getindata

Kubernetes and real-time analytics

How to connect these two worlds with Apache Flink?

Author: Albert Lewandowski

About me

- Big Data DevOps Engineer - GetInData
- Focused on infrastructure, cloud, Big Data, AI, scalable web applications
- Certified Google Cloud Architect
- Certified Kubernetes Administrator

Content

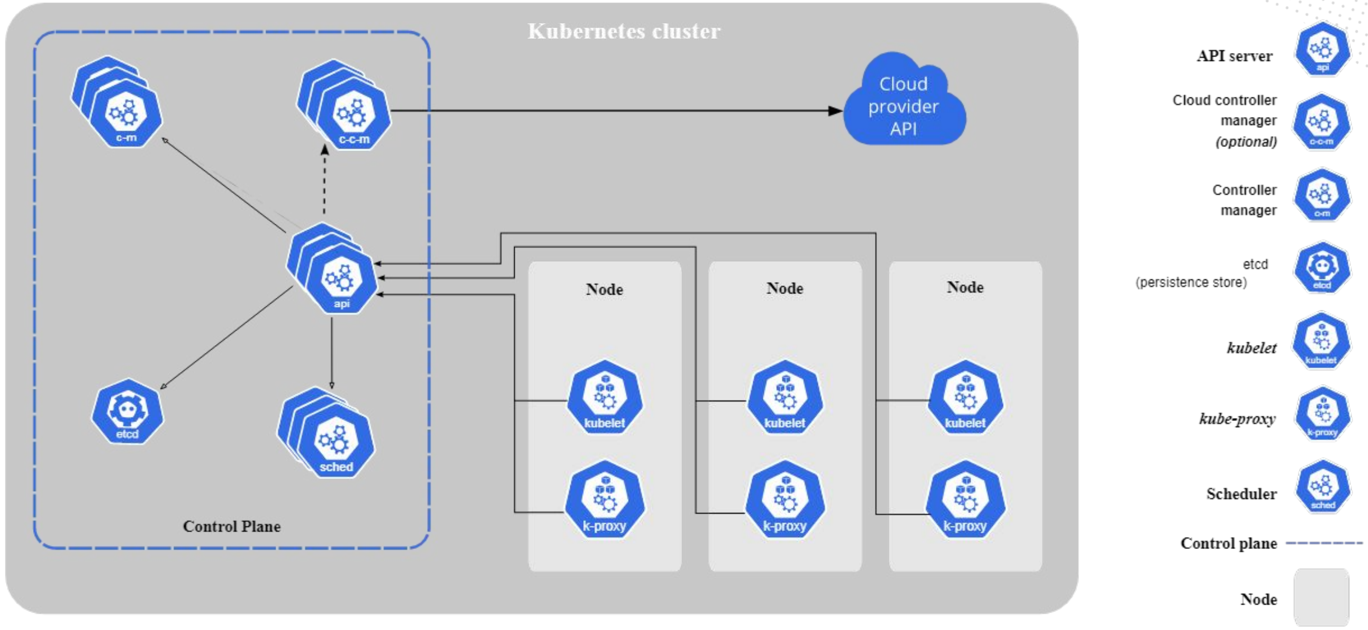
getindata

- Principles in the Big Data world on Kubernetes
- Why real-time data streaming?
- Different faces of Apache Flink.
- Flink and Kubernetes - real life scenarios.
- Observability of the platform.
- Quick start on your computer.

Introduction to the jungle

What is Kubernetes?

Open-source platform for managing containerized workloads and services



Kubernetes - Operators

Method of **deploying** and **managing** app

Automated **provisioning** of resources

One setup for multiple environments

Examples: [pulsar-operator](), [postgres-operator](), [prometheus-operator]()

Kubernetes - Custom Resource Definitions

Defining **custom APIs** as add-ons

Dynamic registration with Kubernetes API

CRDs can be accessed with kubectl

A CRD represents the desired state and an operator makes it happen.

What is Apache Flink?

Flink is an open-source **stream processing framework** that supports both **batch** processing and data **streaming** programs

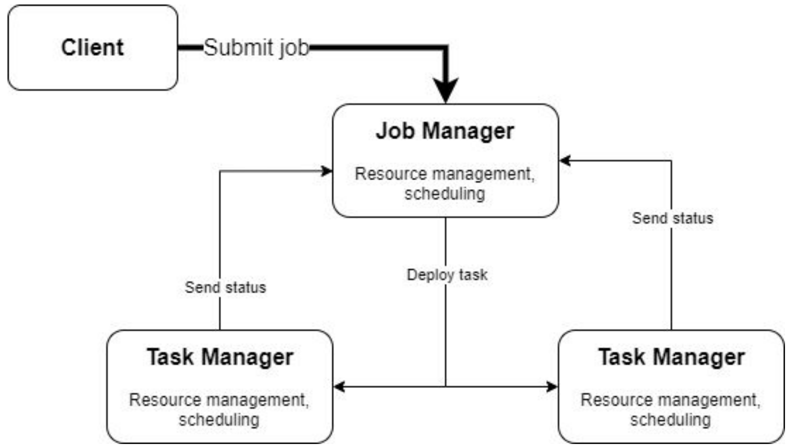
State of the Flink job

A savepoint is a consistent image of the execution state of a streaming job

Flink's Savepoints are different from Checkpoints in a similar way that backups are different from recovery logs in traditional database systems.

