Analytics v2

Following the usage of your platform

Jean Pommier
PSC geOrchestra / pi-Geosolutions
jean.pommier@pi-geosolutions.fr

June 25th, 2025

Implementation

- Dec. 2024 – ? (Work still in progress)
- GIP #11
- Financed by geo2france
- Implemented by pi-Geosolutions
- Resources :
  - GIP: https://github.com/georchestra/improvement-proposals/issues/11
  - GH: https://github.com/georchestra/analytics
  - Doc: https://docs.georchestra.org/analytics/
  - Previous geOcom presentations:
    - geOcom 2024
    - geOcom 2023

Motivation

- Old analytics module is **not** (and won't be) **supported by the Gateway**: cf GIP #8
- Old analytics module only covers OGC logs, not the other apps : we need the capacity to analyse usage on **more apps than just OGC data**
- geOrchestra is modular, each platform is different : we need an easily **configurable**, **extensible** solution

Goals

First step (short-term goals) :

- OGC stats (like old analytics module)
- Vizualization dashboards
- Modular and extensible

Long-term goals :

- Collect usage data on most geOrchestra applications : OGC stats, number of page loads, downloads, bandwidth usage.
- Cover also single-page-applications (no server app).

What to collect ?

We start by collecting the access logs

- Sufficient to cover OGC stats + some other applications (e.g. data api).
- Also support importing historical access logs from the reverse proxy => seed the DB with previous years of data.
- Not invasive on the apps (next steps will require app-specific work, for instance on single page apps).
- Can contain the user-related information (user id, org, roles)

  - Gateway improvements, info provided through MDC (Mapped Diagnostic Context) data :
    - PR #191
    - PR #200
  - SP can be made compatible

Choosing the tools

Use existing tools when possible and relevant.

Several options, already partly discussed on previous geocoms/workshops :

- Matomo
- ElasticSearch + Kibana
- Loki + Grafana
- TimescaleDB + Superset
- and some combinations of the latters

Which tools will be accessible to most platform admins, which ones could add value to the platform, besides the analytics topic ?

Choosing the tools

Use existing tools when possible and relevant.

Several options, already partly discussed on previous geocoms/workshops :

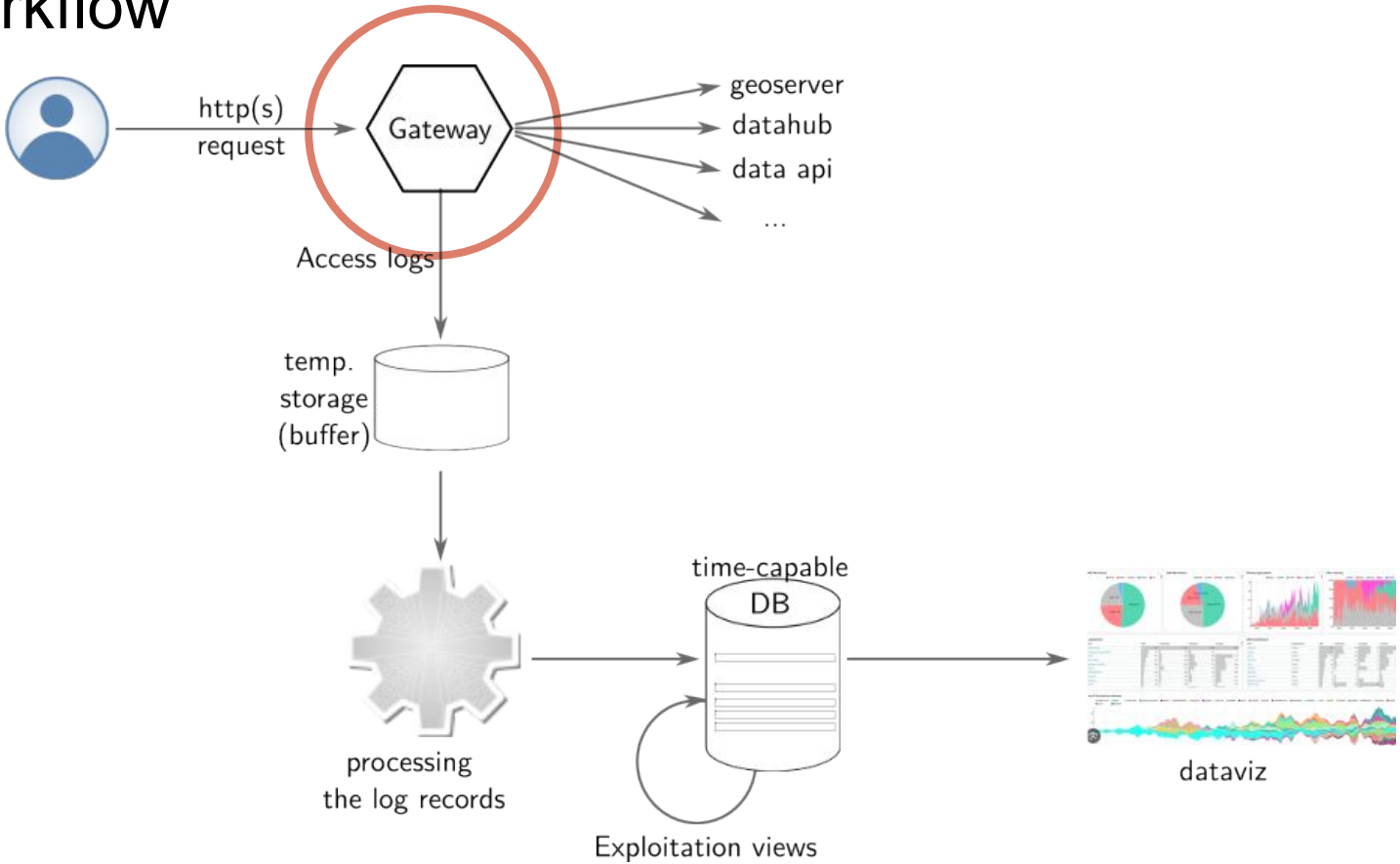
- Matomo
- ElasticSearch + Kibana
- Loki + Grafana
- TimescaleDB + Superset
- and some combinations of the latters

Which tools will be accessible to most platform admins, which ones could add value to the platform, besides the analytics topic ?

