



Philosophy of Observability

Leveraging Cloud-Native Technologies



Richard "RichiH" Hartmann

My Background

- Senior Developer Programs Director @ Grafana Labs
- Prometheus team member (CNCF graduated project)
 - PromCon lead, Prometheus Dev Summit chair
- CNCF Governing Board
- CNCF Technical Oversight Committee
- CNCF Technical Advisory Group Observability chair
- OpenMetrics founder
- OpenTelemetry member



-11

My offer

- Built Europe's most modern datacenter, monitored through SNMP and Modbus/TCP only
- Maintainer of
 - <u>https://github.com/prometheus/snmp_exporter</u>
 - <u>https://github.com/RichiH/modbus_exporter</u>
 - Others..

If you have a cool / large factory to visit and want to trade a talk

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

Today's reality: Disparate systems. Disparate data.

ORACLE











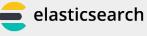


6 Grafana



splunk>







Lector #+ Exe

Back to the basics

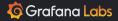
••

Let's rethink this



-

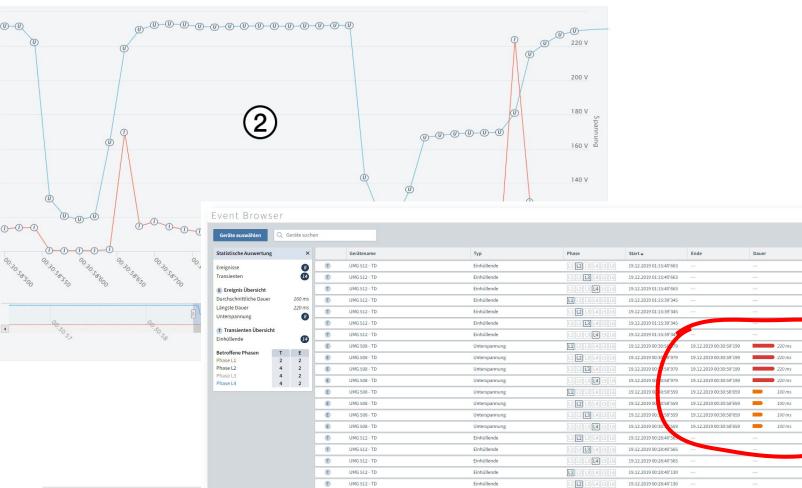
How humanity deals with data



•

-

....



1

UMG 512 - TD

Einhüllende

19.12.2019 00:28:40'130

Grafana Labs

4

20 • < 1 2 >

19.12.2019 - 19.12.2019 00:00 - 23:59

Wert

117.836 V (MIN)

117.806 V (MIN)

117.825 V (MIN)

117.830 V (MIN

118.640 V (MIN)

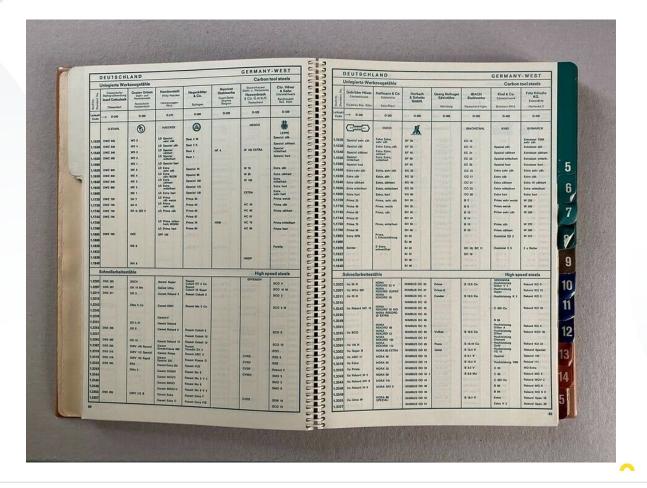
118.612 V (MIN)

