

Observability 2.0 feat. OpenTelemetry

by Przemek Maciołek

cover by Marcin Stożek







sumo logic

About me Przemek

- 15+ years in IT
- PhD in ML/NLP
- Data Science, Big Data, Cloud Architecture, Databases, etc.
- Worked for and founded several startups
- VP of R&D at Collective Sense since 2015; acquired in 2019 by Sumo Logic
- OpenTelemetry contributor
- 🚴 📷 🐱 🧑 🧑 🧑
- <https://www.linkedin.com/in/pmaciolek/>
- Note: opinions here are my own



About me

- 14+ years in IT
- ~~PhD in ML/NLP~~
- ~~Data Science, Big Data, Cloud Architecture, Databases, etc.~~
- ~~Worked for and founded several startups~~
- ~~VP of R&D at Collective Sense since 2015; acquired in 2019 by Sumo Logic~~
- ~~OpenTelemetry contributor~~
- ~~     ~~
- <https://www.linkedin.com/in/pmaciolek/>
- Note: opinions here are Przemek's (and my own)







Charity Majors
@mipsytipsy

...

Updated definition:

 Monitoring is for running and understanding other people's code (aka "your infrastructure")

 Observability is for running and understanding **your** code -- the code you write, change and ship every day; the code that solves your core business problems.

 **Charity Majors** @mipsytipsy · Sep 23, 2017

Monitoring is for operating software/systems
Instrumentation is for writing software
Observability is for understanding systems

[Show this thread](#)

8:48 AM · Sep 14, 2020 · Twitter Web App

<https://twitter.com/mipsytipsy/status/1305398051842871297>

“A software system with a capability to allow a human to ask and answer questions“

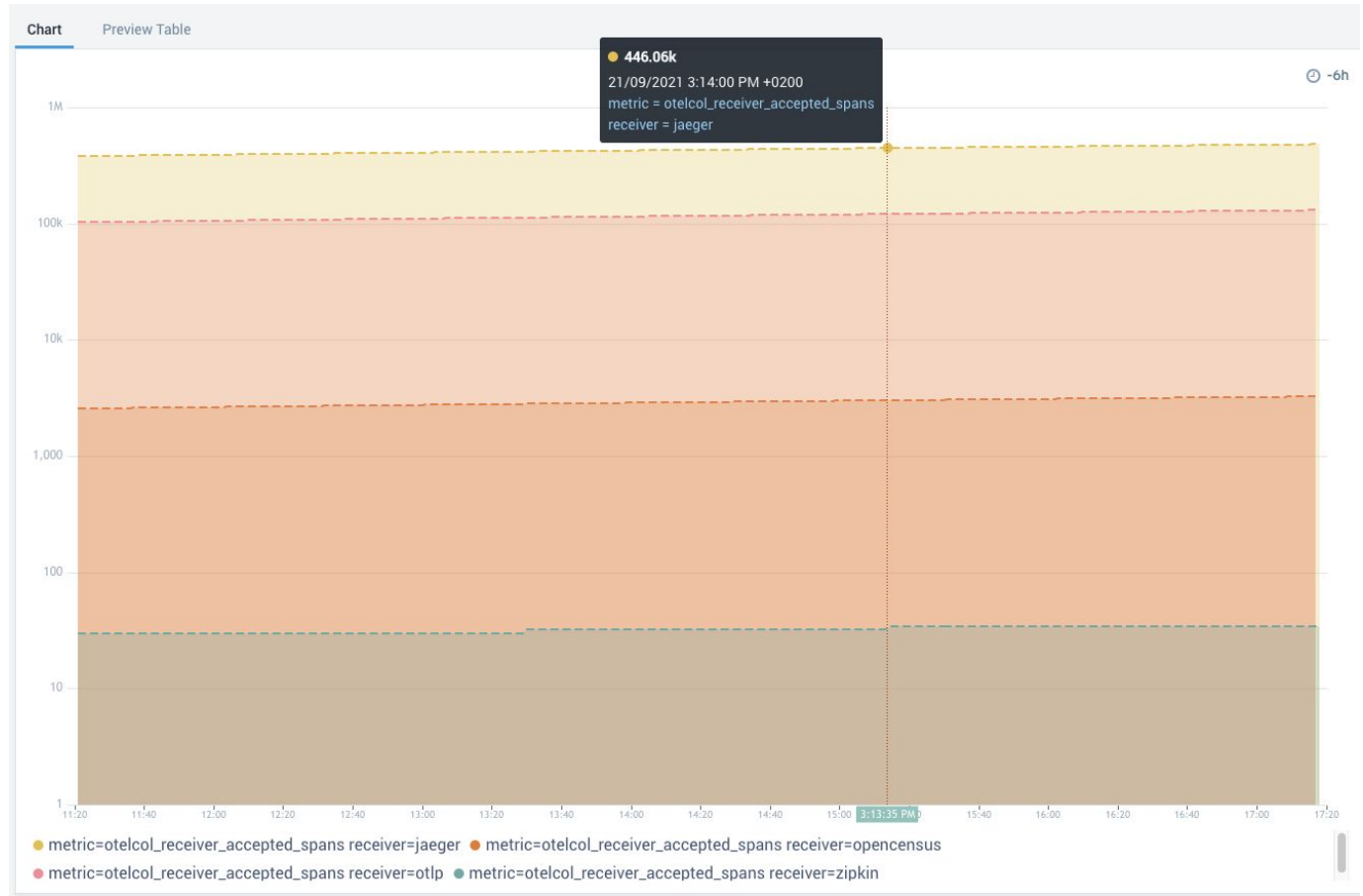
(Yuri Shkuro)

Traditionally, three pillars:

