Data Architecture & Analysis about OpenTelemetry Observability

TrendMicro – Mars Su

Table of Contents

01
Background & Problems

A world safe for exchanging digital information

$2 Billion 2022 Gross Sales†

100 Consecutive Profitable Quarters
Every quarter since going public

424,000+ SaaS Commercial Customers
500,000+ commercial customers, 175+ countries

62M+
SaaS-Protected Assets

#1 in **Cloud** Workload Security
Based on global market share*

A Leader in **XDR**
Based on offering strength and strategy**

A Leader in **EPP**
Based on completeness of vision and ability to execute***

Highest Market Share in **IDPS**
Based on sum of vendor revenue ($) for 2Q23****

#1 in Public Vulnerability Disclosure‡
+ Over **146 Billion threats blocked** in 2022

7500+ Employees in **73** Countries

*IDC Worldwide Cloud Workload Security 2022 Market Shares, #US49669822, May 2023
**The Forrester New Wave™: Extended Detection And Response (XDR) Providers, Q4 2021
‡Quantifying the Public Vulnerability Market, Omdia, May 2022

***Gartner Magic Quadrant for Endpoint Protection Platforms, 31 December 2022
****Gartner Market Share: Enterprise Network Equipment by Market Segment, Worldwide, 2Q23, 25 September 2023

†*Constant currency*

TREND MICRO™

Shift from Security Tools to a Cybersecurity Platform

Prevention

Endpoint EPP

Detection only on managed devices

EDR

Multilayered:
Endpoint, Network,
Cloud Workloads, and E-mail

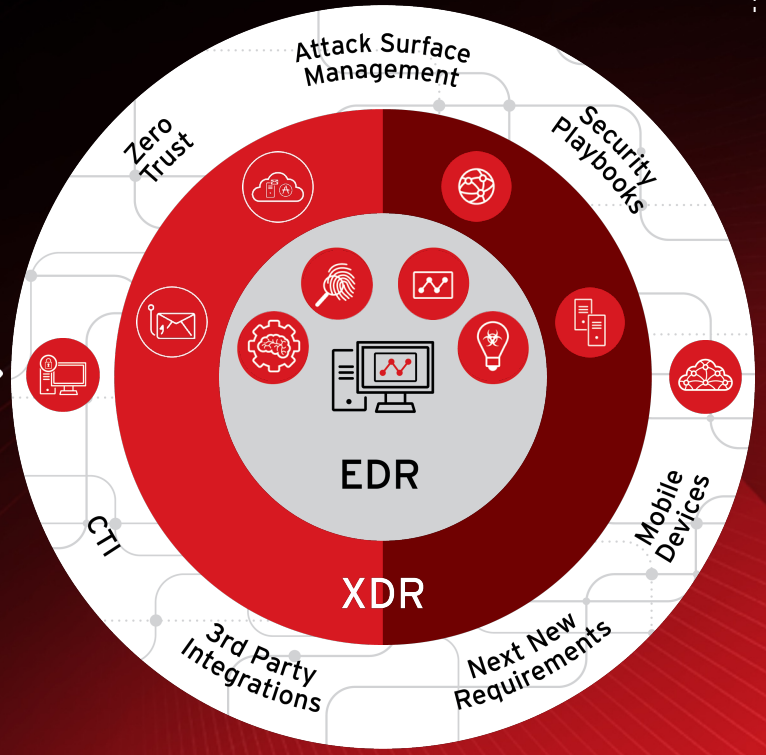
Cross Threat Detection, Correlation and Response

EDR

XDR

Trend Vision One™ - Advanced Cybersecurity Platform

EDR, XDR, Attack Surface Management, Zero Trust, Threat intel, Assessment



New Cyber Security Requirements & Challenges

TREND MICRO™ Vision One

Attack Surface Risk Management

Discover Attack Surface • Assess Risk • Mitigate Risk

Zero Trust Architecture

Managed Services

Ecosystem Integration

Extended Detection and Response (XDR)


User and Identity


Email


Endpoints and Servers


Cloud Infrastructure


Applications


Code Repository


Data


Network

5G


ICS/OT

Email Security | **Endpoint** Security | **Cloud** Security | **Network** Security | **Data** Security | **Identity** Security

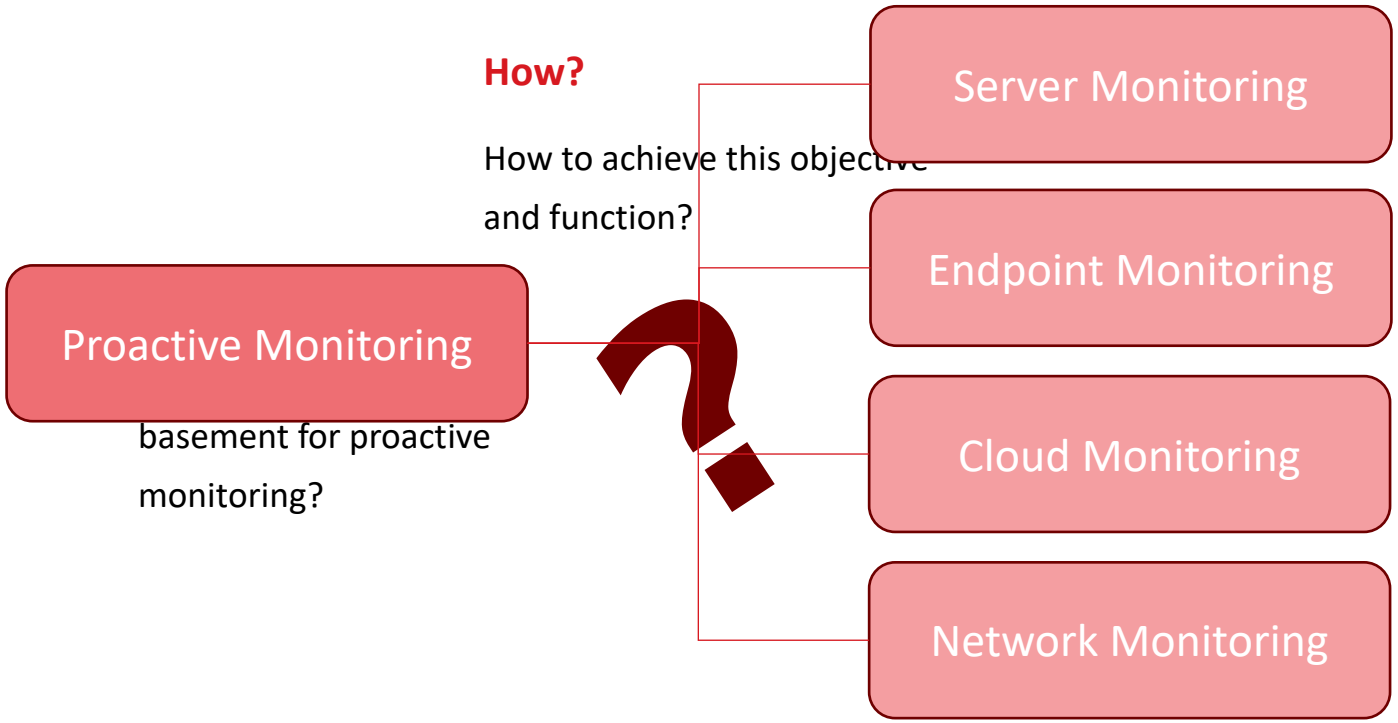
Risk Mitigation • IT Automation — **Orchestration and Automation** — Custom Playbooks • Case Management

Attack Surface Intelligence • Zero Day Initiative — **Global Threat Intelligence** — Threat Research • Big Data Analytics

AI Privacy and Ethics • AI Companion — **AI Native Foundation** — Generative AI • Custom LLM • Machine Learning

What is the Proactive Monitoring?

"The concept of Proactive Monitoring is quickly to position potential statuses or issues and notify us in advance <u>before</u> customers find problems or challenges for our business."



02
OpenTelemetry Concept & Data Architecture

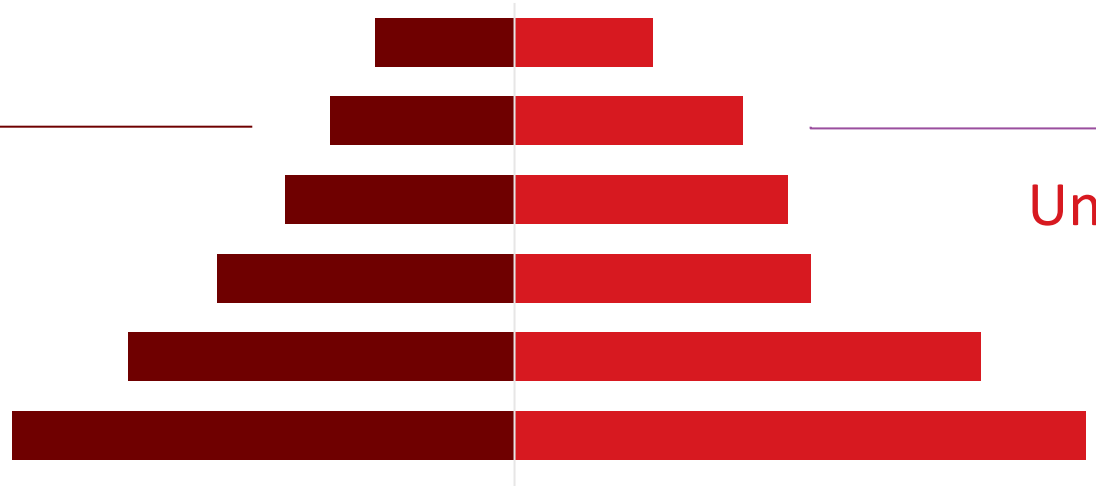
What is the Observability?

Information

Is the information we currently have sufficient? Can it help us to understand the system or service status?

Clear Judgement

Are we able to effectively utilize these information to gain a clear understanding of the status? Is this information truly demonstrating its value?