118.622 V (MIN)

118.631 V (MIN)

7- :=-

Analysieren



Grafana Labs

Second on land This Toke Developed of Hard Arounding for Shales on the Aller quant admitting of the 14 1865 and and with the dame from tot e dames Citudes J. A. 11-1899 interned from these and fire matter the link and shad not mindle built built someting filles subility of andiation attend also totals and same - and and lade lives and pick thing fater have the hannes carding a rations in the whole her Hardines & to 110 have not see hand from tit will some and my the to write said light we graden we are fin have me for with string of Monday schould Contribut for land and fair reacher made 5 Will knot Moring - Litter hard Straining all. win the thank tog included the this It as 11- marchitch Low y him halt southing freshines thing is the wind on both treads Indday S.h. 11 sumplings as entiring and busing graphat 3° soft toutilised fine have from Et to their it is a with ab Middle part the same thatter see work stand Com a ving Midhelier Service 43:311 to Friday Sailing & siddhar of decks that 5 30 Tweeday of 2 35 Rolling Jack 19 tentences sine than and this with strong at 2 de 2 m the stream class much and your and June 128 35 I are butter light the house on alloud Fundas S.M. 11 president of Am The sea so the g to side the ship torning of and I marths from Murran Sound to be Frenderick Colonado treach and latter line continues by fight hims alisto in class one but and The air resurter stands buy? 200 Hacklist





d.

~





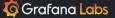
d.

.....

Humanity has optimized detailed accounts into key events into numbers for millenia

Again and again and again





Observability & SRE

Or: Buzzwords, and their useful parts



•

....

Observability, the buzzword

- 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 new questions on the fly
 - Ask **why** it's not working instead of just knowing that it's not
- Terms such as "Observability 2.0", "Observability 3.0", and "Observability,"
 4.0", are other examples of buzzwordiness

[...] observations are [...] approximations to the truth [...] this can be accomplished in no other way than by a suitable combination of more observations than the number absolutely requisite for the determination of the unknown quantities

Carl Friedrich Gauß, 1809

Observability is a measure of how well internal states of a system can be inferred from knowledge of its external outputs.

BB

Rudolf Emil Kálmán, 1960



....

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 (observability...)
 - Can **not** be reduced

Services

- What's a service?
 - Compartmentalized complexity, with an interface
 - Different owners/teams
 - Contracts define interfaces
- 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, hard drive, compute nodes, your lunch _

Grafana Labs

al.

Cloud-native vs client-server vs mainframe vs...

- A mainframe application and a microservices fleet are fundamentally the same
 - You can *move* system-inherent complexity, but...
- Microservices broke up old service and system boundaries
 - Enabling horizontal scalability, arguably at the cost of vertical scalability
- Previous-generation tooling is designed to understand system complexity along existing service boundaries
 - Cloud native tooling is able to deal with this increased complexity
 - NB: This means previous-generation complexity is even easier to observe



SRE

- At its core: Align incentives across the org
 - Error budgets allow devs, ops, PMs, etc. to optimize for shared benefits
- Measure it!
 - SLI: Service Level Indicator: What you measure
 - SLO: Service Level Objective: What you need to hit
 - SLA: Service Level Agreement: When you need to pay
- Discern between different SLIs
 - Primary: service-relevant, for alerting
 - Secondary: informational, debugging, might be underlying's primary

Shared understanding

- Everyone uses the same tools & dashboards
 - Shared incentive to invest into tooling
 - Pooling of institutional system knowledge
 - Shared language & understanding of services

Alerting

- Customers care about services being up, not about individual components
- 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
- Rich ecosystem, 1,000s of instrumentations & exporters
- Cloud-native default

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
 - Requests to a webserver (counter)
 - Temperatures in a datacenter (gauge)
 - Service latency (histograms)



Cloud-native default

- Kubernetes =~ Borg
- Prometheus =~ Borgmon
- Google couldn't have run Borg without Borgmon
- Kubernetes & Prometheus are designed and written with each other in mind

Prometheus 101

- 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
- New release candidate every six weeks

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

-11

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

PromQL

All partitions in my entire infrastructure with more than 100GB capacity that are not mounted on root?

node_filesystem_bytes_total{mountpoint!="/"} / 1e9 > 100

{device="sda1", mountpoint="/home", instance="10.0.0.1"} 118.8
{device="sda1", mountpoint="/home", instance="10.0.0.2"} 118.8
{device="sdb1", mountpoint="/data", instance="10.0.0.2"} 451.2
{device="xdvc", mountpoint="/mnt", instance="10.0.0.3"} 320.0