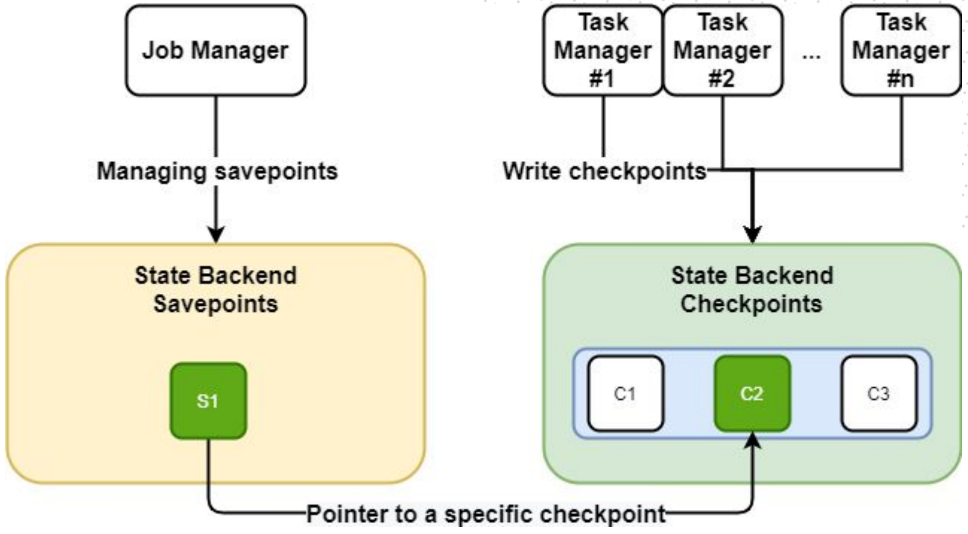
What is Apache Flink?

Job Diagram



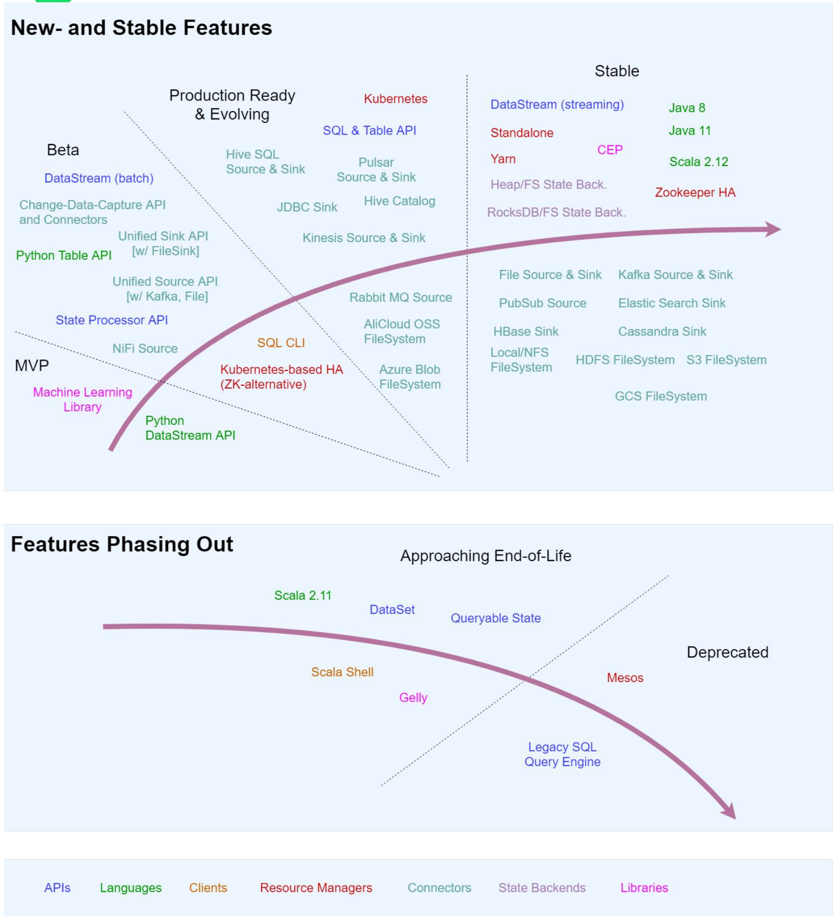
Client
Submit job
Job Manager
Resource management, scheduling
Send status
Deploy task
Send status
Task Manager
Resource management, scheduling
Task Manager
Resource management, scheduling

