#TGIP-CN EP-010

Protocol Handler & Kafka-on-Pulsar

Who are we?

- Sijie Guo (@sijieg)
- Co-Founder & CEO, StreamNative
- PMC Member of Pulsar/BookKeeper
- Ex Co-Founder, Streamlio
- Ex-Twitter, Ex-Yahoo
- Work on messaging and streaming data technologies for many years

Who are we?

- Pierre Zemb (@PierreZ)
- OVHcloud™ Tech lead
- Working around distributed systems
- Newcomer as an Apache contributor
- Involved into local communities

Agenda

- What is Apache Pulsar?
- Why KoP?
- Introduction of protocol handler
- Kafka VS Pulsar, the protocol version
- How we implement KoP
- Demo
- Roadmap
- Q&A

What is Apache Pulsar?

Flexible pub/sub messaging backed by durable log storage

~~Flexible pub/sub messaging~~
~~backed by durable log storage~~

Cloud-Native Event Streaming

Apache Pulsar

- Publish-subscribe: unified messaging model (streaming + queueing)
- Infinite event stream storage: Apache BookKeeper + Tiered Storage
- Connectors: ingest events without writing code
- Process events in real-time
  - Pulsar Functions for serverless / lightweight computation
  - Spark / Flink for unified data processing
  - Presto for interactive queries

Pulsar Highlights

- Multi-tenancy
- Unified messaging (queuing + streaming)
- Layered Architecture
- Tiered Storage
- Built-in schema support
- Built-in geo-replication

The Need of KoP

- Adoptions
- Inbound requests
- Migration

The Existing Efforts

- Kafka Java Wrapper
- Pulsar IO Connector

Implement Kafka protocol on Pulsar?

- Proxy / Gateway
- Implement Kafka protocol on Pulsar broker

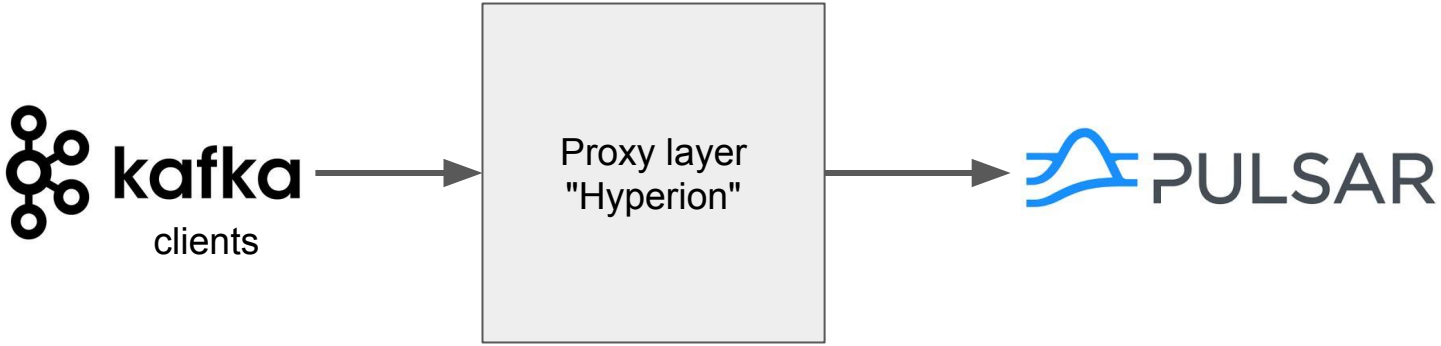
KoP as a proxy, OVHcloud version

We first implemented KoP has a proxy PoC in Rust:

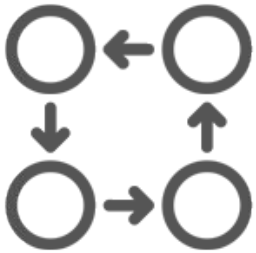
- Rust **async** was out in nightly compiler when we started
- We wanted **no GC** on proxy layers
- Rust has **awesome libraries** at TCP-level