PromQL

What's the ratio of request errors across all service instances?

```
sum by(path) (rate(http_requests_total{status="500"}[5m])) /
sum by(path) (rate(http_requests_total[5m]))
```

```
{path="/status"} 0.0039
{path="/"} 0.0011
{path="/api/v1/topics/:topic"} 0.087
{path="/api/v1/topics} 0.0342
```

Prometheus 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
- Reliable into the tens of millions of active series



Mimir



••

-

•••

Mimir

- For Metrics
- Prometheus \rightarrow Cortex \rightarrow Grafana Enterprise Metrics \rightarrow Mimir
- Scales to more than 1,000,000,000 Active Series
- Blazingly fast query performance
- Hard multi-tenancy, access control, and three-way replication
- Can ingest native OpenTelemetry, DataDog, Graphite, and Influx

-11

Mimir @ Grafana

- 1,000,000,000 Active Series in one cluster
- 1,500 machines
- 7,000 CPU cores
- 30 TiB RAM



1

~



Loki



••

....

Loki 101

- For Logs
- Following the same label-based system as Prometheus
 - Only index what you need often, query the rest
 - "Index the labels, query the data"
- Work with logs at scale, without the massive cost
 - Scalable low latency write path
 - Flexible schema on read
- Access logs with the same label sets as metrics
 - Turn logs into metrics, to make it easier & cheaper to work with them

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

Timestamp	Prometheus-style Labels	Content
with nanosecond precision	key-value pairs	log line
	indexed	unindexed
		line in the second s
		• • •
Grafana Labs		

Loki @ Grafana Labs

- Largest user cluster: 180 TiB per day
- Queries regularly peak at 900GB/s
 - Query 10TB in 12 seconds, including complex processing of result sets

	Kick start your query Label browser Explain	Builder Cod
	<pre>max_over_time({job="query-frontend"} = "metrics.go" logfmt unwrap bytes(throughput)[\$interval]) by (namespace) / 1e9</pre>	
	> Options Legend: - Type: Range	
+	Add query D Query history O Inspector	
Gr	Lines Bars Points Stacked lines	Stacked bars
00		
00		
00		
00		
0		
00		
00	and and the second of the second of the second second and the second s	munduluter
0		3 21:00 01/1





Tempo



••

••••

Tempo

• For **T**races

- Historic problem: Traces require extremely rich metadata for analysis
 - Expensive, slow, and mandates sampling
- Exemplars: Leverage the extracted logs & metrics
 - Exemplars work at Google scale, with the ease of Grafana
 - Native to Prometheus, Cortex, Thanos, and Loki
- Index and search by labelsets available for those who need it
- 100% compatible with OpenTelemetry Tracing, Zipkin, Jaeger

11

Tempo @ Grafana Labs

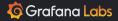
- 1,500,000 samples per second @ 450 MiB/s
 - o 560 MiB/s peak
- 14-day retention @ 3 copies stored
- Latencies:
 - o p99 2.5s
 - o p90 2.3s
 - o p50 1.6s



1

~

Grafana Pyroscope



•

Pyroscope

- For **P**rofiling
- Profiles
 - "How much CPU & RAM am I spending in what areas of the code?"
 - "...and how does this change over time?"
- Go: pprof
- Java: <u>https://github.com/grafana/JPProf</u>



-11

Data (and cost) savings



••

Logs to metrics

- Full text indexing: 10 TiB logs \rightarrow ~20 TiB index
- Loki: 10 TiB logs \rightarrow ~200 MiB index

- Logs @ Grafana ~600 Byte average per line
- Metrics ~1.36 Byte per metric sample

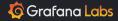
→ 99.8% reduction in storage size for first log line
 ~100% for every follow-up log line