State of Flink job Diagram



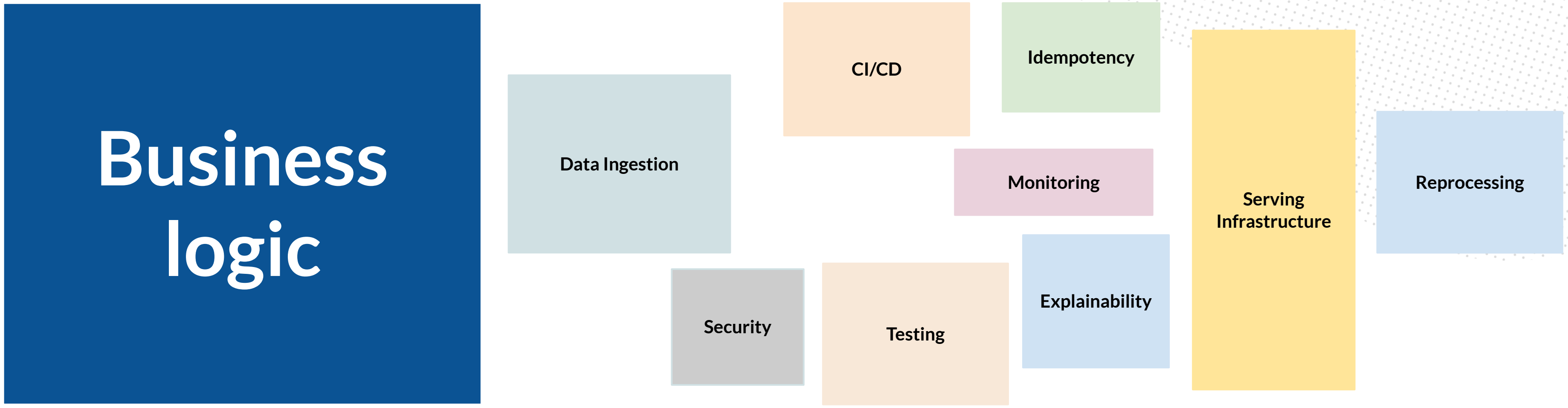
Apache Flink Roadmap

getindata

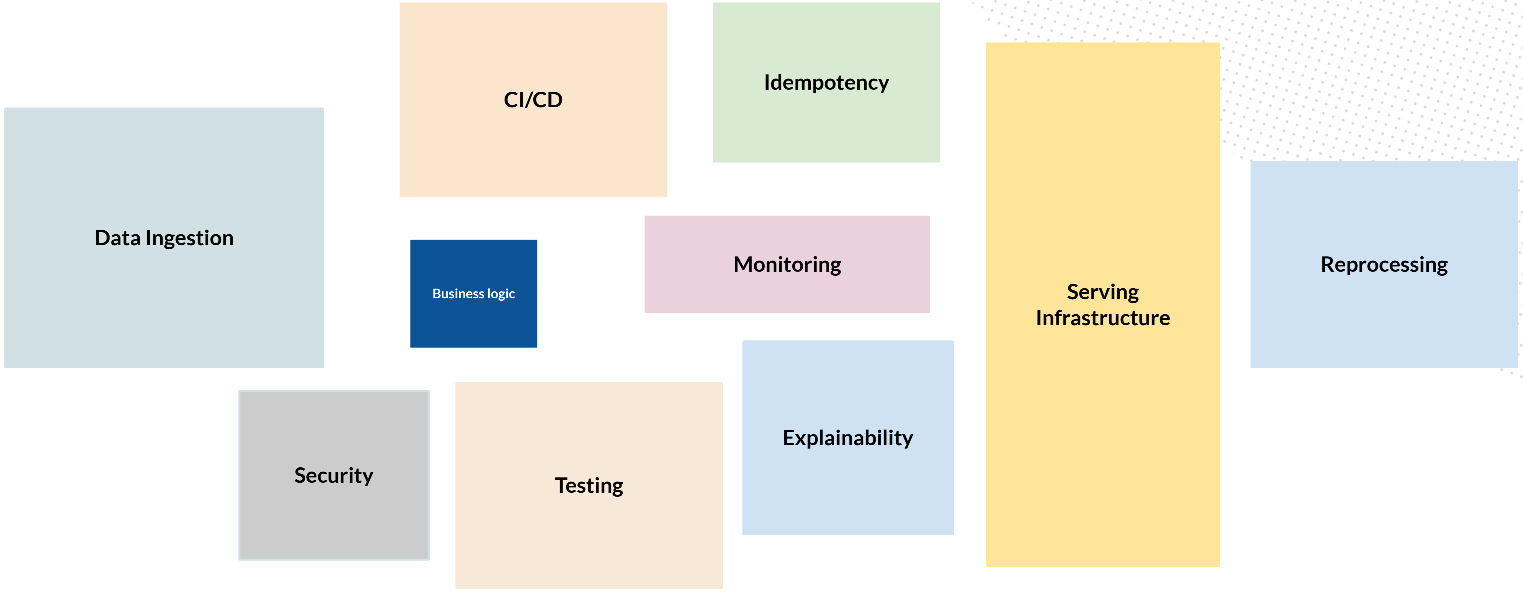


Source: [Roadmap - Apache Flink]

Perception


Business logic
Data Ingestion
CI/CD
Idempotency
Monitoring
Serving Infrastructure
Reprocessing
Security
Testing
Explainability

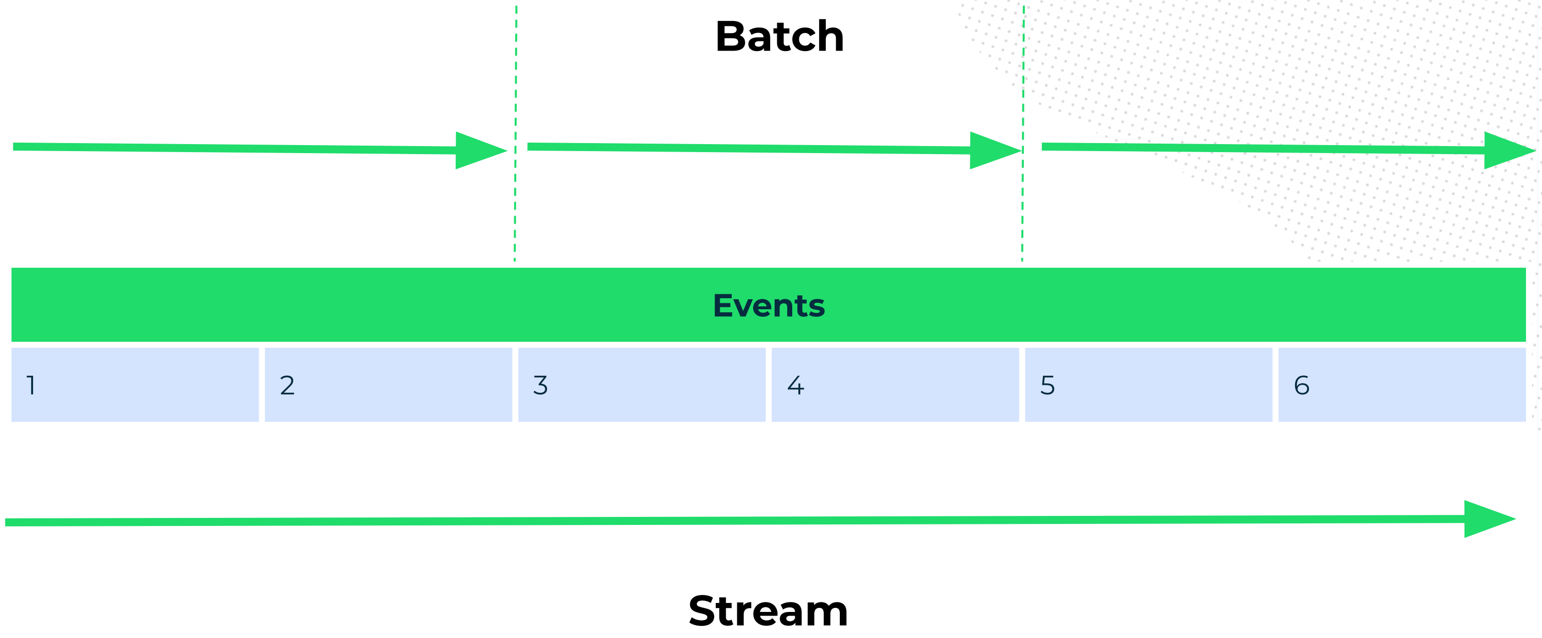
Reality


CI/CD
Idempotency
Data Ingestion
Business logic
Monitoring
Serving Infrastructure
Reprocessing
Security
Testing
Explainability

