

LGTM stack

Or: philosophy of Observability

Richard “RichiH” Hartmann



Why this matters to you

Or: Who is that person?

My Background

- Director of Community @ Grafana Labs
- Prometheus team member
 - PromCon lead, Prometheus dev summit chair
- Founded OpenMetrics, goal is IETF RFC
- CNCF SIG Observability chair
- snmp_exporter & modbus_exporter
- Built Europe's most modern datacenter, monitored through SNMP and Modbus/TCP only
- Extensive paid and pro bono consulting on how to instrument existing and new applications

My Background

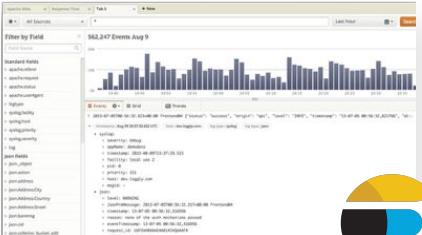
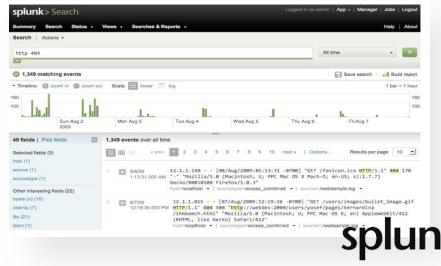
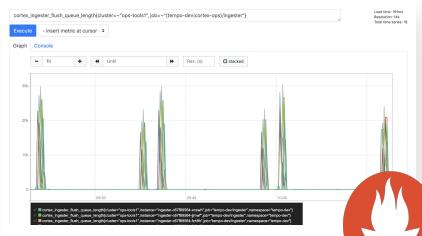
- Ran the backbone of an ISP for eleven years
 - Was the only person on call
 - My life & sanity depended on on-point monitoring & alerting
- Active in RIPE, IETF, DENOG, #networker etc
 - RFC to my name, changed RIPE NCC's IPv4 PI policies, etc.
- Prometheus transition for Germany's oldest ISP, ~5k devices
- Staffed world's largest IRC network for more than a decade
- Run conferences & monitoring from 100s to 18k attendees
 - DENOG, DebConf, FOSDEM, CCC, GrafanaCon, PromCon

