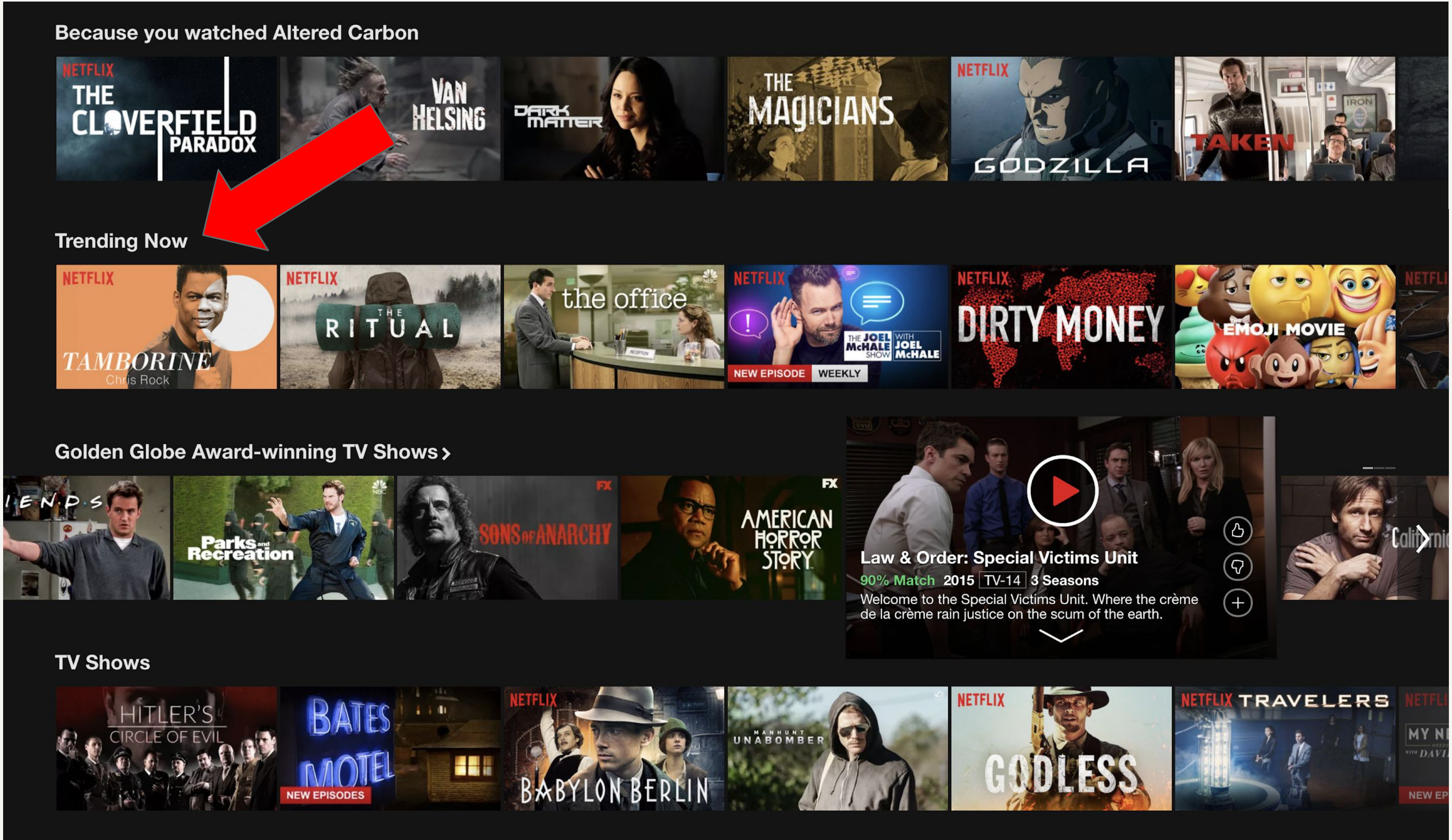
Building Stream Processing as a Service (SPaaS)

Steven Wu

@stevenzwu

NETFLIX

Why stream processing?



Because you watched Altered Carbon

Trending Now

Golden Globe Award-winning TV Shows ›

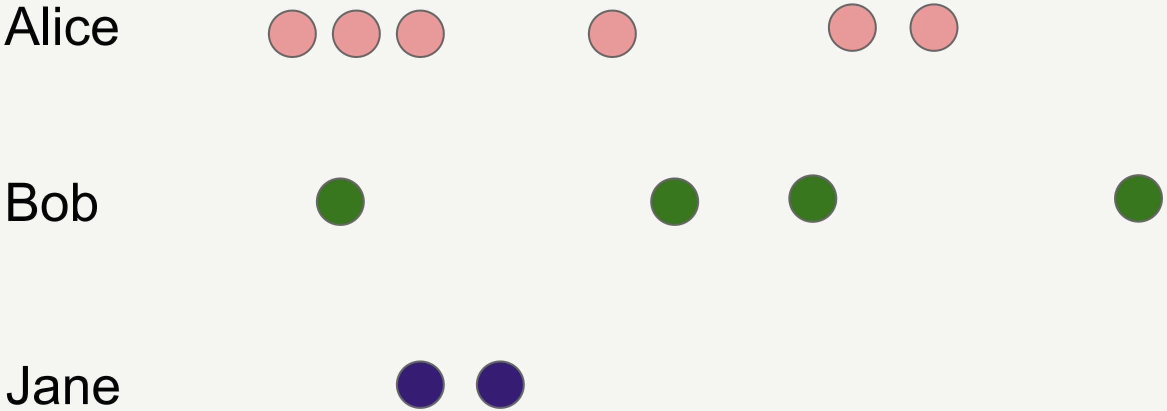
Law & Order: Special Victims Unit

90% Match 2015 TV-14 3 Seasons

Welcome to the Special Victims Unit. Where the crème de la crème rain justice on the scum of the earth.

TV Shows

Unbounded user activity stream

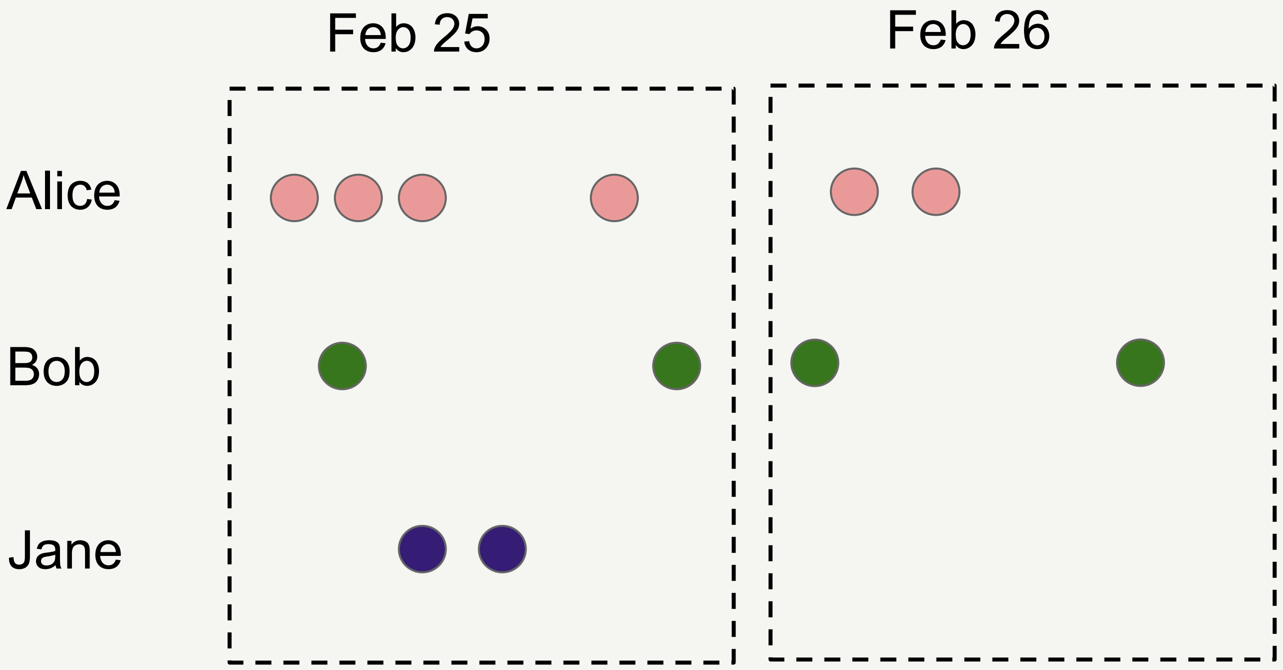


Alice

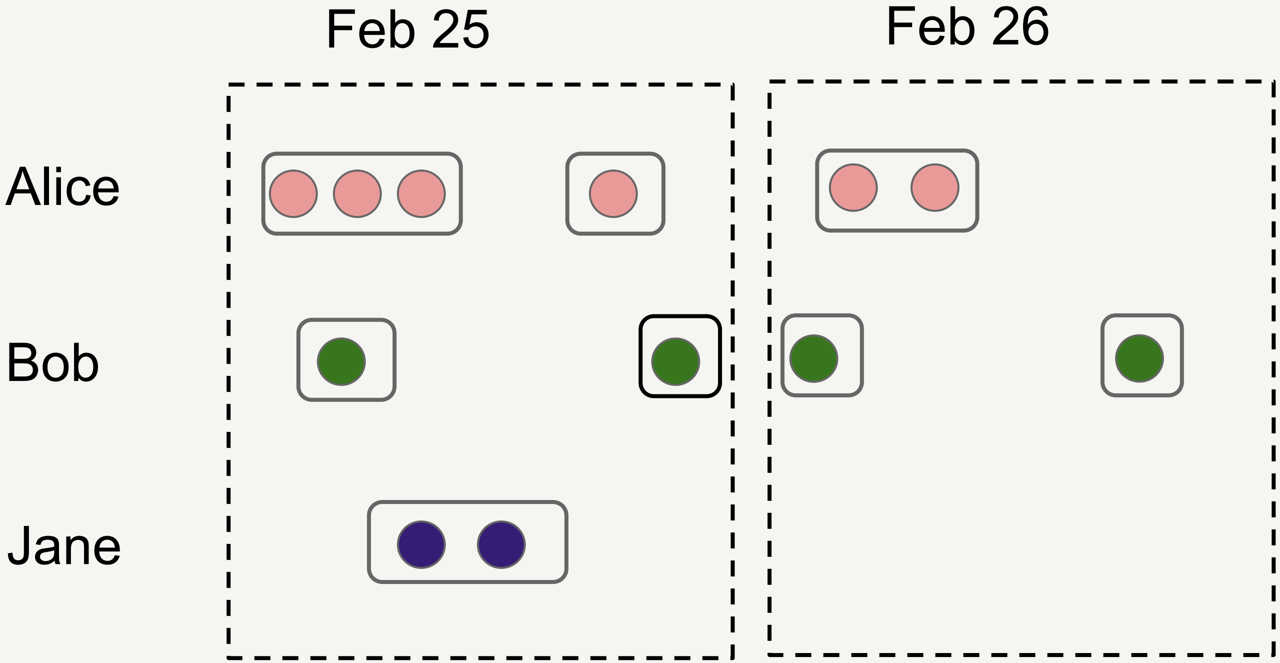
Bob

Jane

Unbounded data - batch



Unbounded data - batch



Feb 25

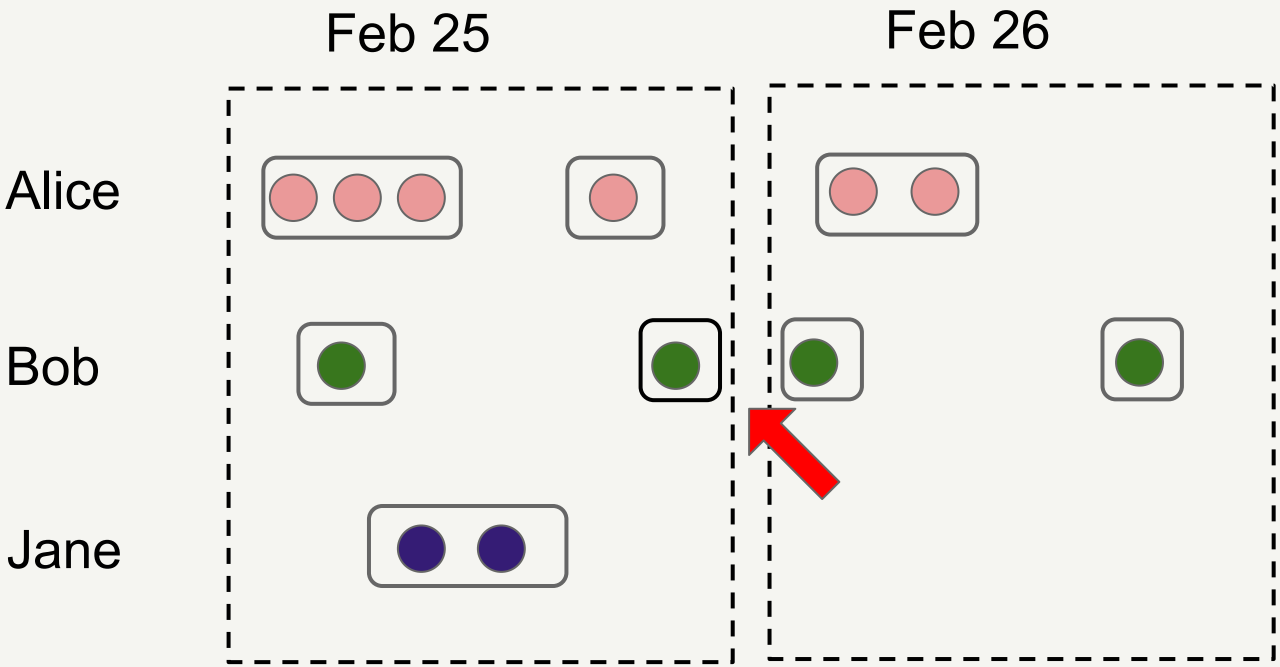
Feb 26

Alice

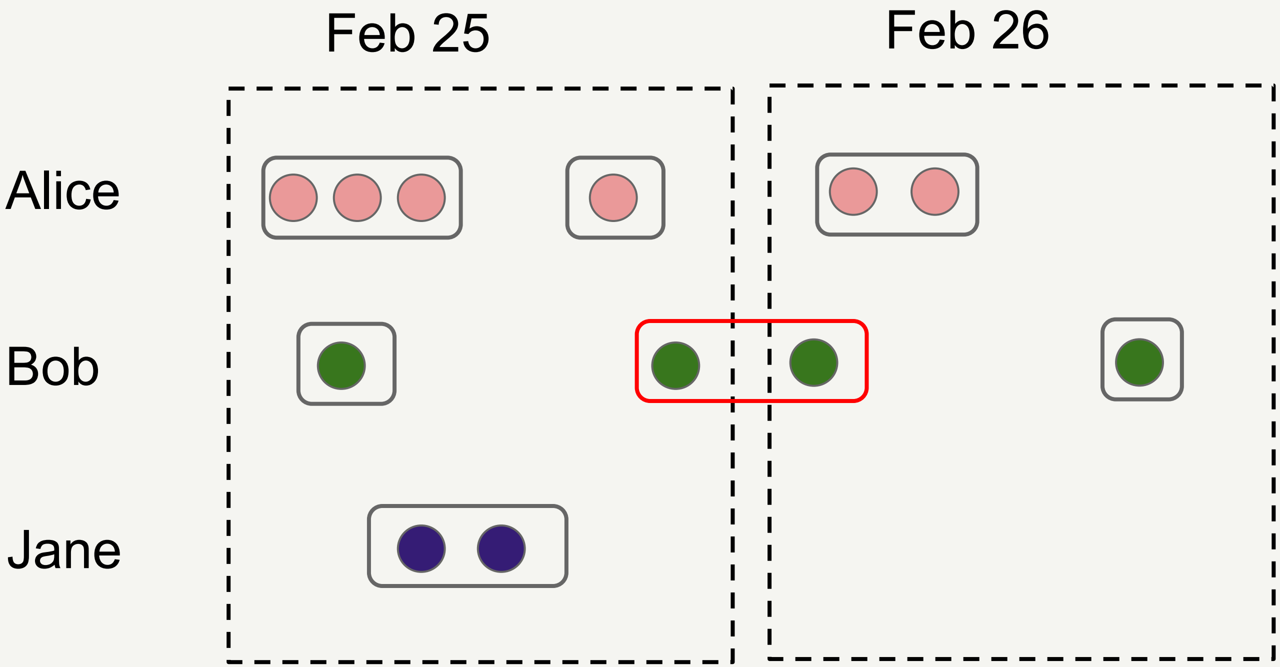
Bob

Jane

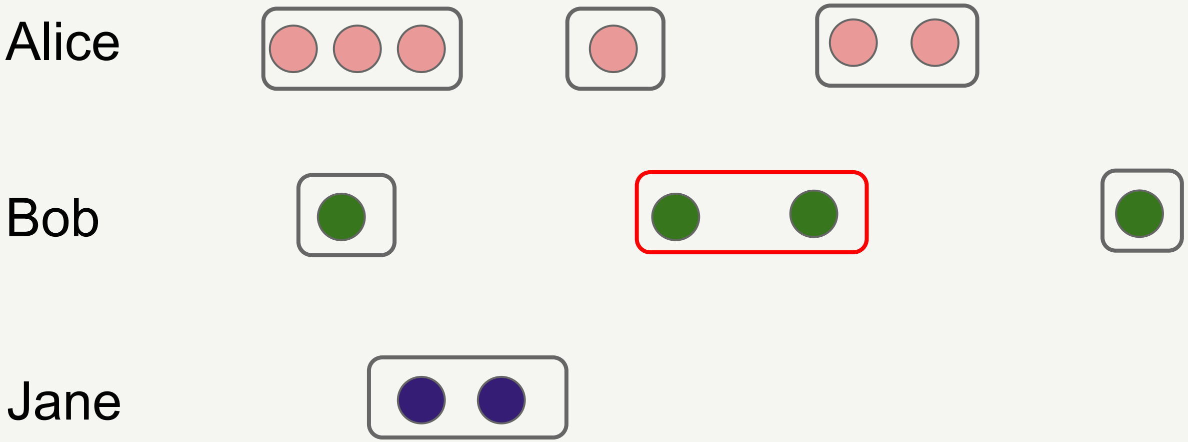
Unbounded data - batch



Unbounded data - batch



Unbounded data - stream



Alice

Bob

Jane

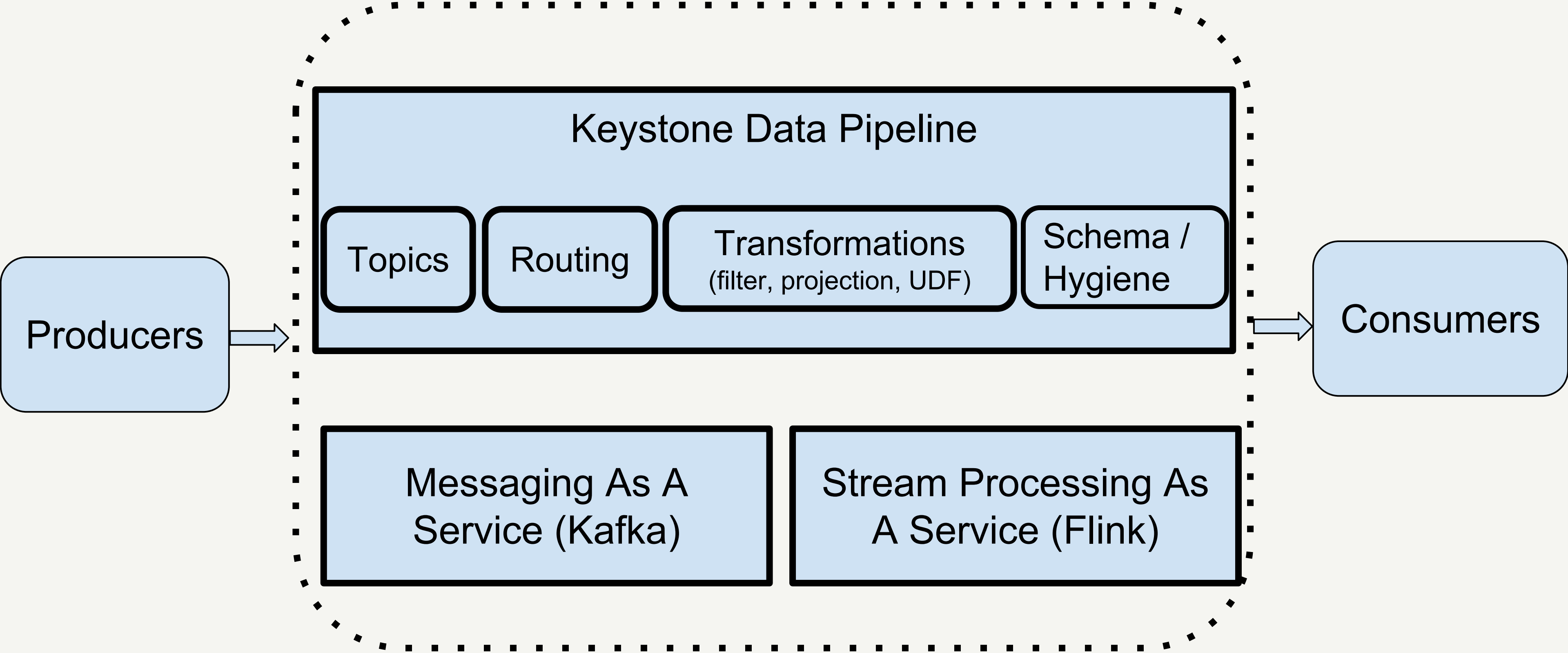
Agenda

- Introduction
- Apache Flink primer
- SPaaS Overview
- Keystone Router
- Custom Stream Processing Applications
- Backfill and Rewind

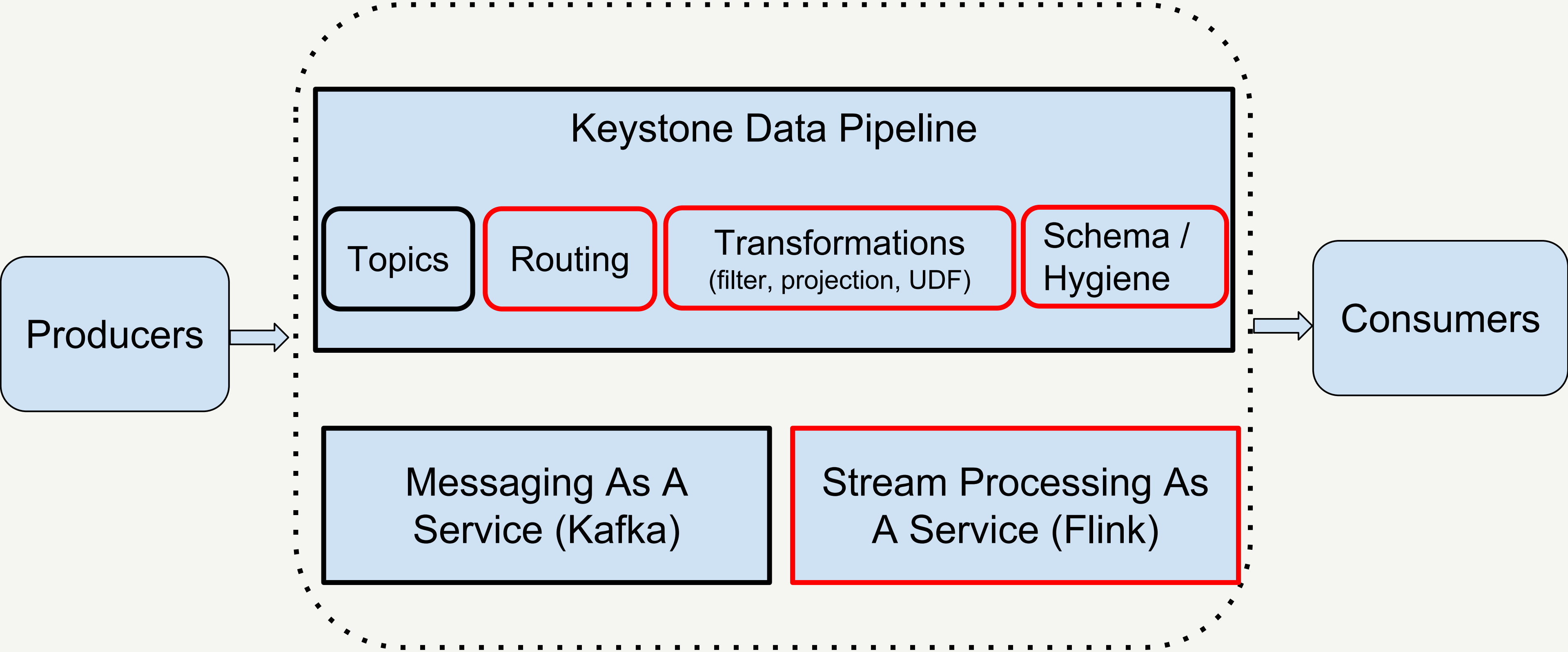
Agenda

- **Introduction**
- Apache Flink primer
- SPaaS Overview
- Keystone Router
- Custom Stream Processing Applications
- Backfill and Rewind

Real Time Data Infrastructure


Producers
Keystone Data Pipeline
Topics
Routing
Transformations
(filter, projection, UDF)
Schema / Hygiene
Messaging As A Service (Kafka)
Stream Processing As A Service (Flink)
Consumers

Stream Processing



Producers

Keystone Data Pipeline

Topics

Routing

Transformations
(filter, projection, UDF)

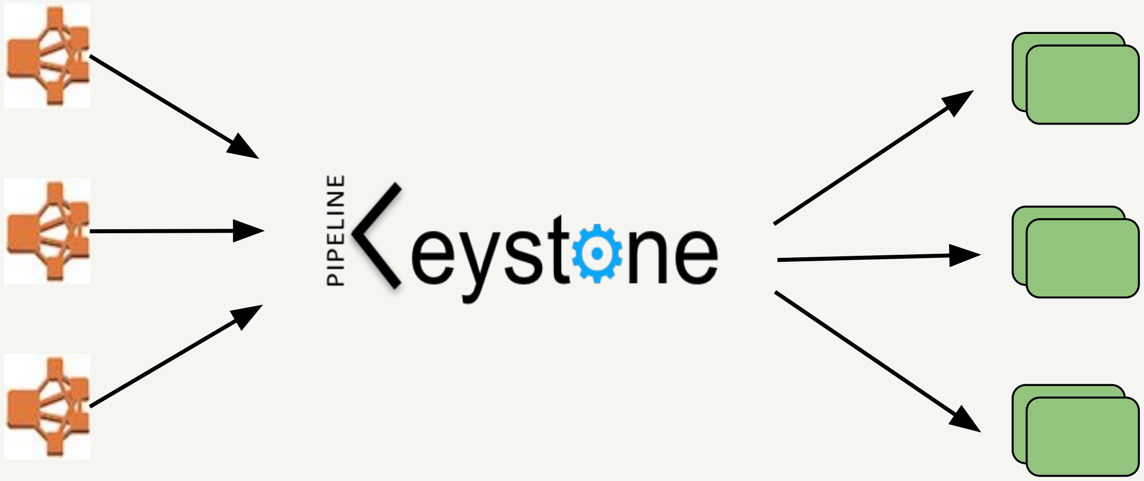
Schema /
Hygiene

Messaging As A Service (Kafka)

Stream Processing As A Service (Flink)

Consumers

highly available ingest pipelines - the backbone of a real-time data infrastructure