Real-time data streaming

Data Streaming vs. Batch

getindata



Use cases

getindata

User activity

Location data

Fraud detection

Logistics

Industrial IoT

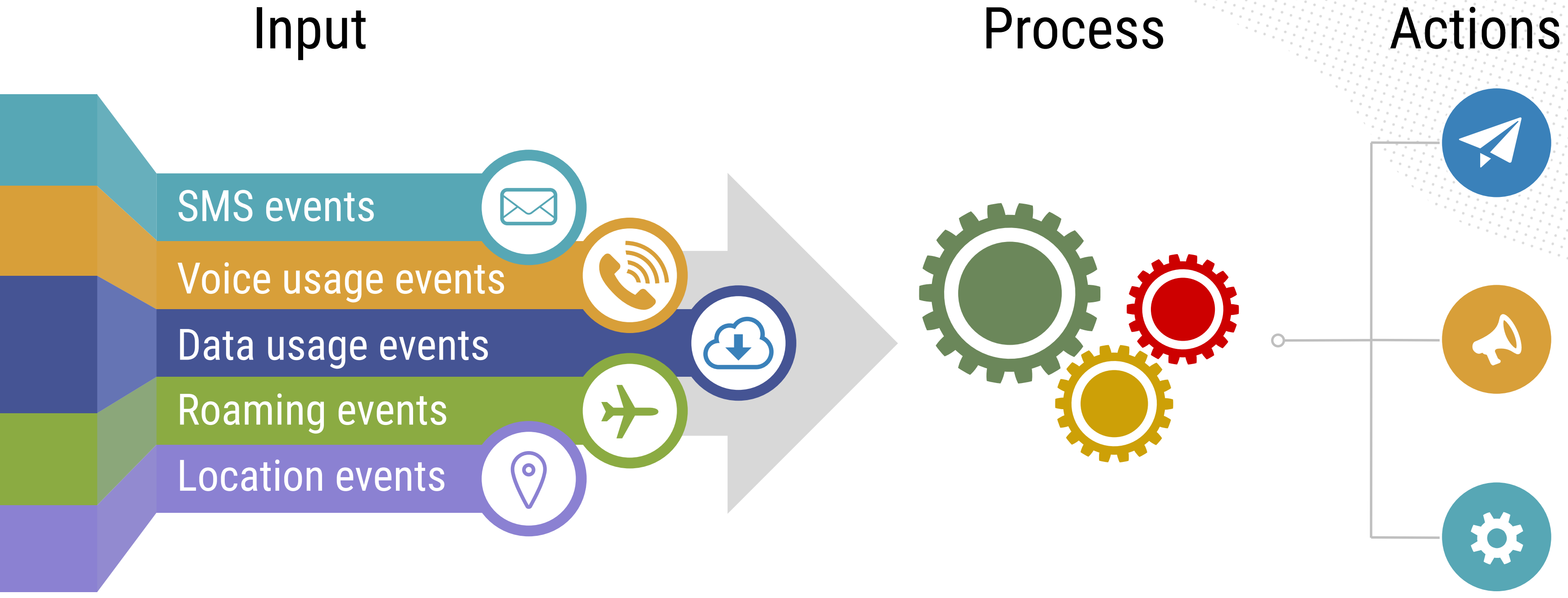
Recommendations

Use case - Kcell - Telecom company in Kazakhstan

2018	10M Subscribers	165K Events / s	22.5 TB / month
2020	10M Subscribers	500K Events / second	40 TB / month

Use case - Kcell

getindata



Use case - Kcell - some scenarios for Flink

getindata

Balance Top Up Case

If subscriber top-ups her balance too often in short period of time. We can offer her a less expensive tariff or auto-payment services.

Fraud case in roaming

Send an email to the anti-fraud unit if subscriber registered in roaming but his balance at the moment is equal to 0.
This situation is impossible in standard case.

Automatic SIM card activation

Send an email to the anti-fraud unit if subscriber registered in roaming but his balance at the moment is equal to 0.
This situation is impossible in standard case.

Dealer Motivation Case

Trigger bonus for a dealer when we discover that purchase happened attributable to him/her.

Apache Flink
One tool, multiple *versions*

One tool, multiple languages

getindata

Java 8 or 11

Scala 2.11 or 2.12

SQL

Python

Where should I install?

getindata

YARN cluster

Kubernetes

Standalone

- CICD process
- Service Discovery - monitoring with Prometheus
- Scalability
- Managing resources
- A/B Testing

High Availability of Flink

getindata

Storage level

- High Availability of storage to/from which Flink writes/reads savepoints and checkpoints
- Performance of storage

JobManager level

- ZooKeeper
- Kubernetes (beta)

Job Strategy

- Data reprocessing policy
- How to deploy new job?

	Flink K8S Operator	Kubernetes Operator for Apache Flink	Ververica Platform	Native Kubernetes - Apache Flink
CRDs	Yes	Yes	No	No
CICD	Kubernetes API	Kubernetes API	REST API or Web UI	Kubernetes API
Installation	Helm chart or raw Kubernetes manifests	Helm chart or raw Kubernetes manifests	Helm chart or raw Kubernetes manifests	No need to install any component
SQL Editor	No	No	Yes	No
Dependencies	No	No	Persistence volume for database Object storage for artifactory	No
Status	beta	beta	production	beta

Flink + Kubernetes = ?

