



Cloud Native Telegraf

Cloud Native London
September 2019



David McKay

InfluxData

Developer Advocate

 Scottish

 Esoteric Programming Languages

 Kubernetes Release Team

 Former SRE

 Former Developer

The background is a dark purple gradient. On the left side, there is a faint, stylized globe with a grid of latitude and longitude lines. Overlaid on the globe are several concentric, semi-transparent circular arcs and a network of thin, light-colored lines connecting small dots, suggesting a global network or data flow. A white horizontal bar is positioned in the center of the image, containing the text.

Cloud Native Telegraf



Can I have one Telegraf,
please?

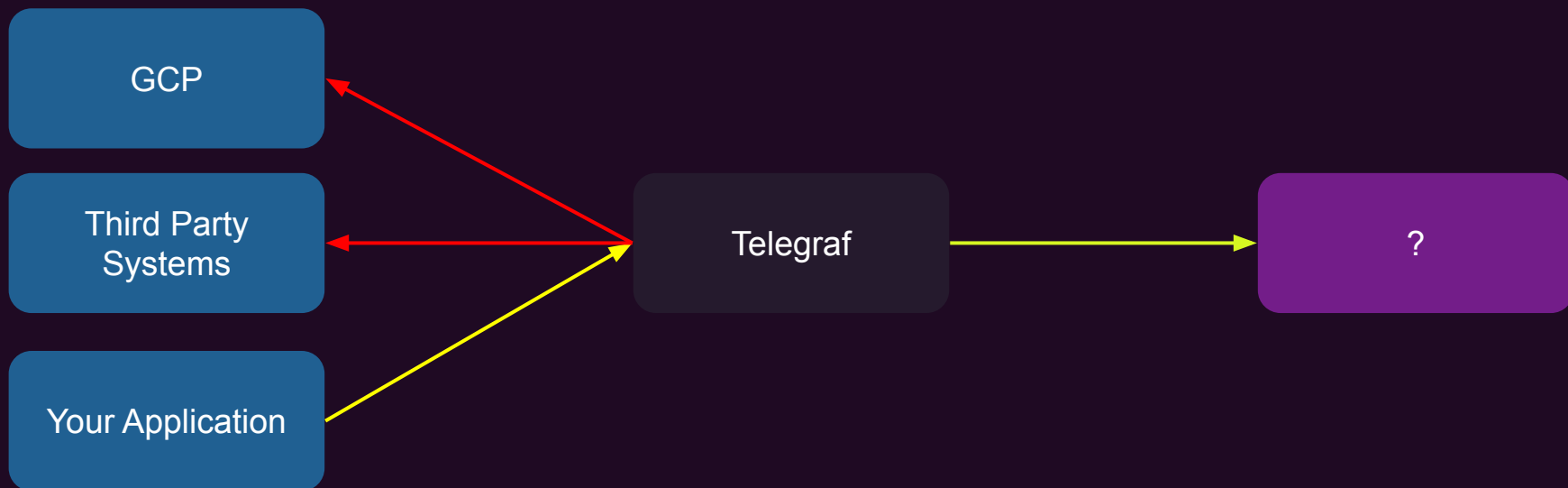
Telegraf

github.com/influxdata/telegraf



Telegraf is an agent for collecting, processing, aggregating, and writing metrics.

