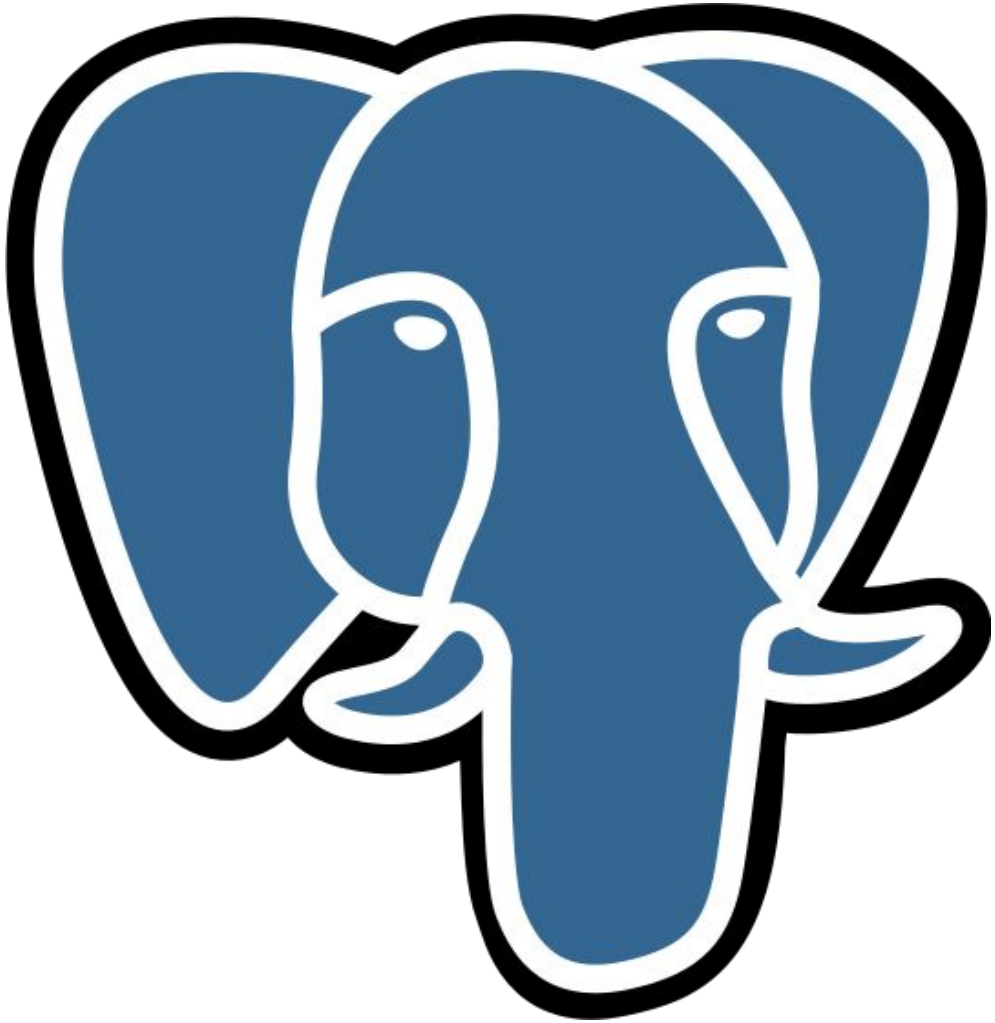
A New Approach to Logging:

Fluent Bit + PostgreSQL

Jonathan González V.



