Kafka 사용사례

2018. 09. 08.

류지형
Kaiser.Ryu

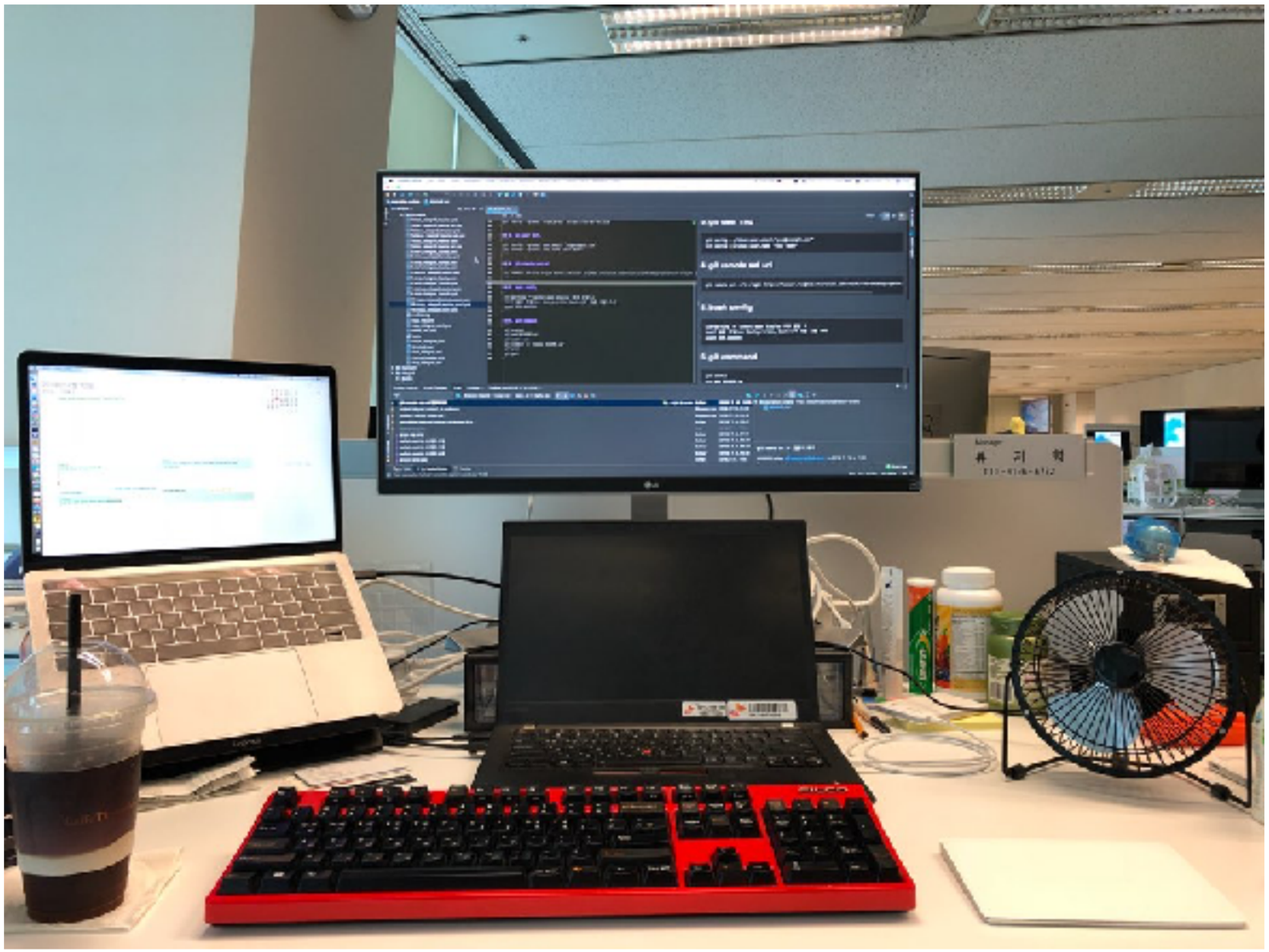
목차

PART 1.

Who am I

Who am I

- 류지형(Kaiser.Ryu)
- SK텔레콤 Datalake 운영
- JAVA, Hadoop, Hive, HBase
  MySQL, Bash shell, Python
- Kafka 경험 약 6개월

