Building Real-Time Incident Analytics with Apache Flink and Kafka on OpenStack

Ananda Dwi Rahmawati
Cloud & DevOps Engineer

Yogyakarta, 19 July 2025

Ananda Dwi Rahmawati

- Cloud & DevOps Engineer, Singapore
- AWS Container Hero & Google Developer Expert Cloud - Modern Architecture
- Master of Computer Science - University of Texas at Austin
- Alumni of TRPL 2019, Universitas Gadjah Mada
- https://linktr.ee/misskecupbung

"Empowering organizations with real-time analytics transforms incident management from reactive to proactive, unlocking faster decisions, deeper insights, and resilient operations."

3

Agenda

- Key Takeaways
- Problem Statement
- Apache Flink
- Apache Kafka
- OpenStack
- Architecture Diagram + Data Flow

Agenda

- Implementation
- The Challenges
- Use Cases
- Q&A

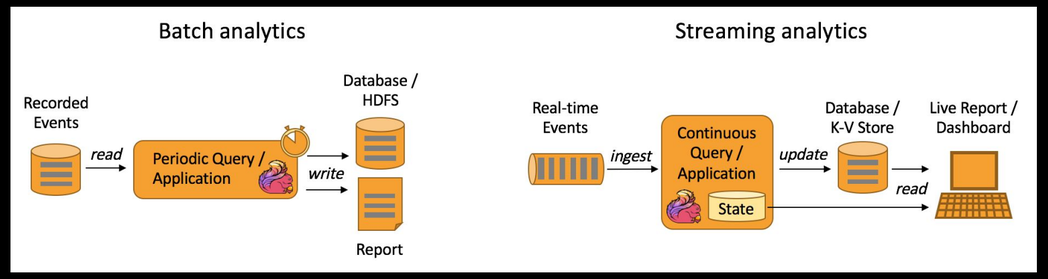
Key Takeaways

- Real-time analytics is crucial for effective incident management
- Scalable, reliable, and fast data processing is essential for modern operations
- Enables proactive incident detection and mitigation
- Cloud-native technologies streamline operations and reduce costs
- Timely data insights support compliance and audit requirements

Problem Statement

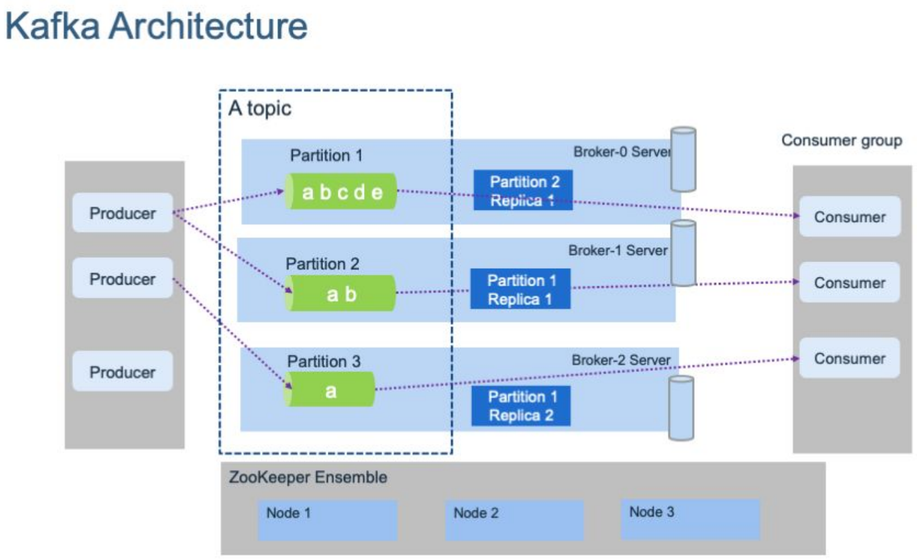
- Traditional batch processing is **slow** for incident response.
- **Growing** data volumes and **complexity**.
- Need for immediate insights and automated actions.
- Difficulty in correlating incidents across multiple data sources in real time.
- Limited **visibility** into ongoing incidents hampers rapid decision-making.

