New Source API — Make it Easy!

秦江杰 / Jiangjie (Becket) Qin

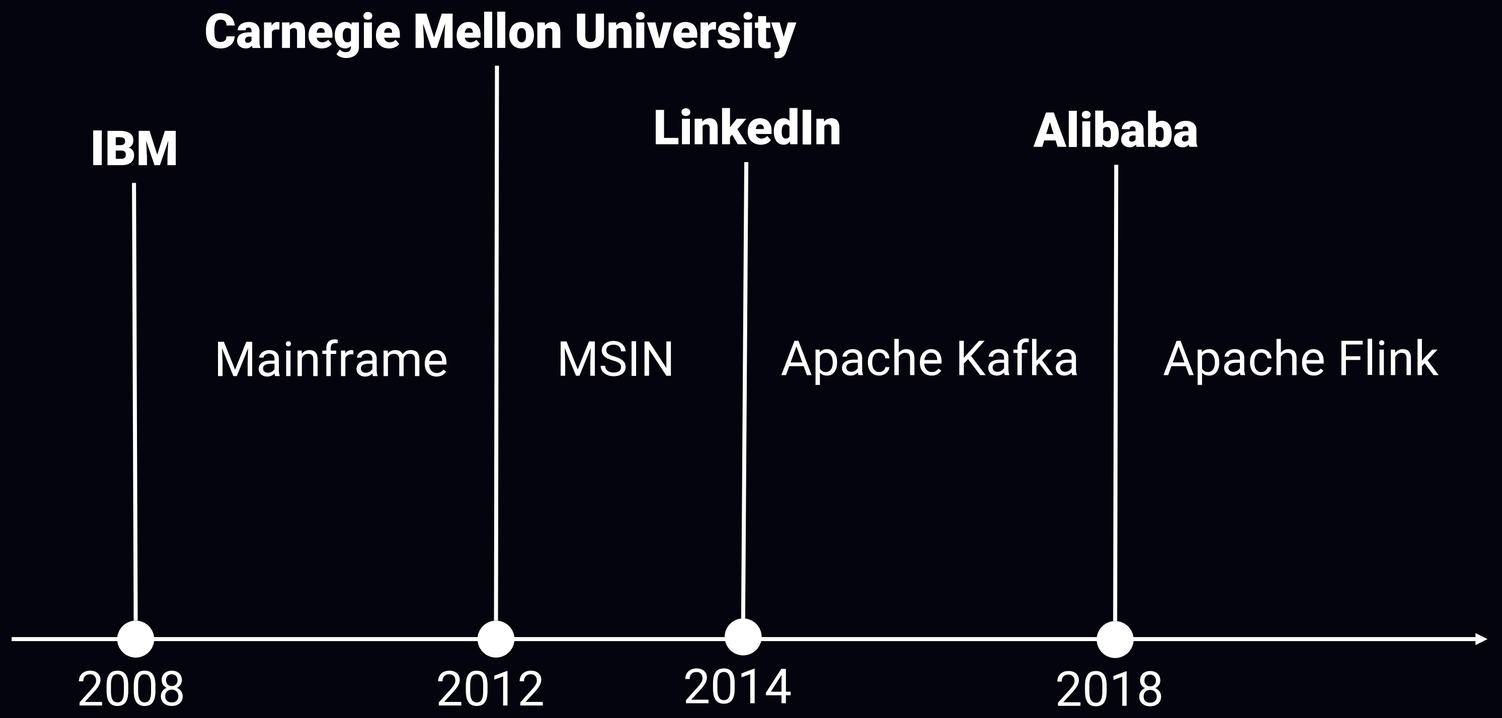
FLINK FORWARD # ASIA

实时即未来 # Real-time Is The Future

FLINK FORWARD

关于我
About me

阿里巴巴
Alibaba Inc.
高级技术专家
Staff Software Engineer & Senior Manager
Apache Flink 和
Apache Kafka
项目管理委员会成员
PMC of Apache Kafka & Apache Flink



目录
Agenda

- 什么是 Flink Source
  What is a Flink source?
- 为什么需要新的 Source?
  Why new Source API?
- 新 Source 的设计
  Design the new Source
  - 设计目标
    The design goals
  - Enumerator – Reader 架构
    Enumerator - Reader architecture
  - Source Reader 的线程模型
    Source Reader Threading model
  - 水印生成
    Watermark generation
  - 任务可协同的算子
    Coordinated operators
  - 状态保存和恢复
    Checkpoint and failover
- 轻松实现生产可用的 Flink Source
  Production-ready Source made easy!

什么是 Flink Source

What is a Flink source

- 从外部系统读取记录

  Read records from the external systems

- 在处理流程中加入控制事件

  Introduce control events to the processing graph

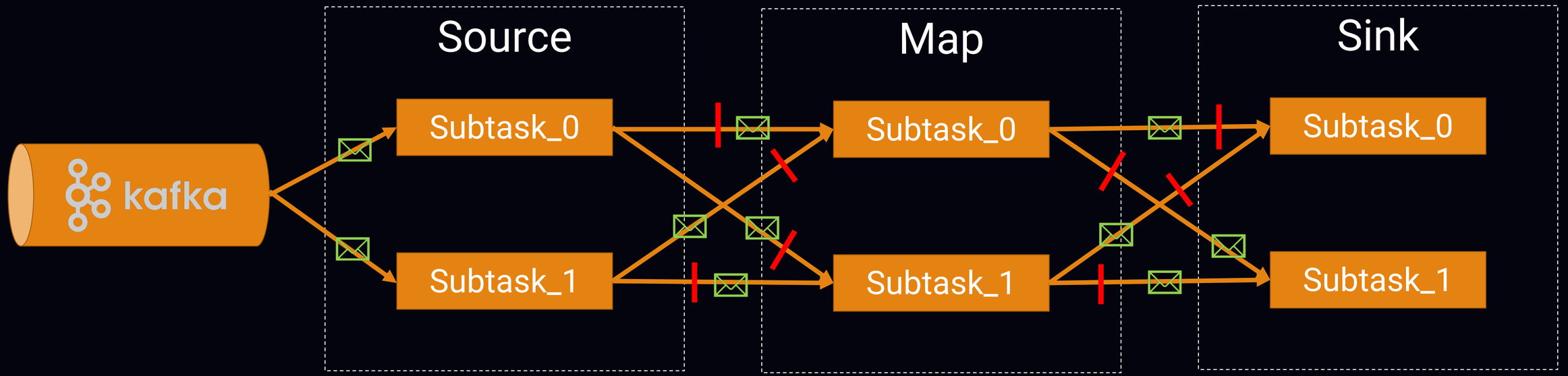
  - 事件时间水印

    Watermarks
  - 检查点对齐记录

    Checkpoint markers

一个简单的例子

A simple example



Flink Source 的工作还包括

A few more things for sources

- 记录片分配
  Source splits assignment
- 事件时间对齐
  Event time alignment
- 根据投递语义创建检查点
  Checkpoint for delivery semantic
- 负载均衡
  Workload balance
- 记录解析
  Record parsing

目录
Agenda

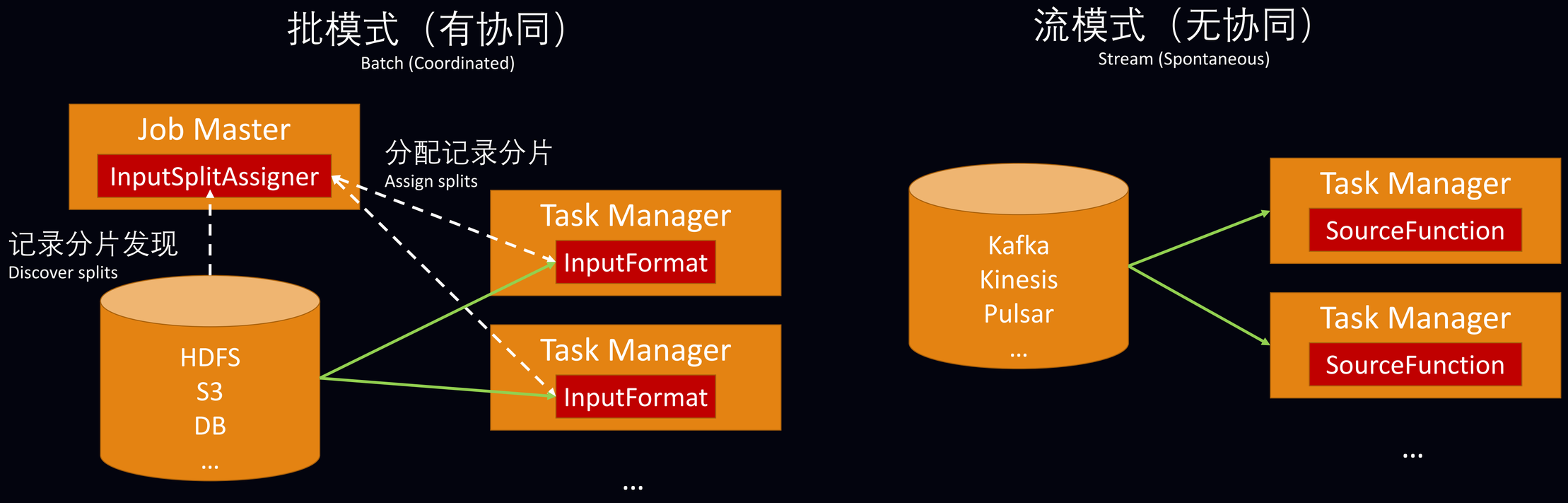
- 什么是 Flink Source
  What is a Flink source?
- 为什么需要新的 Source?
  Why new Source API?
- 新 Source 的设计
  Design the new Source
  - 设计目标
    The design goals
  - Enumerator – Reader 架构
    Enumerator - Reader architecture
  - Source Reader 的线程模型
    Source Reader Threading model
  - 水印生成
    Watermark generation
  - 任务可协同的算子
    Coordinated operators
  - 状态保存和恢复
    Checkpoint and failover
- 轻松实现生产可用的 Flink Source
  Production-ready Source made easy!

当前 Source 的问题

Issues of current Source

- 批和流的执行模式不一致

  Different execution pattern for Stream and Batch



当前 Source 的问题

Issues of current Source

- 难以提供对 Source 的通用实现
  Difficult to implement source common functionalities
  - 每个记录分片 / 分区的事件时间水印
    Per-split /per-partition watermark
  - 事件时间对齐
    Event time alignment
  - 动态记录片分配
    Dynamic split assignment
  - …
    …

当前 Source 的问题

Issues of current Source

- 复杂的多线程环境
  Multi-thread environment is tricky
  - 主线程
    Main thread (processing thread)
  - 检查点线程
    Checkpoint thread
  - 定时器线程
    Timer Thread
- 与新的信箱线程模型无法协同
  Does not work well with the new mailbox threading model

目录
Agenda

- 什么是 Flink Source
  What is a Flink source?
- 为什么需要新的 Source?
  Why new Source API?
- 新 Source 的设计
  Design the new Source
  - 设计目标
    The design goals
  - Enumerator – Reader 架构
    Enumerator - Reader architecture
  - Source Reader 的线程模型
    Source Reader Threading model
  - 水印生成
    Watermark generation
  - 任务可协同的算子
    Coordinated operators
  - 状态保存和恢复
    Checkpoint and failover
- 轻松实现生产可用的 Flink Source
  Production-ready Source made easy!

设计目标
Design goals

- 统一批和流的 Source
  Unify the batch and streaming source

Current Flink Source

Batch
(InputFormat)

....