→ TimescaleDB is PostgreSQL, we know PostgreSQL, seems like a good option.

→ Superset is also relevant outside of the analytics context, for non-geo dataviz.

Workflow


http(s)
request
Gateway
geoserver
datahub
data api
...
Access logs
temp.
storage
(buffer)
time-capable
DB
processing
the log records
Exploitation views
dataviz

Gateway access logs

- Major upgrade since PR #191 (release 2.0.0)
  - Can expose access logs using **OpenTelemetry** json format (*separate from standard logging*)
  - Includes user/org/roles information ***if configured so***
  - Filters access logs log-level depending on regex config
- Read the doc !

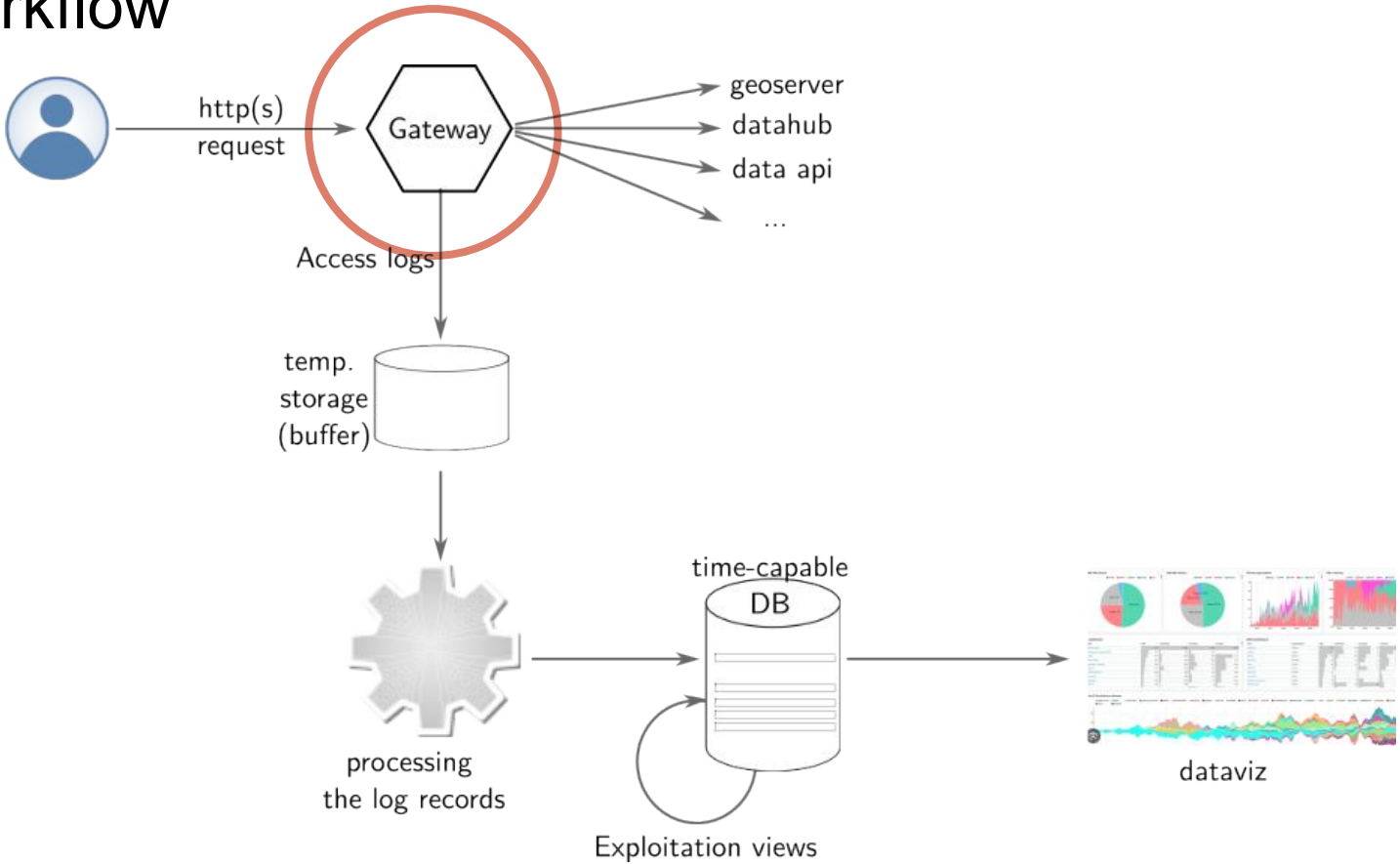
```yaml
# Logging profiles

# default profile:
logging:
  accesslog:
    enabled: true
    info:
      - .*/(?:ows|ogc|wms|wfs|wcs|wps)(?:/.*|\?.*)?$
      - .*/(#/metadata/)(?:/.*|\?.*)?$
    debug:
    - ".*/console/.*"
    trace:
    - ^(?!.*/web/wicket/resource/)(?!.*\.(png|jpg|jpeg|gif|svg|webp
  mdc:
    include:
      user:
        id: true
        roles: true
        org: true
        extras: true
        auth-method: true
      application:
        name: true
        version: true
        instance-id: true
        active-profiles: false
      http:
        id: true
        method: true
```