Overview in the article here

Why Flink on Kubernetes?

getindata

Simpler deployment process

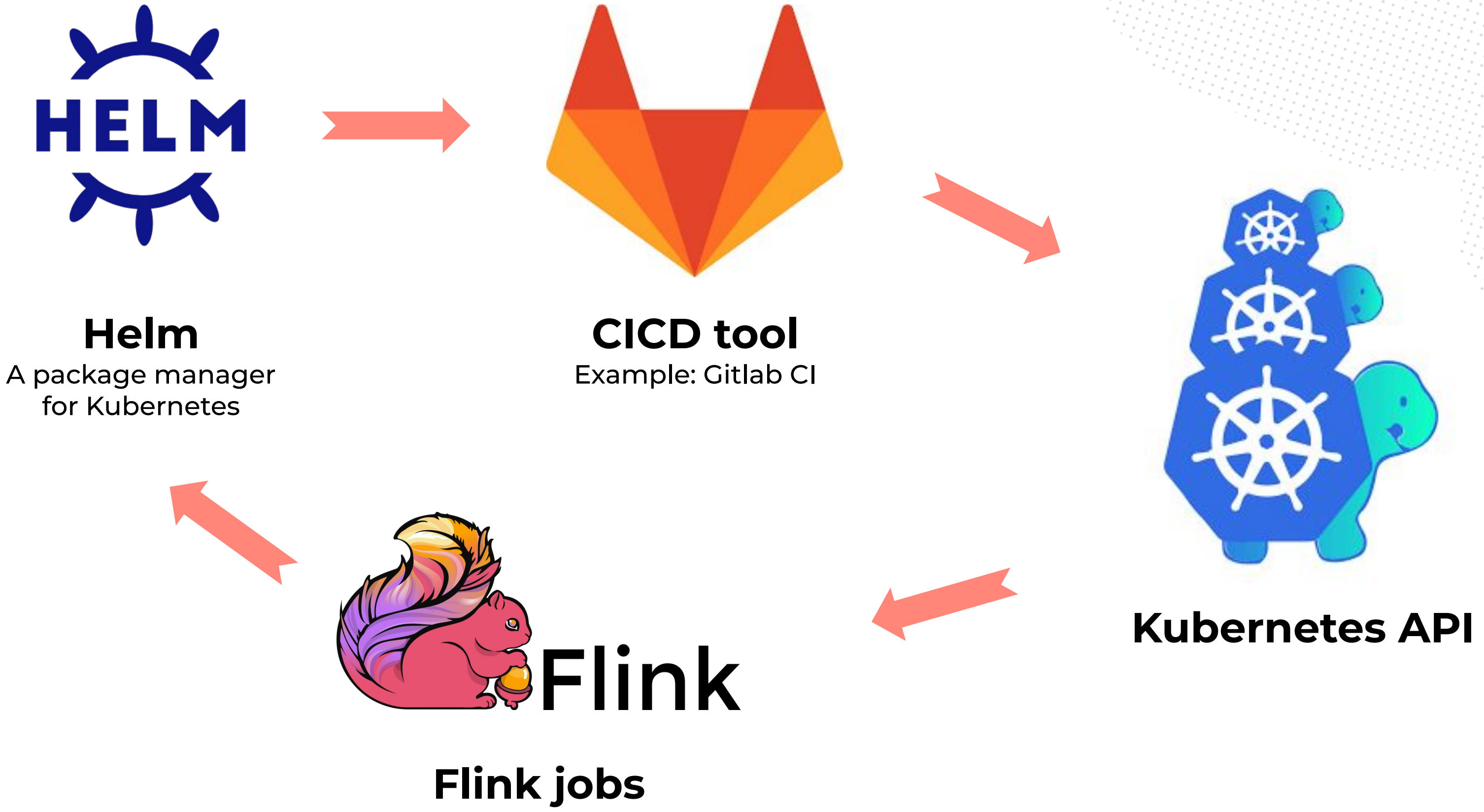
Flexible jobs management

Simple Service Discovery - Prometheus

Flexible testing

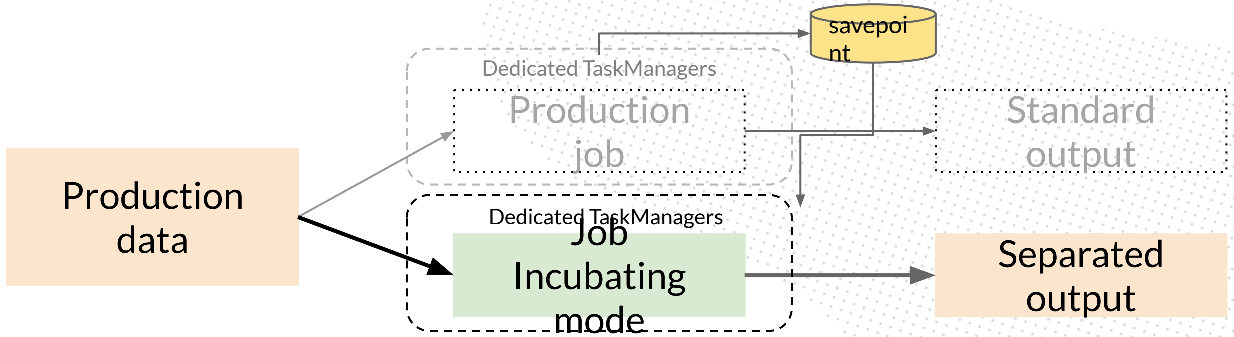
Installation & Configuration

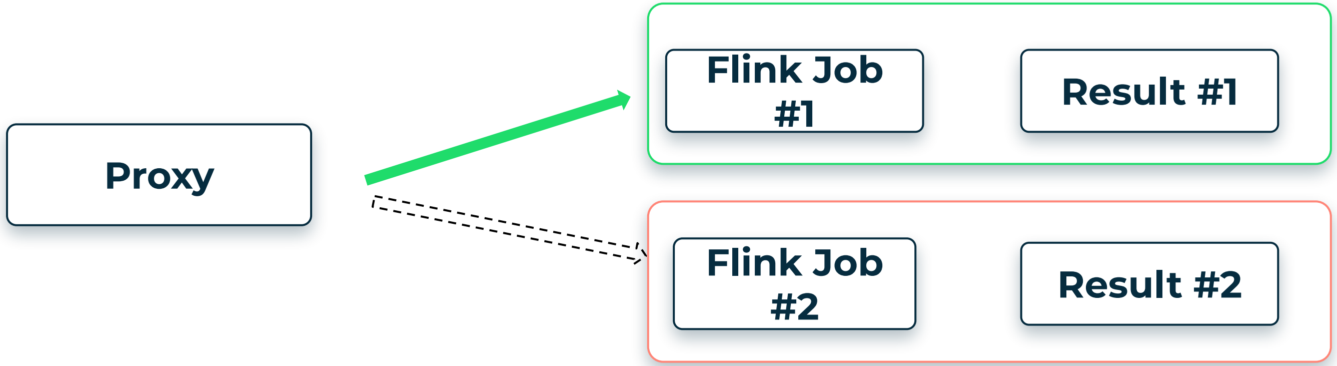
getindata



Testing

Incubating Mode





Blue Green Deployment

A/B Testing

Deployment process

getindata

Git Flow

Unit & Integration tests

Versioning images

Deployment process

Monitoring

Kubernetes aspects

getindata

Dedicated namespaces

Resources

Secured access to Flink (RBAC)

Configuration files

Network performance

Storage for savepoints&checkpoints

Secrets

Self-healing and autoscaling

getindata

Scale based on metrics

Flink restarts

External tool for fixing

Automate manual tasks

Re-create cluster

Job Cluster & Session Cluster

getindata

Job Cluster

Full set of Flink cluster for each individual job

Long running tasks