-11



Bringing it together



••

-

From logs to traces

Log Labels:		Name	TraceID)		Regex (3	(?:traceID trace_id)=(\w+)	×
all 🕀 🔾 pod	app-76bb7d944c-r7gb7	Query	Sí va	alue.raw}					
al 🕀 🔾 stdout									
"II 🕀 🔾 traceID	1e38524b7f6e13f	Internal link		🔫 Tempo	~				
all ⊕ ⊖ _2020_11_24T15_2 all ⊕ ⊖ cluster	20_29_996153289Z	+ Add	Show	example log message					
.∥ ⊕ ⊖ container	app						-		
.d ⊕ ⊖ level	info								
📶 🕀 🔾 namespace	tns								
🔊 🕀 🖯 filename	/var/log/pods/tns_app-76bb7d94	4c-r7gb7_68f64	413f-be42	-486c-88f0-ed86badd1766/ag	pp/3.1	.og			
al 🕀 🔍 job	tns/app								
al \oplus \bigcirc level_extracted	info								
ul 🕀 🔍 msg	HTTP client success								
al 🕀 🔾 duration	53.027253ms								
ul 🕀 🔾 name	app								
"II 🕀 🔾 status	500								
all 🕀 🔾 F									
🔊 🕀 \ominus pod_template_has	sh 76bb7d944c								
al 🕀 🔾 url	http://db								
Parsed Fields:									
all () TraceID	le38524b7f6el3f Tempo								



From metrics to traces

Explore O tns-exemplars		□ Split ② Last 15 minutes ♀ ♀ □ Clear All ♀ Run Query
Metrics - histogram_quantile(0.99, sum by(le) (rate(tns_request_duration_seconds_bucket{instance="app:80")[im	1)))	0.7s 🐵 —
Q Query type Range Instant Both Step O auto		
+ Add query S Query history O Query inspector		
@ Graph		
 C C	2020-10-12 23:15:30 Series labels name tns_request_duration_seconds_bucket instance app:80 job prometheus	00 10 00 10 10/12 23:21 10/12 23:19 10/12 23:20 10/12 23:20
	le 0.1	
Time	method GET	Value #A
2020-10-12 23:23:53	route root	0.01544285714285713
	status_code 500	
	ws false	
	Exemplar	
445	traceID 5b3423d6d4bd0043	
8	timestamp 1602537330008	
0	Value 0.055665	