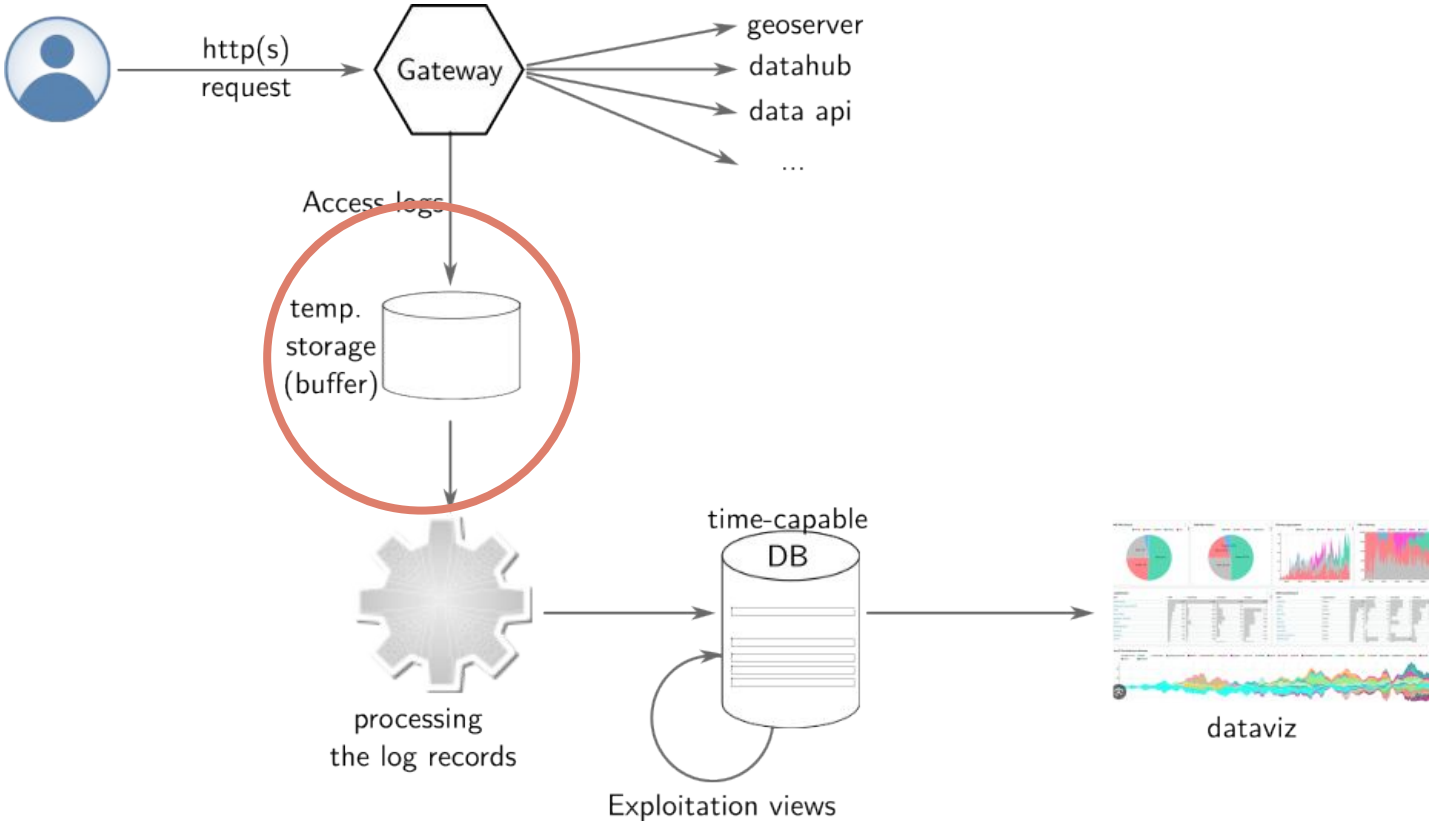
You said **OpenTelemetry** ?

- **OpenTelemetry** = standardized way of exposing metrics, traces and **logs**.
- Logs are provided as json objects. They can include Mapped Diagnostic Context (MDC) shipping several specific data. In our case, user info, roles etc.
- Gateway, based on Spring Framework, can easily expose its logs with OpenTelemetry. Cf PR from Gabriel Roldan + doc.
- Actually, Security Proxy can also quite easily do the same.
- You need an OpenTelemetry-capable collector to retrieve them. Then ship them somehow to the DB => Otel-collector, **vector**, telegraf.
- *Thank you Emilien & the C2C team for the joined brainstorming.*

Workflow


http(s)
request
Gateway
geoserver
datahub
data api
...
Access logs
temp.
storage
(buffer)
time-capable
DB
processing
the log records
Exploitation views
dataviz