Apache Flink



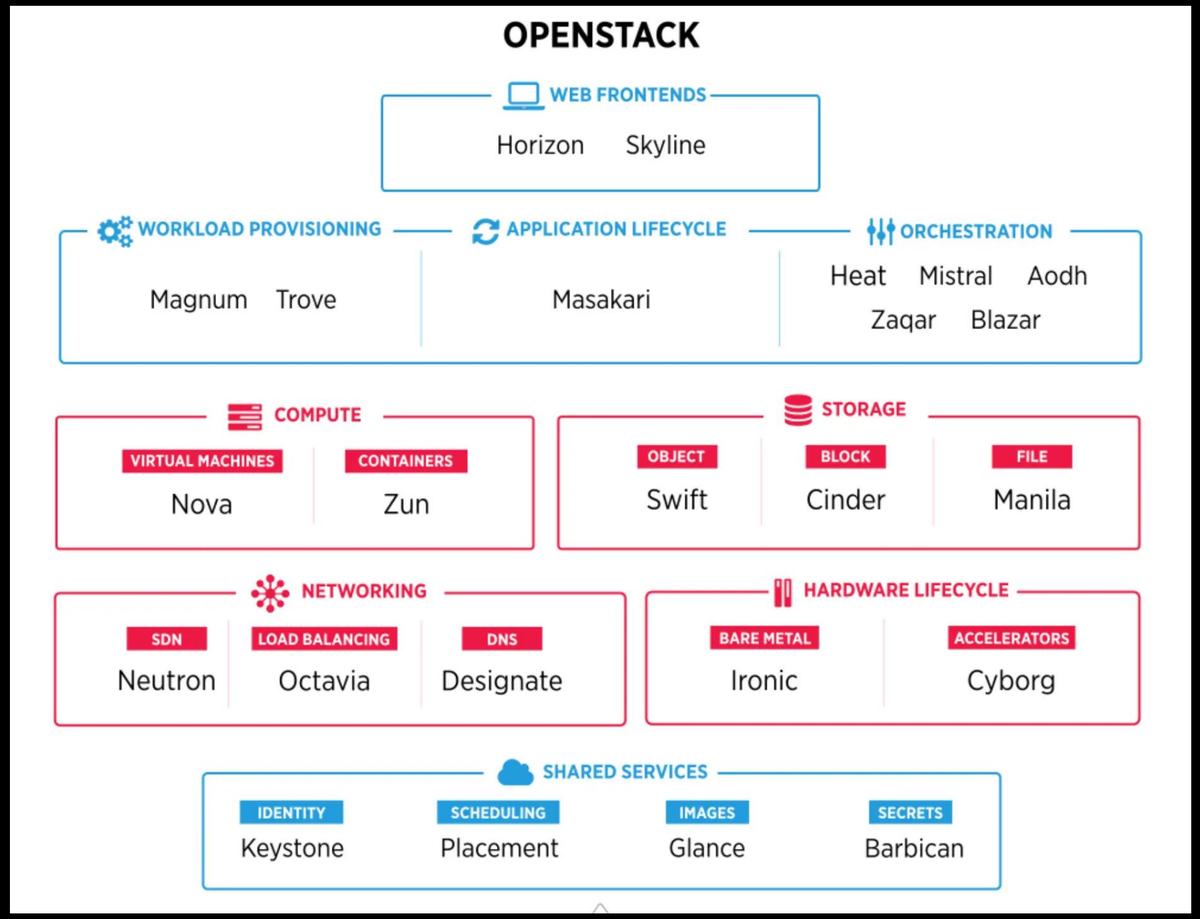
- Stream processing framework
- Low latency, high throughput
- Supports complex event processing
- Supports both stream and batch processing for flexible analytics.
- Provides advanced windowing and state management capabilities.
- Integrates easily with various data sources and sinks.