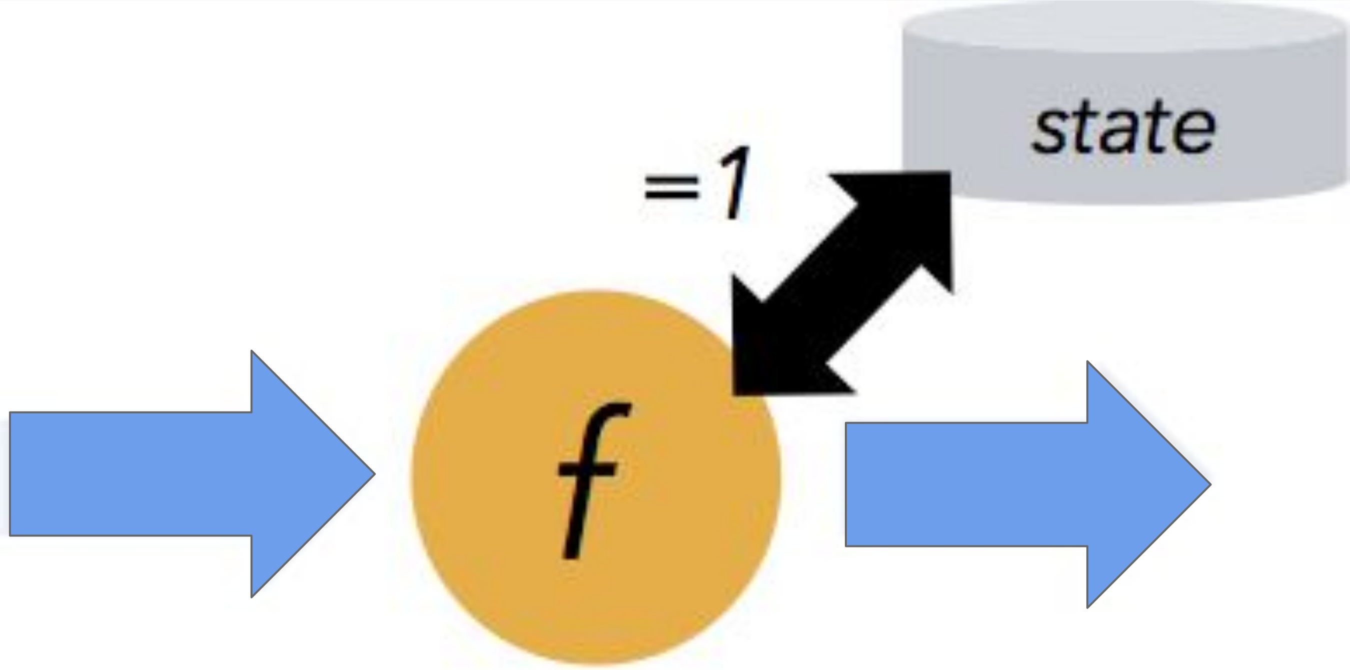
Event Producers

Sinks

Agenda

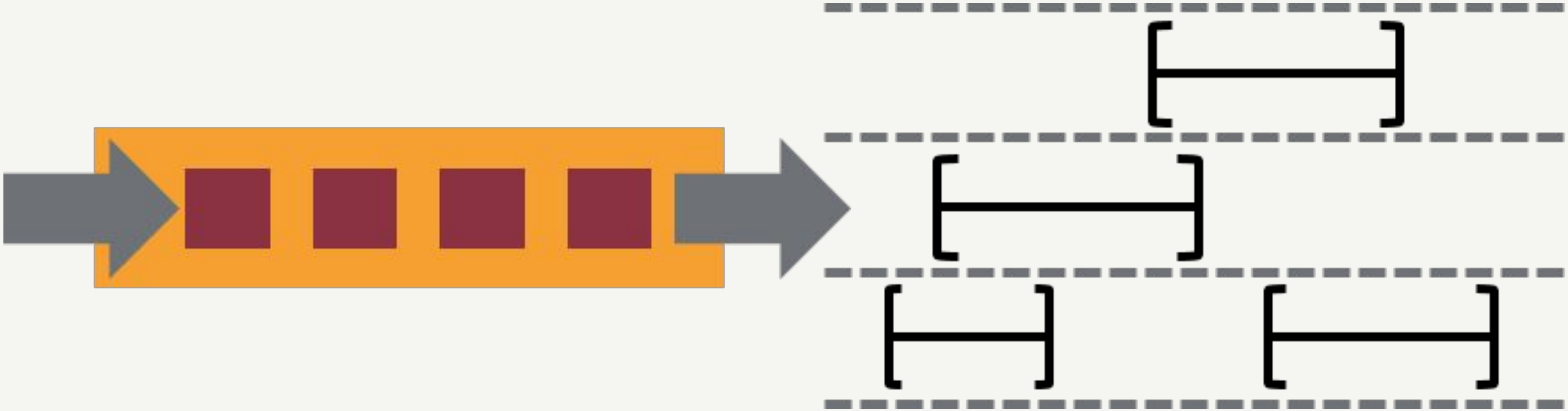
- Introduction
- **Apache Flink primer**
- SPaaS Overview
- Keystone Router
- Custom Stream Processing Applications
- Backfill and Rewind

Exact-once semantics for stateful computation



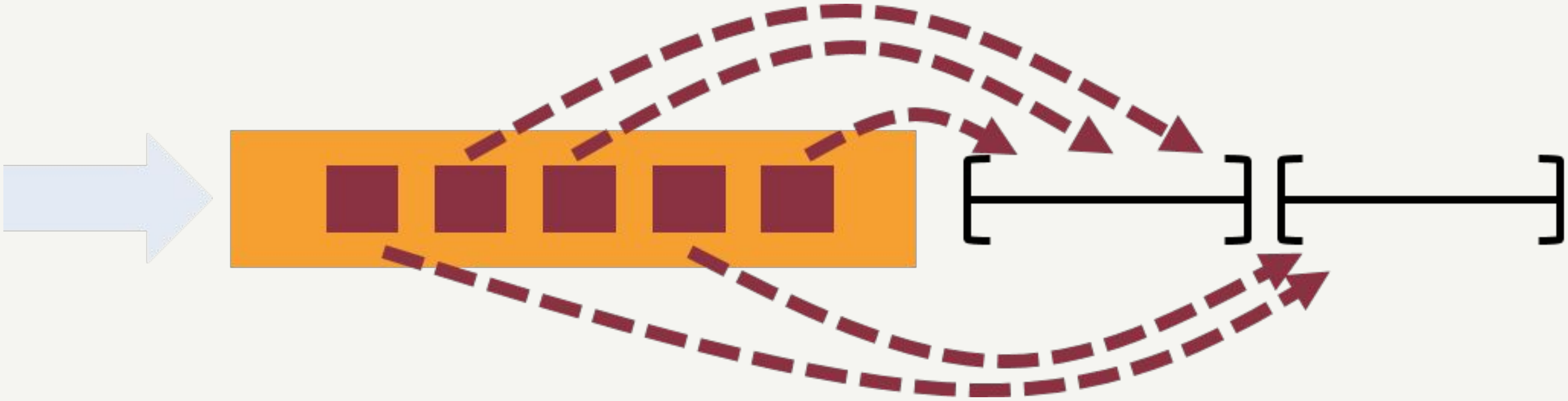
Source: http://flink.apache.org/

Flexible windowing



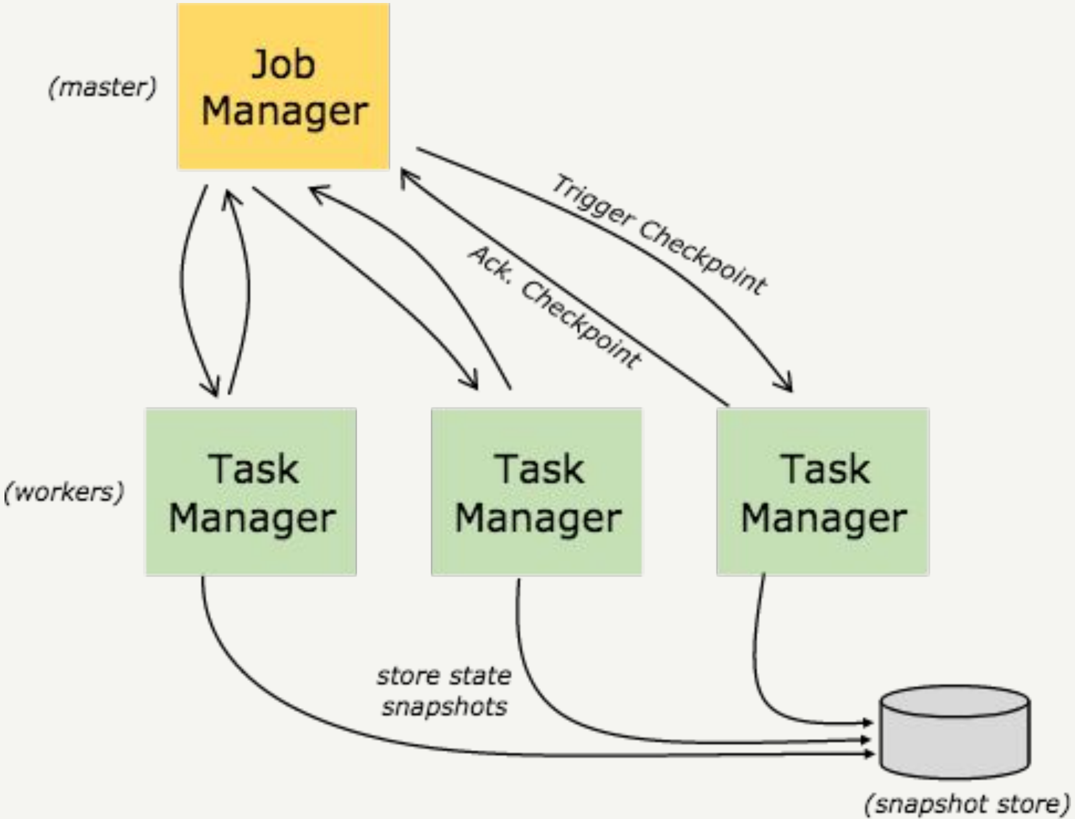
Source: http://flink.apache.org/

Event time semantics



Source: http://flink.apache.org/

State backends and checkpointing

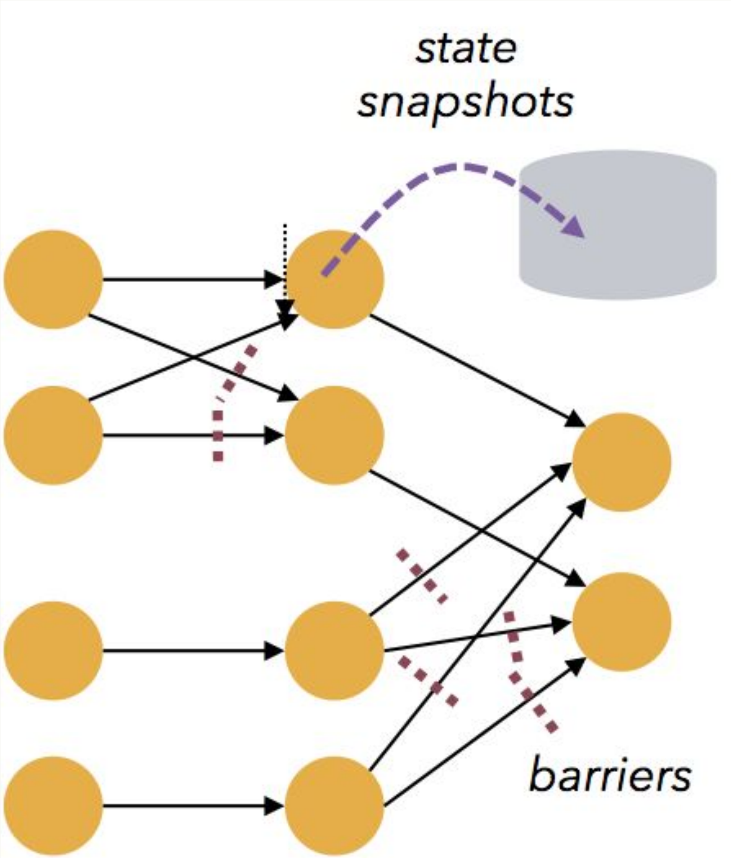


Available

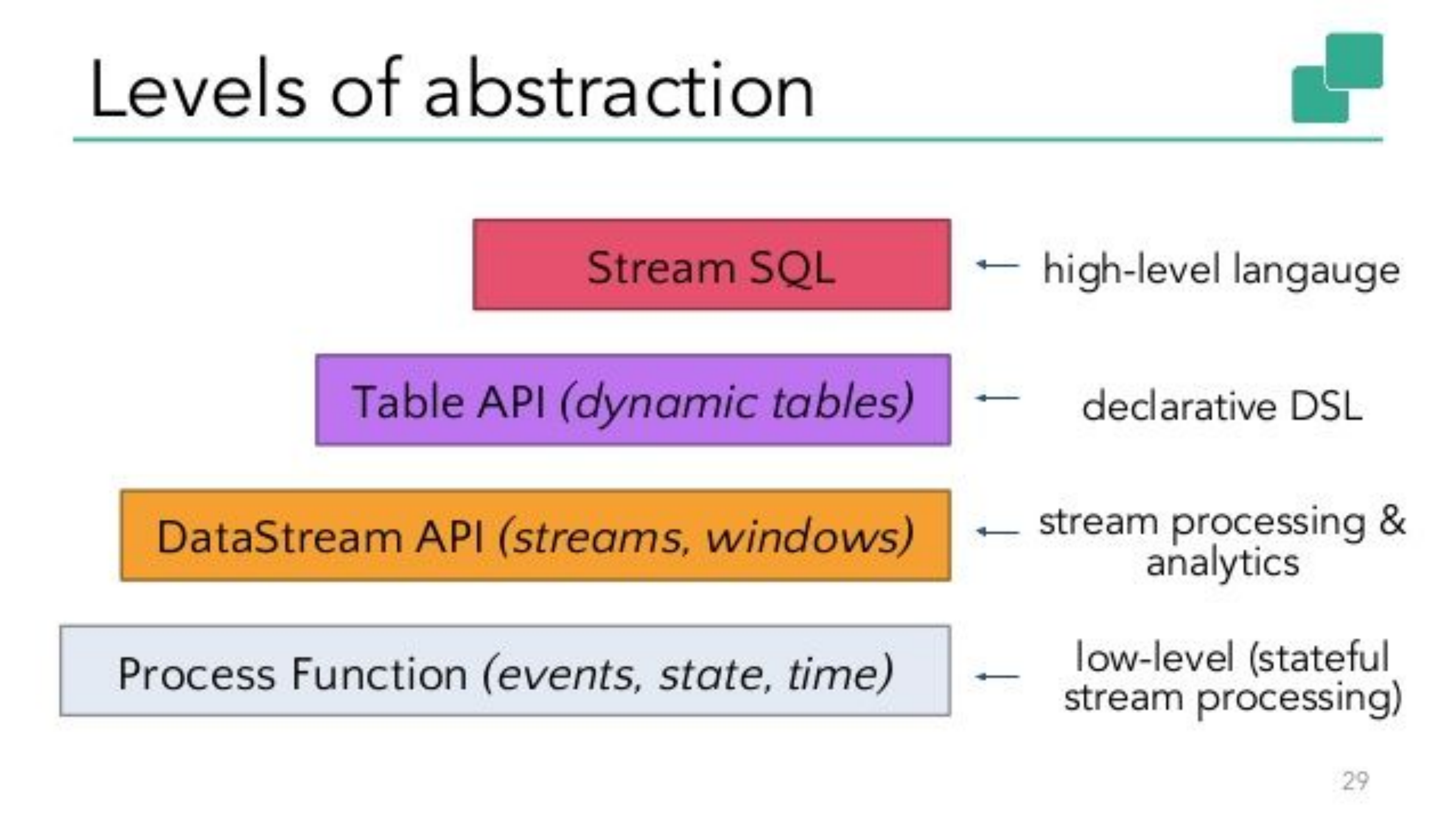
- Memory
- File system
- RocksDB (support incremental checkpoint)

Source: http://flink.apache.org/

Checkpoint is lightweight



Source: http://flink.apache.org/



Levels of abstraction

Stream SQL ← high-level langauge

Table API *(dynamic tables)* ← declarative DSL

DataStream API *(streams, windows)* ← stream processing & analytics

Process Function *(events, state, time)* ← low-level (stateful stream processing)

Source: Stephan Ewen

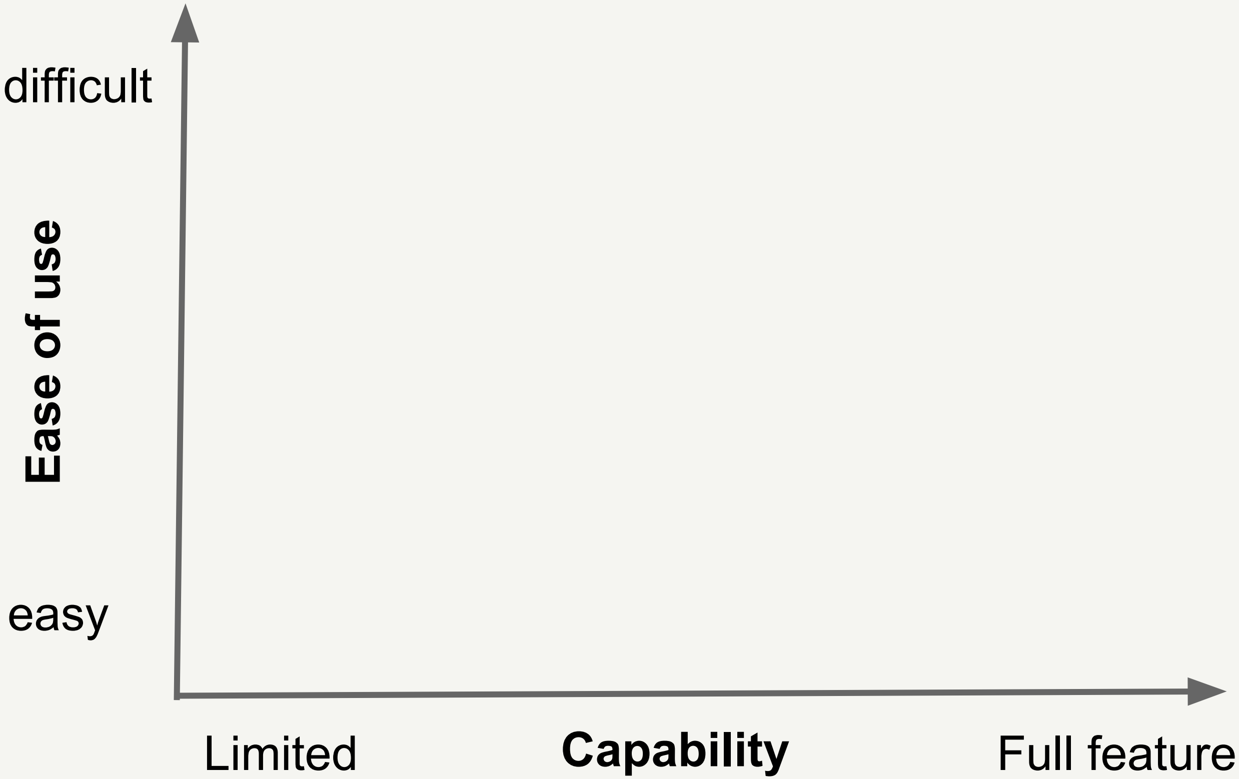
Agenda

- Introduction
- Apache Flink primer
- **SPaaS Overview**
- Keystone Router
- Custom Stream Processing Applications
- Backfill and Rewind

Offerings by complexity

- Simple **drag and drop**: filter, projection, data hygiene
  - *Available now via Keystone router*
- Medium: **SQL**, **UDF** (User Defined Function)
  - *Coming 2018*
- Advanced: **custom** stream processing applications
  - *Available now*