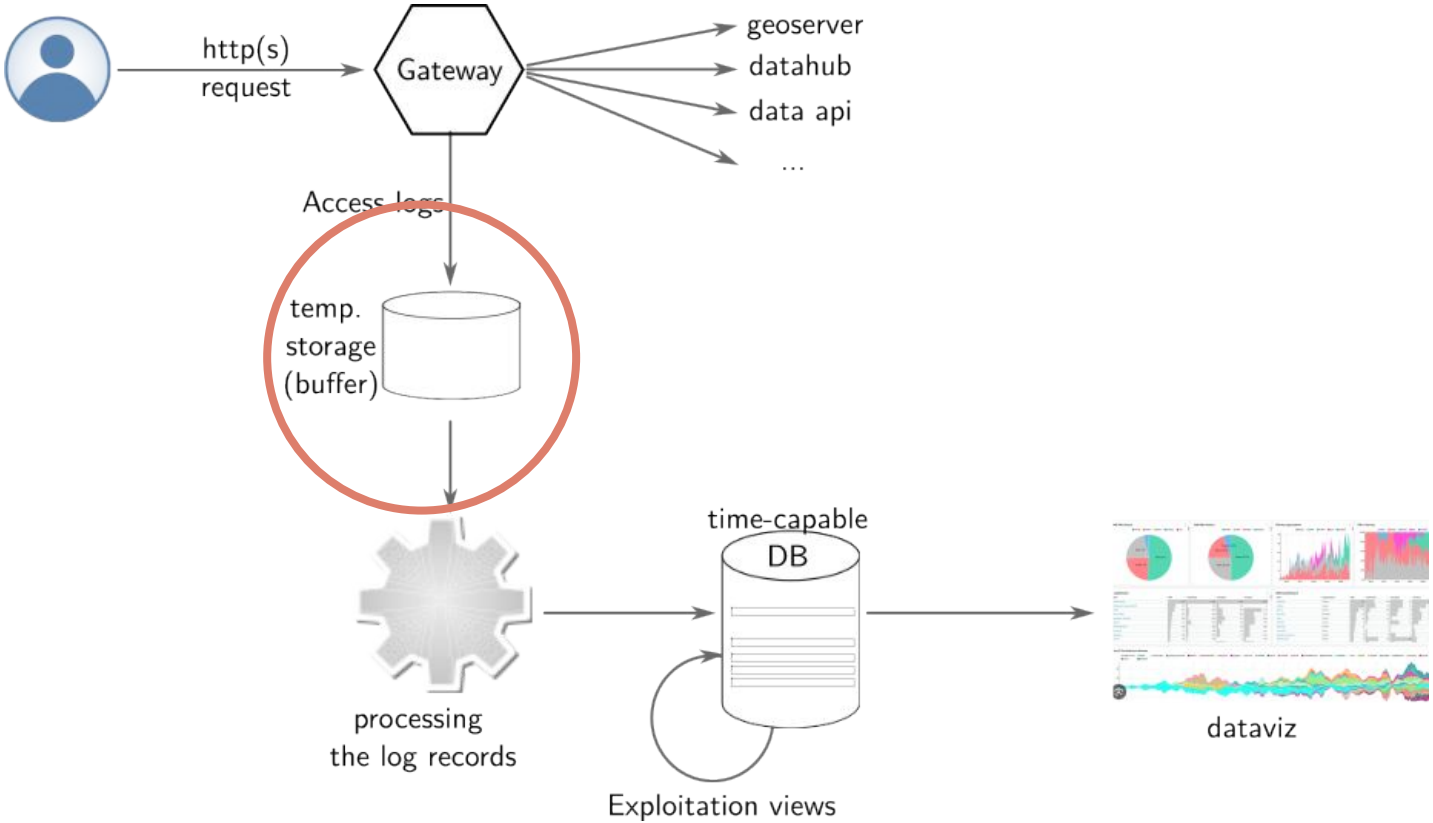
Workflow


http(s)
request
Gateway
geoserver
datahub
data api
...
Access logs
temp.
storage
(buffer)
time-capable
DB
processing
the log records
Exploitation views
dataviz