Our goal was to **convert TCP frames** from Kafka to Pulsar

KoP as a proxy, OVHcloud version



KoP as a proxy, OVHcloud version



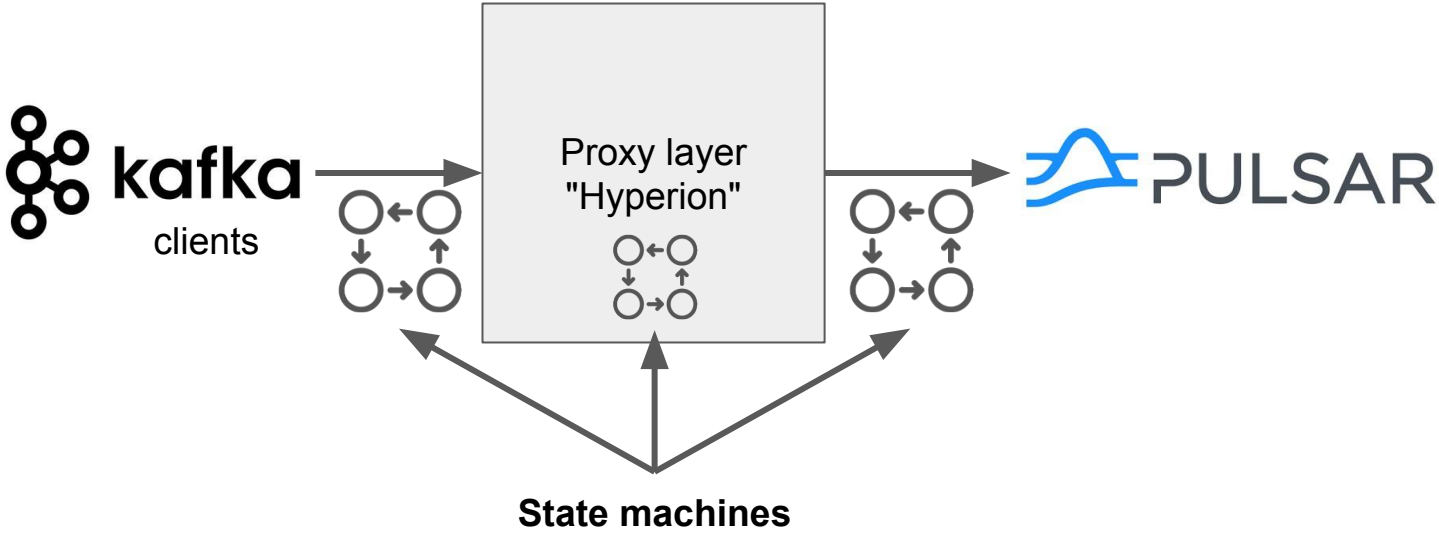
Everything is a state-machine:

- TCP cnx from Kafka clients
- TCP cnx to Pulsar brokers

Those **event-driven finite-state machines** were triggered by **TCP frames** from their **respective protocol.**

A third one was above the two to **provide synchronization**

KoP as a proxy, OVHcloud version



KoP as a proxy, OVHcloud version

Pros

- Working at TCP layer enables **performance**
- nice PoC to **discover** both protocols
- Rust is blazing **fast**
- Proxify production is **easy**
- We **could bump old version** of Kafka frames for old Kafka clients

Cons

- **Rewrite** everything
- Some things were **hard to proxify:**
  - Group coordinator
  - Offsets management
- **Difficult** to open-source (different language)

The group-coordinator/offsets problem

In Kafka, the group coordinator is an **elected actor** within the cluster responsible for:

- assigning partitions to consumers of a consumer group
- managing offsets for each consumer group

In Pulsar, partition assignment is managed **by broker on a per-partition basis.**

Offset management is done by **storing the acknowledgements** in cursors by the owner broker of that partition.

The group-coordinator/offsets problem

In Kafka, the group coordinator is an **elected actor** within the cluster responsible for:

- assigning partitions to consumers of a consumer group
- managing offsets for each consumer group

In Pulsar, partition assignment is managed **by broker on a per-partition basis.**

Offset management is done by **storing the acknowledgements** in cursors by the owner broker of that partition.

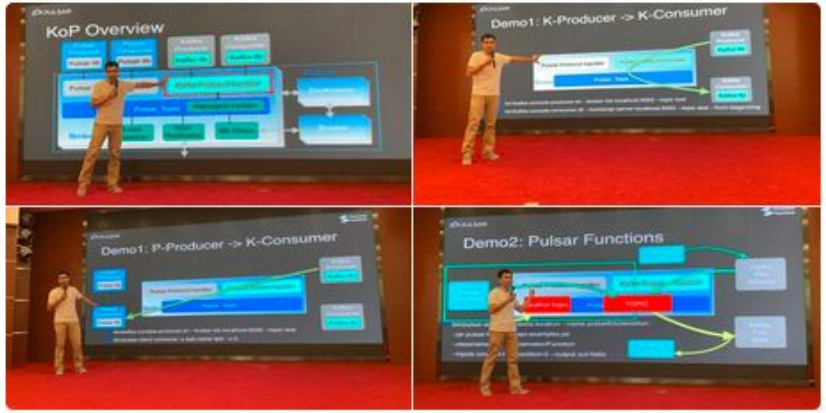
Simulate this at proxy-level **is hard**

(missing low-level info)

And then we saw this 😍

StreamNative
@streamnativeio

Kafka, Pulsar or Kafka + Pulsar? Our own @Jia_Zhai is doing a demo of our new effort Kafka-on-Pulsar (KoP) - adding Kafka protocol support natively in Pulsar broker and looking forward to the collaboration between two messaging ecosystems.