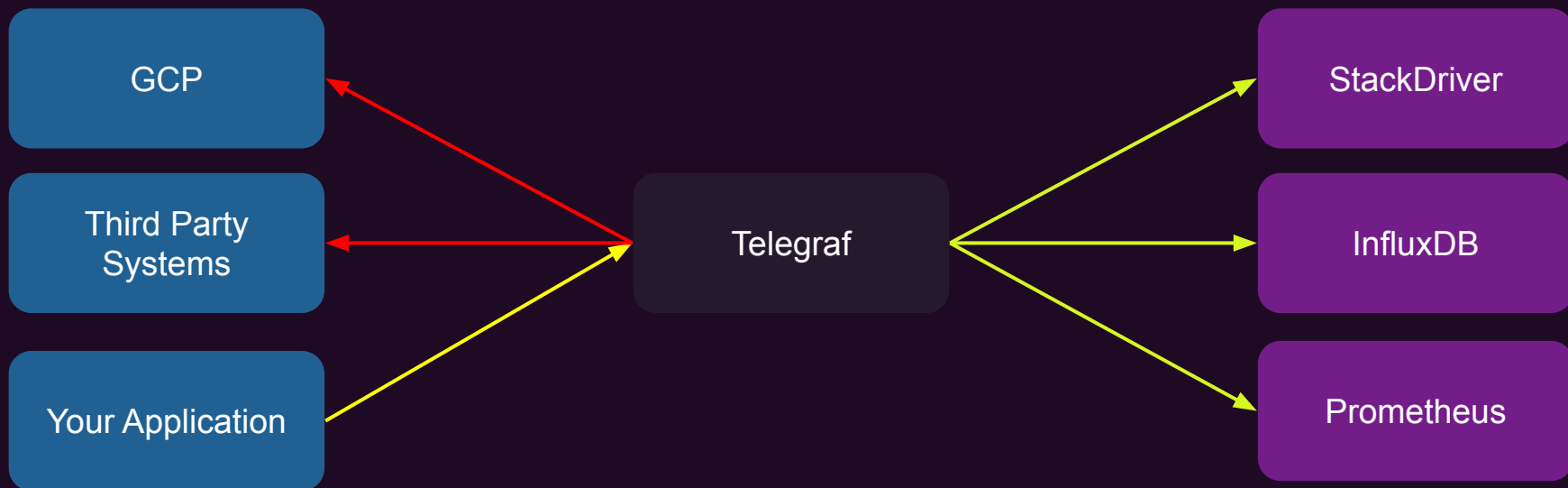
Architecture



An abstract graphic on the left side of the slide, consisting of numerous thin, white, wavy lines that flow from the top left towards the bottom right, creating a sense of movement and depth. The lines are set against a dark blue background.

Telegraf is Agnostic

Architecture



Plugins

Inputs

- ★ Docker
- ★ Kafka
- ★ Kubernetes
- ★ Nats
- ★ Postgres
- ★ System
 - CPU
 - Disk
 - Disk IO
 - Mem
 - Process

Outputs

- CrateDB
- CloudWatch
- DataDog
- Elasticsearch
- Graphite
- InfluxDB
- OpenTSDB
- Prometheus
- StackDriver
- Wavefront

Plugins

Inputs

Outputs

> 160

> 35

Input: activemq

The background features a dark purple gradient with a subtle, glowing globe on the left side. The globe is overlaid with a network of white lines and dots, suggesting a global or digital theme. A white horizontal bar is positioned across the middle of the slide, containing the main text.

Input: kubernetes

Kubernetes

- Should be run as a DaemonSet
- Hits the [stats/summary](#) endpoint of each kubelet
- Is responsible for gathering metrics for pods and their containers
- Will produce high cardinality data

Kubernetes

```
[[inputs.kubernetes]]  
  url = "https://localhost:10255"  
  bearer_token = "/run/secrets/token  
  insecure_skip_verify = true
```

Kubernetes

For Cloud Providers Managed
Kubernetes or minikube

```
[[ inputs.kubernetes ]]  
  url =  
  "https://kubernetes.default/api/v1/nodes/$NODE_NAME/proxy/  
  "
```

Kubernetes

Improvements

- 99.97% of the time, this plugin will run in-cluster
 - ◆ No reference, I made this number up
- So we don't need any configuration
 - ◆ We should trust you to manage RBAC
 - ◆ We'll use mounted ServiceAccount
 - ◆ We'll infer URL



Input: kube_inventory

Kube Inventory

- Should be run as a Deployment, with a single replica
- Hits the APIServer for resource information
- Will give you information on Deployments, DaemonSets, Volumes, etc, etc
- Will produce high cardinality data

Kube Inventory

```
[[inputs.kube_inventory]]  
  url = "https://kubernetes.default"  
  bearer_token = ""  
  resource_exclude = []  
  resource_include = []
```

Kube Inventory

Improvements

- 99.97% of the time, this plugin will run in-cluster
 - ◆ I heard this once before
- So we don't need any configuration
 - ◆ We should trust you to manage RBAC
 - ◆ We'll use mounted ServiceAccount
 - ◆ We'll infer URL

The background features a dark purple gradient with a faint, stylized globe on the left side. The globe is composed of a grid of lines and small dots, suggesting a network or data visualization. A white horizontal bar with a thin blue and purple gradient border is positioned across the middle of the slide.