Apache Kafka



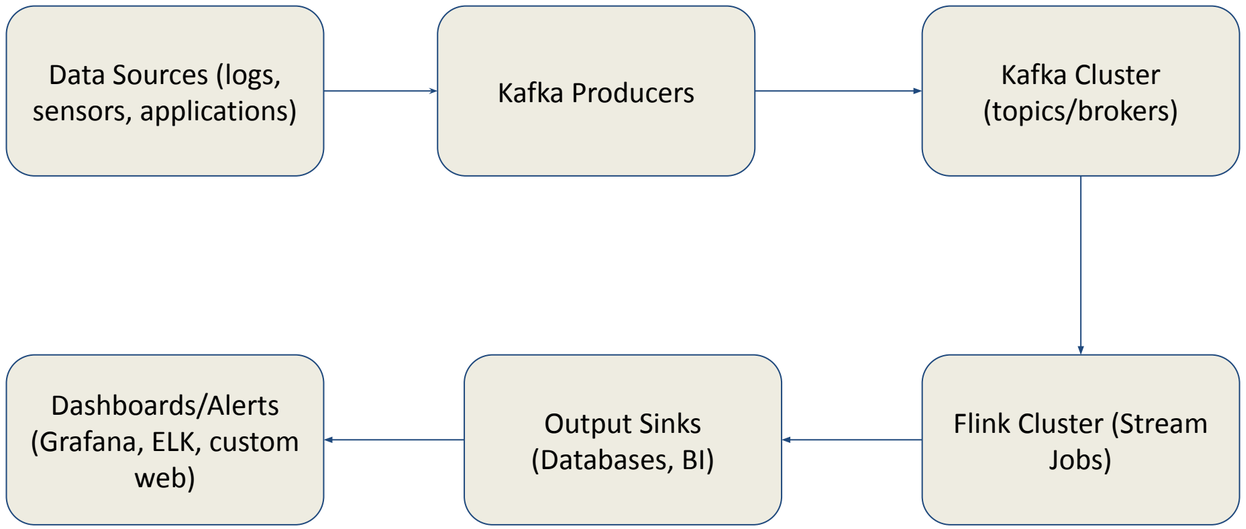
- Distributed messaging system
- Handles high-velocity data streams
- Decouples data producers and consumers
- Guarantees message durability and fault tolerance
- Enables horizontal scaling for high-throughput workloads
- Offers strong ecosystem support for connectors and monitoring