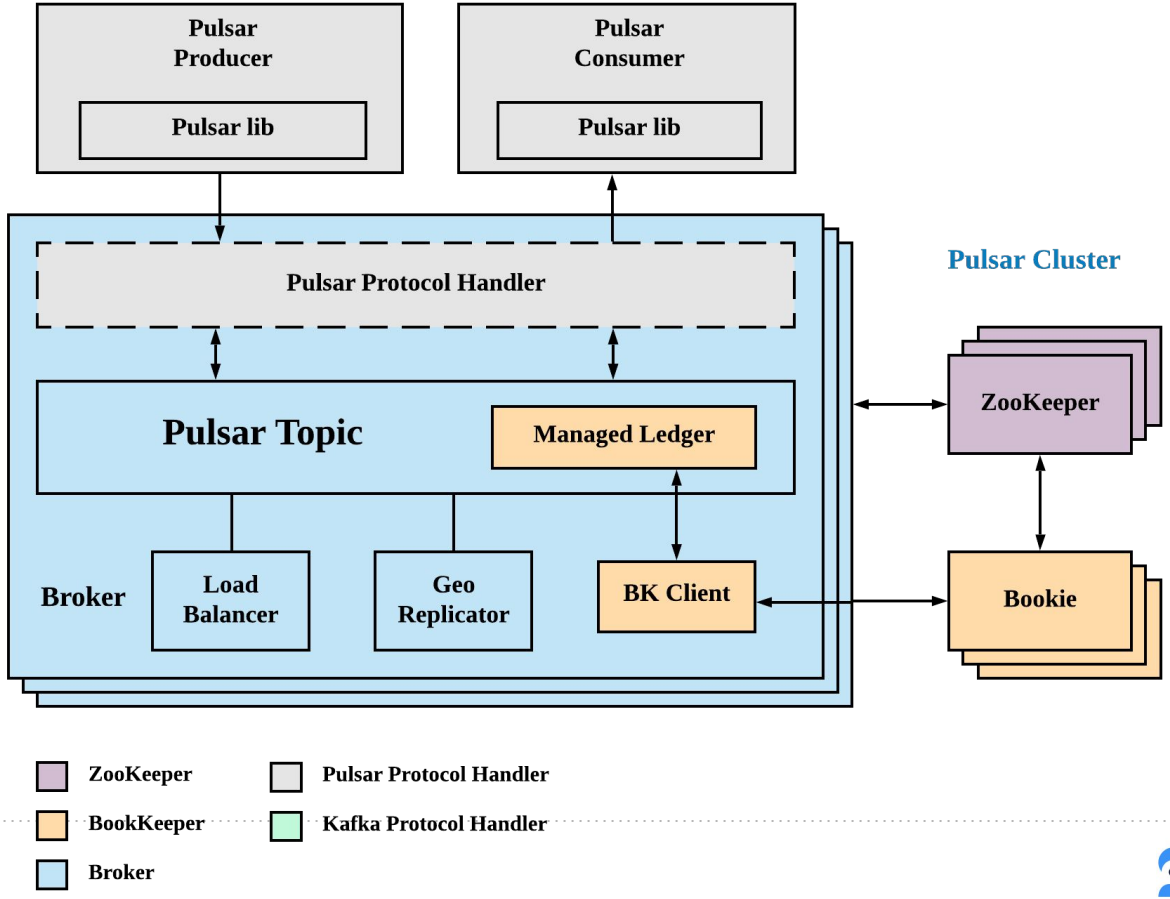
23 4:58 AM - Aug 17, 2019

See StreamNative's other Tweets

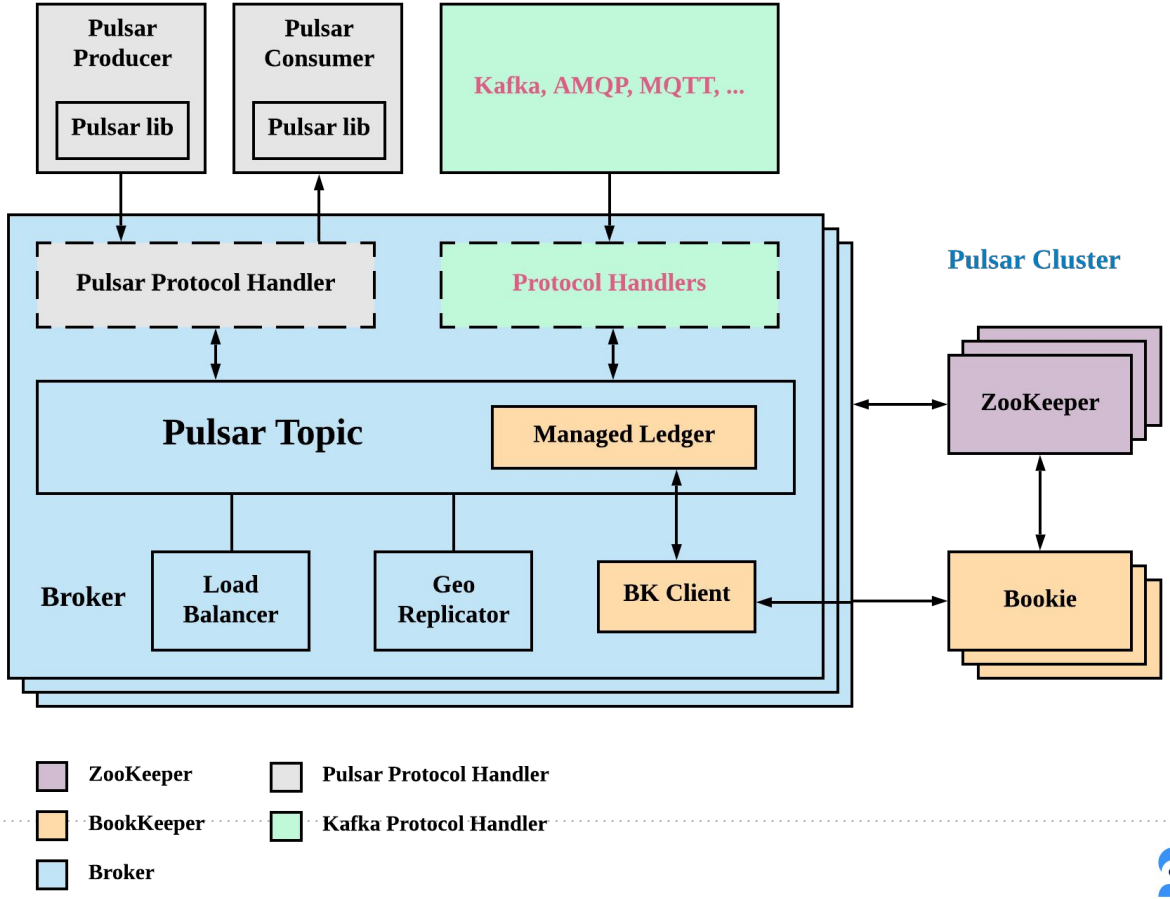
Which lead to our collaboration 🤝

What is Apache Pulsar??

How Pulsar implements its protocol



Protocol Handler



What is the protocol handler?

apache / pulsar

Used by 5

Code | Issues 607 | Pull requests 142 | Actions | Projects

PIP 41: Pluggable Protocol Handler

```
/**
 * The protocol handler interface for support additional protocols on Pulsar brokers.