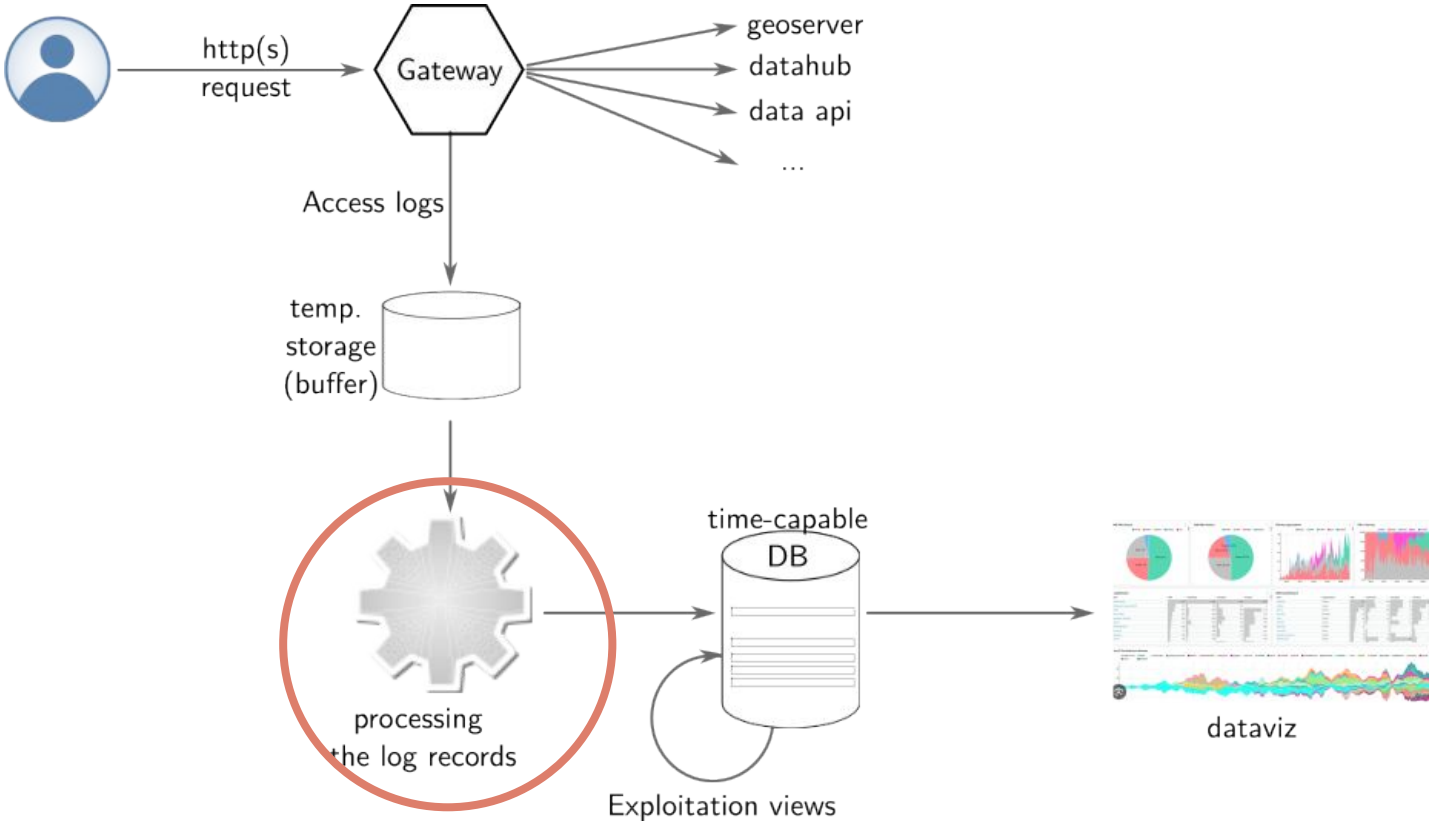
Vector

- An open source product from Datadog.
- **Can read OpenTelemetry data**.
- **Can write to a PostgreSQL DB => *we will write the records to a temp table in the DB.***
- Can process the records in-between if necessary => we don’t. We want also to support use-cases that don’t go through Vector.
- Quite easy to configure, yet powerful.
- https://vector.dev/

Workflow


http(s)
request
Gateway
geoserver
datahub
data api
...
Access logs
temp.
storage
(buffer)
time-capable
DB
processing
the log records
Exploitation views
dataviz

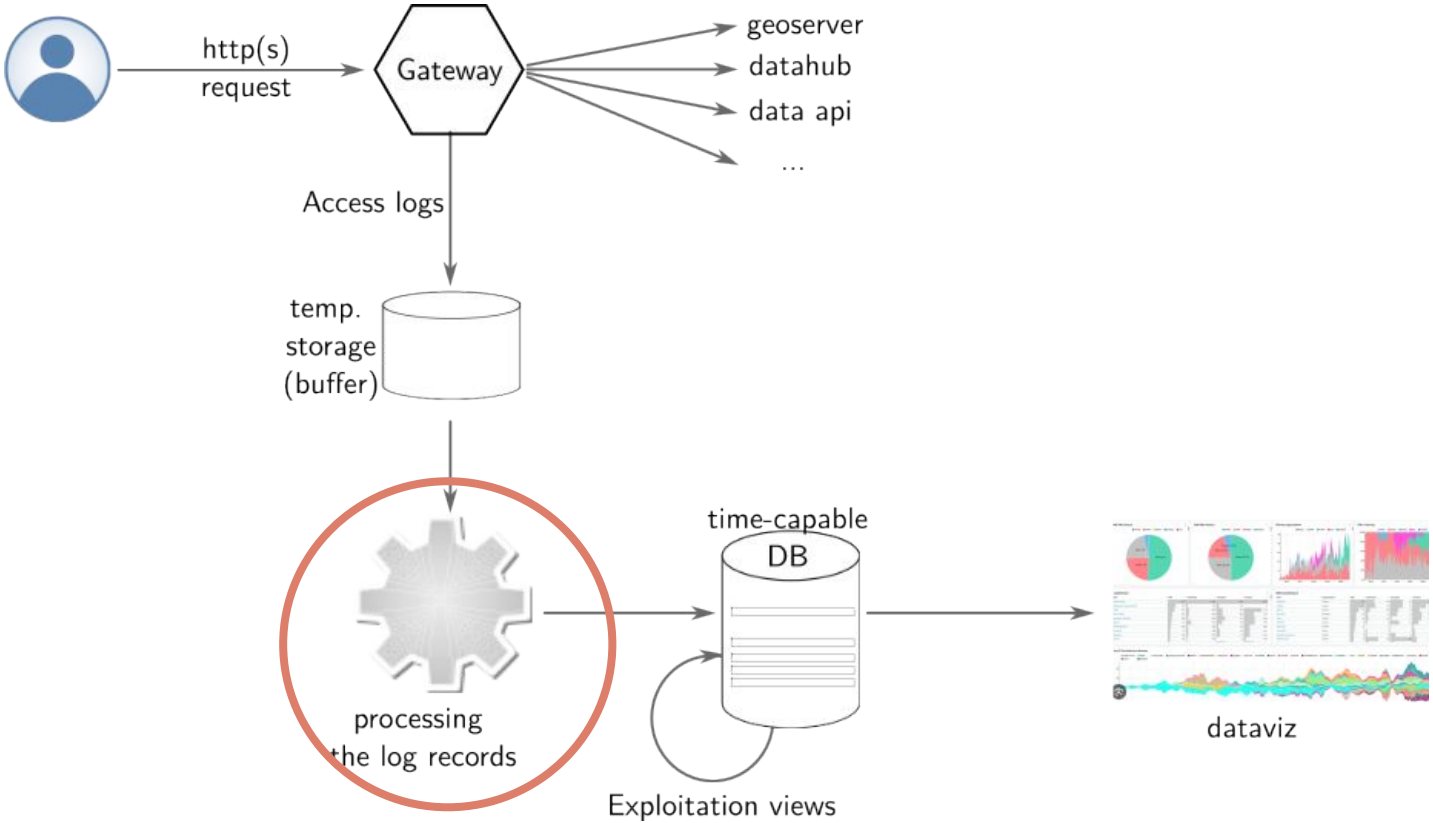
Workflow


http(s)
request
Gateway
geoserver
datahub
data api
...
Access logs
temp.
storage
(buffer)
processing
the log records
time-capable
DB
Exploitation views
dataviz