Separate images for different jobs

Session Cluster

Standalone Flink cluster on Kubernetes

Short running tasks

Ad-hoc queries

Stories from production

- Automate in the beginning
- CICD pipeline is a must
- Verify JVM metrics
- Test different Flink configurations to get the best performance and no restarts
- Secure access to Flink jobs
- Get logs from Flink TMs and JMs

Local Setup

How to start locally?

getindata

- Minikube / Docker Desktop or any different local K8s env
- Ververica Platform
- Locally started Kafka cluster or use a Datagen

APACHE FLINK

KUBERNETES

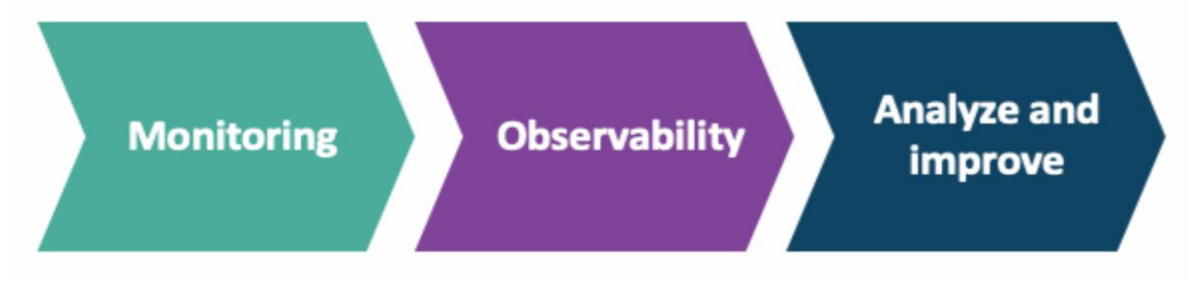
STREAMING SQL

Observability

Whitepaper - here

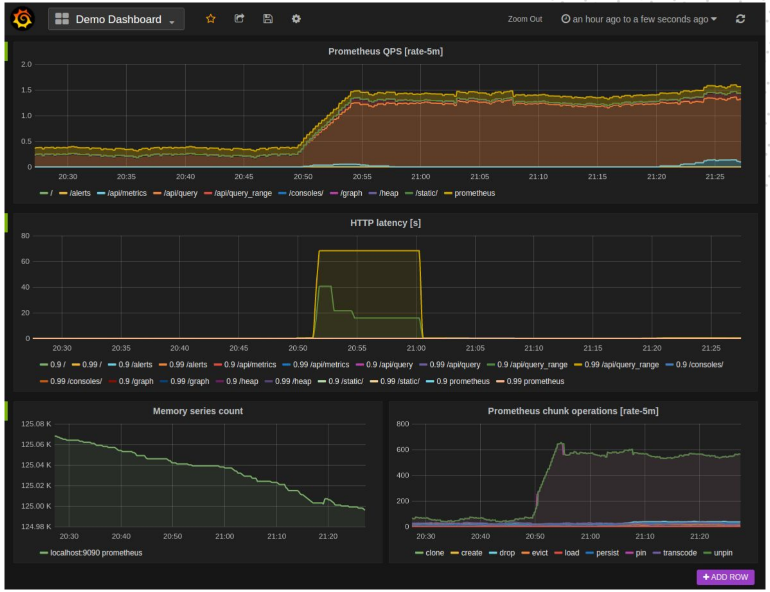
Observability

Observability is about measuring how well internal states of the system can be inferred from knowledge of its external outputs (according to the control theory).



Part One: Metrics

Get metrics from environment and application - but how?



