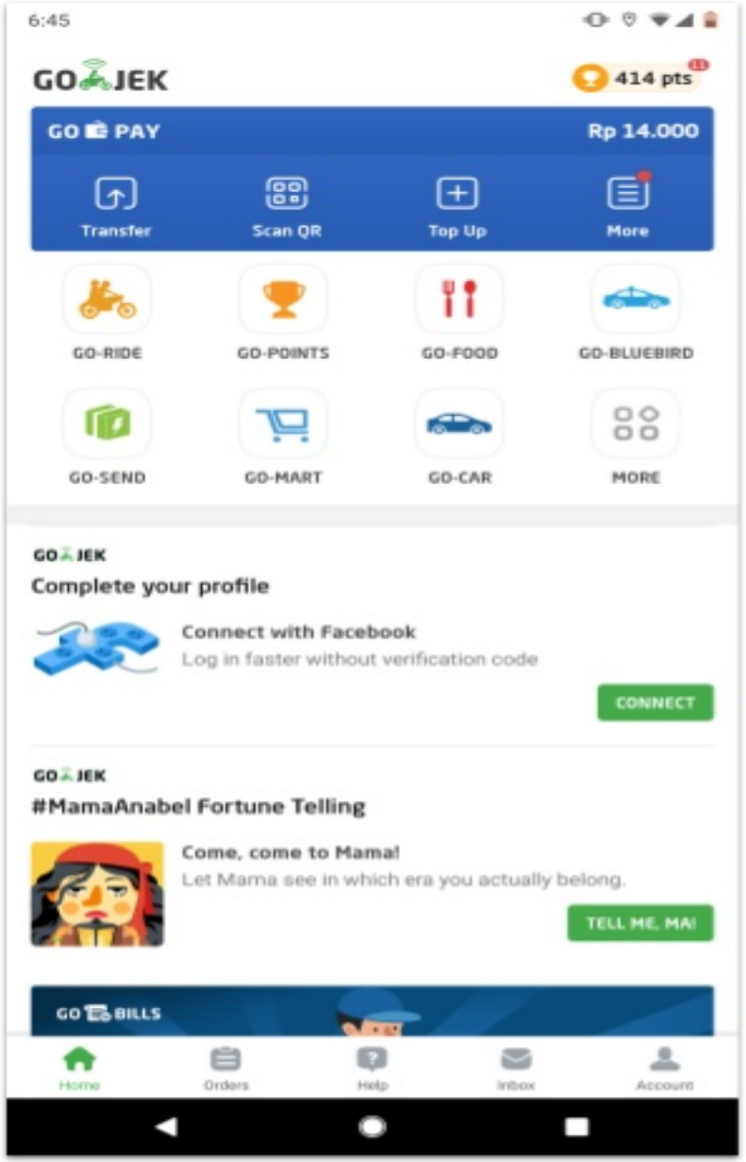
GO JEK

Managing Flink operations at GO-JEK

Ravi Suhag | Sumanth K N



GO-JEK?

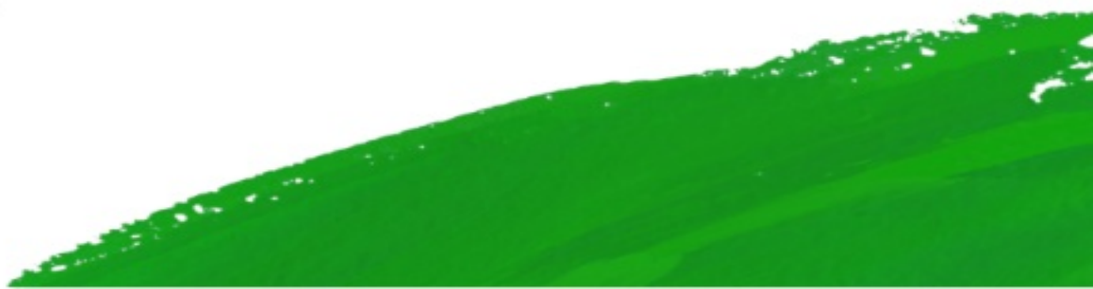
ONE APP FOR ALL YOUR NEEDS

Established in 2010 as a motorcycle ride-hailing phone-based service, GO-JEK has evolved into an on-demand provider of transport and other lifestyle services.

GO JEK

Agenda

- Resource Provisioning
- Resource Isolation
- Data Quality
- Monitoring
- Cluster Failovers
- Chaos and Load Testing



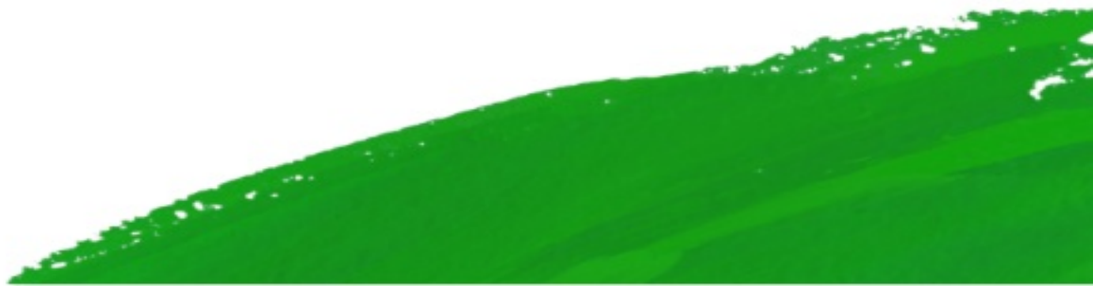
GO JEK

As data engineers, we look out for patterns in data usages and transformation and build tools, infrastructure, frameworks, and services.

SCALE

AUTOMATION

PRODUCT MINDSET



GO-JEK

At GO-JEK, location is built into the fabric of all our products

2B +

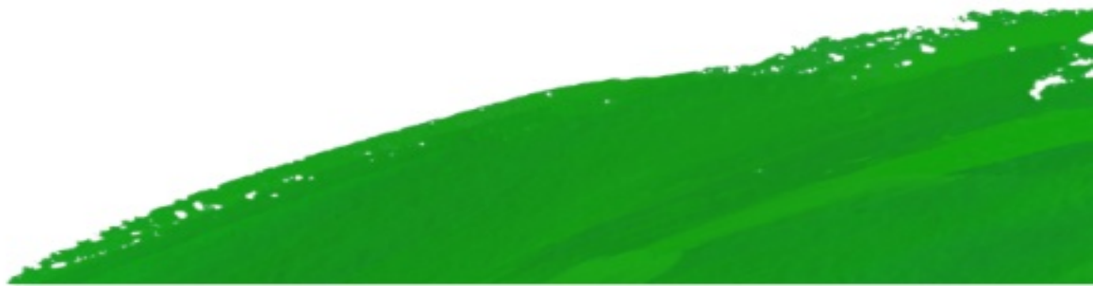
GPS points per day

25M +

Booking events per day

3B +

API log events per day



GO JEK

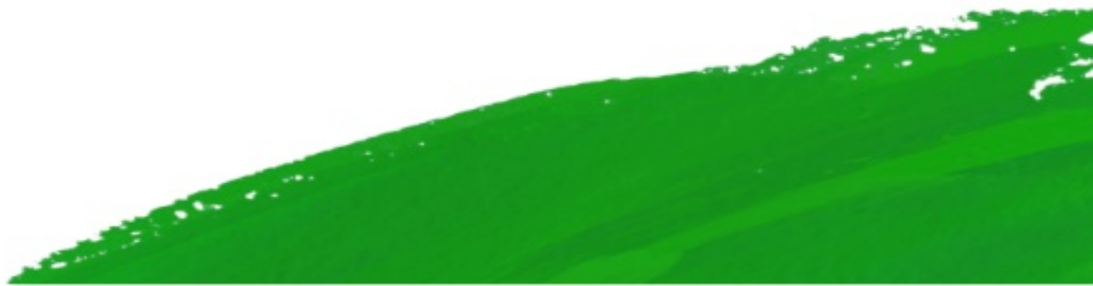
Flink use cases at GO-JEK

Surge Pricing

API Health monitoring

Driver allocation monitoring

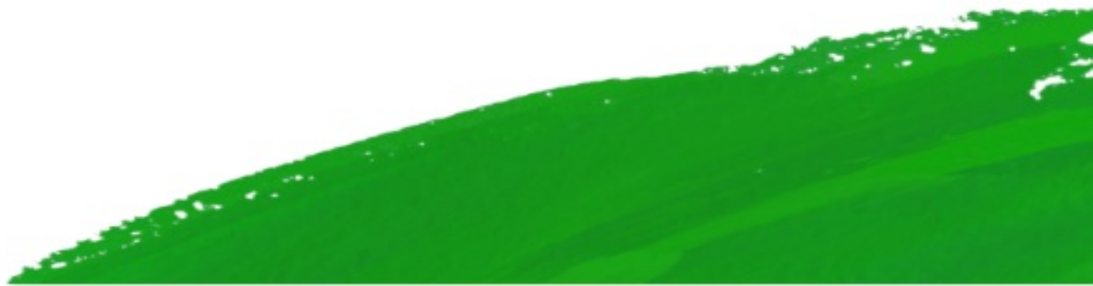
Fraud detection and more ...



GO JEK

Resource Provisioning

How do we create new clusters with increasing number of requests for more than 15 internal teams?