Input: prometheus

Prometheus

- Run it however you want
 - ◆ Globally
 - ◆ Per Namespace
 - ◆ Depends on your workloads
- Will scrape Prometheus endpoints
- Will discover services through Prometheus annotations

Prometheus

```
[[inputs.prometheus]]  
    monitor_kubernetes_pods = true  
    # monitor_kubernetes_pods_namespace = ""  
  
    bearer_token = ""
```

Prometheus

Improvements

- 99.97% of the time, this plugin will run in-cluster
 - ◆ **Definite fact, I've heard this more than once**
- So we don't need any configuration
 - ◆ We should trust you to manage RBAC
 - ◆ We'll use mounted ServiceAccount

Prometheus

Improvements

→ Support ServiceMonitor CRD (Prometheus Operator)



Output: influxdb

InfluxDB

```
[[outputs.influxdb]]  
  urls = ["http://influxdb.monitoring:8086"]  
  
[[outputs.influxdb_v2]]  
  urls = ["http://influxdb.monitoring:9999"]  
  
  organization = "InfluxData"  
  bucket = "kubernetes"  
  token = "secret-token"
```



Output: prometheus_client

Prometheus Client

```
[[outputs.prometheus_client]]  
## Address to listen on.  
listen = ":9273"
```



Telegraf Super Powers



Proxying

Proxying

`influxdb_listener` is a service input plugin that listens for requests sent according to the InfluxDB HTTP API. The intent of the plugin is to allow Telegraf to serve as a proxy/router for the `/write` endpoint of the InfluxDB HTTP API.

Proxying

`http_listener_2` is a service input plugin that listens for metrics sent via **HTTP**. Metrics may be sent in **ANY** supported data format.

Proxying

There's also `socket_listener`, `tcp_listener`, and `udp_listener`



Batching

Batching

Telegraf will send metrics to outputs in batches of at most `metric_batch_size` metrics.

This controls the size of writes that Telegraf sends to output plugins.

