# End-to-End encrypted messaging with Ockam Streams and Kafka

Daniil Fedotov daniil@ockam.io

# Why streams?

## Problem definition

Goal:

- Reliable delivery: at-least-once model

Constrains:

- Unreliable network
- Offline devices

Solution:

- Stream processing

# Why streams?

## Streaming brokers

Reliable, trusted and widely adopted solutions

- Kafka
- Kinesis
- Apache Pulsar
- etc.

# Why streams?

## Challenges integrating with IoT

- Client is not implemented for a platform
- Support of heterogenous network
- Additional configuration and credential to provision to devices
- Challenges integrating End-to-End encryption

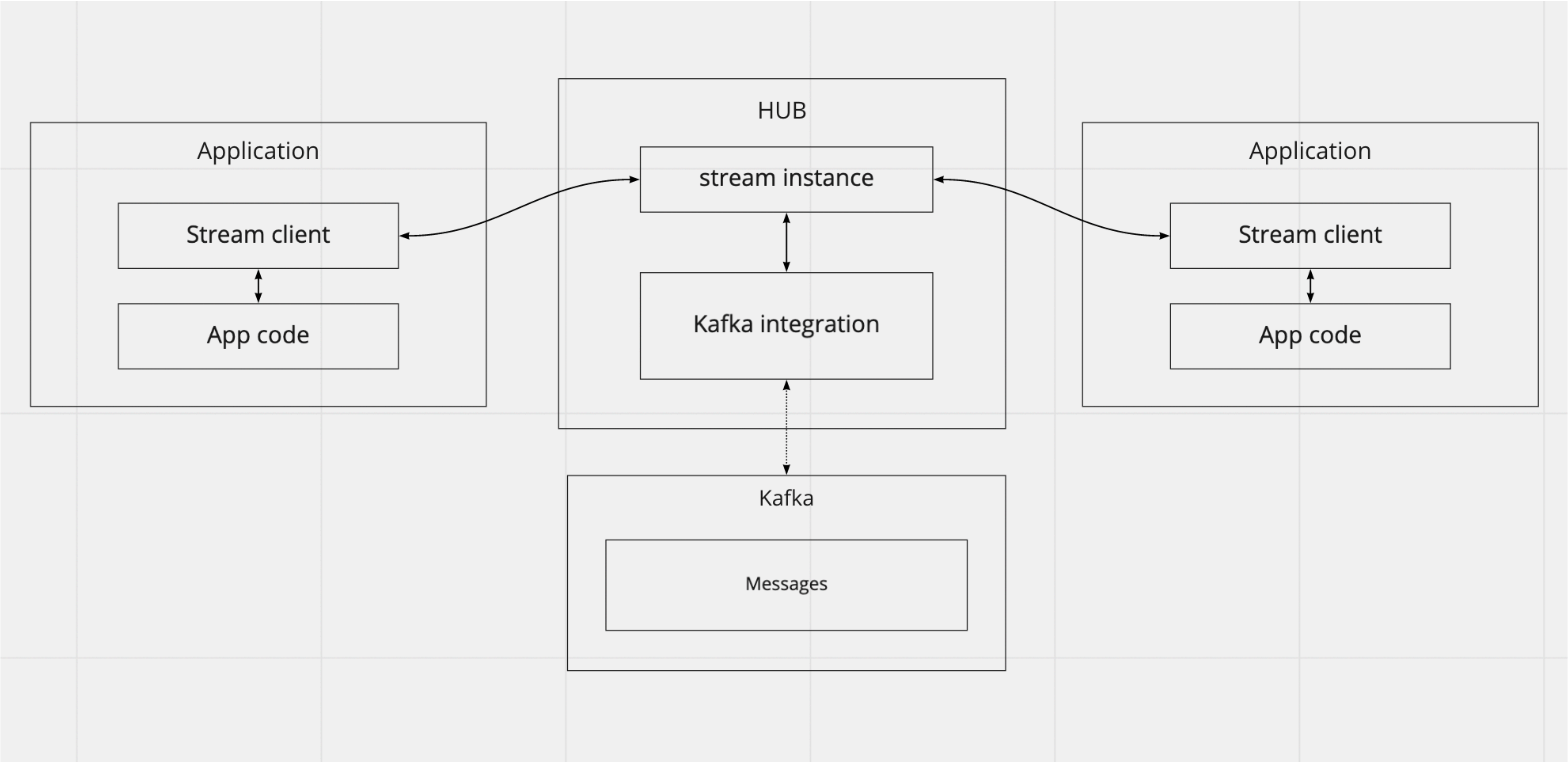
# Ockam Streams

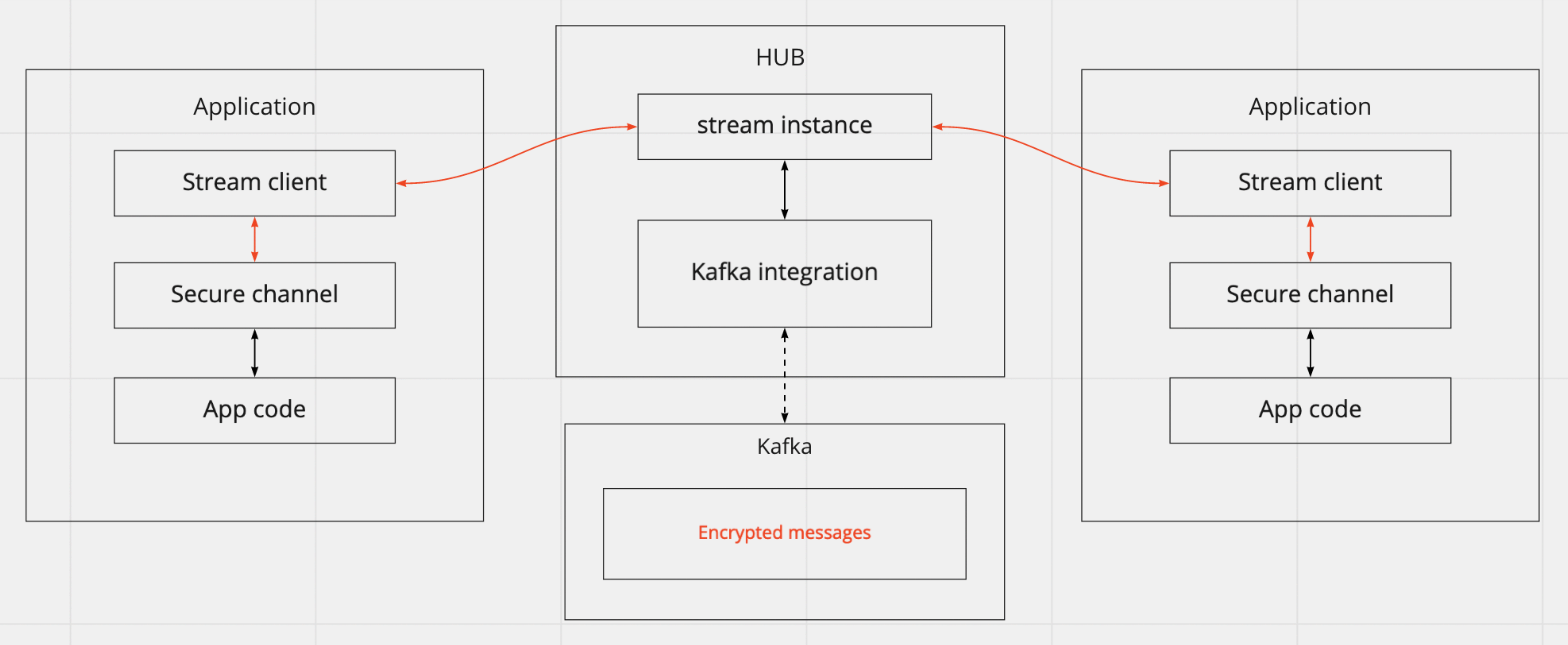
- Minimal streaming API using PULL model
- Integrating with streaming brokers on the server-side
- Using Ockam Routing protocol
- Compatible with Ockam Secure Channels
- Currently targeting Kafka as a pluggable backend