OpenStack Overview



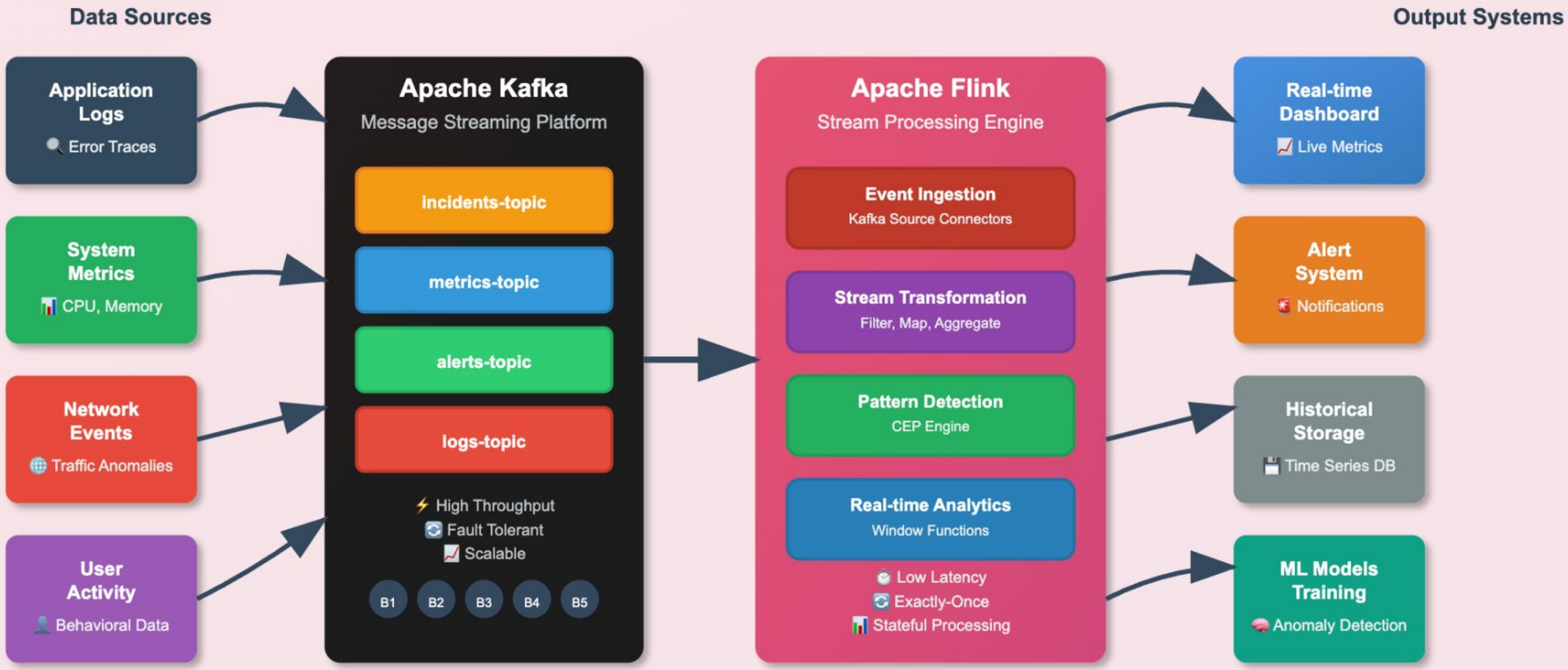
- Open-source cloud platform
- Provides compute, storage, and networking resources
- Enables dynamic scaling of analytics infrastructure
- Supports automation and orchestration for rapid resource provisioning.
- Enables multi-tenancy and isolation for secure analytics environments.

Architecture Diagram



- **Incident Data Generation**
  - Various sources (applications, sensors, logs) generate incident data in real time.
- **Kafka Producers**
  - Producers collect and format incident data.
  - Data is published to specific Kafka topics.
- **Kafka Cluster**
  - Kafka brokers store and manage incoming data streams.
  - Ensures durability, ordering, and availability.
- **Flink Consumers**
  - Flink jobs subscribe to Kafka topics.
  - Flink processes data streams, applies analytics, filtering, and event correlation.
  - Supports windowing, stateful computations, and complex event processing.
- **Output Sinks**
  - Processed results are sent to:
  - Dashboards (e.g., Grafana, Kibana)
  - Alerting systems (e.g., email, SMS, Slack)
  - Databases for storage and further analysis
- **Feedback/Monitoring**
  - Monitoring tools track data flow, system health, and performance.

OpenStack Infrastructure



Key Performance Metrics

- Event Processing Rate: 1M+ events/sec
- End-to-End Latency: < 100ms
- Availability: 99.9% uptime
- Fault Tolerance: Auto-recovery
- Scalability: Horizontal scaling
- Data Retention: 7-day rolling window

Architecture Benefits

- Real-time incident detection and response
- Predictive analytics for proactive monitoring
- Elastic scaling based on workload demands
- Complex event pattern matching
- Multi-tenant isolation and security
- Integration with existing OpenStack services

Common Use Cases

- Security breach detection and response
- Application performance monitoring
- Infrastructure health monitoring
- User behavior analysis
- Compliance and audit trail generation
- Capacity planning and resource optimization

Implementation Step(s)