Ease of use v.s. capability



difficult

Ease of use

easy

Limited

Capability

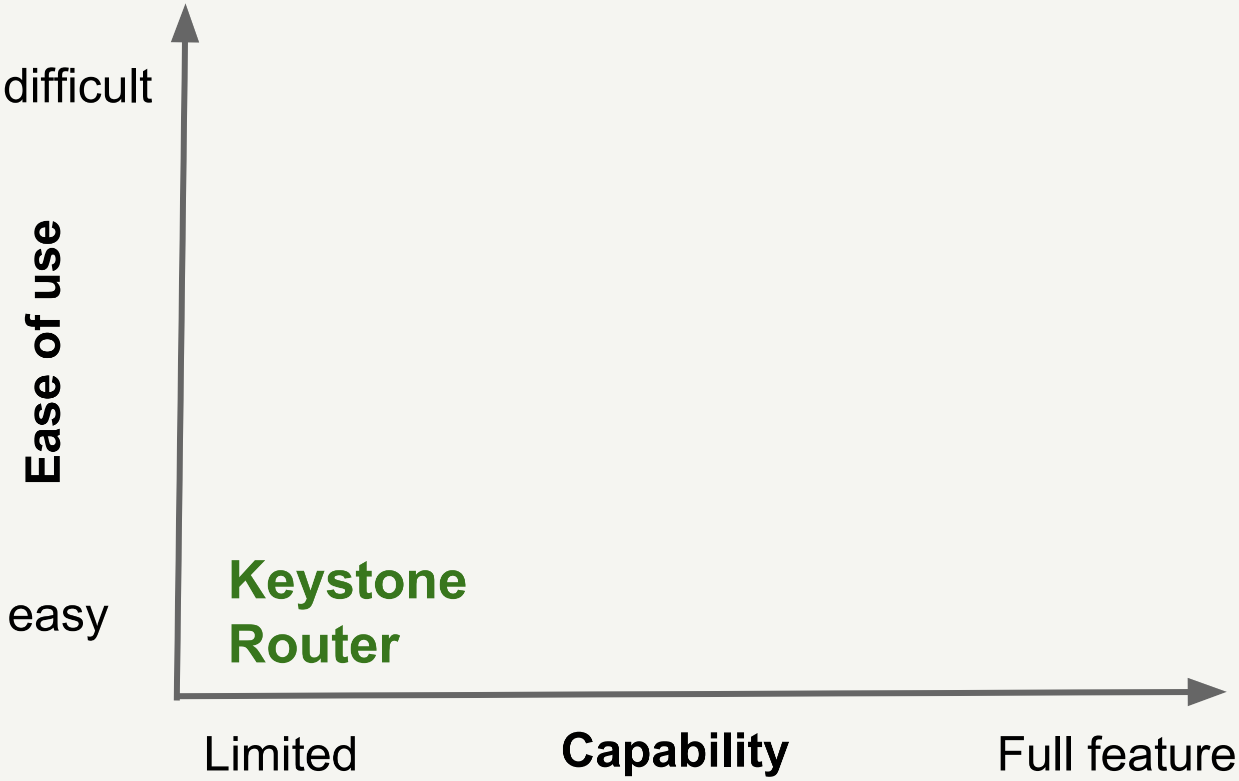
Full feature

Color legend

🟩 Available now

🟥 Coming 2018

Ease of use v.s. capability



difficult

Ease of use

easy

Keystone Router

Limited

Capability

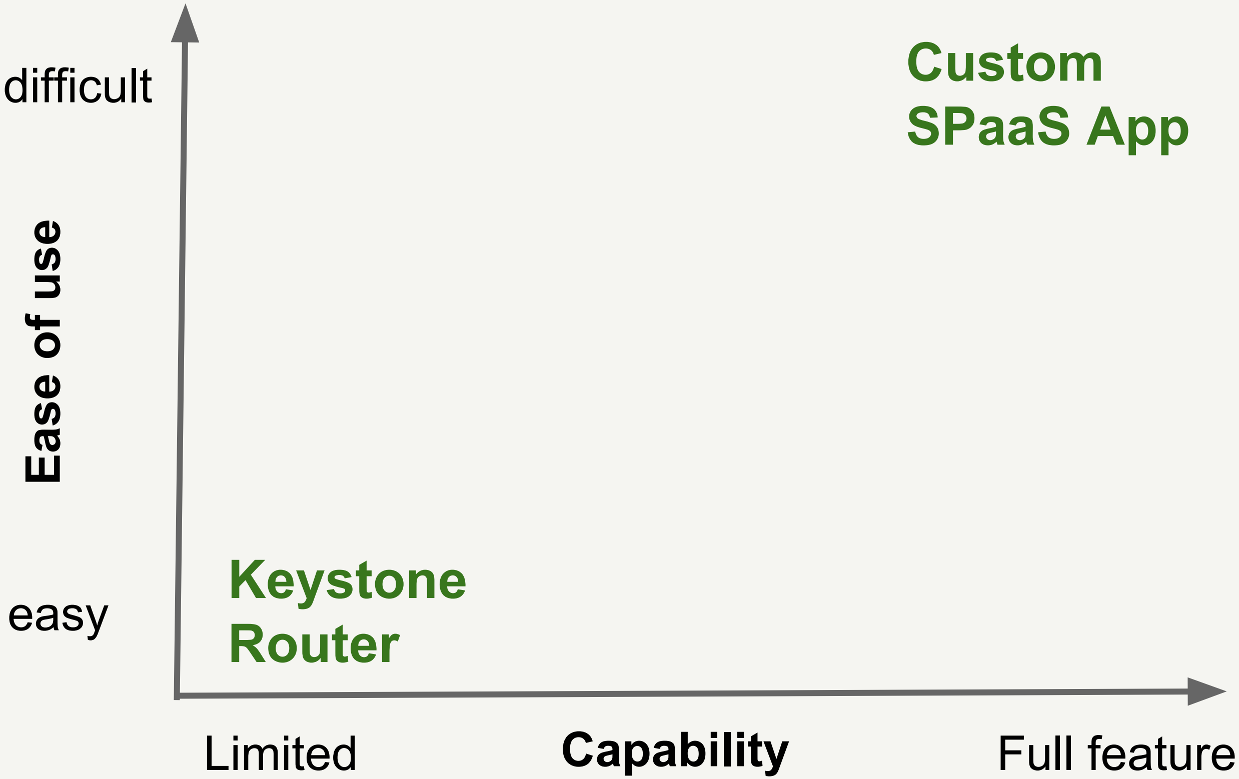
Full feature

Color legend

🟩 Available now

🟥 Coming 2018

Ease of use v.s. capability



difficult

Custom SPaaS App

Ease of use

easy

Keystone Router

Limited

Capability

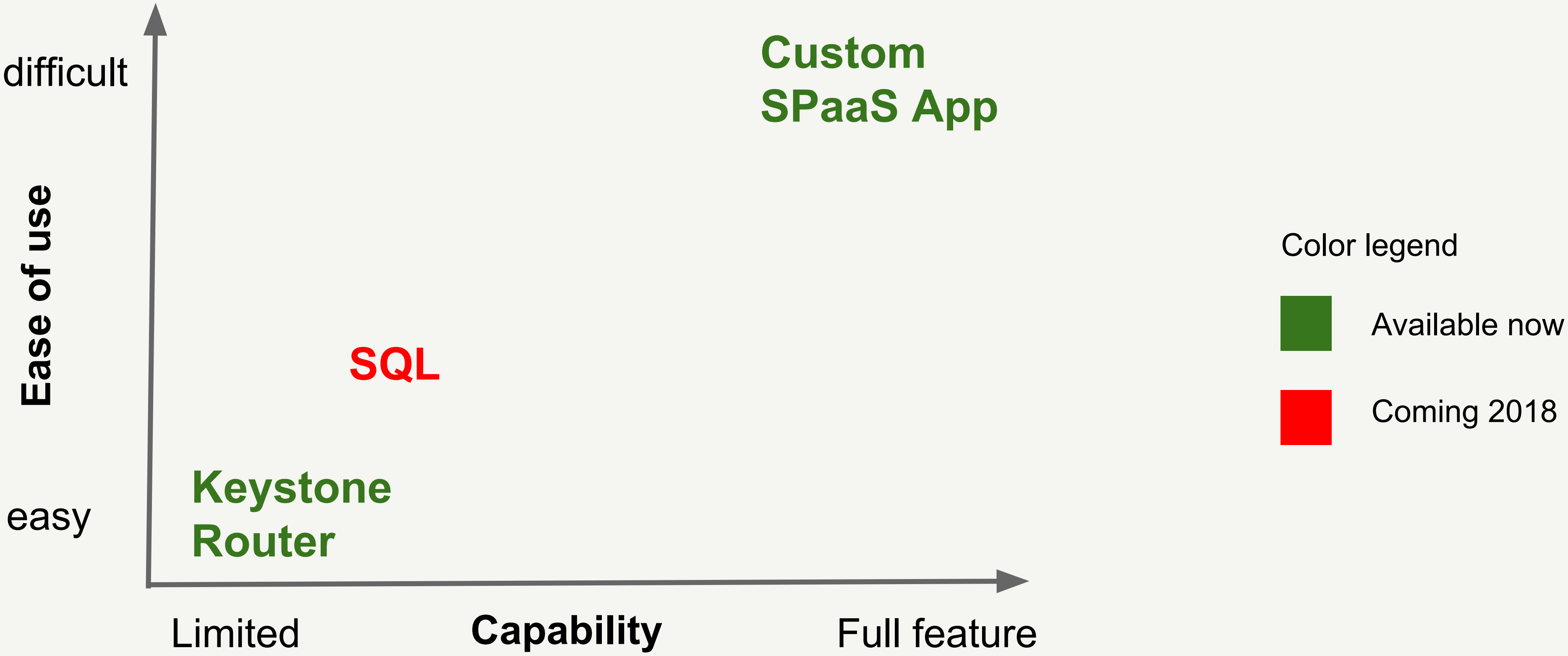
Full feature

Color legend

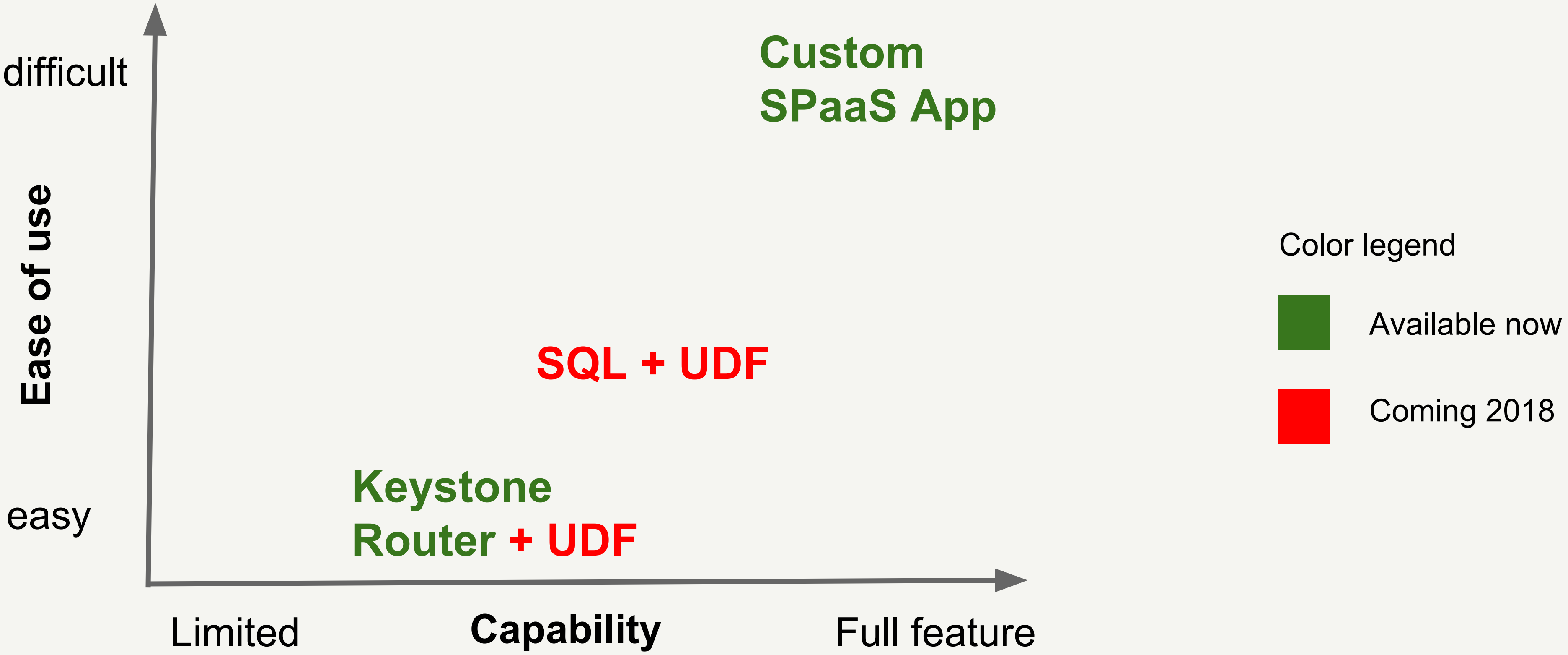
🟩 Available now

🟥 Coming 2018

Ease of use v.s. capability



Ease of use v.s. capability



Ease of use

difficult

easy

Custom SPaaS App

SQL + UDF

Keystone Router + UDF

Limited

Capability

Full feature

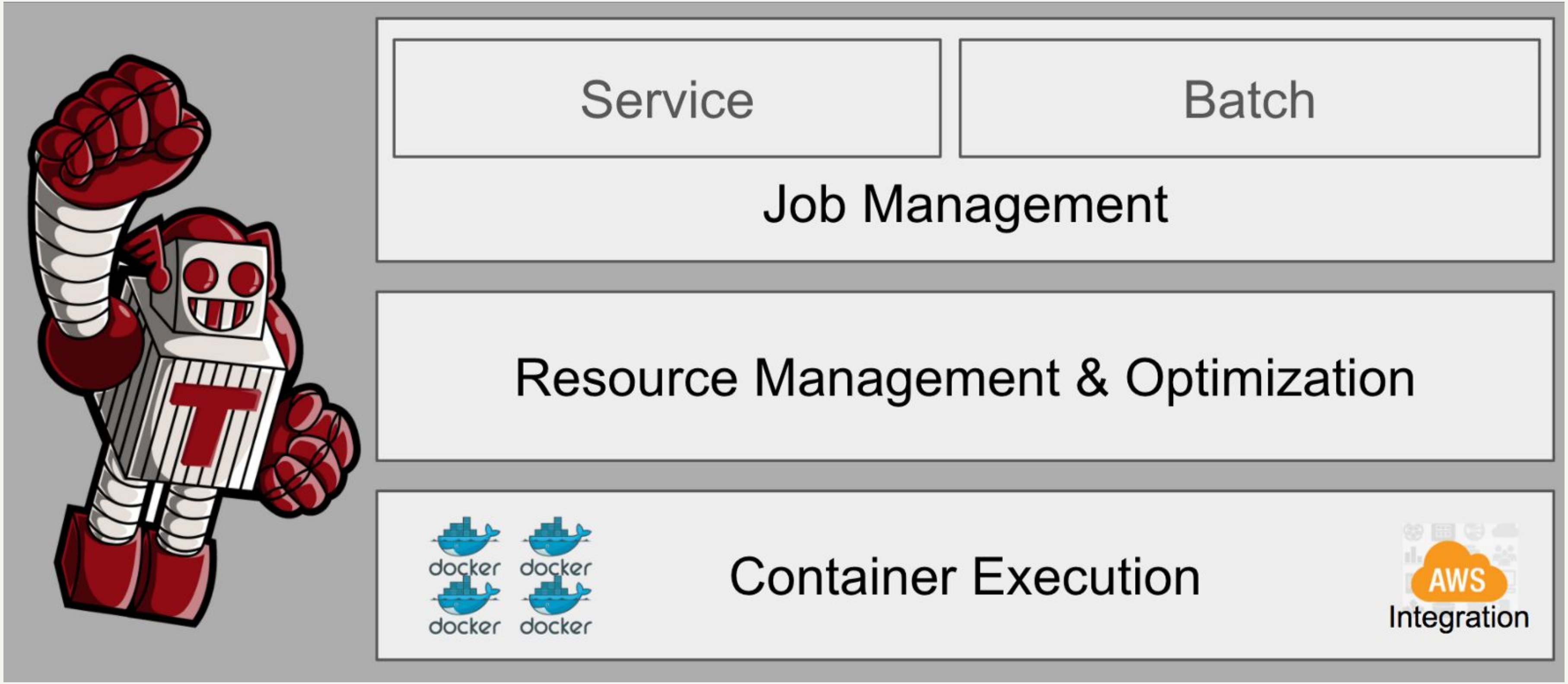
Color legend

🟩 Available now

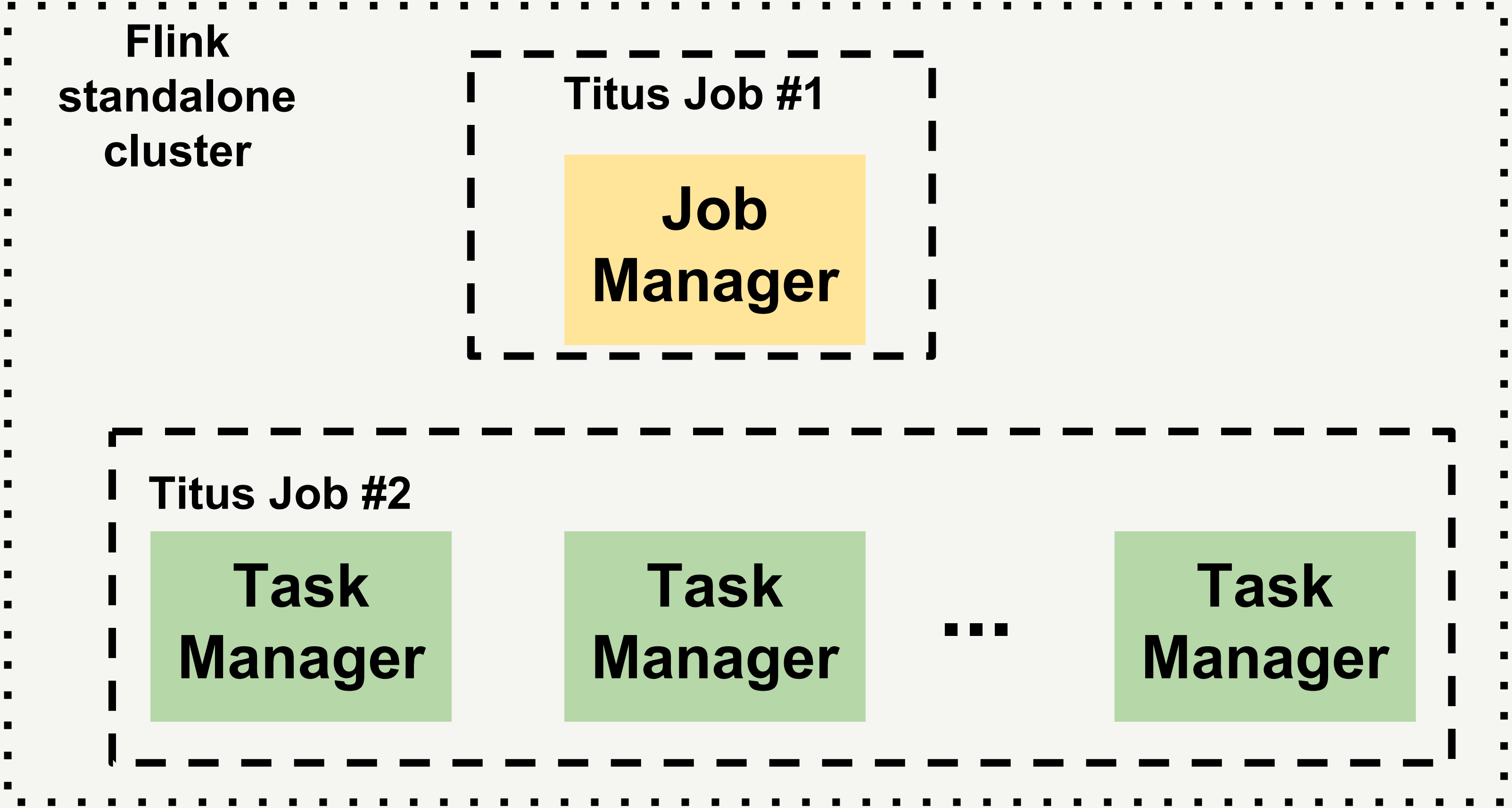
🟥 Coming 2018

SPaaS running on Titus

(Netflix's in-house container runtime)



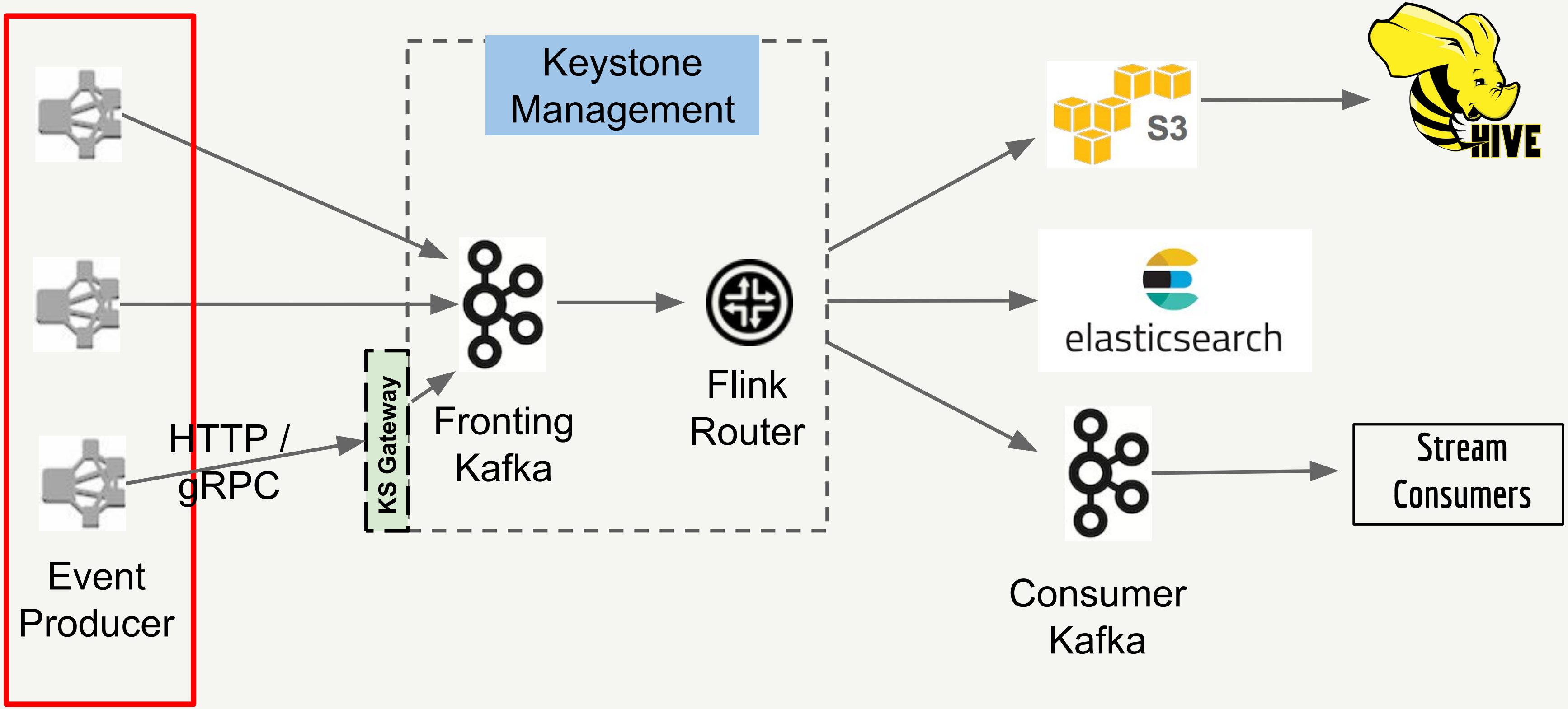
Job isolation: single job



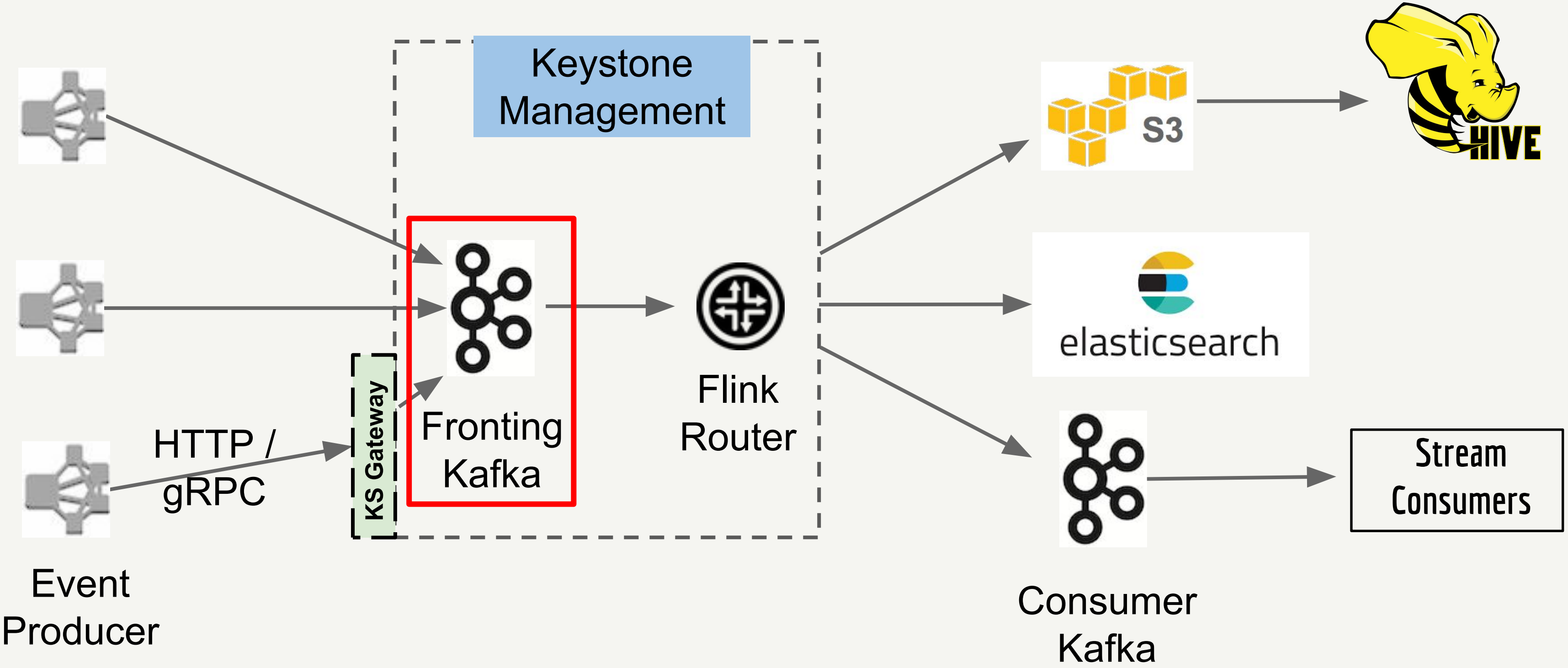
Agenda

- Introduction
- Apache Flink primer
- SPaaS Overview
- **Keystone Router**
- Custom Stream Processing Applications
- Backfill and Rewind

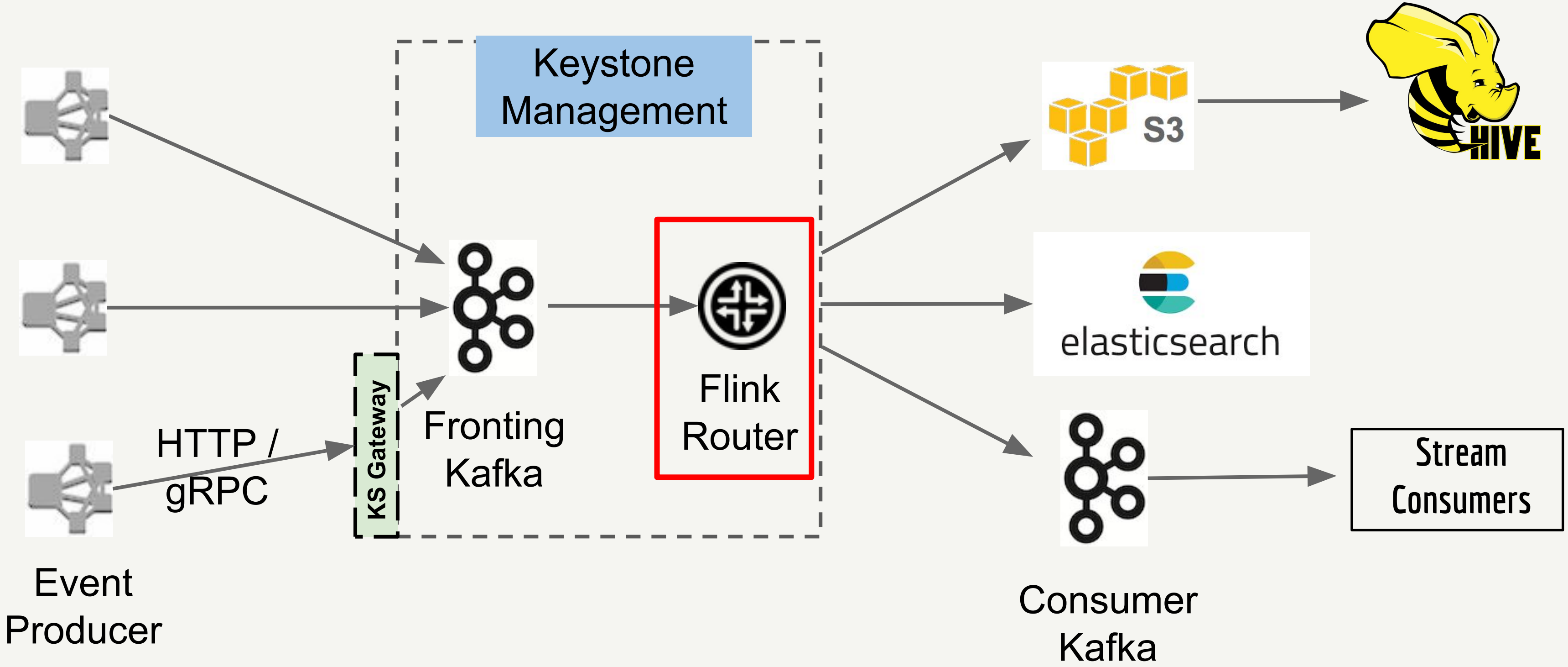
Events are published to fronting Kafka directly or via proxy



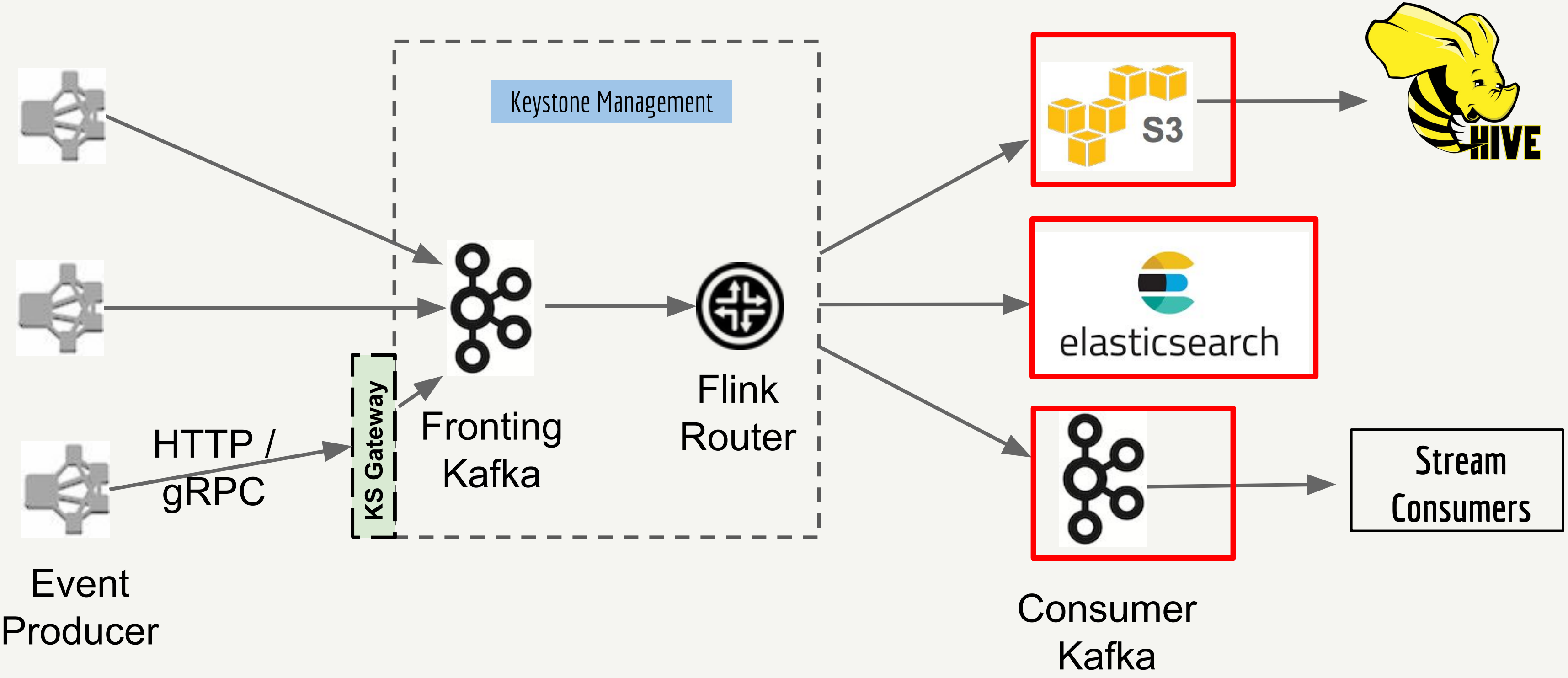
Events land up in fronting Kafka cluster



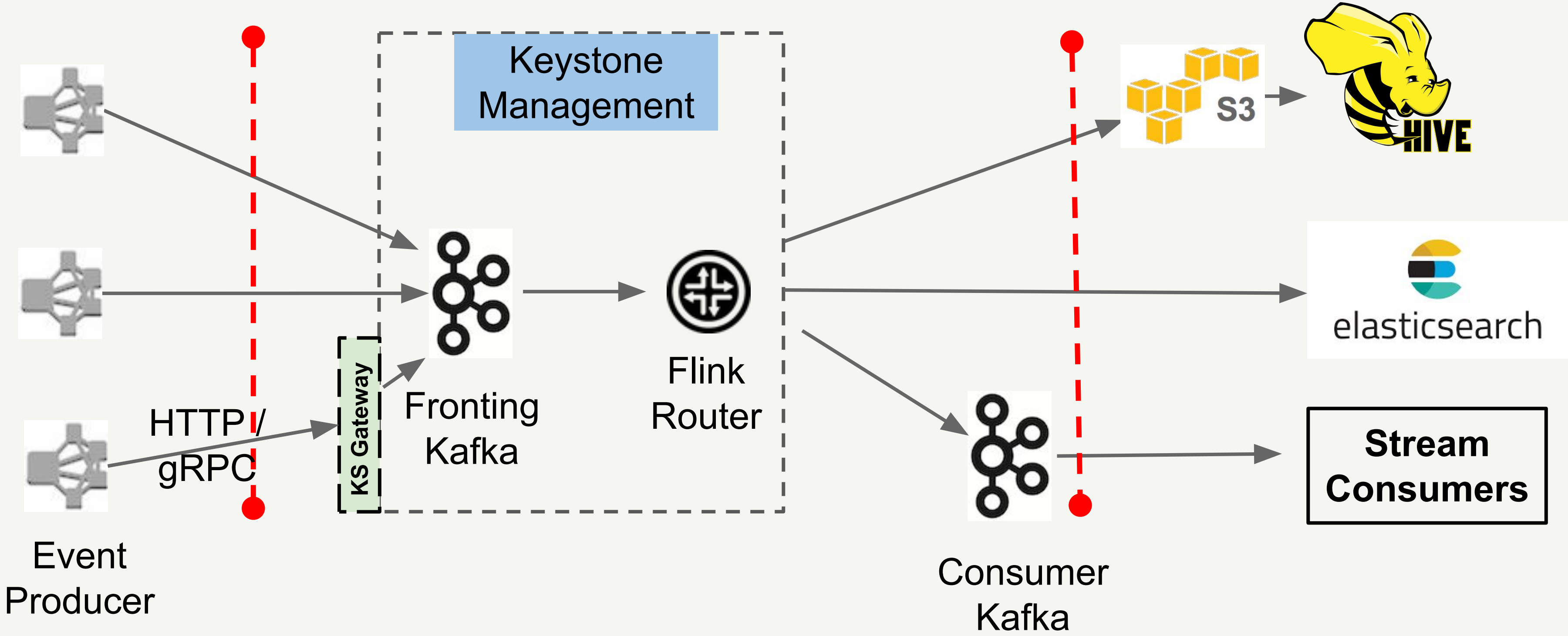
Events are polled by router, filter and projection applied



