CLOUDFLARE

Operating Prometheus in a Serverless World

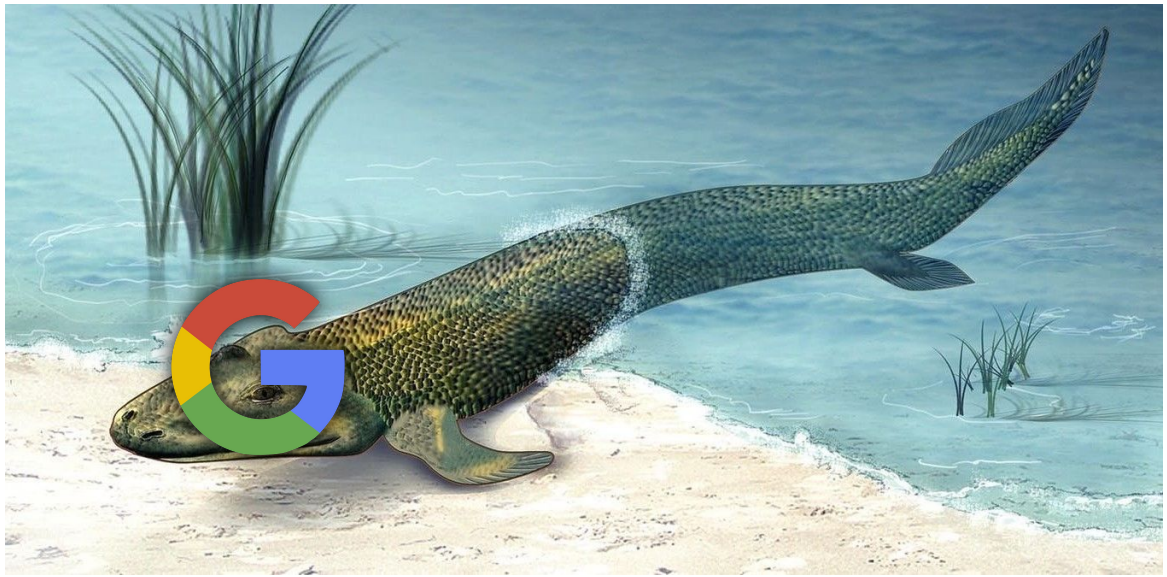
Colin Douch
Tech Lead @ Cloudflare Observability
@sinkingpoint basically everywhere

CLOUDFLARE

Who Am I?

CLOUDFLARE

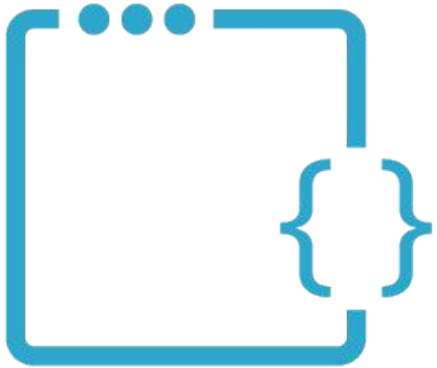
In the beginning, life was good



These SaaS providers decided to invent Serverless Computing one day, and now we have to support the Observability infrastructure

Just Kidding,
Serverless is Great

Serverless At Cloudflare



We made a thing: Cloudflare Workers

And then we started using it...

We had fumbled the ball a bit

Our Fatal Mistake: Closing our eyes for a second

CLOUDFLARE

Metrics!

The downside of Prometheus

Prometheus: Innovative, Great, *Opinionated*

CLOUDFLARE

Your system can be discovered and scraped

Prometheus has to reach out to your service

- So it needs to find it
- And your service needs to stick around long enough to be reachable

CLOUDFLARE

Your system can expose metrics over the network

Prometheus needs to be able to pull your metrics

- So your metrics need to be exposed over HTTP (generally...)
- And it needs to be reachable over the network

Your system can do its own aggregation

Credit: Wikipedia

What if your service only ever processes one request?