Stream
(SourceFunction)

....

New Flink Source

Source Split 分配
Source Splits assignment

记录解析
Record parsing

线程模型
Threading model

事件时间水印
Watermark generation

符合投递语义的检查点机制
Checkpoint for delivery semantic

设计目标
Design goals

- 统一批和流的 Source
  Unify the batch and streaming source
- 简化 Source 连接器的实现
  Make it easy to implement new Source connectors

Flink Source

Batch (InputFormat)

cassandra, HIVE, S3, elastic, hadoop

....

Stream (SourceFunction)

kafka, Google Cloud Pub/Sub, PULSAR, APACHE nifi, RabbitMQ

....

New Flink Source

kafka, S3, APACHE nifi, hadoop

Source Split 分配
Source Splits assignment

记录解析
Record parsing

线程模型
Threading model

事件时间水印
Watermark generation

符合投递语义的检查点机制
Checkpoint for delivery semantic

FlinkSourceBase

cassandra, Google Cloud Pub/Sub, HIVE

PULSAR, RabbitMQ, elastic

目录
Agenda

- 什么是 Flink Source
  What is a Flink source?
- 为什么需要新的 Source?
  Why new Source API?
- 新 Source 的设计
  Design the new Source
  - 设计目标
    The design goals
  - Enumerator – Reader 架构
    Enumerator - Reader architecture
  - Source Reader 的线程模型
    Source Reader Threading model
  - 水印生成
    Watermark generation
  - 任务可协同的算子
    Coordinated operators
  - 状态保存和恢复
    Checkpoint and failover
- 轻松实现生产可用的 Flink Source
  Production-ready Source made easy!

核心抽象
Core abstraction

记录分片
Source Splits

一个有编号的记录集合
An identifiable set of records
读取的进度可以被写入检查点
The reading progress can be checkpointed
包含关于记录分片的所有信息
Include all the information needed for reading

记录分片枚举者
Split Enumerator

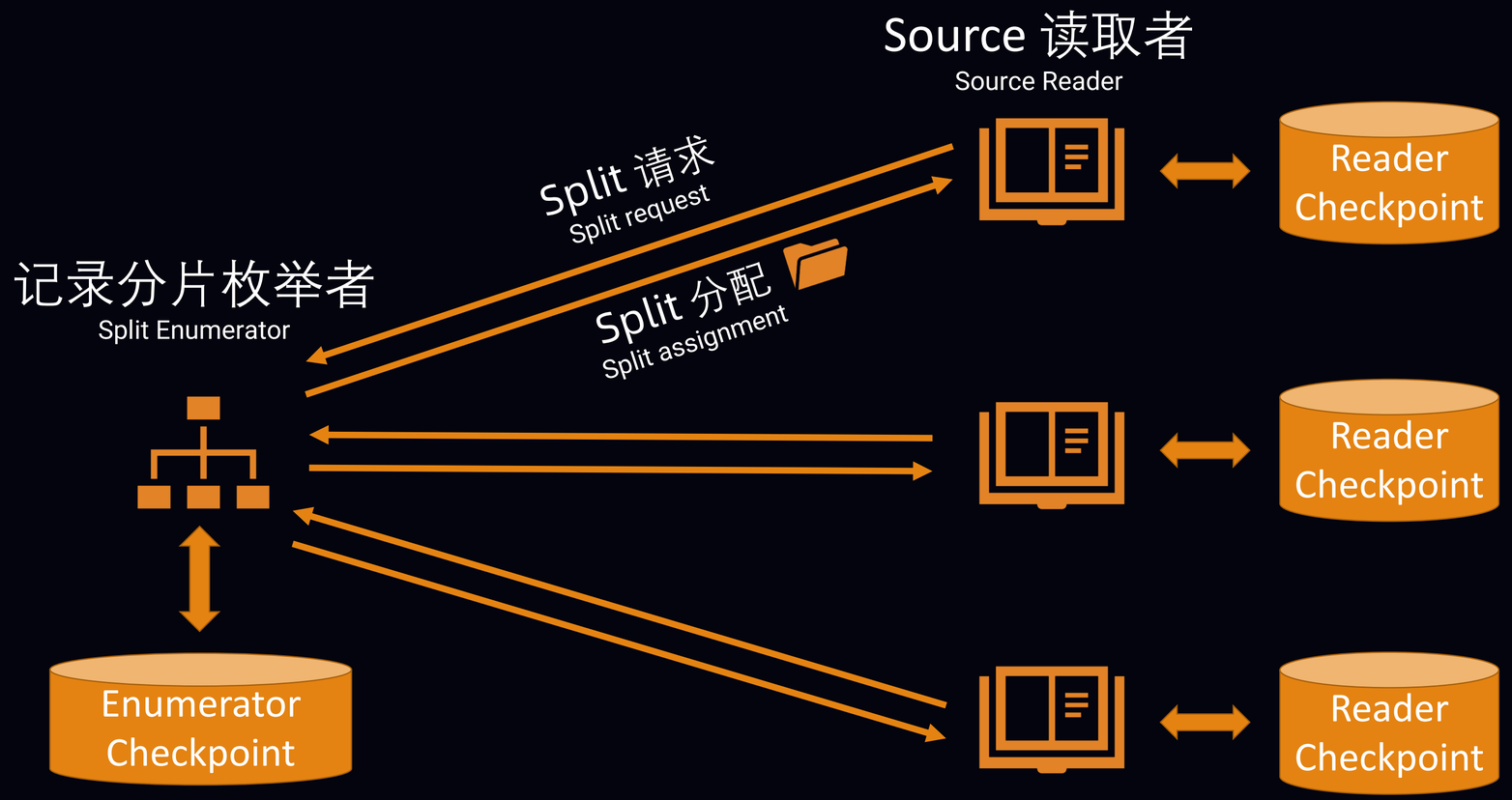
发现记录分片
Discover splits
协调 Source 读取者
Coordinate Source Readers
(例如: 分配记录分片)
(e.g. assign splits)

Source 读取者
Source Reader

从记录分片读取
Read records from splits
产生事件时间水印
Generate watermarks

Enumerator – Reader 架构

Enumerator – Reader Architecture

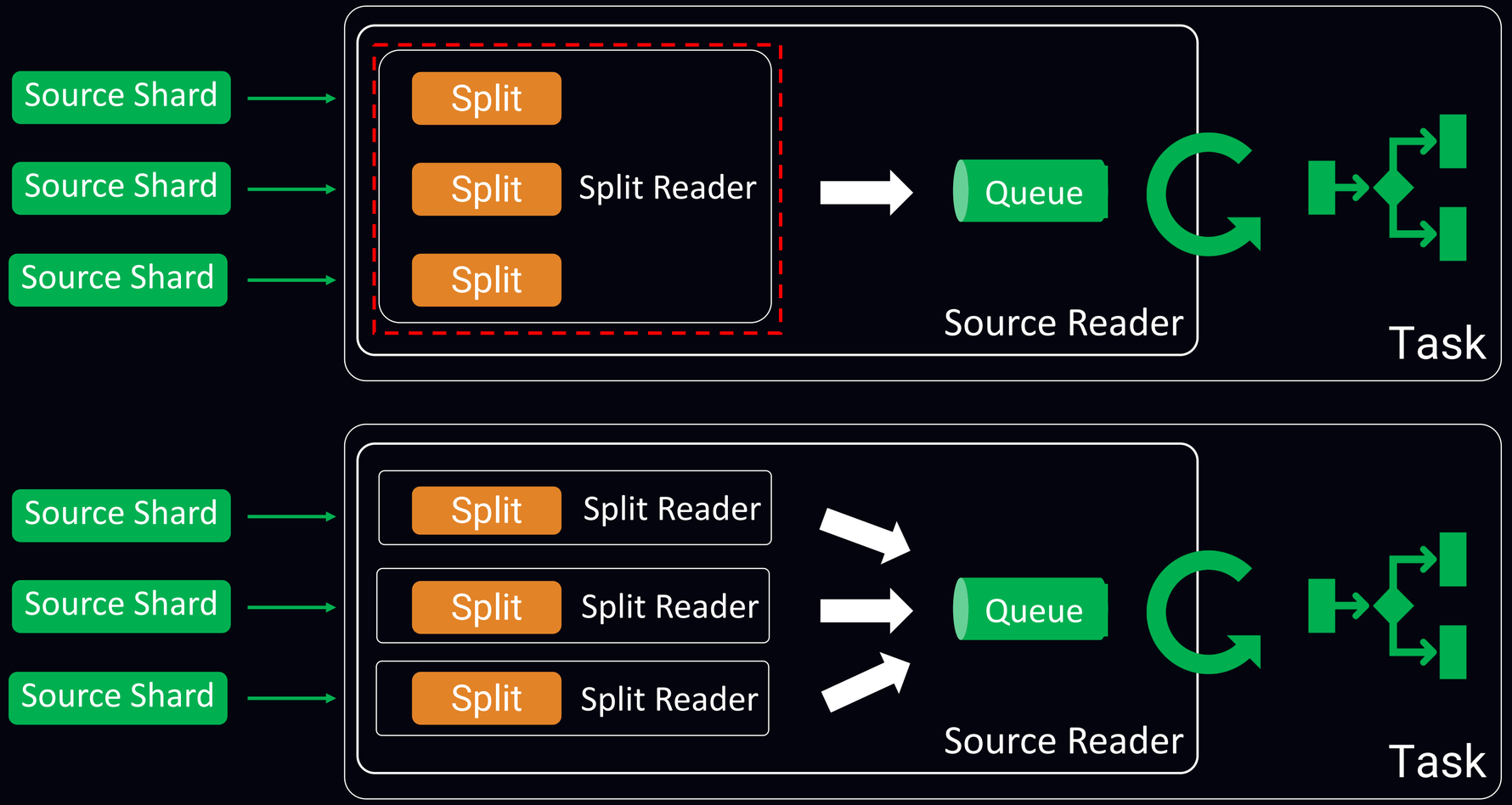


目录
Agenda

- 什么是 Flink Source
  What is a Flink source?
- 为什么需要新的 Source?
  Why new Source API?
- 新 Source 的设计
  Design the new Source
  - 设计目标
    The design goals
  - Enumerator – Reader 架构
    Enumerator - Reader architecture
  - Source Reader 的线程模型
    Source Reader Threading model
  - 水印生成
    Watermark generation
  - 任务可协同的算子
    Coordinated operators
  - 状态保存和恢复
    Checkpoint and failover
- 轻松实现生产可用的 Flink Source
  Production-ready Source made easy!