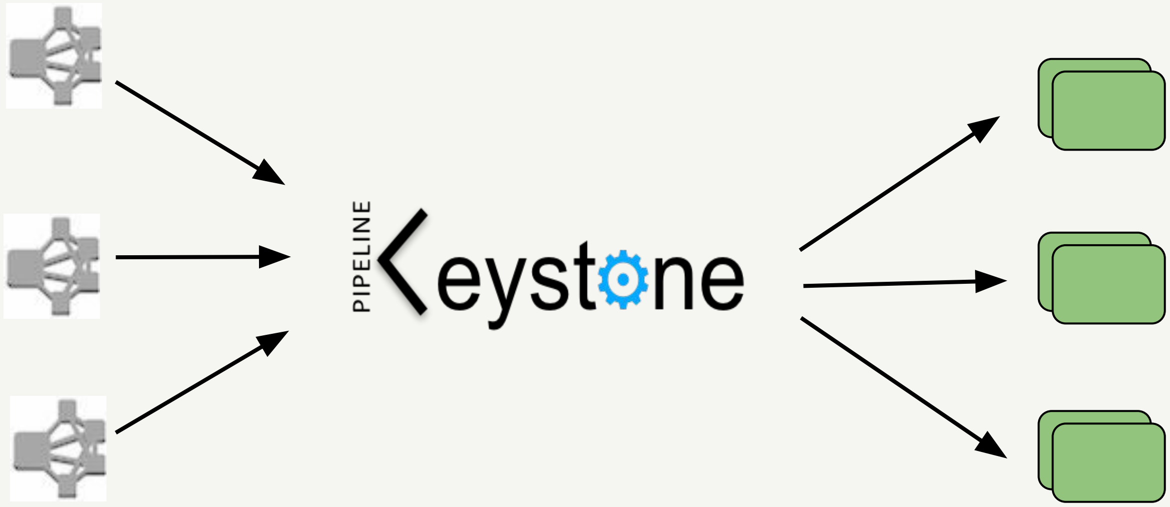
Router sends events to destination



Keystone pipeline system boundary



highly available ingest pipelines



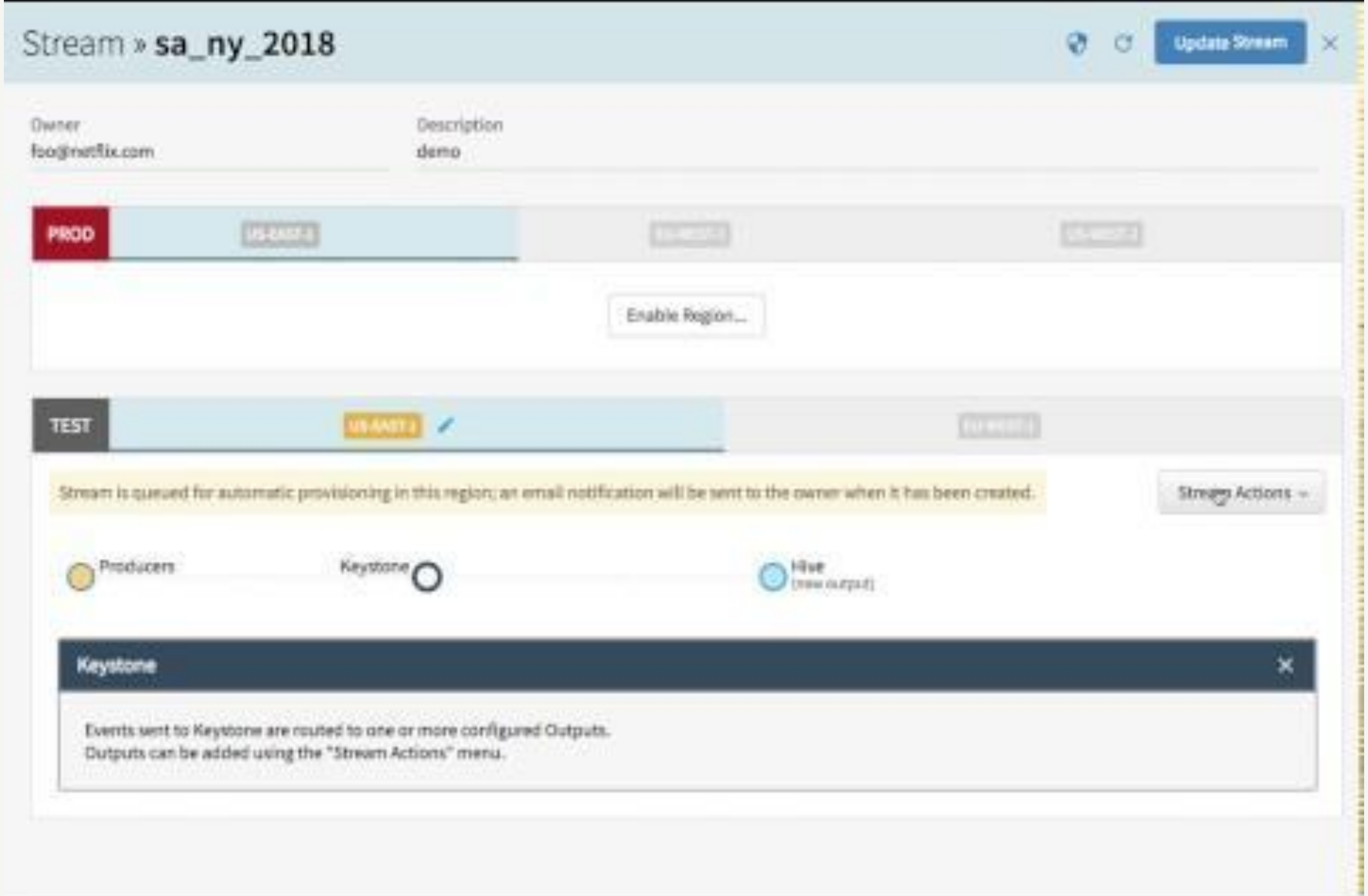
Event Producers

Sinks

Keystone scale

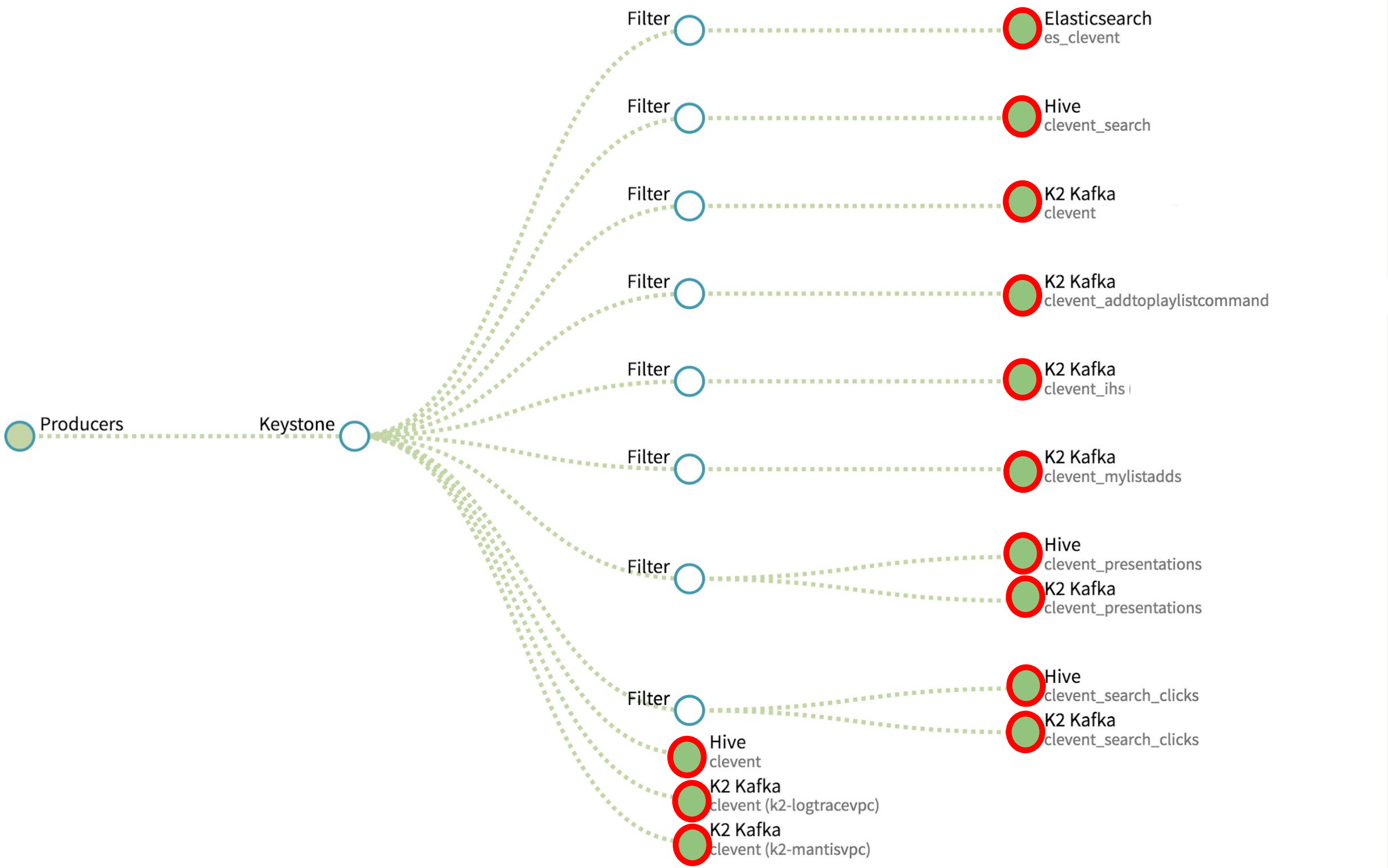
- **>1,000,000,000** unique events ingested per day
- **>99.9999%** of delivery rate

Demo: provision a data stream (mini pipeline)



Stream is queued for automatic provisioning in this region; an email notification will be sent to the owner when it has been created.

Events sent to Keystone are routed to one or more configured Outputs.
Outputs can be added using the "Stream Actions" menu.



Configure outputs

Drag-and-drop Keystone router

- Stateless and embarrassingly parallel
- ~2,000 jobs in prod
- Self serve and fully managed
- At least once delivery semantics
- Isolation

Agenda

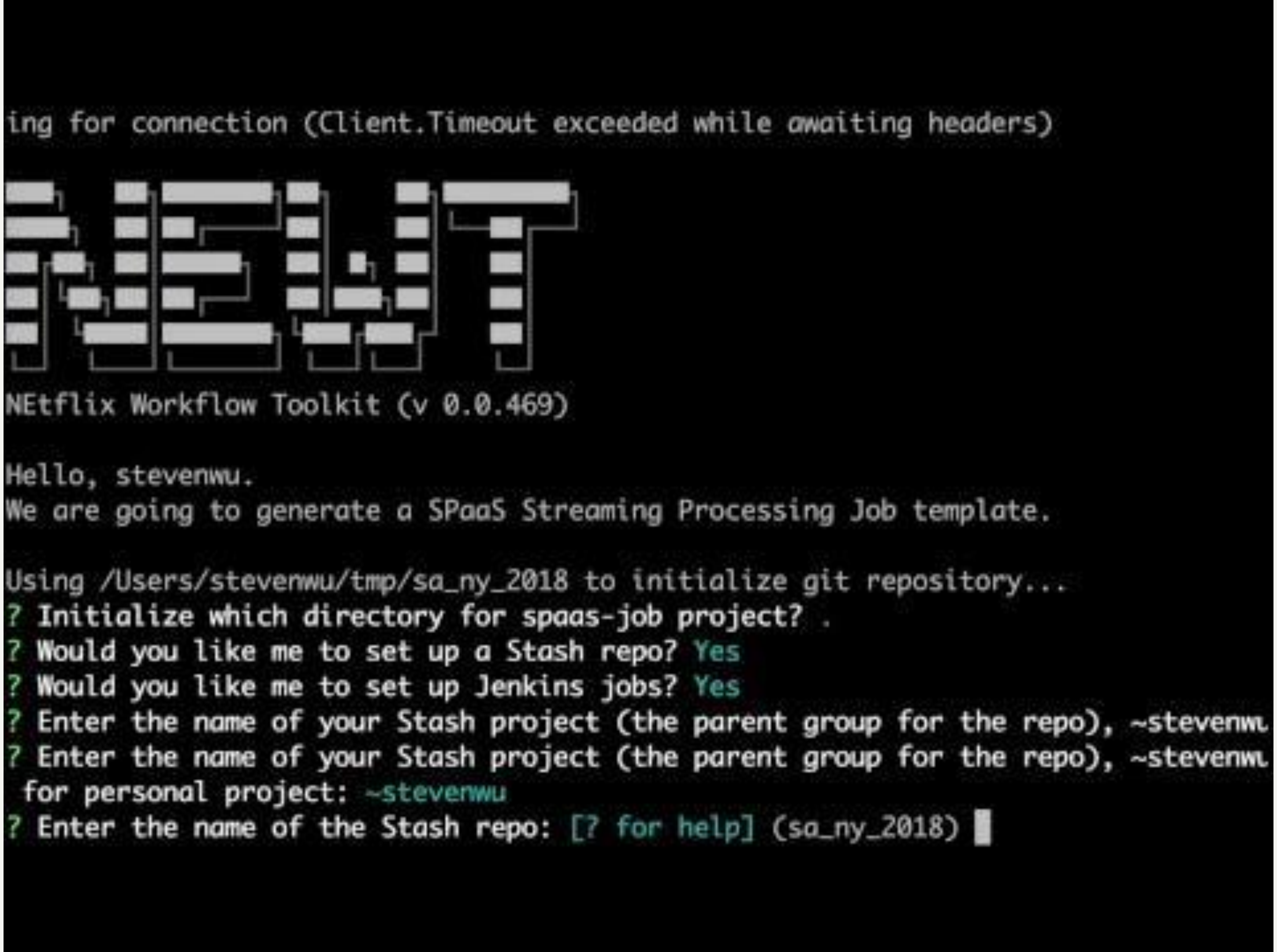
- Introduction
- Apache Flink primer
- SPaaS Overview
- Keystone Router
- **Custom Stream Processing Applications**
- Backfill and Rewind

Out-Of-The-Box Functionality

- Templates (Java / Scala)
- Build and Deployment tooling
- Connectors

- Dashboards
- Logs
- Alerts
- Titus Integration
- Capacity Management

Demo: SPaaS project bootstrap



```
ing for connection (Client.Timeout exceeded while awaiting headers)

NEtflix Workflow Toolkit (v 0.0.469)

Hello, stevenwu.
We are going to generate a SPaaS Streaming Processing Job template.

Using /Users/stevenwu/tmp/sa_ny_2018 to initialize git repository...
? Initialize which directory for spaas-job project? .
? Would you like me to set up a Stash repo? Yes
? Would you like me to set up Jenkins jobs? Yes
? Enter the name of your Stash project (the parent group for the repo), ~stevenwu
? Enter the name of your Stash project (the parent group for the repo), ~stevenwu
 for personal project: ~stevenwu
? Enter the name of the Stash repo: [? for help] (sa_ny_2018) █
```

Skeleton code

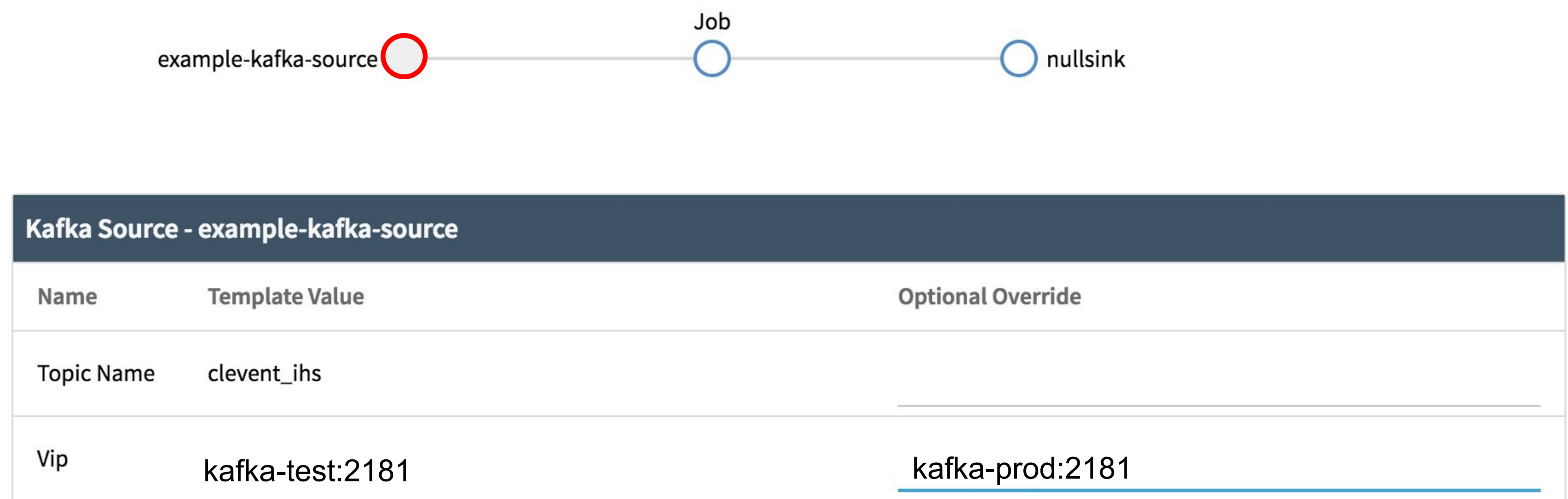
```
createSource("example-kafka-source")
        .addSink(getSink("null-sink")).name(“null-sink”);
```

Add business logic

```
createSource("example-kafka-source")
    .keyBy(<key selector>)
    .window(TumblingProcessingTimeWindows.of(Time.seconds(5)))
    .reduce(<window function>);
    .addSink(getSink(“hive-sink")).name(“hive-sink”);
```

Demo: create a new Flink job

Override source config

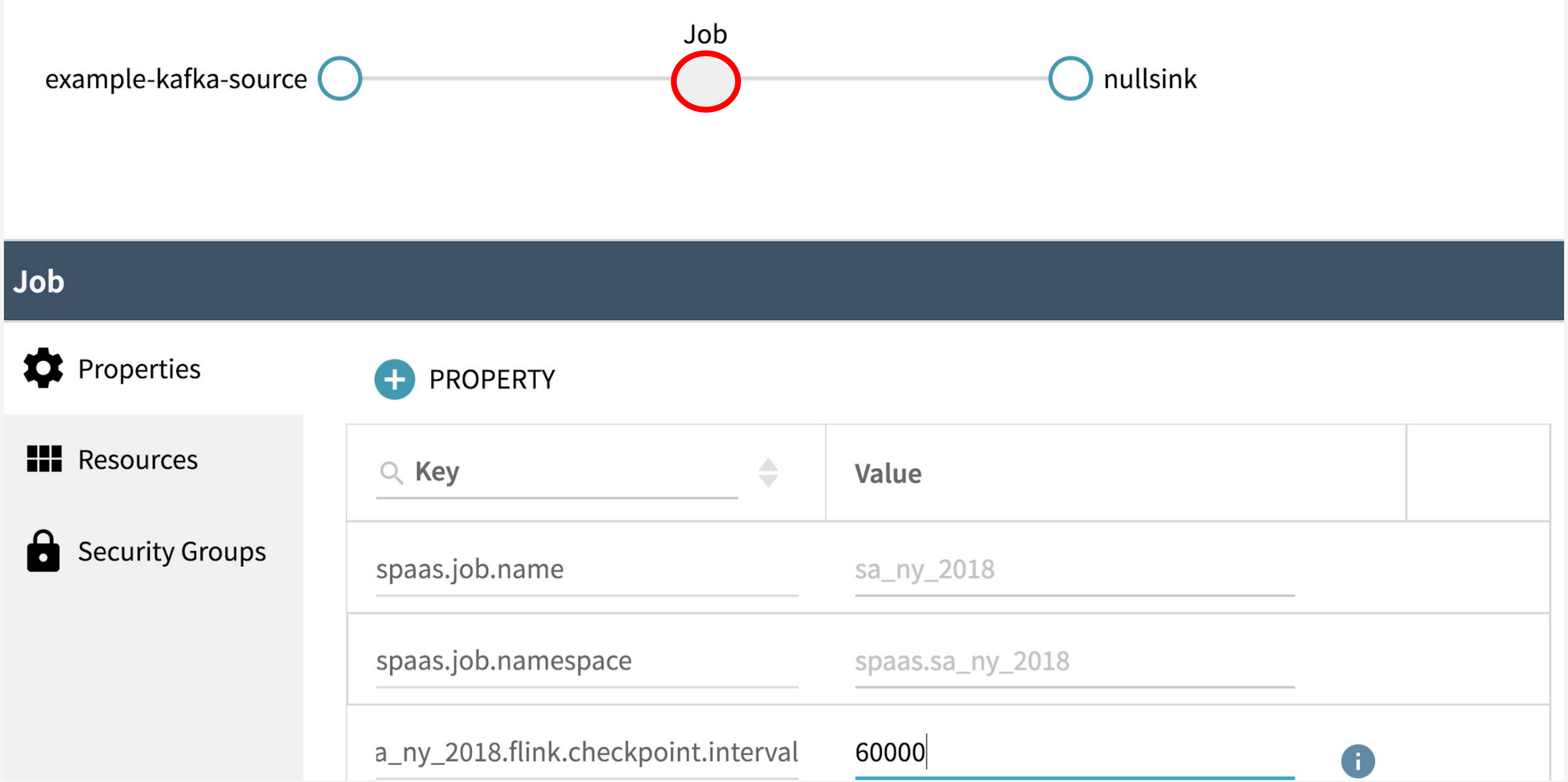


Kafka Source - example-kafka-source

Name	Template Value	Optional Override
Topic Name	clevent_ihs	
Vip	kafka-test:2181	kafka-prod:2181

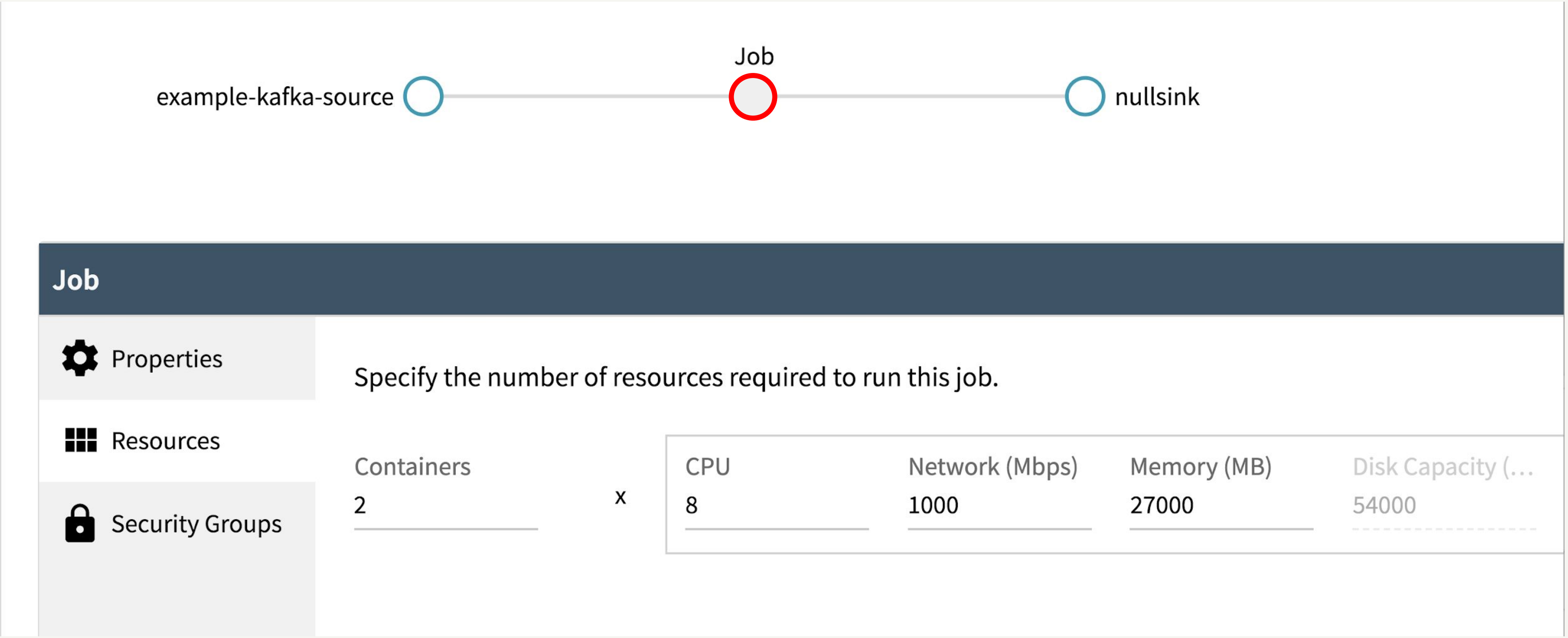
Override Kafka cluster VIP

Override job config



Key	Value
spaas.job.name	sa_ny_2018
spaas.job.namespace	spaas.sa_ny_2018
a_ny_2018.flink.checkpoint.interval	60000

Configure resources

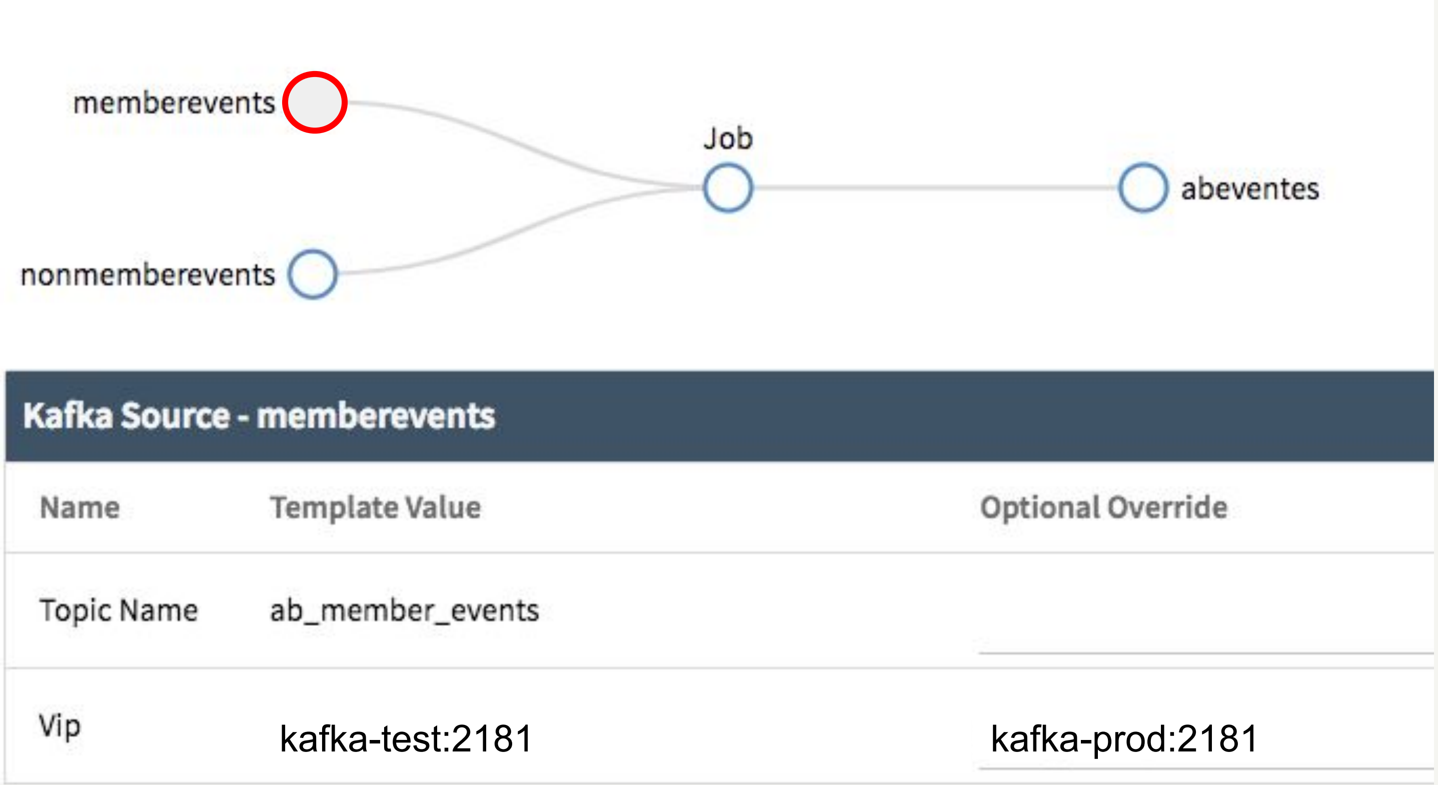


Job

Specify the number of resources required to run this job.