Grafana Labs

-11

From metrics to traces

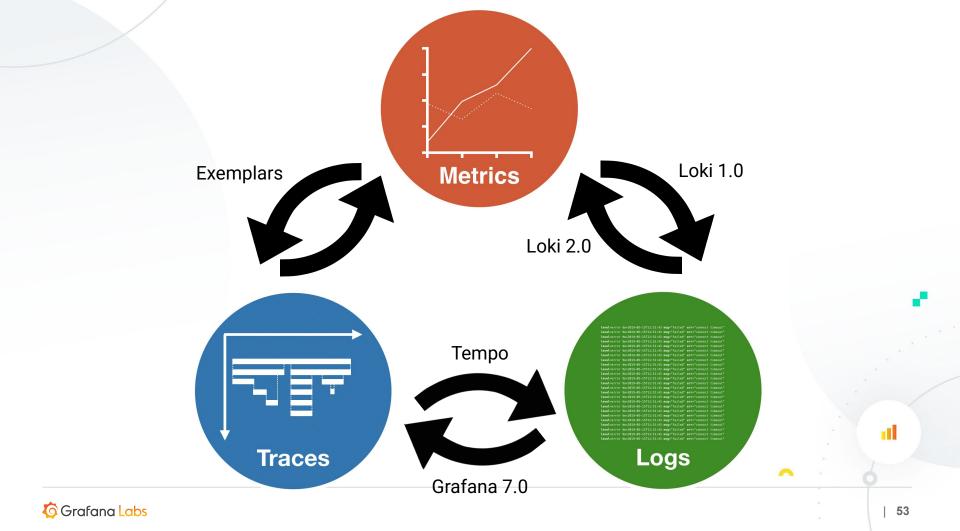
vice & Operation		Oms	1.04ms	2.09ms	3.13ms	4.18ms	nutes ~ Q	命 Clear All	C Run Query
~ cort	rtex-querier HTTP GET - api				2.22ms				0.7s 💿 —
		HTTP GET - api_prom_ap	pi_v1	Service: cortex	x-querier Duration: 2.22ms	Start Time: 0.81ms			
			ethod=GET http.url=/api/prom/api/v1/ 1de4e0 cluster=us-central1 hostnar						
					SpanID:	26fd1fa3da93c5b8 🛛			
~ (cortex-querier promqlExec		1.69ms		-		1 1		
	cortex-querier promal		0ms						
~	cortex-querier promql		1.65ms		-				
	~ cortex-querier pr		0.02ms						
	cortex-querier			1.35n	ms				
			tion 4062 internal once formation		x-querier Duration: 1.35ms	Start Time: 1.34ms	:00 :30	:00 :30)/12 23:22	:00 :30 10/12 23:23
		> Tags: span.kind=client organiza	tion=4963 internal.span.format=proto		x-querier Duration: 1.35ms	Start Time: 1.34ms	:00 :30	:00 :30	
		> Tags: span.kind=client organiza	ttion=4963 internal.span.format=prote)	x-querier Duration: 1.35ms	Start Time: 1.34ms View logs	:00 :30	:00 :30	
		> Tags: span.kind=client organiza)	• • • • • • • • • • • • • • • • • • •		:00 :30	:00 :30	
		> Tags: span.kind=client organiza)	• • • • • • • • • • • • • • • • • • •	View logs	:00 :30	:00 :30)/12 23:22 7	10/12 23:23 Value #/
	2020-10-12 23:23:53	> Tags: span.kind=client organiza) ral")	• • • • • • • • • • • • • • • • • • •	View logs	:00 :30	:00 :30)/12 23:22 7	10/12 23:23
	2020-10-12 23:23:53	> Tags: span.kind=client organiza		method GET	• • • • • • • • • • • • • • • • • • •	View logs	:00 :30	:00 :30)/12 23:22 7	10/12 23:23 Value #/
	2020-10-12 23:23:53	> Tags: span.kind=client organiza		method GET route root	• • • • • • • • • • • • • • • • • • •	View logs	:00 :30	:00 :30)/12 23:22 7	10/12 23:23 Value #4
	2020-10-12 23:23:53	> Tags: span.kind=client organiza		method GET route root status_code 500	• • • • • • • • • • • • • • • • • • •	View logs	:00 :30	:00 :30)/12 23:22 7	10/12 23:23 Value #A
		> Tags: span.kind=client organiza		method GET route root status_code 500 ws false Exemplar	• • • • • • • • • • • • • • • • • • •	View logs	:00 :30	:00 :30)/12 23:22 7	10/12 23:23 Value #4
		> Tags: span.kind=client organiza		method GET route root status_code 500 ws false Exemplar traceID 65342230	SpaniD: '	View logs	:00 :30	:00 :30)/12 23:22 7	10/12 23:23 Value #A