CLOUDFLARE

Summarizing a bit

~~HISTOGRAMS AND~~ SUMMARIES

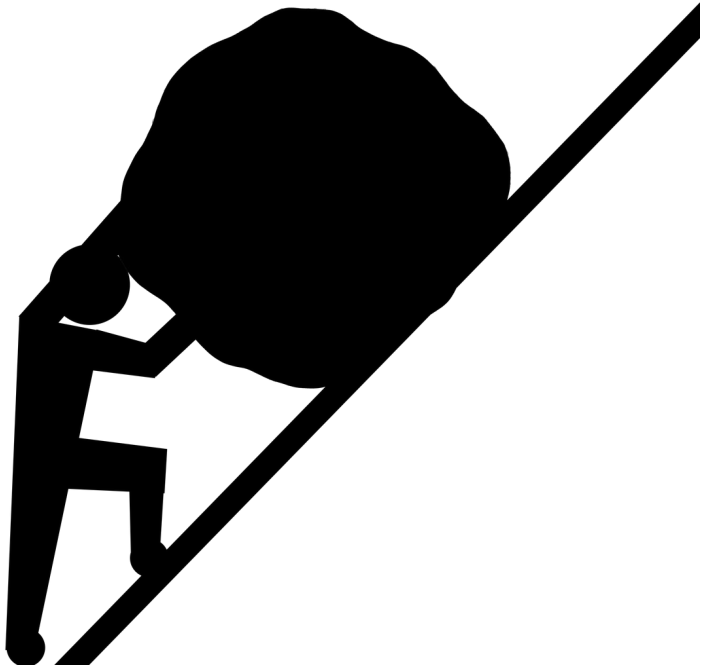
Get it? It’s a Prometheus Joke

CLOUDFLARE

What can we do?

Push Gateway

https://github.com/prometheus/pushgateway



Aggregation Gateway

https://github.com/weaveworks/prom-aggregation-gateway

We did a survey!

- We wanted to work out what sort of metrics our teams actually used
- Do teams actually *use* aggregation?
- What sort of semantics do they need?

What We Found: 😱

- Push Gateway Setups
- Aggregation Gateway Setups
- Internal Bespoke things
- Backends proxying metrics through best-effort UI pipelines
- Sentry.io + Custom Logic to pull metrics from events

What we found: Counters

Teams need counters (perhaps obviously)

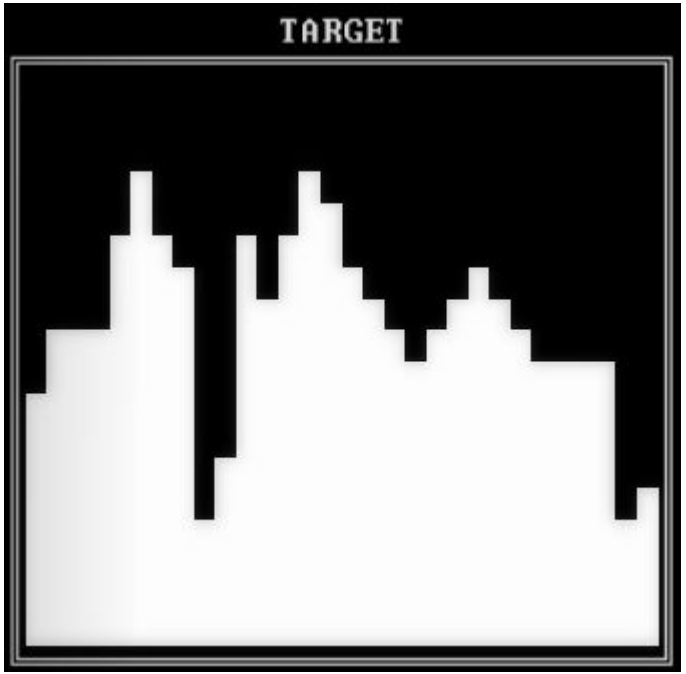
- Numbers that sum together
- E.g
  - Request counts
  - Error counts

What we found: Gauges

Teams need gauges

- Numbers that replace the previous value (*or* aggregate in some sort of mean/median type value)
- E.g.
  - Memory usage
  - CPU usage