Containers		CPU	Network (Mbps)	Memory (MB)	Disk Capacity (...
2	x	8	1000	27000	54000

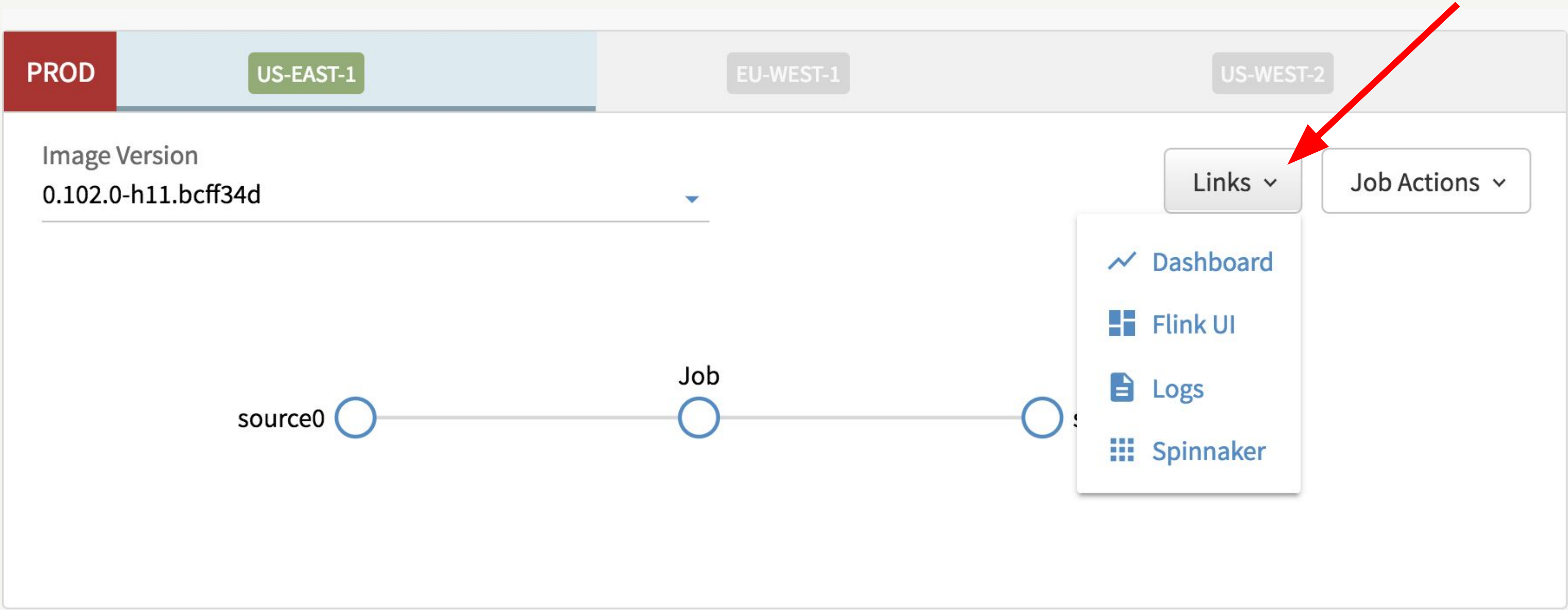
Configure multiple sources or sinks



Kafka Source - memberevents

Name	Template Value	Optional Override
Topic Name	ab_member_events	
Vip	kafka-test:2181	kafka-prod:2181

Deep links



PROD
US-EAST-1
EU-WEST-1
US-WEST-2
Image Version
0.102.0-h11.bcff34d
Links
Job Actions
Dashboard
Flink UI
Logs
Spinnaker
Job
source0

Duplo blocks

- Filter
- Projector
- Data Hygiene
- Connectors

Supported Source and Sink Connectors

Sources

- Kafka
- Hive

Sinks

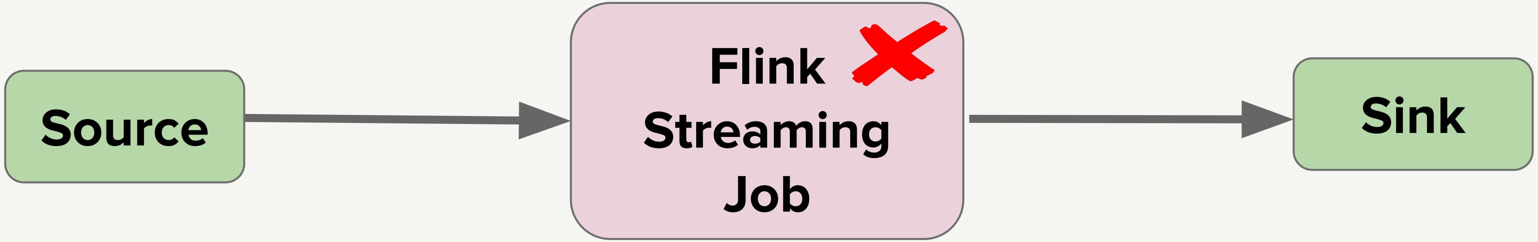
- Elasticsearch
- Kafka
- Hive
- Keystone

Agenda

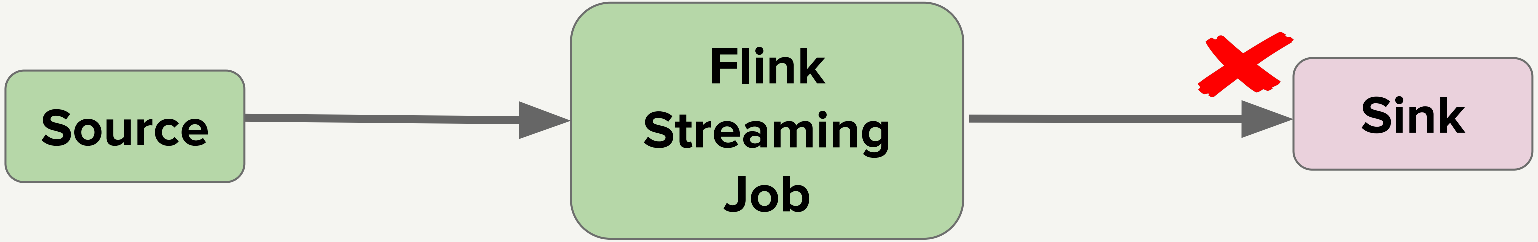
- Introduction
- Apache Flink primer
- SPaaS Overview
- Keystone Router
- Custom Stream Processing Applications
- **Backfill and Rewind**

Things can go wrong

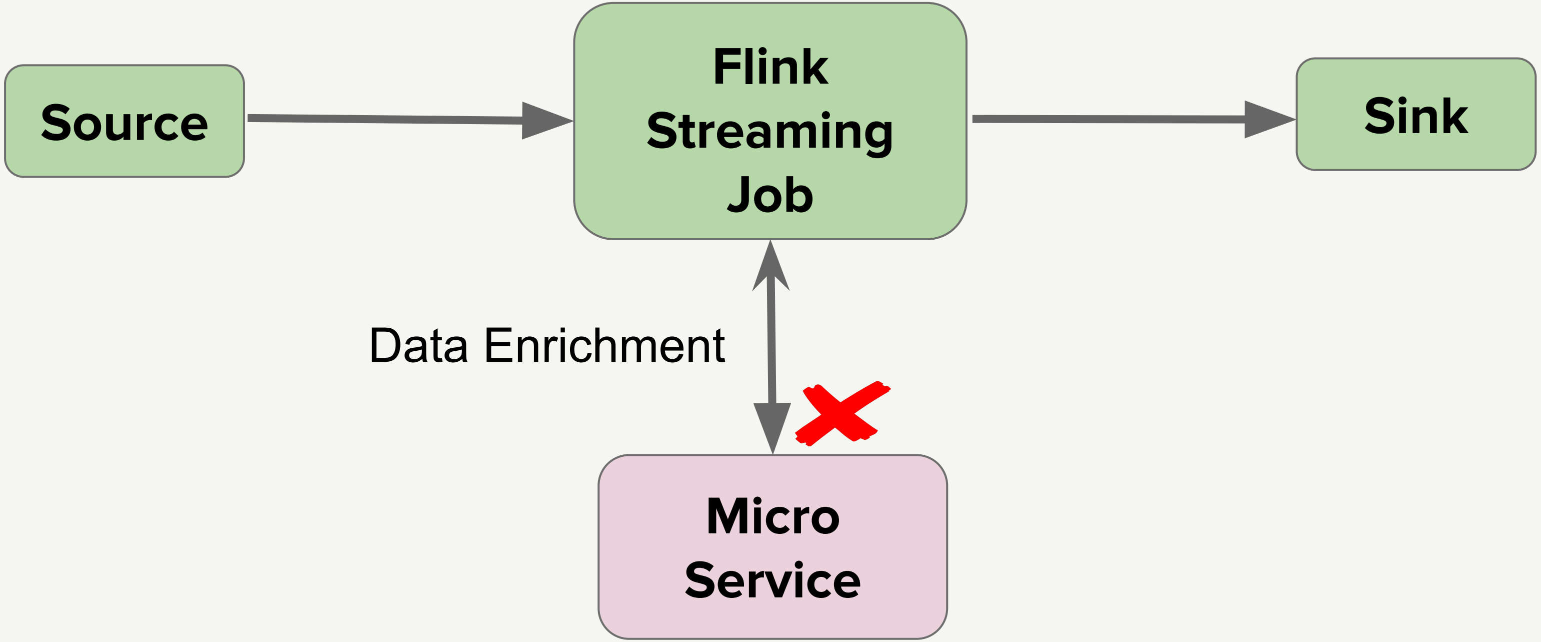
Application bug



Sink failure



Dependency service failure



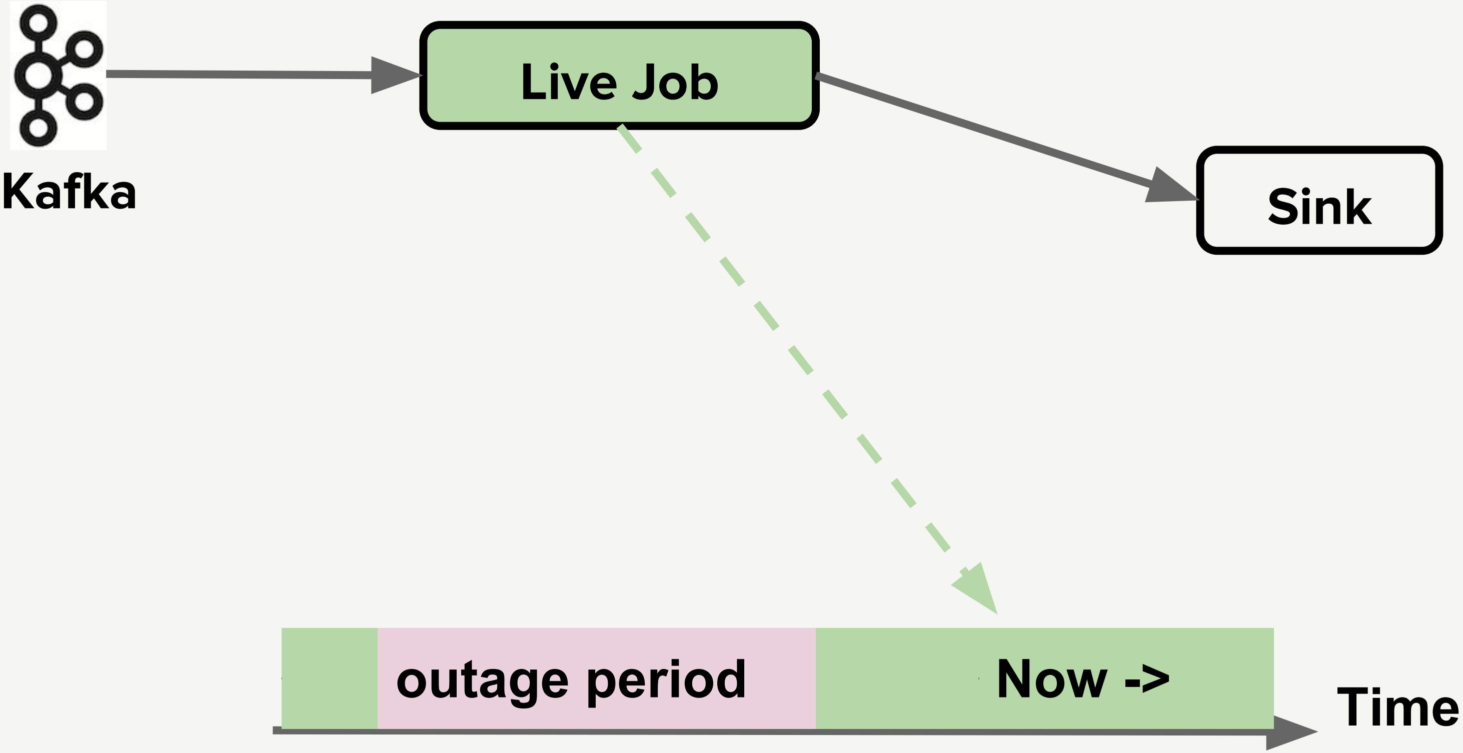
How to recover

- Backfill *(available now)*
- Rewind Flink job *(coming soon)*

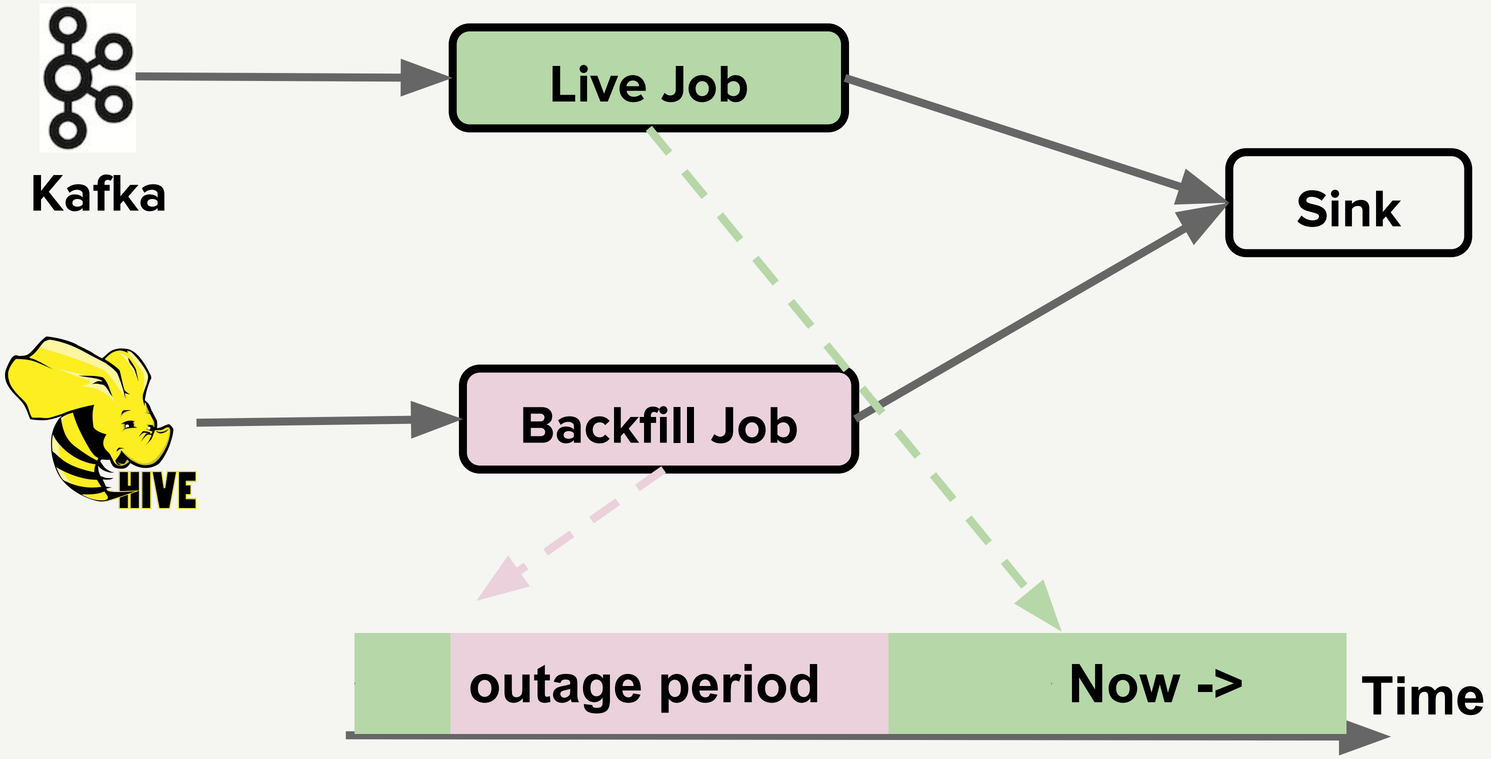
How to recover

- Backfill
- Rewind Flink job

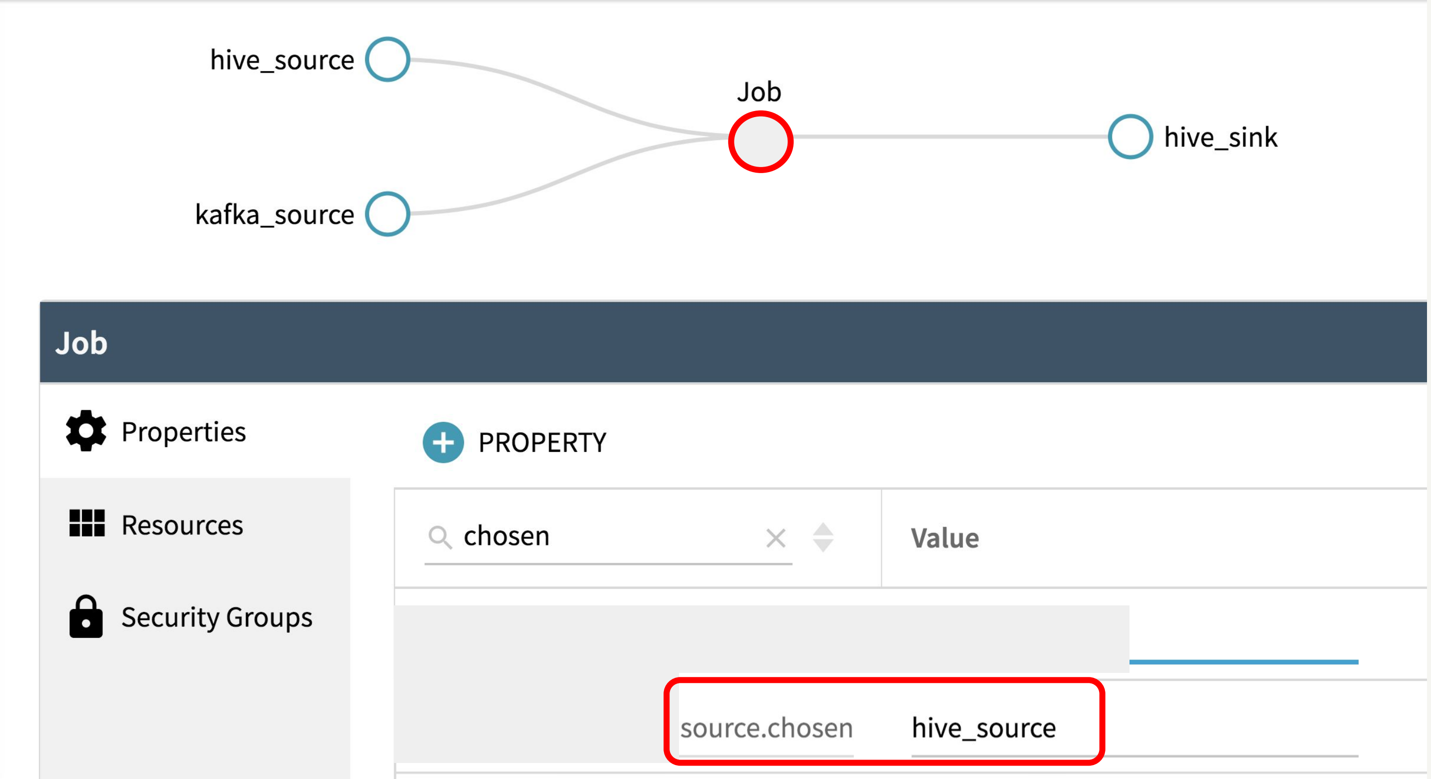
Live job continues



Hive as backfill source



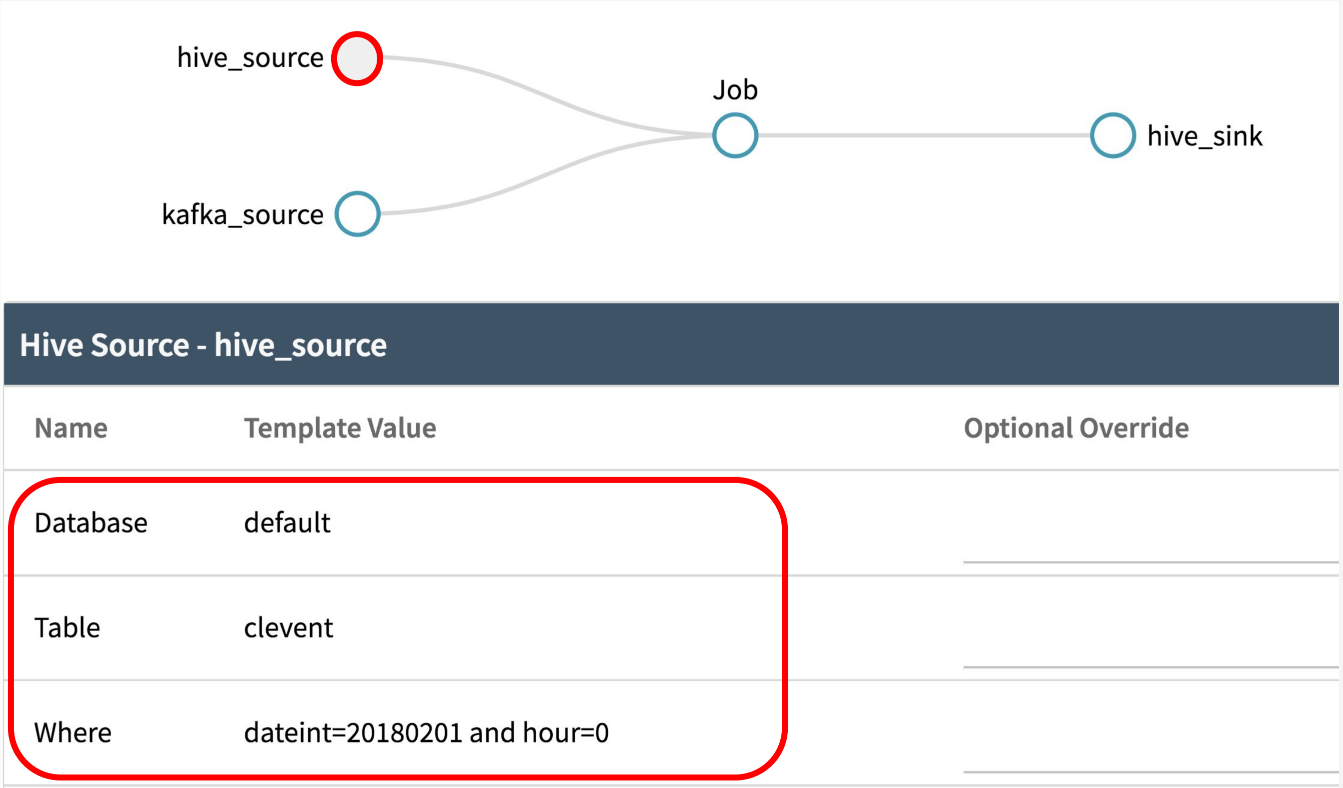
Choose Hive source



chosen

source.chosen hive_source

Configure Hive source



Hive Source - hive_source

Name	Template Value	Optional Override
Database	default	
Table	clevent	
Where	dateint=20180201 and hour=0	

Not a lambda architecture

- Single streaming code base
- Just switch source from Kafka to Hive

Hive backfill probably not for stateful jobs

- Warm-up issue
- Ordering issue

Hive backfill probably not for stateful jobs

- Warm-up issue
- Ordering issue

Stateful stream processor

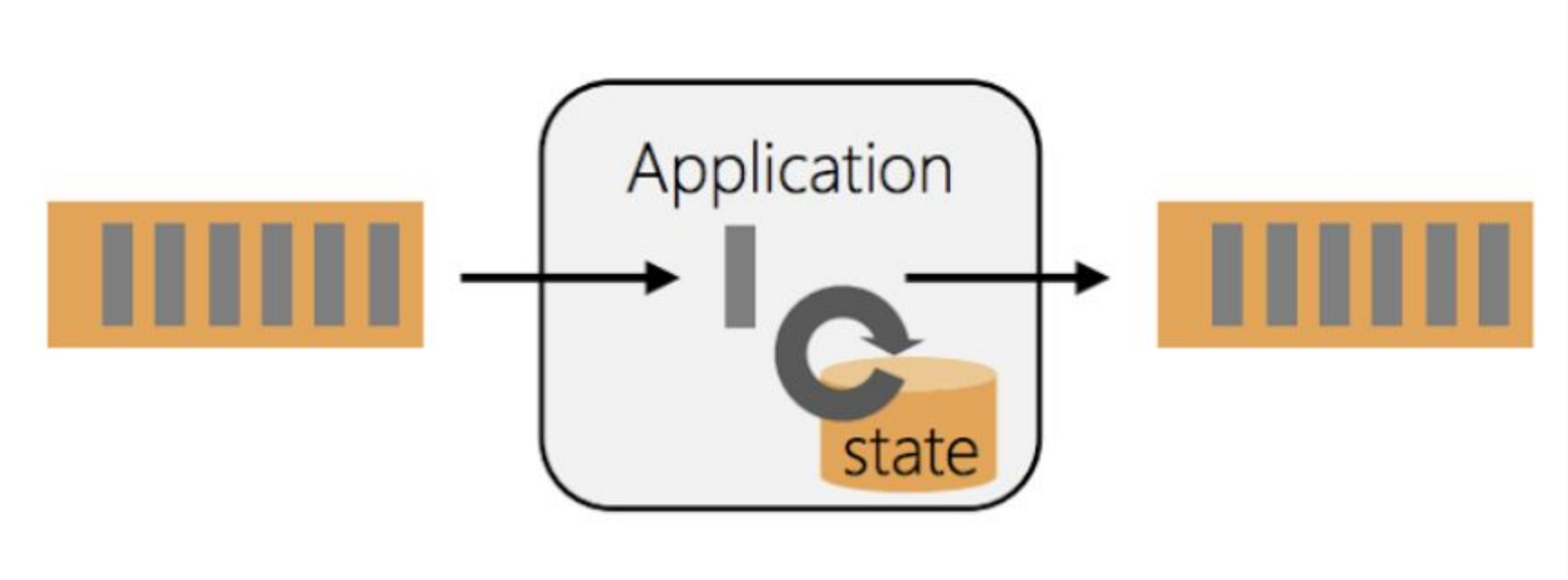
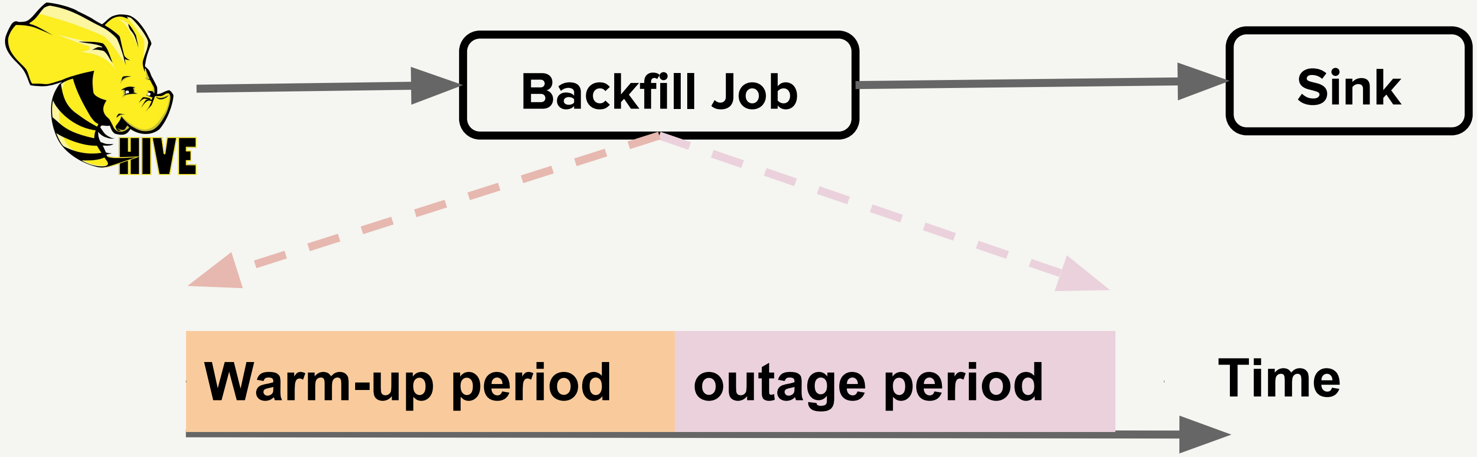
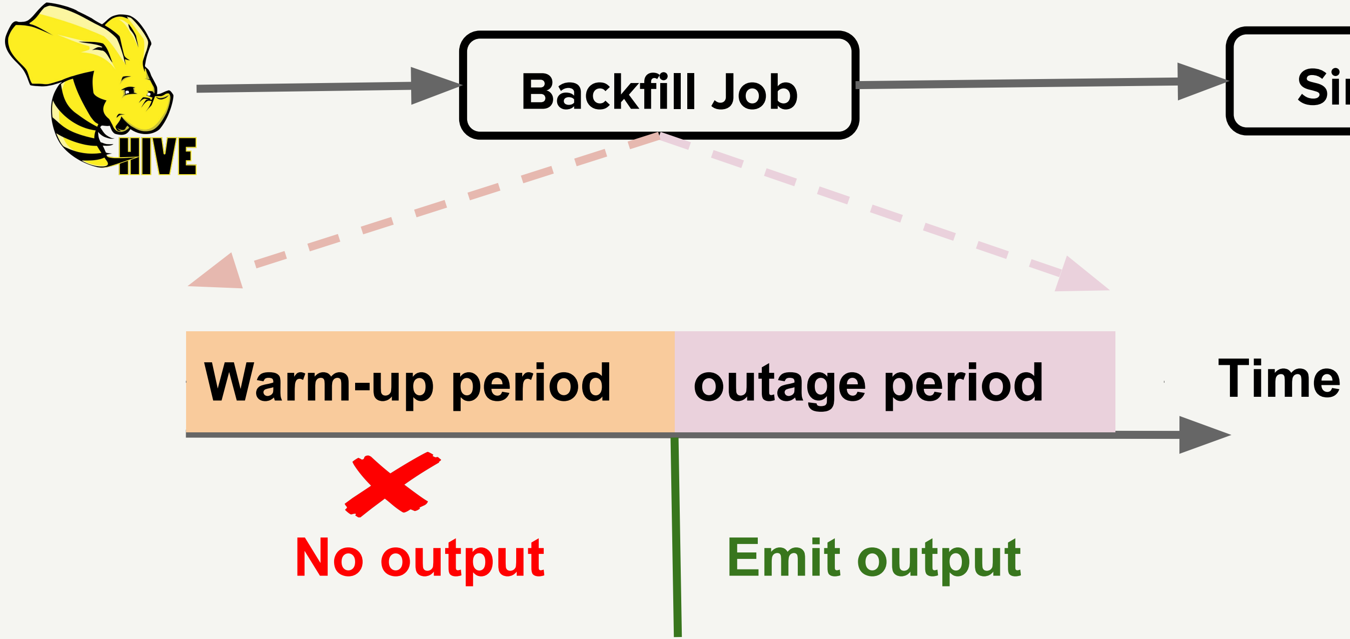