- **metrics**
- **logs**
- **traces**

**Observability is for monitoring
what
devops is for operations**

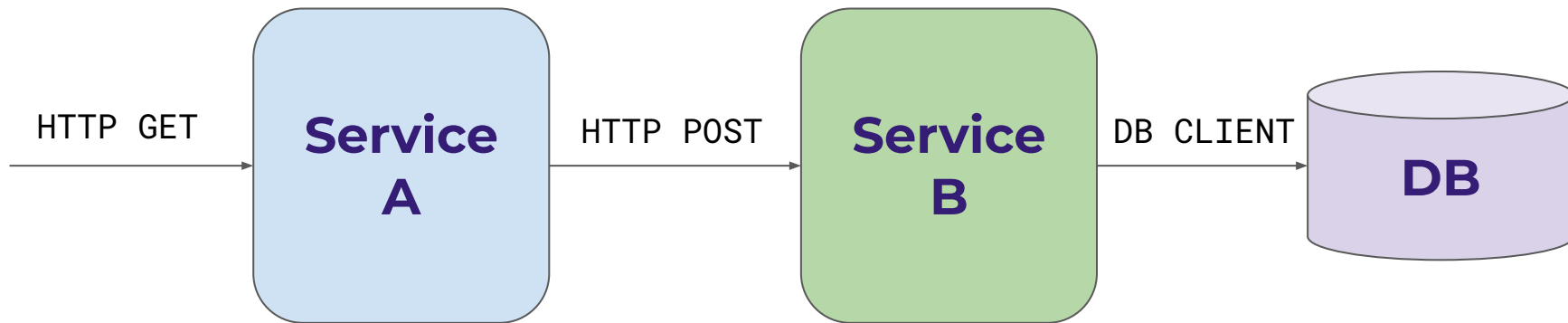
Metrics



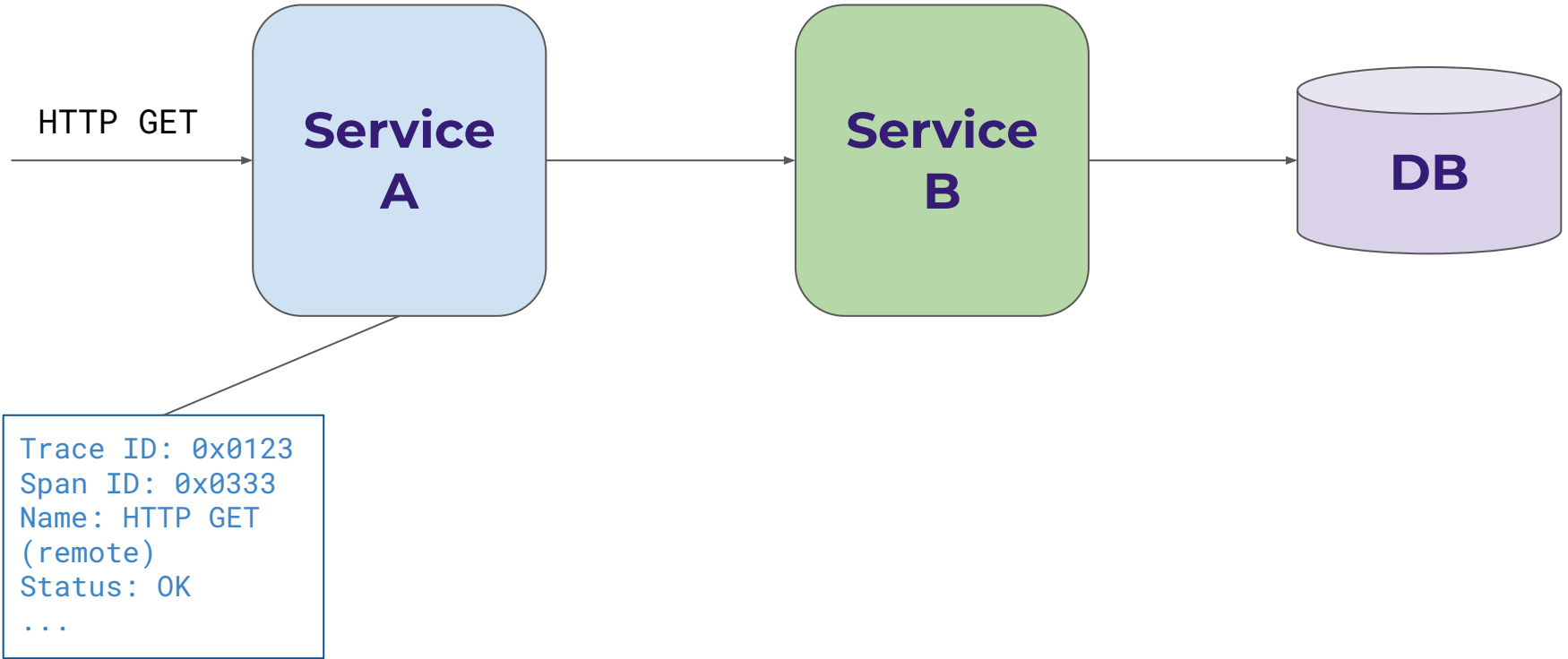
Logs

```
2021-09-21 15:11:44,345 - werkzeug - INFO - 10.0.116.67 - - [21/Sep/2021 15:11:44] "\u001B[33mPOST /order
HTTP/1.1\u001B[0m\" 404 - -
2021-09-21 15:11:45,206 - root - INFO - Preparing espresso coffee
2021-09-21 15:11:46,269 - root - INFO - Get product price: cornetto
2021-09-21 15:11:45,024 - werkzeug - INFO - 10.0.58.218 - - [21/Sep/2021 15:11:45] \"OPTIONS /order HTTP/1.1\" 200 - -
2021-09-21 15:11:45,246 - root - ERROR - Missing some ingredients
2021-09-21 15:11:46,270 - root - INFO - Query DB for price of product: cornetto
2021-09-21 15:11:45,074 - root - INFO - Check if tiramisu is available
2021-09-21 15:11:46,272 - root - ERROR - FATAL: remaining connection slots are reserved for non-replication superuser
connections
...
```

