Event Delivery

Overview

This assignment gives you the opportunity to implement one of the most interesting aspects of rudder-server.

We ask you to implement a system that receives events from multiple users from an HTTP endpoint and delivers (broadcast) them in multiple destinations. Following the technical requirements listed below.

Requirements

At-least-once delivery of events to multiple destinations while maintaining order for the same userID.

Durability

Once an ingested event is accepted, it will remain in the system even if the system crashes or until it is delivered, or it has been exhaustively retried.

At-least-once delivery

Assume we have minimal control over the supported destinations.

Retry backoff and limit

Messages should be retried using a backoff algorithm, after a number of delivery attempts, the event should be drained from the system.

Maintaining order

Events of the same user should always be delivered in the order the system received them. Events have a userID property that uniquely identifies the user.

Delivery Isolation

Delays or failures with event delivery of a single destination should not affect ingestion or other delivery on other destinations.

Bonus: Delays or failures with event delivery of a userId should not affect the delivery of other events.

Implementation

- Write your implementation **in Go**.
- **Use Kafka** as the persistent component in your solution.
- Delivery to actual destinations is beyond the scope of this assignment. You can use mock destinations and **emulate random failure or delays**, to demonstrate your solution.
- You can use the following struct for the **event** :

```
struct { UserID string; Payload string}.
```