Image adapted from Stephen Ewen

Warm-up period



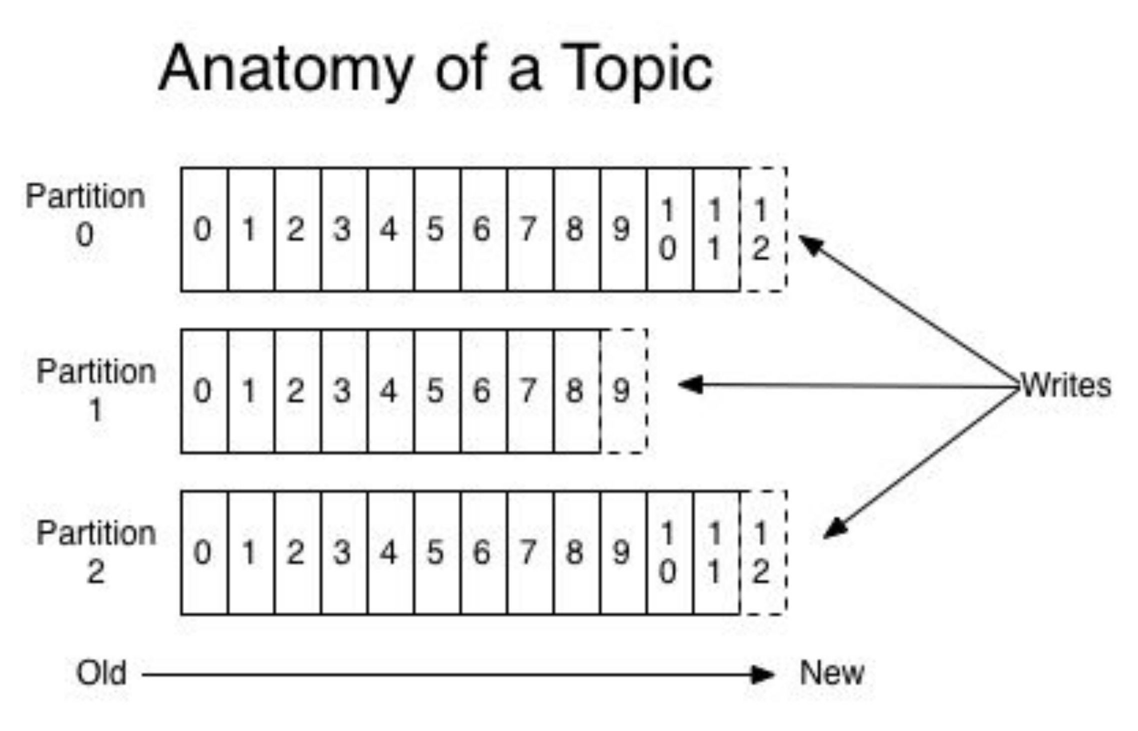
No output emit during warm-up



Hive backfill probably not for stateful jobs

- Warm-up issue
- Ordering issue

Kafka: messages ordered within a partition

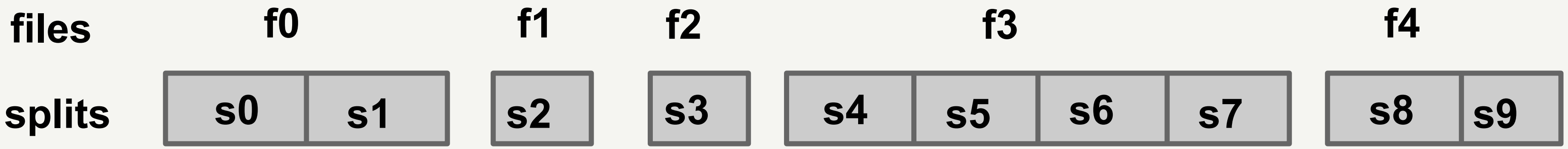


Source: kafka.apache.org

Hadoop input split

files f0 f1 f2 f3 f4

Hadoop input split



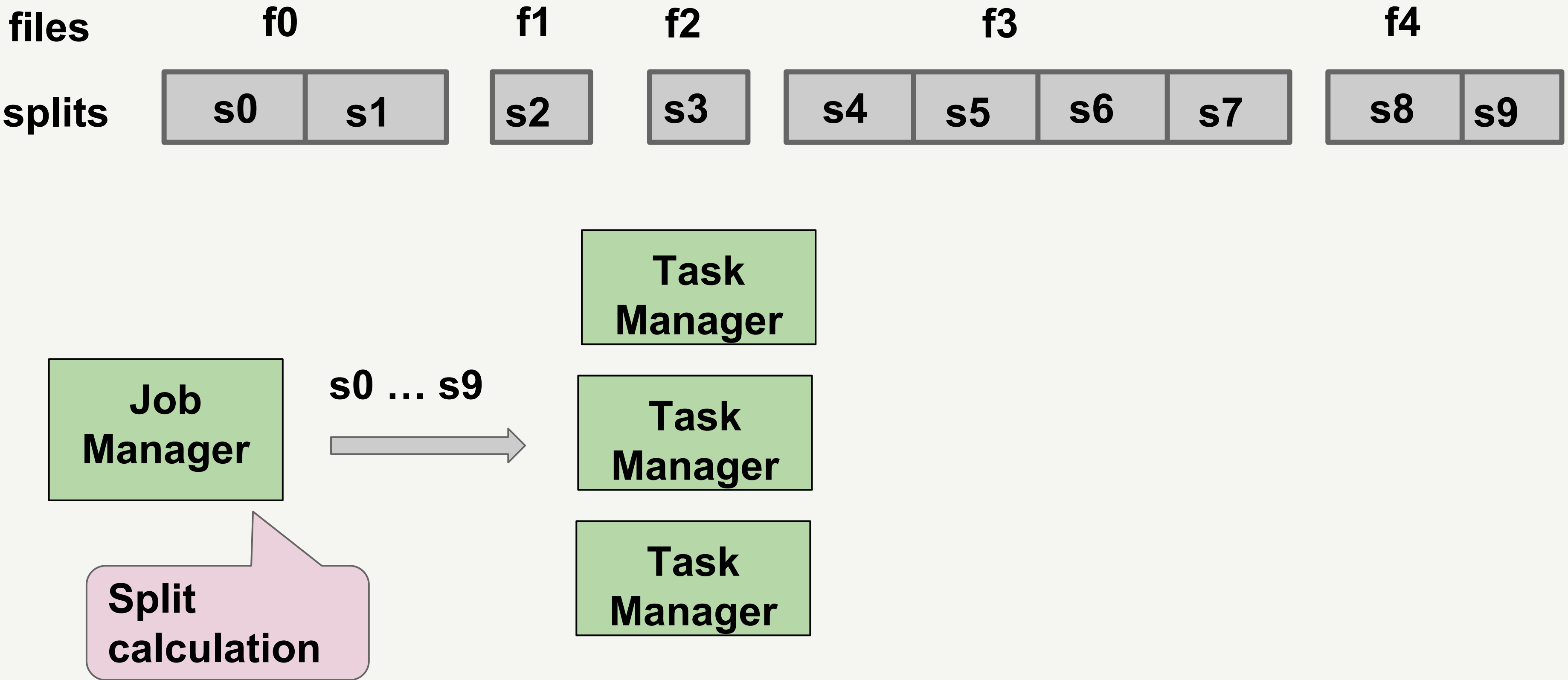
files

f0 f1 f2 f3 f4

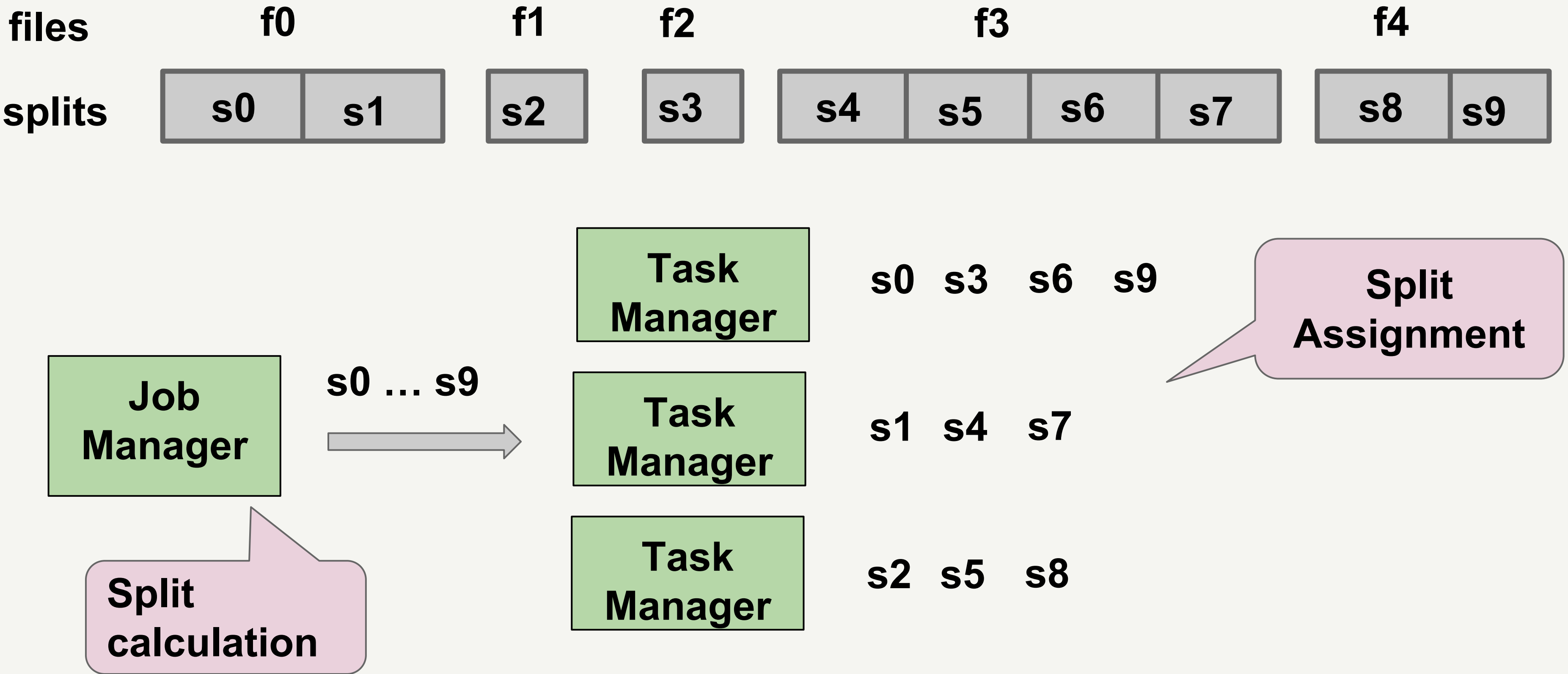
splits

s0 s1 s2 s3 s4 s5 s6 s7 s8 s9

Hadoop input split



Hadoop input split



files f0 f1 f2 f3 f4

splits s0 s1 s2 s3 s4 s5 s6 s7 s8 s9

Job Manager

s0 … s9

Task Manager

s0 s3 s6 s9

Task Manager

s1 s4 s7

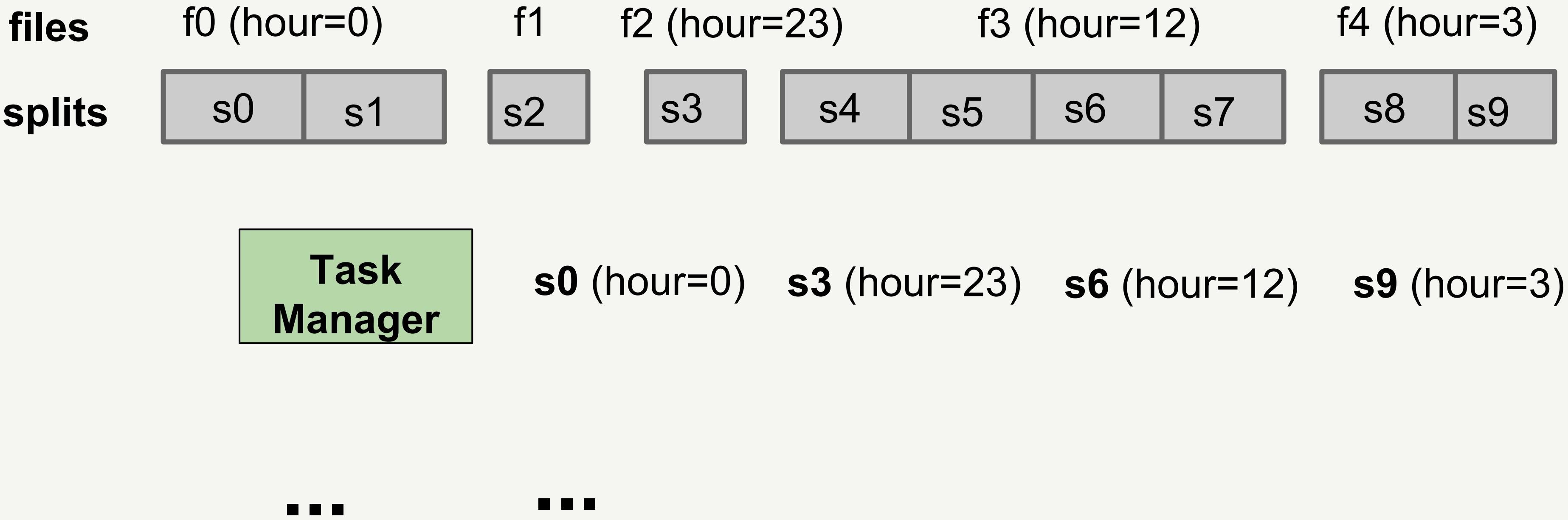
Task Manager

s2 s5 s8

Split Assignment

Split calculation

Where is the order?



files f0 (hour=0) f1 f2 (hour=23) f3 (hour=12) f4 (hour=3)

splits s0 s1 s2 s3 s4 s5 s6 s7 s8 s9

Task Manager **s0** (hour=0) **s3** (hour=23) **s6** (hour=12) **s9** (hour=3)

... ...

Does time/ordering matters?

- Probably not for stateless computation
- Probably important for stateful computation

Window with allowed lateness

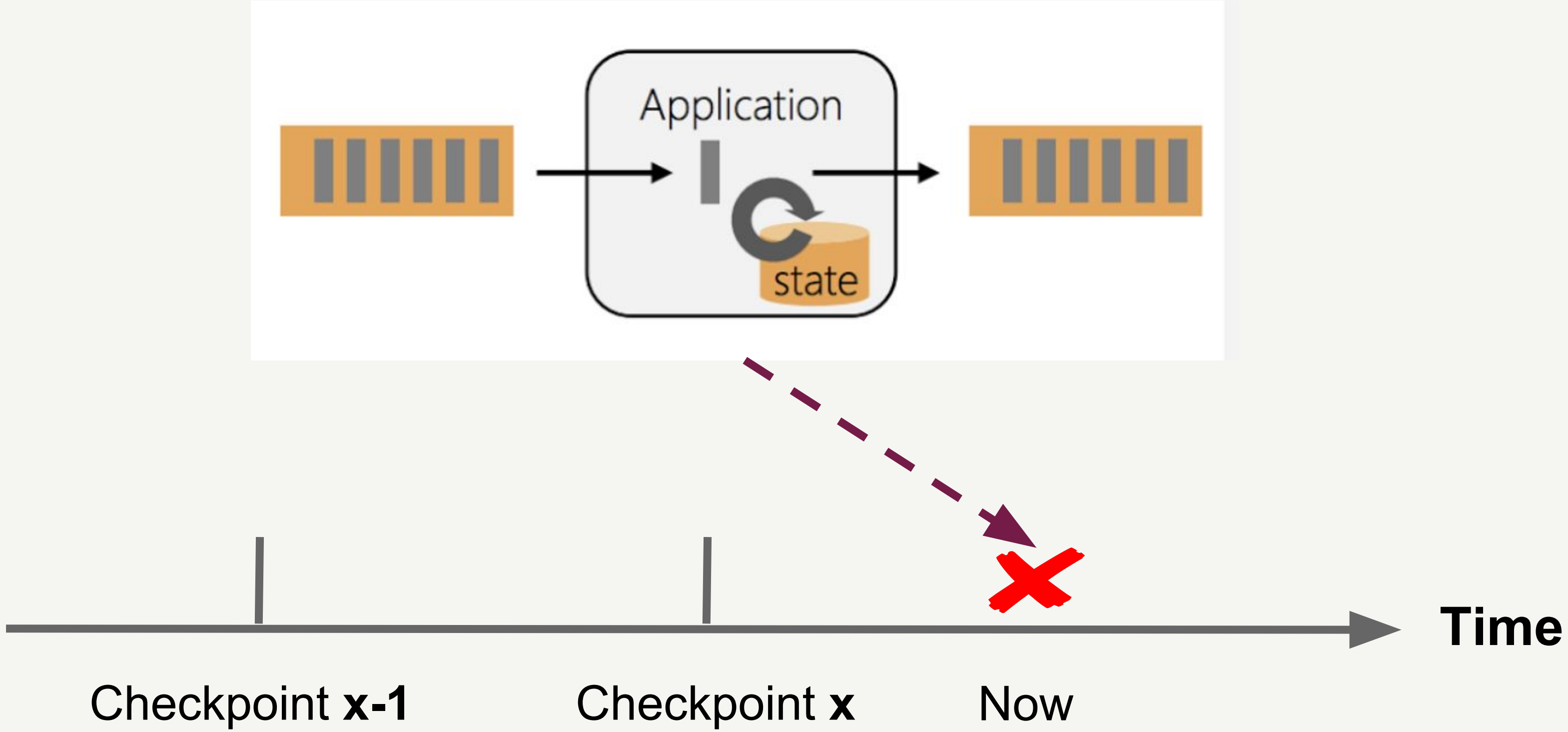
```
DataStream<T> input = ...;
input
    .keyBy(<key selector>)
    .window(<window assigner>)
    .allowedLateness(<time>)
    .<windowed transformation>(<window function>);
```