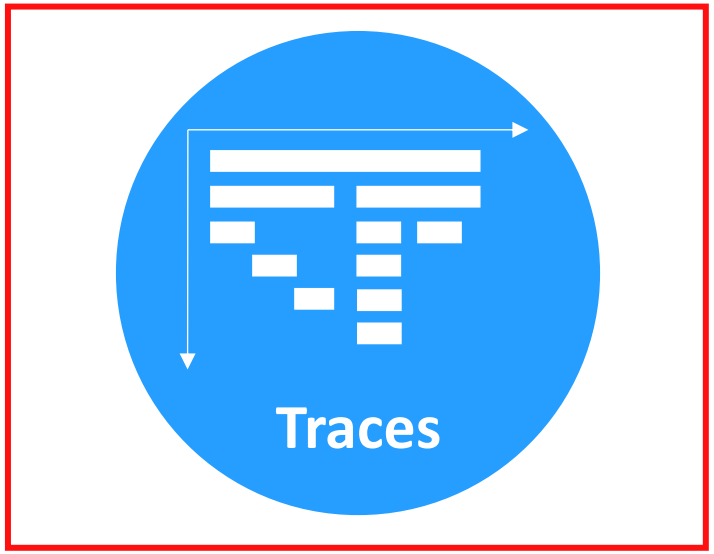


Observability Signal

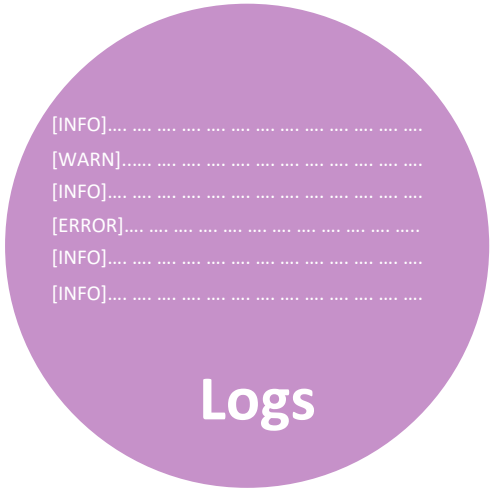
Unified & Correlation

Observability Signal – Metrics, Traces & Logs

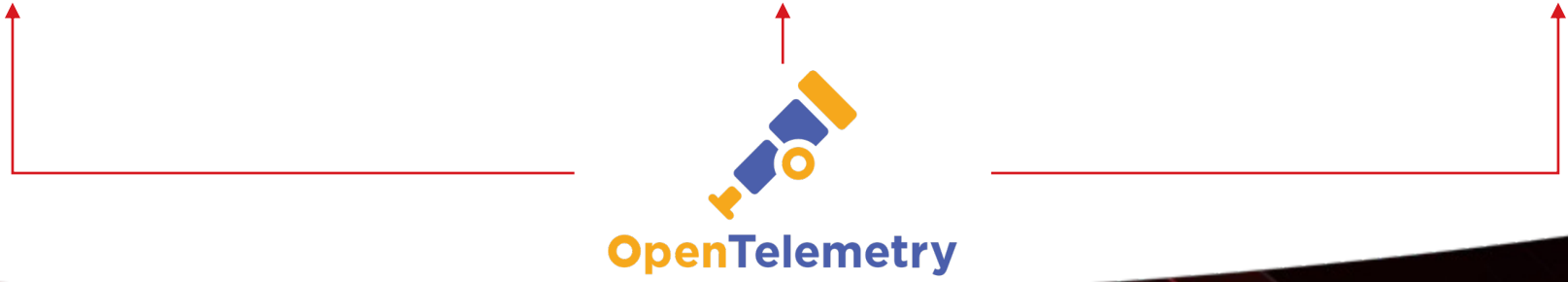
Normally this is time series data that is used in trends for memory usage and latency.



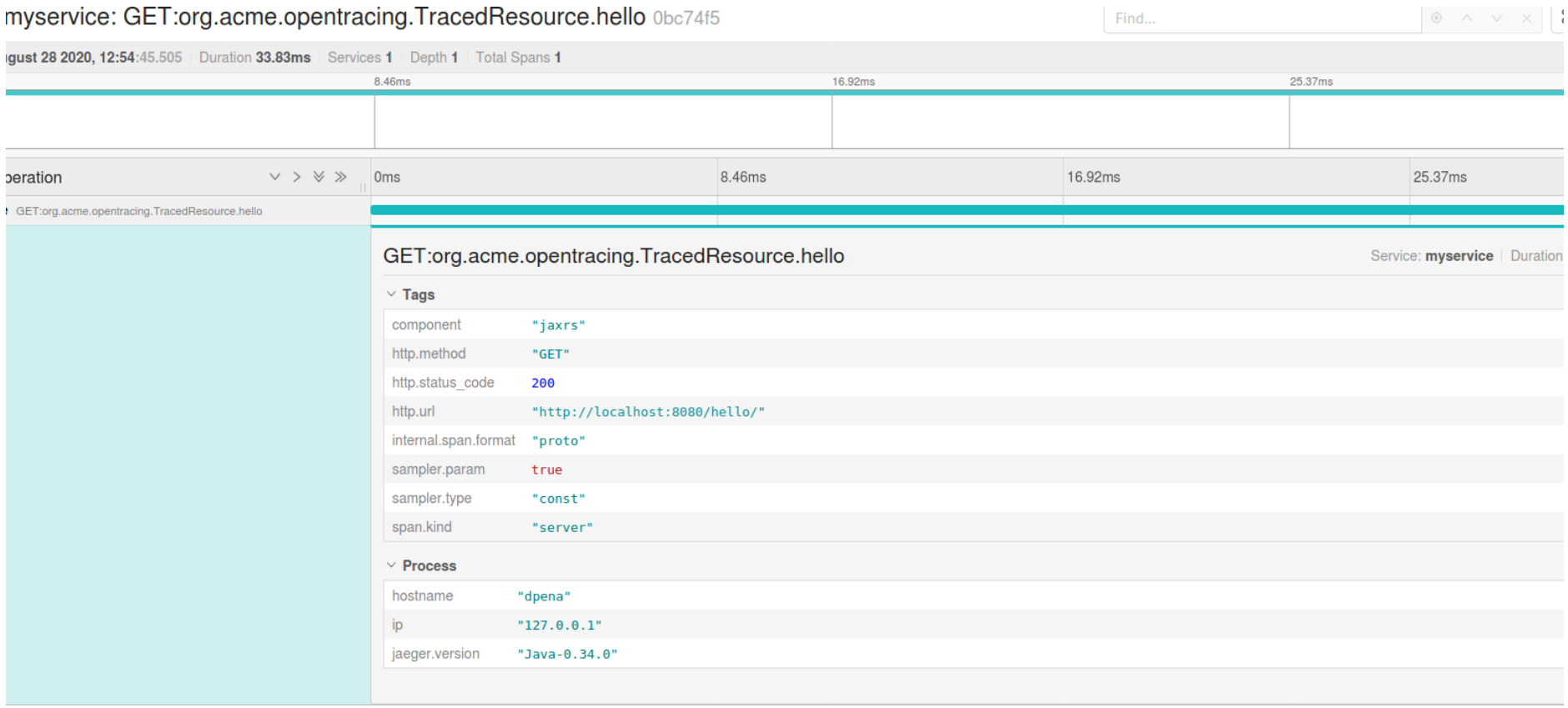
Traces as a way of recording a request as it traverses through the many services in your application.



Logs, or events, are what comes out of your containers in Kubernetes.



Trace data on Jaeger UI



Additionally native span attributes, we also can **customize** or add **additional attributes** to identify or correlate data for our scenario.

Trace Data brief schema in SQL

```
CREATE TABLE IF NOT EXISTS otel_raw_trace(
    Timestamp DateTime64,
    TraceId String,
    SpanId String,
    ParentSpanId String,
    TraceState String,
    SpanName String,
    SpanKind String,
    ServiceName String,
    ResourceAttributes Map(String, String),
    ScopeName String,
    ScopeVersion String,
    SpanAttributes Map(String, String),
    Duration Int64,
    StatusCode String,
    StatusMessage String,
    Events Nested (Timestamp DateTime64, Name String, Attributes Map(String, String)),
    Links Nested (TraceId String, SpanId String, TraceState String, Attributes Map(String, String))
)
```

If need additional key to group aggregation, need to additionally parsing.
Therefore, please discuss feasibility in advance.

[Data by Scenario]

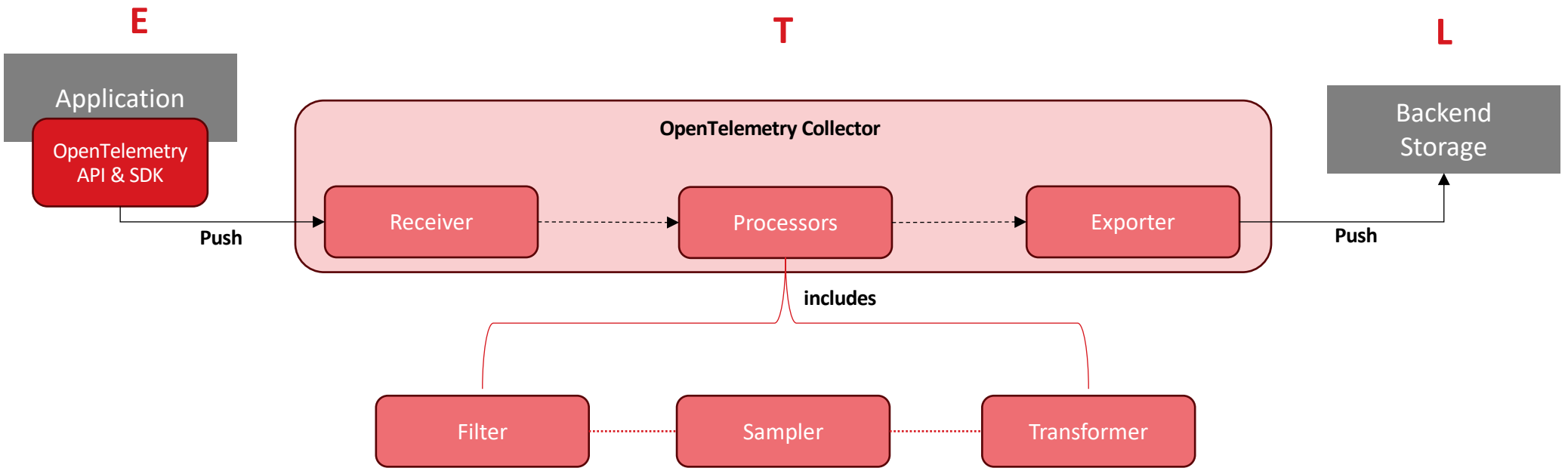
```
SELECT
    ....
FROM otel_raw_trace
WHERE ....
GROUP BY ....
```

```
SELECT
    ....
FROM otel_raw_trace
WHERE ....
GROUP BY ....
```

OpenTelemetry Collector

As an **interface between the service and the backend**, the Collector allows services to simply send data to it. The Collector processes the data before sending it to the backend, achieving decoupling. Additionally, the Collector provides various processors for **data transformation, filtering, sampling, and other processing tasks**.