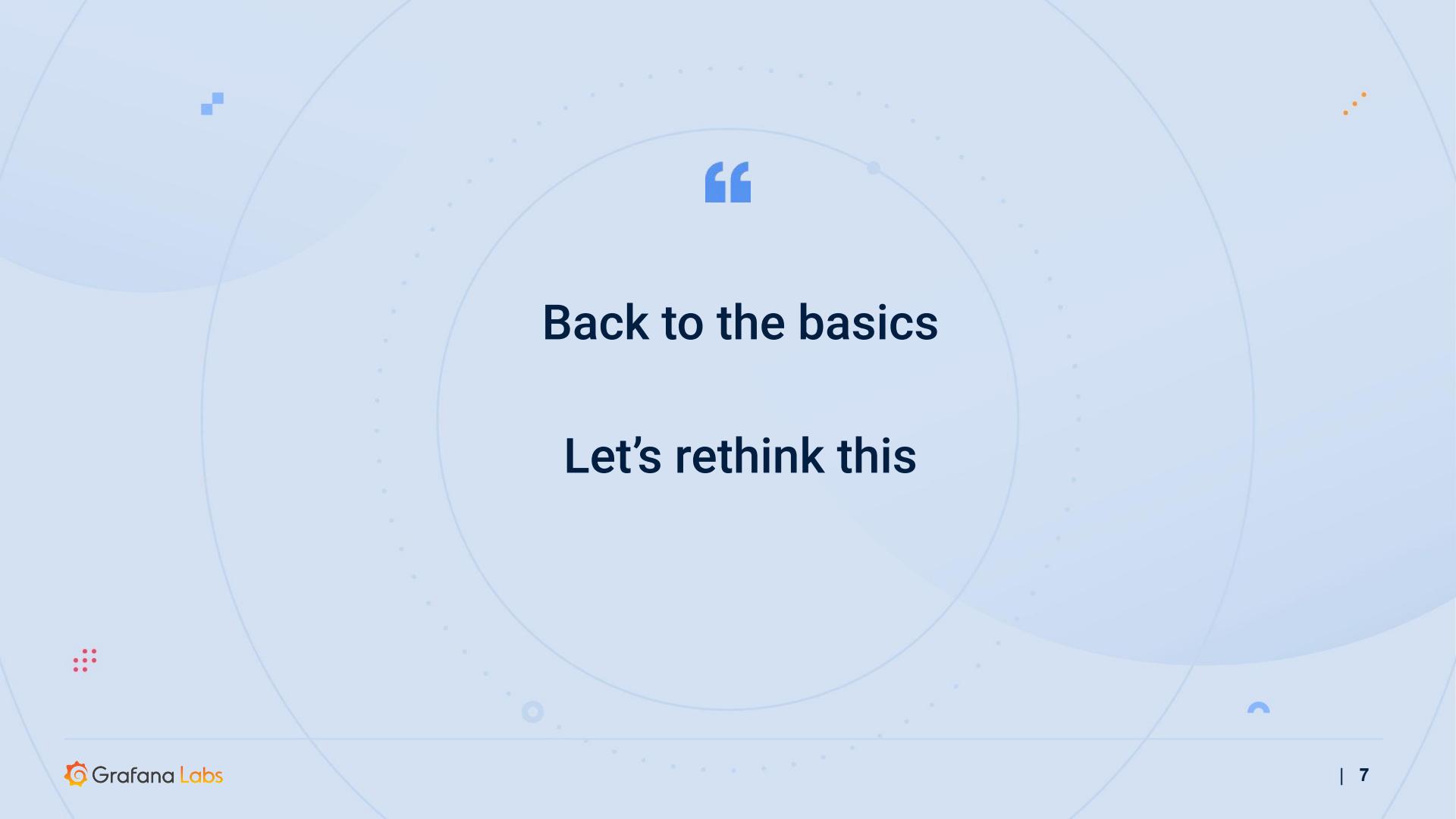
“

I come from the trenches of tech

I know your pains and want to help

Today's reality: Disparate systems. Disparate data.





“

Back to the basics

Let's rethink this

Observability & SRE

Or: Buzzwords, and their useful parts

Buzzword alert!

- Cool new term, almost meaningless by now, what does it mean?
 - Pitfall alert: Cargo culting
 - It's about changing the behaviour, not about changing the name
- “Monitoring” has taken on a meaning of collecting, not using data
 - One extreme: Full text indexing
 - Other extreme: Data lake
- “Observability” is about enabling humans to understand complex systems
 - Ask **why** it’s not working instead of just knowing that it’s not

IDENTIFYING WOOD

ACCURATE RESULTS
WITH SIMPLE TOOLS

R. Bruce Hoadley

HODDAR.COM



YEP, IT'S WOOD

Complexity

- Fake complexity, a.k.a. bad design
 - Can be **reduced**
- Real, system-inherent complexity
 - Can be **moved** (monolith vs client-server vs microservices)
 - Must be **compartmentalized** (service boundaries)
 - Should be **distilled meaningfully**

SRE

- Another buzzword ;)
- At its core: Align incentives across the org
 - Easier said than done
- SLI, SLO, SLA
 - Error budgets allow devs, ops, PMs, etc. to optimize for shared benefits

SRE

- Everyone uses the same tools & dashboards
 - Shared incentive to invest into tooling
 - Pooling of institutional system knowledge
 - Shared language & understanding of services
- Service?
 - Compartmentalized complexity, with an interface
 - Different owners/teams
 - Contracts define interfaces

Services

- Why “contract”: Shared agreement which MUST NOT be broken
 - Internal and external customers rely on what you build and maintain
- Other common term: layer
 - The Internet would not exist without network layering
 - Enables innovation, parallelizes human engineering
- Other examples: CPUs, harddisk, compute nodes, your lunch

Services

- Customers care about services being up, not about individual components
- Discern between different SLIs
 - Primary: service-relevant, for alerting
 - Secondary: informational, debugging, might be underlying's primary

**Anything currently or imminently impacting customer service must be alerted upon
But nothing(!) else**

Prometheus

Prometheus 101

- Inspired by Google's Borgmon
- Time series database
- unit64 millisecond timestamp, float64 value
- Instrumentation & exporters
- Not for event logging
- Dashboarding via Grafana

Main selling points

- Highly dynamic, built-in service discovery
- No hierarchical model, n-dimensional label set
- PromQL: for processing, graphing, alerting, and export
- Simple operation
- Highly efficient

Main selling points

- Prometheus is a pull-based system
- Black-box monitoring: Looking at a service from the outside (Does the server answer to HTTP requests?)
- White-box monitoring: Instrumenting code from the inside (How much time does this subroutine take?)
- Every service should have its own metrics endpoint
- Hard API commitments within major versions

Time series

- Time series are recorded values which change over time
- Individual events are usually merged into counters and/or histograms
- Changing values are recorded as gauges
- Typical examples
 - Access rates to a webserver (counter)
 - Temperatures in a datacenter (gauge)
 - Service latency (histograms)