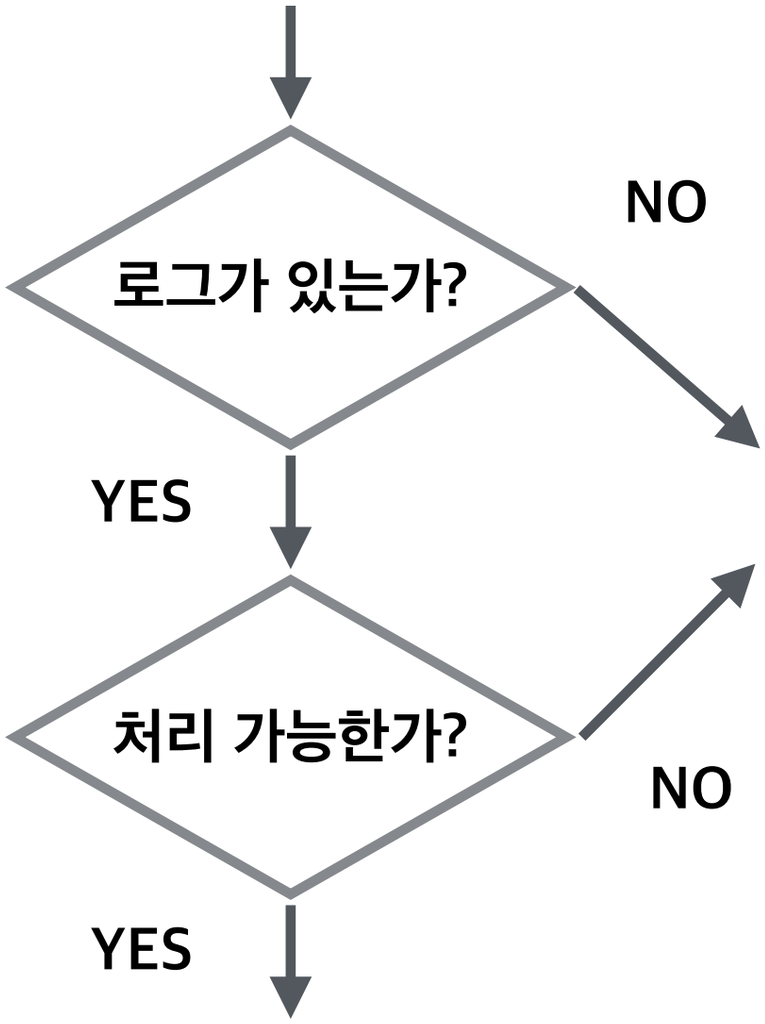


PART 2.
모니터링 시스템

모니터링 시스템

왜 필요한가?