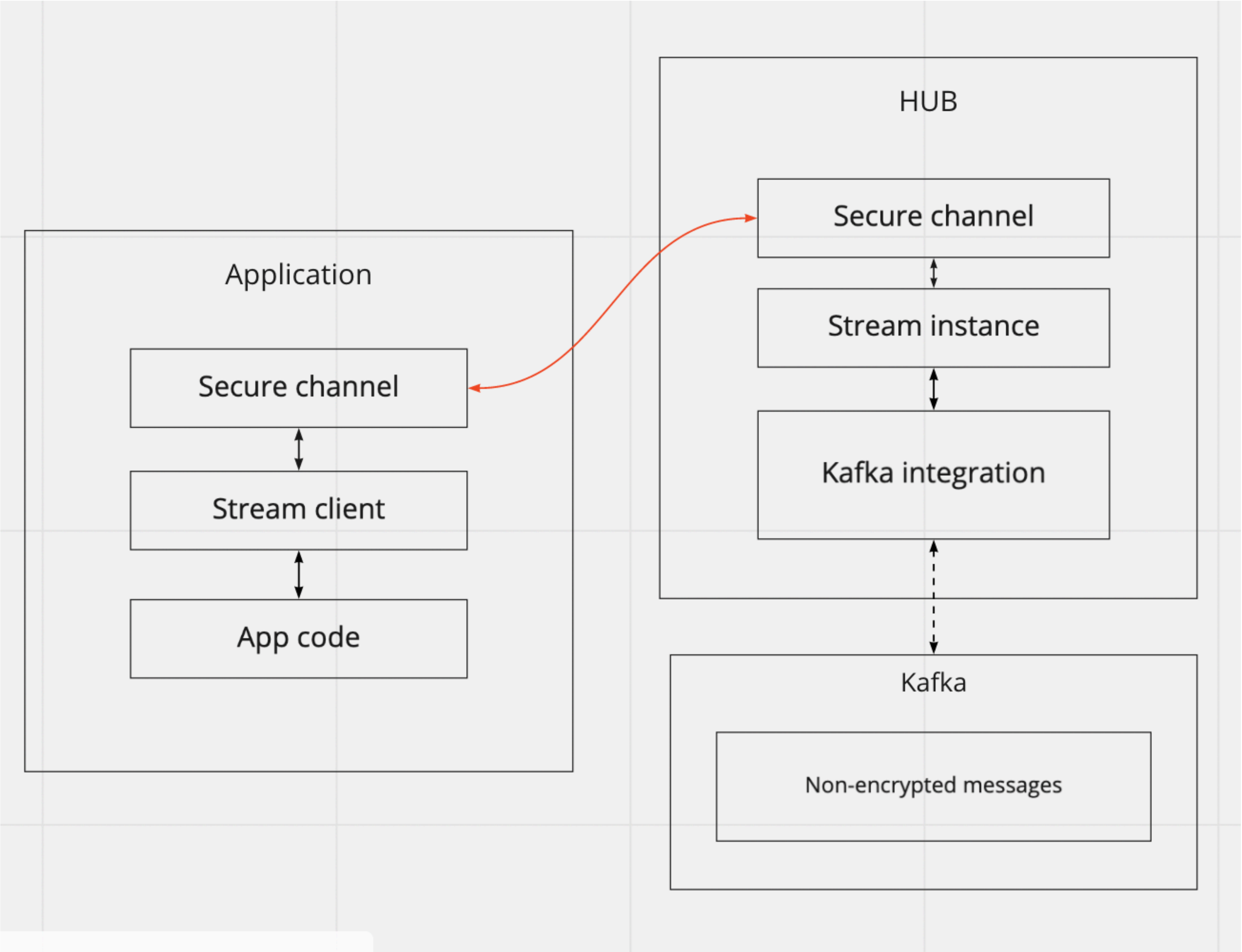
# Ockam Stream protocol

https://github.com/ockam-network/proposals/tree/main/design/0009-stream-protocol

- Create: create a stream instance (kafka topic create)
- Push: put message into a stream (kafka produce)
- Pull: retrieve a set of messages (kafka fetch)
- Index Get: get a current offset for a consumer (kafka offset_get)
- Index Save: save an offset for a consumer (kafka offset_commit)


HUB
stream instance
Kafka integration
Application
Stream client
App code
Application
Stream client
App code
Kafka
Messages


HUB
stream instance
Kafka integration
Application
Stream client
Secure channel
App code
Application
Stream client
Secure channel
App code
Kafka
Encrypted messages


HUB
Secure channel
Stream instance
Kafka integration
Application
Secure channel
Stream client
App code
Kafka
Non-encrypted messages

# Demo time

Clients:

- Elixir:

  https://github.com/ockam-network/ockam/tree/develop/implementations/elixir/ockam/ockam/lib/ockam/stream/client

- Rust:

  https://github.com/ockam-network/ockam/tree/spacekookie/rust-stream-protocol/implementations/rust/ockam/ockam/src/stream

# Demo time

## Elixir stream API demo

You can find the examples code here:

https://github.com/ockam-network/ockam/tree/develop/implementations/elixir/ockam/ockam/lib/ockam/examples/stream

# Integrating brokers

- 2 Elixir modules with 4 and 3 callbacks:

https://github.com/ockam-network/ockam/blob/develop/implementations/elixir/ockam/ockam/lib/ockam/stream/storage.ex

https://github.com/ockam-network/ockam/blob/develop/implementations/elixir/ockam/ockam/lib/ockam/stream/index/storage.ex

- Kafka integration:

https://github.com/ockam-network/ockam/tree/develop/implementations/elixir/ockam/ockam_kafka

# Closing notes

We achieved:

- Reliable message delivery (at-least-once)
- With network interruptions
- Network agnostic with Ockam Routing
- End-to-End encrypted with Ockam Secure Channels