- Set Up OpenStack Environment
- Deploy Kafka Cluster
  - Install Kafka on provisioned VMs.
  - Configure broker settings and start Kafka services.
  - Example (Kafka start):

```
# On each Kafka node
wget https://downloads.apache.org/kafka/3.6.0/kafka_2.13-3.6.0.tgz
tar -xzf kafka_2.13-3.6.0.tgz
cd kafka_2.13-3.6.0
bin/zookeeper-server-start.sh config/zookeeper.properties &
bin/kafka-server-start.sh config/server.properties &
```

Implementation Step(s)

- Deploy Flink Cluster
  - Install Flink on provisioned VMs.
  - Start Flink JobManager and TaskManager.
  - Example (Flink start):

```
wget https://archive.apache.org/dist/flink/flink-1.17.1/flink-1.17.1-bin-scala_2.12.tgz
tar -xzf flink-1.17.1-bin-scala_2.12.tgz
cd flink-1.17.1
./bin/start-cluster.sh
```

Implementation Step(s)

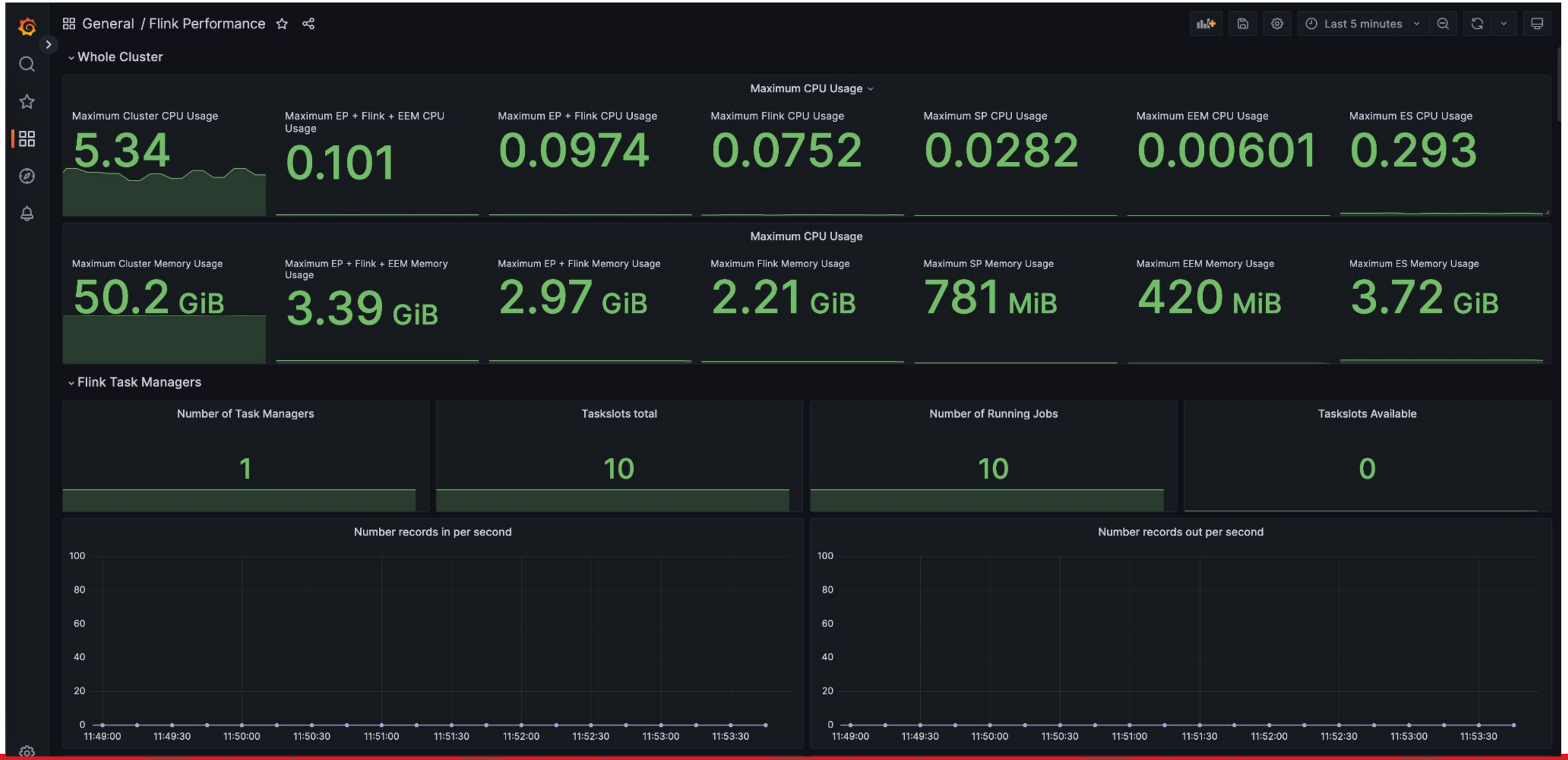
- Develop Flink Jobs for Incident Analytics (Python)
  - Write a Flink job using **PyFlink** to consume from Kafka and process incidents.
  - Example (PyFlink job):

```
from pyflink.datastream import StreamExecutionEnvironment
from pyflink.datastream.connectors import FlinkKafkaConsumer
from pyflink.common.serialization import SimpleStringSchema

env = StreamExecutionEnvironment.get_execution_environment()
properties = {'bootstrap.servers': 'localhost:9092', 'group.id': 'incident-group'}
consumer = FlinkKafkaConsumer(
    topics='incident-topic',
    deserialization_schema=SimpleStringSchema(),
    properties=properties
)
stream = env.add_source(consumer)
alerts = stream.filter(lambda data: 'CRITICAL' in data)
alerts.print()
env.execute('Incident Analytics Job')
```