GO JEK

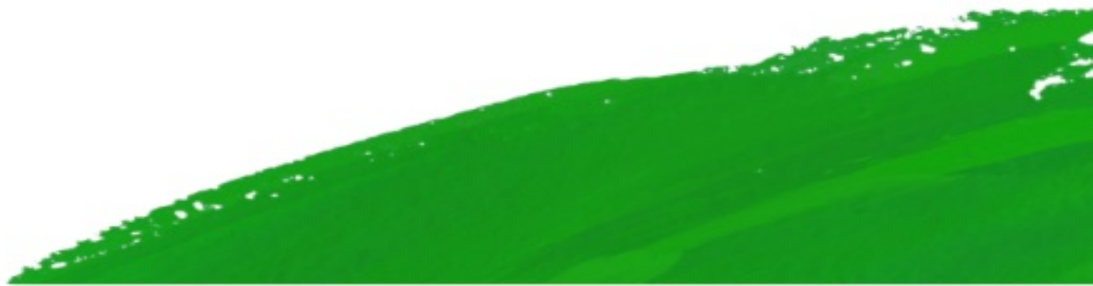
Challenges

Multiple cloud providers and DCs

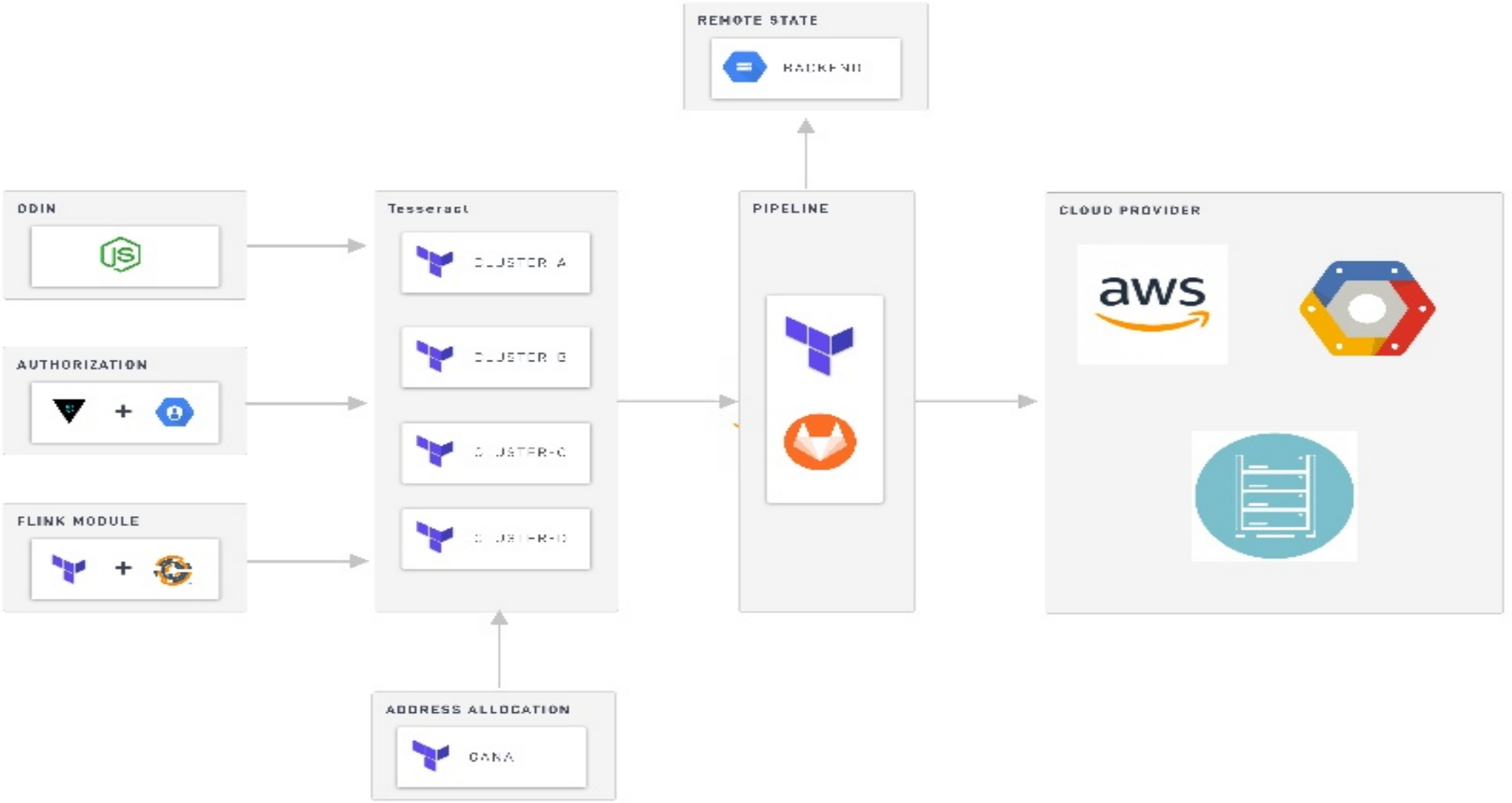
Frequent need for cluster provisioning

Man-hour intensive and repetitive

Error-prone process



Architecture



Example

```
module "p_de_daggers_flink_playground" {
  source        = "../../../modules/flink/resource"
  cluster_name  = "p-de-daggers-flink-playground"
  chef_role     = "de_daggers_flink_playground"
  master_nodes  = 3
  worker_nodes  = 6
  chef_server   = "${var.chef_server}"
  ssh_user      = "${var.ssh_user}"
  ssh_key       = "${var.ssh_key}"
  subnet        = "${var.subnet}"
  zone          = "asia-east1-b"
}
```

GO JEK

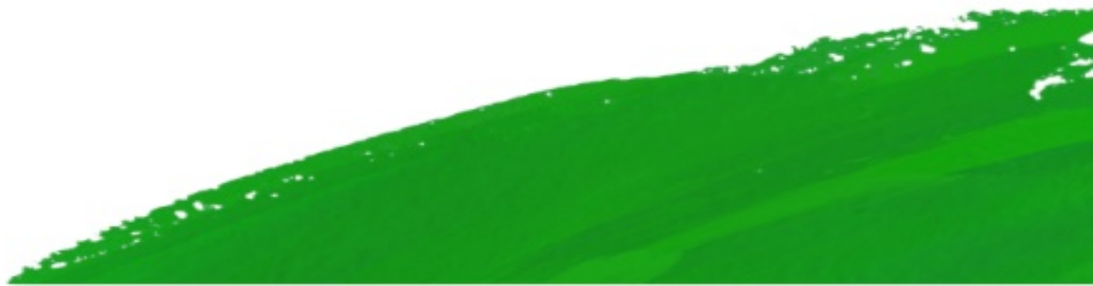
Impact

Provisioning time reduced by 90%

On the fly infra for load testing

Infrastructure As Code

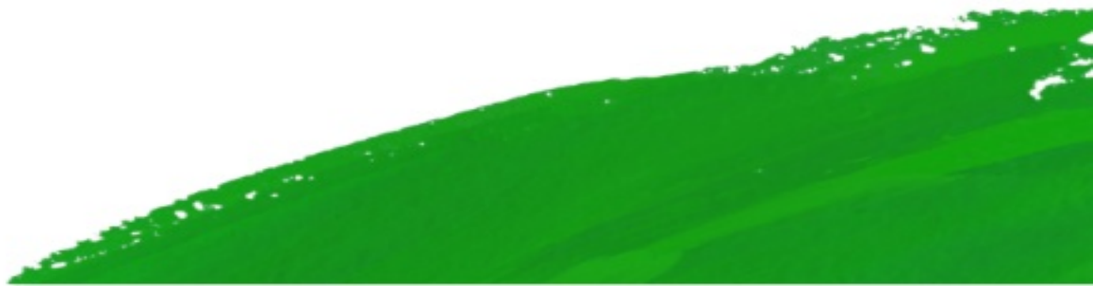
Self-serve and no workflows



GO JEK

Resource Isolation

How do we isolate resources for security, resilience, and segregation for more than 15 internal teams?



GO JEK

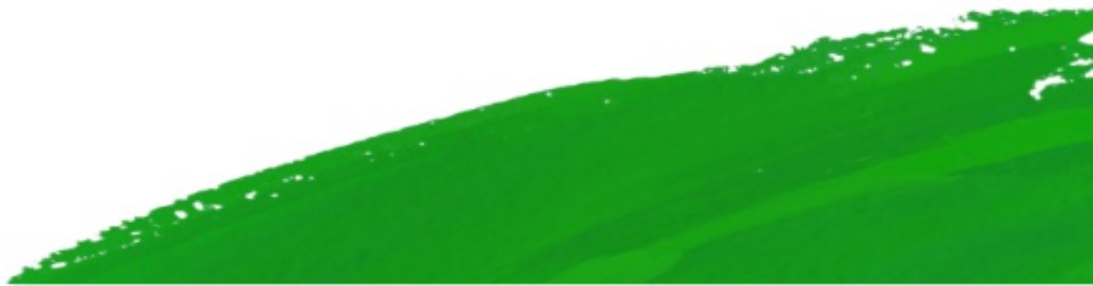
Why

Critical kafka topics handicapped by low priority scripts

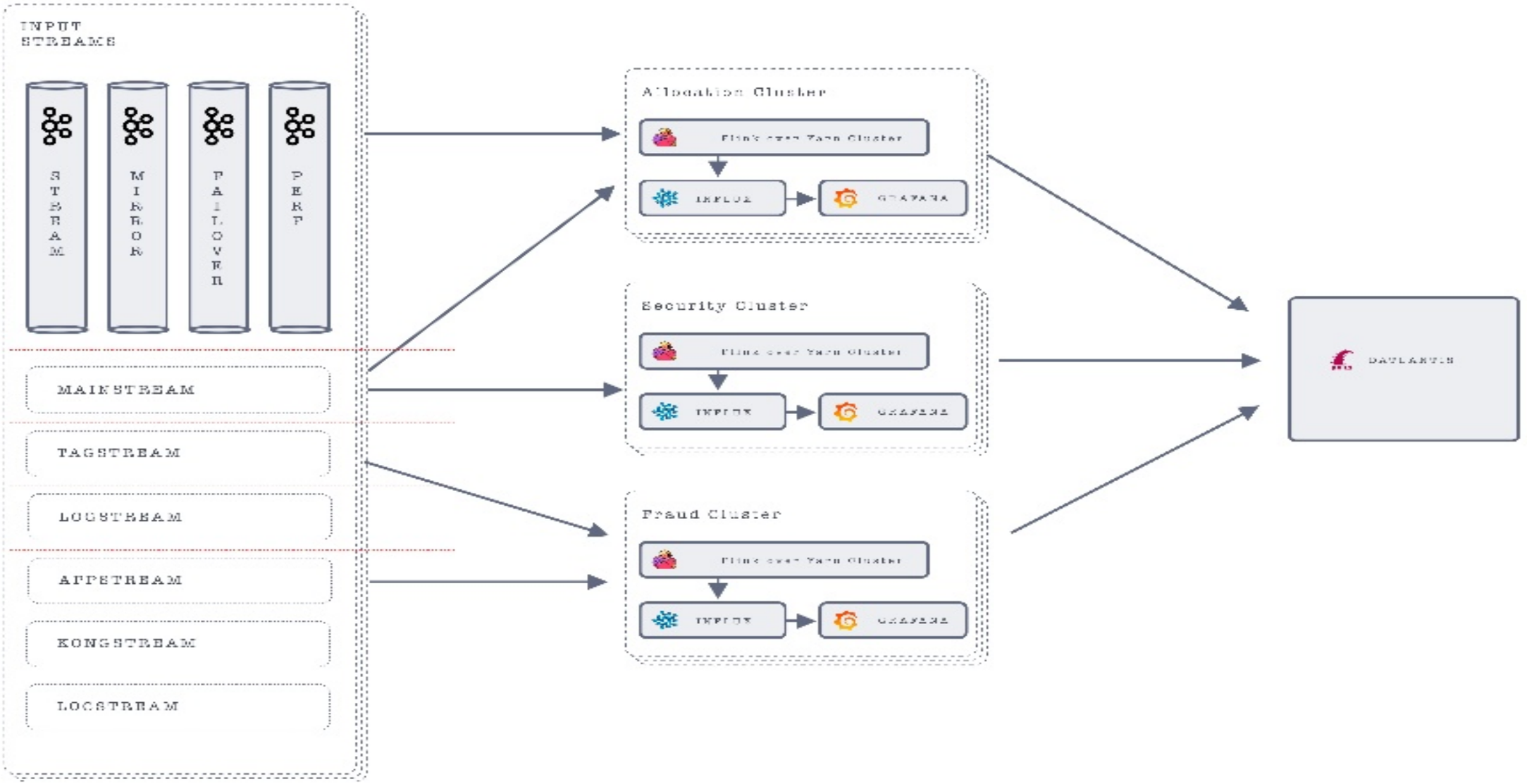
Non-critical jobs wipe out resources of critical jobs