OpenTelemetry Collector Mechanism Example



OpenTelemetry Collector **Core**: Basic oltp or http protocol to transfer data.

OpenTelemetry Collector **Contrib**: : Provide many opensource & third party components, ex. Kafka, S3, ClickHouse, etc.

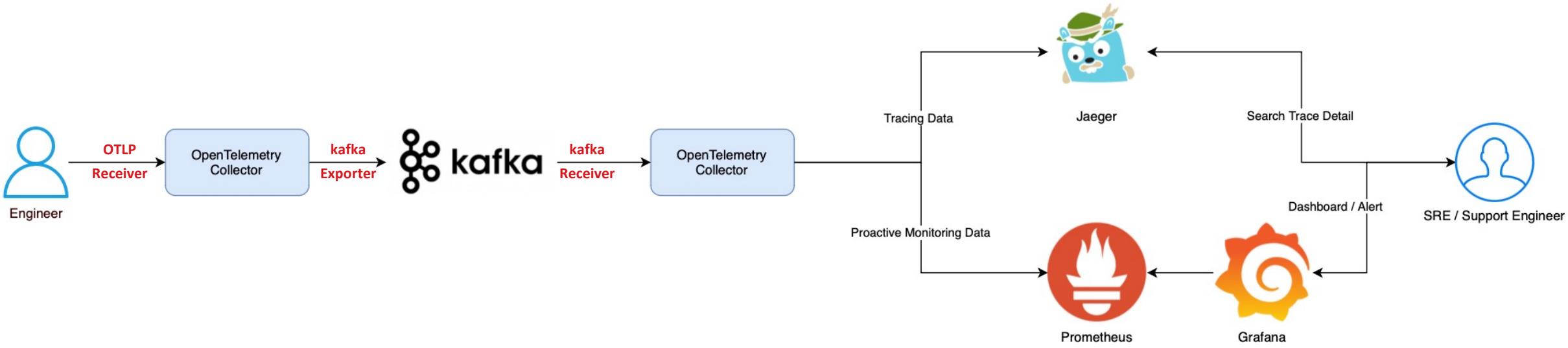
OpenTelemetry SDK on Application

Auto Instrument: Basic & general information.

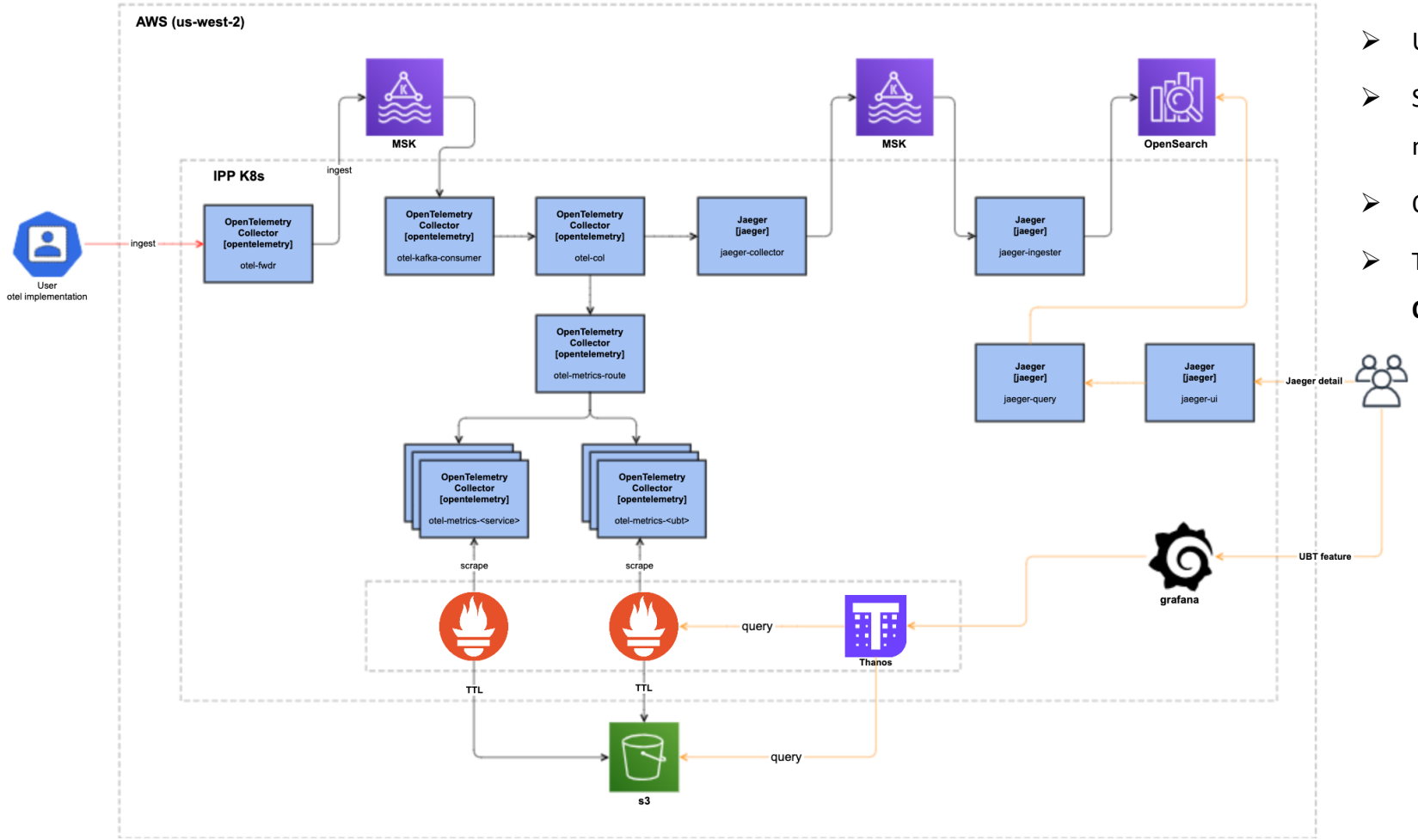
Manual Instrument: Provide customized function or information in traces, logs or metrics

```python
from opentelemetry import trace
from opentelemetry.sdk.trace import TracerProvider
from opentelemetry.sdk.trace.export import (
    BatchSpanProcessor,
    ConsoleSpanExporter,
)

provider = TracerProvider()
processor = BatchSpanProcessor(ConsoleSpanExporter())
provider.add_span_processor(processor)

# Sets the global default tracer provider
trace.set_tracer_provider(provider)

# Creates a tracer from the global tracer provider
tracer = trace.get_tracer("my.tracer.name")

current_span = trace.get_current_span()

current_span.set_attribute("operation.value", 1)
current_span.set_attribute("operation.name", "Saying hello!")
current_span.set_attribute("operation.other-stuff", [1, 2, 3])
```

Rough Architecture V1 – Prometheus/OpenSearch as Storage



Data Architecture V1 – Prometheus/OpenSearch as Storage

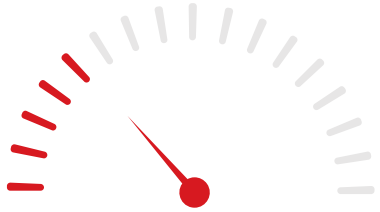


- Using Otel component to do data ingestion & processing
- Scratch tracing data by specific tags data to be the metrics label
- Query proactive monitoring data with **Thanos**.
- Tracing data will pass to **jaeger** and ingest to **OpenSearch** for **query detailed information**.

Data Architecture V1 – Problems & Limitation

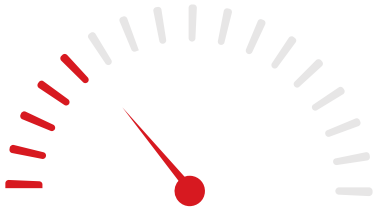
Have some limitation we suffered…

Performance



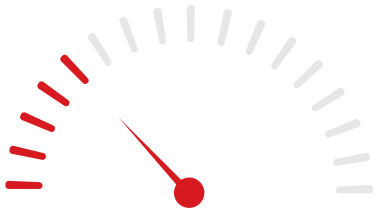
With time moving and increasing of customers, the Prometheus performance not good enough

Flexibility



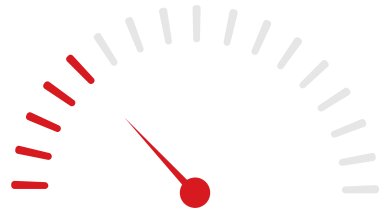
Add fields will greatly affects the performance

Correctness



According to the Prometheus **scrape** data mechanism, the data trend near to the raw data, but the **details is hard to prove it.**

Maintenance



Redundant data store to different storage so that cost and maintenance effort.

How to conquer these limitation…

How to conquer these limitation…

Problem

How to improve the efficiency of querying and analyze large data volume to a certain extent?

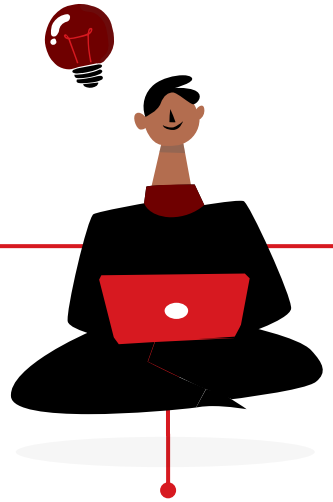
Solution

Back to problem, need provide the proper storage for analysis.