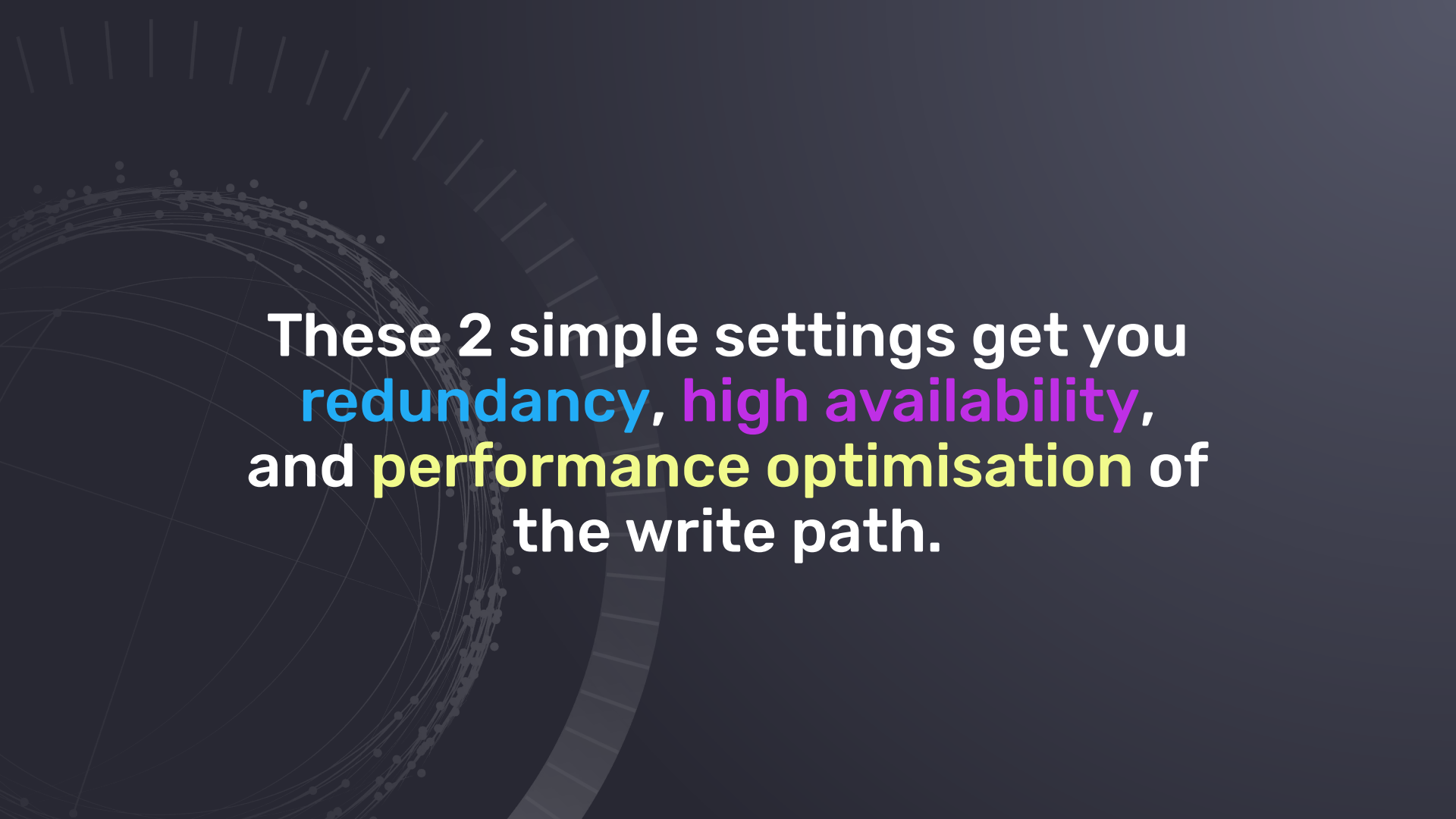


Buffering

Buffering

If a write to an output fails, Telegraf will hold `metric_buffer_limit` worth of metrics in-memory before data is lost.

This is **PER** output

The background features a dark blue gradient with a faint, stylized globe on the left side. The globe is composed of thin, light-colored lines representing latitude and longitude. Overlaid on the globe are several concentric, semi-transparent circular patterns made of small dots and thin lines, suggesting a network or data flow. The overall aesthetic is technical and modern.

These 2 simple settings get you
redundancy, high availability,
and **performance optimisation** of
the write path.

The left side of the slide features a decorative graphic consisting of numerous thin, white, wavy lines that flow from the top to the bottom, creating a sense of movement and depth. These lines are set against a dark purple background.

Telegraf as a Sidecar

Telegraf as a Sidecar

Hopefully from everything I've discussed, you can see how Telegraf could be a useful addition to any application as a sidecar.

1. It can consume logs
2. You can write events / traces from your code
3. It can act as a local metric buffer during DB downtime

Telegraf as a Sidecar

Unfortunately ...

The Telegraf **binary** is around **80MiB**

The Telegraf **image** is around **250MiB / 80MiB**



BYOT: Bring Your Own Telegraf

Bring Your Own Telegraf

```
FROM rawkode/telegraf:byo AS build
```

```
FROM alpine:3.7 AS telegraf
```

```
COPY --from=build /etc/telegraf /etc/telegraf
```

```
COPY --from=build
```

```
/go/src/github.com/influxdata/telegraf/telegraf  
/bin/telegraf
```



Telegraf Operator

Telegraf Operator

```
apiVersion: influxdata.com/v1
```

```
kind: Telegraf
```

```
metadata:
```

```
  name: mine
```

```
spec:
```

```
  version: "1.12"
```

```
  scrape_prometheus: false
```

```
  sidecar_injection: true
```

```
  metric_server: true
```



Demo Time



David McKay @rawkode · 21h
Twitter, I need your help. Which slide? 😂



12 replies · 2 likes



David McKay
@rawkode

I'll go with whatever slide has the most votes

[#TeamProfessional](#)

Vs

[#TeamFun](#)



39 votes · Final results

10:11 PM · Sep 3, 2019 from Paisley, Scotland · Twitter for Android



@rawkode



Thank You



influxdata[®]

Act in Time