Extensive downtime due to issues in Kafka

Human errors



Isolation Architecture



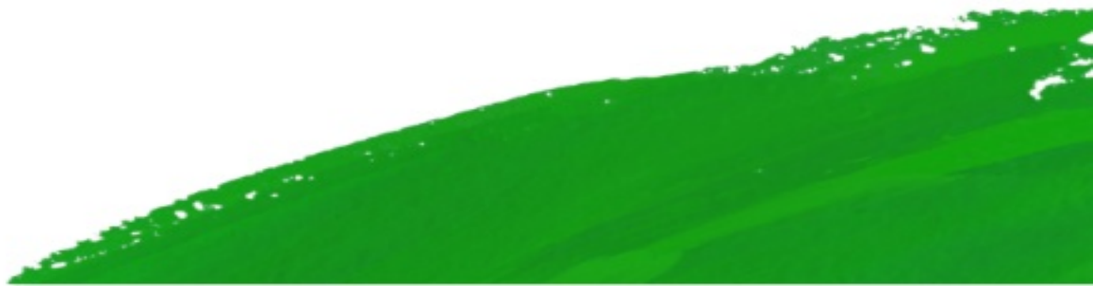
GO JEK

Kafka input stream

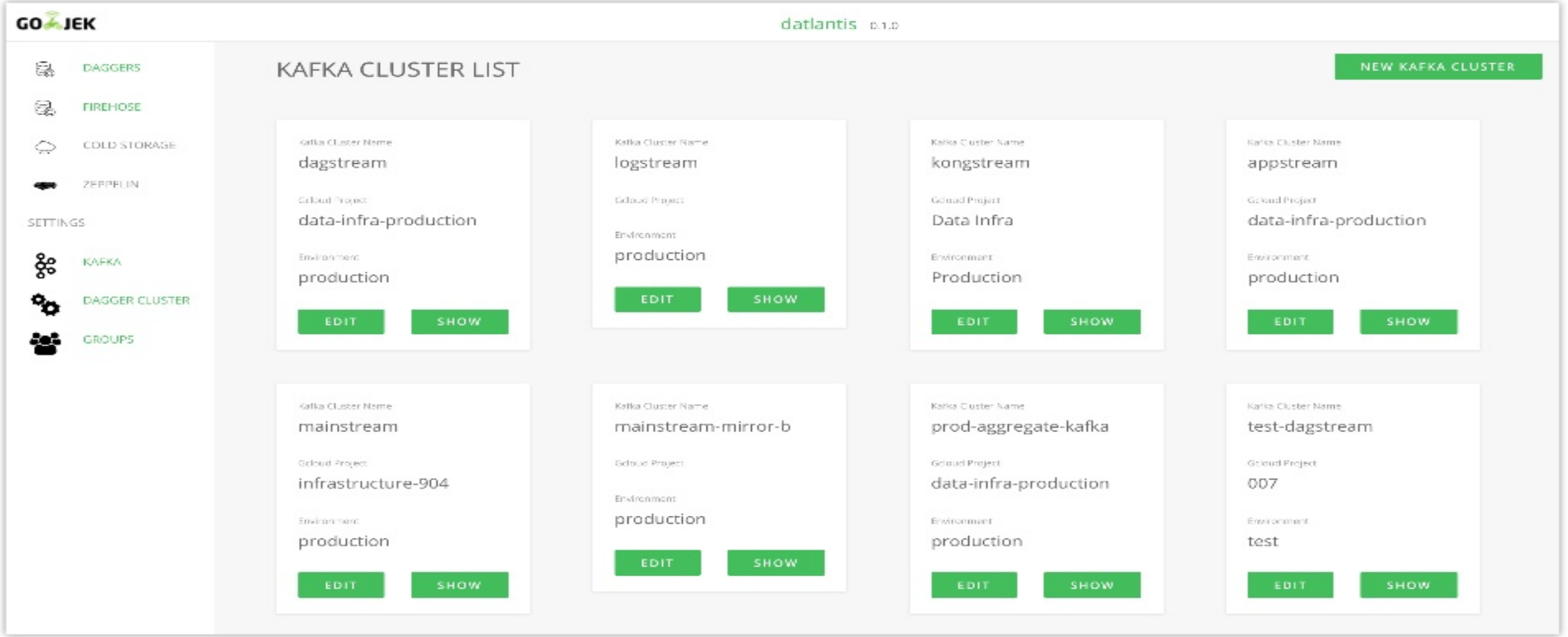
Nature, time and transactional criticality, sensitivity and volume of data.

Flink clusters

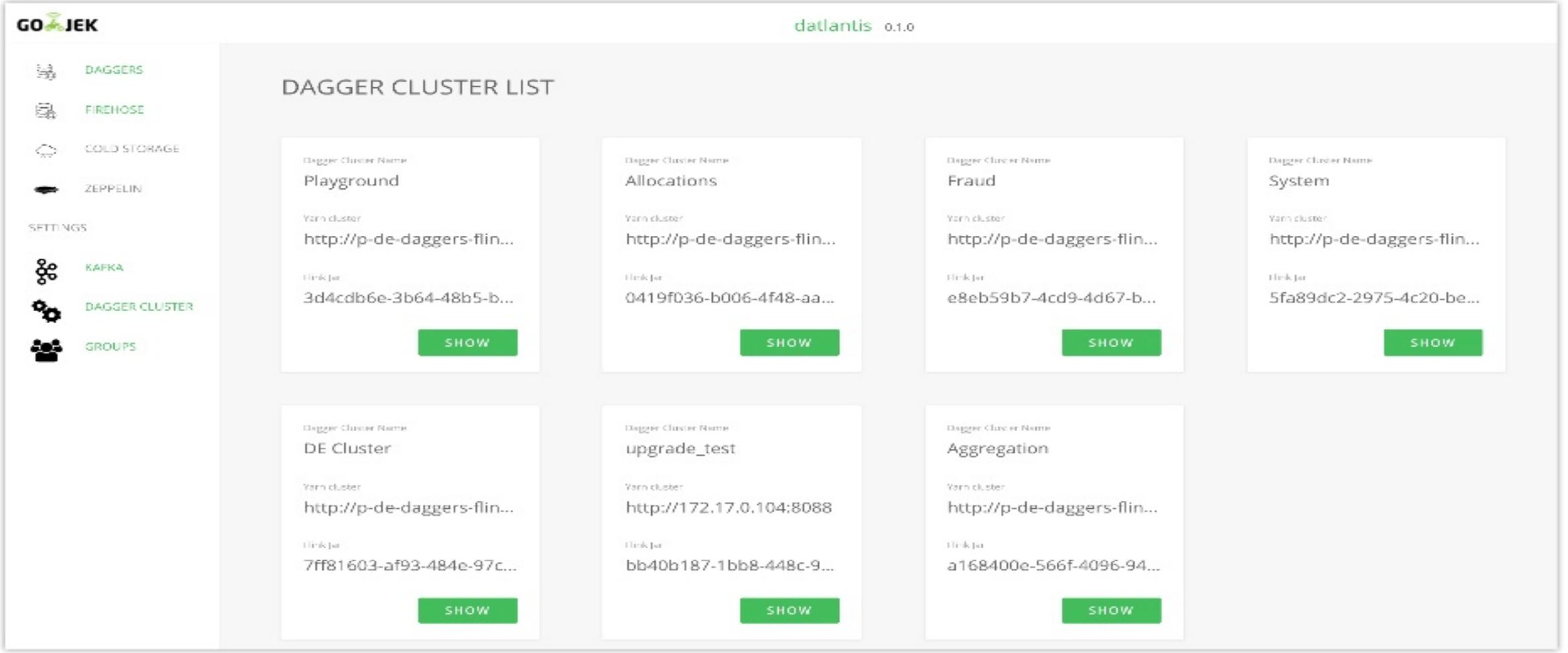
Security concerns, team segregation, job loads and criticality which comes at the cost of handling large volume data replication and maintenance.