Definition of concepts

Throughput

Can handle massive data ingestion and queries



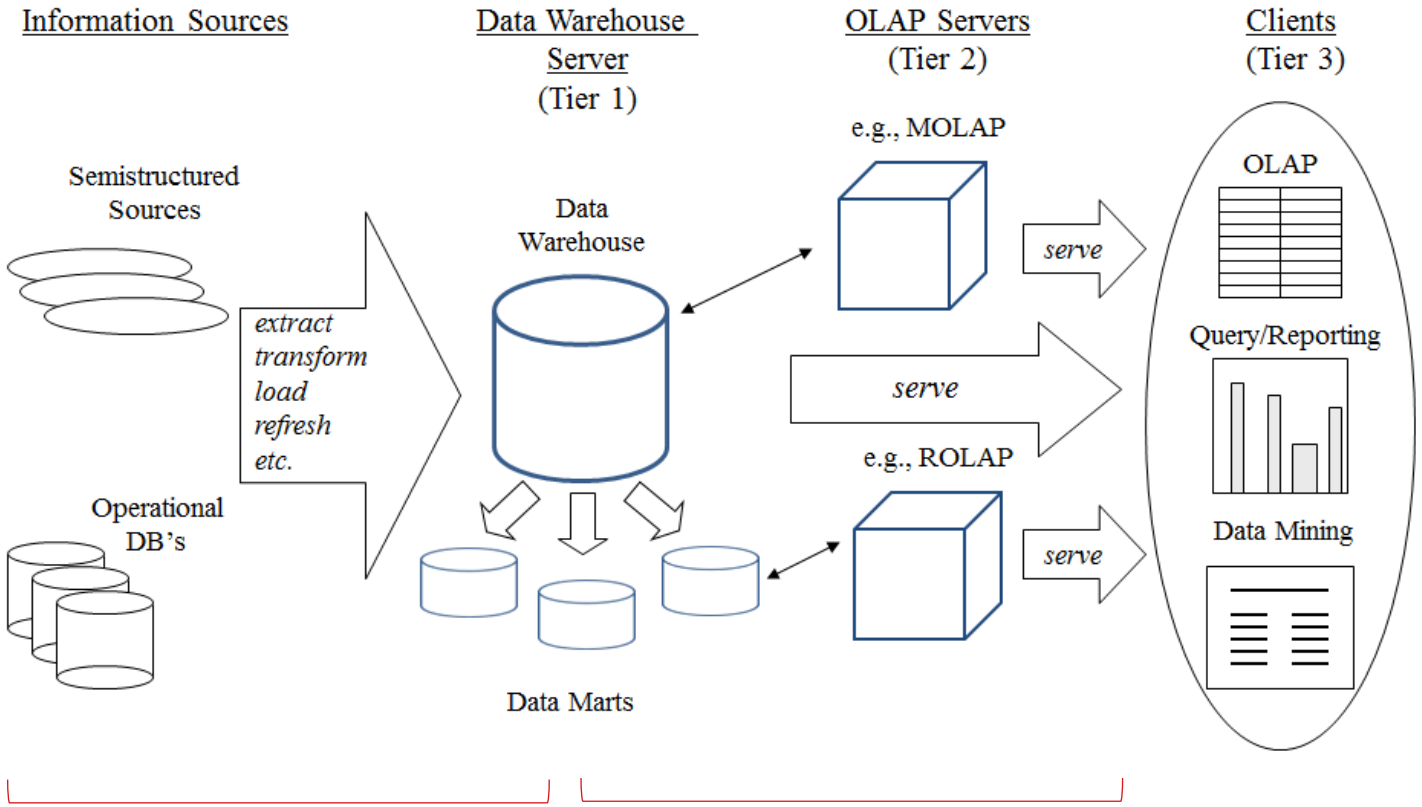
Analysis

Easily generate different analysis for corresponding scenario

Cost

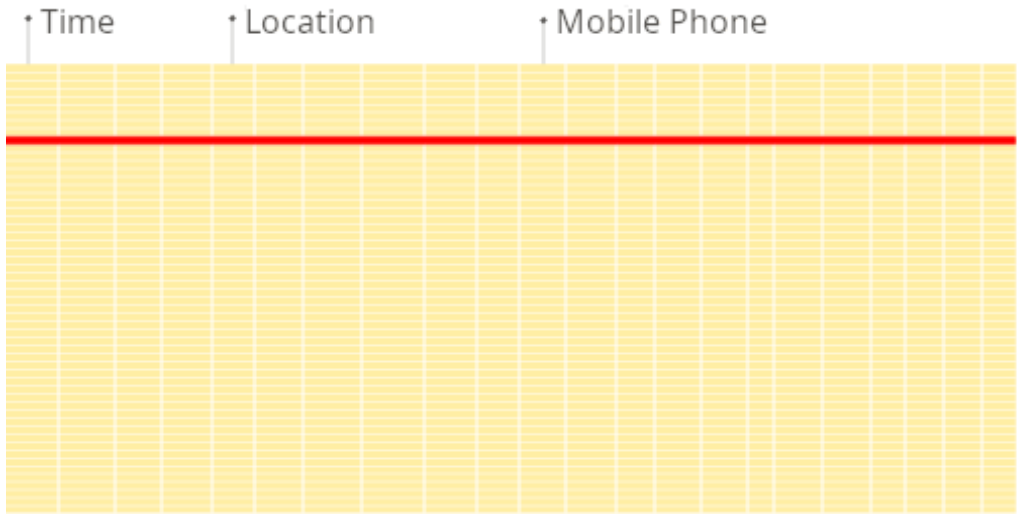
Cost Control Strategy

OLAP Data Warehouse



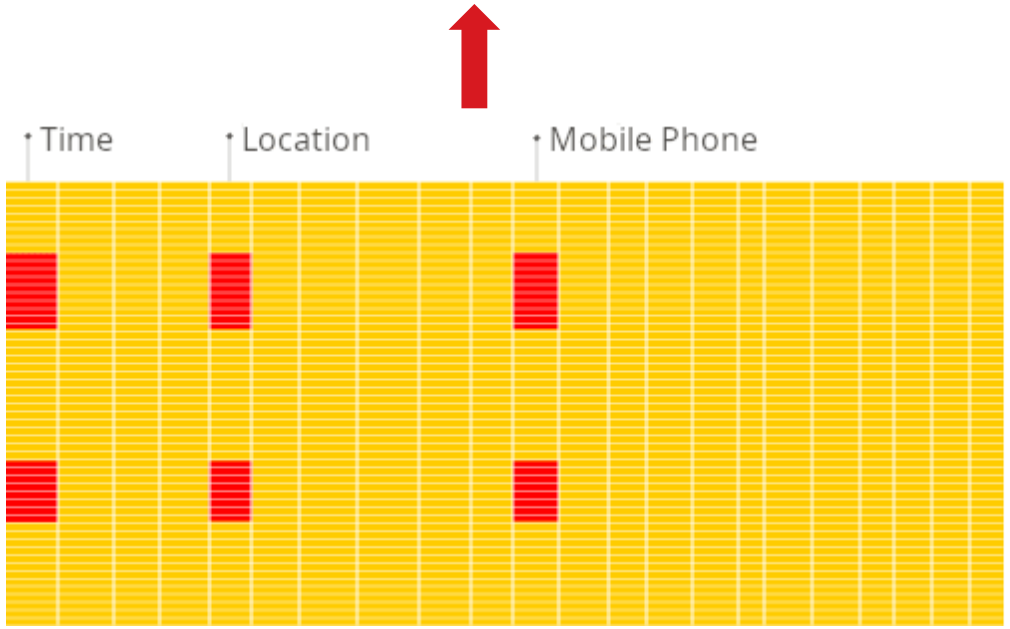
- Column Based Storage
  - Higher storage compaction
  - Using Bit-Wise Scan instead of fulling scan
  - Higher performacne in big data analysis
- Ansi-SQL Language for flexibility
- Pre-Aggragation Mechanism
- Support historical data anaylsis
- Support HA-Cluster Infrastructure.
- Easily Data Mining for value explore
- Easily integrate ML Application.

Row-Based vs. Column-Based



Row Based Storage
(OLTP)

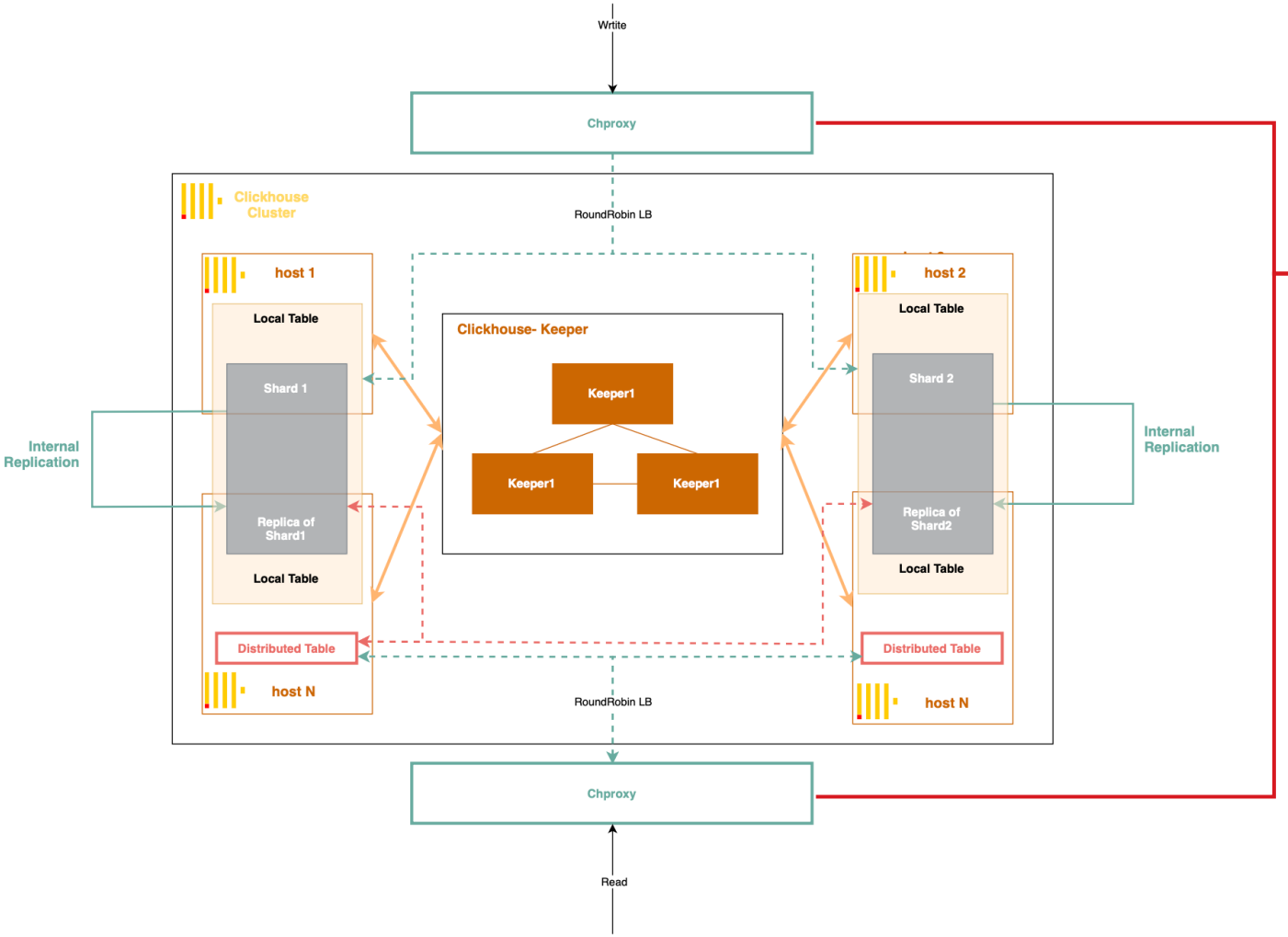
Column Based Storage
(OLAP)



Which OLAP is the best choice?