- 장애예방
- 빠른 장애 인지
- 정확한 원인 파악
- 신속한 처리
- 재발 방지



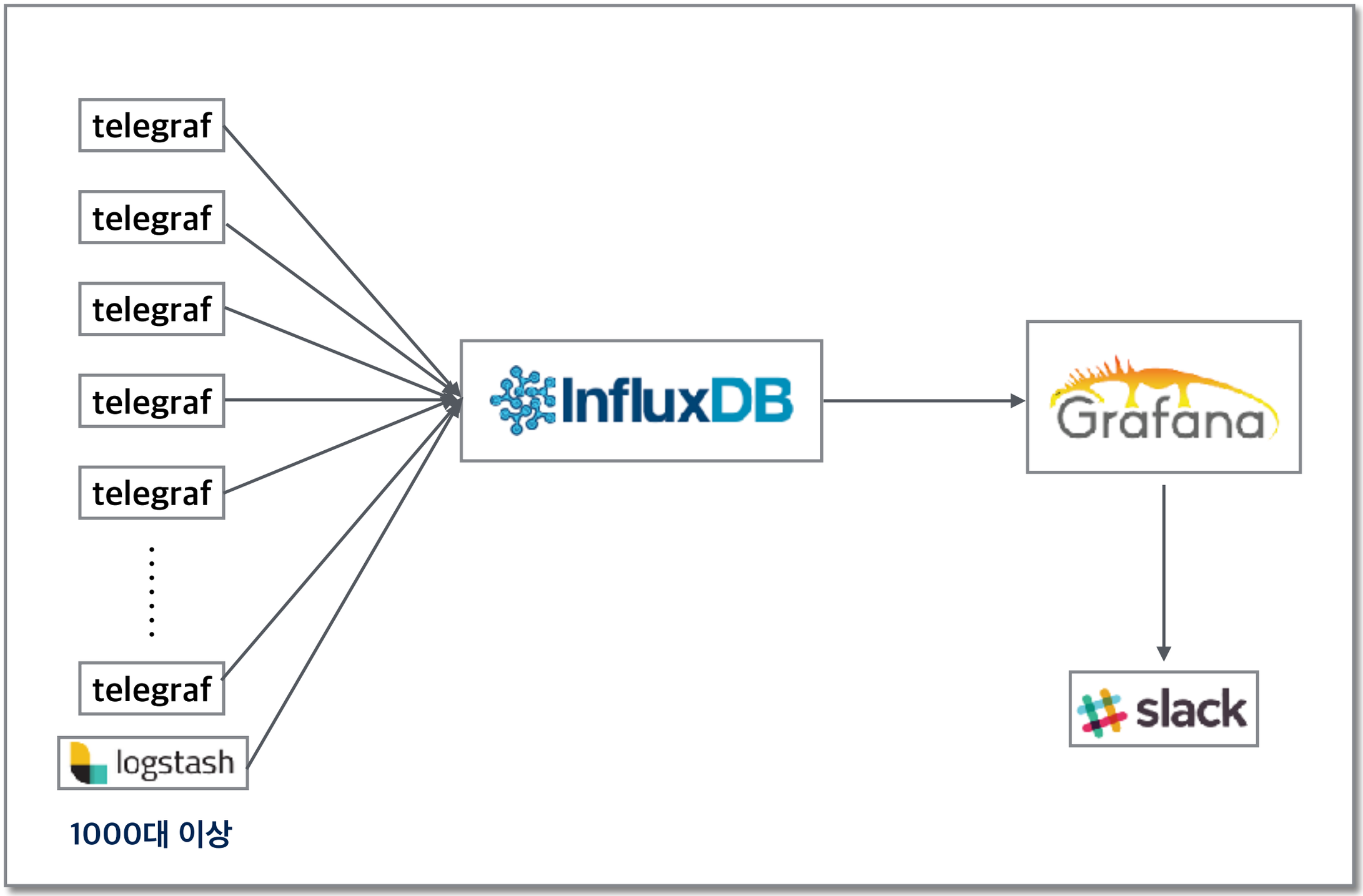
로그 및 메트릭 수집은 매우 중요

PART 3.
서비스 구성도

AS-IS

수집 서버 1대가 1000대 이상 서버의 metric을 전송받는 형태

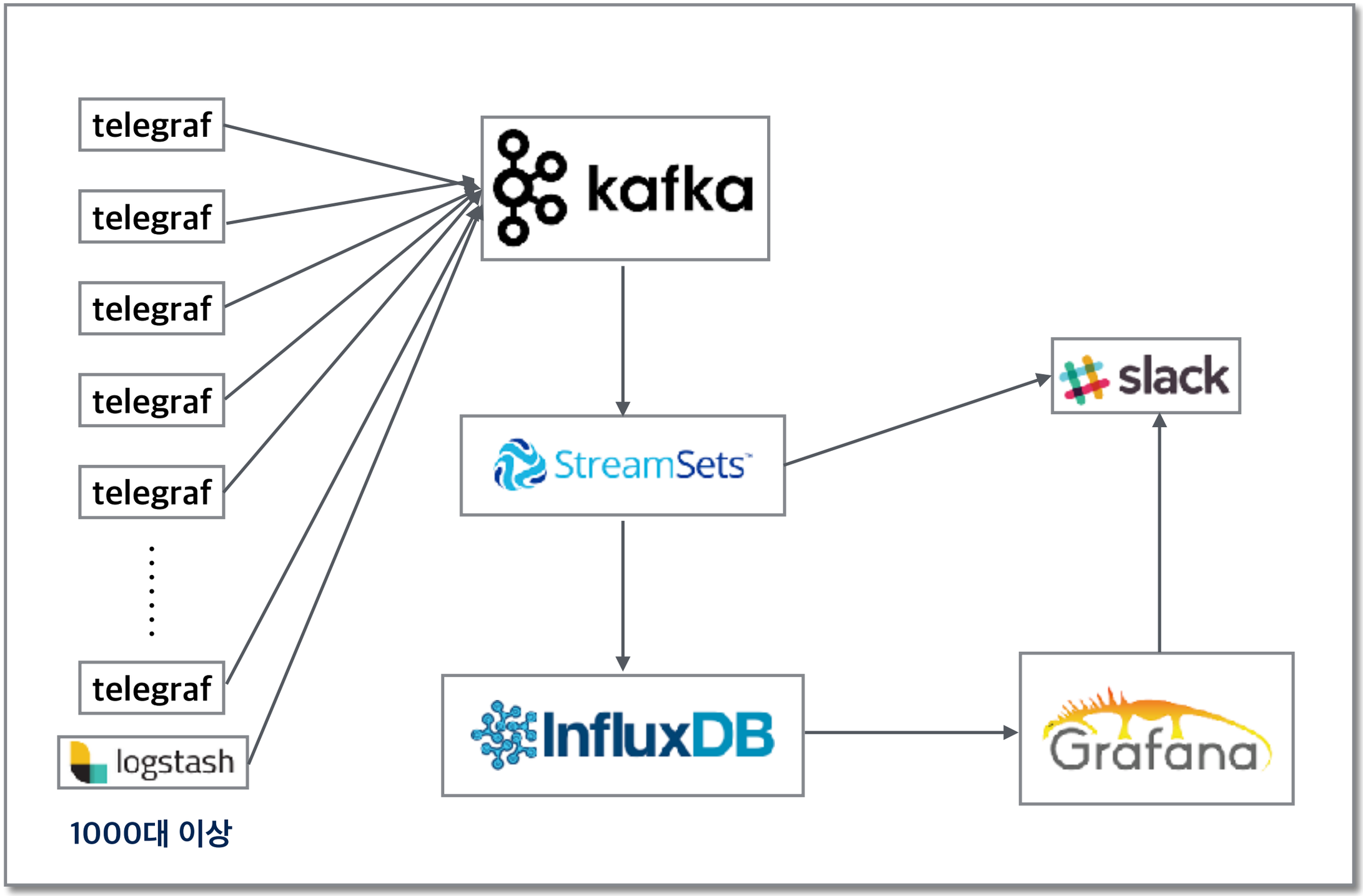
수집 서버가 다운될 경우 해당 시간동안 모든 metric 유실



TO-BE

모든 클러스터의 mertric을 카프카로 전송.