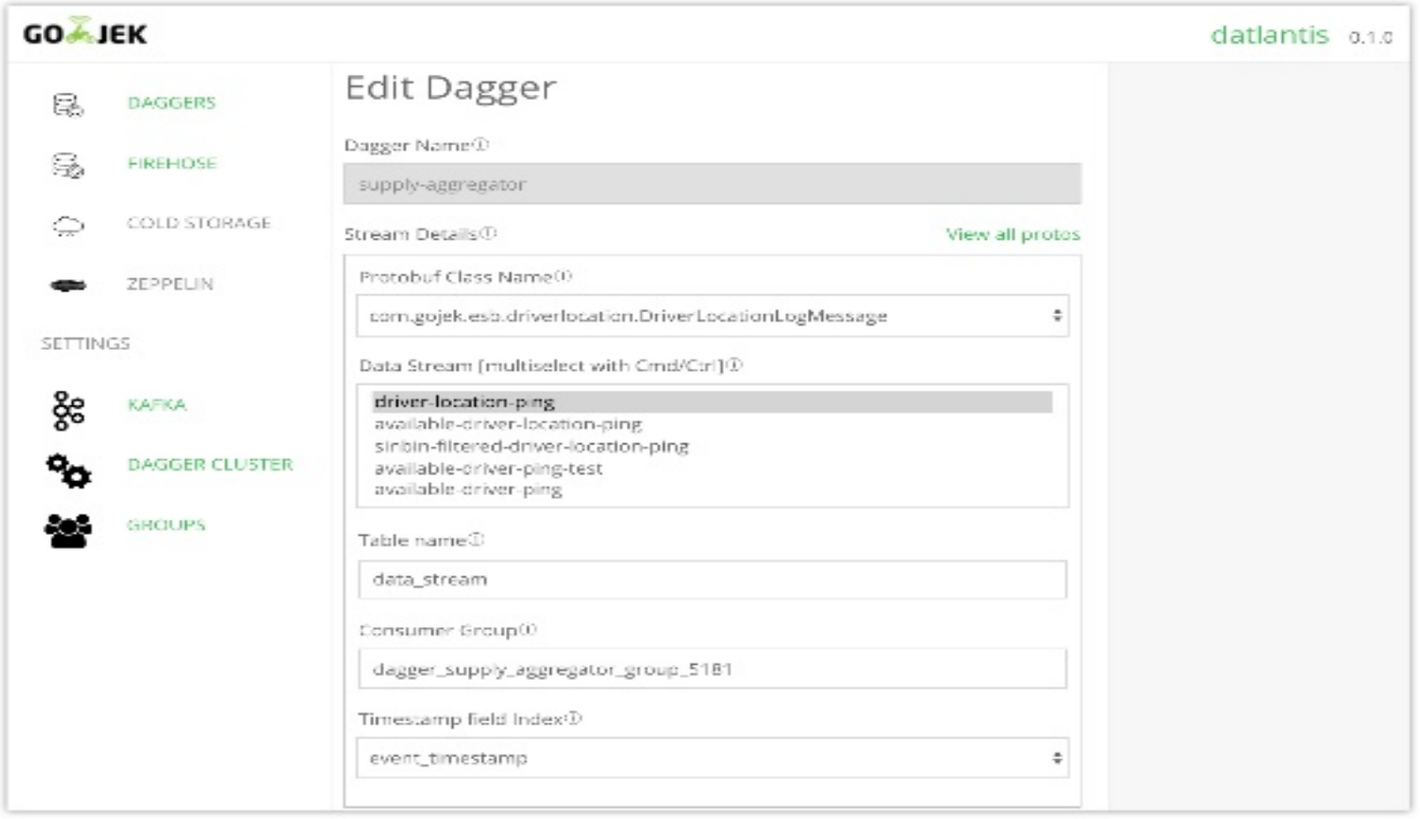
Kafka Clusters Management

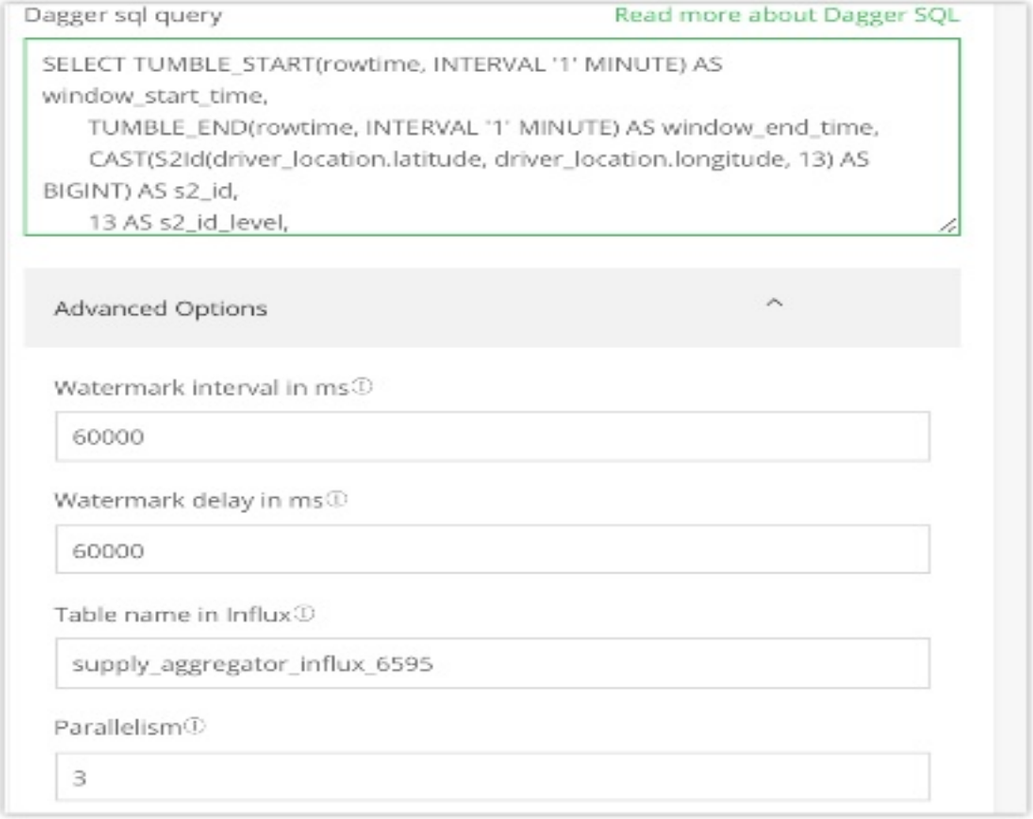


Flink Clusters Management



Job Management

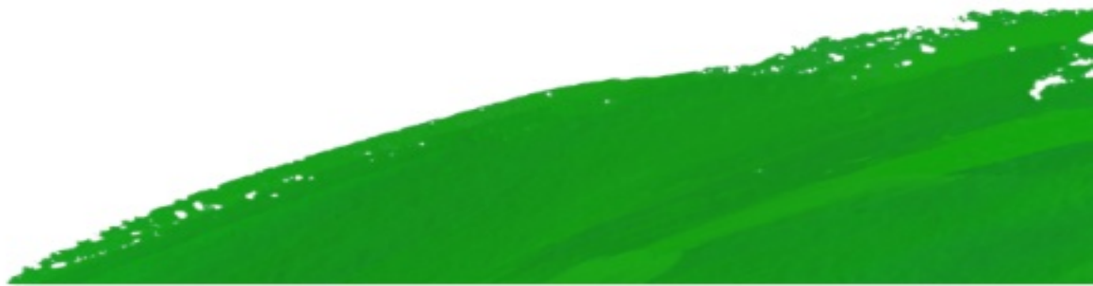




GO JEK

Data Quality

How do we manage quality of data for aggregation with more than 15 teams being responsible for producing data?



GO JEK

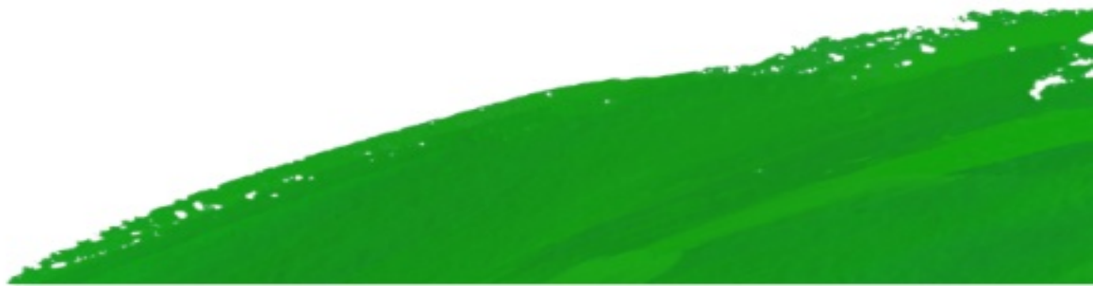
Challenges

JSON weekly typed

Errors during schema update

Bad deployment sequence of an incompatible schema change

No data for real-time actors



GO JEK

Measures

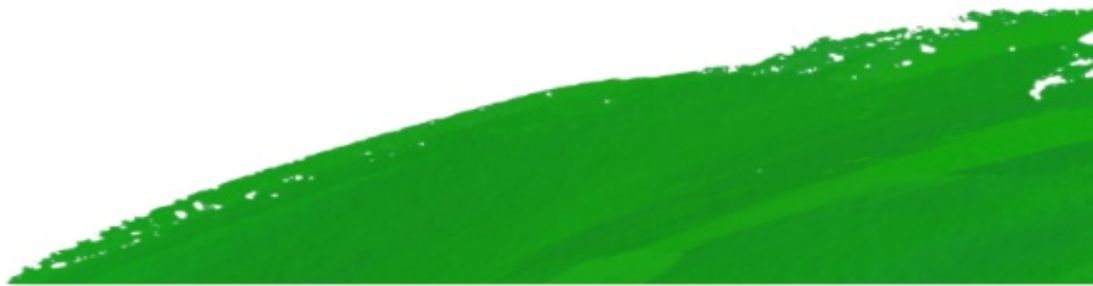
Protobuf - Strongly typed

Maintains version of the schema

Version locking per job

Allows to re-run a job with different schema

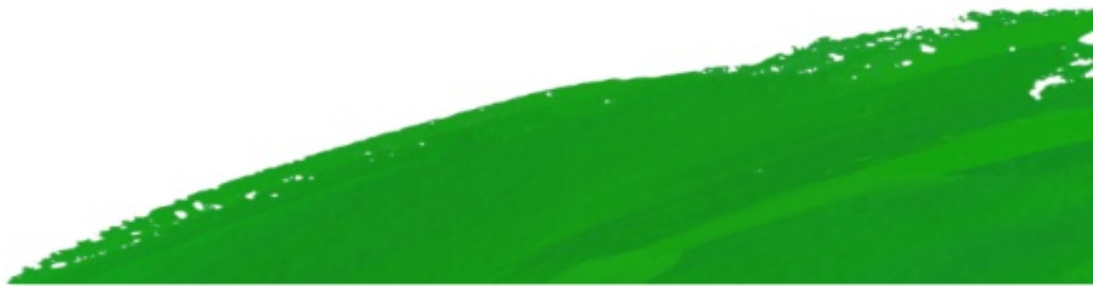
Upstream actors to have an `inadequate data points` behavior