What we found: Histograms / Summaries



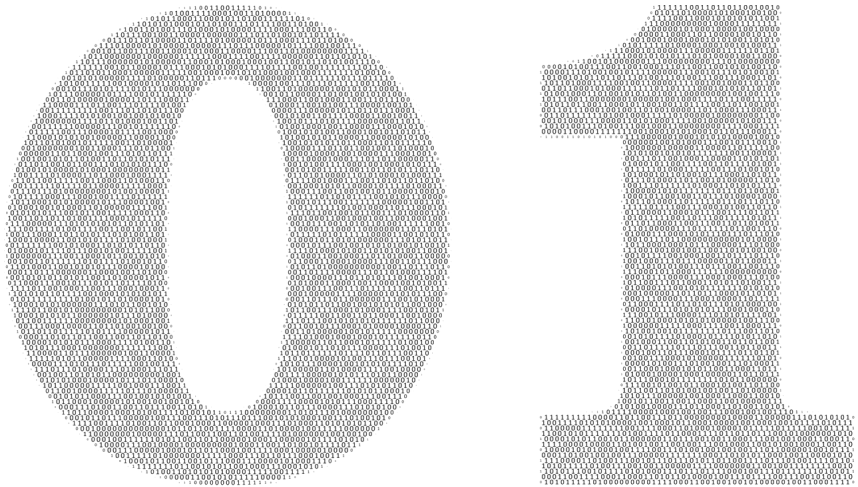
Everyone’s favourite metric type

CLOUDFLARE

What we found: Info metrics

Teams need Info Metrics

- Numbers that replace the entire *metric family* when a new labelset comes in
- E.g
  - Version metrics
  - Enums



CLOUDFLARE

So where does that leave us?

CLOUDFLARE

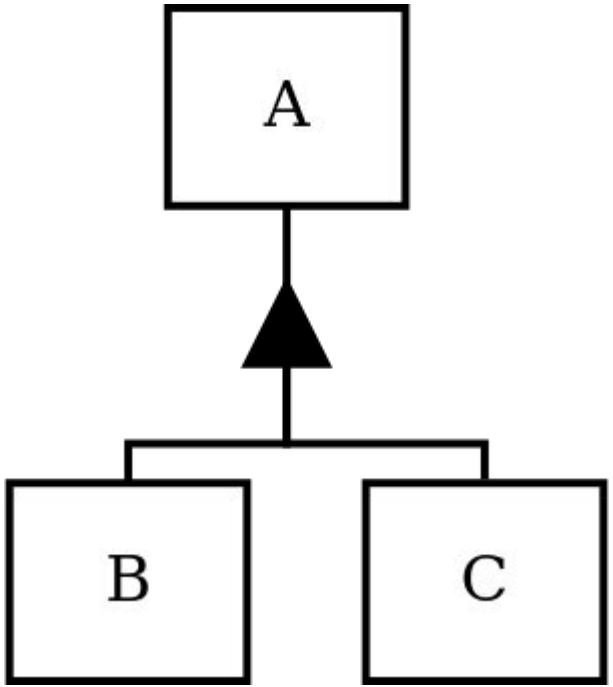
Push Gateway

Well Push Gateway doesn’t work...

Usually, the only valid use case for the Pushgateway is for capturing the outcome of a service-level batch job.

Aggregation Gateway

Aggregation Gateway gets us closer...



CLOUDFLARE

Something Event Based?

We could leave behind Text Exposition *entirely*

Events2prom, Cleodora

CLOUDFLARE

What we came up with

The Gravel Gateway

Credit: Wikipedia

https://github.com/sinkingpoint/gravel-gateway

CLOUDFLARE

Some functional requirements

- Needs to be able to accept pushes, like the Push Gateway
- Needs to be able to aggregate like the Aggregation Gateway
- Needs to be able to aggregate in a number of different ways
  - Sums
  - Medians/Means/Percentiles
  - Info/Boolean Semantics

CLOUDFLARE

But we have a problem

“How do we express all the semantics we need, in the limited amount of space we have”?

```
# TYPE build_info gauge
build_info{goversion="go1.17",time="2022-02-24-0656 UTC",version="2022.2.1"} 1
# TYPE go_threads gauge
go_threads 13
```

CLOUDFLARE

That’s easy: We Cheat

Let’s smuggle semantic information in the labels!

CLOUDFLARE

Introducing the “Clear Mode Label”

```
wshim_build_info{go_version="1.18",clear_mode="info"}
```

CLOUDFLARE

Summing

```
wshim_request_count{clear_mode="sum"} 1
wshim_request_count{clear_mode="sum"} 1
```

```
wshim_request_count 2
```

CLOUDFLARE

Info Metrics

```
wshim_info{version="1.0",go_version="1.17,clear_mode="family"} 1
wshim_info{version="1.1",go_version="1.18",clear_mode="family"} 1
```