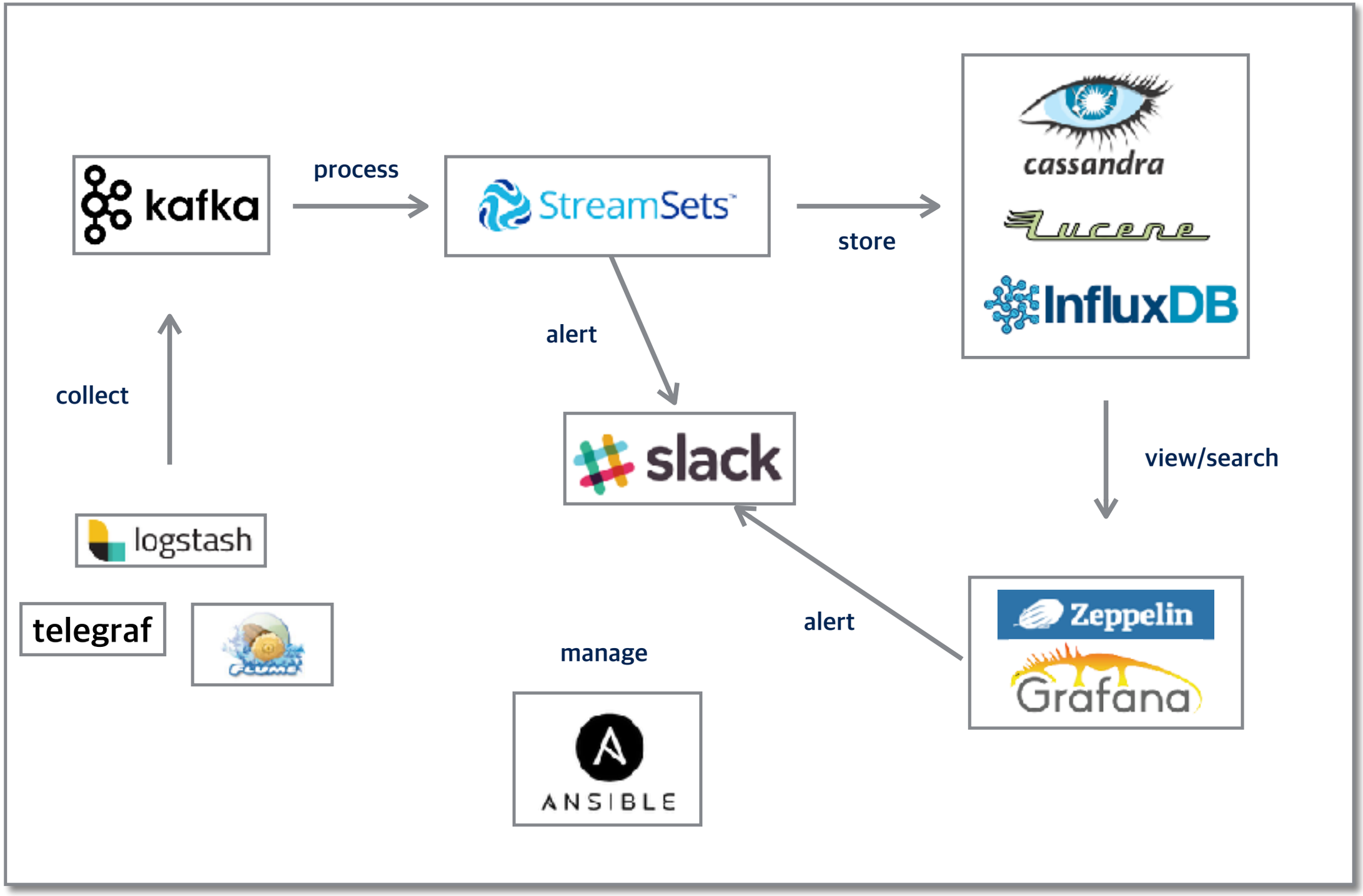
SPOF를 줄이고 필요한 경우 StreamSets을 이용하여 전처리 및 Alert 전송



서비스 구성도

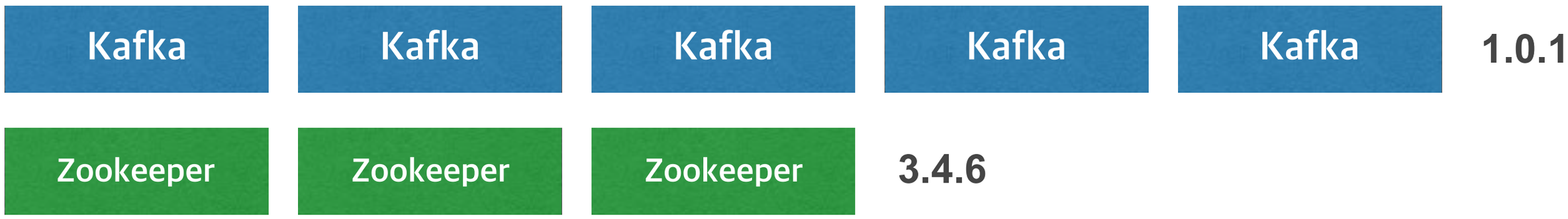
telegraf, logstash, flume을 이용하여 metric 및 로그 kafka 전송

influxDB 및 cassandra에 적재된 데이터를 grafana, zeppelin을 이용하여 시각화



PART 4.
System 구성

Kafka Broker



- Disk 2TB x 2 (총 20TB 구성)
- Topic 1개로 구성
  - cpu, memory, network, disk, hadoop metric, jvm metric, switch metric …
- Replication Factor 3
- Partition 5
- Retention 7days