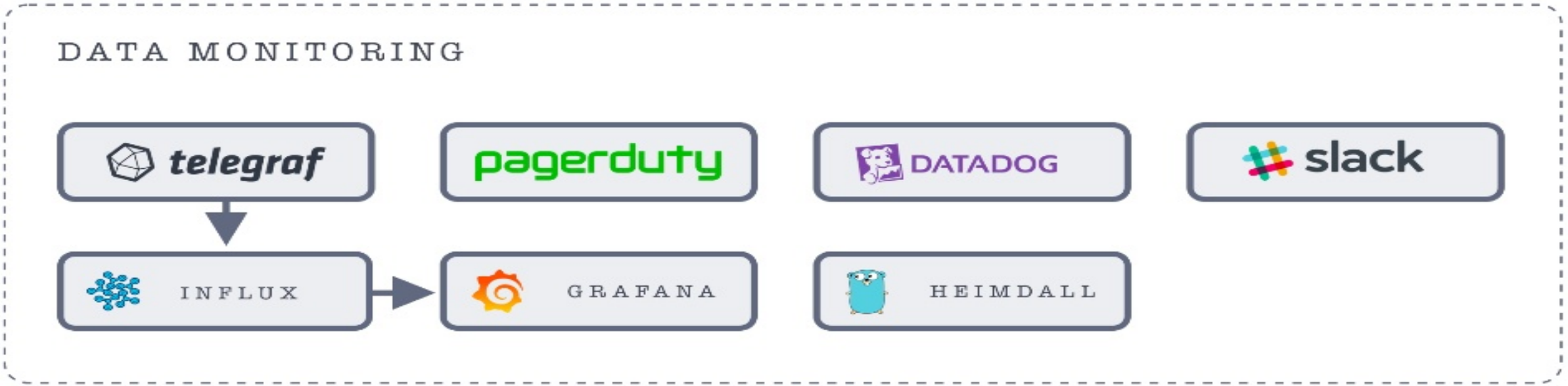
GO JEK

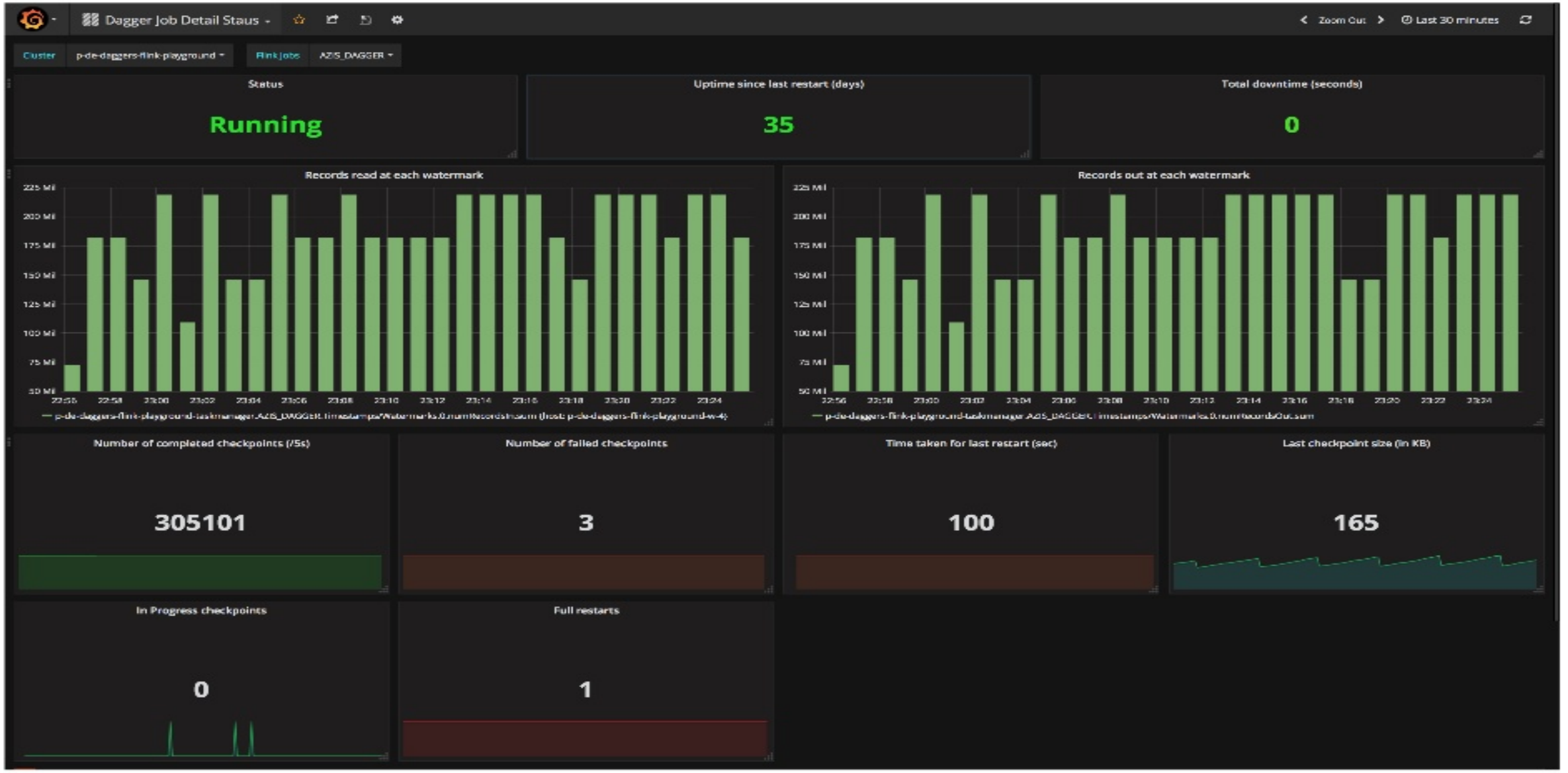
Monitoring

How do we manage monitoring for multiple clusters across data-centers?

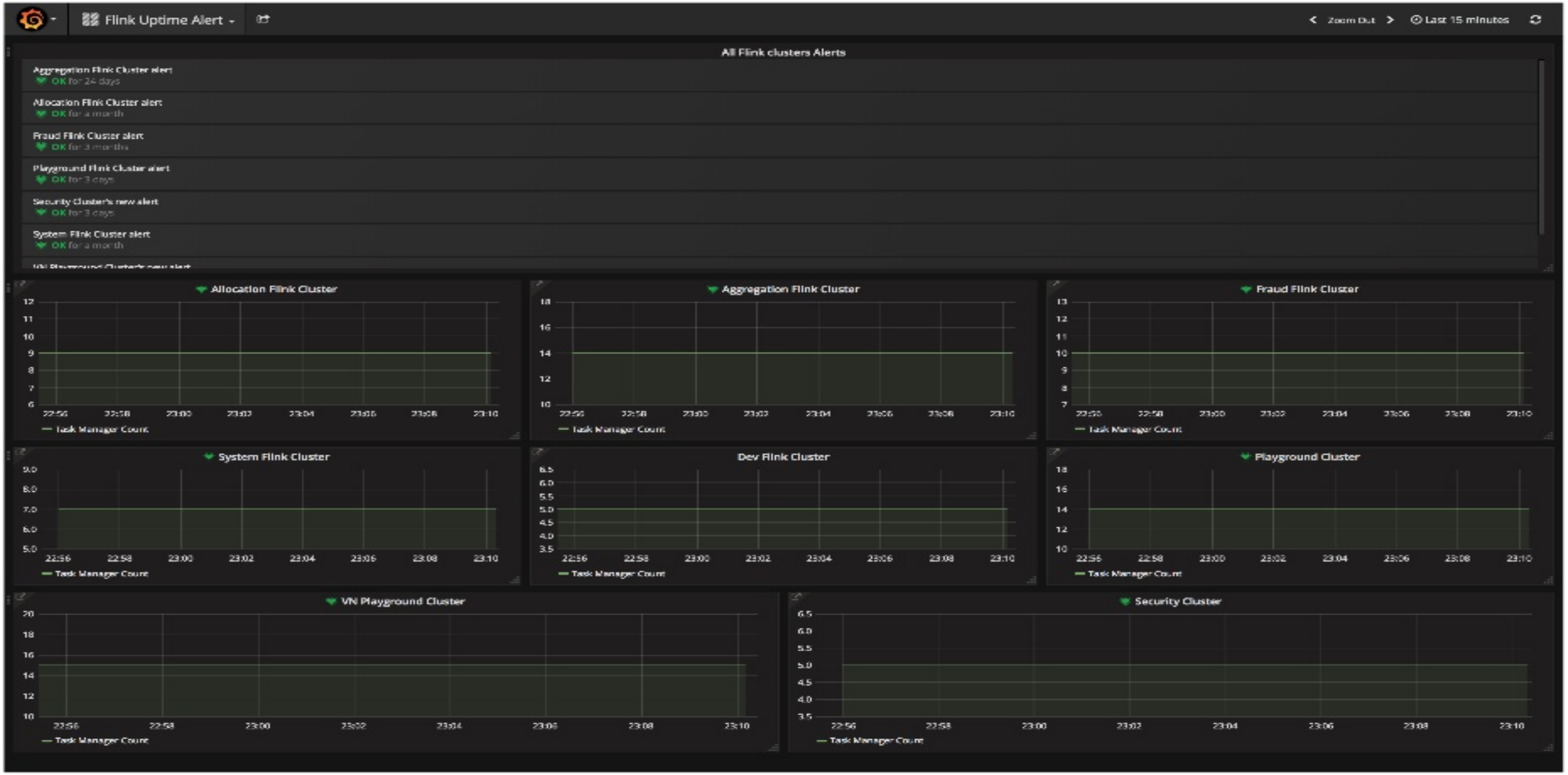


GO JEK


DATA MONITORING
telegraf
pagerduty
DATADOG
slack
INFLUX
GRAFANA
HEIMDALL


Dagger Job Detail Staus
Zoom Out
Last 30 minutes
Cluster
p-de-daggers-flink-playground
Flink Jobs
AZIS_DAGGER
Status
Running
Uptime since last restart (days)
35
Total downtime (seconds)
0
Records read at each watermark
Records out at each watermark
Number of completed checkpoints (/5s)
305101
Number of failed checkpoints
3
Time taken for last restart (sec)
100
Last checkpoint size (in KB)
165
In Progress checkpoints
0
Full restarts
1

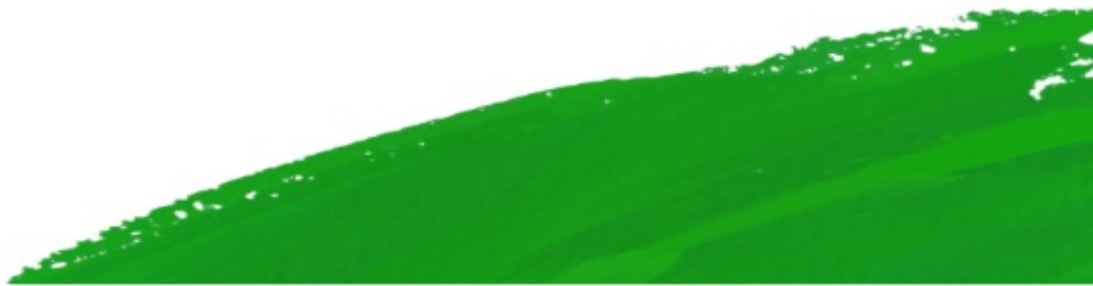

Flink Machines Details
Zoom Out
Last 24 hours
Cluster
p-de-daggers-flink-dev
Overall
Task Managers
5
Jobs Running
1
Total Task Slots
15
Task Slots Available
11
Job Manager
Max heap memory (GB)
3
Heap memory used (GB)
0.46
CPU Load
0.003
Thread count
56
Task Manager
Heap Memory max
Heap Memory used
CPU Load
Thread count
p-de-daggers-flink-dev-w-1
p-de-daggers-flink-dev-w-4
p-de-daggers-flink-dev-w-3
p-de-daggers-flink-dev-w-2


Flink Uptime Alert
Zoom Out
Last 15 minutes
All Flink clusters Alerts
Aggregation Flink Cluster alert
OK for 24 days
Allocation Flink Cluster alert
OK for a month
Fraud Flink Cluster alert
OK for 3 months
Playground Flink Cluster alert
OK for 3 days
Security Cluster's new alert
OK for 3 days
System Flink Cluster alert
OK for a month
Allocation Flink Cluster
Aggregation Flink Cluster
Fraud Flink Cluster
System Flink Cluster
Dev Flink Cluster
Playground Cluster
VN Playground Cluster
Security Cluster
Task Manager Count

GO JEK

Failovers

How do we manage kafka input stream failover and flink cluster failover for multiple clusters?