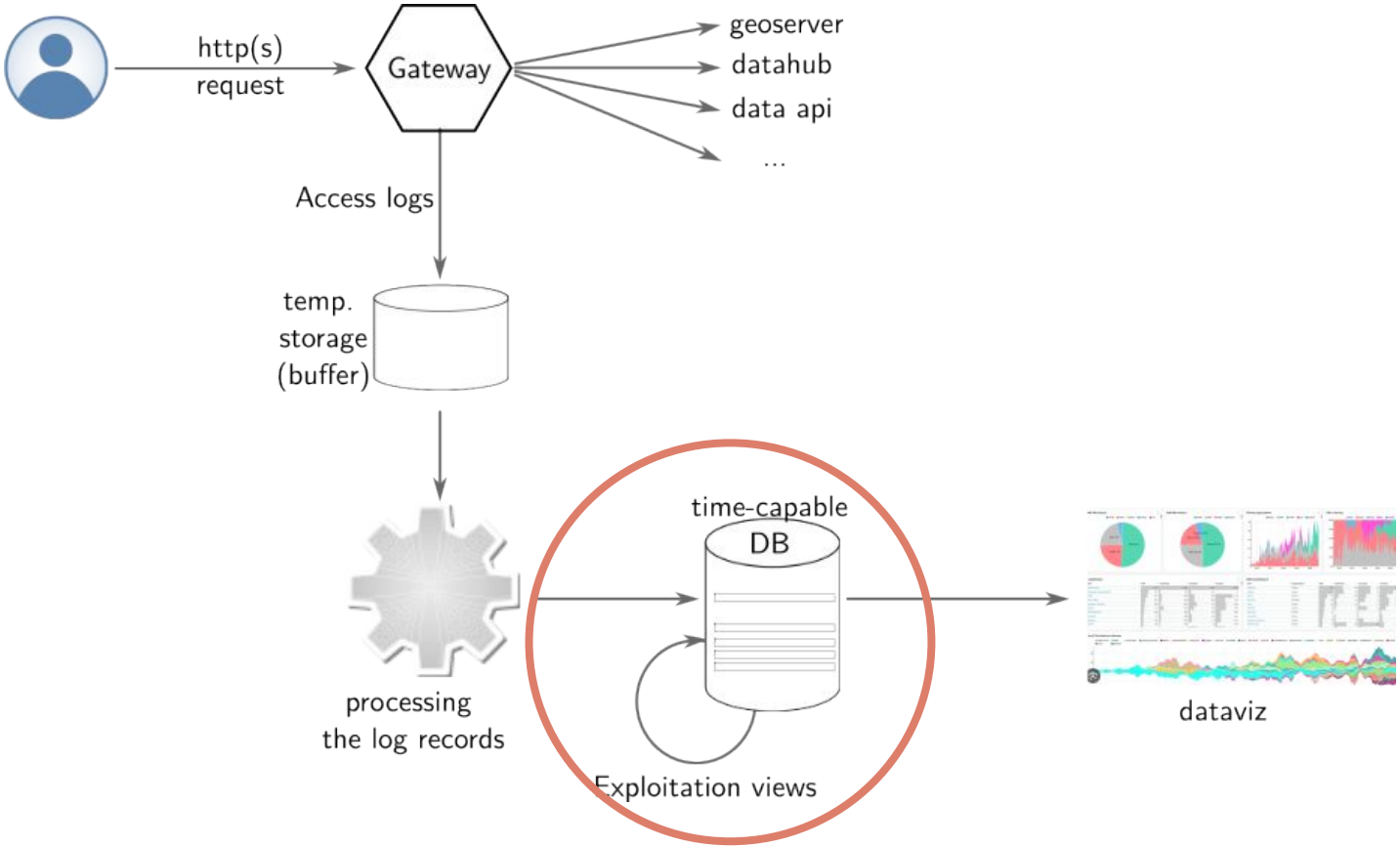
Processing the log records: Analytics CLI

- Extracts app-specific information from the access logs records buffer table.
- Writes processed output in a permanent, time-managed table (TimescaleDB hypertable).
- Custom Python code (technically affordable for most platform administrators).
- Supports several data sources:
  - temp. DB table containing OTEL records
  - text-based access logs files (historical records, reverse-proxy access logs)
- Handles app-specific logic in the records processing. Easily extendable to more apps.
- For now mostly supports geoserver.
- Can be run manually. Thought to be run as a cron-like task.

Workflow


http(s)
request
Gateway
geoserver
datahub
data api
...
Access logs
temp.
storage
(buffer)
processing
the log records
time-capable
DB
Exploitation views
dataviz

Workflow


http(s)
request
Gateway
geoserver
datahub
data api
...
Access logs
temp.
storage
(buffer)
time-capable
DB
processing
the log records
Exploitation views
dataviz