6TB Storage 사용

850GB per day

35GB per hour

(3 factor)

Kafka Producer(telegraf)

telegraf.conf

```
[[outputs.kafka]]
## URLs of kafka brokers
brokers = ["kafka-01-01:9092","kafka-01-02:9092"]

## Kafka topic for producer messages
topic = "datalake"

## CompressionCodec represents the various compression codecs recognized by
## Kafka in messages.
##  0 : No compression
##  1 : Gzip compression
##  2 : Snappy compression
compression_codec = 0

##  RequiredAcks is used in Produce Requests to tell the broker how many
##  replica acknowledgements it must see before responding
##   0 : the producer never waits for an acknowledgement from the broker.
##       This option provides the lowest latency but the weakest durability
##       guarantees (some data will be lost when a server fails).
##   1 : the producer gets an acknowledgement after the leader replica has
##       received the data. This option provides better durability as the
##       client waits until the server acknowledges the request as successful
##       (only messages that were written to the now-dead leader but not yet
##       replicated will be lost).
##   -1: the producer gets an acknowledgement after all in-sync replicas have
##       received the data. This option provides the best durability, we
##       guarantee that no messages will be lost as long as at least one in
##       sync replica remains.
required_acks = 1

##  The total number of times to retry sending a message
max_retry = 3

## https://github.com/influxdata/telegraf/blob/master/docs/DATA_FORMATS_OUTPUT.md
data_format = "influx"
```

Kafka Producer(Logstash)

switch_metric.conf

```
event['now'] = Time.now.to_i * 1000000000;

output {
    kafka {
        topic_id => "datalake"
        bootstrap_servers => "kafka-01-01:9092,kafka-01-02:9092"
        acks => "1"
        codec => plain {
            format => “switch,inout=%{inout} 1001=%{1001},2001=%{2001} %{now}”
        }
    }
}
```

influx 포맷

```
switch,inout=in 1001=0i,2001=0i 1525417999000000000
switch,inout=out 1001=8848634164655i,2001=837385484971i 1525417999000000000
switch,inout=in 1001=8625697583336i,2001=9182721564736i 1525417999000000000
switch,inout=out 1001=0i,2001=0i 1525417999000000000
```

Kafka Consumer(StreamSets)

Kafka Consume => Record Merge => InfluxDB Insert 의 과정으로 수행된다.

(Record Merge는 1000개 Row 기준)