Grafana Labs

...and from traces to logs

Tempo			
(tempe	~	~ O × 2 Q 🗎 C ×	Loki ・ C · C · C · C · C · C · C · C · C · C
race ID 467250c2ecf	f3a1c	0.1s 🐵 –	Loglabels ~ {cluster="tns", namespace="tns", pod="app- 7d6766684f-t8d96"} Line limit auto 0.5s ③
- Add query 🕤 Query I	history ① Query Inspector		+ Add query S Query history O Query inspector
HTTP Client 04672	250c2ecf3a1c	Find	Logs
ce Start October 21 20	20, 09:45:12.877 Duration 125.72ms Se	ervices 3 Depth 6 Total Spans 6	
	31.43ms 62.86ms	94.29ms 125.72ms	40
vi × > × »	0ms 31.43ms	62.86ms 94.29ms 125.72m	09:44:50 09:44:55 09:45:00 09:45:05 09:45:10 09:45:15
			- info - debug - error - warning
Ib HTTP Client			
V 🛛 🕒 Ib HTTP GET	D		Time O Unique labels Wrap lines
✓ ● app HTTP			Time onque labers wiap intes
	LITT OFT		Dedup None Exact Numbers Signature
	HTTP GET - root Serv	vice: app Duration: 123.03ms Start Time: 2.04ms	Elip regulte order
	HIIPGET-FOOT Sen	wee app Duration, 123.03ms Start Time, 2.04ms	Flip results order
			Common time app app time/app time app=?ide/766684f==tid96 Limit 1000 (1000 Total bytes 12
	> Tags: component = net/http err	ror stage http:method = GET http:status_code	Common res app tran/app res app-7407666847-tests Limit: 1000 (1000 Total bytes 12
		ror stage http:method = GET http:status_code	Common the upp stp the/app the upp-7467666847-tHate Limit 1000 (1000 Total bytes 12 labels: TatTesset Processed: MB
	> Tags: component = net/http err	ror Have http.method = GET http.status_code	Common ne app ter/app ter app-7467666847-t8496 labels: 745766684 > 2020-10-21 09:45:18 2020-10-21T13;45:18.7632176982 stdout F level-warn traceID=743aba780582878c m
✓ app HTT	> Tags: component = net/http err > Process: client-uuid = 5bc68c06a	ror tope http:method = GET http:status_code 88e1f2d3 cluster ths container = app host.h	Common tes app tes tes app-747766887-8496 Limit 1000 (1000 Totalbytes 12 returned) returned) processed: MB > 2020-10-21 09:45:18 2020-10-21T13:45:18.7632176982 stdout F level-warn trace10-743aba780952780e ="GET / (500) 52.011102ms Response; \"\\\" w: false, Accept-Encoding; gzip nection: close; Content-Type: application/x-www-form-urlencoded; Referer: ht
	 > Tags: component = net/http err > Process: client-uuid = 5bc68c06a >	ror tope http:method = GET http:status_code 88e1f2d3 cluster ths container = app host.h	Common me app app tra/sep me app-7dd7deeeff-table Limit 1000 (1000 Total bytes 12 returned) processed: MB > 2020-10-21 09:45:18 2020-10-21T13:45:18.76:22176982 stdout F level-warn trace1D-7d3haa780582878c m = GET / (500) 52.011102ms Responsel \"\\\\" war falser Accept-Encoding: gripp nmetton: close; Content-Type: application/x-www-form-uflecoedd; Referer: ht p://app/post/ Uber-Trace-Idi 7d3haa780582378c:448820582078c:44820582782:4488478058278
✓ ● ap	> Tags: component = net/http err > Process: client-uuid = 5bc68c06a	ror tope http:method = GET http:status_code 88e1f2d3 cluster ths container = app host.h	Common ne pop app hns/app ne app-7407666847-8439 Limit 1000 (1000 Total bytes 12 returned) processed: MB > 2020-10-21 09:45:18 2020-10-21T13:45:18.7632176982 stdout F level+warn tracelD=743aba780582878c = ="GET / (300) 52.01102ms Response; \"\\\\" wa: false; Accept=Encoding: grip; nnection: close; Content=Type: application/x+wow-form-urlencoded; Referen: ht p://app/post; Uber-Trace-Id: 743aba780582878c:44b8269e2607d2a:743aba78058287 1; User-Agent: Go-http-client/li; *
✓ ● ap	 > Tags: component = net/http err > Process: client-uuid = 5bc68c06a >	ror tope http:method = GET http:status_code 88e1f2d3 cluster ths container = app host.h	Common ne pop app tms/ppp nm app-7867666857-18438 Limit 1000 (1000 Total bytes 12 returned) processed: MB > 2020-10-21 09:45:18 2020-10-21T13:45:18.7632176982 stdout F level+warn traceID=743aba780582878c m ="GET / (500) 52.01102ms Response; \"\\\\\" wa: false; Accept=Encoding: grip; nnection: close; Content=Type: application/www-form-urlencoded; Referen: ht p://app/post; Uber-Trace-Id: 743aba780582878c:44b85269e2607d2a:743aba78058287 1; User-Agent: Go-http-client/li; "
