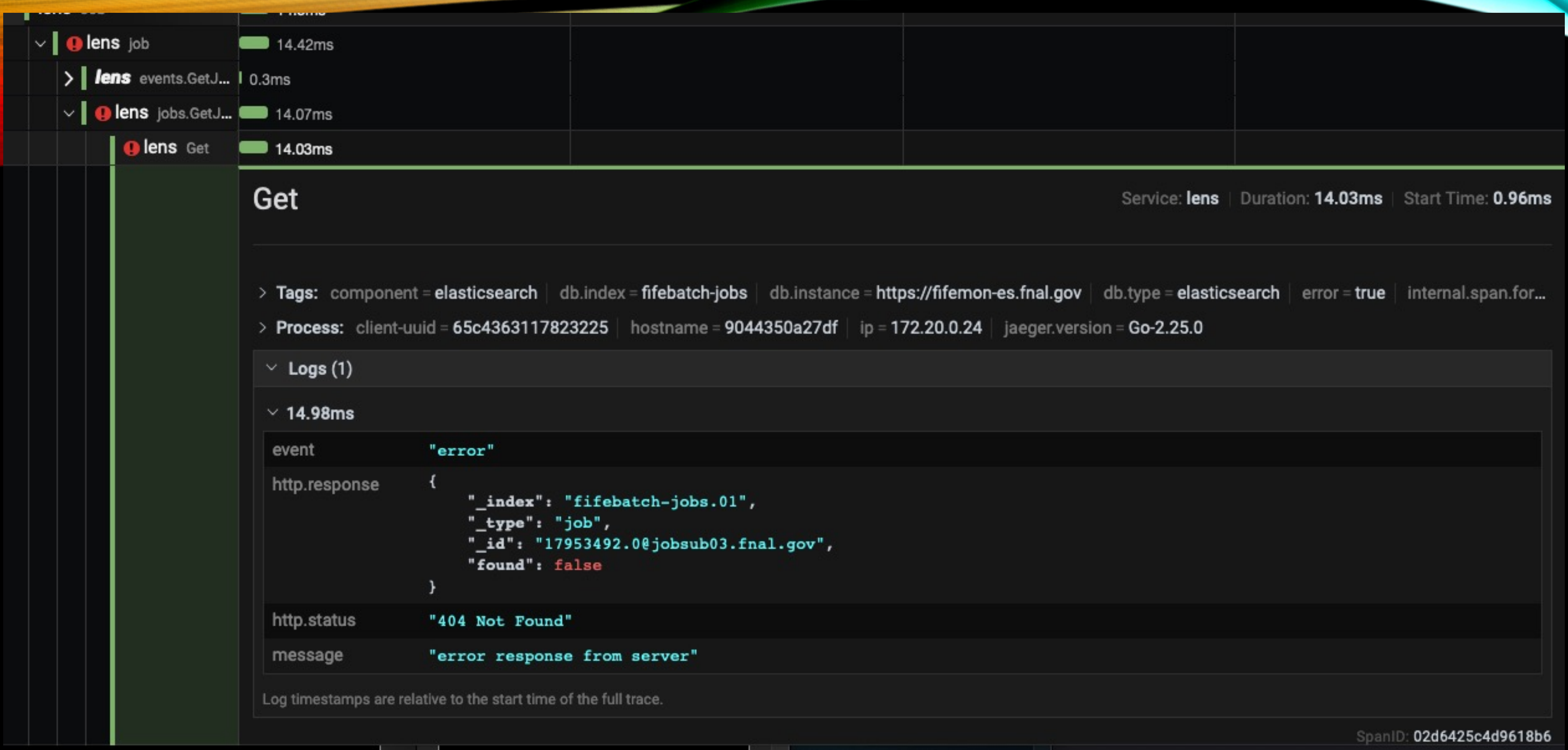
69bec7f69fd8f8b0

jobview: /job 69bec7f69fd8f8b0

Trace Start **March 9 2021, 13:56:39.543** Duration **663.63ms** Services **2** Depth **13** Total Spans **32**

```
Service & Operation                               0ms   165.91ms   331.81ms   497.72ms   663.63ms
jobview /job
  jobview identifyUser
    jobview job.ServeFile
      jobview job.authorizeAndServeFile
        jobview job.jobFromRequest                16.76ms
          jobview lens.GetJobInfo                 16.74ms
        jobview job.authorizeRequestForJob        95.11ms
          jobview ferry.IsUserGroupSuperu...      38.63ms
          jobview ferry.IsUserGroupSuperu...      17.78ms
          jobview ferry.IsUserMemberOfGr...       21.81ms
          jobview ferry.DoesUserHaveRole...       16.83ms
        jobview jobsub.WriteFile                  551.48ms
```

TO THE TRACE!



Get

Service: lens | Duration: 14.03ms | Start Time: 0.96ms

Tags: component = elasticsearch | db.index = fifebatch-jobs | db.instance = https://fifemon-es.fnal.gov | db.type = elasticsearch | error = true | internal.span.for...

Process: client-uuid = 65c4363117823225 | hostname = 9044350a27df | ip = 172.20.0.24 | jaeger.version = Go-2.25.0

Logs (1)

14.98ms

```
event           "error"
http.response   {
                    "_index": "fifebatch-jobs.01",
                    "_type": "job",
                    "_id": "17953492.0@jobsub03.fnal.gov",
                    "found": false
                }
http.status     "404 Not Found"
message         "error response from server"
```

Log timestamps are relative to the start time of the full trace.

SpanID: 02d6425c4d9618b6

OH, SO IT REALLY WAS A LENS PROBLEM

OTHER RESOURCES

- Monitoring for everyone
- How Humans see data
- Fool-Proof Kubernetes Dashboards for Sleep-Deprived Oncalls
- Stacked Area Graphs Are Not Your Friend
- Friends don’t let friends abuse pie charts
- The SRE book: Practical Alerting from Time-Series

List at https://landscape.fnal.gov/docs/using/resources/