GO JEK

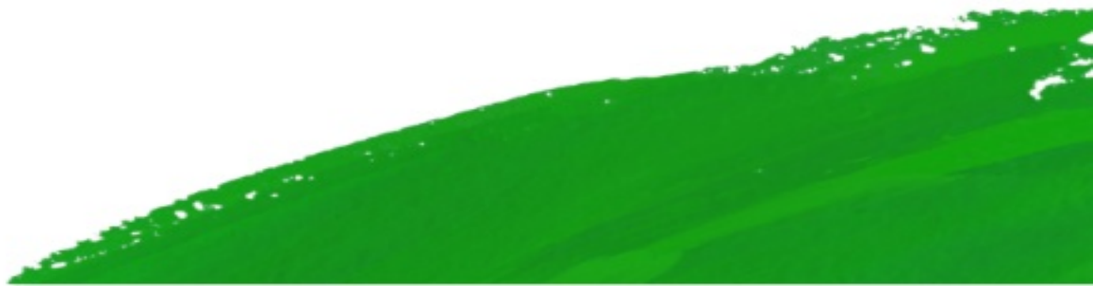
Why need failover?

Kafka cluster down

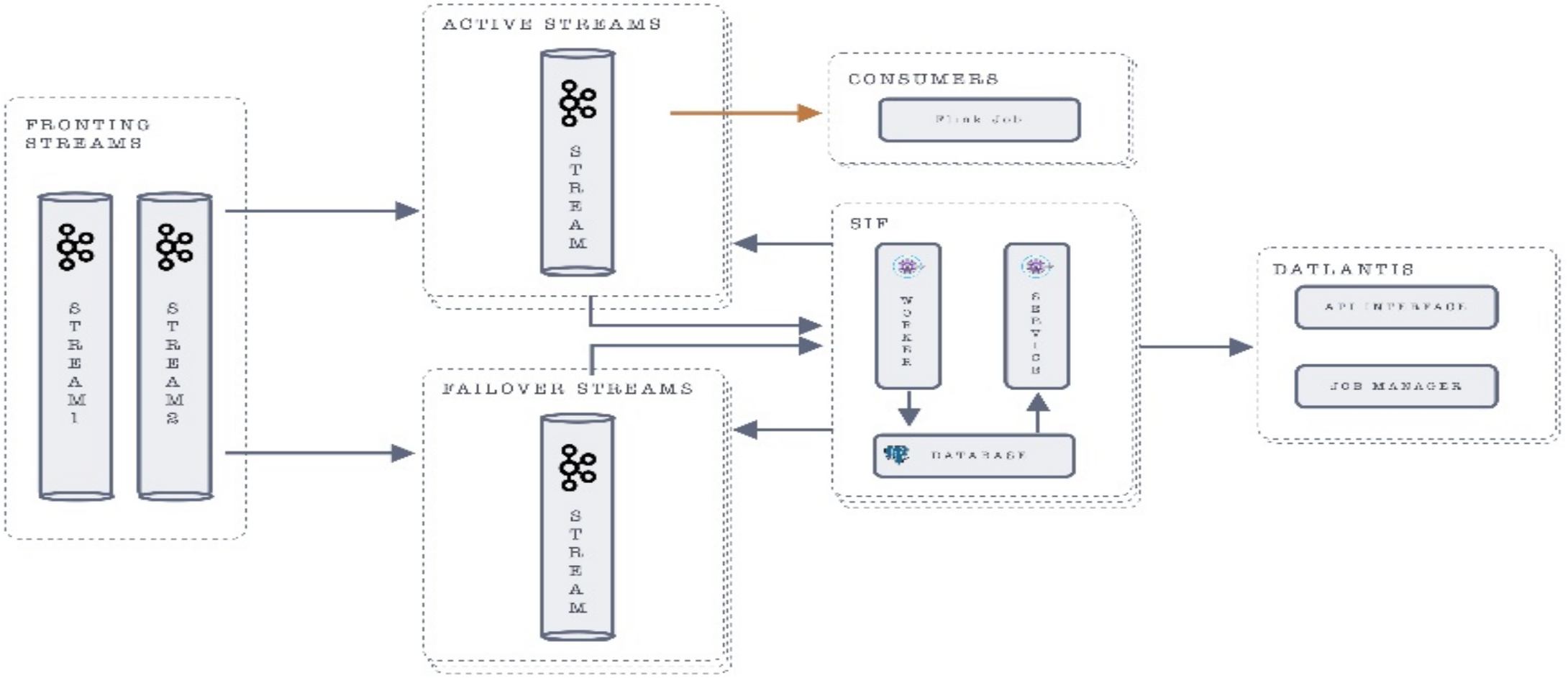
Flink clusters down or in bad state

Kafka or Flink cluster migrate

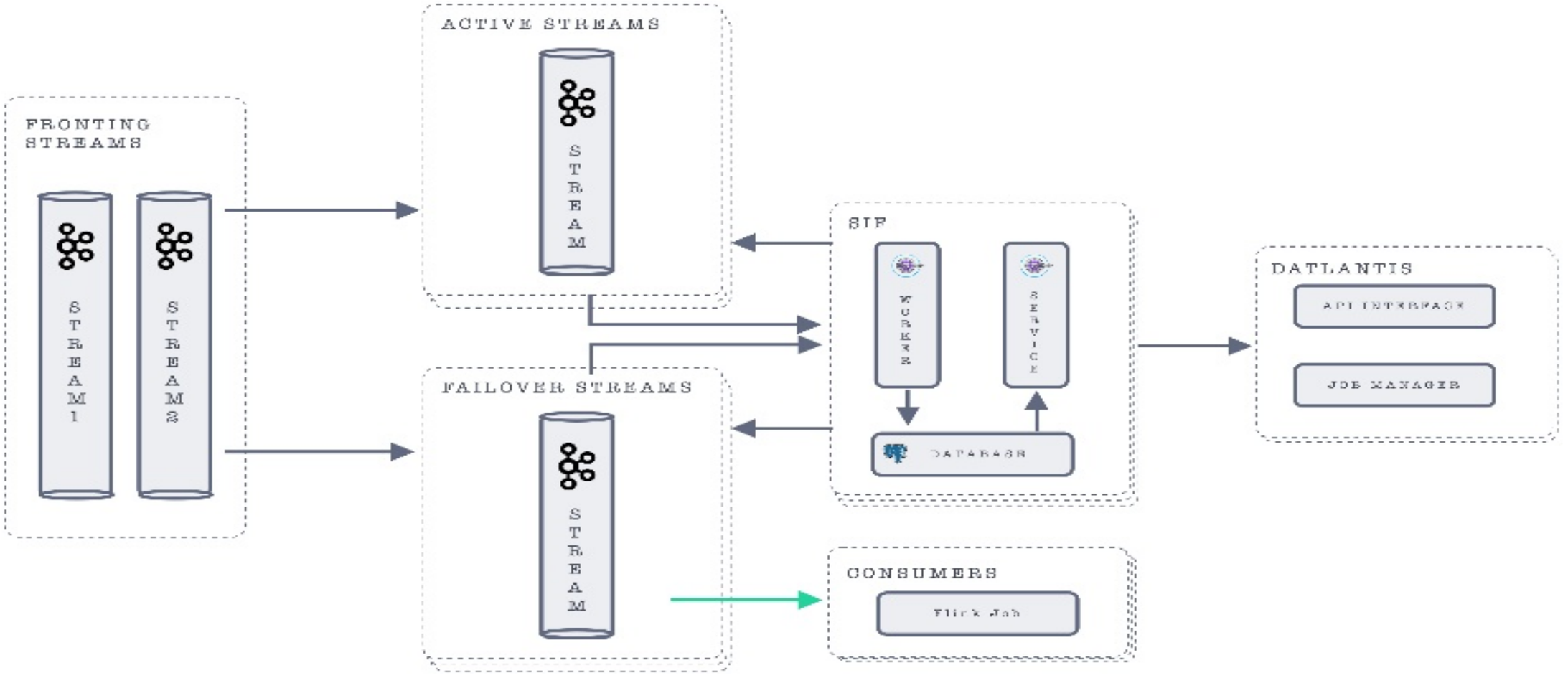
Job migration for teams



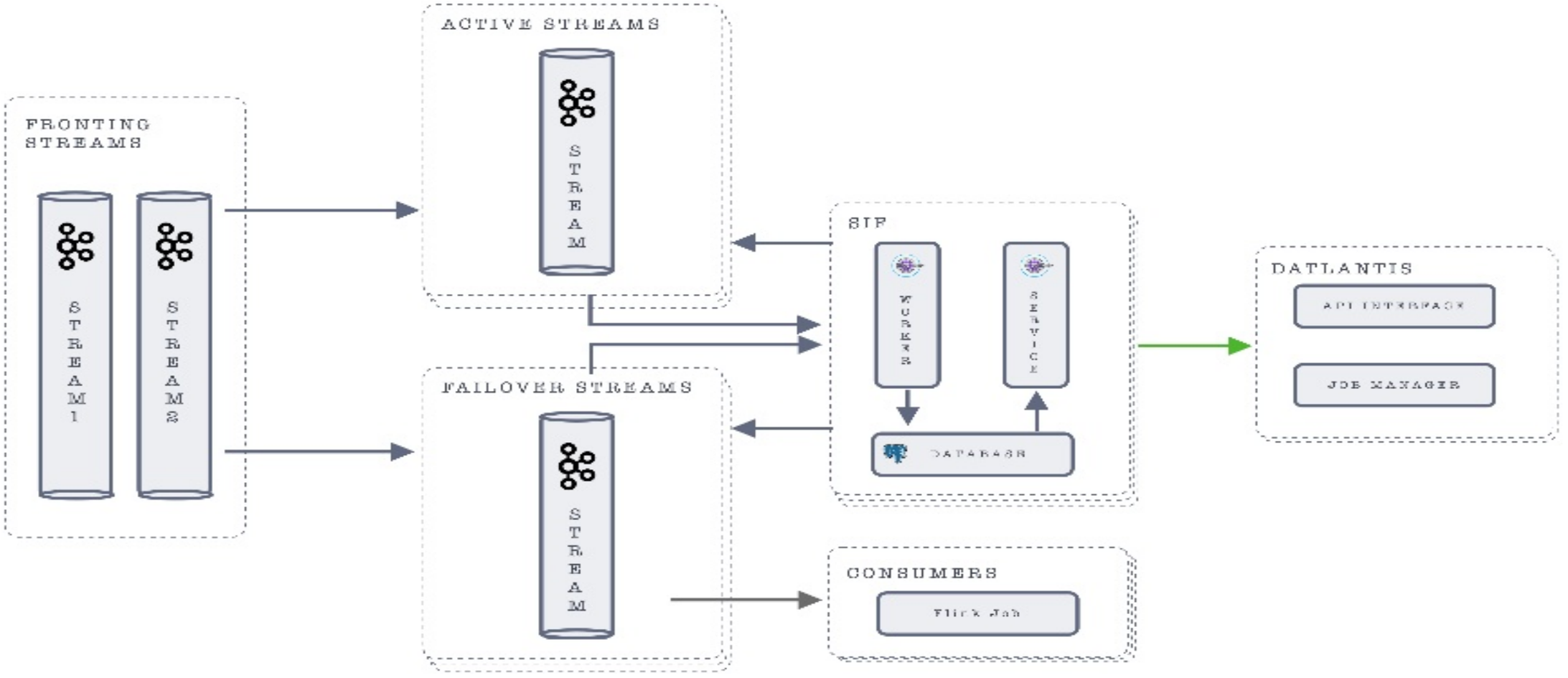
Kafka Failover



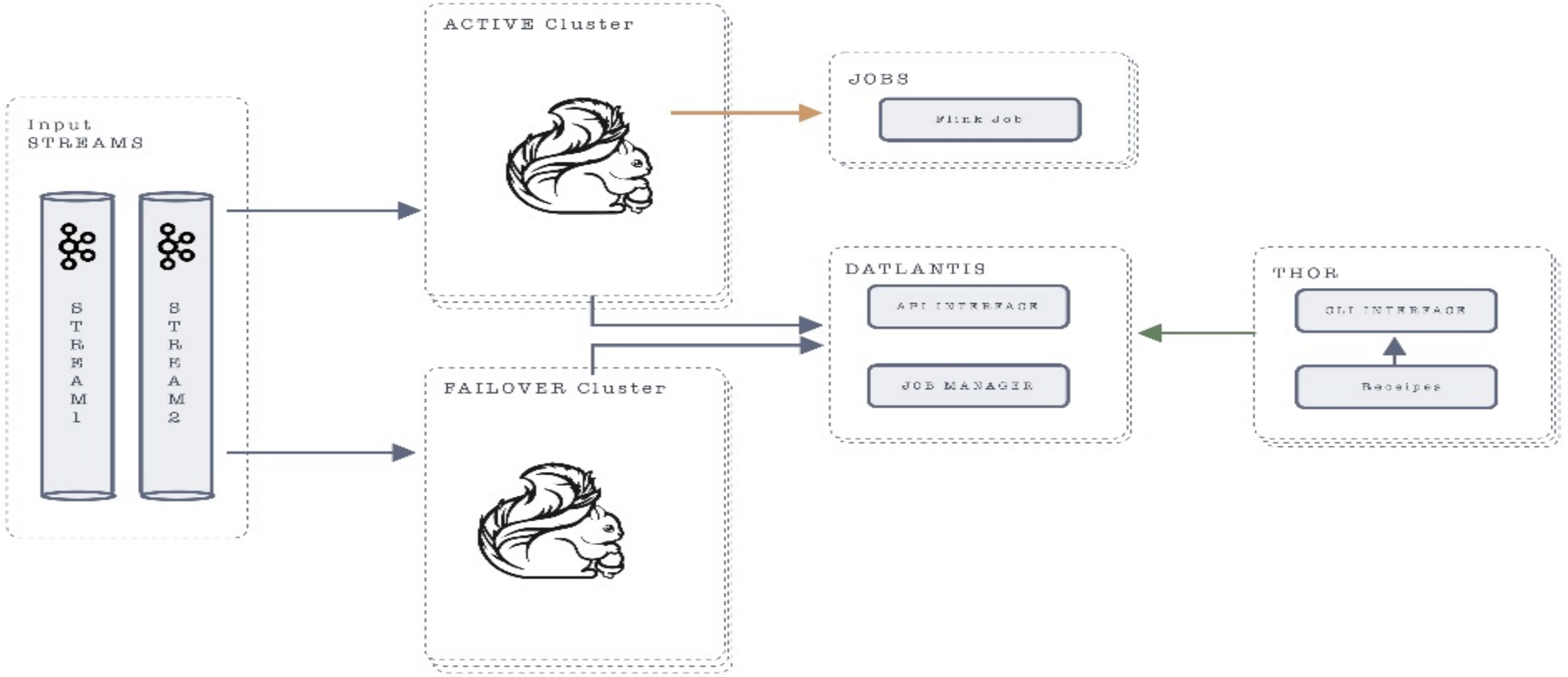
Kafka Failover



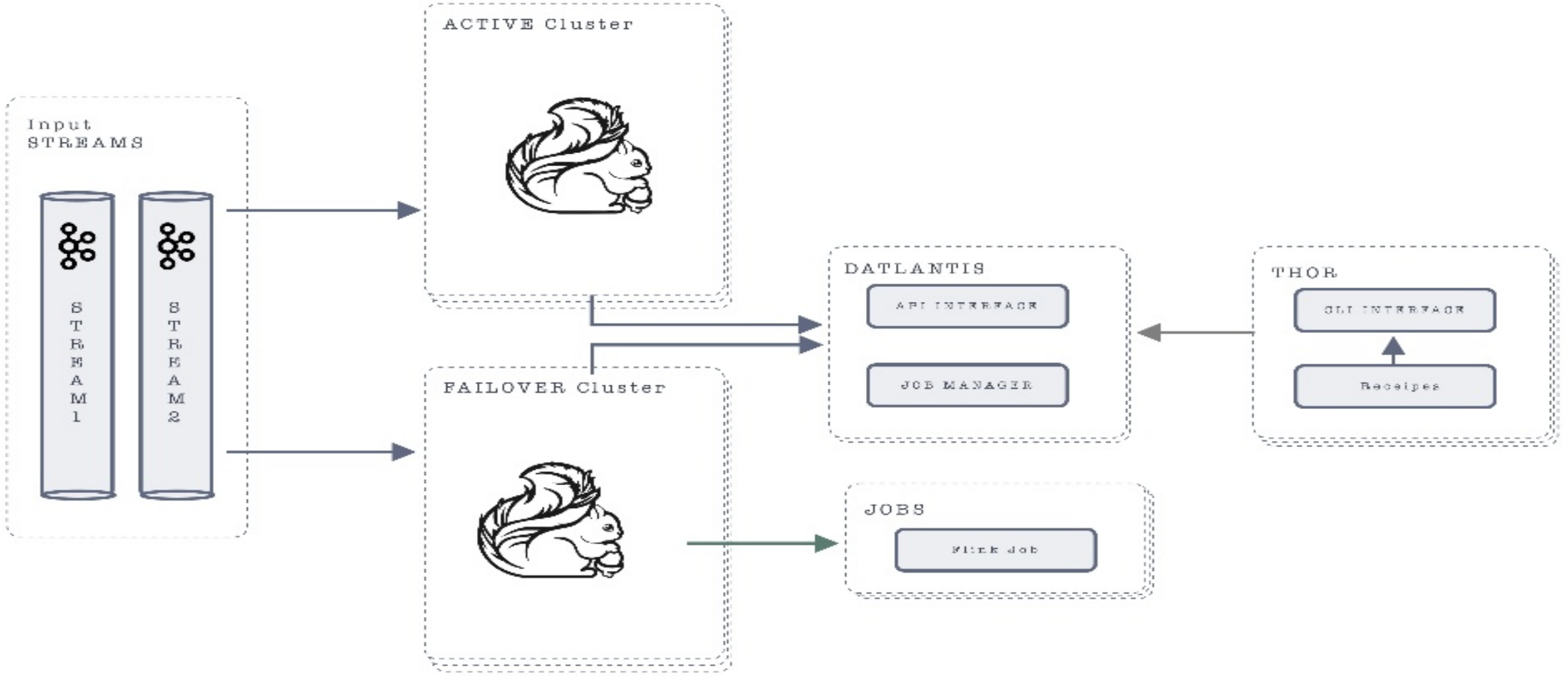