Source Reader的线程模型

Source Reader Threading Model



案例：顺序分片读取者

Example – Sequential Split Reader

读取者顺序读取被分配到的记录分片

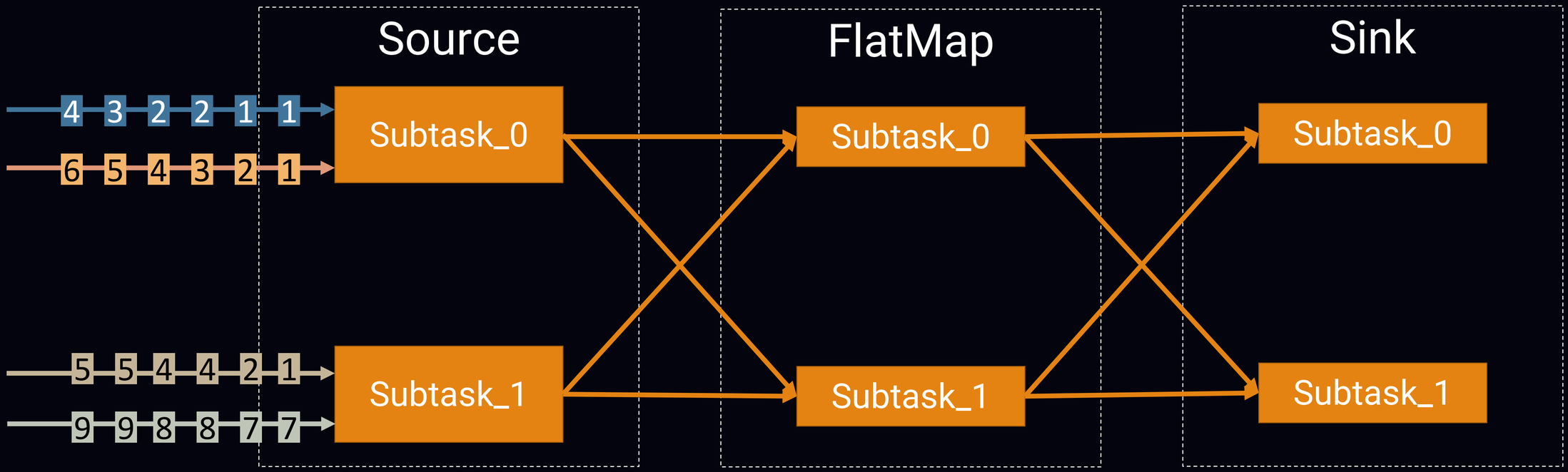
A split reader that reads assigned splits sequentially

目录
Agenda

- 什么是 Flink Source
  What is a Flink source?
- 为什么需要新的 Source?
  Why new Source API?
- 新 Source 的设计
  Design the new Source
  - 设计目标
    The design goals
  - Enumerator – Reader 架构
    Enumerator - Reader architecture
  - Source Reader 的线程模型
    Source Reader Threading model
  - 水印生成
    Watermark generation
  - 任务可协同的算子
    Coordinated operators
  - 状态保存和恢复
    Checkpoint and failover
- 轻松实现生产可用的 Flink Source
  Production-ready Source made easy!

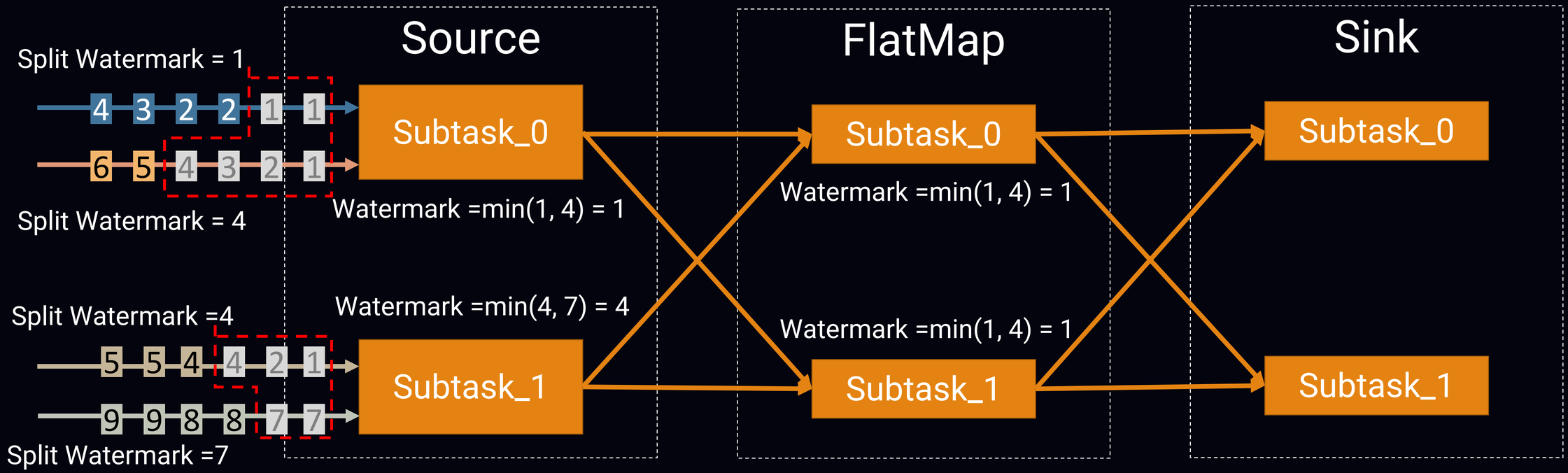
事件时间水印回顾

Recap on watermark



事件时间水印回顾

Recap on watermark



生成事件时间水印
Watermark generation

- 水印何时生成
  Watermarks are generated
  - 收到一条记录时生成
    On receiving a record
  - 周期性生成
    Periodically
  - Source 空闲时生成
    Idleness
- 为每个记录分片生成独立的水印
  Per split level watermark generation

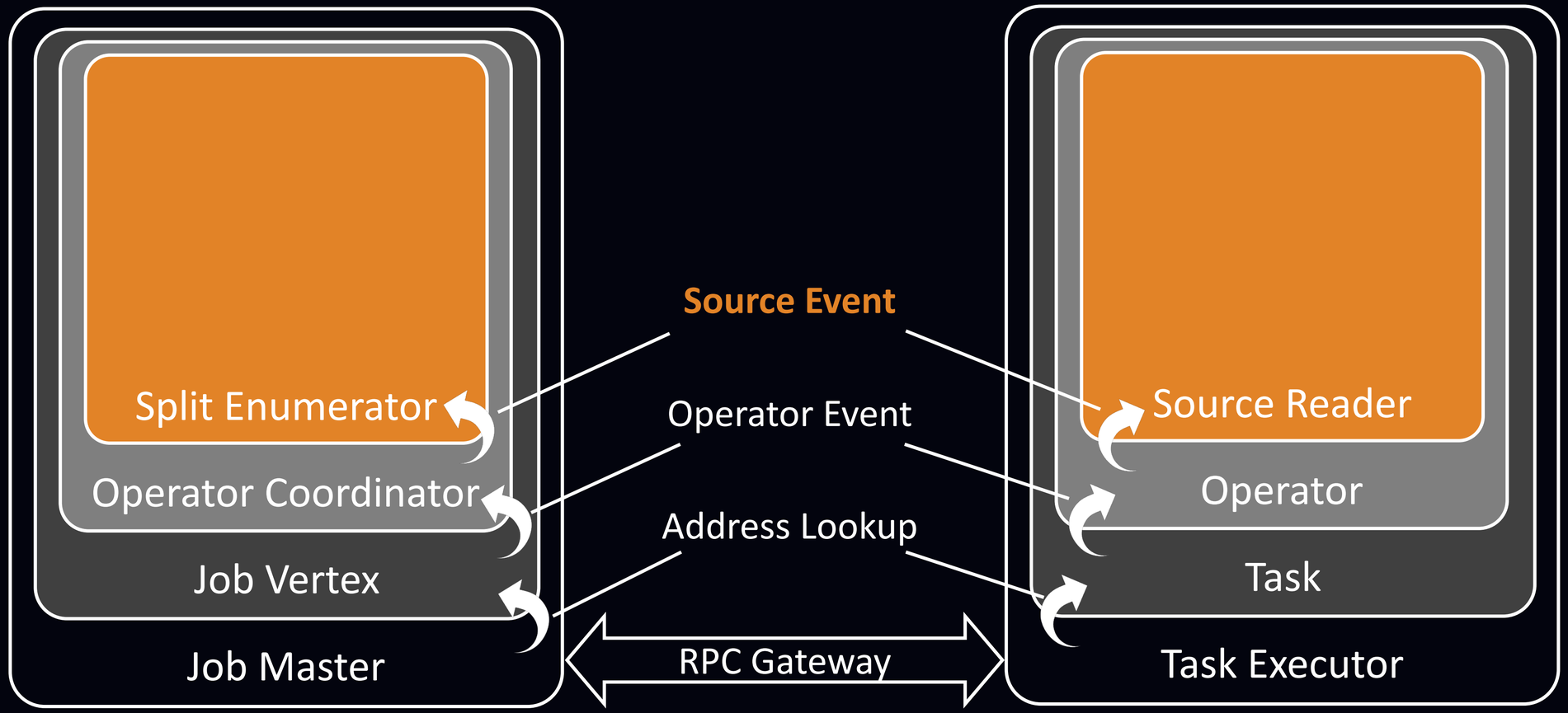
目录
Agenda

- 什么是 Flink Source
  What is a Flink source?
- 为什么需要新的 Source?
  Why new Source API?
- 新 Source 的设计
  Design the new Source
  - 设计目标
    The design goals
  - Enumerator – Reader 架构
    Enumerator - Reader architecture
  - Source Reader 的线程模型
    Source Reader Threading model
  - 水印生成
    Watermark generation
  - 任务可协同的算子
    Coordinated operators
  - 状态保存和恢复
    Checkpoint and failover
- 轻松实现生产可用的 Flink Source
  Production-ready Source made easy!

有协同的算子
Coordinated Operators

- 通用可扩展的协同机制
  A generic extendable mechanism for coordination

有协同的算子
Coordinated Operators

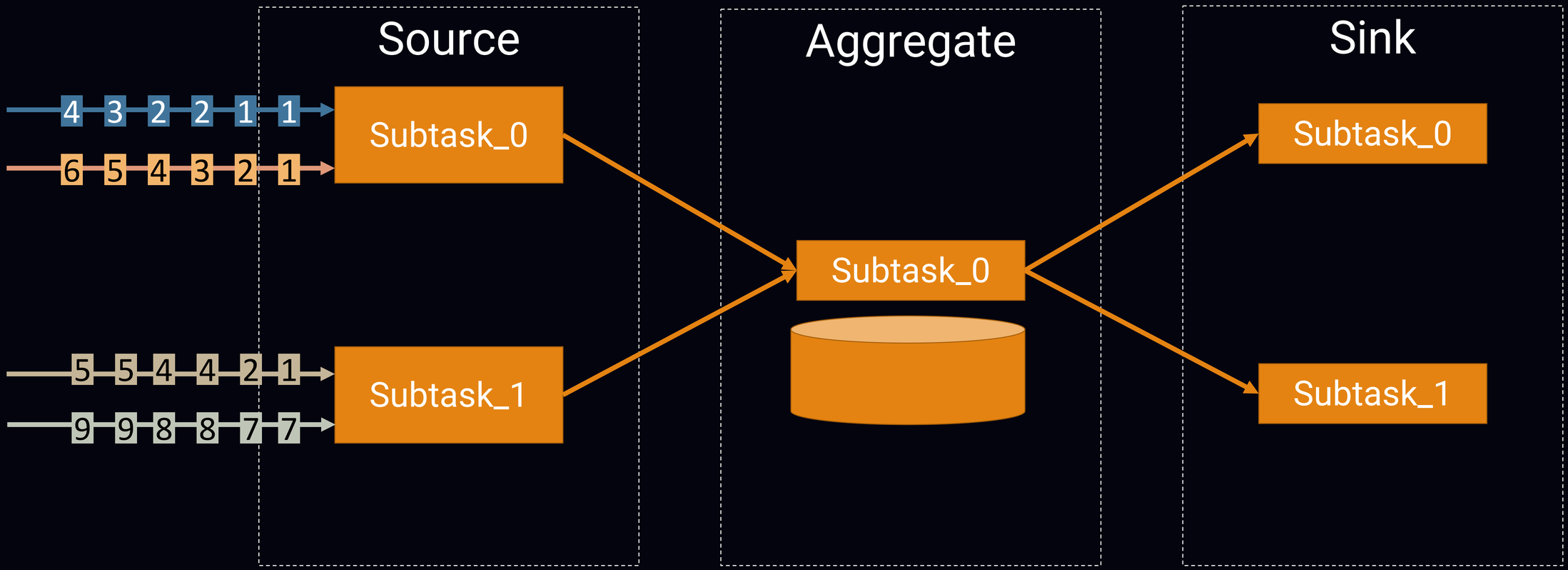


有协同的算子
Coordinated Operators

- 算子事件
  Operator events
  - ReaderRegisterEvent （读取者注册事件）
  - ReaderFailedEvent（读取者失败事件）
  - AddSplitEvent（分配记录分片事件）
  - RequestSplitEvent（请求记录分片）
  - ...
- Source 事件
  Source Events
  - Source 实现中客户化的事件
    Custom events by specific source implementation