Super easy to emit, parse & read

```
http_requests_total{env="prod",method="post",code="200"} 1027
http_requests_total{env="prod",method="post",code="400"} 3
http_requests_total{env="prod",method="post",code="500"} 12
http_requests_total{env="prod",method="get",code="200"} 20
http_requests_total{env="test",method="post",code="200"} 372
http_requests_total{env="test",method="post",code="400"} 75
```

Scale

- Kubernetes is Borg
- Prometheus is Borgmon
- Google couldn't have run Borg without Borgmon (plus Omega and Monarch)
- Kubernetes & Prometheus are designed and written with each other in mind

Scale

- 1,000,000+ samples/second no problem on current hardware
- ~200,000 samples/second/core
- 16 bytes/sample compressed to 1.36 bytes/sample

The highest we saw in production on a single Prometheus instance were 15 million active times series at once!

Long term storage

- Two long term storage solutions have Prometheus-team members working on them
 - Thanos
 - Historically easier to run, but slower
 - Scales storage horizontally
 - Cortex
 - Easy to run these days
 - Scales both storage, ingester, and querier horizontally
- Both converge on tech again; have annoyed people with “Corthanos” for years

OpenMetrics

OpenMetrics

- Prometheus is the de-facto standard in cloud-native metric monitoring and beyond
 - Same is true for Prometheus exposition format
- Ease of exposing data has lead to an explosion in compatible metrics endpoints
 - Thousands of exporters and integrations
- Standard exporters and libraries make integrating this easy

OpenMetrics

- Some other projects & vendors are torn about adopting something from a competing product or project
- Especially traditional vendors prefer to support official standards
- Re-use installed base of Prometheus
 - Ease of adoption
 - No kitchen sink, do one thing well, remain focused and opinionated
- Many competing companies collaborated on OpenMetrics and helped shape
- The result is an actually neutral standard
 - Official IETF RFC process

Three pillars

- The “three pillars”
 - Metrics: Alerting, dashboarding, AI/ML
 - Logs: Due diligence, debugging, incident response
 - Traces: Debugging, performance tuning
- Plan since ‘15 was to have a label-based logging format
- Google shared scaling information about traces in ‘16, it was clear that exemplars were the only way
 - First-class support for exemplars in OpenMetrics

“

Prometheus changed how the world did metric-based monitoring. I started OpenMetrics to change how the world does Observability.
(Not joking.)

Loki

Loki 101

- Following the same label-based system like Prometheus
- No full text index needed, incredible speed
- Work with logs at scale, without the massive cost
- Access logs with the same label sets as metrics
- Turn logs into metrics, to make it easier to work with them
- Make direct use of syslog data, via promtail

2019-12-11T10:01:02.123456789Z {app="nginx", instance="1.1.1.1"} GET /about

Timestamp

with nanosecond precision

Prometheus-style Labels

key-value pairs

Content

log line

indexed

unindexed

Query Frontend → Sharding



owen 6:01 PM

all my queriers restarting pretty regularly though

we [REDACTED] did it

image.png ▾



that's the new upper bound LE



Cyril 🖌 6:03 PM

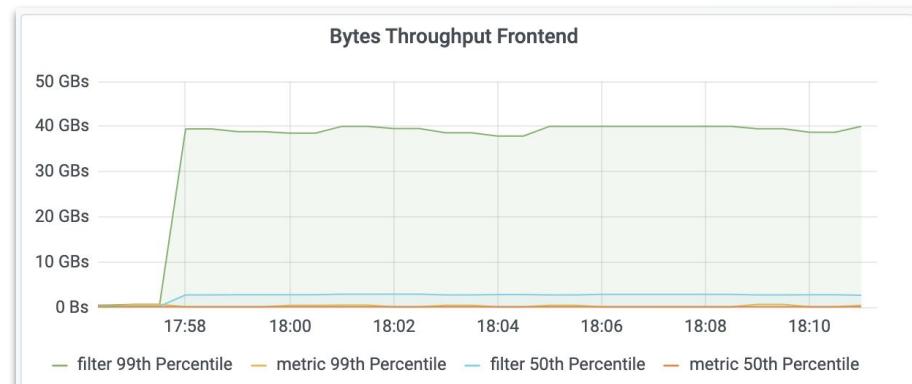
75 CPU

lol



Cyril 🖌 6:13 PM

Hey that's 20s for 1TB of data.



