ClickHouse and Grafana: Aggregation-based Monitoring

Tuấn-Anh Nguyễn

Dec 28, 2022

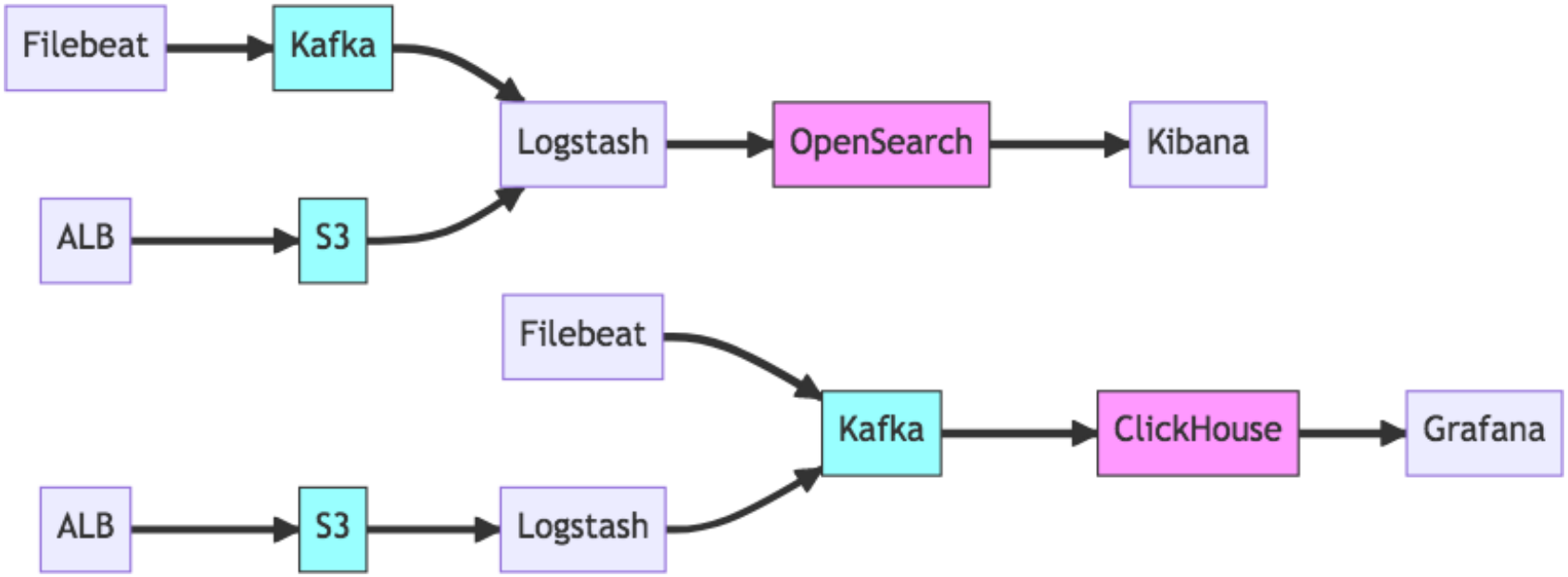
Grafana Basics

- Dashboards, charts, alerts
- Time range as a first-class concept
- Monitoring and troubleshooting

Metrics: Intrinsic vs. Aggregated Log (Events)

Continuous over time	Discrete in time
Meter	Counter
Metrics	Structured Log Events
Resources	Tasks, Transactions
Infrastructure	Application
Prometheus, InfluxDB	OpenSearch, ClickHouse
Sample, Align, Interpolate	Bucketize, Aggregate

Current Structured Log Setup



- On-site log agents: Filebeat (ELK)
- Log buffer: Kafka, S3 (ALB)
- Log collector: Logstash
- Database: OpenSearch, ClickHouse
- User interface: Kibana, Grafana

Kibana to Grafana

- Kibana is {Open,Elastic}Search-only.
- Grafana is more federated: datasources, plugins.
- Grafana is less focused: UX quality varies.

OpenSearch to ClickHouse

- Columnar storage: Better retention (10d $\rightarrow$ 3m)
- Vectorized execution: Improved query speed
- Streaming pre-agg: Even more speed
- SQL: Typically used as an analytics DB

ClickHouse: Access

- Through Grafana
- Through CLI: Requires Tailscale (VPN)

```
curl https://clickhouse.com/ | sh

clickhouse client -h chi-infra -u tailscale
```

ClickHouse: Bucketization and Aggregation

- Log fields: dimensions vs. metrics
- Bucketize: `to{Start,End}OfDuration()`, `GROUP BY`
- Aggregate: `sum`, `count`, `quantile`

```
FROM log_table
SELECT dim1, dim2, bucket_fn(time_column)
     , aggregate_fn()
WHERE interesting_time_range(time_column)
  AND other_dimensional_conditions
GROUP BY ALL
```

Log: ELK

- Originally intended for HTTP requests.
- Name didn't age well.
  - We are replacing the E and the K.
  - Replacing L is in the backlog.
- Services repurposed it for other stuff.
- Dimensions: `service`, `request`, `status`, `hostname`…
- Metrics: `request_time`, `body_bytes_sent`, `body_bytes_response`…
- Nulls are converted into sentinel values: 0, empty string, 1970-01-01.

Log: ALB

- HTTP requests from load balancers' PoV.
- Dimensions: `loadbalancer`, `target_group_arn`, `method`, `request`, `elb_status_code`...
- Metrics: `target_processing_time`, `sent_bytes`, `received_bytes`...
- Special note: status 460, or processing time 0, means server timed out...

Log: What Else Can We Add?

- Infra-level: nginx, DNS, k8s events, CI/CD
- FE: user interactions.
- FE: request metrics (wait time vs. read time).
- React routes: render duration, error rate, usage frequency.
- ElasticSearch writes: duration, error rate, frequency.
- Duration of DB queries.
- Time each record spends in each Flink operator, and e2e.

Log: Repurpose ELK Fields

- RPC over HTTP.
- Reuse RPC-related fields.
- Repurpose HTTP-specific fields.

Log: Add Extra ELK Fields

- Add service-specific fields to ELK log records.
- Create service-specific materialized views.
- Or just use `JSONExtract` if `WHERE` is not needed.

Log: Add New Event Types

- Write JSON to a new Kafka topic.
- Create a new table in ClickHouse to read it.
- No intermediate log collector e.g. Logstash.

Migration from Kibana

- Explore using small time intervals.
- Figure out additional columns to filter on.
- Make it work with the time selector.
- Query over longer time period to check whether it performs.
- If it would be looked at a lot, build a materialized view.

ClickHouse: Materialized Views

- Declarative.
- Sort-of misleading name. More like triggers:
  - Continuously updated.
  - Don't touch existing data.
- Partial aggregation and streaming.
  - `State` and `Merge` functions.

References

- ClickHouse plugin for Grafana
- ClickHouse functions: date time, JSON, aggregation
- ClickHouse output formats