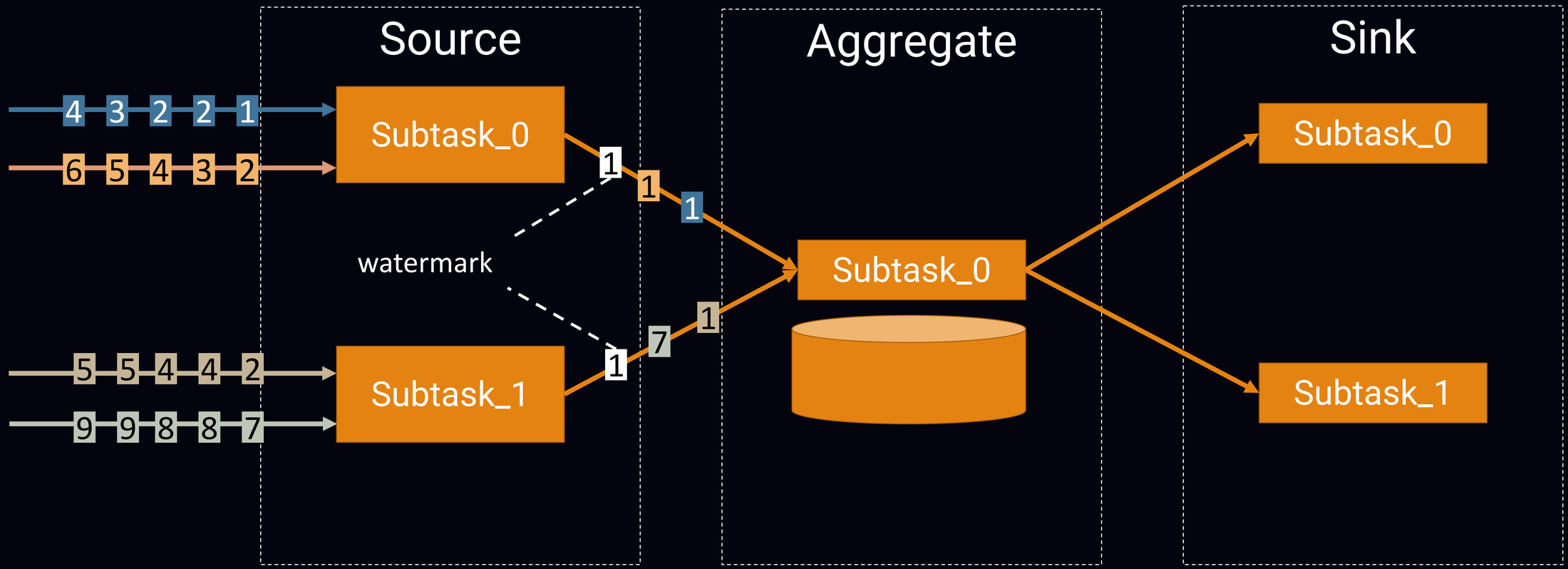
案例：事件时间对齐

Use cases – Event time alignment



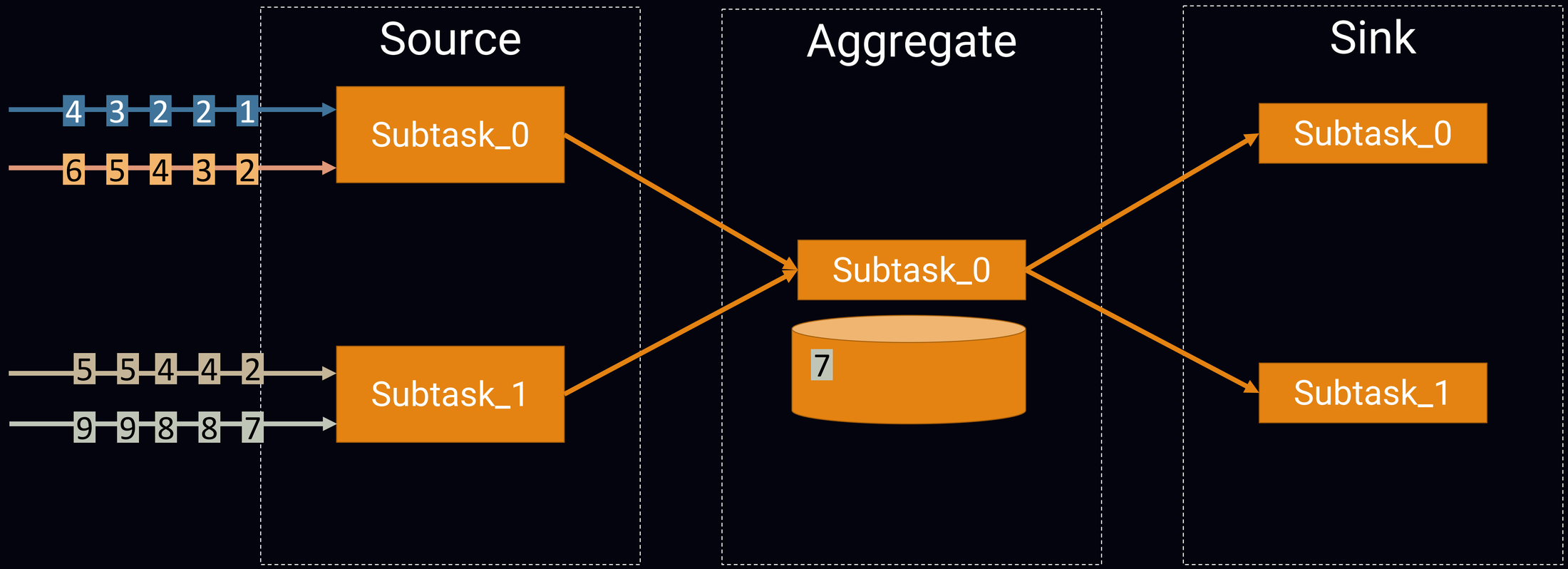
案例：事件时间对齐

Use cases – Event time alignment



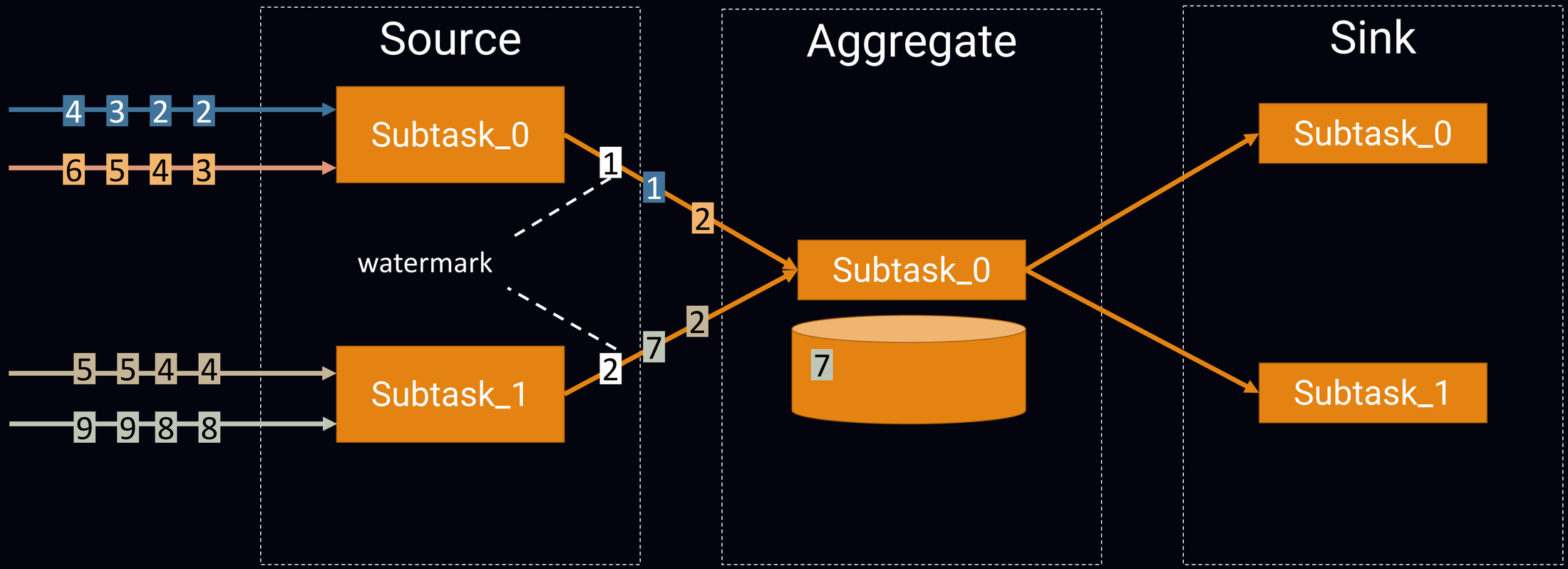
案例：事件时间对齐

Use cases – Event time alignment



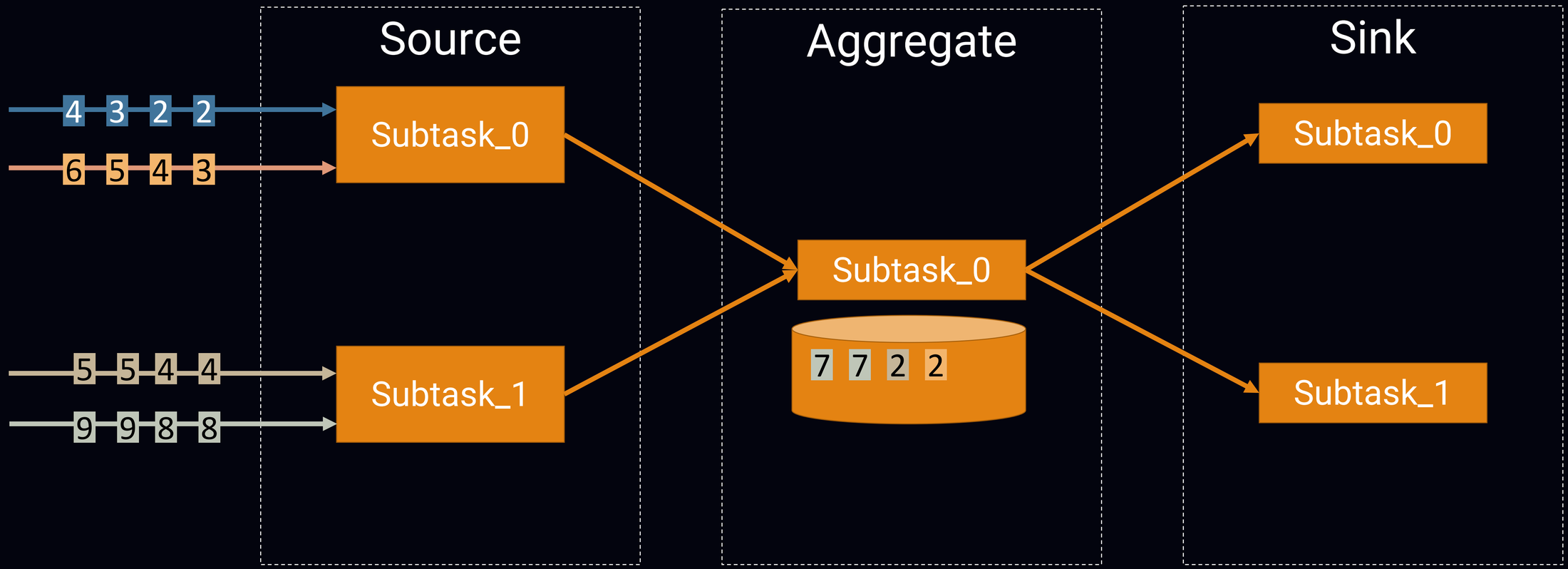
案例：事件时间对齐

Use cases – Event time alignment



案例：事件时间对齐

Use cases – Event time alignment

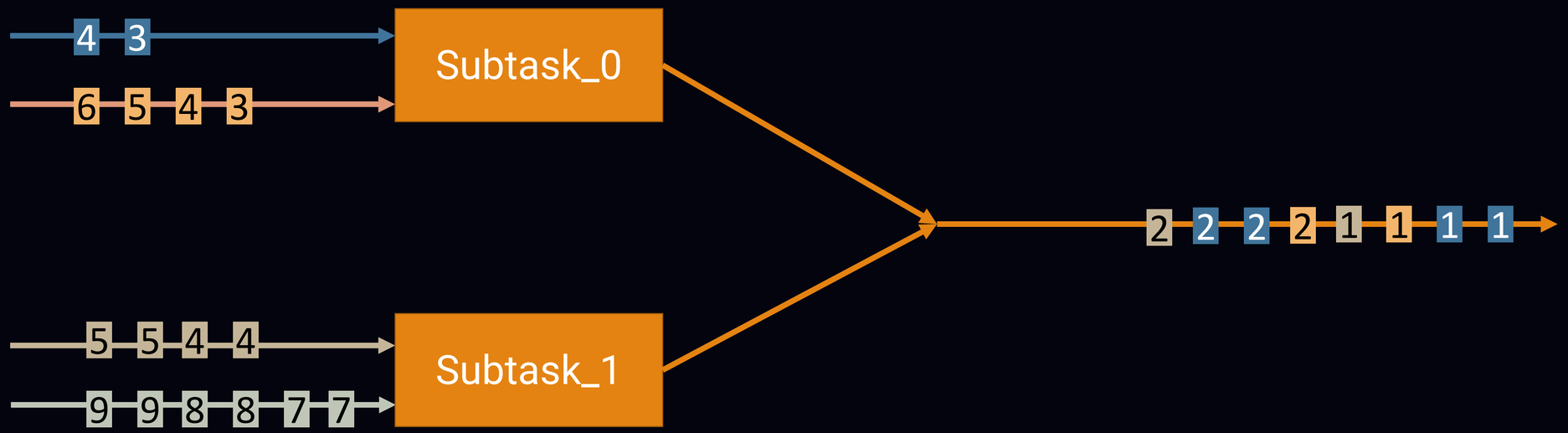


事件时间在事件时间水印之后的记录将被存入 State.

Records beyond event time watermarks are saved to state.

案例：事件时间对齐

Use cases – Event time alignment



Source 只读取事件时间比水印低的记录

Only read records whose event time is below or equals to watermark.

在上图的状态中，Subtask_1 如何知道需要停止读取记录？