Grafana Loki: Blazin' Fast LogQL

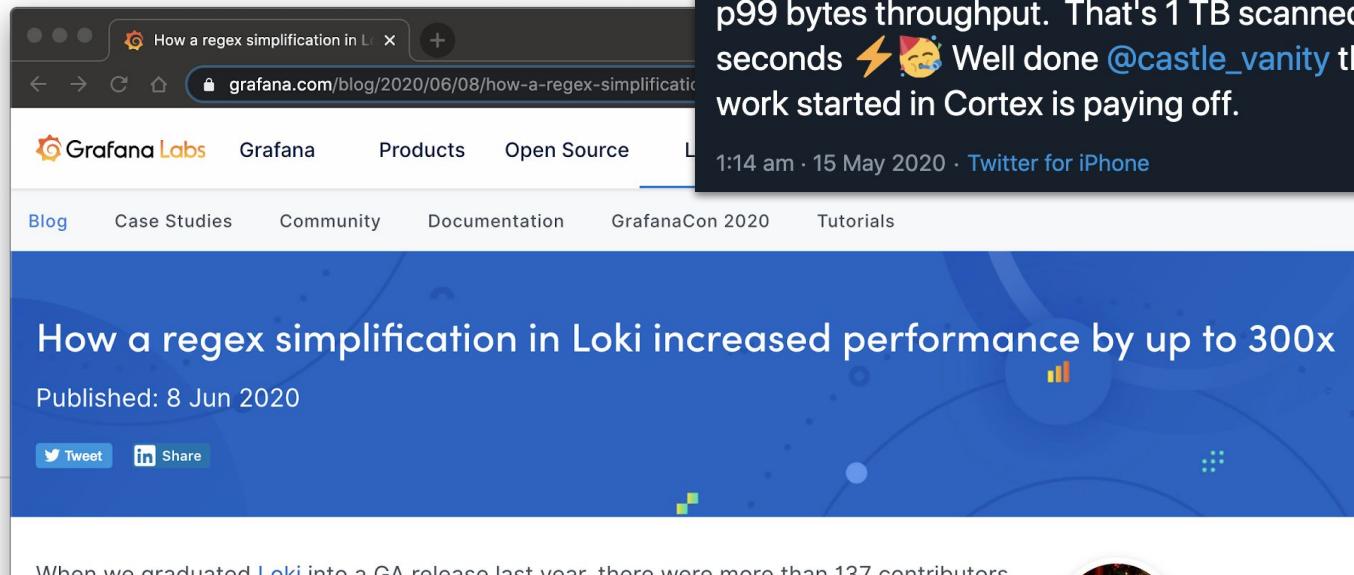
 Cyril Tovena @Kuqd · Feb 19
Just ran a metric query in Loki that pulled out 420GB of log data in 45 seconds. Almost 10GB/s ⚡, we'll get there 🚀.