OLAP Data Warehouse – ClickHouse

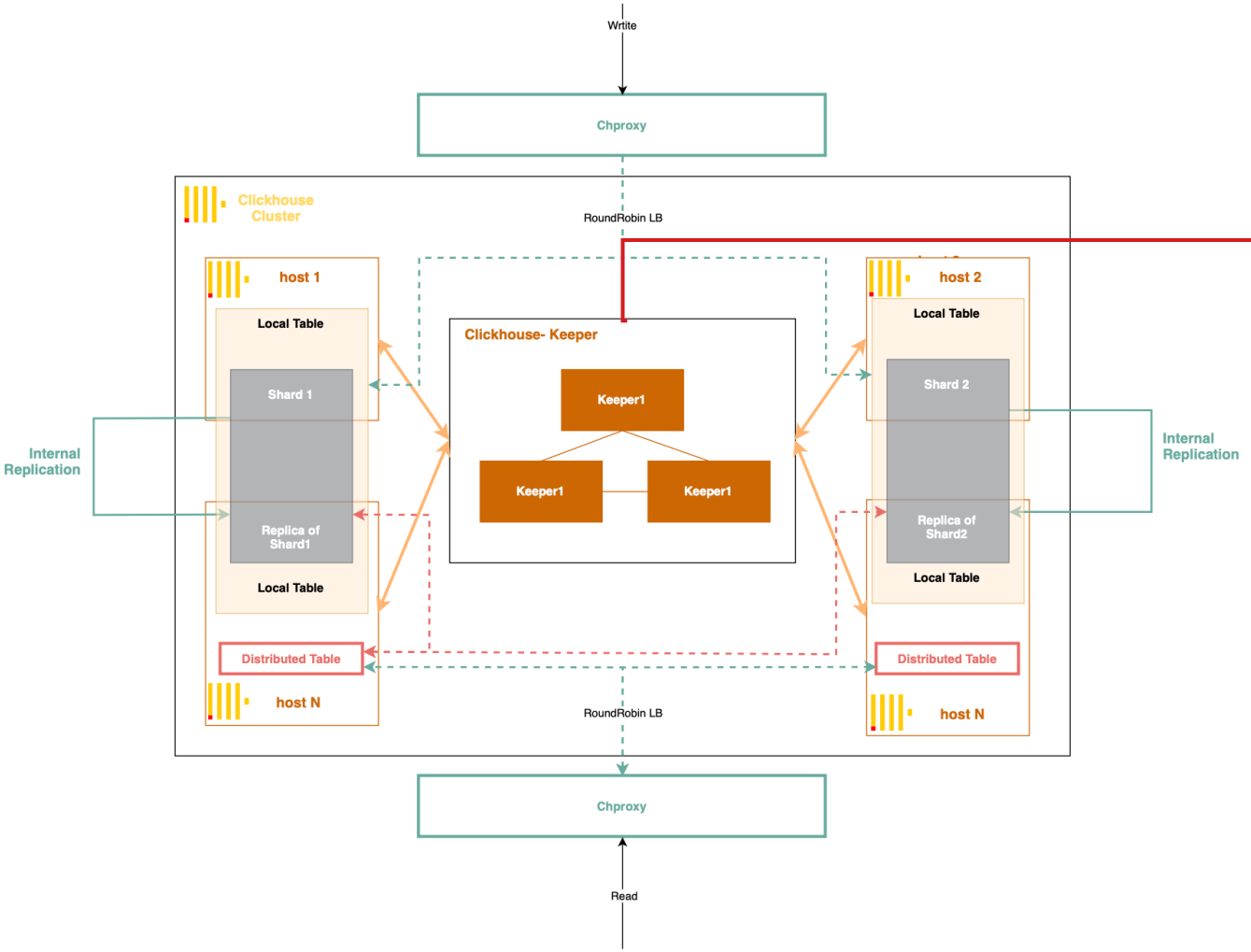
ClickHouse OLAP Architecture



Chproxy

- Routing
- Load Balancing
- User Management (Multi-Tenant)
- Caching (Integrate Redis)

ClickHouse OLAP Architecture

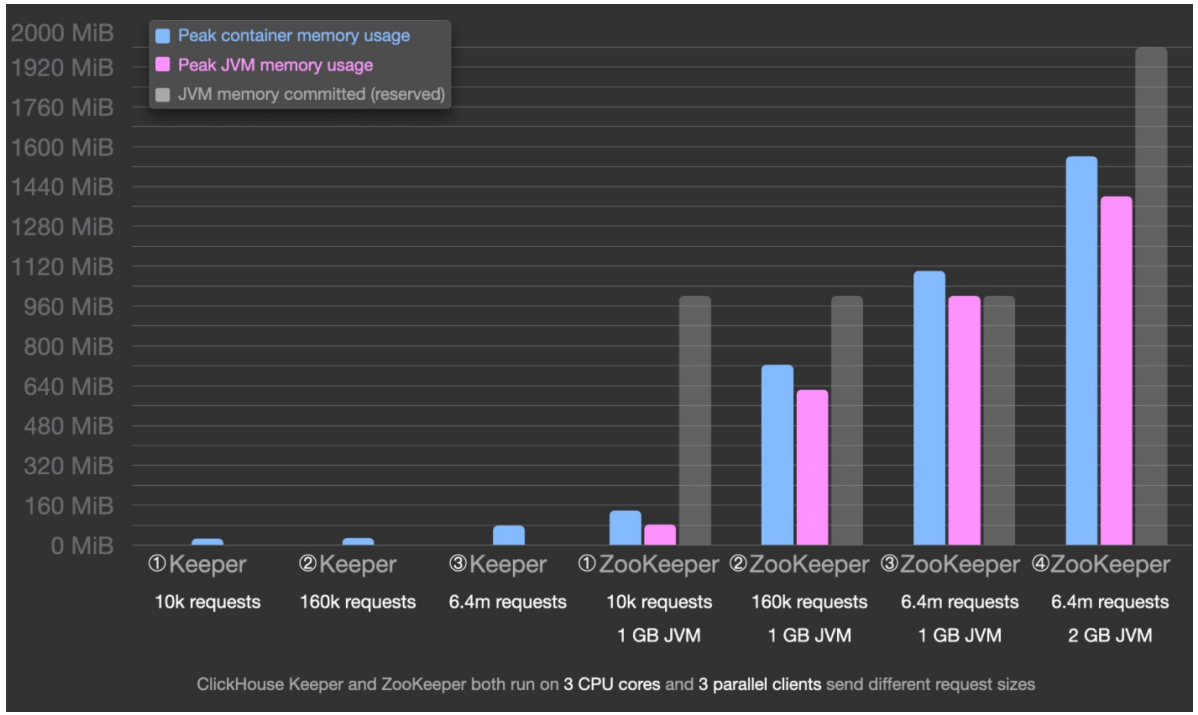


Keeper

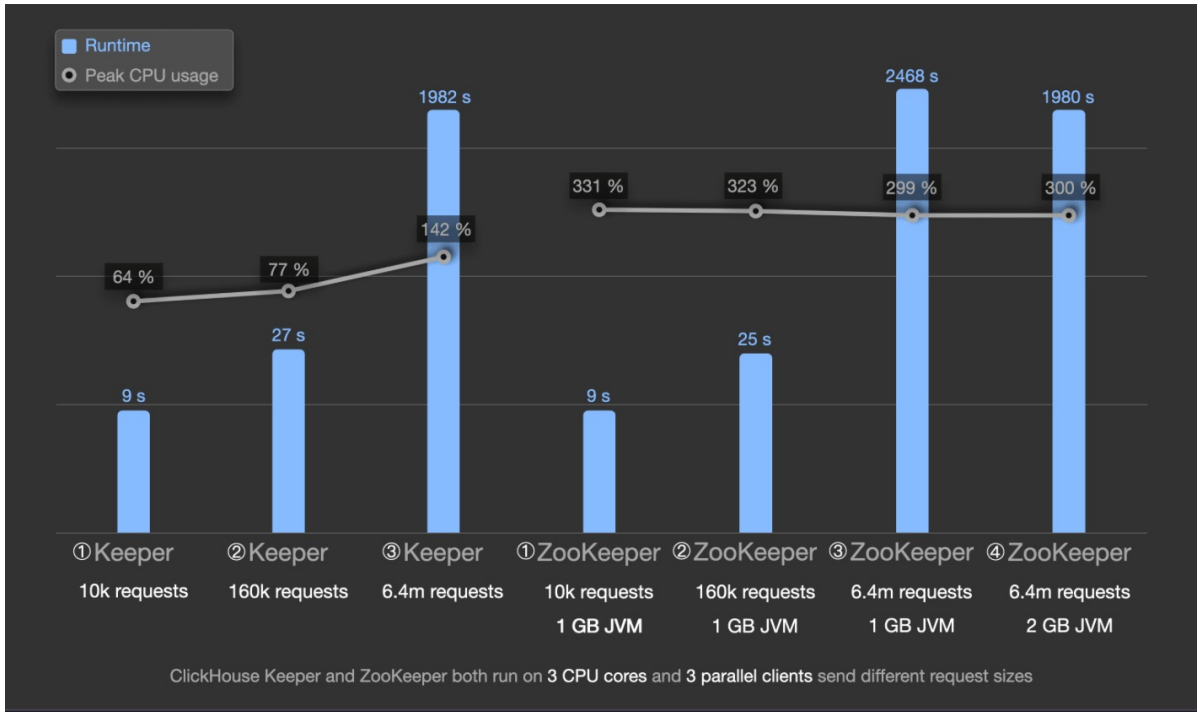
- Replica Sync and node recovery
- **Raft** Protocol for consensus
- Avoid zxid overflow
- Support snapshot and log compaction
- Read/Write Linear Consistency
- Compared to Apache Zookeeper, can use **lower** RAM & CPU to achieve powerful performance.