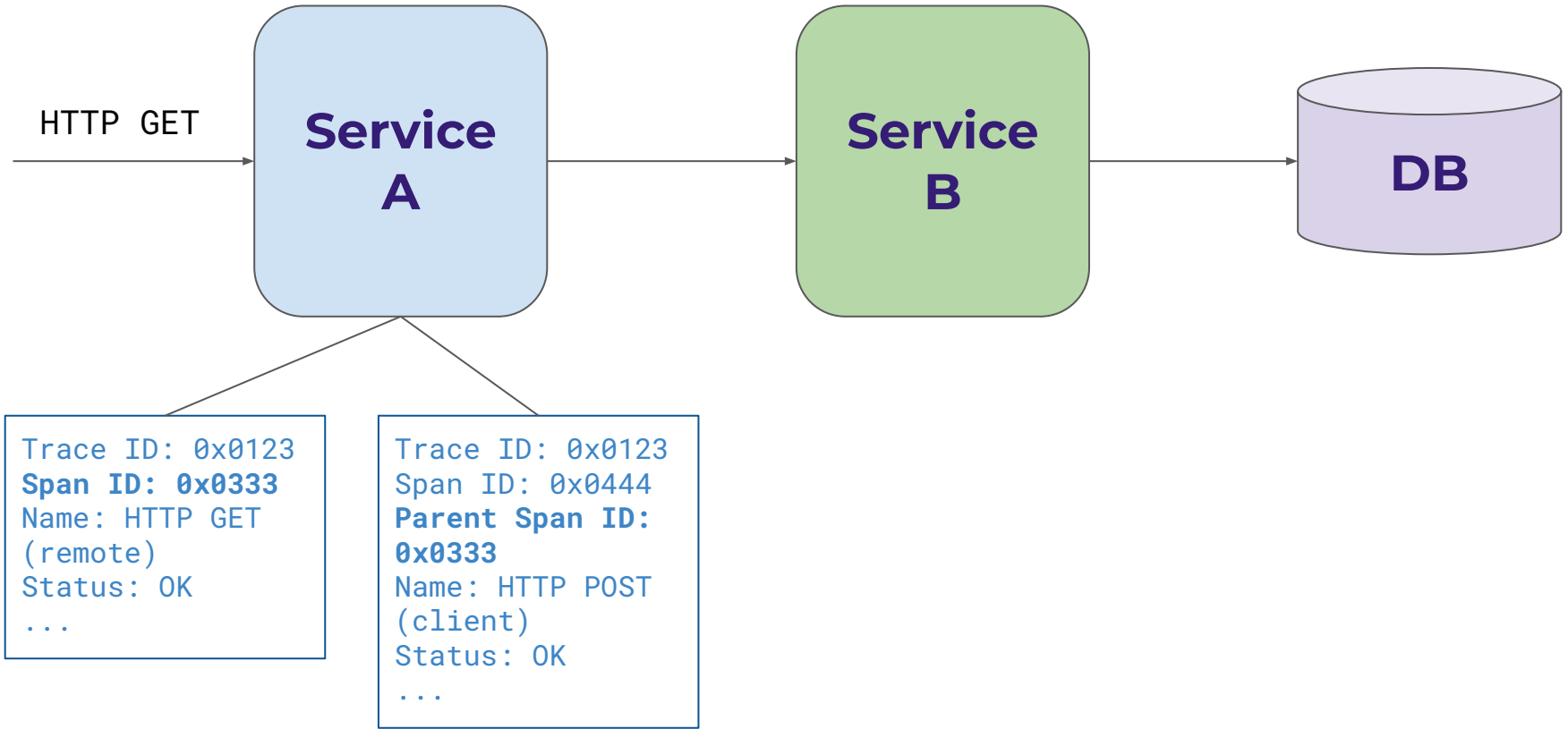
Traces



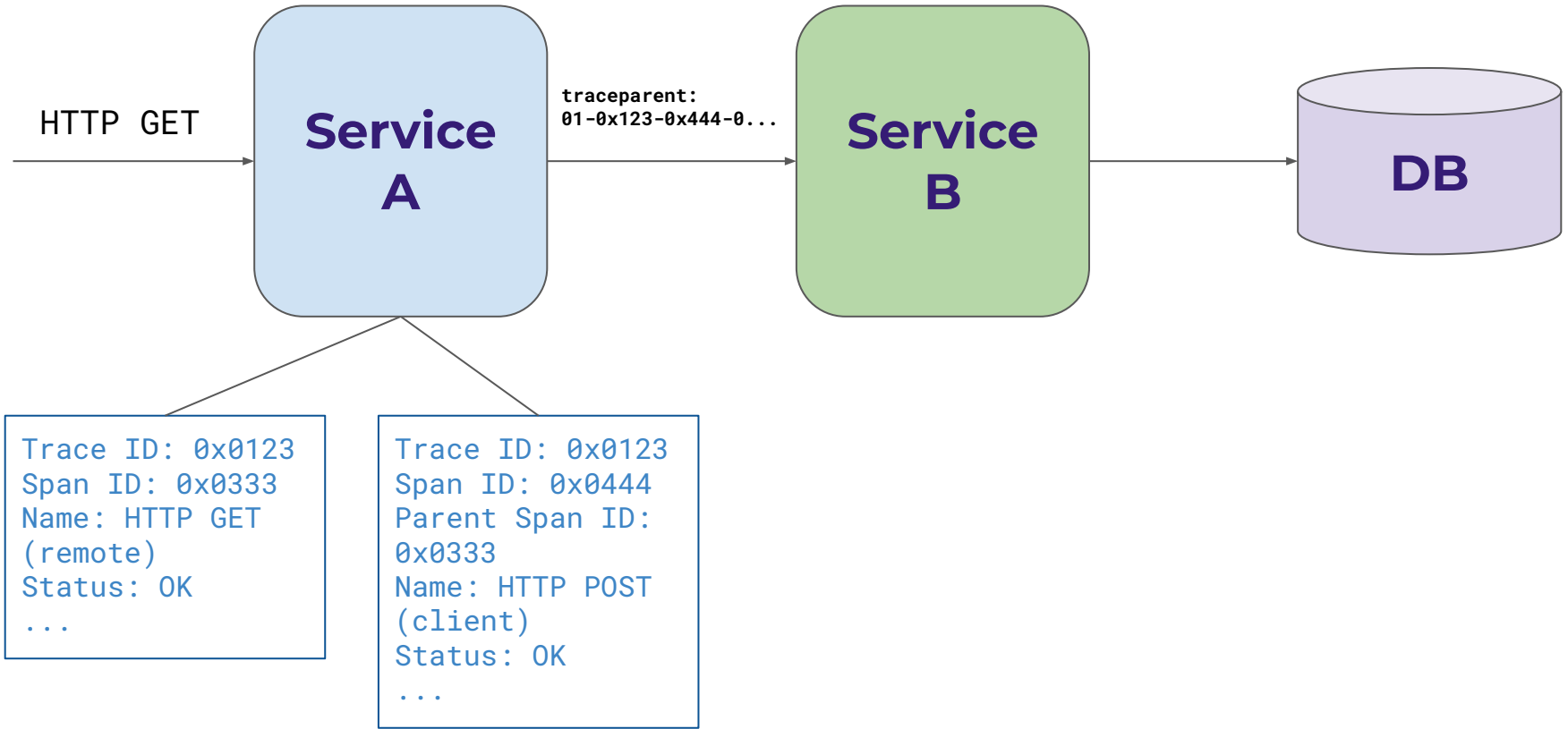
Traces



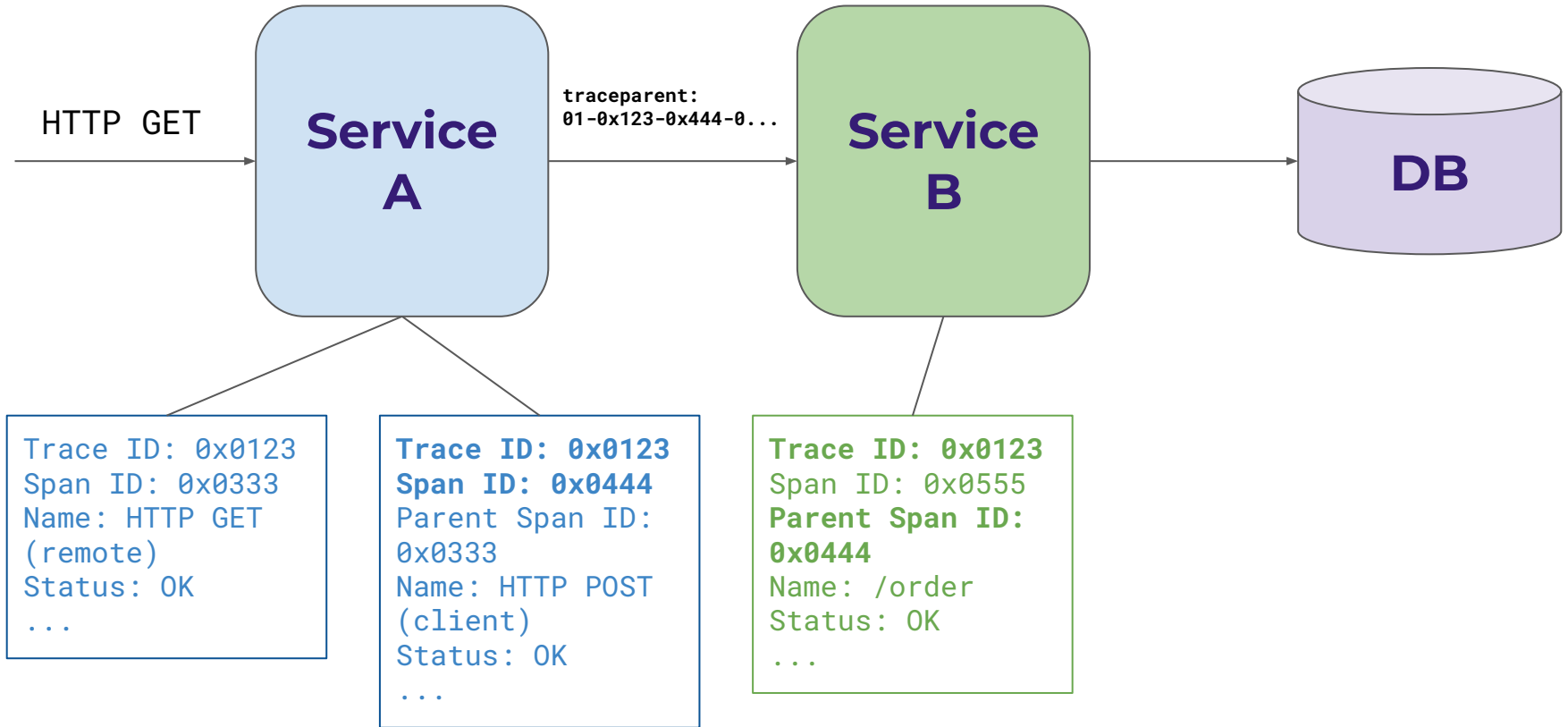
Traces



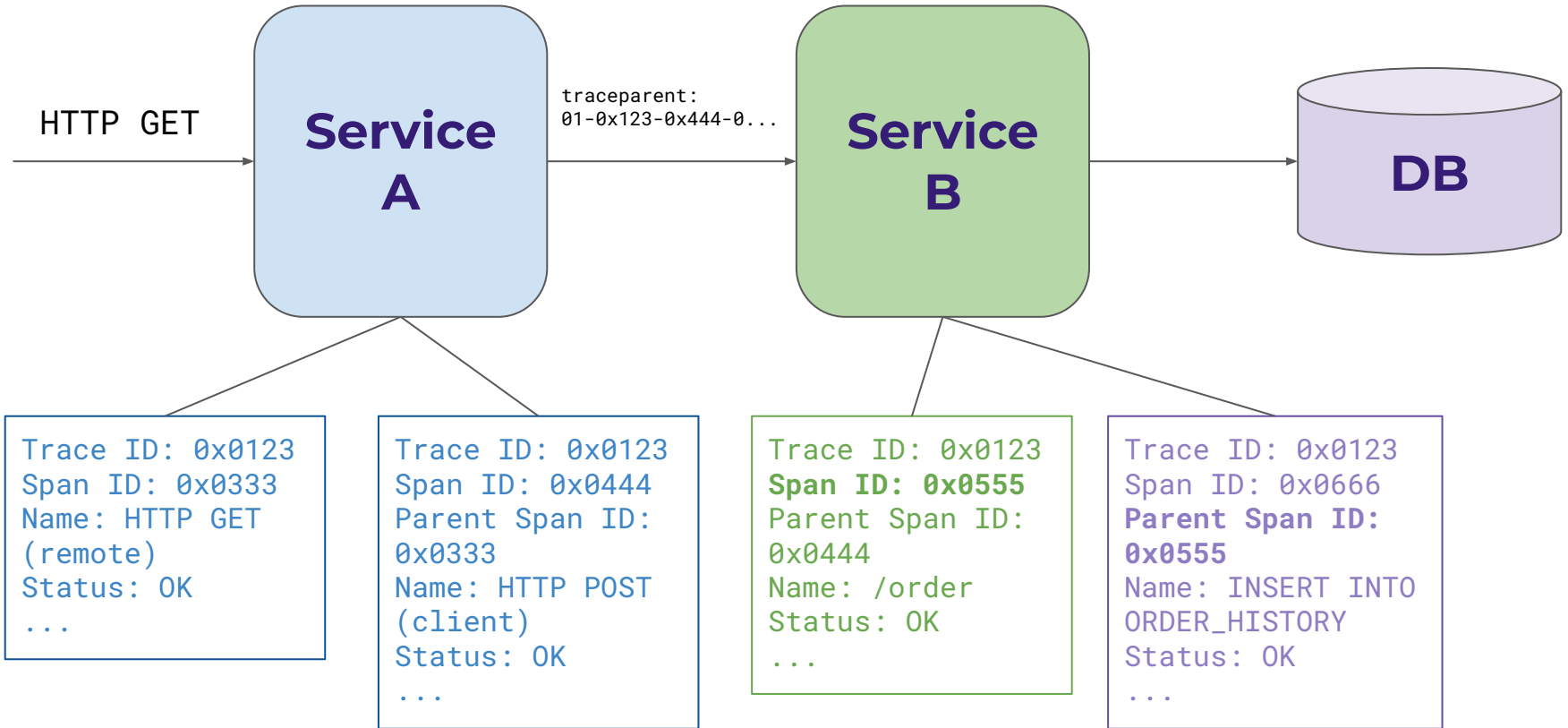
Traces

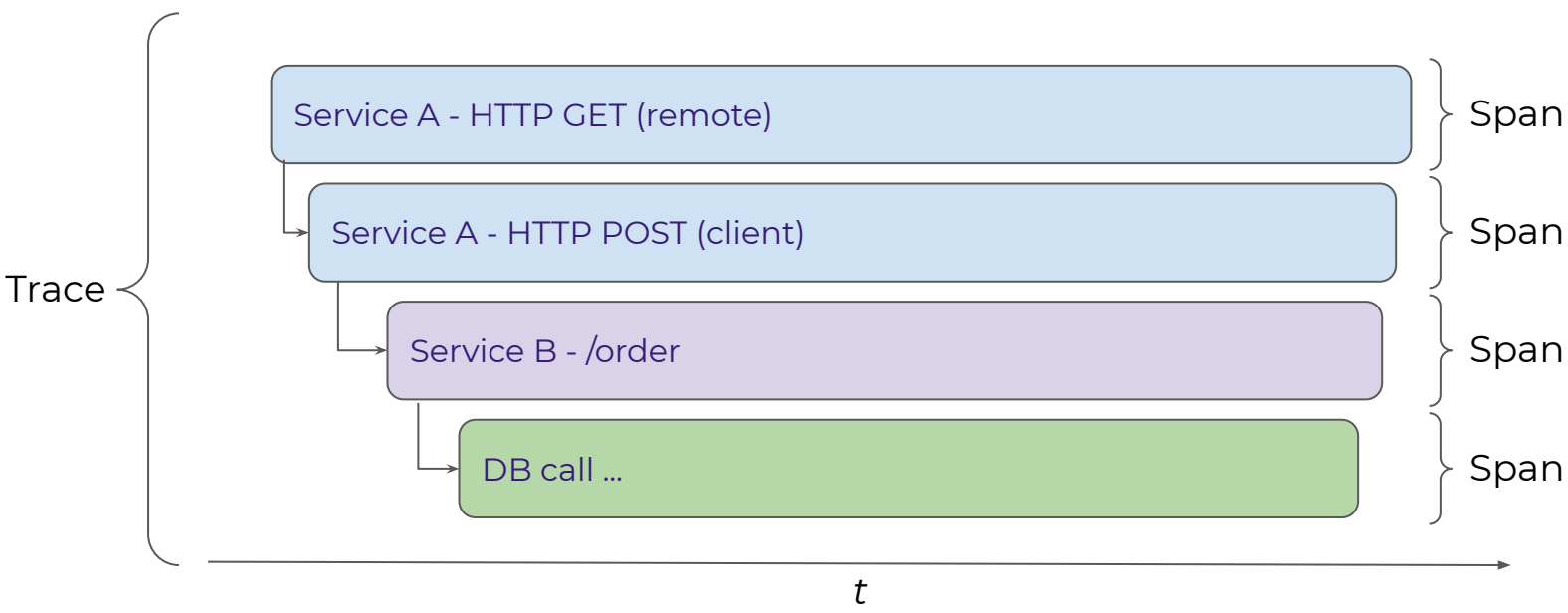


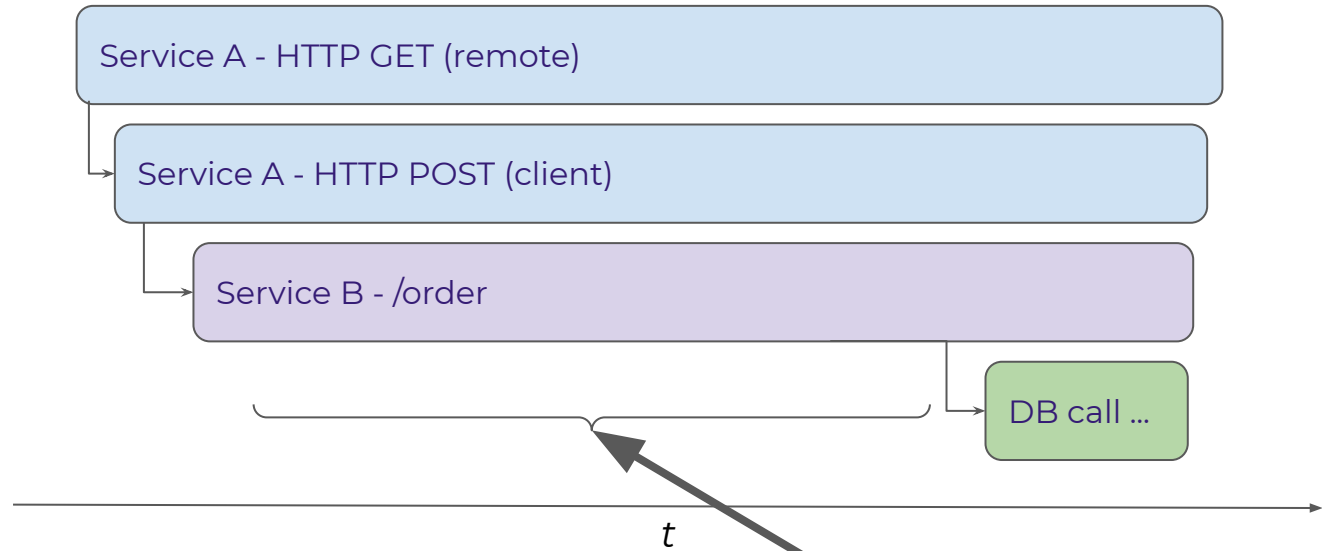
Traces



Traces







What happened here?

Maybe logs or metrics could help?

Summary

Metadata

Entities

application

aws.ecs.cluster.arn

aws.ecs.launchtype

aws.ecs.task.arn

aws.ecs.task.family

aws.ecs.task.revision

cloud.account.id

cloud.availability_zone

cloud.platform

cloud.provider

cloud.region

aws-ecs-coffee-bar

arn:aws:ecs:us-west-2:12345667890:cluster/sumologic

fargate

arn:aws:ecs:us-west-2:1234567890:task/sumologic/caaabbccdddee

collector-aws-otel-collector-fargate

9

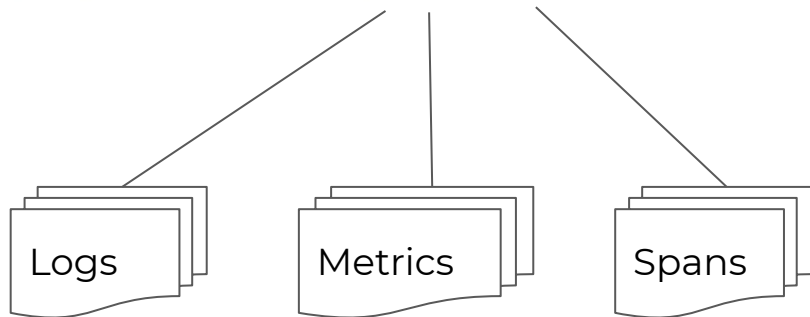
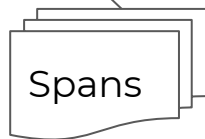
1234567890

us-west-2a

aws_ecs

aws

us-west-2



Bringing this all together

View as Raw

```
{
  timestamp: 1632237607594,
  message: "2021-09-21 15:20:07,594 - root - WARNING - Sweet: tiramisu is not available - trace_id=828cdd67e9dd56750aeeee5b84aec25b - span_id=1d040a884181999c",
  requestID: "INFO",