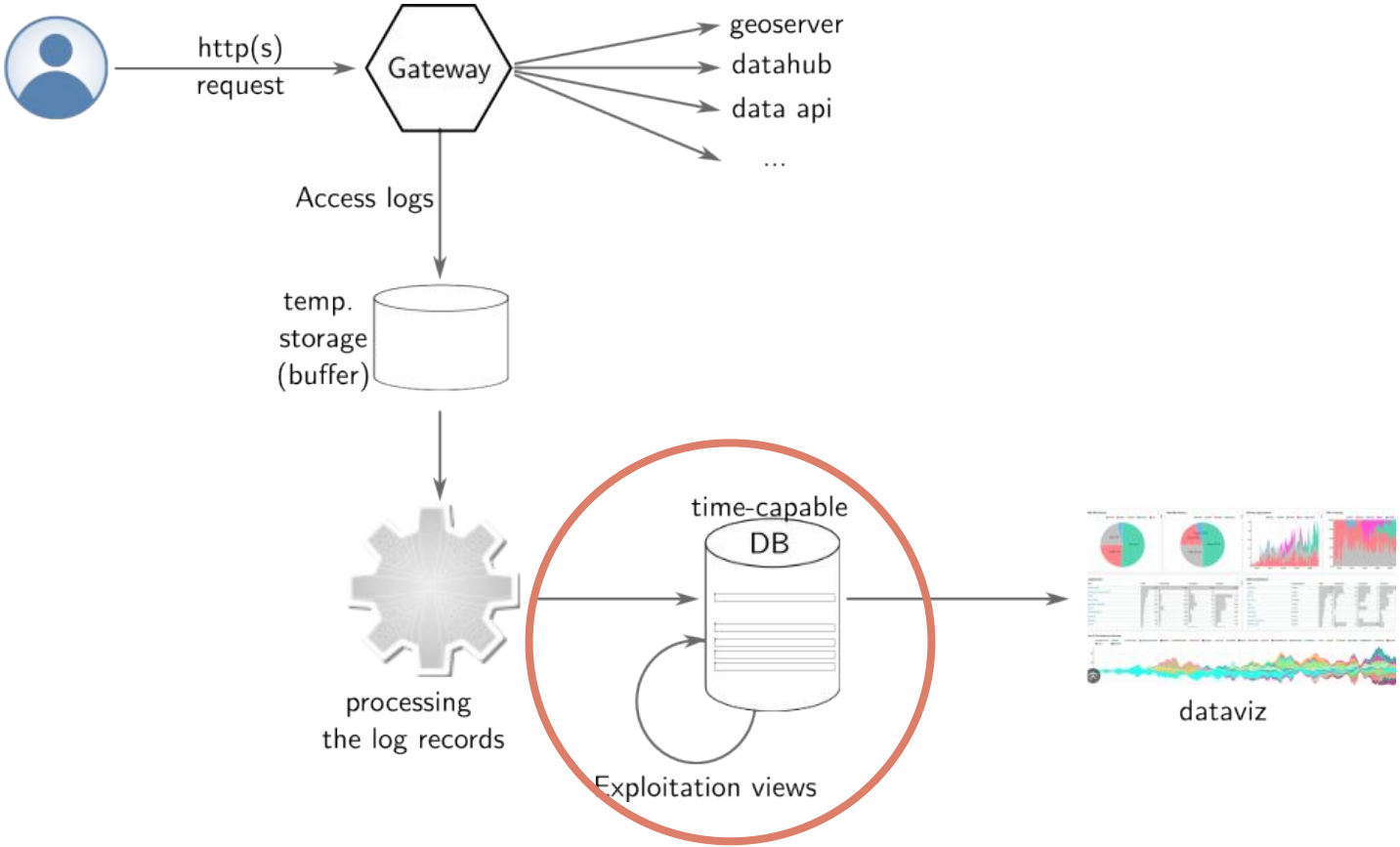
TimescaleDB: basics

- Handles time-dimensioned data (huge)
- Automatic partition on time dimension
- Compression
- Retention period
- Continuous Aggregates: Materialized views with
  - automatic incremental update
  - time granular aggregation (daily, weekly, monthly etc)
  - retention period