How does Subtask_1 know it should stop reading?

案例：事件时间对齐

Use cases – Event time alignment

- 事件时间水印传播
  Watermark propagation

更多案例
More use cases

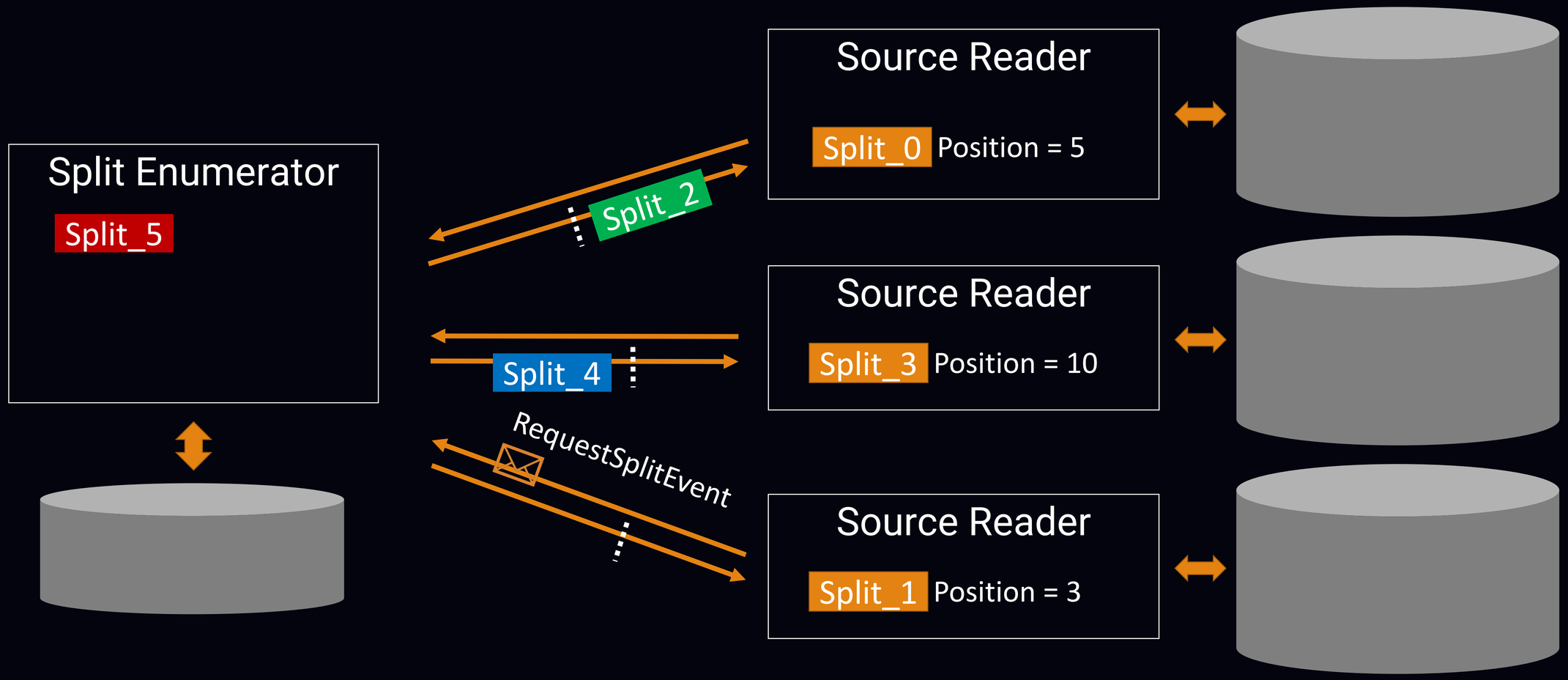
- 负载均衡
  Workload balance
- 客户化流控策略
  Custom flow control policy
- …
  …

目录
Agenda

- 什么是 Flink Source
  What is a Flink source?
- 为什么需要新的 Source?
  Why new Source API?
- 新 Source 的设计
  Design the new Source
  - 设计目标
    The design goals
  - Enumerator – Reader 架构
    Enumerator - Reader architecture
  - Source Reader 的线程模型
    Source Reader Threading model
  - 水印生成
    Watermark generation
  - 任务可协同的算子
    Coordinated operators
  - 状态保存和恢复
    Checkpoint and failover
- 轻松实现生产可用的 Flink Source
  Production-ready Source made easy!