  logStream: "2021/09/21/[$LATEST]3796125db1a549eea326b3a463821fd8",
  logGroup: "/aws/lambda/SweetsFunction"
}
```

Host: /aws/lambda/SweetsFunction ▾ Name: 2021/09/21/[\$LATEST]3796125db1a549eea326b3a463821fd8 ▾ Category: aws/observability/cloudwatch/logs ▾

View as Raw

```
{
  timestamp: 1632237607534,
  message: "2021-09-21 15:20:07,534 - root - INFO - Call Sweets stock service - trace_id=828cdd67e9dd56750aeeee5b84aec25b - span_id=4ba240f27eda9606",
  requestID: "61744919-f297-44e0-add1-8b8b7b0b098d",
  logStream: "2021/09/21/[$LATEST]e709e86d65354054a42e21066be66b75",
  logGroup: "/aws/lambda/CheckSweetsFunction"
}
```

Host: /aws/lambda/CheckSweetsFunction ▾ Name: 2021/09/21/[\$LATEST]e709e86d65354054a42e21066be66b75 ▾ Category: aws/observability/cloudwatch/logs ▾

View as Raw

```
{
  timestamp: 1632237607534,
  message: "2021-09-21 15:20:07,533 - root - INFO - Got request to check if \"tiramisu\" is available - trace_id=828cdd67e9dd56750aeeee5b84aec25b - span_id=4ba240f27eda9606",
  requestID: "61744919-f297-44e0-add1-8b8b7b0b098d",
  logStream: "2021/09/21/[$LATEST]e709e86d65354054a42e21066be66b75",
  logGroup: "/aws/lambda/CheckSweetsFunction"
}
```