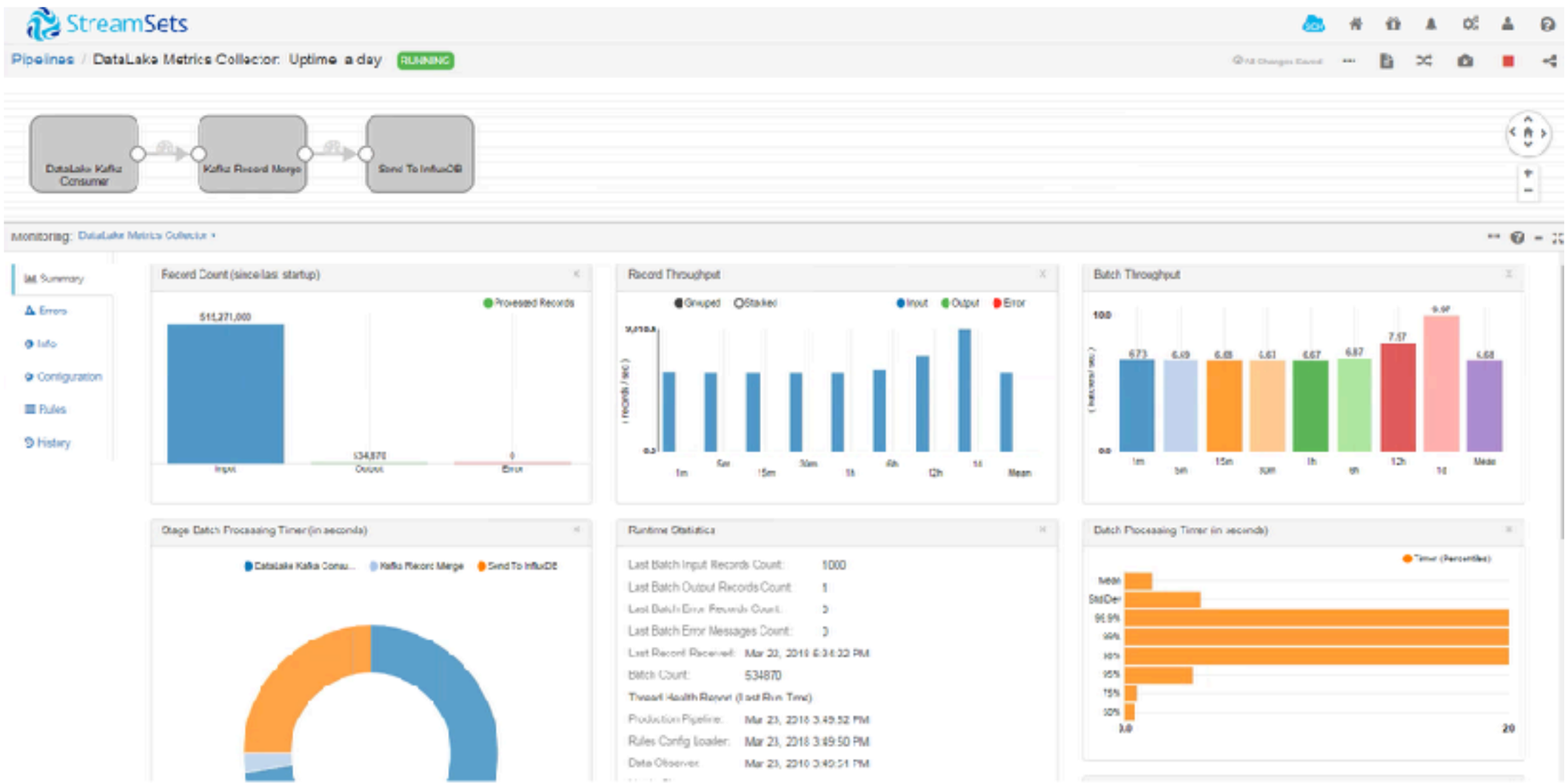
Configuration

Broker URI : kafka-01-01:9092,kafka-01-02:9092

ZooKeeper URI : kafka-01-03:2181

Consumer Group : datalakeConsumerGroup

Topic : datalake



Kafka Management(Ansible)

ansible을 이용하여 kafka 기동 스크립트 작성

start, status, stop 스크립트

1.start_kafka_nodes.yml

```
- name: kafka broker nodes
  hosts: kafka_nodes

  tasks:
  - name: kafka node start
    command: su kafka -c "/data/kafka/bin/kafka-server-start.sh -daemon /data/kafka/config/server.properties”
```

2.status_kafka_nodes.yml

```
- name: kafka broker nodes
  hosts: kafka_nodes

  tasks:
  - name: kafka node status
    command: /data/kafka/bin/kafka-server-status.sh
    register: status
  - debug: msg="{{ status.stdout }}”
```