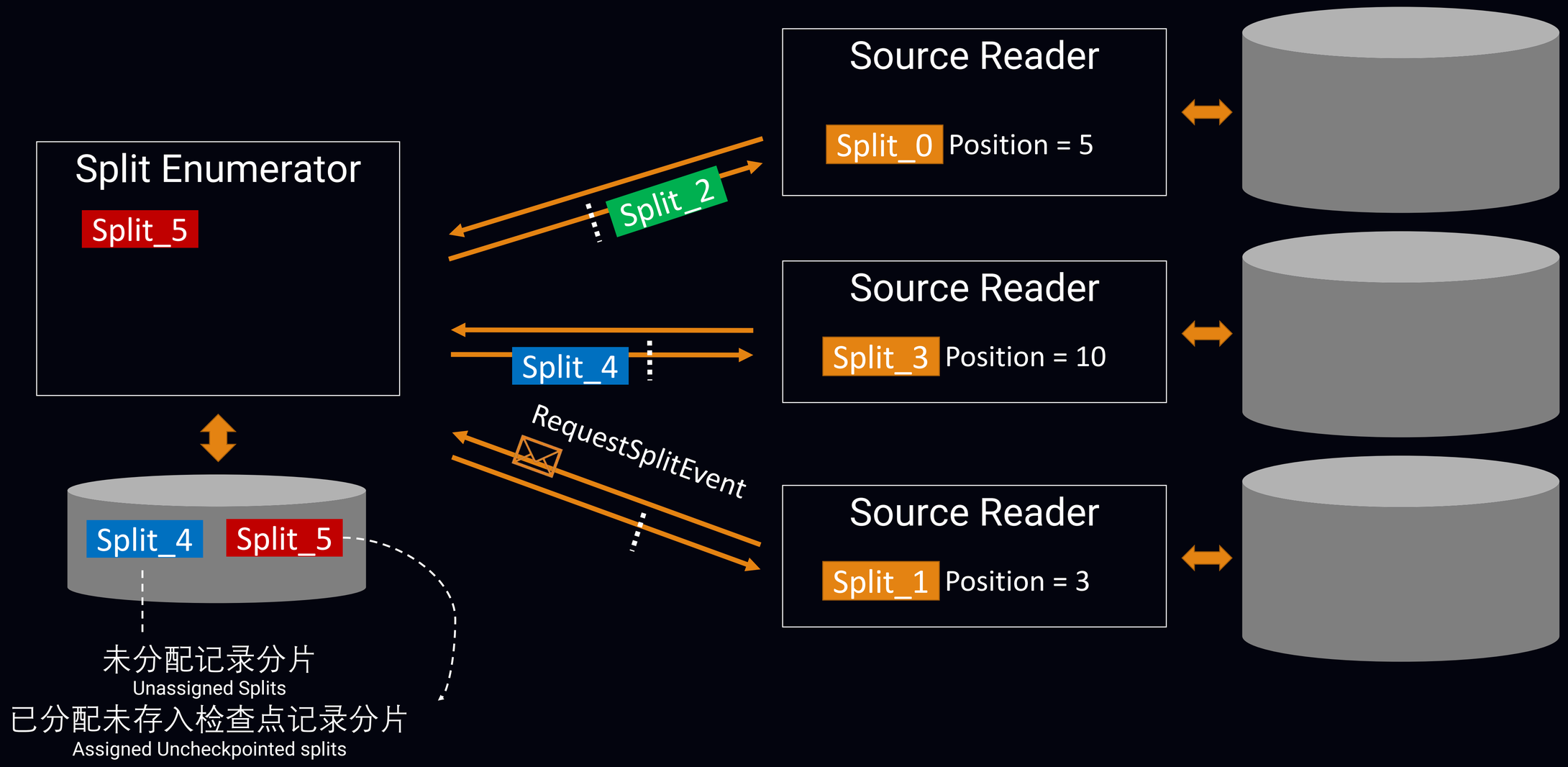
创建检查点

Take a checkpoint



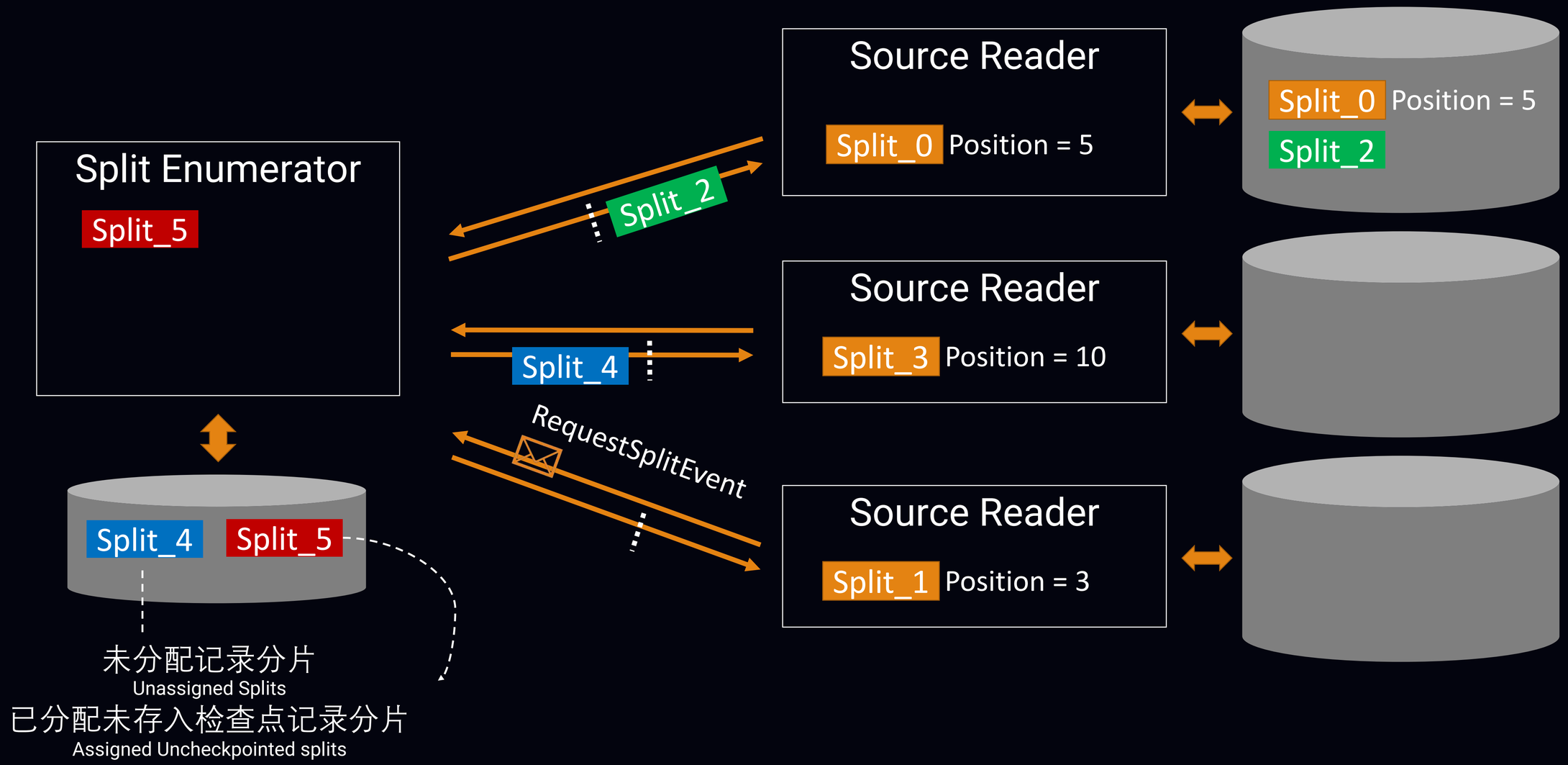
创建检查点

Take a checkpoint



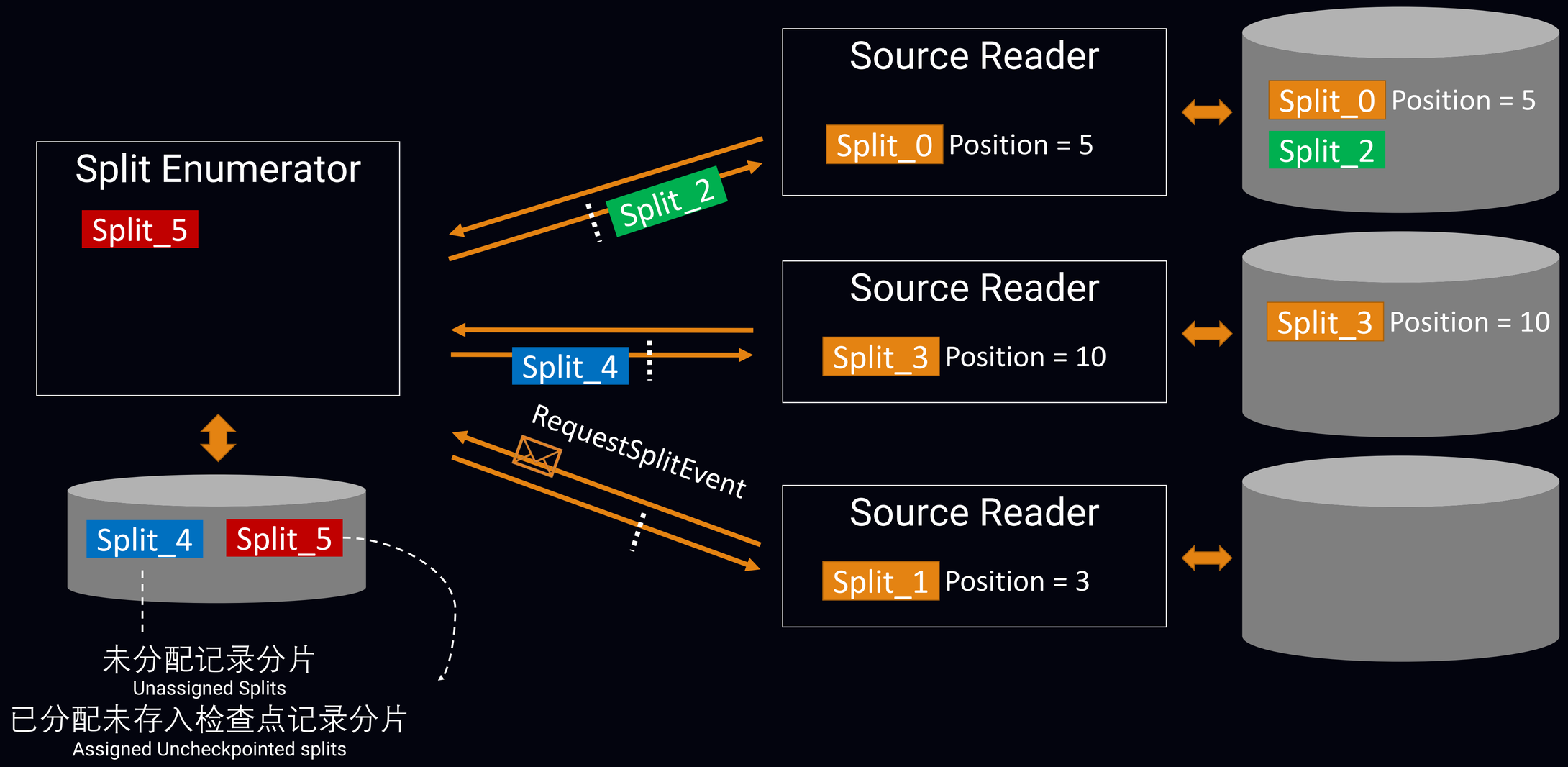
创建检查点

Take a checkpoint



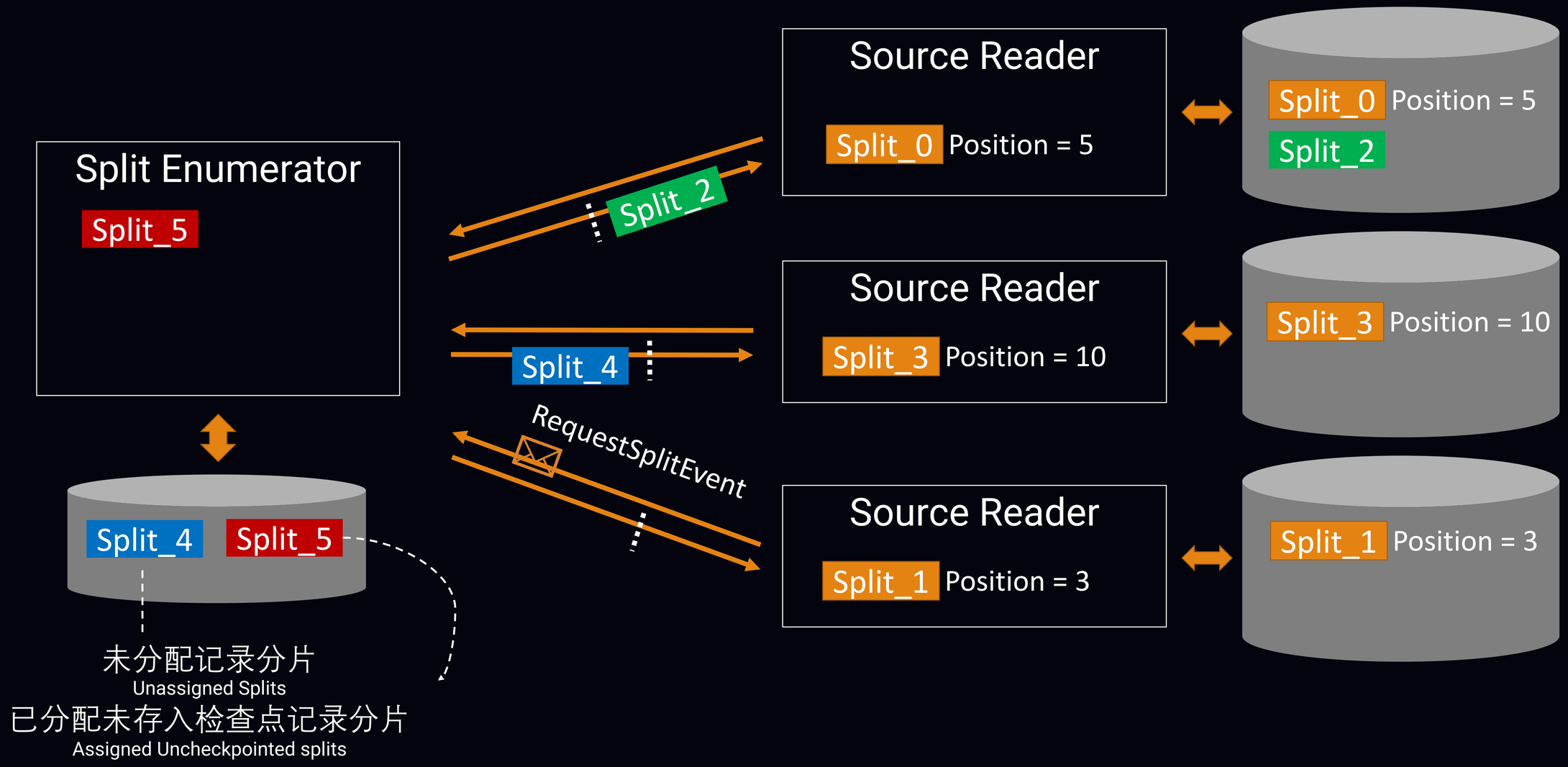
创建检查点