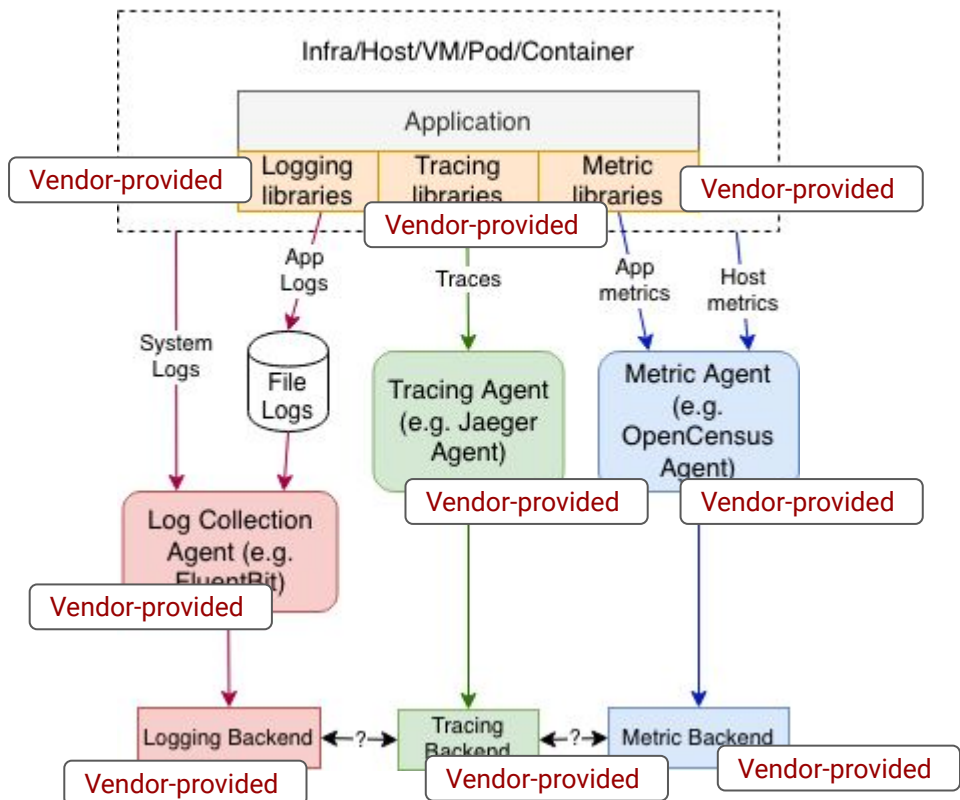
Host: /aws/lambda/CheckSweetsFunction ▾ Name: 2021/09/21/[\$LATEST]e709e86d65354054a42e21066be66b75 ▾ Category: aws/observability/cloudwatch/logs ▾

Not a new problem

- Prometheus
- Grafana
- Fluentd/Fluent Bit
- ELK
- Jaeger
- Zipkin
- OpenTracing
- Vector
- ...

Observability 1.0

Separate Collection



Instrumentation

Collector

Backend



OpenCensus:

- metrics and tracing focused
- originated at Google, based on Census concepts
- Omnicore started incorporating it into a complete observability solution



OpenTracing:

- distributed-tracing focused
- originated at Google, based on Dapper concepts
- CNCF project since 2016
- API used by many vendors (Jaeger, DataDog, etc.)



OpenCensus:

- metrics and tracing focused
- originated at Google, based on Census concepts
- Omnicore started incorporating it into a complete observability solution



OpenTracing:

- distributed-tracing focused
- originated at Google, based on Dapper concepts
- CNCF project since 2016
- API used by many vendors (Jaeger, DataDog, etc.)



OpenTelemetry:

- merge of OpenCensus + OpenTracing
- announced May 2019
- backed by all major vendors
- CNCF project (incubating since Aug 2021)

Former rivals at OpenTelemetry lock in tracing specification, to focus on metrics next

By **Julia Schmidt** - October 22, 2020

The future of tracing is open



Ilan Rabinovitch

@irabinovitch

Published: September 12, 2019

OpenTelemetry: Future-Proofing Your Instrumentation



By **John Watson and Lavanya Chockalingam** · Jun. 22nd, 2020 · **New Relic News and Products**

observability, open instrumentation, OpenTelemetry, telemetry

December 17, 2020 | By **Dave Sudia**

Everywhere in One Place: OpenTelemetry and Observability in Sumo Logic

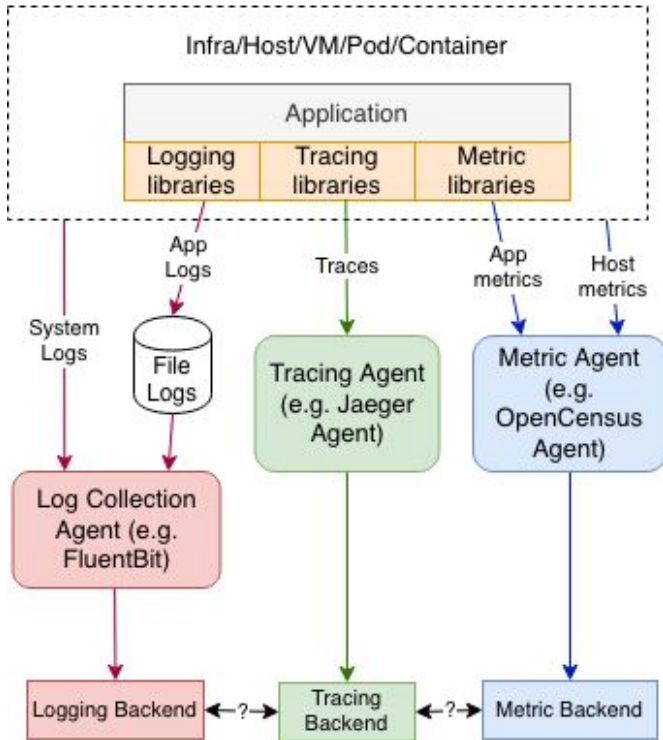
<https://devclass.com/2020/10/22/opentelemetry-tracing-spec-rc/>

<https://www.datadoghq.com/blog/opentelemetry-instrumentation/>

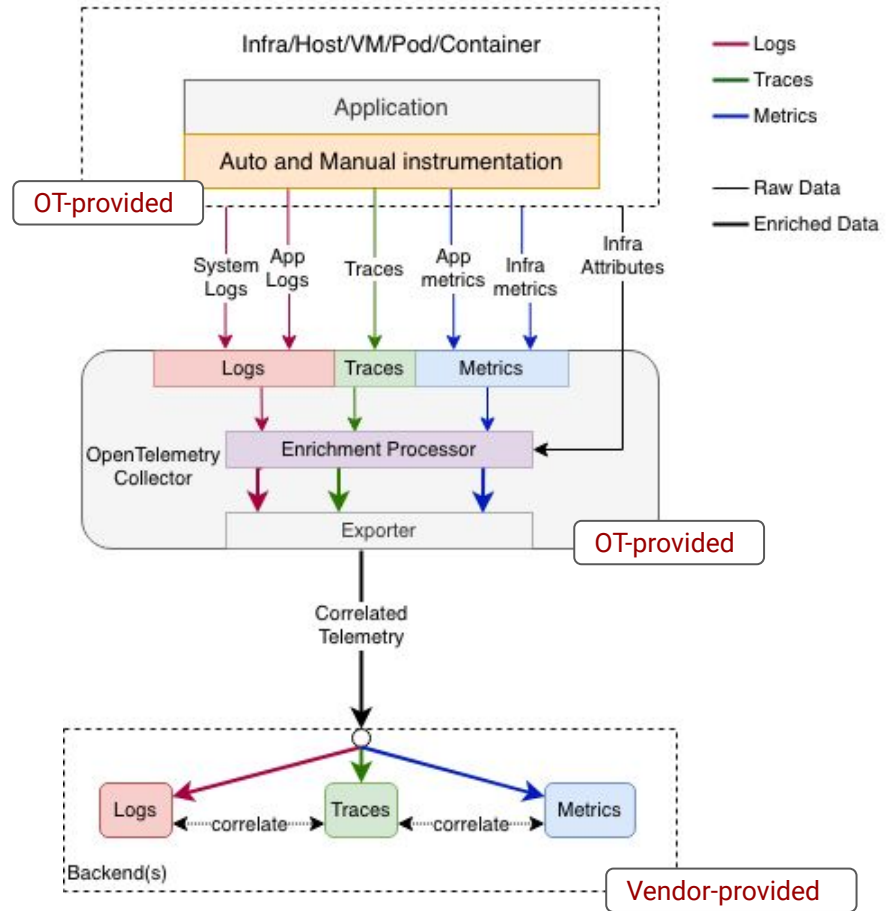
<https://blog.newrelic.com/product-news/what-is-opentelemetry/>

Observability 1.0 vs 2.0 (aka "the promise")

Separate Collection



OpenTelemetry Collection



<https://github.com/open-telemetry/opentelemetry-specification/blob/master/specification/logs/overview.md>