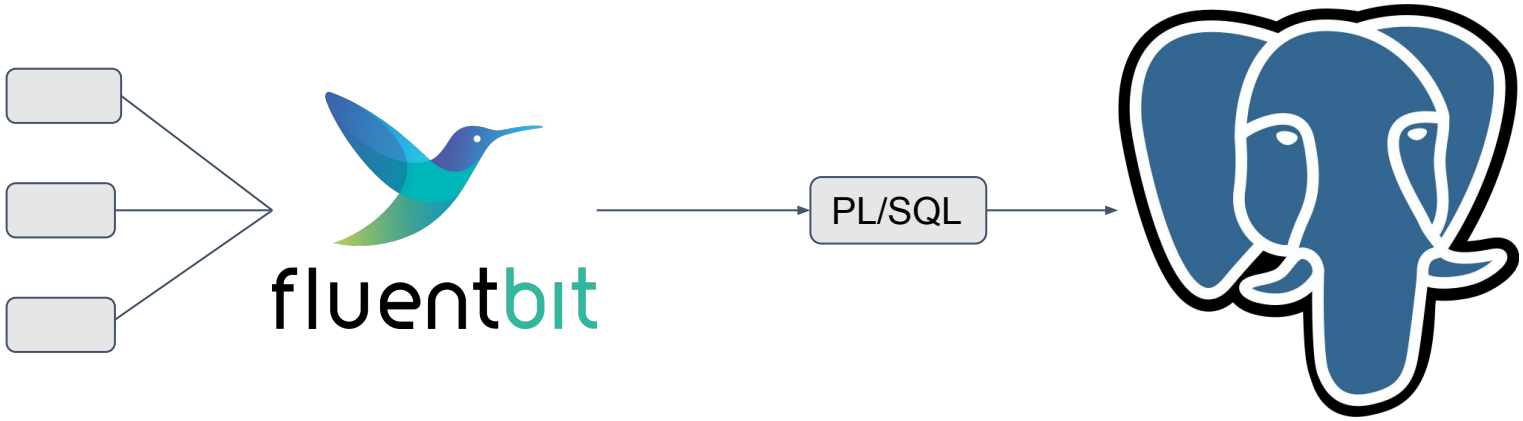
Presentation: my myself and I and all that

Motivation

- Play with PostgreSQL type JSONB
- Years of logs easy to find
- Generate usage reports, InfoSec reports
- Create stats with years of logs

Initial idea

And after some ideas



How to handle the data?

- Fluent Bit send the logs as a JSON Object
- Store raw data into one main table
- PL/SQL to process and split data to any table you want
- Use any field in the JSON object to decide

PostgreSQL: JSONB

- JavaScript Object Notation
- SQL Technical Report (sqltr-19075-6)
- JSONB supports indexing
- Easy to query:
  - select data->'date' from fluentbit;
- Easy to export and use in other apps

Configuration

- Main configuration options needed:

Host	Hostname/IP address of the PostgreSQL instance (default: 127.0.0.1)
Port	PostgreSQL port (default: 5432)
User	PostgreSQL username (default: current user)
Password	Password of PostgreSQL username
Database	Database name to connect to
Table	Table name where to store data

- Also support for CockroachDB using:
  - cockroachdb=true

Full list of options

- Full list of option in the Fluent Bit documentation

Configuration Parameters

Key	Description	Default
`Host`	Hostname/IP address of the PostgreSQL instance	- (127.0.0.1)
`Port`	PostgreSQL port	- (5432)
`User`	PostgreSQL username	- (current user)
`Password`	Password of PostgreSQL username	-
`Database`	Database name to connect to	- (current user)
`Table`	Table name where to store data	-
`Timestamp_Key`	Key in the JSON object containing the record timestamp	date
`Async`	Define if we will use async or sync connections	false
`min_pool_size`	Minimum number of connection in async mode	1
`max_pool_size`	Maximum amount of connections in async mode	4
`cockroachdb`	Set to `true` if you will connect the plugin with a CockroachDB	false

Query sample data

- Data from **cpu** input plugin

```
fluentbit=# select * from fluentbit where tag='cpu.0' limit 1;
-[ RECORD 1
]----------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------
tag  | cpu.0
time | 2020-10-12 10:40:49.505912
data | {"date": 1602510049.505912, "cpu_p": 4.0, "user_p": 2.5, "system_p": 1.5, "cpu0.p_cpu": 5.0,
"cpu1.p_cpu": 4.0, "cpu2.p_cpu": 4.0, "cpu3.p_cpu": 3.0, "cpu0.p_user": 4.0, "cpu1.p_user": 2.0,
"cpu2.p_user": 3.0, "cpu3.p_user": 2.0, "cpu0.p_system": 1.0, "cpu1.p_system": 2.0,
"cpu2.p_system": 1.0, "cpu3.p_system": 1.0}
```