ClickHouse Keeper vs. Apache Zookeeper

Memory Usage

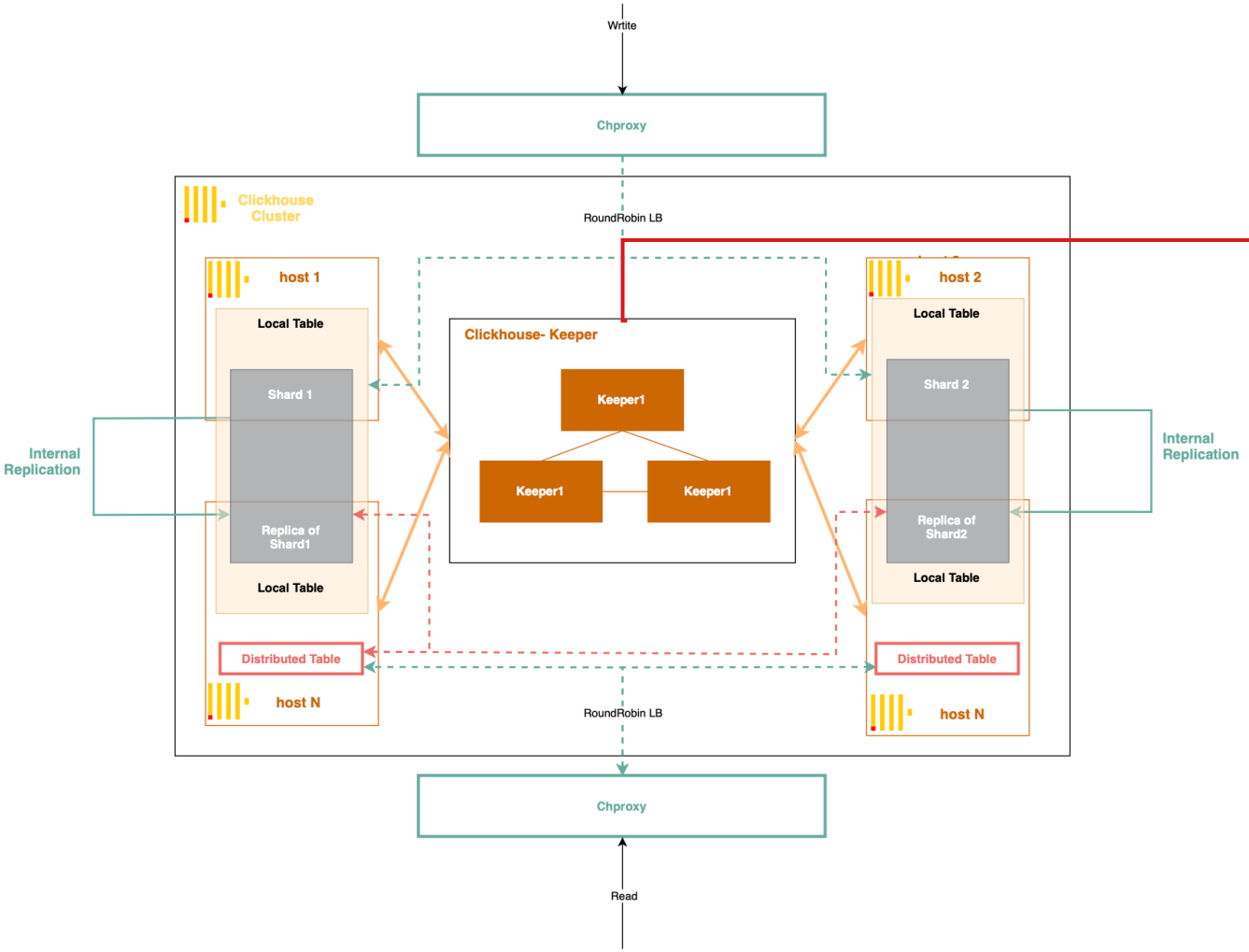


Runtime & CPU Usage



Ref: ClickHouse Keeper: A Zookeeper alternative written in C++

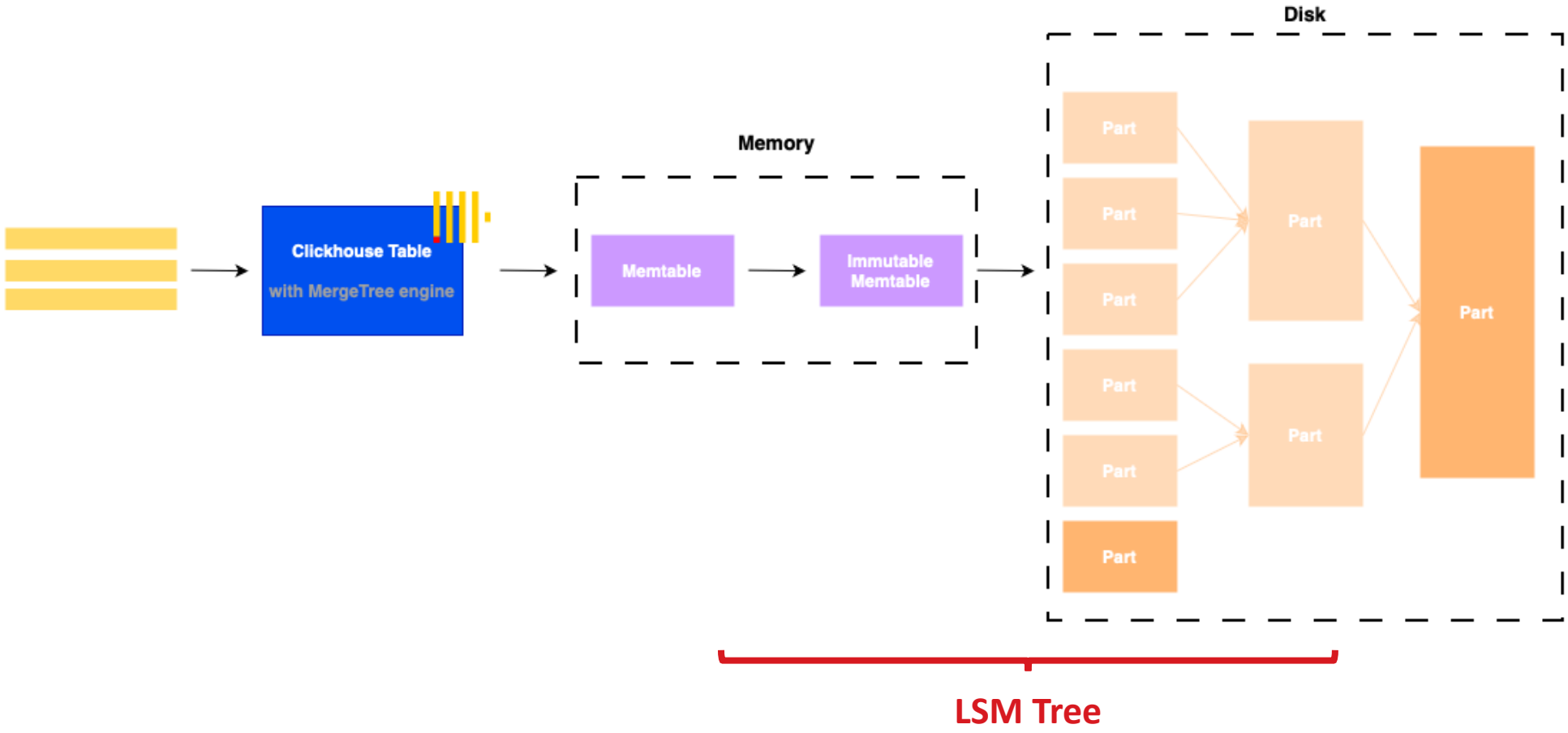
ClickHouse OLAP Architecture



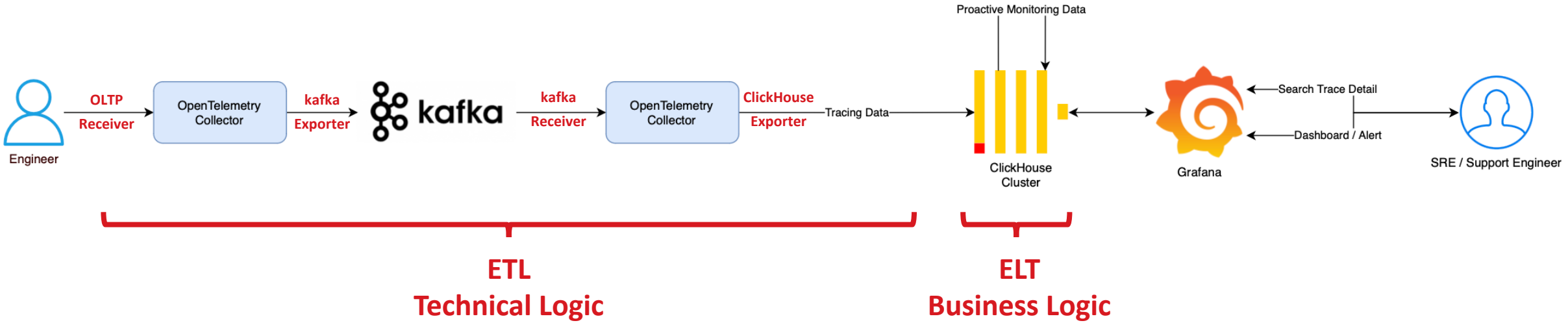
Local & Distributed Table

- **Write** Local Table / **Read** Distributed Table
- **Internal Replication** to sync data to replica from shard