The components

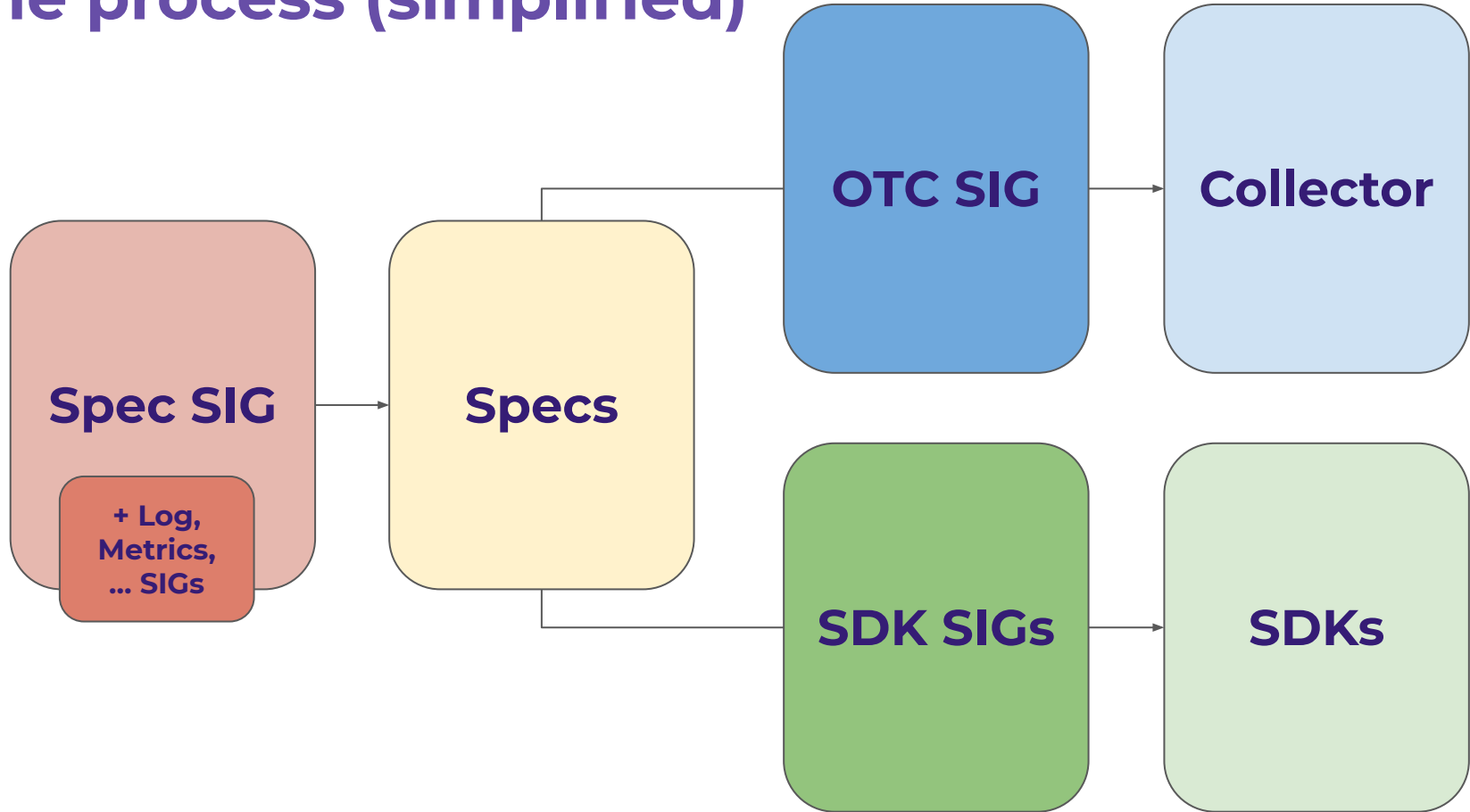


Specs

SDKs

Collector

The process (simplified)

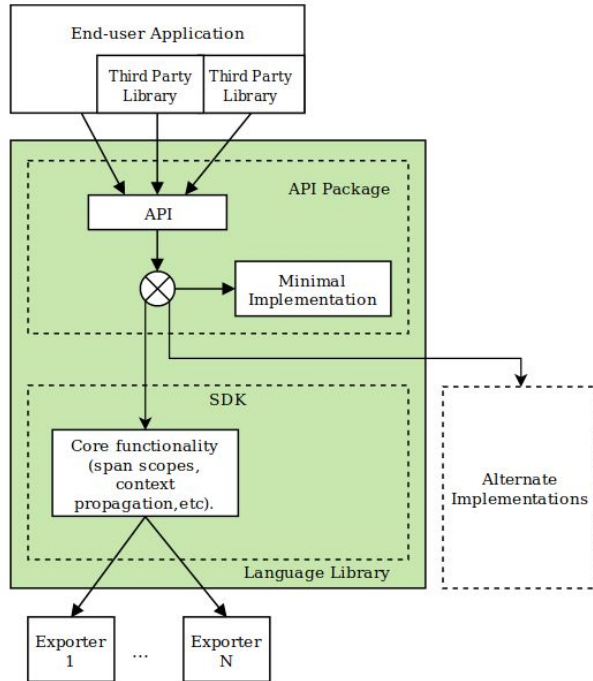


OpenTelemetry Specs

- Specification, including guidelines, API, SDK, semantic conventions
<https://github.com/open-telemetry/opentelemetry-specification>
- OTEPS (Enhancement Proposals) - for discussing any major changes
<https://github.com/open-telemetry/oteps>
- Proto - language independent interface types
<https://github.com/open-telemetry/opentelemetry-proto>

Language Library Generic Design

Here is a generic design for a language library (arrows indicate calls):



Expected Usage

The OpenTelemetry Language Library is composed of 2 packages: API package and SDK package. In this specification, *package* is used as a conceptual separation and does not prescribe the exact structure of the artifacts making up the language implementations. Whether the API and SDK packages are bundled as two all-in-one artifacts or split across multiple ones (e.g. one for api-trace, one for api-metric, one

AWS ECS

type: `aws.ecs`

Description: Resources used by AWS Elastic Container Service (ECS).

Attribute	Type	Description	Examples	Required
<code>aws.ecs.container.arn</code>	string	The Amazon Resource Name (ARN) of an ECS container instance .	<code>arn:aws:ecs:us-west-1:123456789123:container/32624152-9086-4f0e-acae-1a75b14fe4d9</code>	No
<code>aws.ecs.cluster.arn</code>	string	The ARN of an ECS cluster .	<code>arn:aws:ecs:us-west-2:123456789123:cluster/my-cluster</code>	No
<code>aws.ecs.launchtype</code>	string	The launch type for an ECS task.	<code>EC2 ; Fargate</code>	No
<code>aws.ecs.task.arn</code>	string	The ARN of an ECS task definition .	<code>arn:aws:ecs:us-west-1:123456789123:task/10838bed-421f-43ef-870a-f43feacbbb5b</code>	No
<code>aws.ecs.task.family</code>	string	The task definition family this task definition is a member of.	<code>opentelemetry-family</code>	No

`aws.ecs.launchtype` MUST be one of the following:

Value	Description
EC2	ec2
Fargate	fargate

APIs and SDKs

Auto- and manual- instrumentation libraries, including:

- Java
- Ruby
- Swift
- Rust
- JavaScript
- Python
- C++
- Erlang
- Go
- .NET
- PHP

Traces

Feature	Optional	Go	Java	JS	Python	Ruby	Erlang	PHP	Rust	C++	.NET	Swift
TracerProvider												
Create TracerProvider		+	+	+	+	+	+	+	+	+	+	+
Get a Tracer		+	+	+	+	+	+	+	+	+	+	+
Get a Tracer with schema_url		+										
Safe for concurrent calls		+	+	+	+	+	+	+	+	+	+	+
Shutdown (SDK only required)		+	+	+	+	+	-		+	+	+	+
ForceFlush (SDK only required)		+	+	-	+	+	-		+	+	+	+
Trace / Context interaction												
Get active Span		N/A	+	+	+	+	+		+	+	+	+
Set active Span		N/A	+	+	+	+	+		+	+	+	+
Tracer												
Create a new Span		+	+	+	+	+	+	+	+	+	+	+
Get active Span		N/A	+	+	+	+	+	+	+	+	+	+
Mark Span active		N/A	+	+	+	+	+	+	+	+	+	+
Safe for concurrent calls		+	+	+	+	+	+	+	+	+	+	+
SpanContext												
IsValid		+	+	+	+	+	+	+	+	+	+	+
IsRemote		+	+	+	+	+	+	+	+	+	+	+
Conforms to the W3C TraceContext spec		+	+	+	+	+	+		+	+	+	+
Span												

+ <https://opentelemetry.io/status/>

<https://github.com/open-telemetry/opentelemetry-specification/blob/master/spec-compliance-matrix.md>

A trivial Golang http server..

```
func main() {
    r := mux.NewRouter()
    r.HandleFunc("/users/{id:[0-9]+}", func(w http.ResponseWriter, r *http.Request) {
        name := getUser(r.Context(), mux.Vars(r)["id"])
        reply := fmt.Sprintf("user %s (id %s)\n", name, id)
        _, _ = w.Write([]byte(reply))
    })
    http.Handle("/", r)
    _ = http.ListenAndServe(":8080", nil)
}
```

```
func getUser(ctx context.Context, id string) string {
    if id == "123" {
        return "otelmux tester"
    }
    return "unknown"
}
```


+ tracing

```
import (  
    "go.opentelemetry.io/contrib/instrumentation/github.com/gorilla/mux/otelmux"  
    ...  
)  
  
var tracer = otel.Tracer("mux-server")  
  
func main() {  
    shutdown := helper.InitTracer("demo-server")  
    defer shutdown()  
    r := mux.NewRouter()  
    r.Use(otelmux.Middleware("my-server"))  
    r.HandleFunc("/users/{id:[0-9]+}", func(w http.ResponseWriter, r *http.Request) {  
        name := getUser(r.Context(), mux.Vars(r)["id"])  
        reply := fmt.Sprintf("user %s (id %s)\n", name, id)  
        _, _ = w.Write([]byte(reply))  
    })  
    http.Handle("/", r)  
    _ = http.ListenAndServe(":8080", r)  
}  
  
func getUser(ctx context.Context, id string) string {  
    _, span := tracer.Start(ctx, "getUser", oteltrace.WithAttributes(attribute.String("id", id)))  
    defer span.End()  
    if id == "123" {  
        return "otelmux tester"  
    }  
    return "unknown"  
}
```