Prometheus - Kubernetes-native solution

open-source systems
monitoring and alerting toolkit

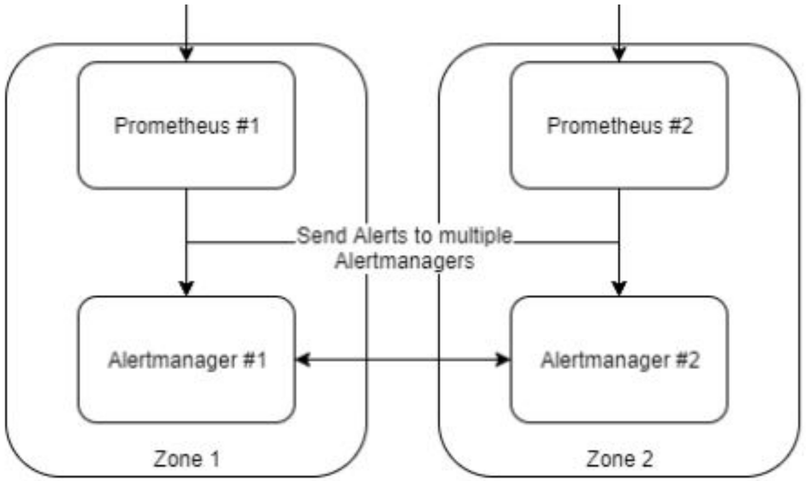
joined the **Cloud Native Computing Foundation** in **2016** as the second hosted project, after Kubernetes

a lot of **exporters**
you can write your own easily

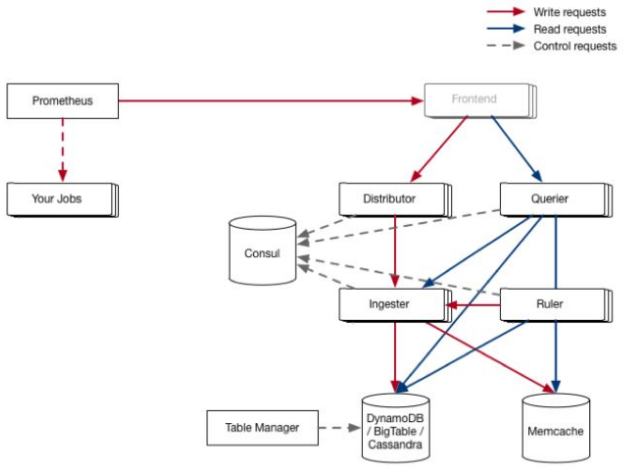
mature ecosystem
PushGateway, Blackbox, AlertManager, etc.

Prometheus - simple or complex High Availability?

Simple



Complex



Example solutions: Cortex (above), Thanos, M3DB

Pull vs. push-based monitoring

getindata

Pull	Push
Collector takes metrics	Agents push metrics
Workload on central poller increases with the number of devices polled.	Polling task fully distributed among agents, resulting in linear scalability.
Polling protocol can potentially open up system to remote access and denial of service attacks.	Push agents are inherently secure against remote attacks since they do not listen for network connections.
Flexible: poller can ask for any metric at any time.	Relatively inflexible: pre-determined, fixed set of measurements are periodically exported.

Prometheus - Stories

getindata

service discovery
simple on k8s

limited **security**

archived data
how old data is required?

monitor **monitoring**

Part Two: Logs analytics

1. Get logs from app or environment.
2. Save logs.
3. Query them.
4. Make your system self-healing and discover what’s happening inside your platform.

Logs analytics - which tool should I choose?

getindata

Logs Analytics for **Developers**

Loki

Logs Analytics for **Business**

ElasticSearch

ELK vs. Loki

getindata

	ELK	Loki + Promtail/Fluentd
Indexing	Keys and content of each key	Only labels
Query language	Query DSL or Lucene QL	LogQL
Tool for data visualisation	Kibana	Grafana
Query performances	Faster due to indexed all the data	Slower due to indexing only labels
Resource requirements	Higher due to the need of indexing	Lower due to index only labels

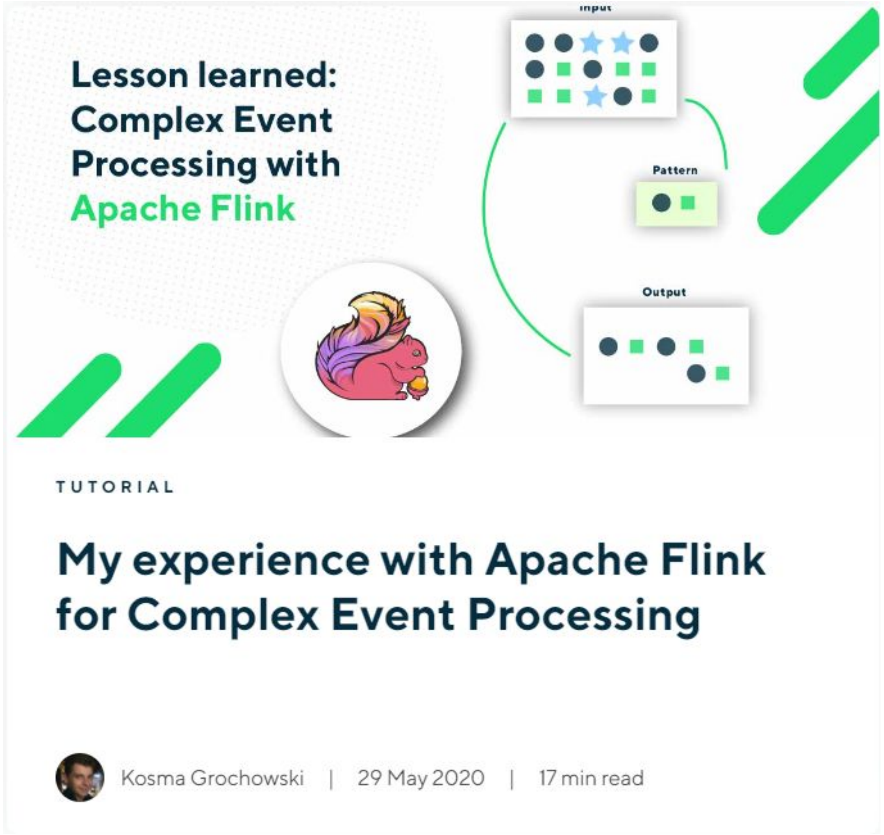
What about alerts?