 */
@Beta
public interface ProtocolHandler extends AutoCloseable {
```

What is the protocol handler?

How to load plugins in a jvm **without using classpath**?

Pulsar is using **NAR** to load plugins!

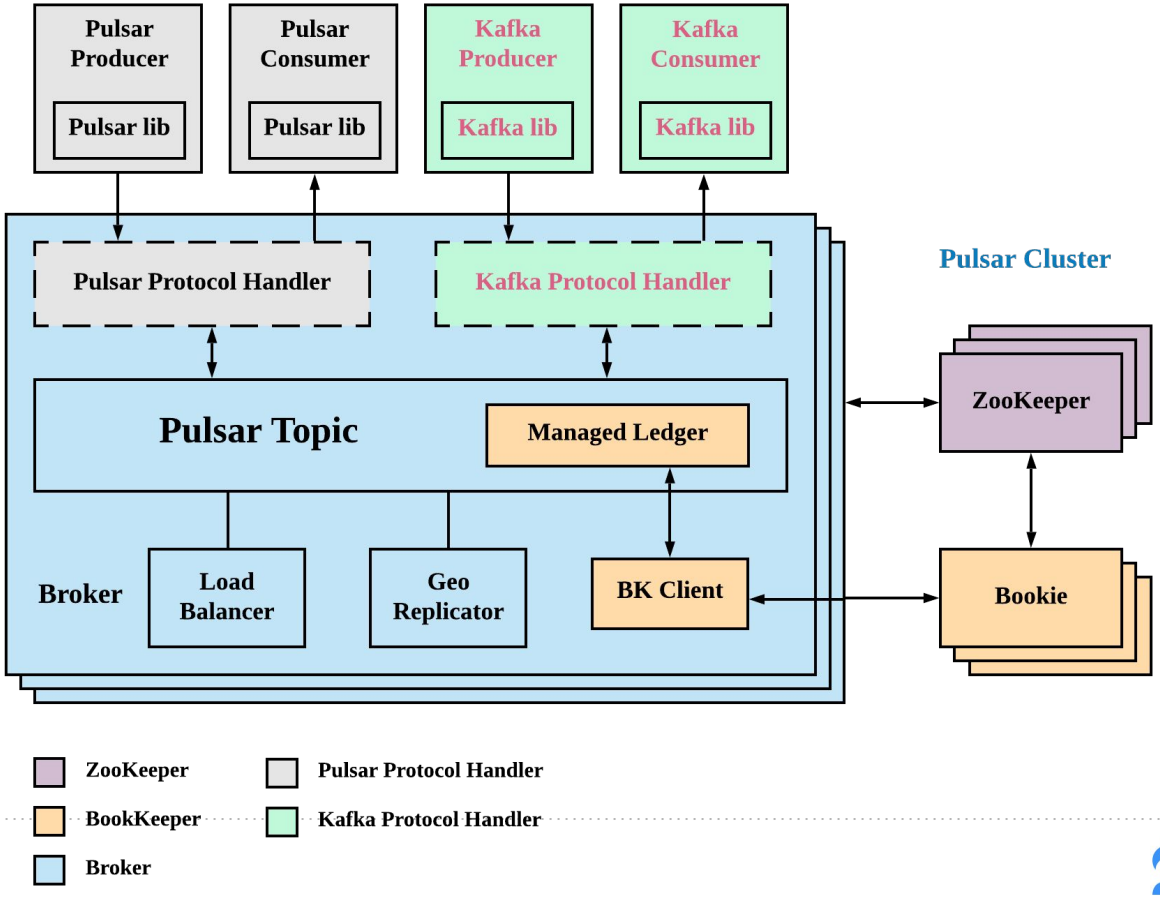
- Pulsar Function
- Pulsar Connector
- Pulsar Offloader
- **Pulsar Protocol Handler**

How-to load protocol handlers?

1. Upgrade your cluster to 2.5
2. Set the following configurations:
3. Configure each protocol handlers
4. Restart your broker
5. Enjoy!