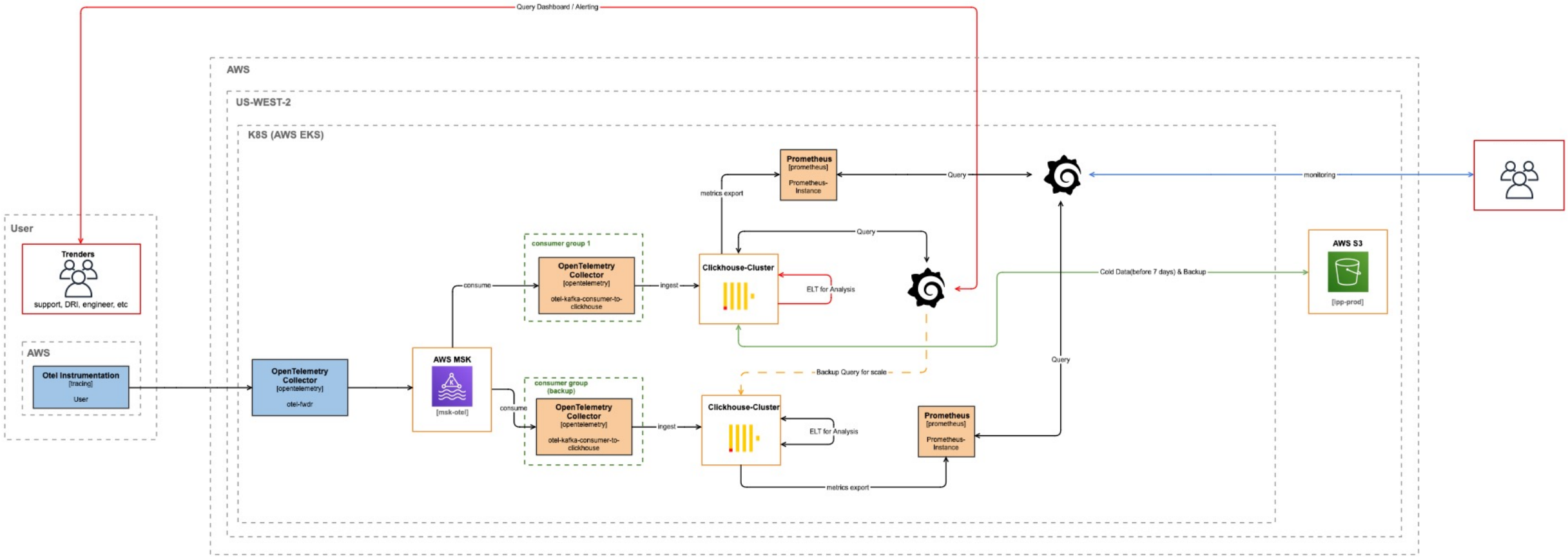
ClickHouse – Data Store Mechanism



Rough Data Architecture V2 – ClickHouse as Storage



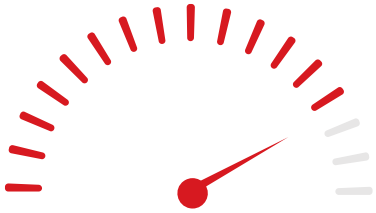
Data Architecture V2 – ClickHouse as Storage



Data Architecture V2 – ClickHouse as Storage

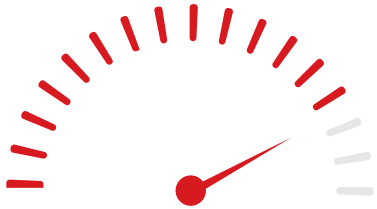
We Improve…

Performance



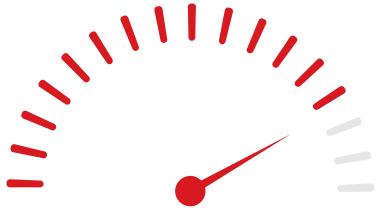
Due to column-based and data compaction, for **higher data volume query** can get shorter query latency.

Flexibility



Apply **Materialized View** & **Projection** mechanism, can quickly create different analysis query table.

Correctness



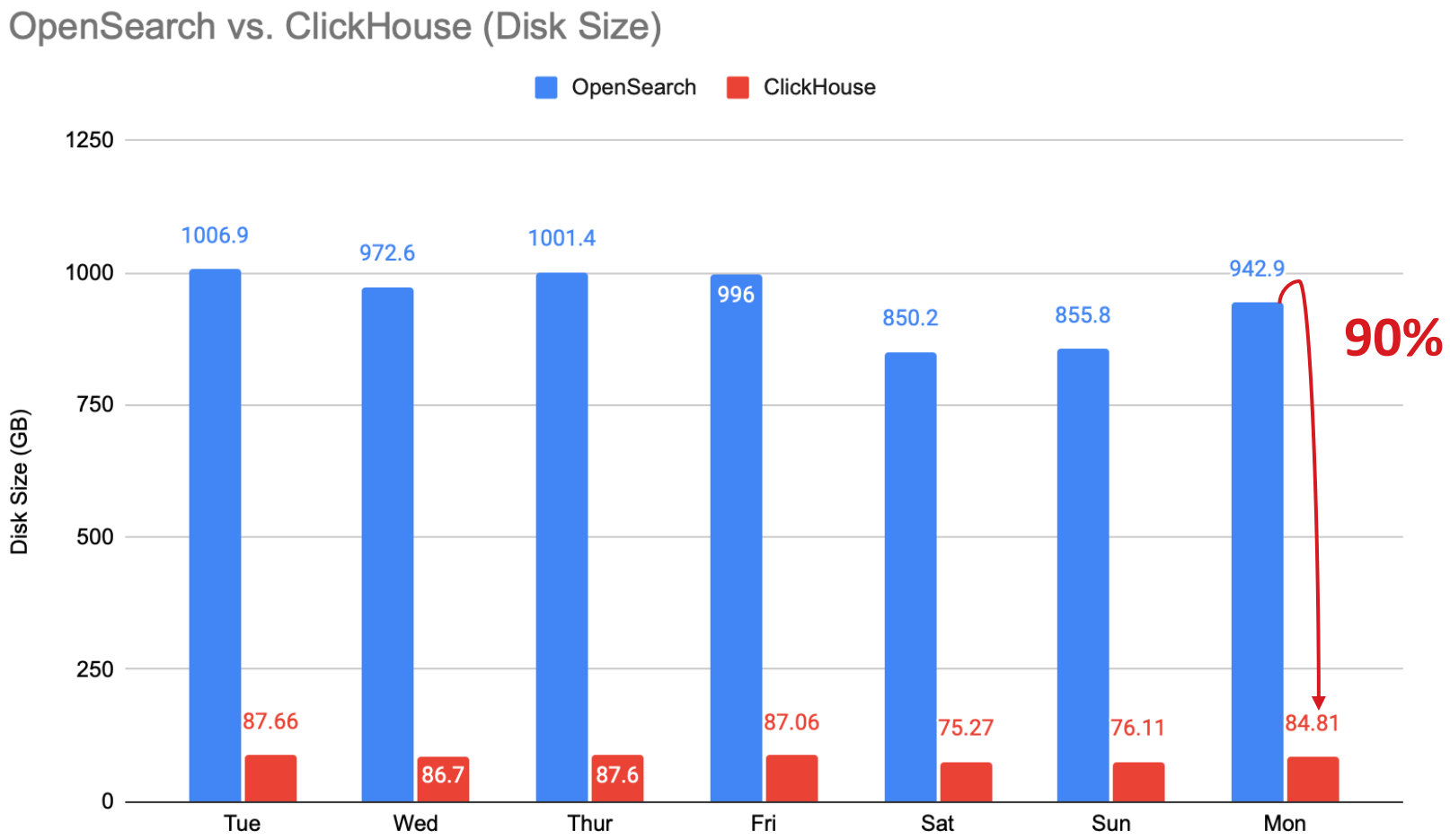
Due to **directly consume raw data** to sink to ClickHouse, so can review trace detailed information.

Maintenance

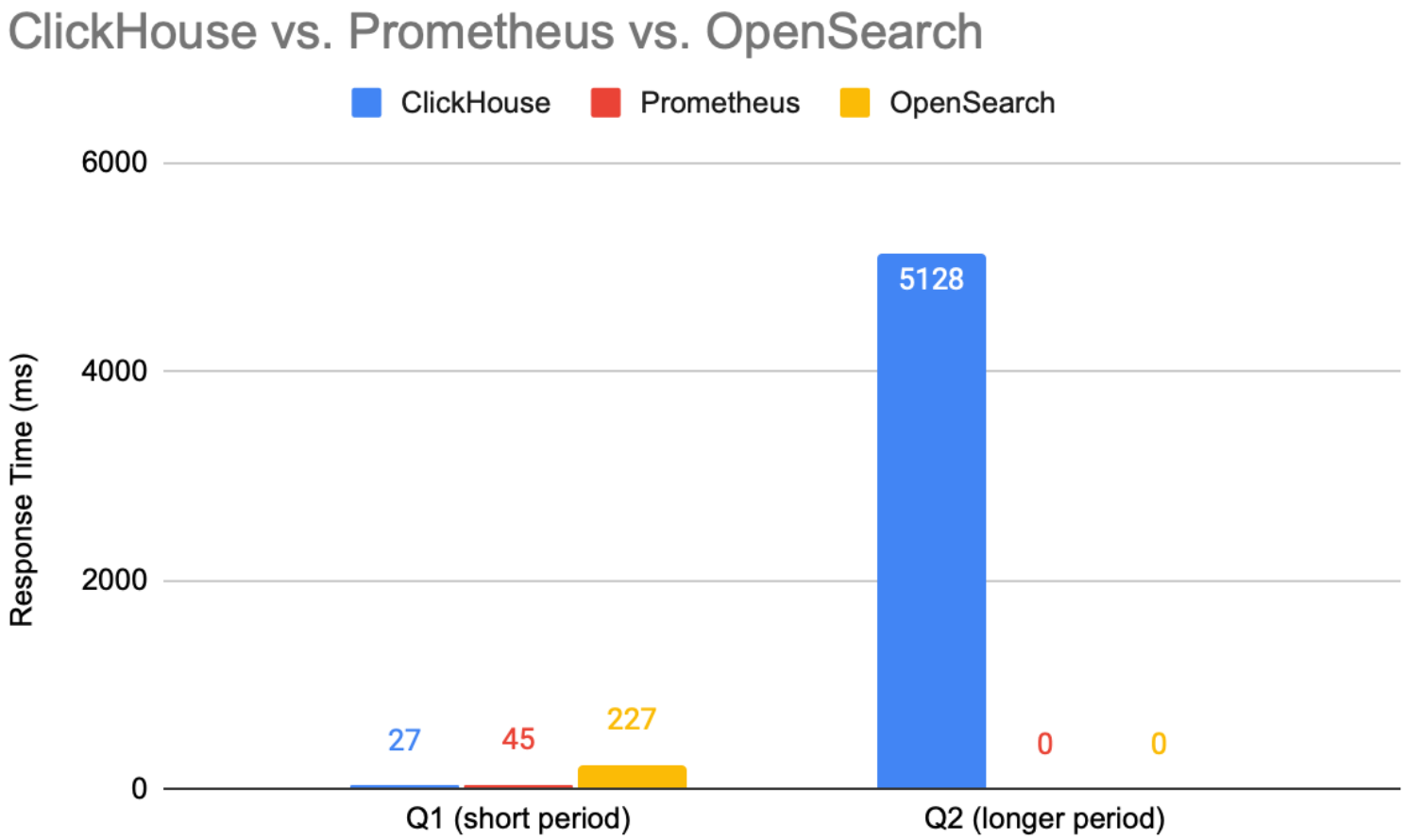
Consolidate storage to provide different analysis for corresponding scenario.