2 12 44

 Cyril Tovena
@Kuqd

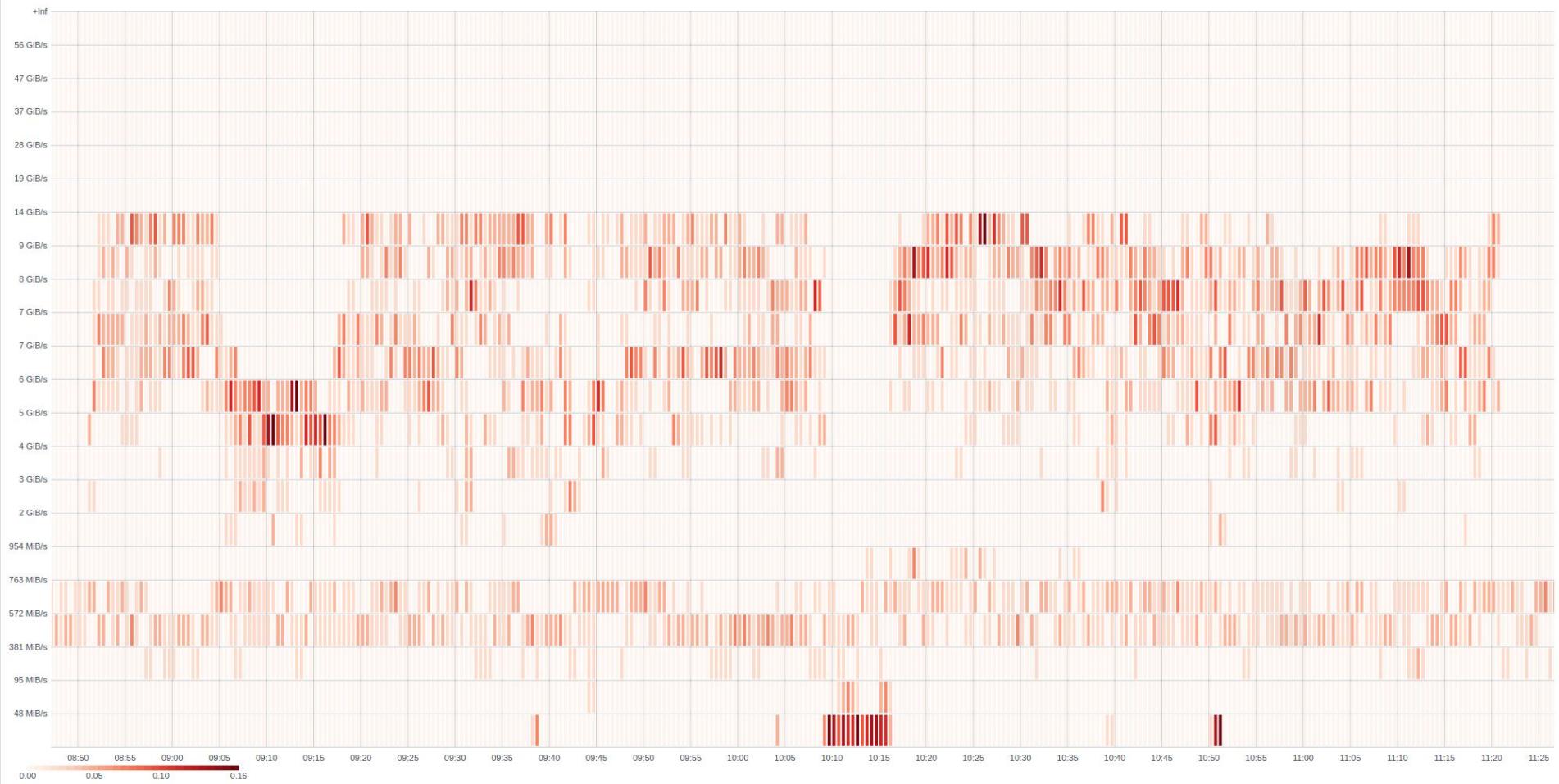
Today I've seen Loki being pushed to its limits, 40gb/s p99 bytes throughput. That's 1 TB scanned in 25 seconds ⚡🚀 Well done @castle_vanity the sharding work started in Cortex is paying off.

1:14 am · 15 May 2020 · Twitter for iPhone

A screenshot of a web browser displaying a blog post from the Grafana Labs website. The post title is "How a regex simplification in Loki increased performance by up to 300x". Below the title, it says "Published: 8 Jun 2020". At the bottom of the page, there are "Tweet" and "Share" buttons, and a note about 137 contributors.

When we graduated **Loki** into a GA release last year, there were more than 137 contributors

Bytes Throughput Frontend



Tempo

Tempo

- Exemplar-based: Jump from relevant logs & metrics
 - Native to Prometheus, Cortex, Thanos, and Loki, but works with all
 - Exemplars work at Google scale, with the ease of Grafana
- Object store only: No Cassandra, Elastic, etc.
- 100% compatible with OpenTelemetry Tracing, Zipkin, Jaeger
- 100% of your traces, no sampling

Tempo @ Grafana Labs

- 1,000,000 samples per second; 100% “sampling”
- 14-day retention, 3 copies stored, compression part of CPU cost
- ~200 CPU cores
- ~300 GiB RAM
- 40 TiB object storage

Concrete plans for next 10x jump already being implemented

Bringing it together

From logs to traces

The screenshot shows the Grafana interface with a sidebar containing various monitoring icons. The main area displays a log entry and a configuration panel for derived fields.

Log Labels:

- all @ pod app-76bb7d944c-r7gb7
- all @ stdout
- all @ traceID le38524b7f6e13f
- all @ _2020_11_24T15_20_29_996153289Z
- all @ cluster tns
- all @ container app
- all @ level info
- all @ namespace tns
- all @ filename /var/log/pods/ns_app-76bb7d944c-r7gb7_68f6413f-be42-486c-88f0-ed86badd1766/app/3.log
- all @ job tns/app
- all @ level_extracted info
- all @ msg HTTP client success
- all @ duration 53.027253ms
- all @ name app
- all @ status 500
- all @ F
- all @ pod_template_hash 76bb7d944c
- all @ url http://db

Parsed Fields:

- all @ TraceID le38524b7f6e13f Tempo
- all @ duration 53.027253ms
- all @ level info