Property	Set it to the following value	Default value
`messagingProtocols`	kafka	null
`protocolHandlerDirectory`	Location of KoP NAR file	./protocols

Kafka-on-Pulsar Protocol Handler



The KoP Implementation

- “distributedlog”
  - Kafka and Pulsar are built on same data structure
- Similarities
  - Topic Lookup
  - Topic / Partitions / Messages / Offset
  - Produce
  - Consume
  - Consumption State

KoP Implementation

- **Topic flat map**: Brokers set `kafkaNamespace`
- **MessageID and offset**: LedgerId + EntryId
- **Message**: Convert key/value/timestamps/headers
- **Topic Lookup**: Pulsar admin topic lookup -> owner broker
- **Produce**: Convert message, then *PulsarTopic.publishMessage*
- **Consume**: Convert requests, then *nonDurableCursor.readEntries*
- **Group Coordinator**: Keep group information in topic `*public/__kafka/__offsets*`

KoP for production

- What Pulsar Provides
  - Multi-Tenancy
  - Security
    - TLS Encryption
    - Authentication, Authorization
    - Data Encryption
  - Geo-replication
  - Tiered storage
  - Schema
  - Integrations with big data ecosystem (Flink / Spark / Presto)

KoP for production

- ❏ What Pulsar Provides
  - ❏ Multi-Tenancy
  - ❏ Security
    - ❏ TLS Encryption
    - ❏ Authentication, Authorization
    - ❏ Data Encryption
  - ✓ Geo-replication
  - ✓ Tiered storage
  - ❏ Schema
  - ❏ Integrations with big data ecosystem (Flink / Spark / Presto)

KoP for production

- ❏ What Pulsar Provides
  - ✓ Multi-Tenancy
  - ✓ Security
    - ✓ TLS Encryption
    - ✓ Authentication, Authorization
    - ❏ Data Encryption
  - ✓ Geo-replication
  - ✓ Tiered storage
  - ❏ Schema
  - ❏ Integrations with big data ecosystem (Flink / Spark / Presto)

KoP for production

- ❑ What Pulsar Provides
  - ✓ **Multi-Tenancy**
  - ✓ **Security**
    - ✓ **TLS Encryption**
    - ✓ **Authentication, Authorization**
    - ❑ Data Encryption
  - ✓ Geo-replication
  - ✓ Tiered storage
  - ❑ Schema
  - ❑ Integrations with big data ecosystem (Flink / Spark / Presto)

KoP multi-tenancy

Pulsar has great support for multi-tenancy, how-to use it in KoP?

SASL-PLAIN is used to inject info:

- The username of Kafka JAAS is the **tenant/namespace**
- The password must be your **classic Pulsar token authentication parameters**

TLS can be added over SASL-PLAIN

KoP multi-tenancy

```
String tenant = "ns1/tenant1";
String password = "token:xxx";

String jaasTemplate =
"org.apache.kafka.common.security.plain.PlainLoginModule required
username=\"%s\" password=\"%s\";";
String jaasCfg = String.format(jaasTemplate, tenant, password);
props.put("sasl.jaas.config", jaasCfg);
props.put("security.protocol", "SASL_PLAINTEXT");
props.put("sasl.mechanism", "PLAIN");
```

KoP Compatibility checklist

Integrations tests are runned with **Kafka official Java client** and **popular Kafka clients** in other languages

Golang

- https://github.com/Shopify/sarama
- https://github.com/confluentinc/confluent-kafka-go

Rust

- https://github.com/fede1024/rust-rdkafka

NodeJS

- https://github.com/Blizzard/node-rdkafka

Demo time!