Now with Apache logs

- Data from **tail** plugin using **apache2** parser

```
fluentbit=# select * from fluentbit where tag='apache' limit 1;
-[ RECORD 1
]----------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------
tag  | apache
time | 2020-10-02 00:35:51
data | {"code": "200", "date": 1601609751.0, "host": "\n::1", "path": "*", "size": "126", "user": "-",
"agent": "Apache/2.4.41 (Ubuntu) (internal dummy connection)", "method": "OPTIONS", "referer":
"-"}
```

Years of Apache Logs

- Analyze 3 years of Apache logs it’s a big task
- ~1Tb of data and just two weeks to load them
- Using **tail** plugin plus **PostgreSQL** output plugin
- PL/SQL to process and split data
- Billions of rows inside one table
- Using partition per month
- To query a month less than 1 second

Let’s deploy inside K8s

Just a simple command you can use to deploy and test Fluent Bit (open source in GitHub)

```
kubectl apply -k github.com/sxd/fluent-bit-kustomize/postgresql/
namespace/logging unchanged
serviceaccount/fluent-bit unchanged
clusterrole.rbac.authorization.k8s.io/fluent-bit-read unchanged
clusterrolebinding.rbac.authorization.k8s.io/fluent-bit-read unchanged
configmap/fluent-bit-config configured
daemonset.apps/fluent-bit unchanged
```

Video: Show and explain configuration plugin

Let’s use some PL/SQL

- We want all the data in a separate table
- Let’s partition that table
- Use conditions to fulfill empty fields

Video adding PL/SQL function

PL/SQL Function


```
CREATE OR REPLACE FUNCTION split_insert_trigger()
RETURNS TRIGGER AS $$
BEGIN

   IF NEW.data->'kubernetes' IS NULL THEN
     RETURN NEW;
   END IF;

   INSERT INTO k8s_log (ts, host, container, container_image,
     namespace, labels, annotations)
   SELECT
     NEW.time::TIMESTAMP,
     NEW.data->'kubernetes'->>'host',
     NEW.data->'kubernetes'->>'container_image',
     NEW.data->'kubernetes'->>'container_name',
     NEW.data->'kubernetes'->>'namespace_name',
     NEW.data->'kubernetes'->'labels',
     NEW.data->'kubernetes'->'annotations';

   RETURN NULL;
END;
$$
LANGUAGE plpgsql;
```

Query the data


```
fluentbit=# select distinct container from k8s_log;
-[ RECORD 1 ]---------------------------------
container |
-[ RECORD 2 ]---------------------------------
container | calico/node:v3.15.2
-[ RECORD 3 ]---------------------------------
container | kubernetesui/metrics-scraper:v1.0.5
-[ RECORD 4 ]---------------------------------
container | k8s.gcr.io/kube-apiserver:v1.17.6
-[ RECORD 5 ]---------------------------------
container | calico/kube-controllers:v3.15.2
```

PL/SQL Function

- Sometimes we may want to send some data to another table
- Let’s split the data depending on the tag
- If the input doesn’t match a tag send it to the default table

PL/SQL Function


```
CREATE OR REPLACE FUNCTION split_insert_trigger()
RETURNS TRIGGER AS $$
BEGIN

   IF NEW.tag = 'apache' THEN
   INSERT INTO apache_log (ts, host, path, code)
   SELECT
     NEW.time::TIMESTAMP,
     NEW.data->>'host',
     NEW.data->>'path',
     NEW.data->>'code';
   END IF;

   IF NEW.data->'kubernetes' IS NULL THEN
      RETURN NEW;
   END IF;

   INSERT INTO k8s_log (ts, host, container, container_image,
     namespace, labels, annotations)
   SELECT
     NEW.time::TIMESTAMP,
     NEW.data->'kubernetes'->>'host',
     NEW.data->'kubernetes'->>'container_image',
     NEW.data->'kubernetes'->>'container_name',
     NEW.data->'kubernetes'->>'namespace_name',
     NEW.data->'kubernetes'->'labels',
     NEW.data->'kubernetes'->'annotations';

   RETURN NULL;
END;
$$
LANGUAGE plpgsql;
```

Video show queries

Some ideas

- Graph the data using Grafana
- Split data per week not just months
- Create script to create automatics reports, per month or per year

Closing video going to last slide asking for questions

Questions?


KEEP CLOUD NATIVE
EVERYWHERE
KubeCon
CloudNativeCon
North America 2020
Virtual
HELM
KV