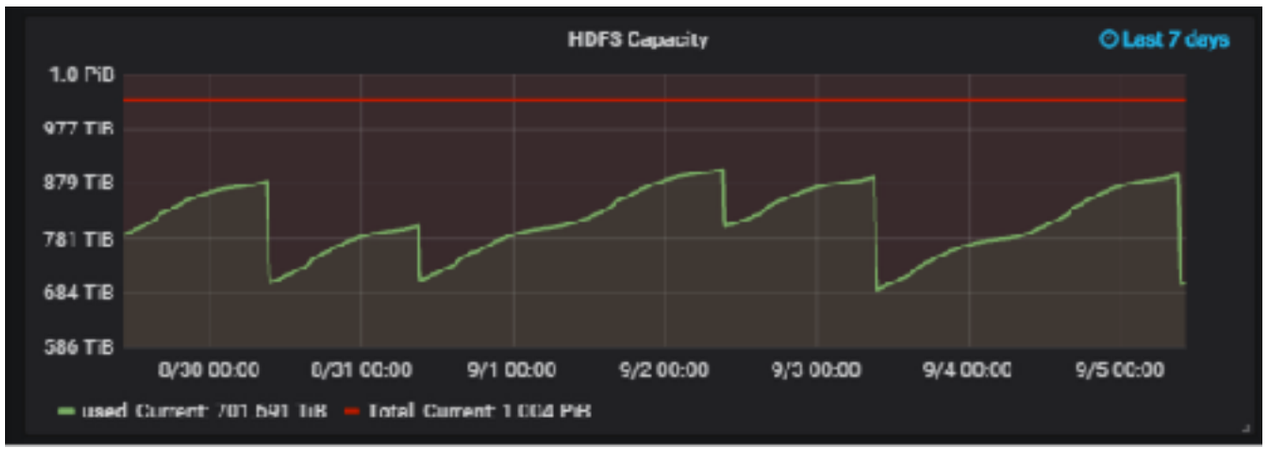
3.stop_kafka_nodes.yml

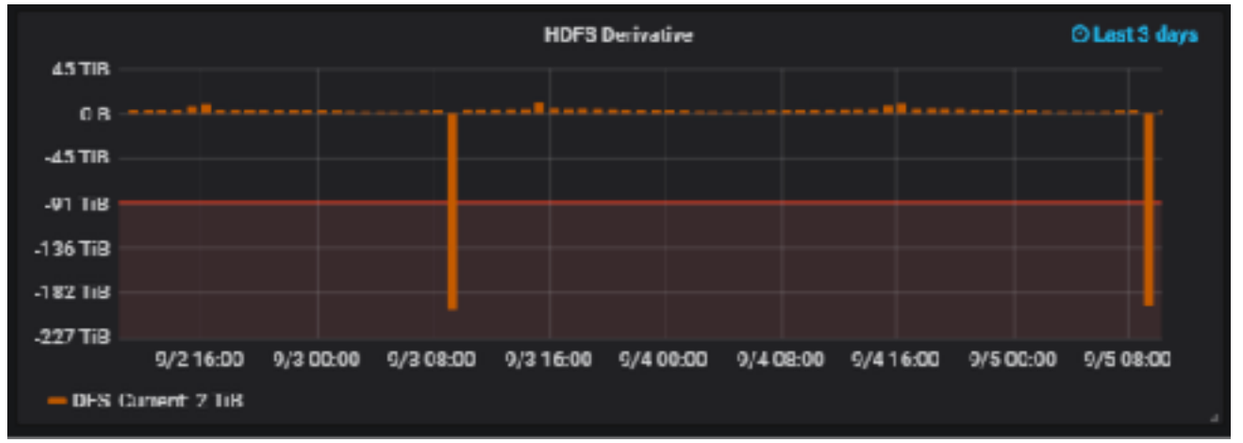
```
- name: kafka broker nodes
  hosts: kafka_nodes

  tasks:
  - name: kafka node stop
    command: /data/kafka/bin/kafka-server-stop.sh
```