Derived fields

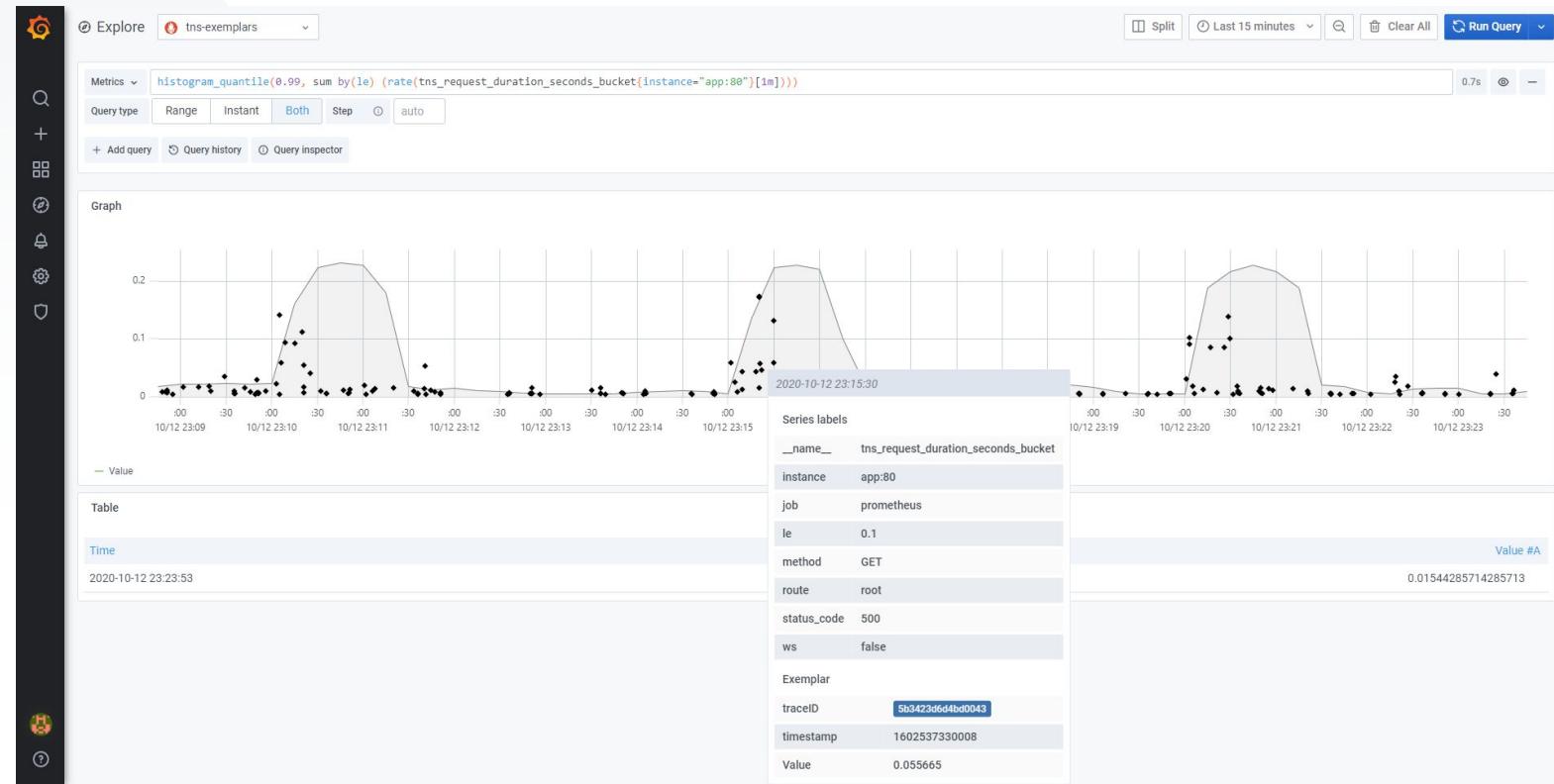
Derived fields can be used to extract new fields from the log message and create link from it's value.

Configuration:

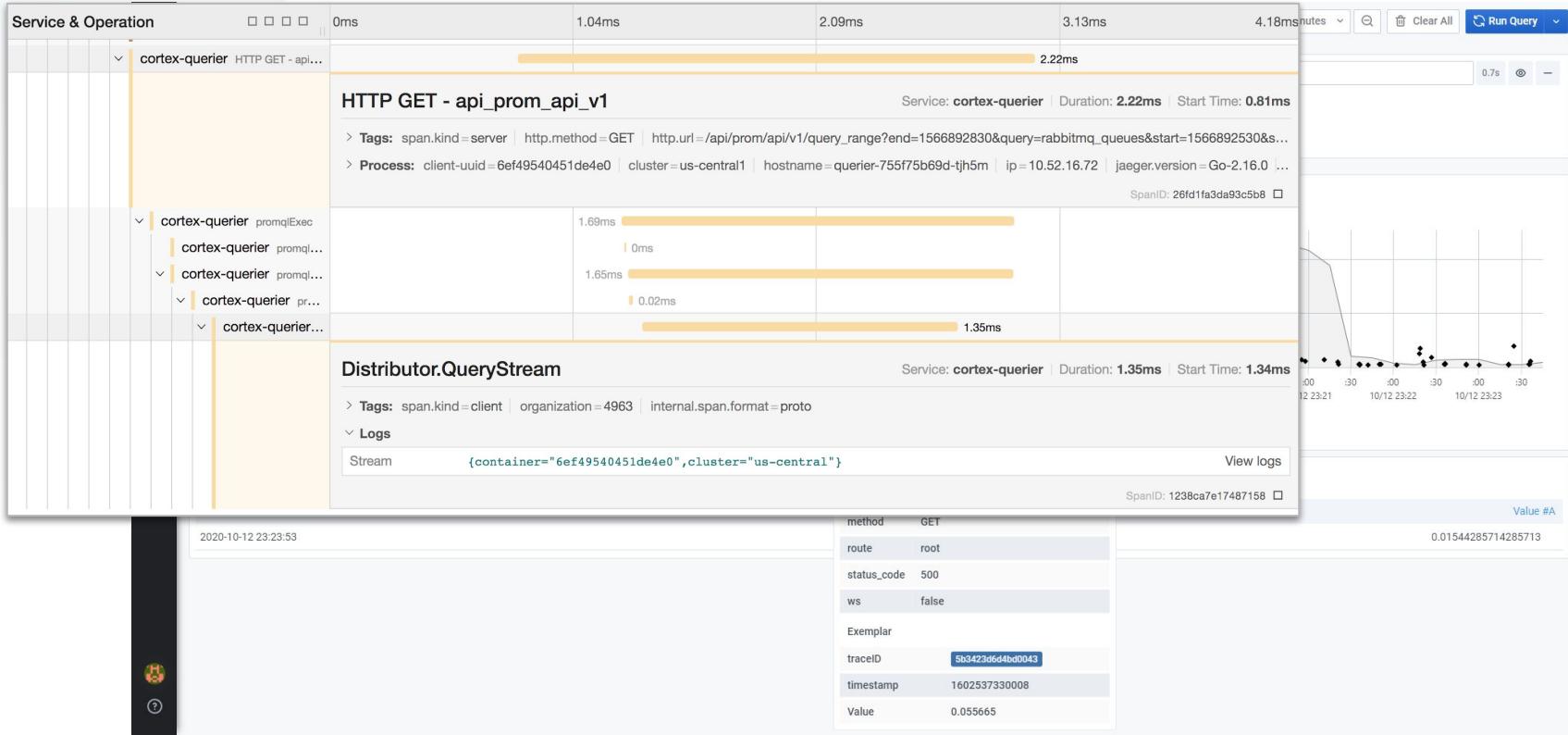
Name	TracelD	Regex	(?:traceID trace_id)=(\w+)
Query	\$_value.raw		
Internal link	<input checked="" type="checkbox"/> Tempo		