getindata

Alerts signify that ***a human needs to take action immediately*** *in response to something that is either happening or about to happen, in order to improve the situation.*

Quick start

Flink - Complex Event Processing



TUTORIAL

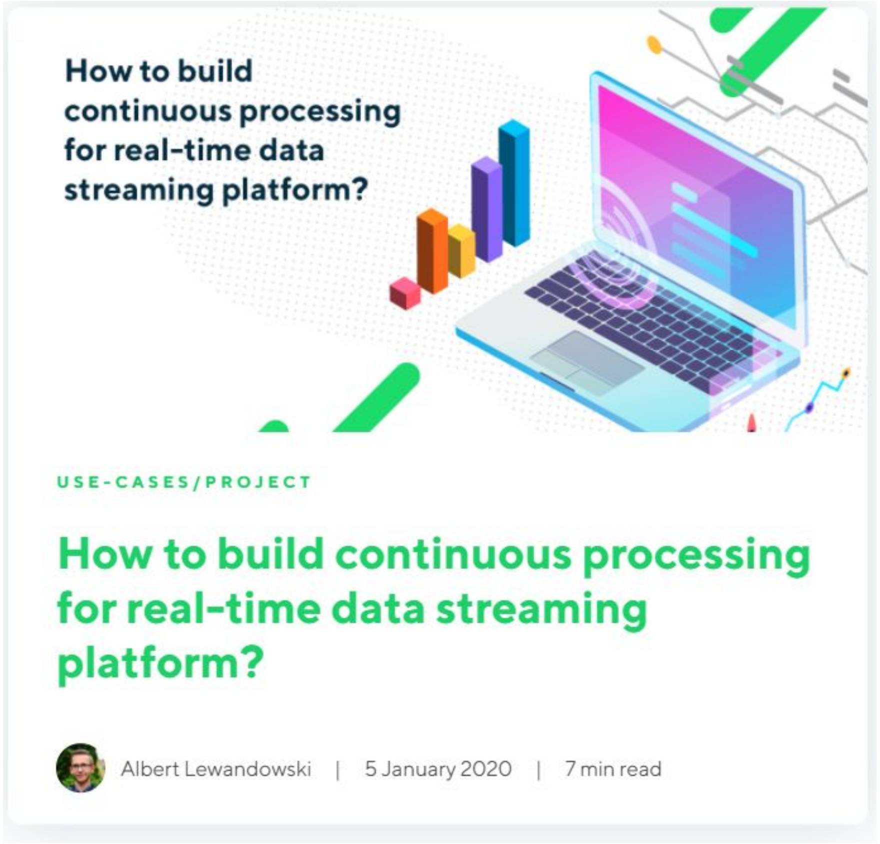
My experience with Apache Flink for Complex Event Processing

Kosma Grochowski | 29 May 2020 | 17 min read

[Article](#).

[Codebase for example](#).

DevOps best practises



How to build continuous processing for real-time data streaming platform?

USE-CASES/PROJECT

How to build continuous processing for real-time data streaming platform?

Albert Lewandowski | 5 January 2020 | 7 min read

Article.

Kubernetes - first setup

- Minikube
- Kind
- Use Kubernetes service from public cloud provider like AWS, GCP, Azure during free tier

Kubernetes + Flink - Operator

Requirements: Kubernetes cluster, kubectl

```
$ kubectl create -f https://raw.githubusercontent.com/lyft/flinkk8soperator/v0.5.0/deploy/crd.yaml
$ kubectl create -f https://raw.githubusercontent.com/lyft/flinkk8soperator/v0.5.0/deploy/namespace.yaml
$ kubectl create -f https://raw.githubusercontent.com/lyft/flinkk8soperator/v0.5.0/deploy/role.yaml
$ kubectl create -f https://raw.githubusercontent.com/lyft/flinkk8soperator/v0.5.0/deploy/role-binding.yaml
$ kubectl create -f
https://raw.githubusercontent.com/lyft/flinkk8soperator/v0.5.0/deploy/flinkk8soperator.yaml
```

Verify if it works:

```
$ kubectl -n flink-operator get po
```

Run the example job:

```
$ kubectl create -f
https://raw.githubusercontent.com/lyft/flinkk8soperator/v0.5.0/examples/wordcount/flink-operator-custom-resou
rce.yaml
```

Verify if it is running and its status:

```
$ kubectl get flinkapplication.flink.k8s.io -n flink-operator wordcount-operator-example -o yaml
```

Kubernetes + Ververica Platform

getindata

Requirements: Kubernetes cluster, kubectl, Helm

Install Ververica Platform locally with Helm

```
$ helm repo add ververica https://charts.ververica.com
$ helm install vvp ververica/ververica-platform
$ helm install vvp ververica/ververica-platform --set acceptCommunityEditionLicense=true
```

Verify if Ververica is up

```
$ kubectl get po
```

Access the web user interface and REST API

```
$ kubectl port-forward service/vvp-ververica-platform 8080:80
```

Do you want to test Flink SQL feature? Use Flink Faker (a data generator source connector)

https://github.com/knaufk/flink-faker/

It requires changing used image for vvp-gateway.

Join Us!

Data Engineer
Spark, Kafka, Airflow, public cloud
Link

Backend Engineer
Java / Scala, microservices
Link

MLOps Engineer
MLOps tools, Python, public cloud
Link

DevOps / SRE
GCP, Terraform, Prometheus
Link

Q&A

Contact details

getindata

albert.lewandowski@getindata.com

LinkedIn:
https://www.linkedin.com/in/albert-lewandowski

getindata

Thank you for your attention!