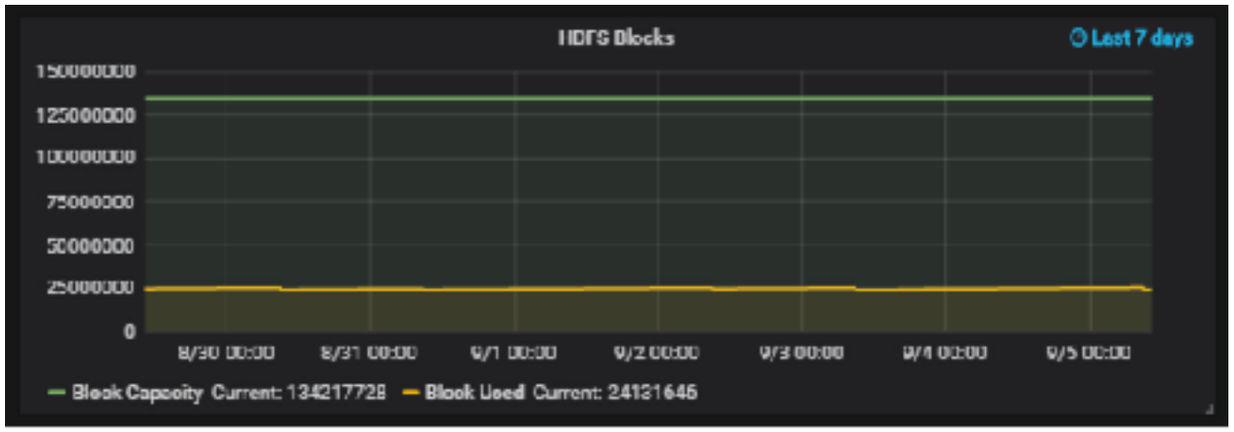
PART 5.
Dashborad

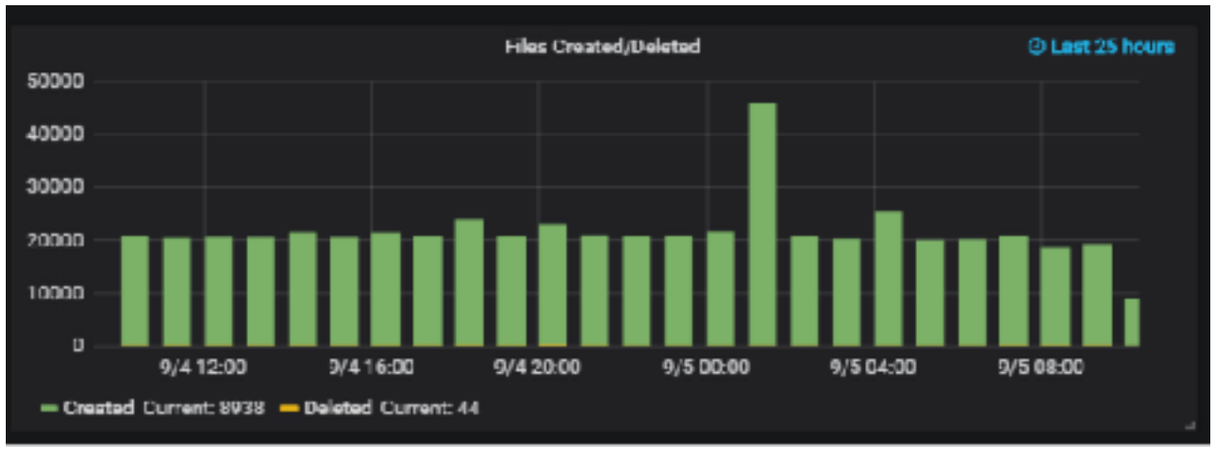
Grafana

Hadoop 메트릭



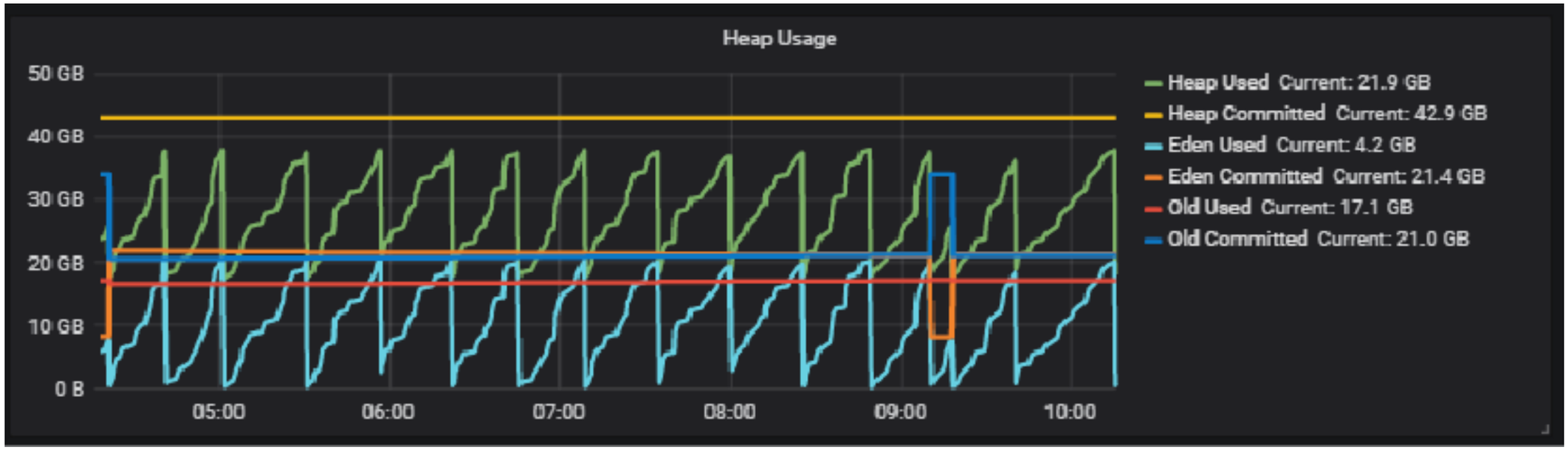


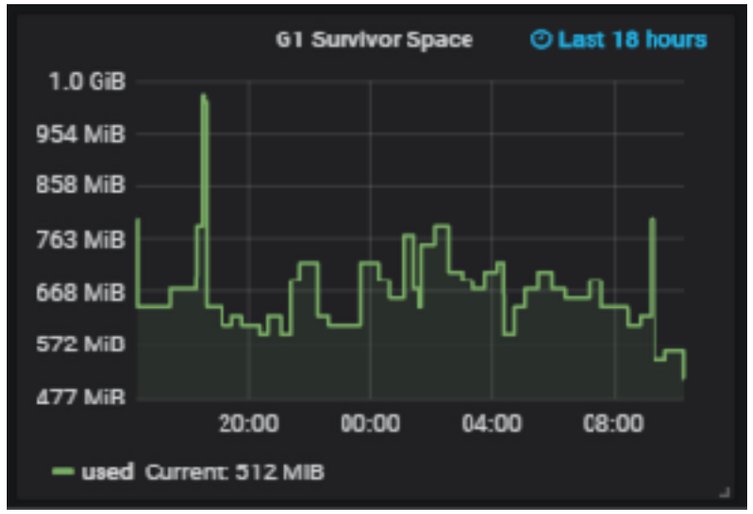


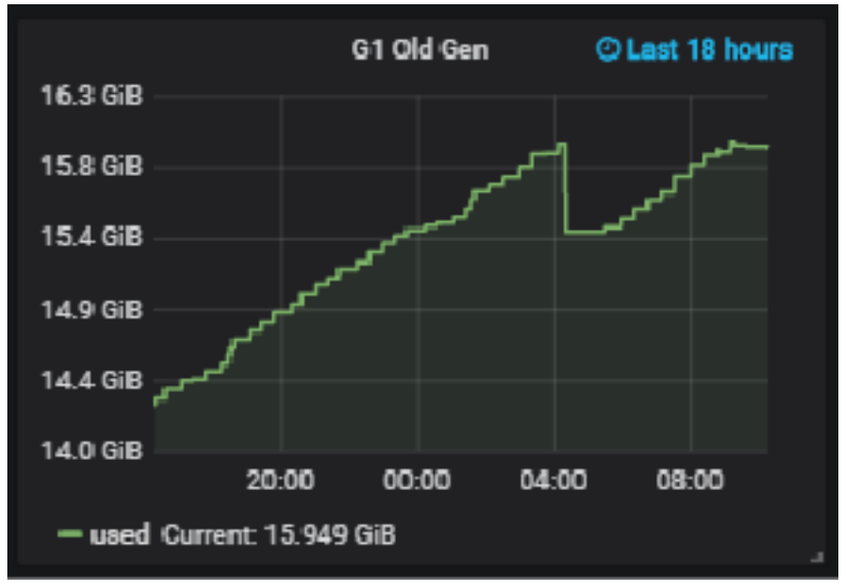


Grafana

JVM 메트릭







PART 6.

Future work

그런데 카프카 모니터링은?

실시간 처리에서의 카프카는?

Thank You!!!

Q & A