Source: flink.apache.org

How to recover

- Backfill
- Rewind Flink job

Flink checkpoint and fault tolerance



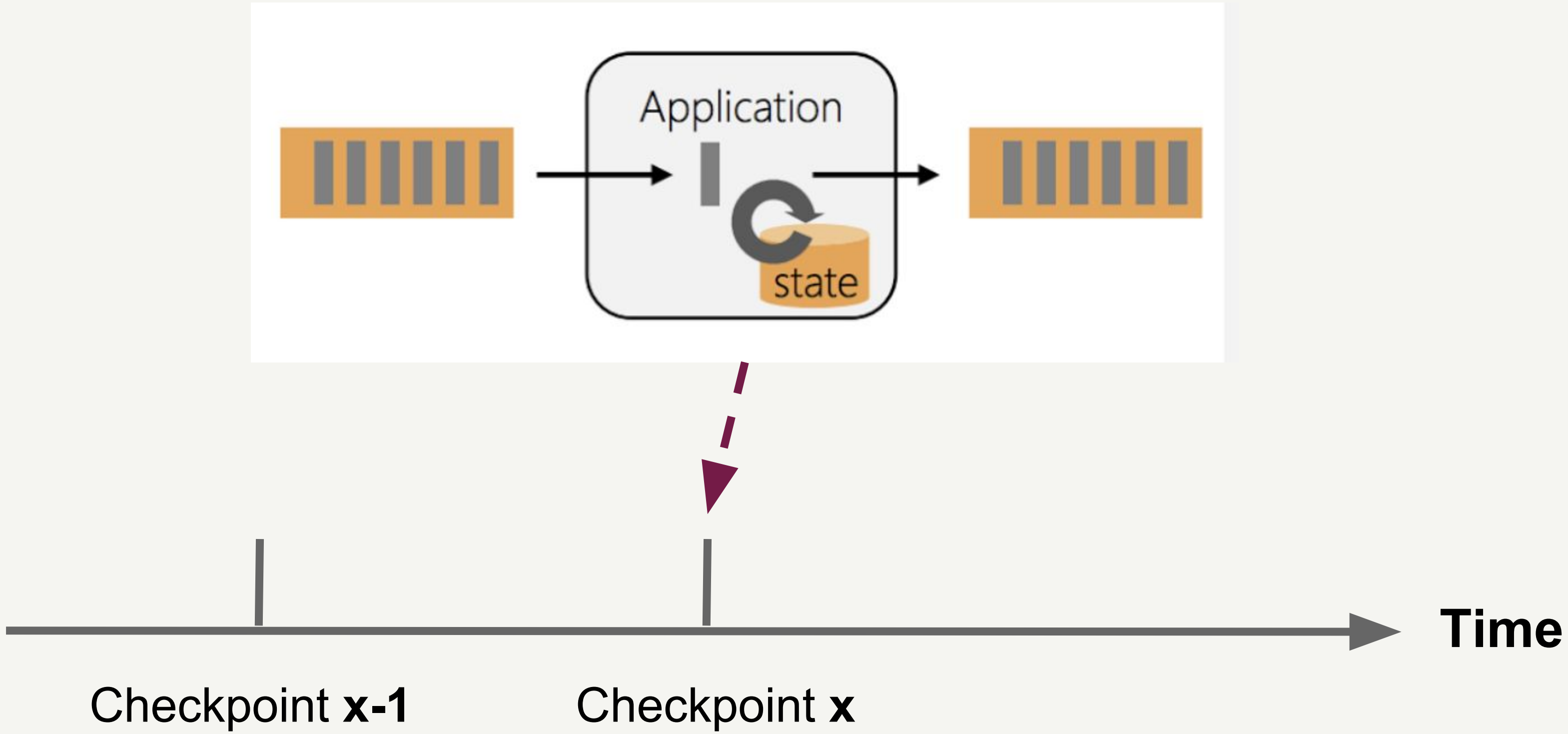
Time

Checkpoint **x-1**

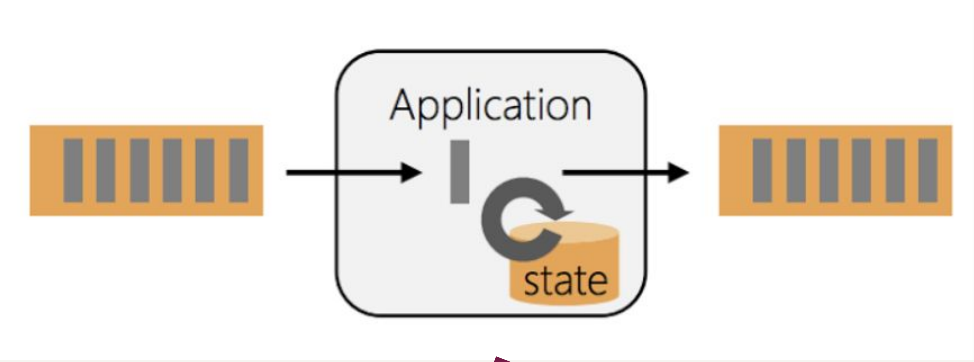
Checkpoint **x**

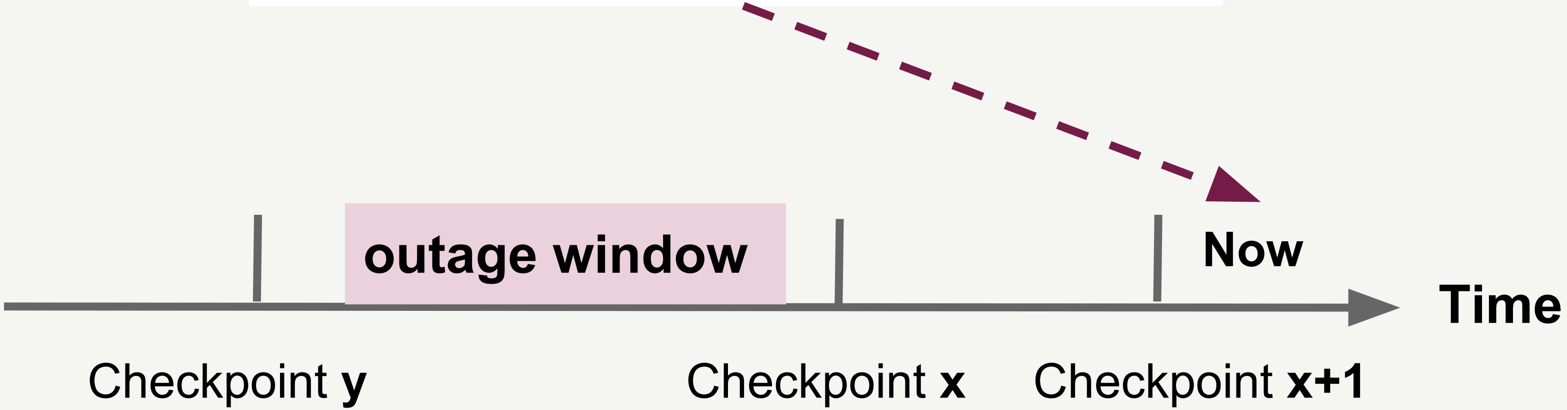
Now

Flink checkpoint and fault tolerance

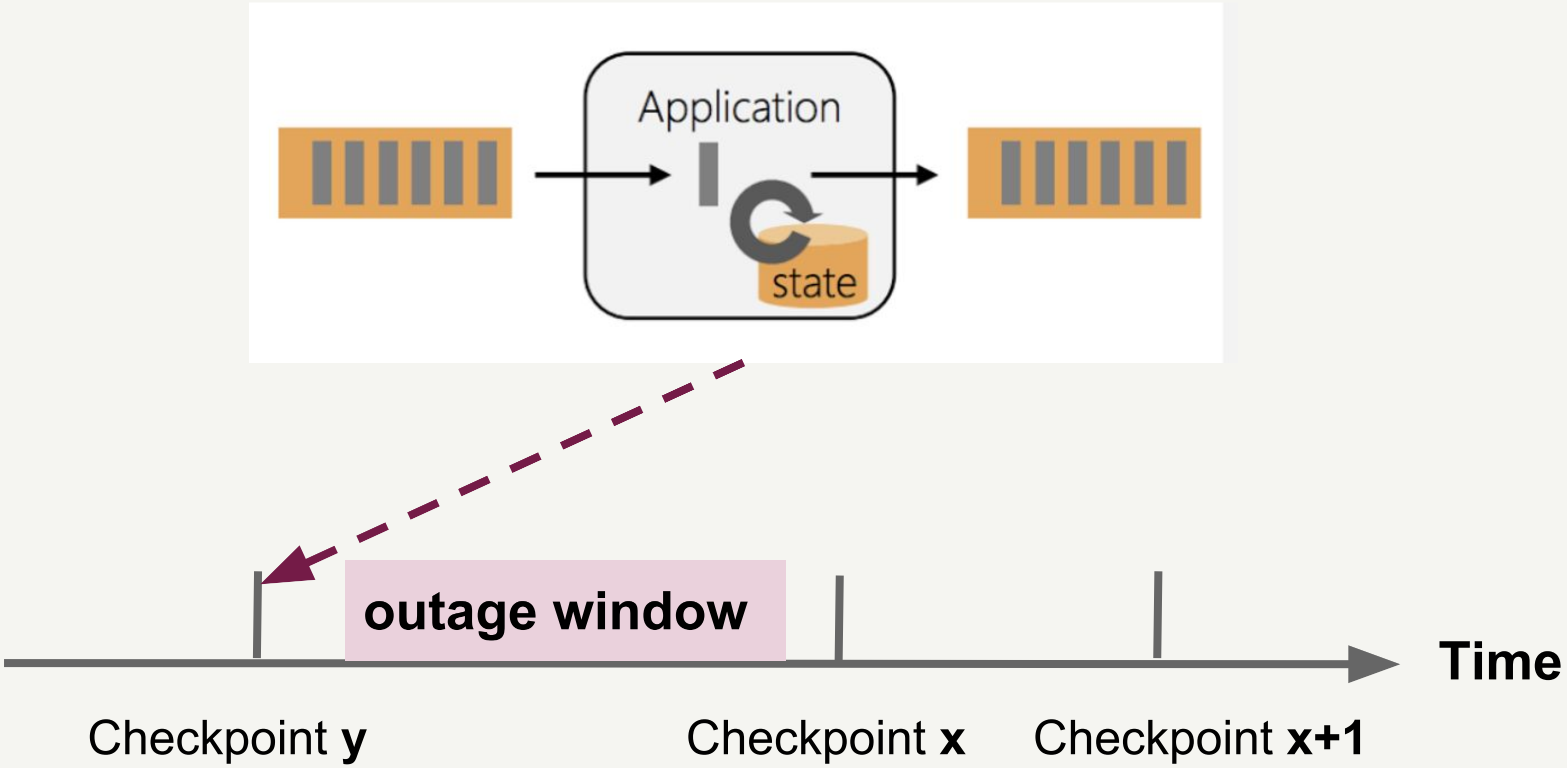


Flink rewind





Flink rewind



outage window

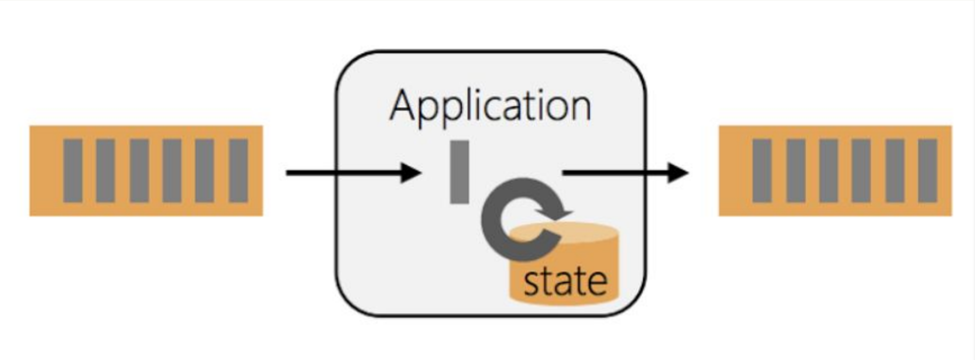
Time

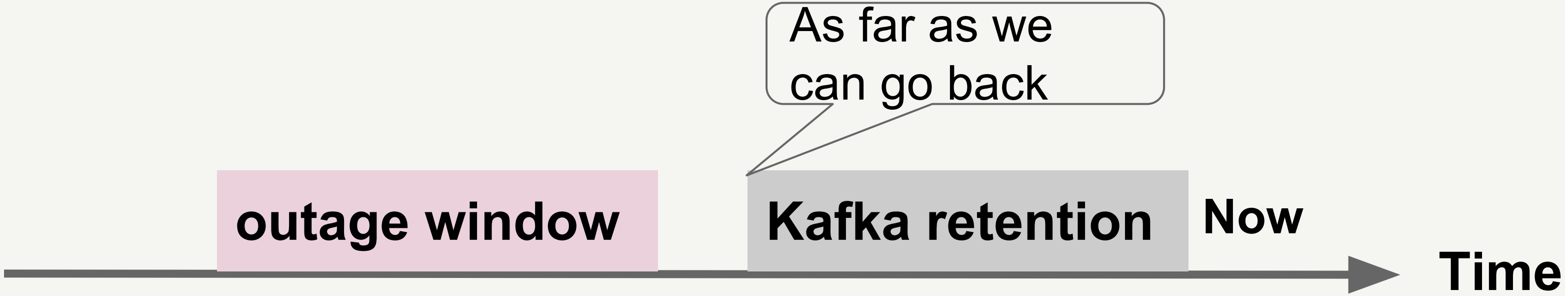
Checkpoint **y**

Checkpoint **x**

Checkpoint **x+1**

Kafka retention





As far as we can go back

outage window

Kafka retention

Now

Time

Hive backfill v.s. Flink rewind

	Hive backfill	Flink rewind
Warm-up issue	Yes	No
Ordering issue	Yes	No
Data retention	Months	Hours or days
Applicability	Stateless	Stateless and stateful

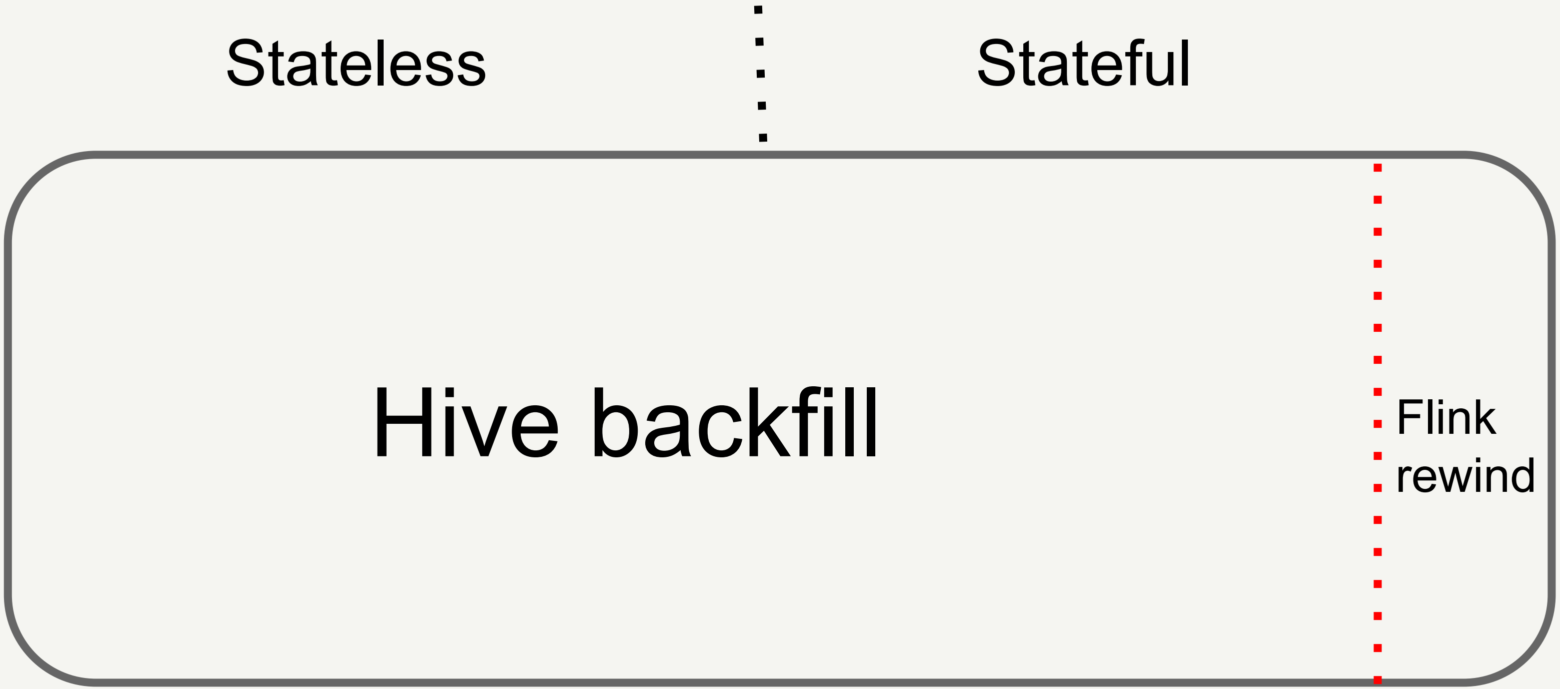
Pros for Hive backfill source

- Long-term storage (a few months)
- Fast recovery
  - S3 is very scalable
  - Runs in parallel with live job

Today’s recommendation

Stateless	Stateful
Hive backfill	Flink rewind

Is this the future?



Stateless

Stateful

Hive backfill

Flink rewind

Or is this the future?

Stateless

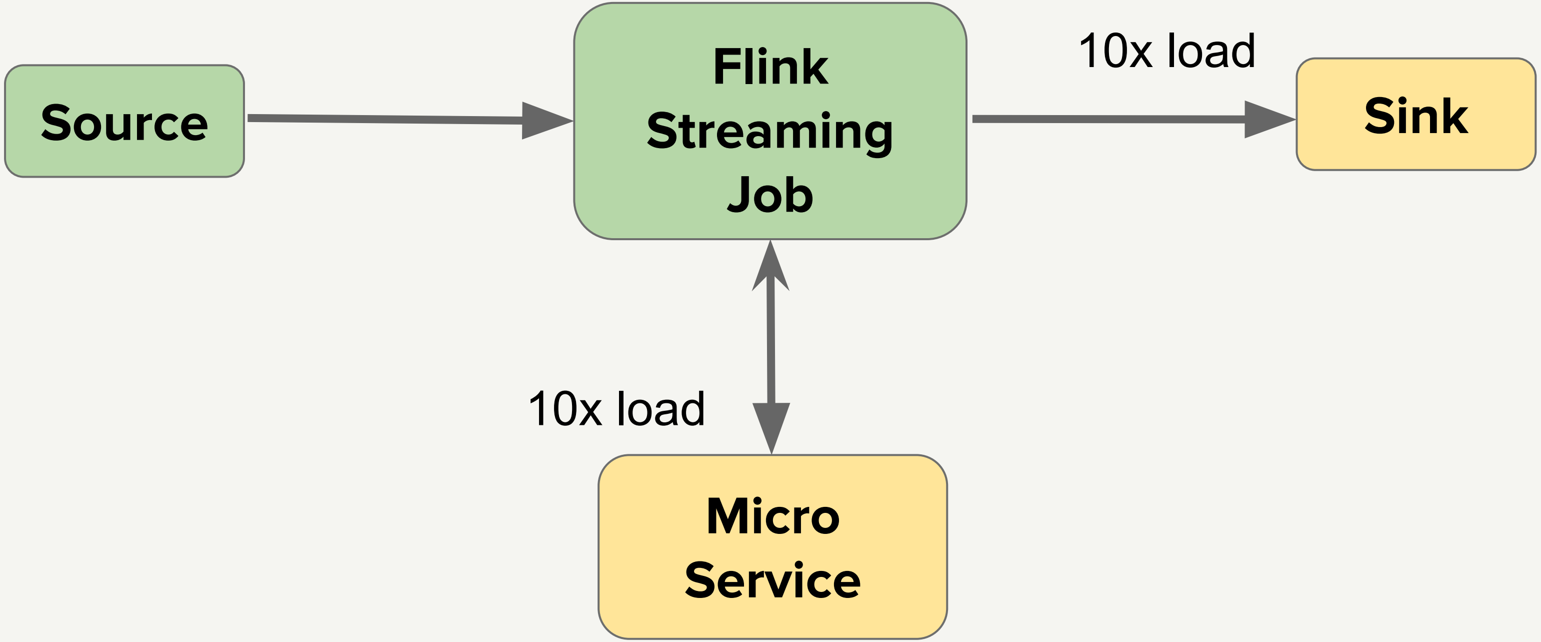
Stateful

Flink rewind

Caveats for reprocessing

- Does not overwhelm external services
- Non-retractable sink output
- Non-replayable dependencies

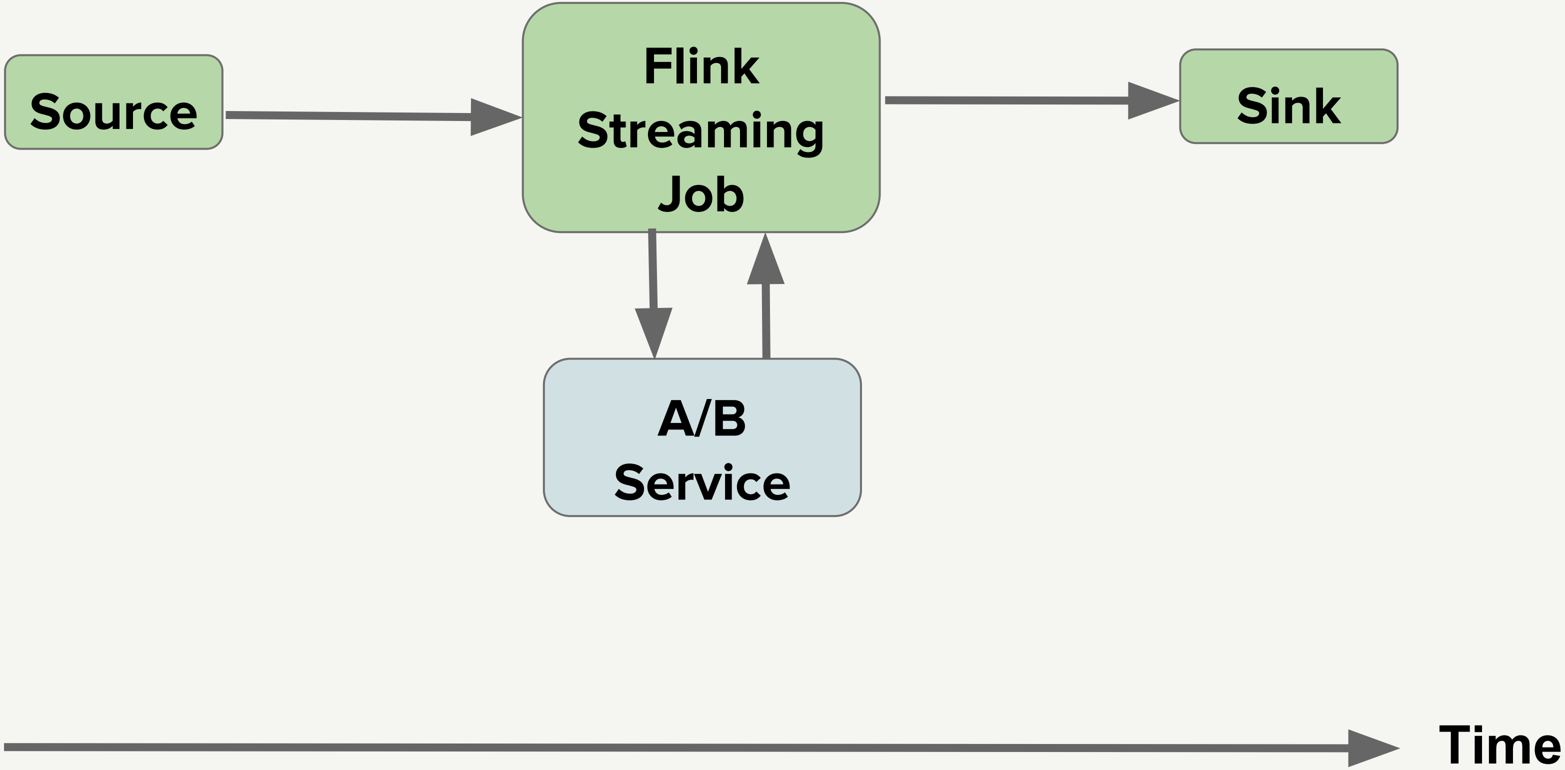
Does not overwhelm external services



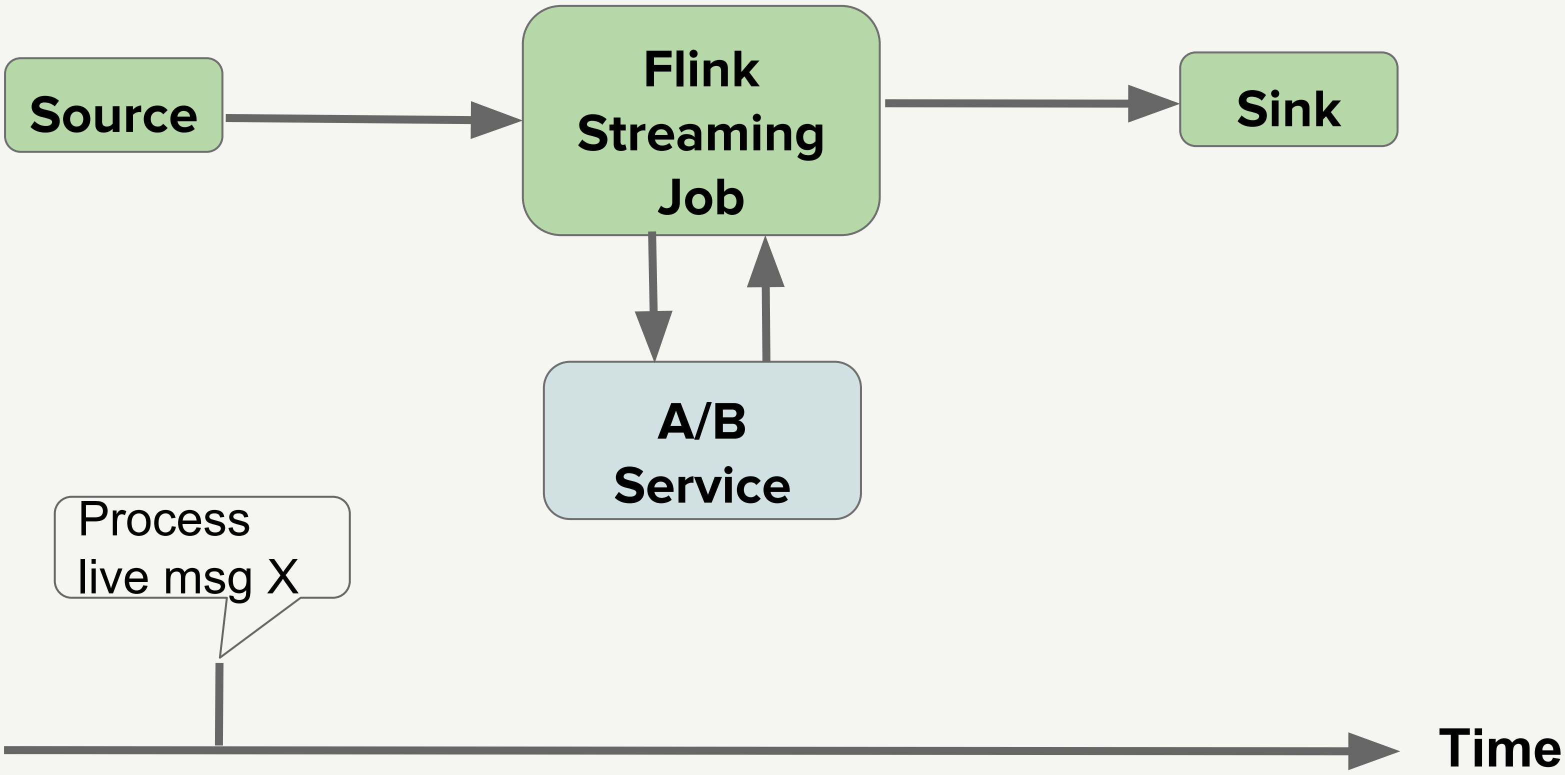
Non-retractable sink output

- Duplicates are ok
- Idempotent sink
- Cleanable sink
  - e.g. drop Hive partition with bad data