✓ ● ap	> Tags: component = net/http err > Process: client-uuid = 5bc68c06a	ror tope http:method = GET http:status_code 88e1f2d3 cluster ths container = app host.h	Common ms app app hms/app ms app-7437666847-8439 Limit 1000 (1000 Total bytes 12 labels: returned) processed: Mg > 2020-10-21 09:45:18 2020-10-21113:45:18.7632176982 tadout F level=warn traceID=743aba78058278 m metion:
✓ ● ap	> Tags: component = net/http err > Process: client-uuid = 5bc68c06a	ror tope http:method = GET http:status_code 88e1f2d3 cluster ths container = app host.h	Common ms pps app imm/spp imm ppp=7df76fffff-1659K Limit 1000 (1000 Total bytes 12 labels: returned) processed: Mg > 2020-10-21 03:45:18 2020-10-21T13:45:18.763217698Z stdout F level-warn traceID=7d3aba780582878c m mg = "GET / (500) 52.01102ms Response: \n'.\n\n'* ws: false; Accept=Encoded; Referer: ht p://app/post; Uber-Tacerici 7d3aba780582878c:d4b85269207d2a:7d3aba78058287 j://app/post; Uber-Tacerici 7d3aba780582878c:d4b85269207d2a:7d3aba78058287 j:/app.col: > 2020-10-21 09:45:18 2020-10-21T13:45:18.763153007Z stdout F level=eror msg=#HTTP cquest faild* tus=300 body= > 2020-10-21 09:45:18 2020-10-21T13:45:18.76315007Z stdout F level=info msg=#HTTP client success* tus=300 body= > 2020-10-21 09:45:18 2020-10-21T13:45:18.76315007Z stdout F level=info msg=#HTTP client success*
✓ ● ap	> Tags: component = net/http err > Process: client-uuid = 5bc68c06a	ror tope http:method = GET http:status_code 88e1f2d3 cluster ths container = app host.h	Common ms pps app imm/spp imm ppp=7df76fffff-1659K Limit 1000 (1000 Total bytes 12 labels: returned) processed: Mg > 2020-10-21 03:45:18 2020-10-21T13:45:18.763217698Z stdout F level-warn traceID=7d3aba780582878c m mg = "GET / (500) 52.01102ms Response: \n'.\n\n'* ws: false; Accept=Encoded; Referer: ht p://app/post; Uber-Tacerici 7d3aba780582878c:d4b85269207d2a:7d3aba78058287 j://app/post; Uber-Tacerici 7d3aba780582878c:d4b85269207d2a:7d3aba78058287 j:/app.col: > 2020-10-21 09:45:18 2020-10-21T13:45:18.763153007Z stdout F level=eror msg=#HTTP cquest faild* tus=300 body= > 2020-10-21 09:45:18 2020-10-21T13:45:18.76315007Z stdout F level=info msg=#HTTP client success* tus=300 body= > 2020-10-21 09:45:18 2020-10-21T13:45:18.76315007Z stdout F level=info msg=#HTTP client success*
✓ ● ap	> Tags: component = net/http err > Process: client-uuid = 5bc68c06a	ror tope http:method = GET http:status_code 88e1f2d3 cluster ths container = app host.h	Common Test peop T
∽ <mark>0</mark> ap	> Tags: component = net/http err > Process: client-uuid = 5bc68c06a	ror tope http:method = GET http:status_code 88e1f2d3 cluster ths container = app host.h	Common hales HER #02
 ✓ ● ap 	> Tags: component = net/http err > Process: client-uuid = 5bc68c06a	ror tope http:method = GET http:status_code 88e1f2d3 cluster ths container = app host.h	Common me app app inn/mop inn app-7407666664-1899 Limit 1000 (1000 Total bytes 12 returned) returned) processed: MB









Grafana Labs



State timeline with time series + thresholds





Status history with time series + thresholds

	10.4	16.8	10.9	18.3	25.2	33.6	27.1	15.7	25.3	18.4	7.68	7.63
	10.1	15.2	9.79	14.2	7.64	5.68	-1.0	4.76	2.69	7.69	3.26	-2.0
	11.6	6.29	6.92	12.8	4.74	12.4	4.02	0.252	3.99	12.6	4.45	15.1
Bedroom	14.6	11.2	8.55	10.8	15.2	20.1	24.6	25.4	21.4	29.2	36.7	34.3
	11:59	12:29	12:59	13:29	13:59	14:29	14:59	15:29	15:59	16:29	16:59	17:29



Base layer ArcGIS wold imagery



Heatmap data layer





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

GG

(But we will also sell it to you happily)







Thank you!

richih@richih.org

https://chaos.social/@RichiH

https://github.com/RichiH/talks