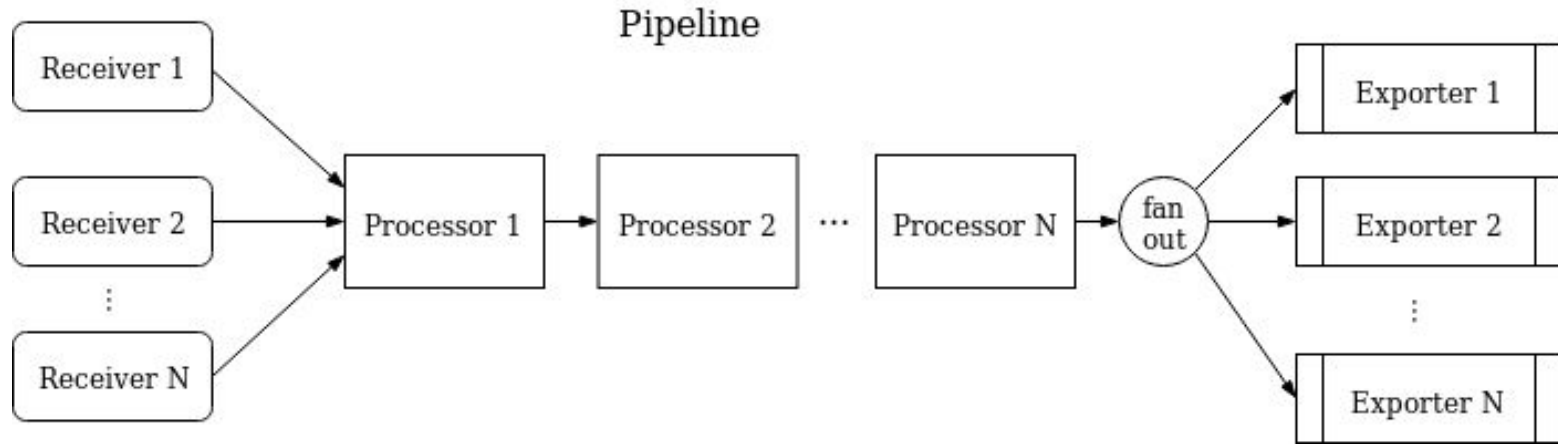
+ tracing + metrics

```
import (  
    ...  
    otelmetric "go.opentelemetry.io/otel/metric"  
    ...  
)  
  
var tracer = otel.Tracer("mux-server")  
var meter = global.Meter("demo-meter")  
  
func main() {  
    shutdown := helper.InitTracer("demo-server")  
    defer shutdown()  
    metricShutdown := helper.InitMeter()  
    defer metricShutdown()  
  
    userCounter := otelmetric.Must(meter).NewInt64Counter("users_req_count",  
        otelmetric.WithDescription("Number of requests to /users"))  
  
    r := mux.NewRouter()  
    r.Use(otelmux.Middleware("my-server"))  
    r.HandleFunc("/users/{id:[0-9]+}", func(w http.ResponseWriter, r *http.Request) {  
        userCounter.Add(context.Background(), 1)  
        name := getUser(r.Context(), mux.Vars(r)["id"])  
        reply := fmt.Sprintf("user %s (id %s)\n", name, id)  
        _, _ = w.Write([]byte(reply))  
    })  
    http.Handle("/", r)  
    _ = http.ListenAndServe(":8080", r)  
}
```

+ tracing
+ metrics
+ logs

```
yourFavoriteLoggingLibrary.log("hello world")
```

OpenTelemetry Collector



Receivers: Prometheus, OTLP, Jaeger, Kafka, Fluentforward, Host metrics, Log File Receiver, AWS XRay, JMX, Nginx, SignalFX, Splunk, Carbon, Collectd, Dockerstats, ...

Processors: Attributes, Filter, Span, Sampling, K8s, Resource Detection, Metrics Transformation, ...

Exporters: File, Logging, OTLP, Prometheus, Carbon, Zipkin, Jager, vendor-specific (DataDog, Dynatrace, Honeycomb, Logz.IO, NewRelic, ... Sumologic), ...

OTLP Receiver

Receives data via gRPC or HTTP using [OTLP](#) format.

Supported pipeline types: traces, metrics, logs

⚠️ OTLP metrics format is currently marked as "Alpha" and may change in incompatible way any time.

Getting Started

All that is required to enable the OTLP receiver is to include it in the receiver definitions. A protocol can be disabled by simply not specifying it in the list of protocols.

```
receivers:
  otlp:
    protocols:
      grpc:
      http:
```

The following settings are configurable:

- `endpoint` (default = 0.0.0.0:4317 for grpc protocol, 0.0.0.0:4318 http protocol): host:port to which the receiver is going to receive data. The valid syntax is described at <https://github.com/grpc/grpc/blob/master/doc/naming.md>.

Advanced Configuration

Several helper files are leveraged to provide additional capabilities automatically:

Attributes Processor

Supported pipeline types: traces, logs.

The attributes processor modifies attributes of a span. Please refer to [config.go](#) for the config spec.

It optionally supports the ability to [include/exclude spans](#).

It takes a list of actions which are performed in order specified in the config. The supported actions are:

- `insert` : Inserts a new attribute in spans where the key does not already exist.
- `update` : Updates an attribute in spans where the key does exist.
- `upsert` : Performs insert or update. Inserts a new attribute in spans where the key does not already exist and updates an attribute in spans where the key does exist.
- `delete` : Deletes an attribute from a span.
- `hash` : Hashes (SHA1) an existing attribute value.
- `extract` : Extracts values using a regular expression rule from the input key to target keys specified in the rule. If a target key already exists, it will be overridden. Note: It behaves similar to the Span Processor `to_attributes` setting with the existing attribute as the source.

For the actions `insert`, `update` and `upsert`,

- `key` is required
- one of `value` or `from_attribute` is required
- `action` is required.

```
# Key specifies the attribute to act upon.
- key: <key>
  action: {insert, update, upsert}
# Value specifies the value to populate for the key.
# The type is inferred from the configuration.
  value: <value>

# Key specifies the attribute to act upon.
- key: <key>
  action: {insert, update, upsert}
# FromAttribute specifies the attribute from the span to use to populate
# the value. If the attribute doesn't exist, no action is performed.
  from_attribute: <other key>
```

For the `delete` action

OTLP gRPC Exporter

Exports data via gRPC using [OTLP](#) format. By default, this exporter requires TLS and offers queued retry capabilities.

⚠️ OTLP metrics and logs formats are currently marked as "Alpha" and may change in incompatible way any time.

Supported pipeline types: traces, metrics

Getting Started

The following settings are required:

- `endpoint` (no default): host:port to which the exporter is going to send OTLP trace data, using the gRPC protocol. The valid syntax is described [here](#). If a scheme of `https` is used then client transport security is enabled and overrides the `insecure` setting.

By default, TLS is enabled:

- `insecure` (default = `false`): whether to enable client transport security for the exporter's connection.

As a result, the following parameters are also required:

- `cert_file` (no default): path to the TLS cert to use for TLS required connections. Should only be used if `insecure` is set to false.
- `key_file` (no default): path to the TLS key to use for TLS required connections. Should only be used if `insecure` is set to false.

Example:

```
exporters:
  otlp:
    endpoint: otelcol2:4317
    cert_file: file.cert
    key_file: file.key
  otlp/2:
    endpoint: otelcol2:4317
    insecure: true
```

Advanced Configuration

```
receivers:
  otlp:
    protocols:
      grpc:
      http:

processors:
  batch:
  memory_limiter:
    # 75% of maximum memory up to 4G
    limit_mib: 1536
    # 25% of limit up to 2G
    spike_limit_mib: 512
    check_interval: 5s

exporters:
  logging:
    logLevel: debug

service:
  pipelines:
    traces:
      receivers: [otlp]
      processors: [memory_limiter, batch]
      exporters: [logging]
    metrics:
      receivers: [otlp]
      processors: [memory_limiter, batch]
      exporters: [logging]
```


Collector & Misc

- Two OpenTelemetry Collector flavors
 - <https://github.com/open-telemetry/opentelemetry-collector>
 - <https://github.com/open-telemetry/opentelemetry-collector-contrib>
 - + OTC builder
 - <https://github.com/open-telemetry/opentelemetry-collector-builder>
 - + vendor distros:
 - <https://github.com/aws-observability/aws-otel-collector>
 - <https://github.com/SumoLogic/sumologic-otel-collector>
- ...
- Helm chart
- Kubernetes Operator
- Lambda Extension

Community

- GitHub
- Mailing lists
- ~~Gitter~~ CNCF Slack
- SIGs!
- Calendar
- Community page
<https://github.com/open-telemetry/community>