Take a checkpoint

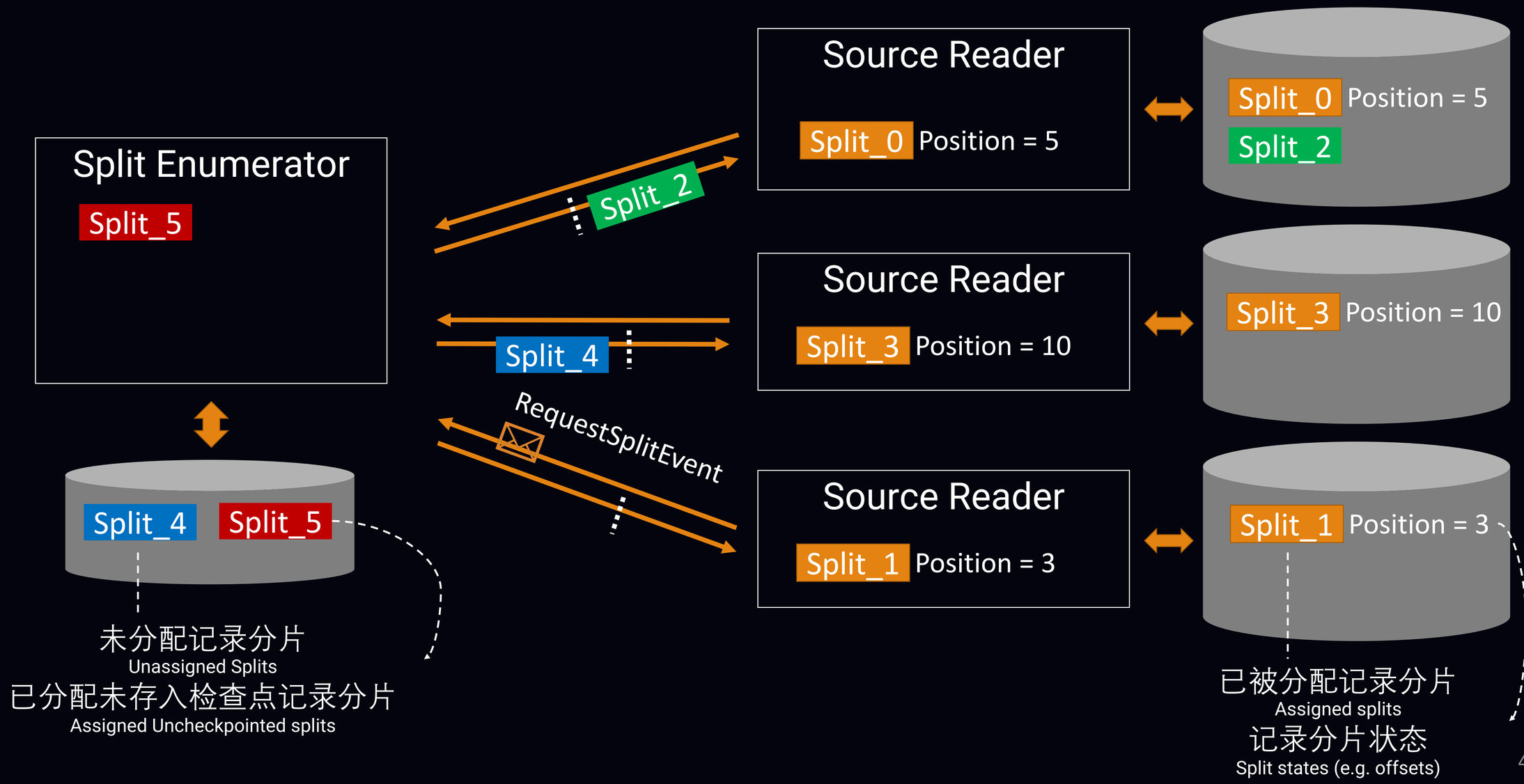


创建检查点

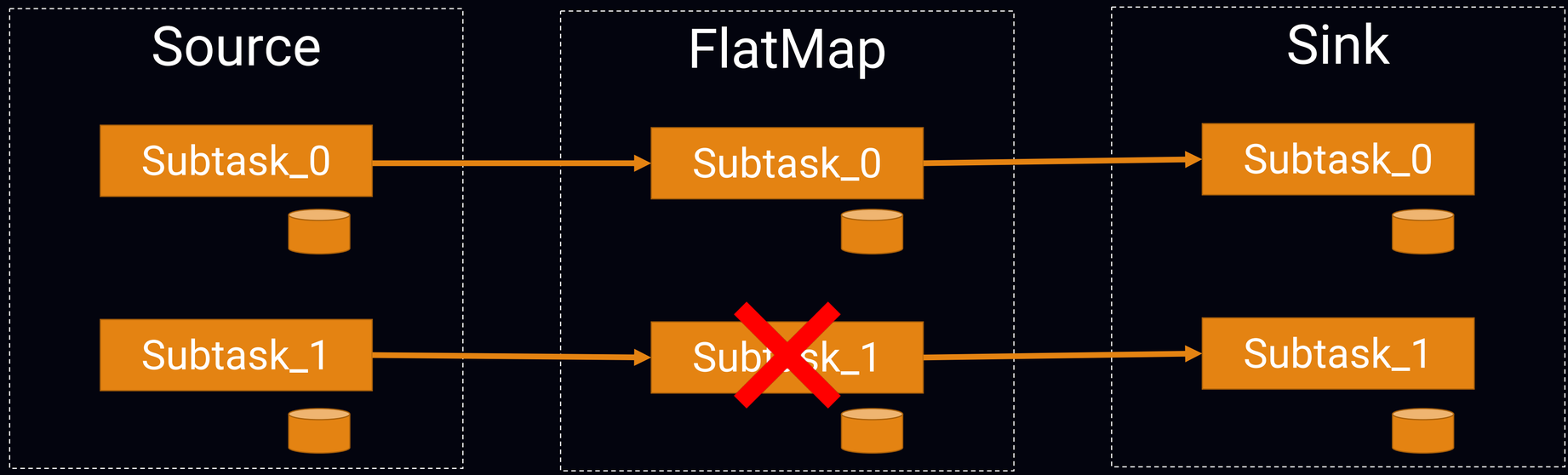
Take a checkpoint



创建检查点
Take a checkpoint

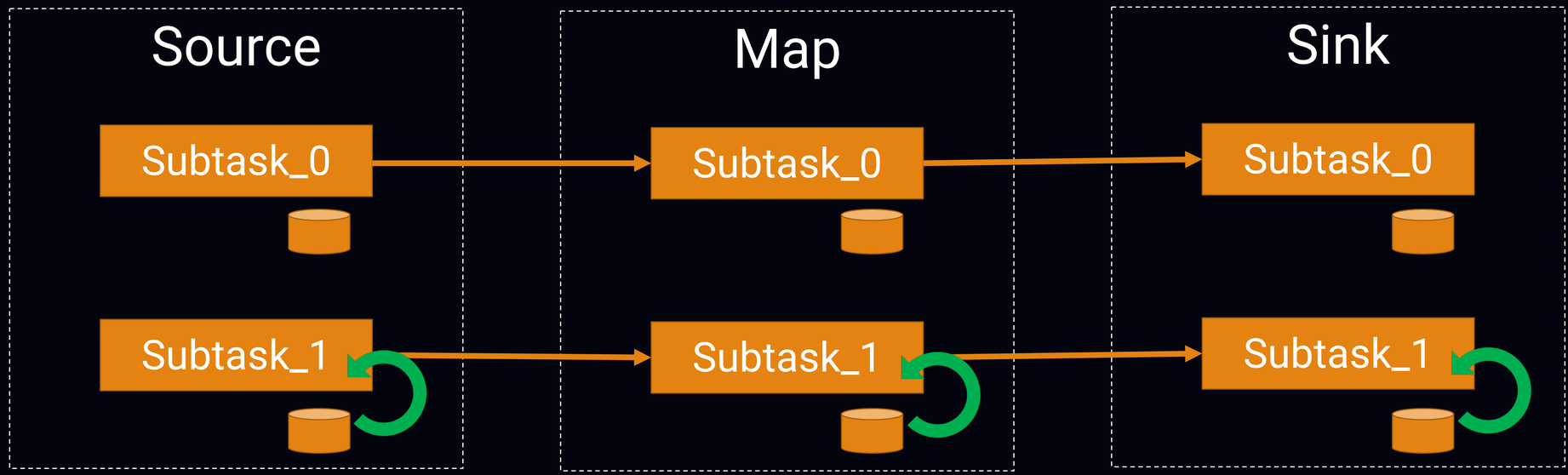


出错恢复
Failover



无协同的 Source 恢复
Failure boundary becomes larger.