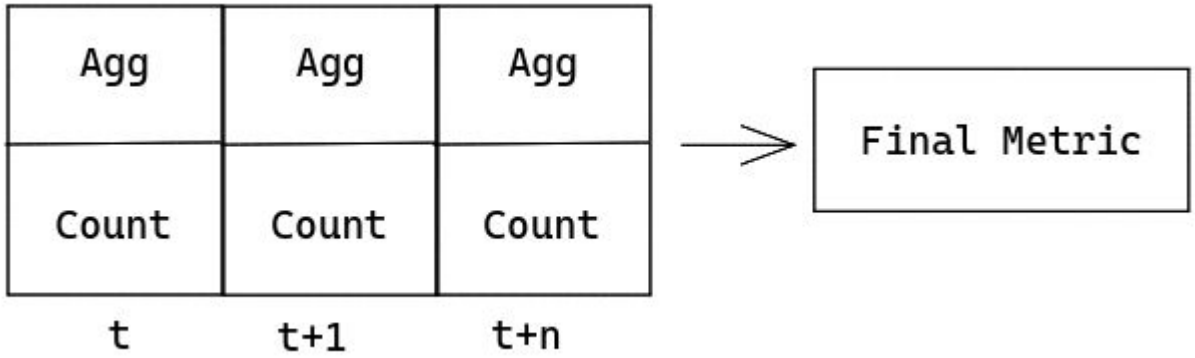
```
wshim_info{version="1.1",go_version="1.18"} 1
```

CLOUDFLARE

Means/Medians

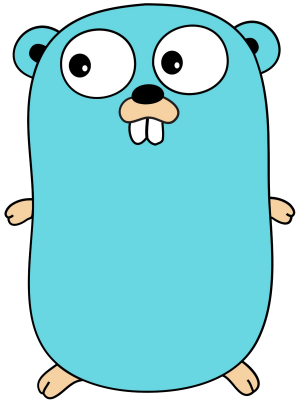
```
wshim_memory_usage_bytes{clear_mode="mean5m"} 0
wshim_memory_usage_bytes{clear_mode="mean5m"} 5
wshim_memory_usage_bytes{clear_mode="mean5m"} 10
```

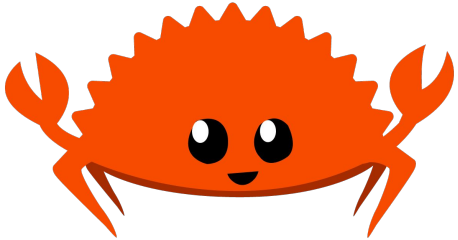
wshim_memory_usage_bytes 5



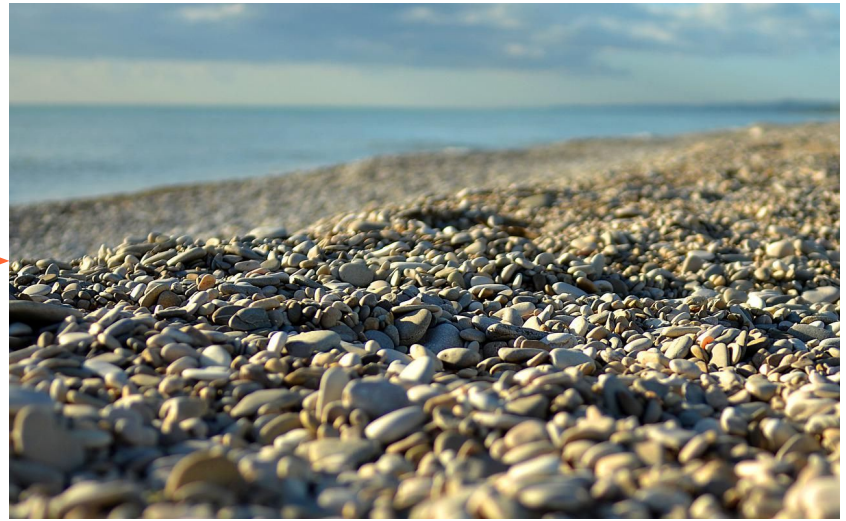
CLOUDFLARE

Compatibility with existing client libraries





Scaling it up



CLOUDFLARE

How this looks in practice

```
const { Pushgateway, register } = require('prom-client')
const pushGw = new Pushgateway(
 'https://gravel.sinkingpoint.com',
 register
);

const counter = new client.Counter({name: 'requests'});
counter.inc();

const info = new client.Gauge({name: 'info'});
info.set(1, {'clear_mode': 'family', 'version': '1.0'});

pushGw.push({ jobName: 'test' }, (err, resp, body) => {});
```

Looking Forward to the Future

Where Gravel Gateway goes from here

Credit: Wikipedia

Open Metrics

OPEN METRICS

CLOUDFLARE

Cleodora and the shift towards Open Telemetry

CLOUDFLARE

Thanks!

https://blog.sinkingpoint.com/posts/prometheus-for-faas/

https://github.com/sinkingpoint/gravel-gateway