Name	Meeting Time	Meeting Notes	Meeting Link	Discussions
Maintainers weekly meeting	Every Monday at 09:00PT	Google Doc	Zoom	Slack
Collector	Every Wednesday at 09:00 PT	Google Doc	Zoom	Slack
C/C++: SDK	Every week alternating between Monday at 15:00 PT and Wednesday at 10:00 PT	Google Doc	Zoom	Slack
DotNET: Instrumentation	Every Wednesday at 10:30 PT	Google Doc	Zoom	Slack
DotNET: SDK	Every Tuesday alternating between 11:00 and 16:00 PT	Google Doc	Zoom	Slack
Erlang/Elixir: SDK	Every Thursday alternating between 07:00 and 15:00 PT	Google Doc	Zoom	Slack
GoLang: SDK	Every Thursday alternating between 10:00 and 15:00 PDT	Google Doc	Zoom	Slack
Instrumentation: Semantics	Every Monday at 11:30 PT	Google Doc	Zoom	Slack
Instrumentation: General	Every Tuesday at 16:00 PT	Google Doc	Zoom	Slack
Instrumentation: Messaging	Every Thursday at 8:00 PT	Google Doc	Zoom	Slack
				SDK and

Filters

Labels 37

Milestones 3

New pull request

28 Open ✓ 1,037 Closed

Author Label Projects Milestones Reviews Assignee Sort

Introduce semantic conventions for CloudEvents •

#1951 opened 3 minutes ago by joaopgrassi • Draft

1

Add resource detectors to compliance matrix ✓ editorial

#1950 opened 4 minutes ago by arminru • Review required

Add container.restart_count Resource attribute ✓

#1945 opened 8 hours ago by dmitryax • Changes requested

1

Refer and link to exception semantic conventions ×

#1944 opened 9 hours ago by GrahamLea • Review required

1

1

Revert "Add InstrumentationLibrary to Sampler.ShouldSample (#1850)" ✓ area:sampling area:sdk spec:trace

#1942 opened 22 hours ago by iNikem • Approved

5

Add missing fields to logs mapping from universal model to Splunk HEC ✓

#1941 opened yesterday by dmitryax • Review required

Update host field in mapping from HEC to the Unified model ✓

#1940 opened 2 days ago by dmitryax • Review required

Remove MeasurementProcessor from the Metrics SDK spec ✓ area:sdk spec:metrics

#1938 opened 4 days ago by reyang • Approved Metrics API/SD...

6

Add ExponentialHistogram to Metrics data model ✓ area:data-model spec:metrics

#1935 opened 5 days ago by jmacd • Review required

1

3

Add detail about explicit histogram buckets and boundaries • area:data-model spec:metrics

#1933 opened 5 days ago by jmacd • Approved

9

Upgrade to semantic convention generator 0.7.0 × area:semantic-conventions spec:trace

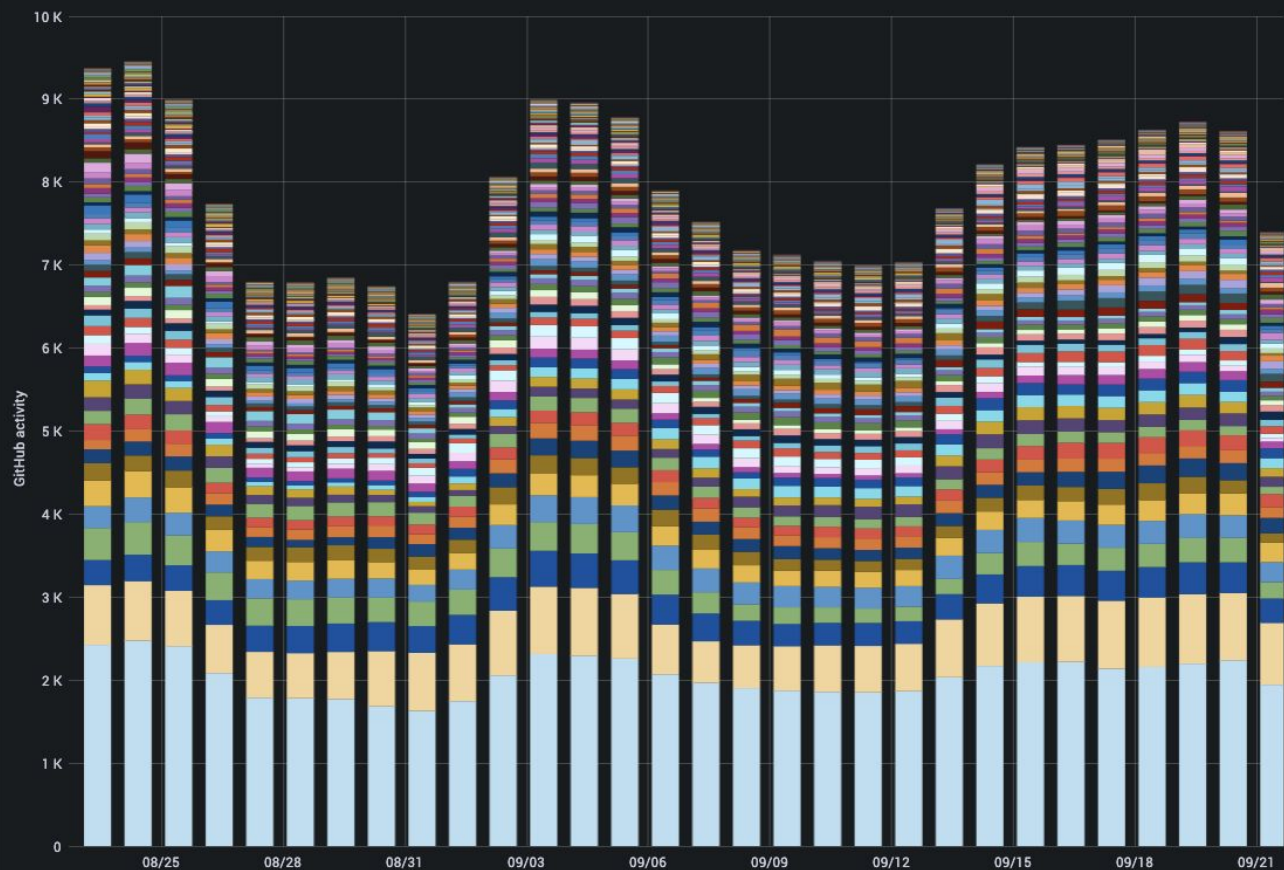
#1928 opened 6 days ago by Oberon00 • Review required

Release v1.7.0 ✓

#1926 opened 6 days ago by carlosalberto • Approved

6

All CNCF GitHub activity in repository groups (All, 7 Days MA)



	min	max	avg	current	total
Kubernetes	2 K	2 K	2 K	2 K	61 K
OpenTelemetry	499	838	684	745	21 K
Envoy	268	433	338	293	10 K
KubeVirt	173	393	284	197	9 K
gRPC	201	324	268	242	8 K
Argo	182	317	234	234	7 K
TiKV	114	216	174	114	5 K
OpenEBS	90	221	163	185	5 K
Prometheus	112	187	147	130	4 K
Backstage	111	192	145	157	4 K
Antrea	87	196	145	87	4 K
CNCF	67	171	130	122	4 K
NATS	52	200	120	103	4 K
containerd	48	145	111	121	3 K
Rook	49	144	99	118	3 K
Longhorn	49	154	98	83	3 K
KubeVela	43	145	95	43	3 K
KubeEdge	51	141	91	51	3 K
Linkerd	53	130	90	91	3 K
Flux	48	130	79	78	2 K
Keptn	49	100	75	100	2 K
Operator Framework	44	92	70	78	2 K
Vitess	42	99	69	57	2 K
Metal [®]	42	96	68	54	2 K
Helm	50	85	64	50	2 K
Kuma	37	144	64	47	2 K
OPA	36	99	64	75	2 K

How to start contributing

- There are frequently issues labeled with *"help wanted"* or *"good first issue"*
- E.g. here's the [list for opentelemetry-java](#)
- The process is typically to leave a note in the issue, asking for assignment. Keep your PR's small if possible
- Each repo also has a [CONTRIBUTING.md](#) doc, which describes the specific details. Please read it before contributing :)
- **Contributor License Agreement**

○ Create a separate module for JMH benchmarks	Feature Request	help wanted
#3083 opened on 31 Mar by plotr-sumo		
○ Annotations support for non-auto instrumented code	Feature Request	help wanted
#3035 opened on 17 Mar by chandru-kumar		
○ ImmutableBaggage.isKeyValid allows non-zero-length strings that contain nothing but spaces and/or tabs		
blocked:spec Bug good first issue help wanted		
#2948 opened on 26 Feb by jimshowalter		
○ Add jmx exporter	Feature Request	help wanted
#2522 opened on 14 Jan by maneeshbunwal123		
○ Investigate if we can/want to make fields and return values @NonNull by default.	Feature Request	help wanted
release:after-ga		
#2337 opened on 17 Dec 2020 by Oberon00		
○ Use a single named logger instance for all invalid API usages	API Feature Request	help wanted
#2093 opened on 18 Nov 2020 by jkwatson		
○ Prometheus Push-based MetricExporter	Feature Request	help wanted
#1687 opened on 23 Sep 2020 by pavloffay		
○ Add the ability to create more than one ZPageServer per VM	help wanted	release:after-ga
#1414 opened on 15 Jul 2020 by jkwatson		
○ Add OSGi support for API and SDK	help wanted	release:after-ga
#768 opened on 17 Jan 2020 by Sandared		

See original (youtube, in english)



https://youtu.be/DA_0KgpbnPc