Kafka Failover



Flink Failover



Flink Failover



GO JEK

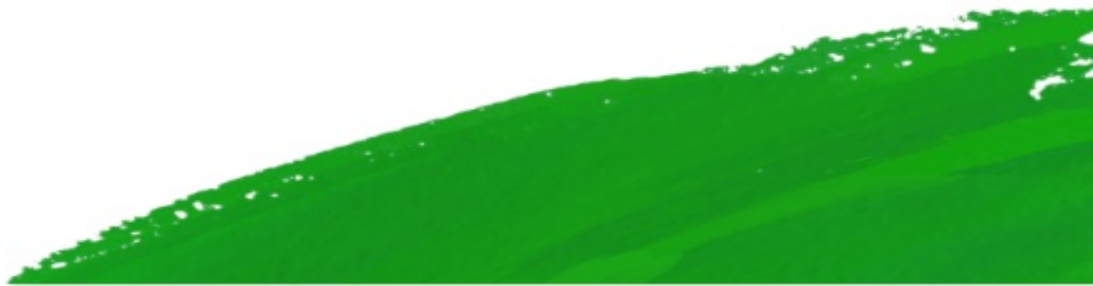
Impact

Kafka input stream resiliency

Allocate jobs dynamically to clusters

Testing and upgradation becomes easy

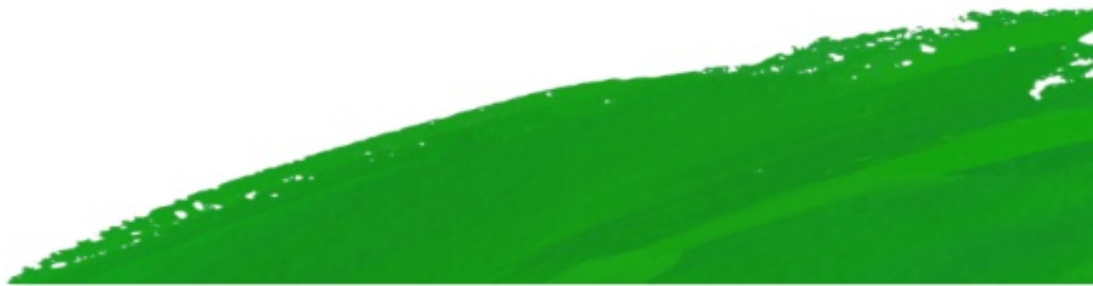
Self-serve and no workflow



GO JEK

Chaos and load Testing

How do we build confidence in system behaviour through disaster simulation experiments?