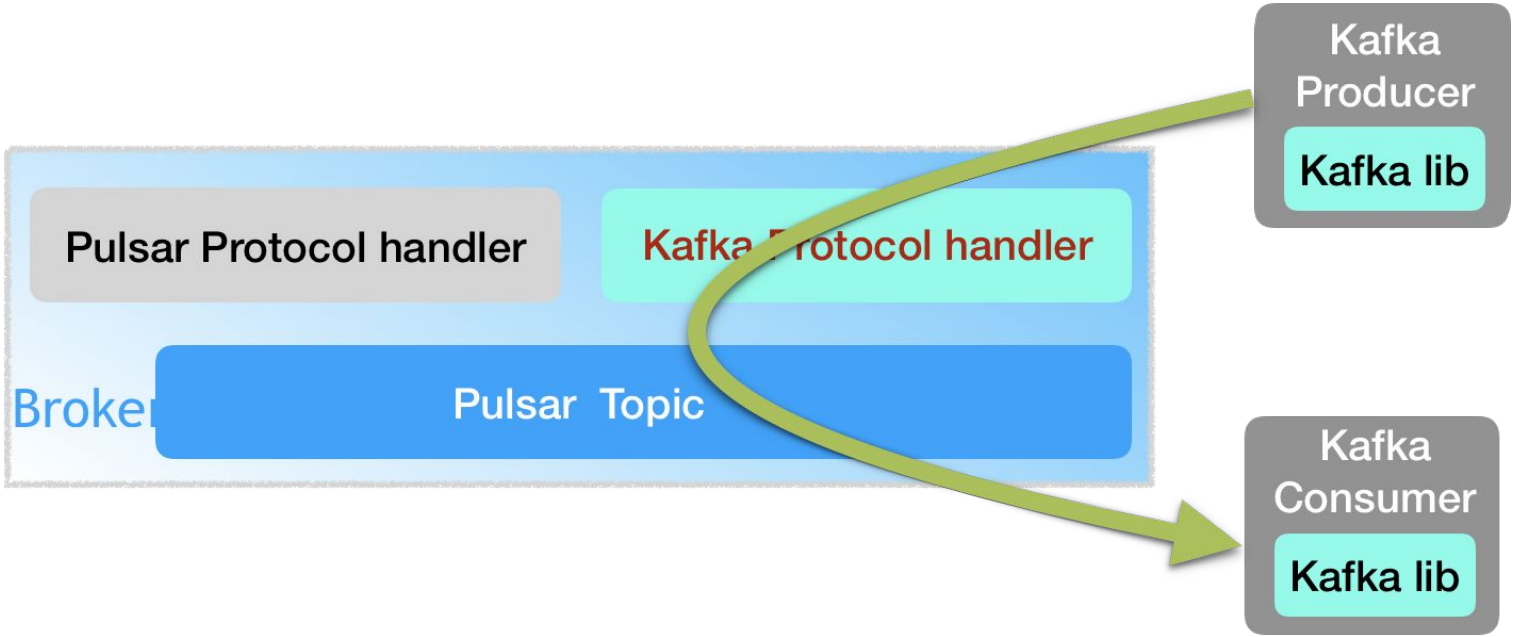
Demo

- **K/P-Producer -> K/P-Consumer**
  - **TLS & SASL-PLAIN**
- **K-Producer -> Pulsar Functions**
- **P-Producer -> Kafka Connect**

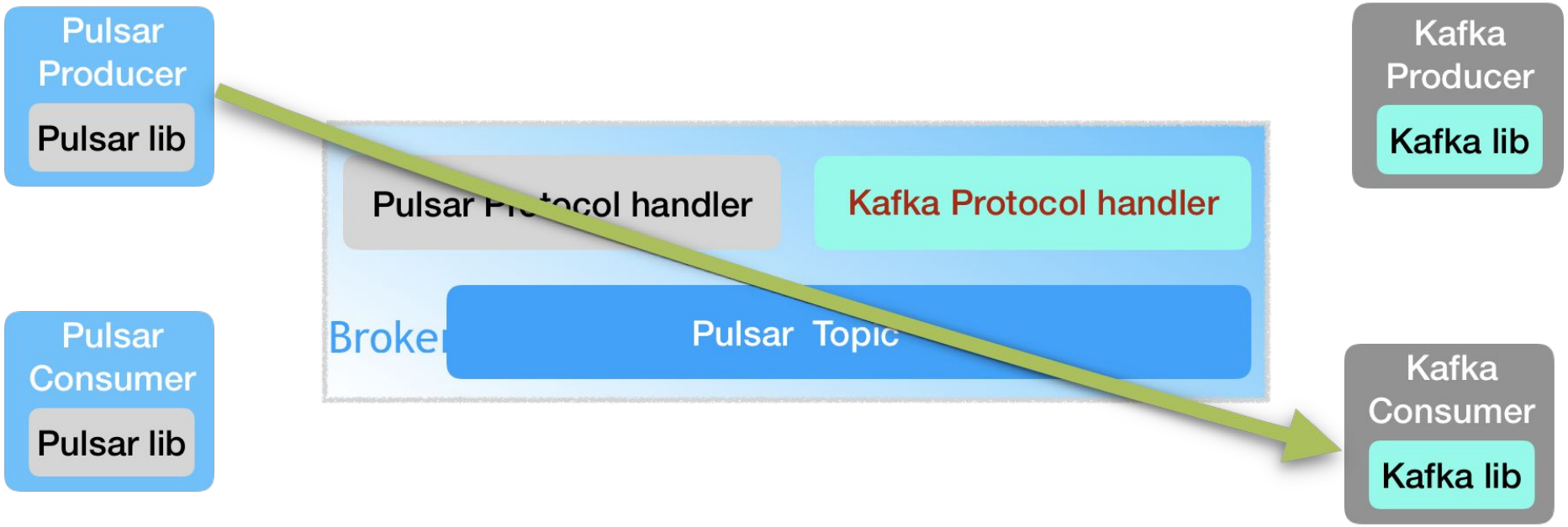
*** All demos run with TLS and SASL-PLAIN**