The Challenges (the Next Step!)

- Managing state and fault tolerance in Flink
- Ensuring data consistency in Kafka
- Resource allocation and scaling on OpenStack
- Security and access control
- Integrating with legacy systems and diverse data sources.
- Monitoring and troubleshooting distributed components.
- Balancing performance with cost efficiency.


General / Flink Performance
Last 5 minutes
Whole Cluster
Maximum CPU Usage
Maximum Cluster CPU Usage
5.34
Maximum EP + Flink + EEM CPU Usage
0.101
Maximum EP + Flink CPU Usage
0.0974
Maximum Flink CPU Usage
0.0752
Maximum SP CPU Usage
0.0282
Maximum EEM CPU Usage
0.00601
Maximum ES CPU Usage
0.293
Maximum CPU Usage
Maximum Cluster Memory Usage
50.2 GiB
Maximum EP + Flink + EEM Memory Usage
3.39 GiB
Maximum EP + Flink Memory Usage
2.97 GiB
Maximum Flink Memory Usage
2.21 GiB
Maximum SP Memory Usage
781 MiB
Maximum EEM Memory Usage
420 MiB
Maximum ES Memory Usage
3.72 GiB
Flink Task Managers
Number of Task Managers
1
Taskslots total
10
Number of Running Jobs
10
Taskslots Available
0
Number records in per second
100
80
60
40
20
0
11:49:00
11:49:30
11:50:00
11:50:30
11:51:00
11:51:30
11:52:00
11:52:30
11:53:00
11:53:30
Number records out per second
100
80
60
40
20
0
11:49:00
11:49:30
11:50:00
11:50:30
11:51:00
11:51:30
11:52:00
11:52:30
11:53:00
11:53:30


OpenTelemetry / Apache Flink Receiver
Add
Last 1 hour
1m
host
sandbox-gorgon-30s-0004
JVM GC Collections
5
4
3
2
14:15
14:30
14:45
15:00
JVM Threads Count
100
80
60
14:15
14:30
14:45
15:00
Monitoring Apache Flink with the OTel Flinkmetrics Receiver
Requirements
Java 8 or higher (Java 11 or 17 recommended)
Linux (tested on Ubuntu 20.04)
Install & Unpack Flink
wget https://downloads.apache.org/flink/flink-1.18.1/flink-1.18.1-bin-scala_2.12.tgz
sudo tar -xzf flink-1.18.1-bin-scala_2.12.tgz
cd flink-1.18.1
Start Flink Cluster
sudo ./bin/start-cluster.sh
CPU Load
2
1.5
1
0.5
0
14:15
14:30
14:45
15:00
CPU Time
600 ms
400 ms
200 ms
0 ns
14:30
15:00
Total Managed Mem
512 MiB
Direct Mem Used
108 MiB
Heap Mem Used
384 MiB
256 MiB
128 MiB
14:30
15:00
Heap
Non Heap
Heap Mem Committed
768 MiB
512 MiB
256 MiB
0 B
14:30
15:00
Heap Committed
Non Heap Committed