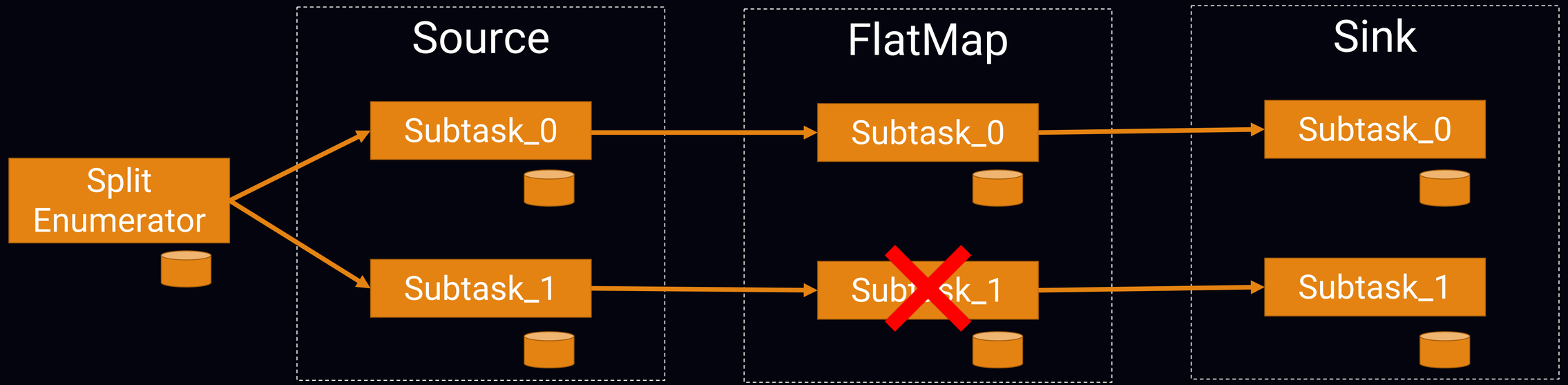
出错恢复
Failover



无协同的 Source 恢复

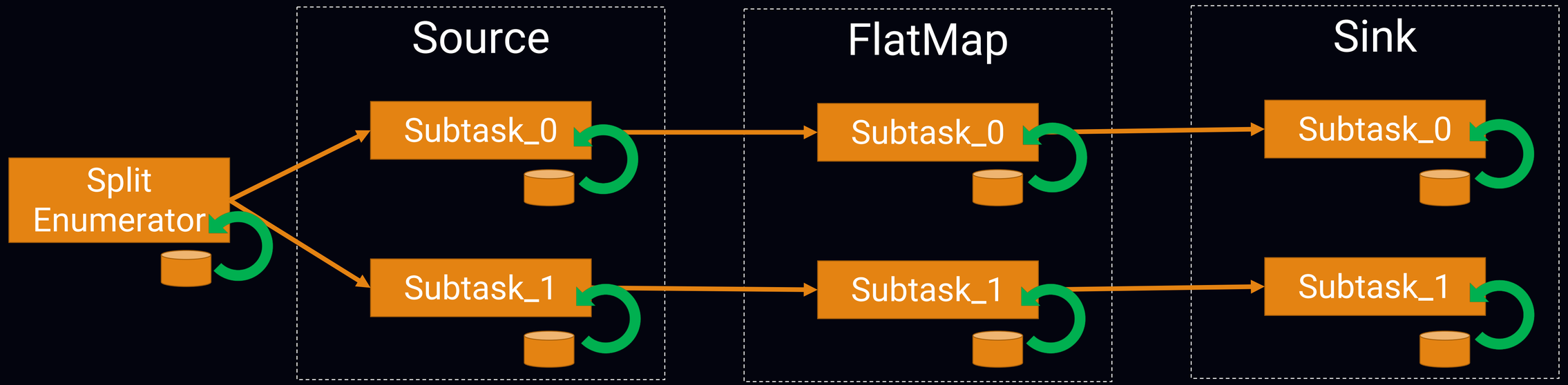
Failure boundary becomes larger.

出错恢复
Failover



有协同 Source 的恢复
Failure boundary becomes larger.

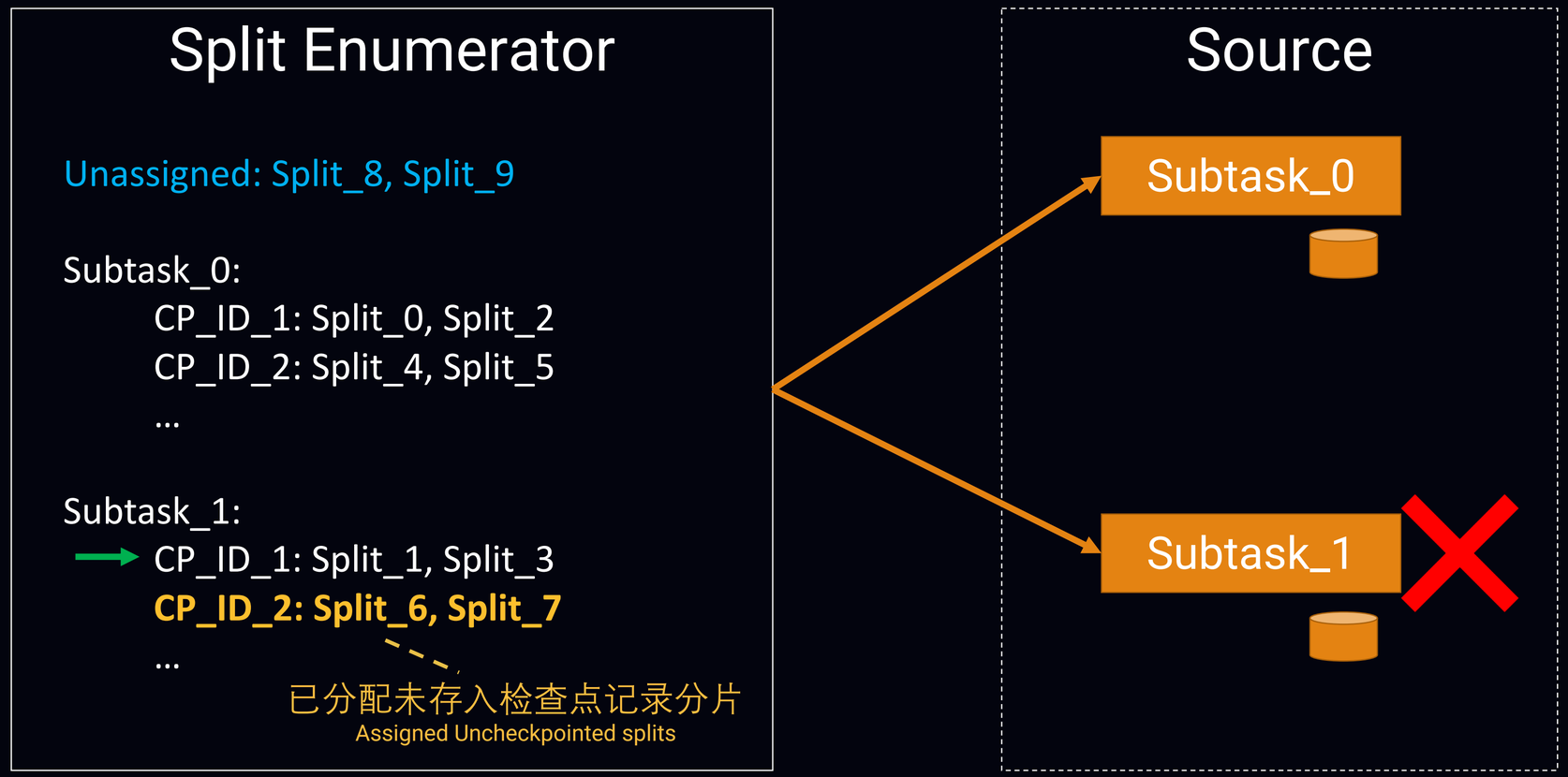
出错恢复
Failover



恢复影响范围变大
Failure boundary becomes larger.

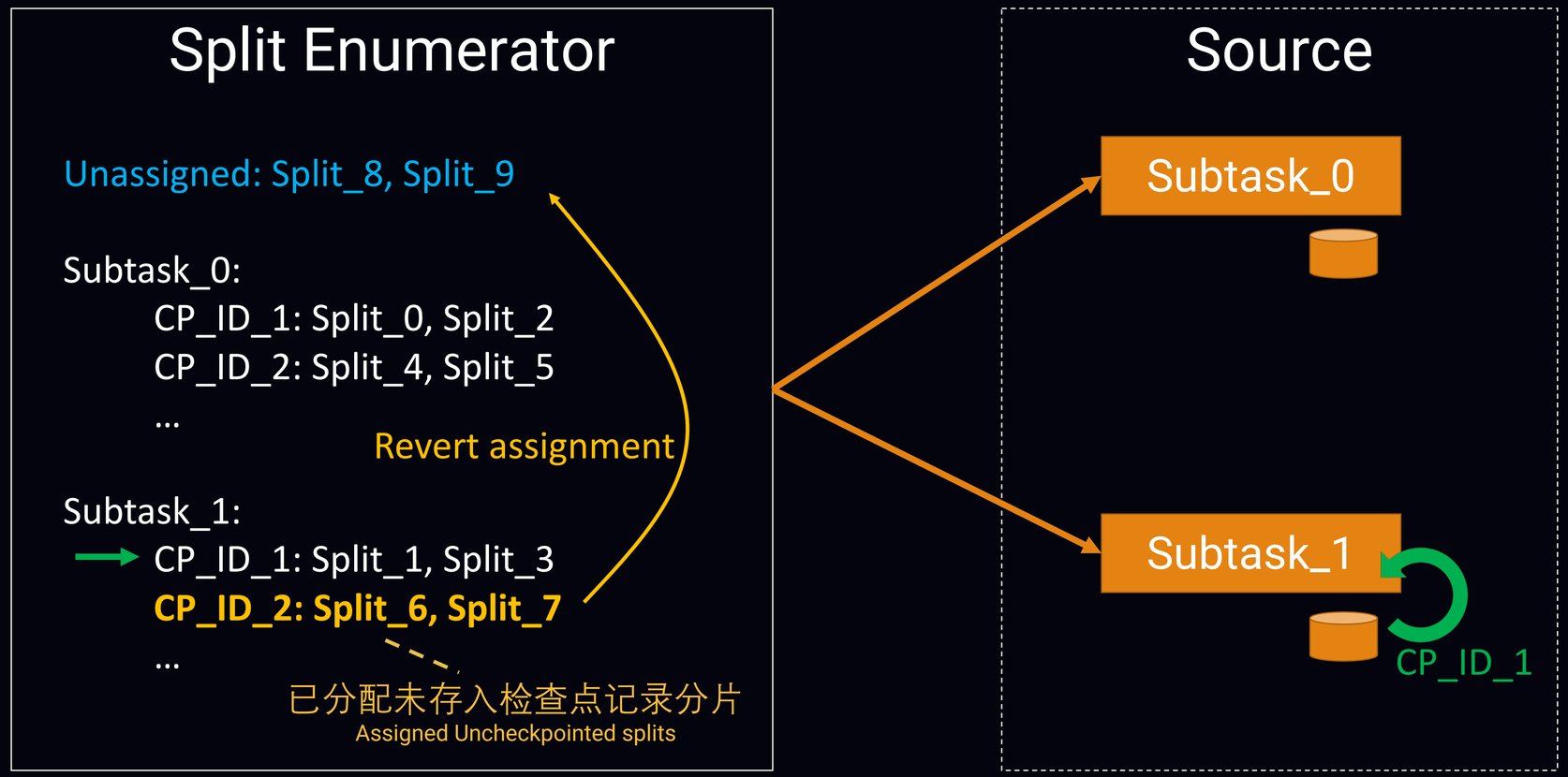
记录分片枚举者局部状态恢复

Enumerator partial state recovery



记录分片枚举者局部状态恢复

Enumerator partial state recovery



目录
Agenda

- 什么是 Flink Source
  What is a Flink source?
- 为什么需要新的 Source?
  Why new Source API?
- 新 Source 的设计
  Design the new Source
  - 设计目标
    The design goals
  - Enumerator – Reader 架构
    Enumerator - Reader architecture
  - Source Reader 的线程模型
    Source Reader Threading model
  - 水印生成
    Watermark generation
  - 任务可协同的算子
    Coordinated operators
  - 状态保存和恢复
    Checkpoint and failover
- 轻松实现生产可用的 Flink Source
  Production-ready Source made easy!

轻松实现 Flink Source!

Source is made easy!

- 新 Source 为实现者提供了
  The new source handles
  - 线程同步
    Synchronization
  - 事件时间水印生成
    Watermark generation
  - 子任务粒度的检查点和出错恢复
    Subtask level checkpoint and failover
  - 多种线程模型
    Various threading model out-of-the-box
  - 良好的可扩展性
    Good extensibility
    - 有协同的算子支持
      New primitive of coordinated operator

FLINK FORWARD

THANKS

FLINK FORWARD

招人！招人！招人！
WE ARE HIRING!!!

Jiangjie.qj@Alibaba-inc.com