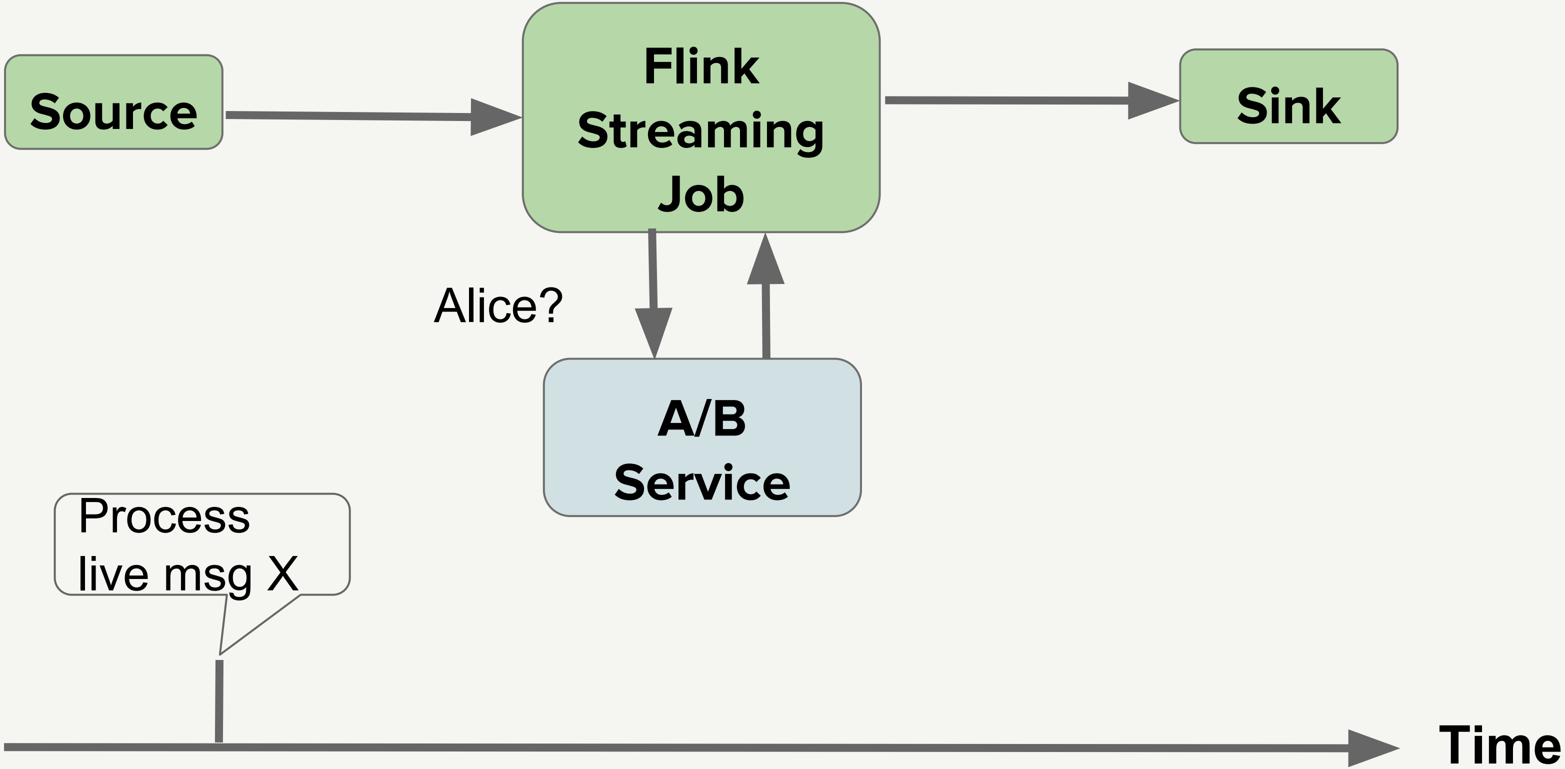
Non-replayable dependencies



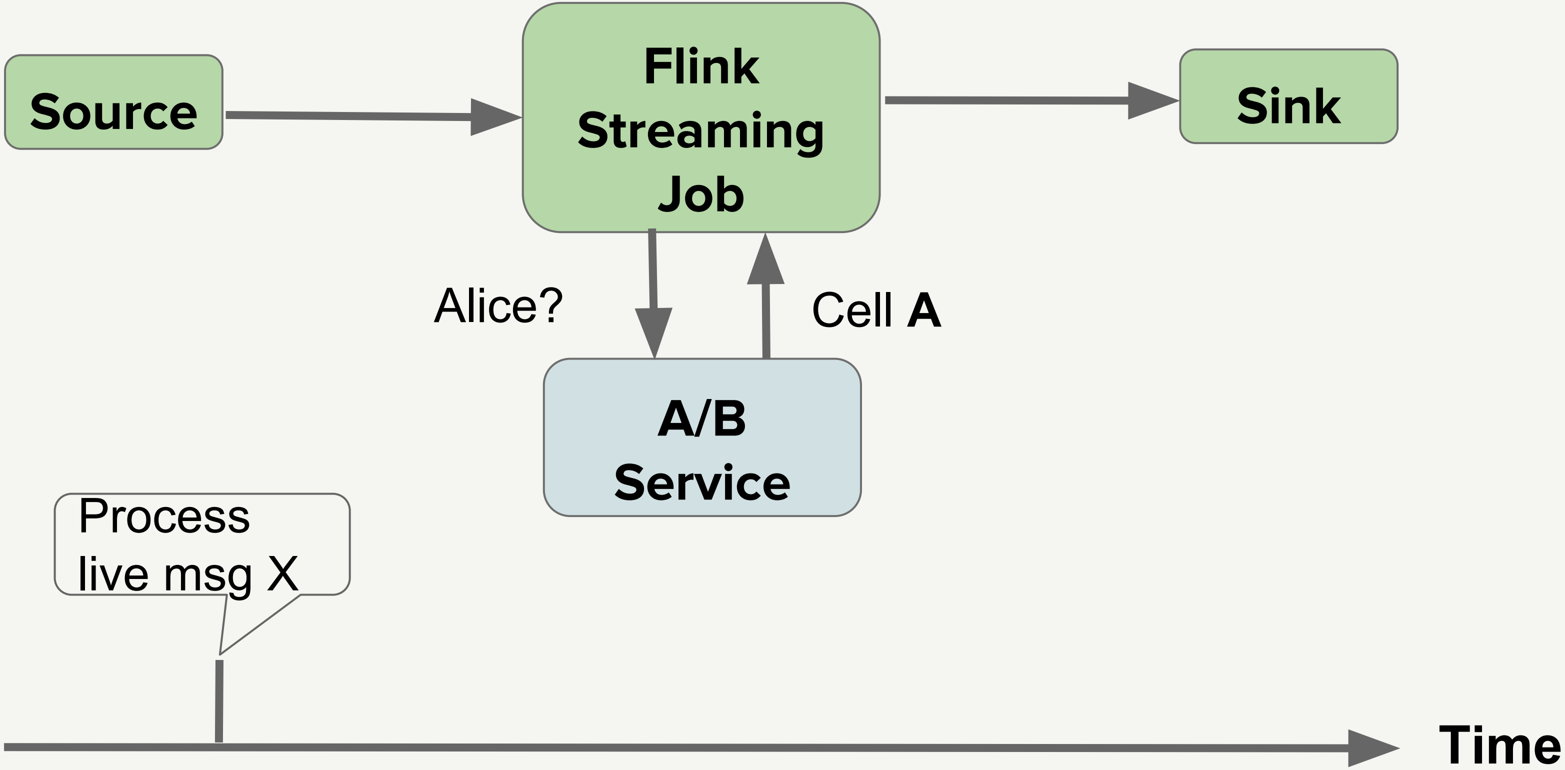
Non-replayable dependencies



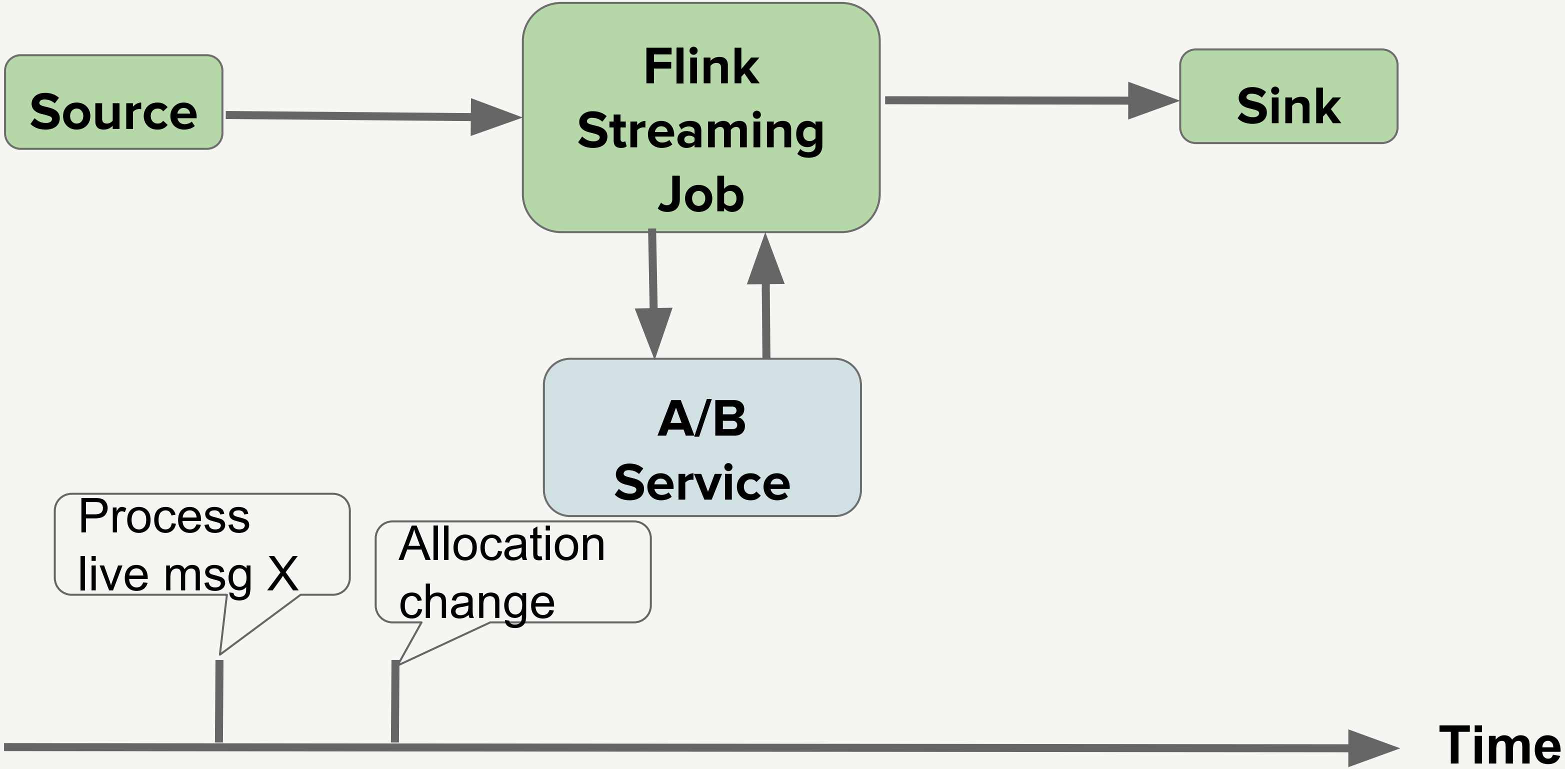
Non-replayable dependencies



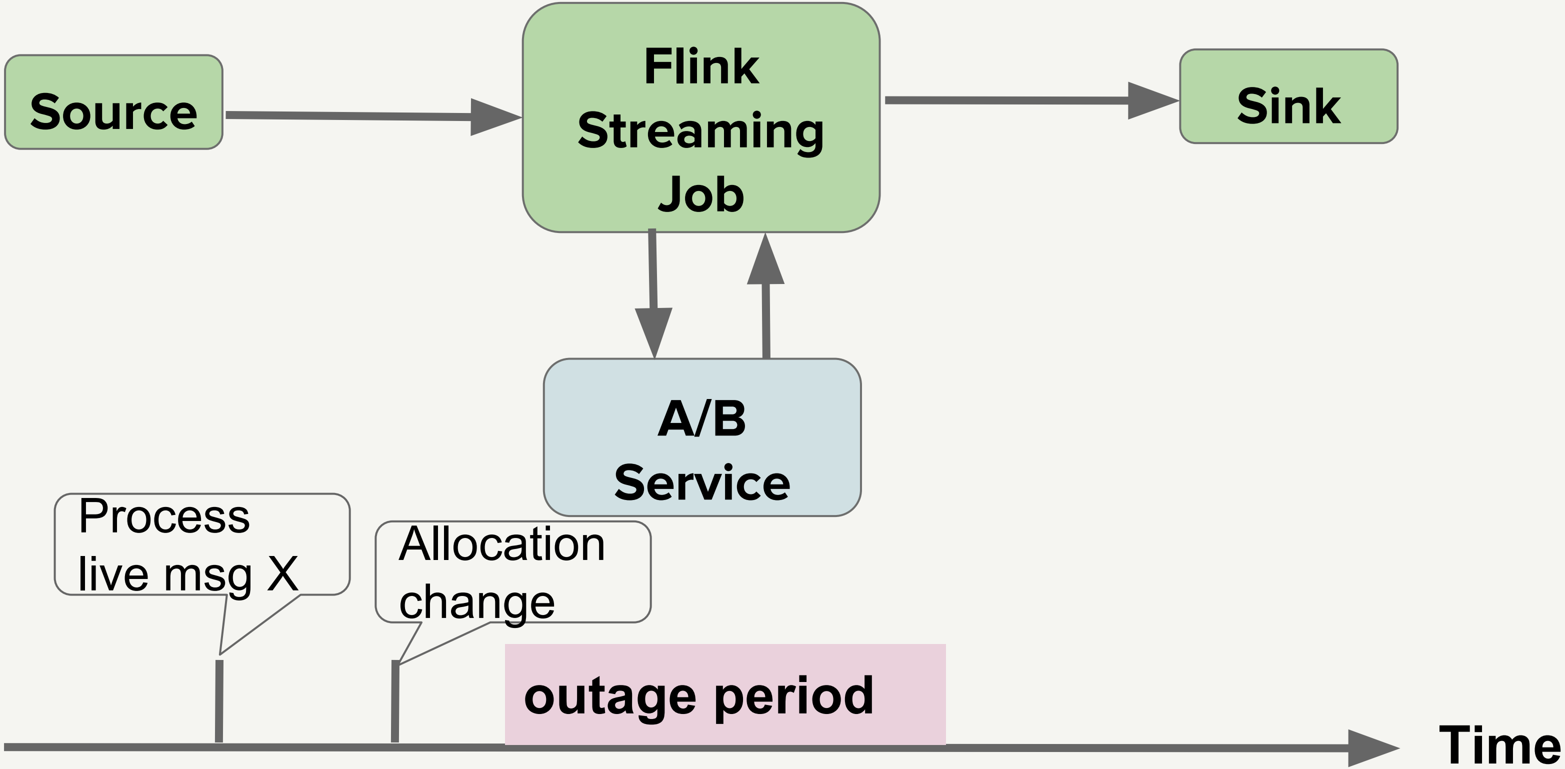
Non-replayable dependencies



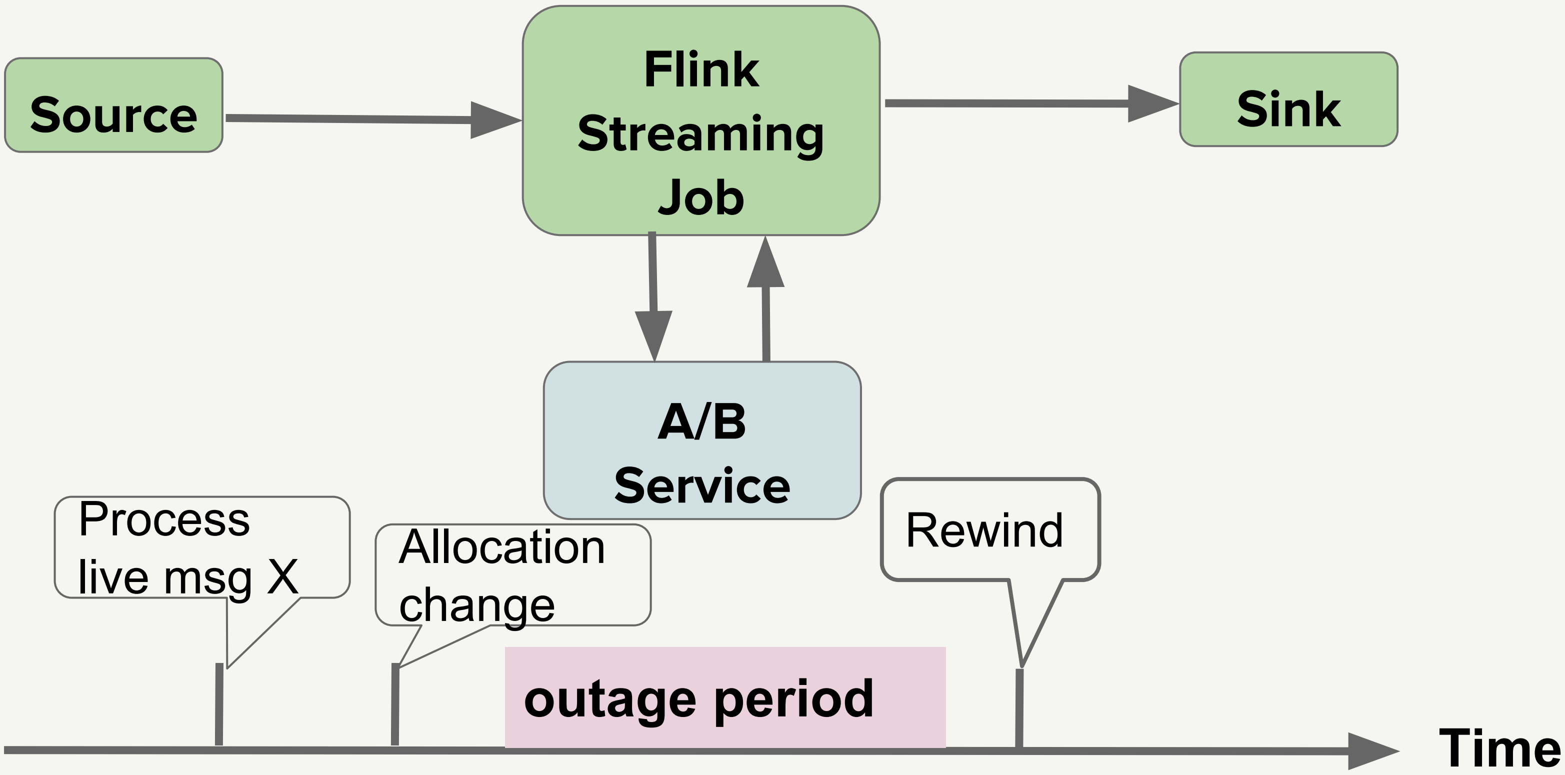
Non-replayable dependencies



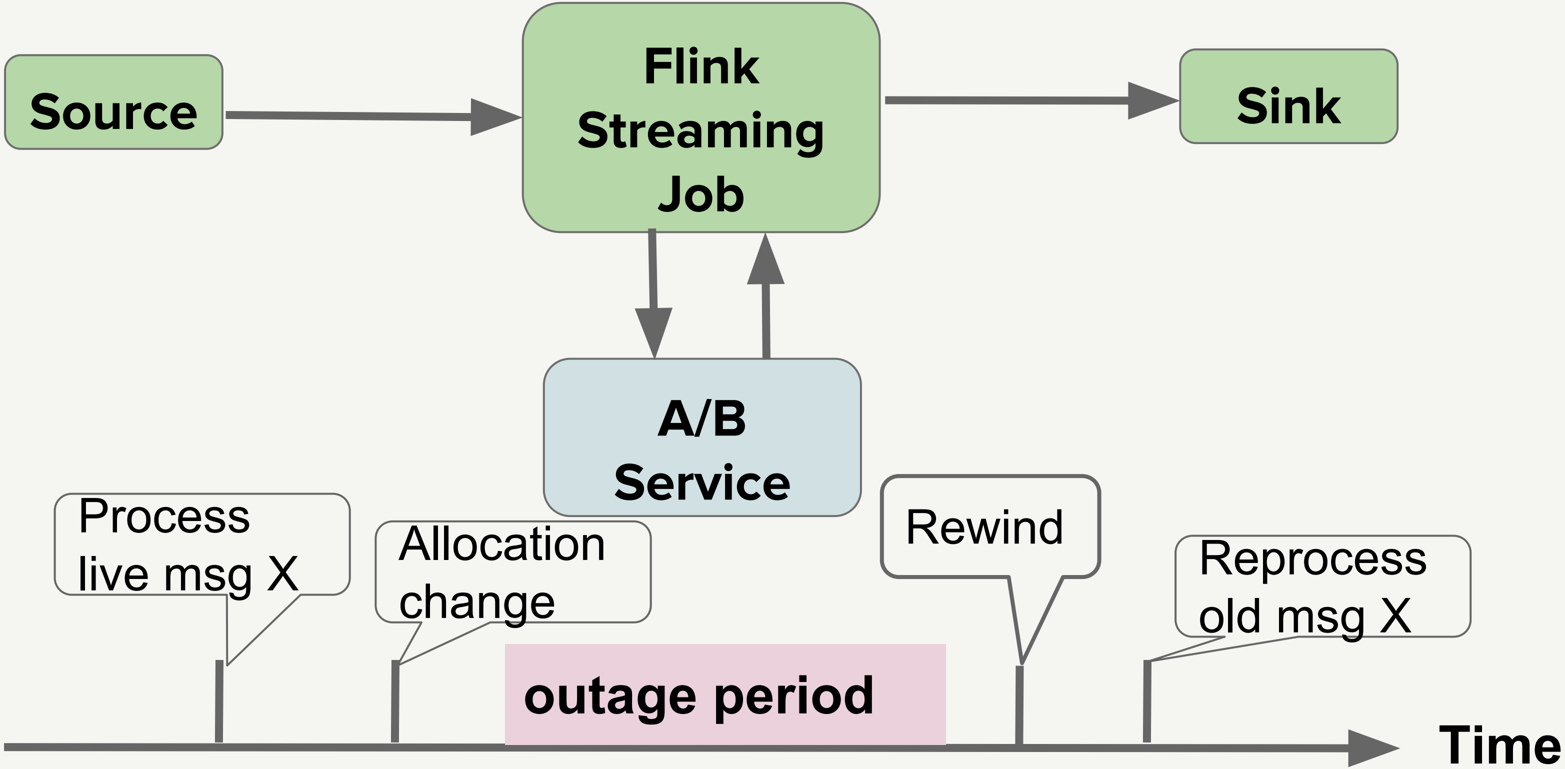
Non-replayable dependencies



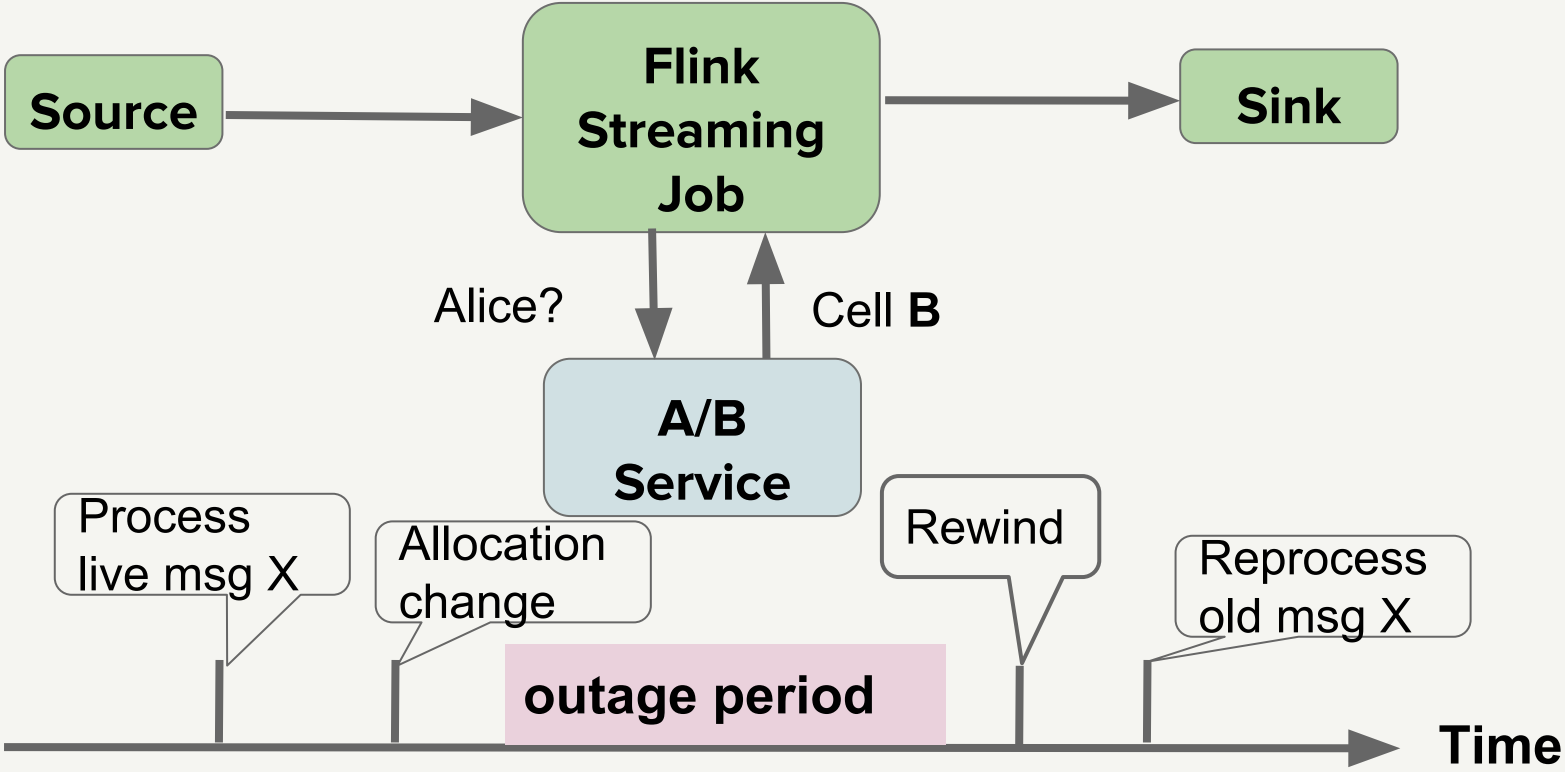
Non-replayable dependencies



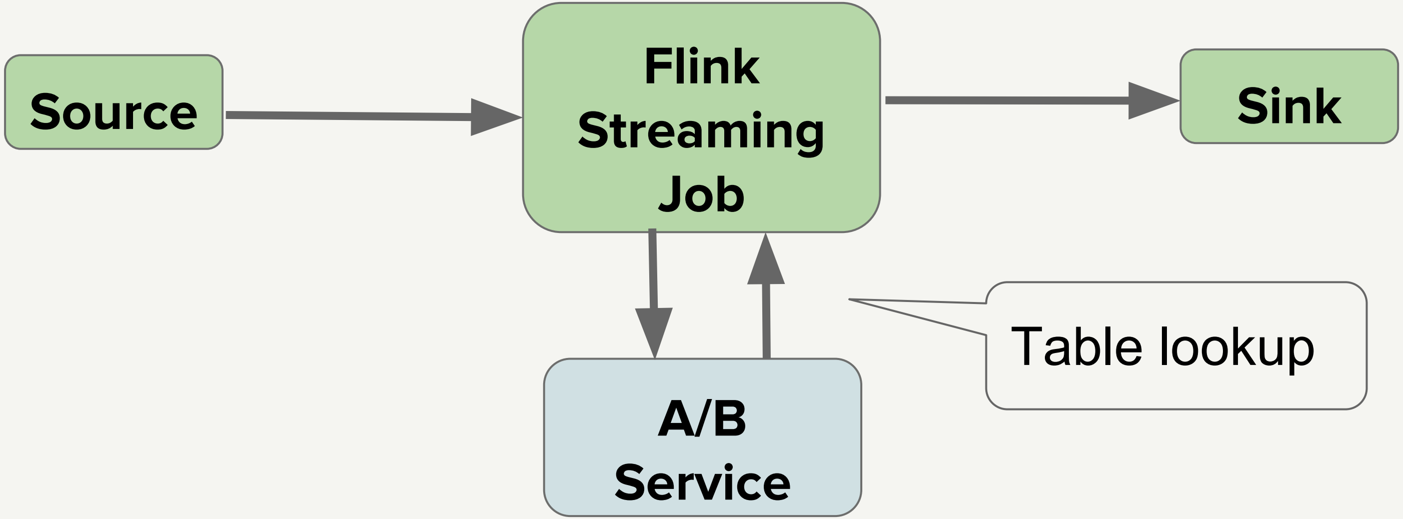
Non-replayable dependencies



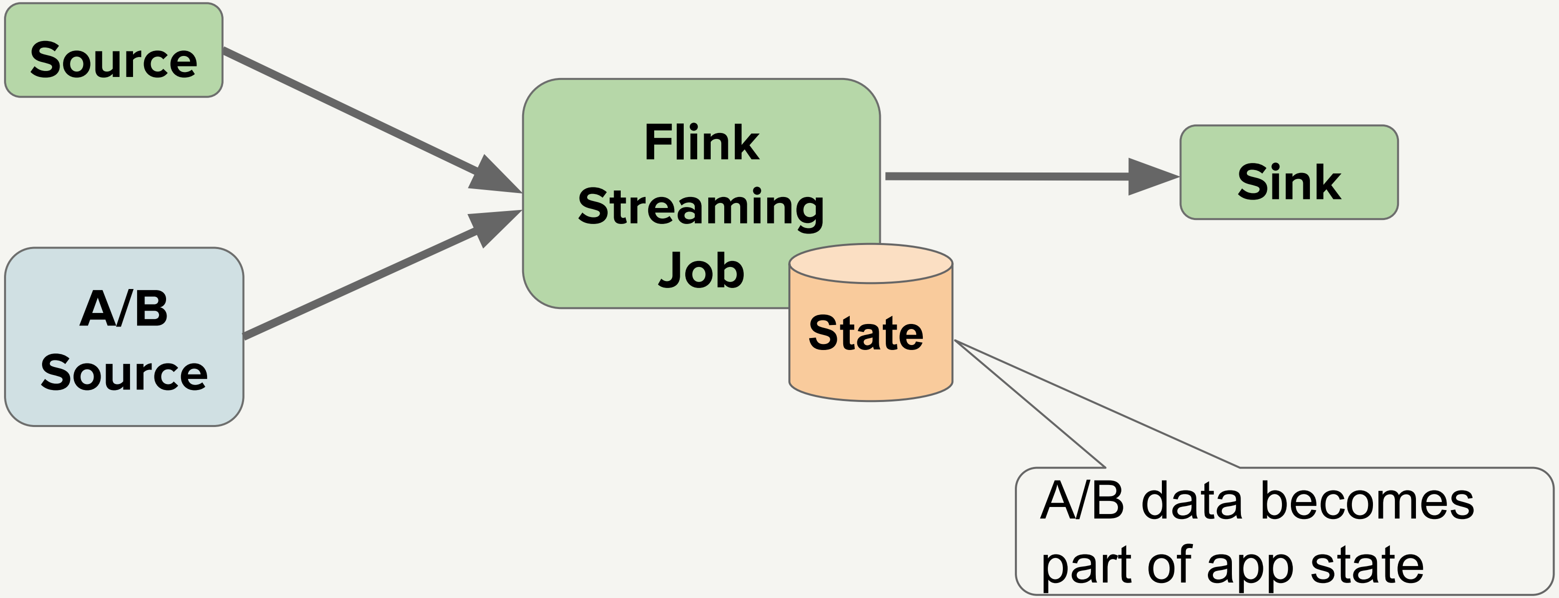
Non-replayable dependencies



Convert table to stream



Convert table to stream

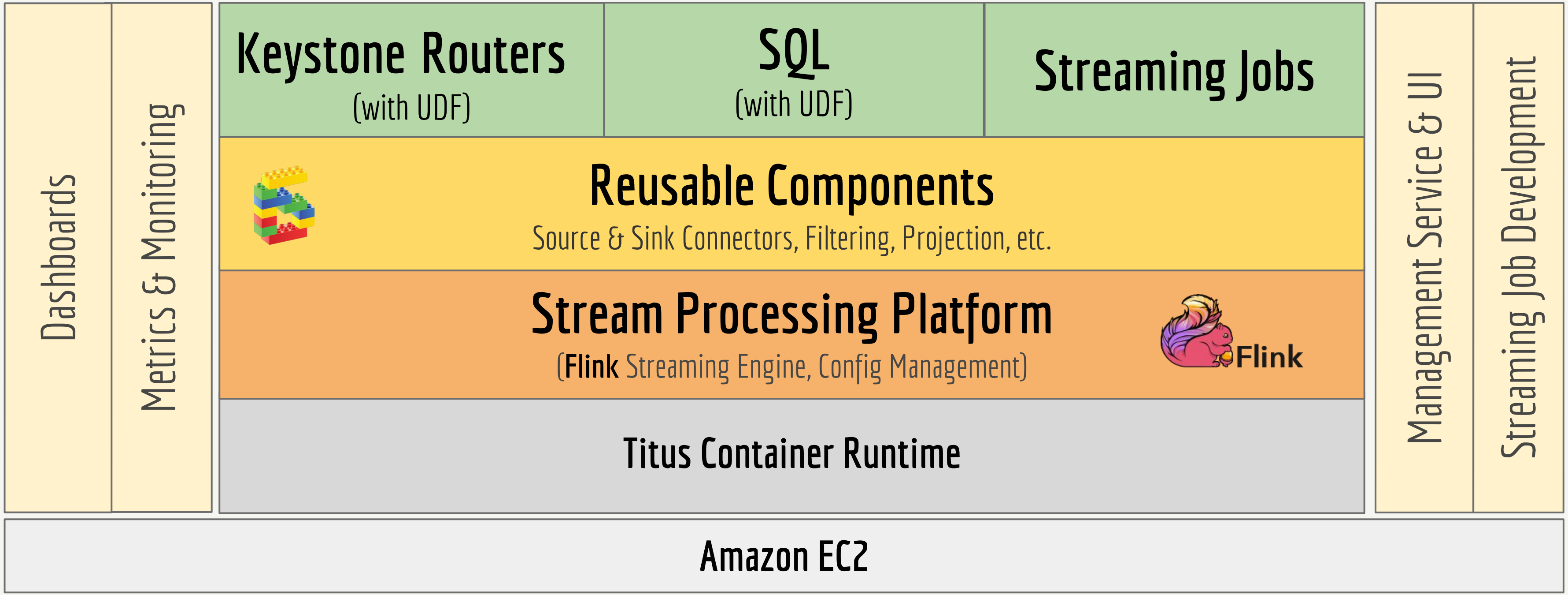


A/B data becomes part of app state

Stream Kong

Putting together

SPaaS Layered Cake



Dashboards

Metrics & Monitoring

Keystone Routers
(with UDF)

SQL
(with UDF)

Streaming Jobs

Reusable Components
Source & Sink Connectors, Filtering, Projection, etc.

Stream Processing Platform
(**Flink** Streaming Engine, Config Management)

Titus Container Runtime

Management Service & UI

Streaming Job Development

Amazon EC2

Geico caveman, https://memegenerator.net

Thank you!

NETFLIX

@stevenzwu