QnA

(Answers not Guarantee)

EasyStack open cloud computing

AMD

datacomm flexi WOWRACK

ZConverter Cloud

SIVALI CLOUD TECHNOLOGY

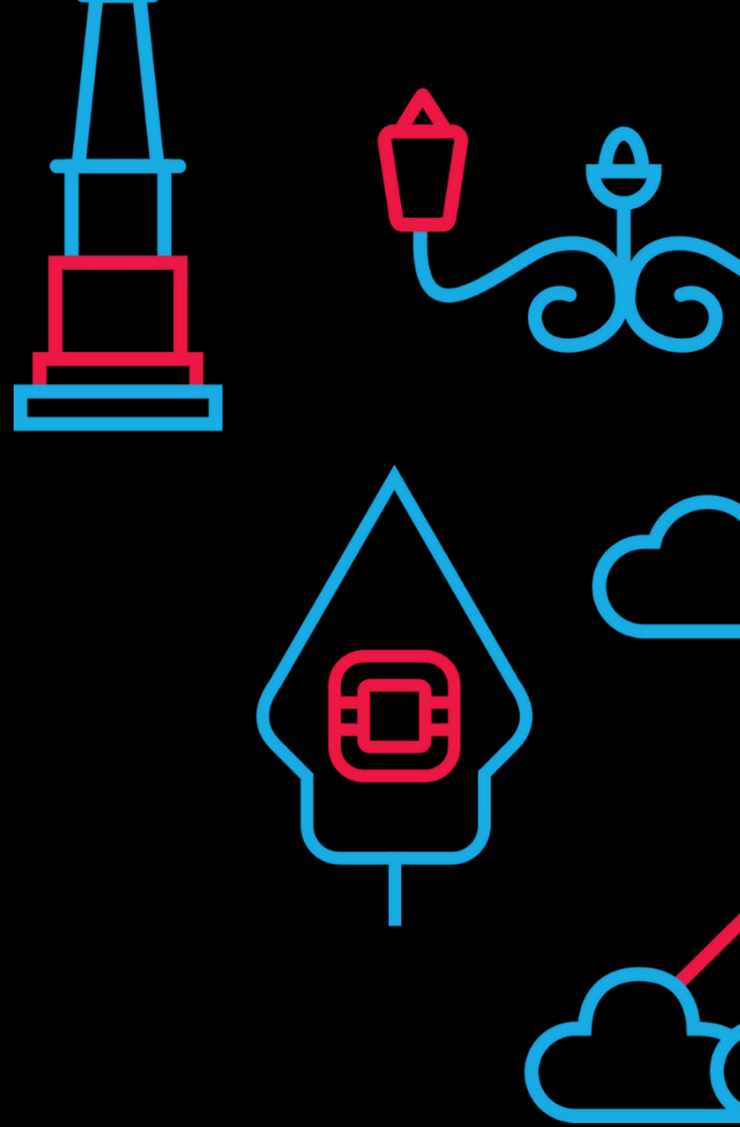
boer technology

NASHTAGROUP

nevacloud

Yogyakarta, 19 July 2025

THANK YOU



EasyStack open cloud computing

AMD

datacomm flexi WOWRACK

ZConverter Cloud

SIVALI CLOUD TECHNOLOGY

boer technology

NASHTAGROUP

nevacloud

Yogyakarta, 19 July 2025