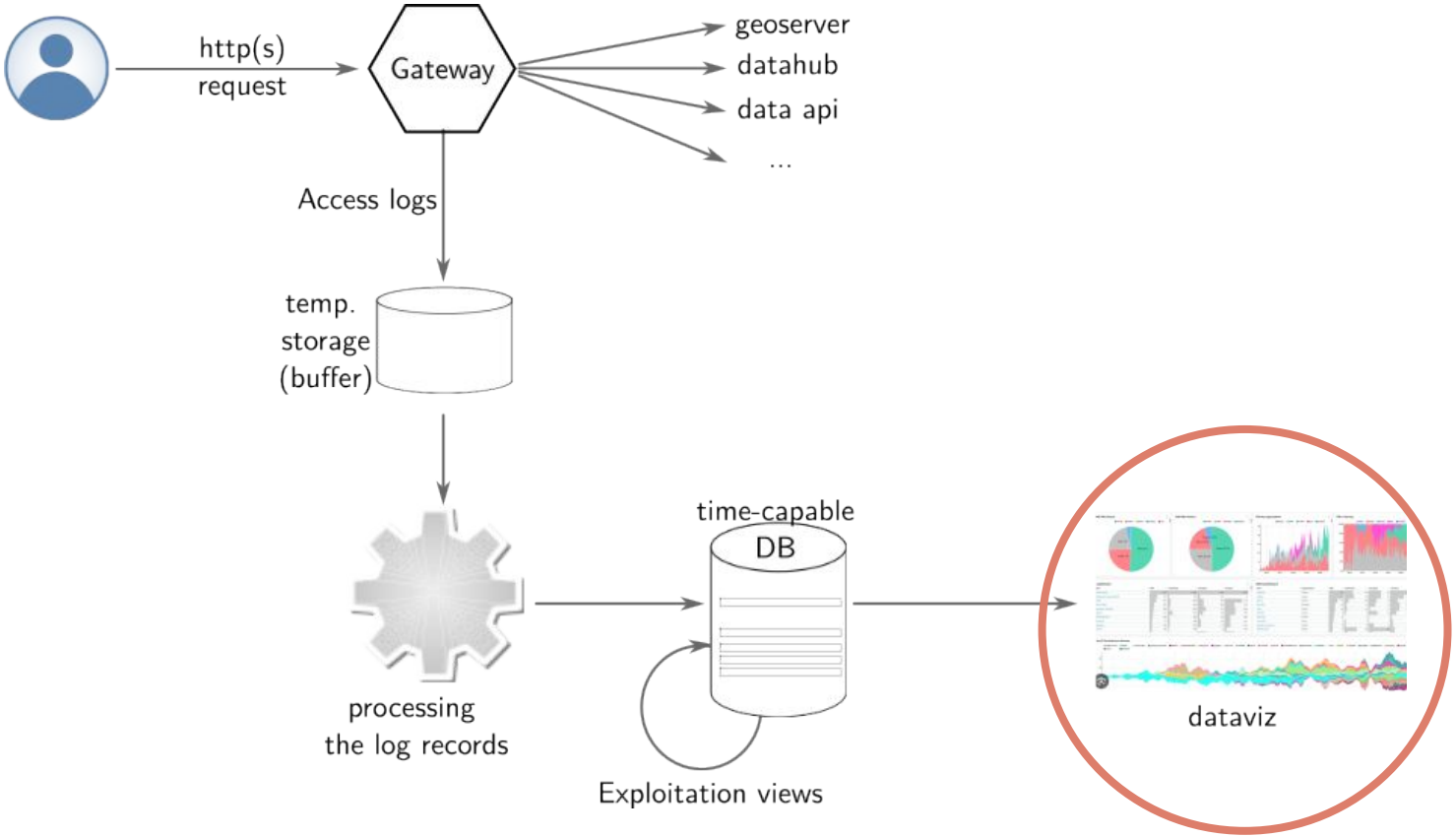
TimescaleDB for Analytics

- Defaults settings provided. Can be customized.
- Base table is access_logs.
- Continuous aggregates provide the exploitation views for dataviz (app-centric & global)
  - geoserver daily requests
  - geoserver monthly requests
  - bandwidth / user or org
  - etc

Workflow

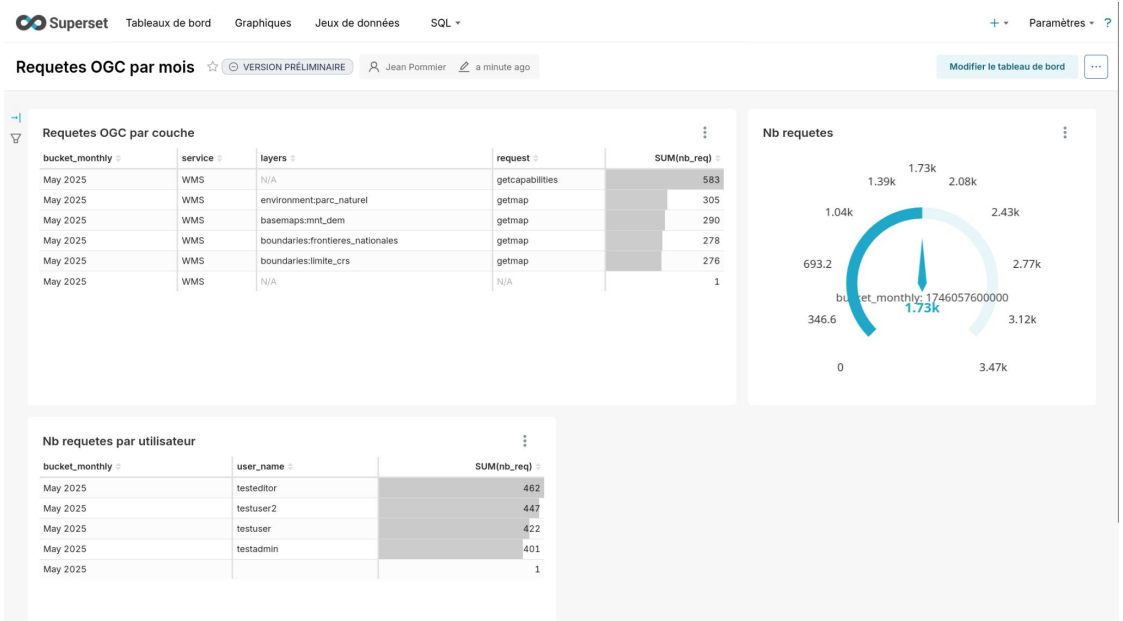

http(s)
request
Gateway
geoserver
datahub
data api
...
Access logs
temp.
storage
(buffer)
time-capable
DB
processing
the log records
Exploitation views
dataviz

Workflow


http(s)
request
Gateway
geoserver
datahub
data api
...
Access logs
temp.
storage
(buffer)
processing
the log records
time-capable
DB
Exploitation views
dataviz

Dataviz: Superset

- Already integrated in geOrchestra (see pres. from Tuesday 24th)
- Can read postgresql tables & views
- Graphs the exploitation views from the TimescaleDB access log data



And now ?

This is still a work in progress, almost just a Proof of Concept.

Remaining:

- Validate the current data structure
- Add support for more server apps (e.g. data API)
- Add support for single-page apps (mviewer, mapstore)
- Add more dataviz
- Consolidate the whole analytics module

A community work

- The result of a long-term analysis:
  - 3 geocoms (2023, 2024 & 2025)
  - community sprints
  - dedicated workshops
- A community effort. Special thanks to Mael Reboux, Stephane Ritzenthaler, Guillaume Ryckelynck, Emilien Devos, Gabriel Roldan, Pierre Mauduit & the whole C2C geOrchestra team.
- And of course, many thanks to geo2france for funding this.

Resources

- GIP: https://github.com/georchestra/improvement-proposals/issues/11
- GH: https://github.com/georchestra/analytics
- Doc: https://docs.georchestra.org/analytics/
- Previous geOcom presentations:
  - geOcom 2024
  - geOcom 2023

Thanks !

Any questions ? Shoot ! (the questions)