https://hackmd.io/nLj5M9BEQIacKcZsNrDxmQ

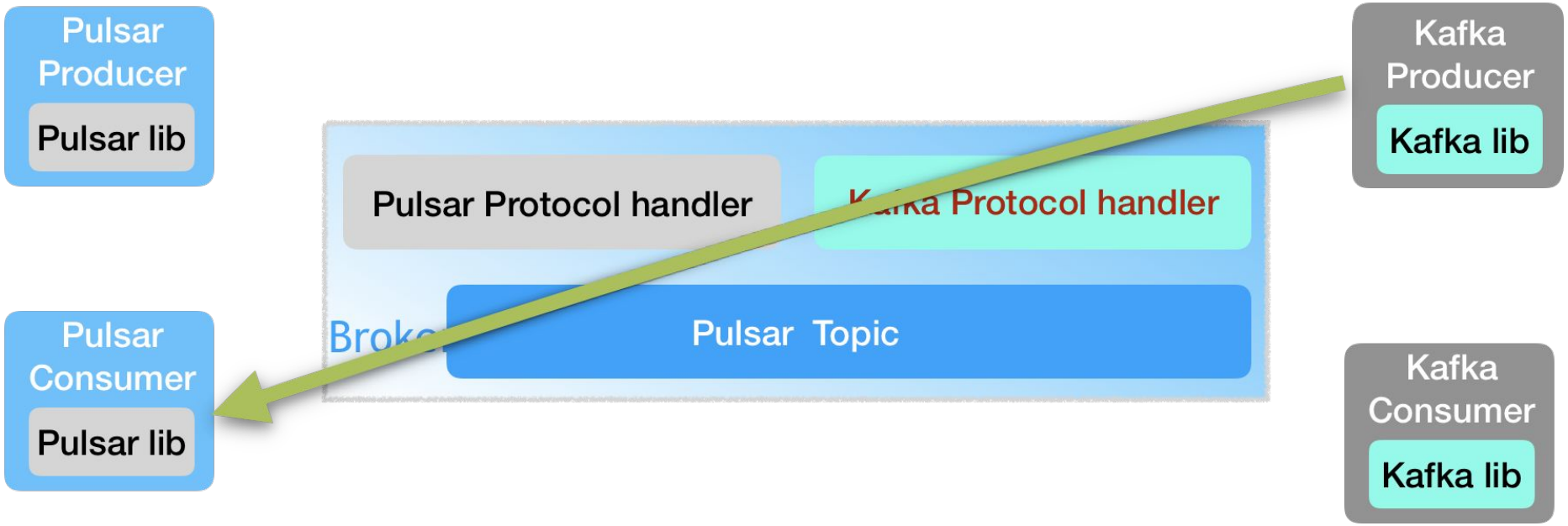
Demo 1: K-Producer -> K-Consumer



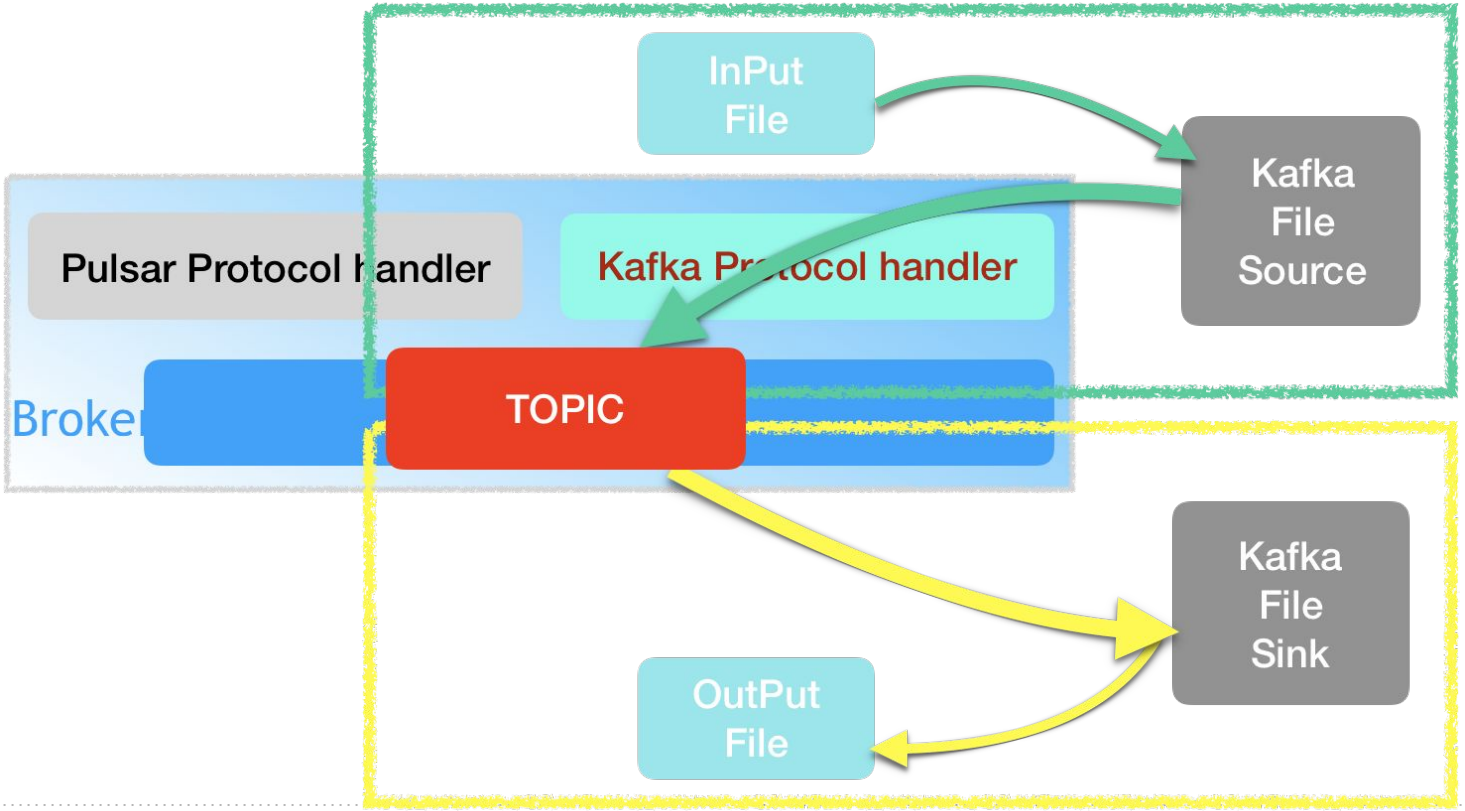
Demo 2: P-Producer -> K-Consumer



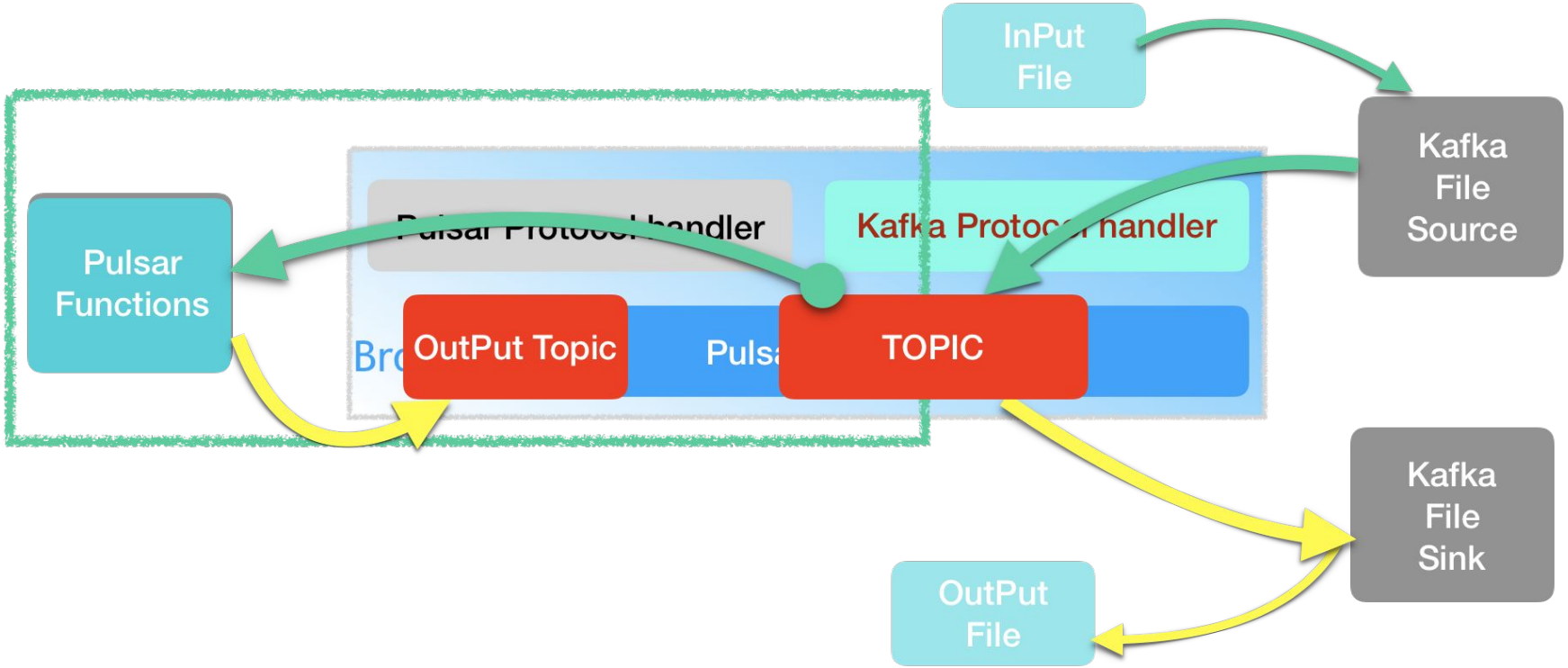
Demo 3: K-Producer -> P-Consumer



Demo 4: Kafka Connect



Demo 5: Pulsar Functions



Apache Pulsar + Apache Kafka

Roadmap / Future work

- KoP Proxy
- Schema
- Kafka transaction (Waiting for Pulsar transaction)
- Kafka 1.X support
- Kafka > 2.0 support

Try it now!

- Download and try it out today!
- https://github.com/streamnative/kop

More protocol handlers are coming!

- **AoP - AMQP on Pulsar** ← First week of May
- **MoP - MQTT on Pulsar**
- ...

2020 Pulsar User Survey Report

https://streamnative.io/whitepaper/sn-apache-pulsar-user-survey-report-2020/

TGI Pulsar Weekly Live Stream

https://www.youtube.com/channel/UCywxUI5HlIyc0VEKYR4X9Pg/live

Q & A

Follow us!

- Follow us at Twitter
  - Pierre Zemb (@PierreZ)
  - Sijie Guo (@sijieg)
  - Apache Pulsar (@apache_pulsar)
  - StreamNative (@streamnativeio)
  - OVHcloud (@OVHcloud)
- Join us at #***kop*** channel on Pulsar slack!