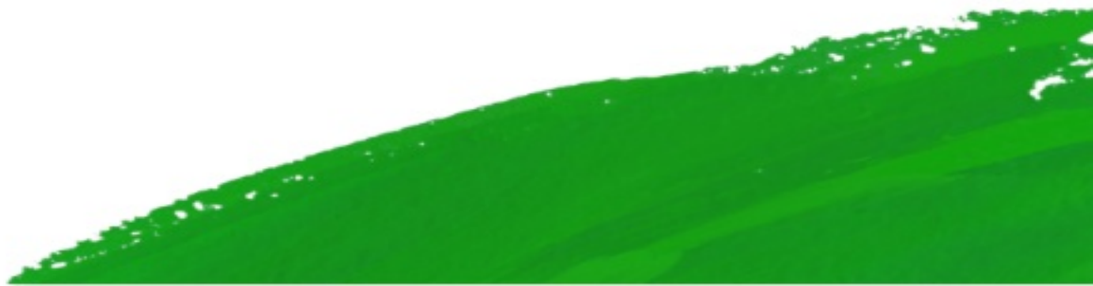


GO JEK

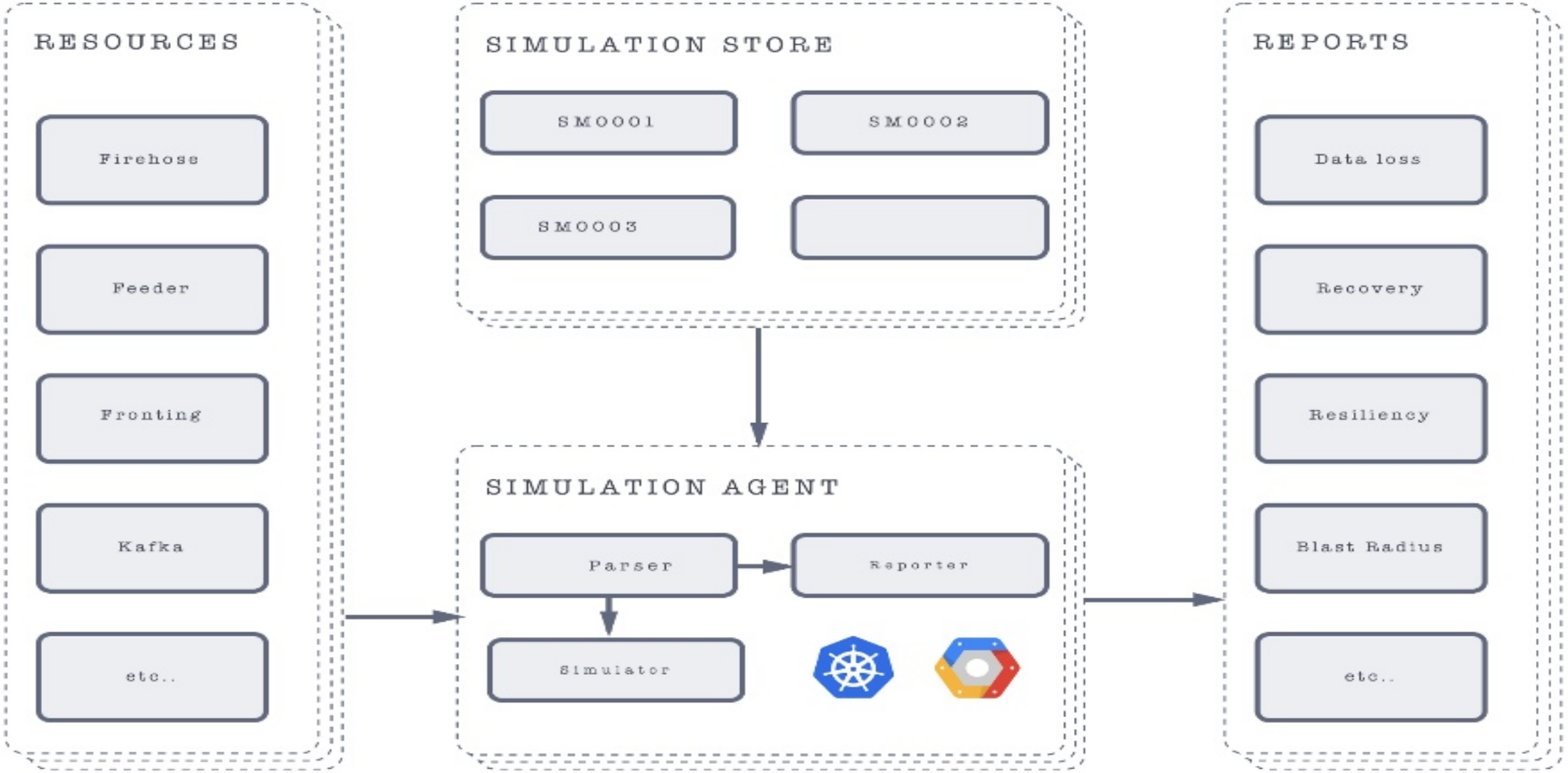
LOKI for chaos engineering

- Disaster simulation
- Load testing
- Reports
- CLI interface

https://bit.ly/2NkGE9L



Architecture



Simulation

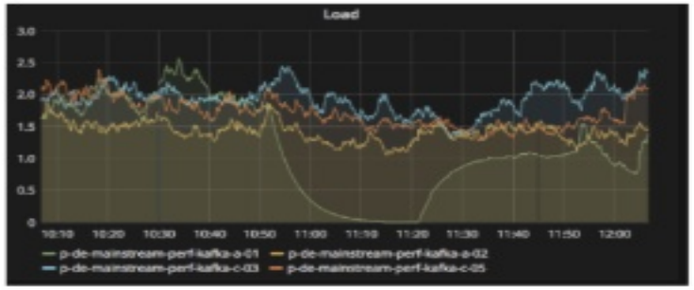
Simulate throughtput in flink jobs with kafka ingestion

Play chaos simulation on flink cluster or kafka cluster

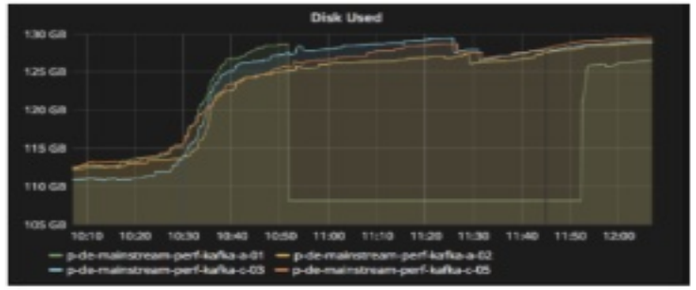
Generate reports

Prepare playbook

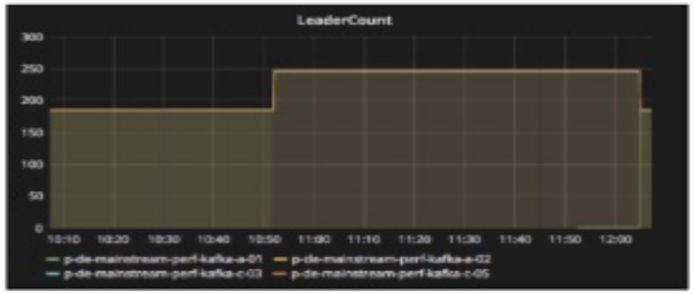
Reports


Load
p-de-mainstream-perf-kafka-a-01
p-de-mainstream-perf-kafka-a-02
p-de-mainstream-perf-kafka-c-03
p-de-mainstream-perf-kafka-c-05

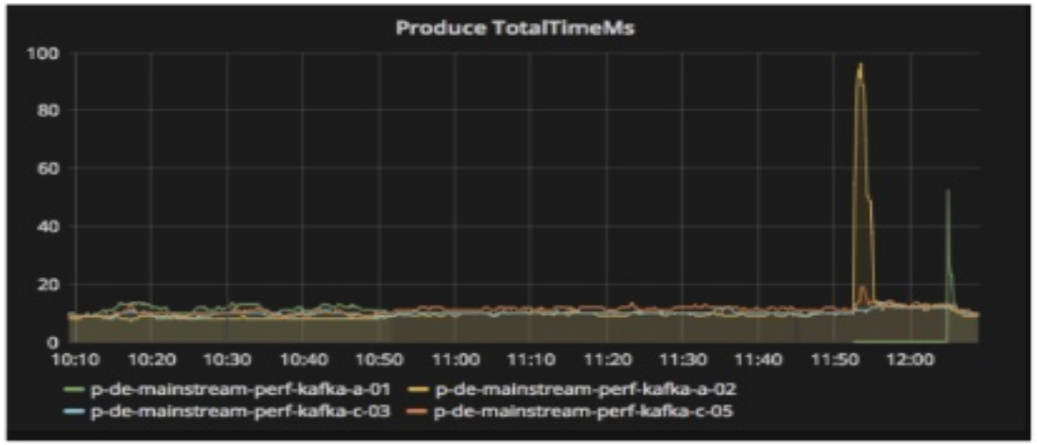

CPU
p-de-mainstream-perf-kafka-a-01
p-de-mainstream-perf-kafka-a-02
p-de-mainstream-perf-kafka-c-03
p-de-mainstream-perf-kafka-c-05

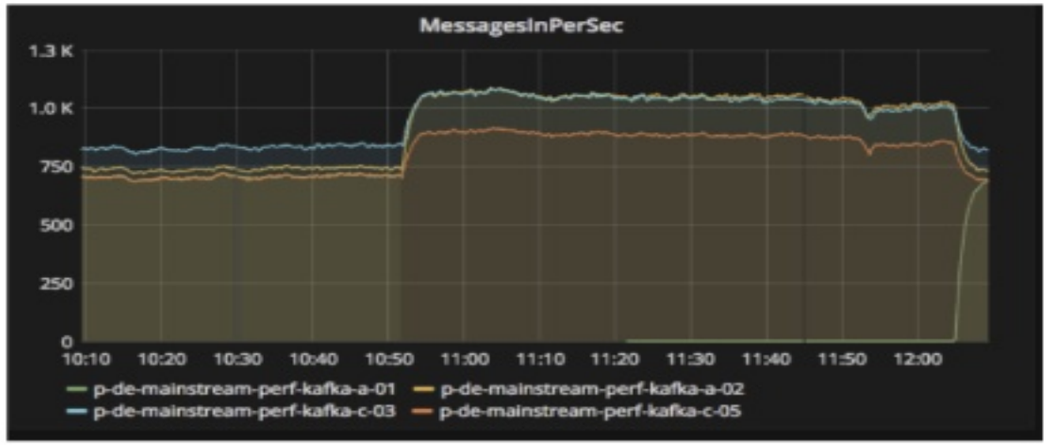

Disk Used
p-de-mainstream-perf-kafka-a-01
p-de-mainstream-perf-kafka-a-02
p-de-mainstream-perf-kafka-c-03
p-de-mainstream-perf-kafka-c-05


UnderReplicatedPartitions
p-de-mainstream-perf-kafka-a-01
p-de-mainstream-perf-kafka-a-02
p-de-mainstream-perf-kafka-c-03
p-de-mainstream-perf-kafka-c-05


LeaderCount
p-de-mainstream-perf-kafka-a-01
p-de-mainstream-perf-kafka-a-02
p-de-mainstream-perf-kafka-c-03
p-de-mainstream-perf-kafka-c-05


ReplicaManager.IsrExpandsPerSec
p-de-mainstream-perf-kafka-a-01
p-de-mainstream-perf-kafka-a-02
p-de-mainstream-perf-kafka-c-03
p-de-mainstream-perf-kafka-c-05


Produce TotalTimeMs
p-de-mainstream-perf-kafka-a-01
p-de-mainstream-perf-kafka-a-02
p-de-mainstream-perf-kafka-c-03
p-de-mainstream-perf-kafka-c-05


MessagesInPerSec
p-de-mainstream-perf-kafka-a-01
p-de-mainstream-perf-kafka-a-02
p-de-mainstream-perf-kafka-c-03
p-de-mainstream-perf-kafka-c-05

Road Ahead

Flink on Kubernetes

Complex Event Processing

Data Enrichment

Let's talk!

Ravi Suhag

@ravi_suhag

medium.com/@ravisuhag

Sumanth K N

medium.com/@kn.sumanth