Buttons: + Add, Show example log message

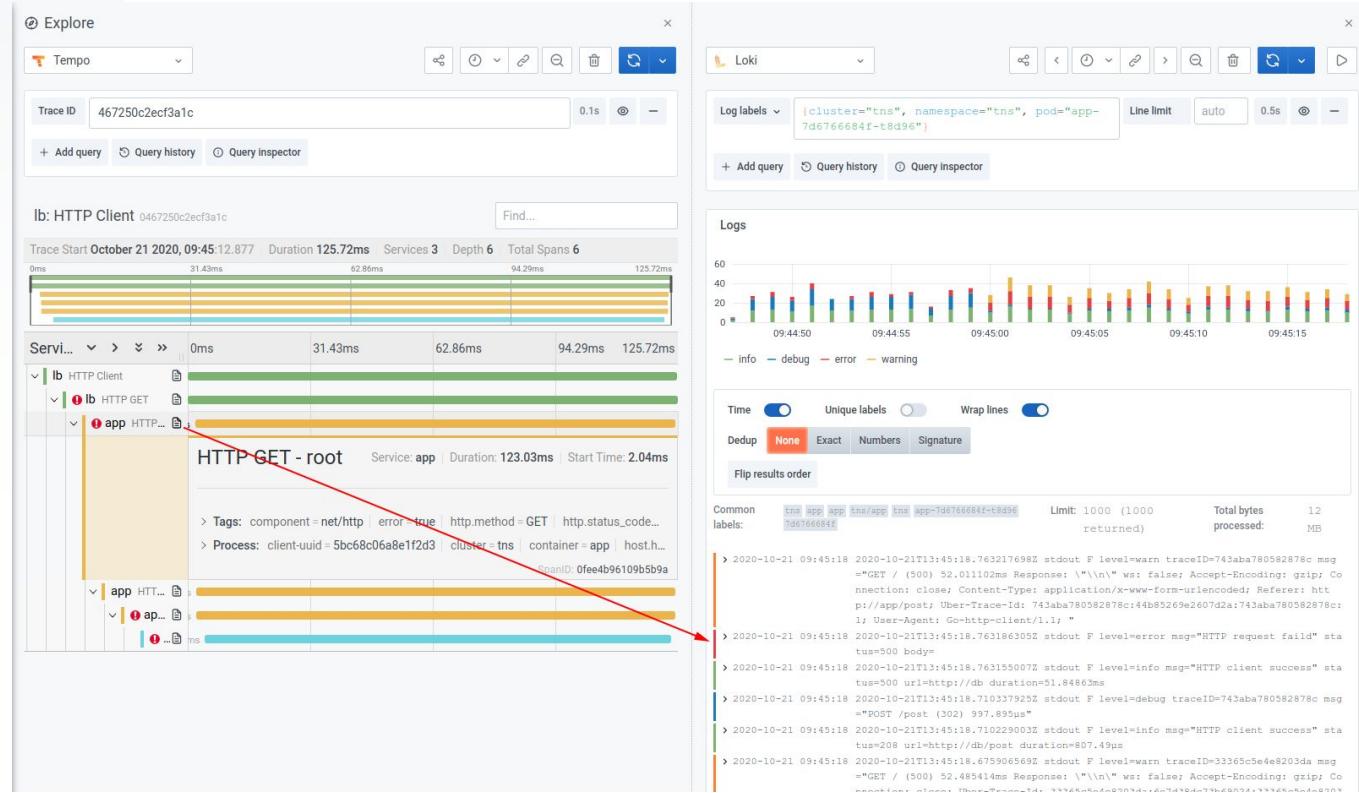
From metrics to traces



From metrics to traces



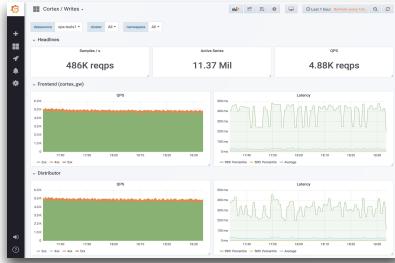
...and from traces to logs



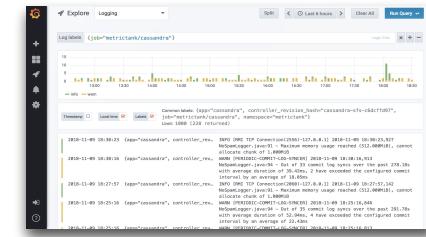
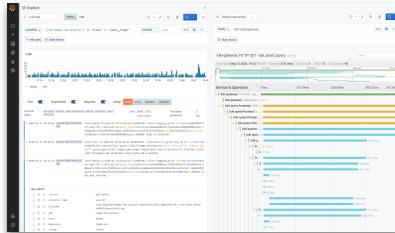
All of this is Open Source and you can run it yourself

The Watchmen APP 5:48 PM
[FIRING:3] CFRoutesTooLow (cf router pcf-dev 10.130.7.229:9186 pcf-dev-firehose codelab-monitor gorouter)
The number of routes in the router's routing table is dangerously low: 113

The Watchmen APP 5:54 PM
[RESOLVED] CFRoutesTooLow (cf router pcf-dev 10.130.7.229:9186 pcf-dev-firehose codelab-monitor gorouter)
The number of routes in the router's routing table is dangerously low: 113



9/9
0:590 Action selected - action ✓ { 1/290 9:02:49 8:22
13:26:20 10:13:26 10:13:26 10:13:26 10:13:26 10:13:26
0:590 x 2:13:09:05:05
Cortex 2:13:09:27:05
Relay 6:0 10:13:26 10:13:26 10:13:26 10:13:26 10:13:26
Picking 6:0 10:13:26 10:13:26 10:13:26 10:13:26 10:13:26
Started Cortex, changed 10:13:26
Selected Router, Router Tech
Relay#70 Panel F
(Not in relay)
ISAS
First actual case of bug being found:
show how.
10:13:26



Thank You!

richih@richih.org
twitter.com/TwitchiH
github.com/RichiH/talks