03
Expected Benefit & Future

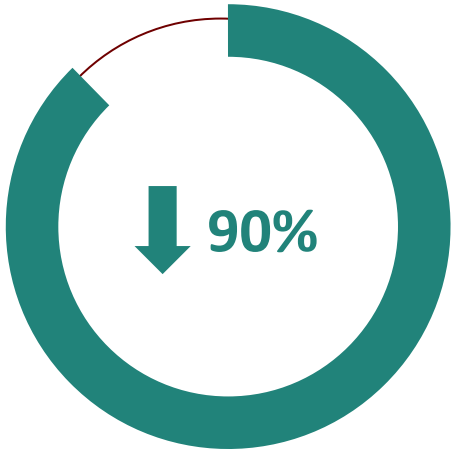
Comparison – Disk Usage



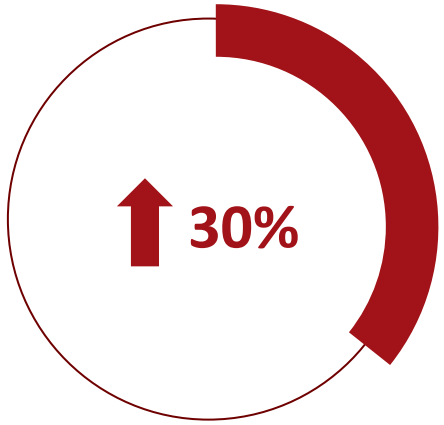
Comparison – Query Performance



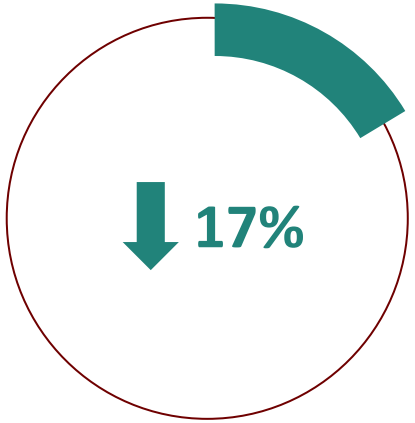
New Data Architecture – Overall Benefit



Disk Usage Size



Query / Write Performance



Cost Effective

Real Notification for Proactive Monitoring

Which Service

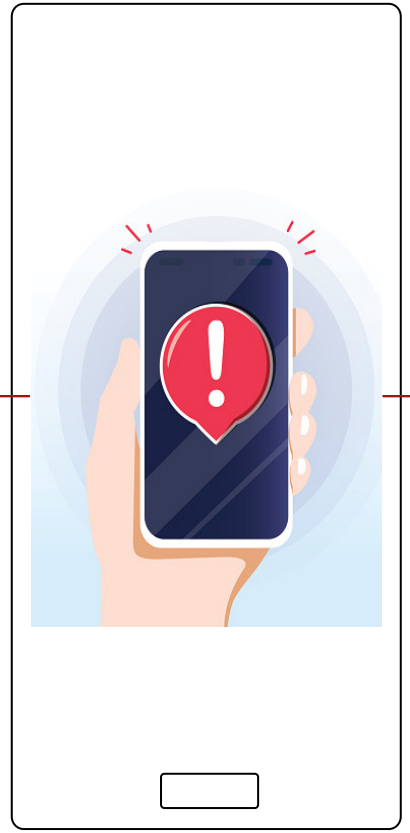
Which service occurs problems?

Which endpoint

Which endpoint of service result in incidents?

Who calls

Who calls this service so that result in incidents?



Trace Status

Real trace status during incident occurs.

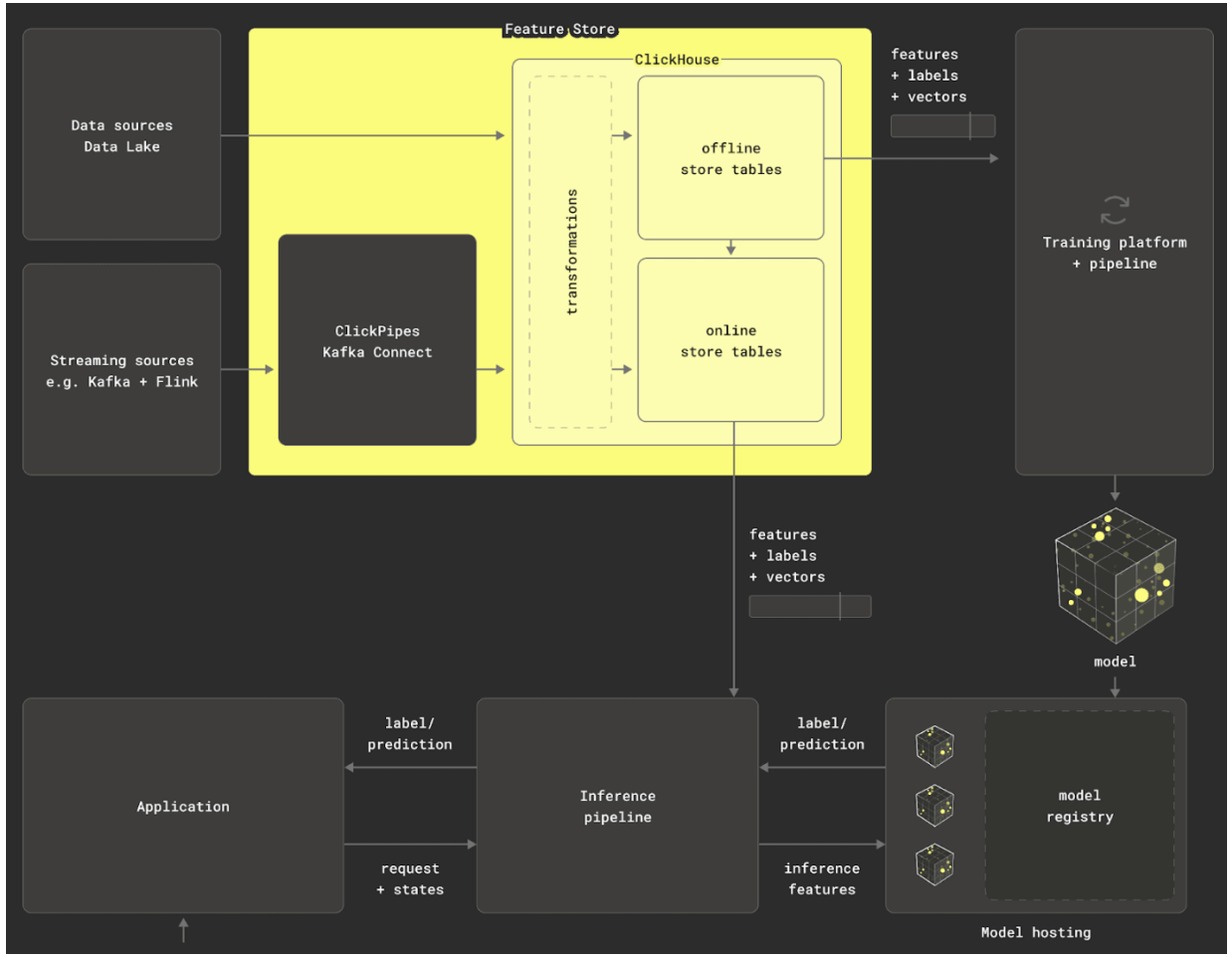
Affected Customers

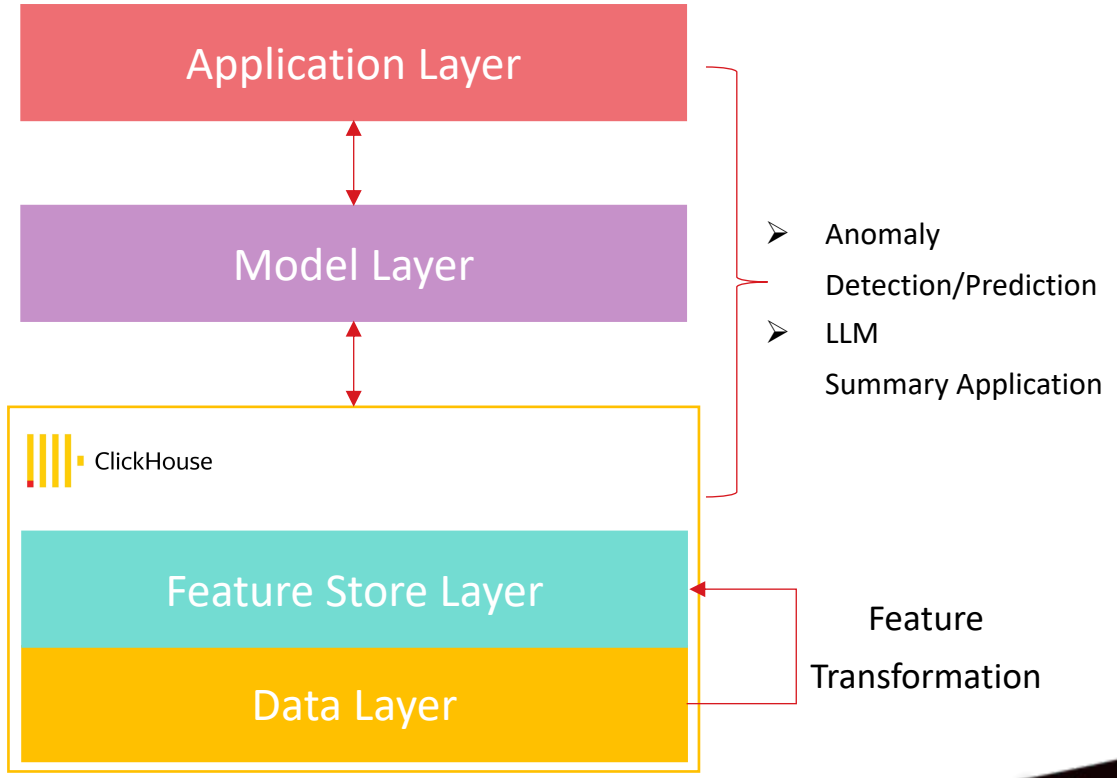
During incidents, how many customers are affected on corresponding level?

Severity Level

Which severity level for incident? High or Moderate?

Next Step in the future – ML/AI Layer





Ref: Powering Feature Stores with ClickHouse

Thanks for Listening.