Utilizing Apache Pulsar, Apache NiFi and MiNiFi for EdgeAI IoT at Scale

Tim Spann | Developer Advocate

Tim Spann

Developer Advocate

DZone Zone Leader and Big Data MVB Data DJay

- https://www.datainmotion.dev/
- https://github.com/tspannhw/SpeakerProfile
- https://dev.to/tspannhw
- https://sessionize.com/tspann/

StreamNative

Founded by the original developers of Apache Pulsar and Apache BookKeeper, StreamNative builds a cloud-native event streaming platform that enables enterprises to easily access data as real-time event streams.

Apache Pulsar

StreamNative

Apache  is an open source, cloud-native distributed messaging and streaming platform.

What are the Benefits of Pulsar?

Multi-Tenancy

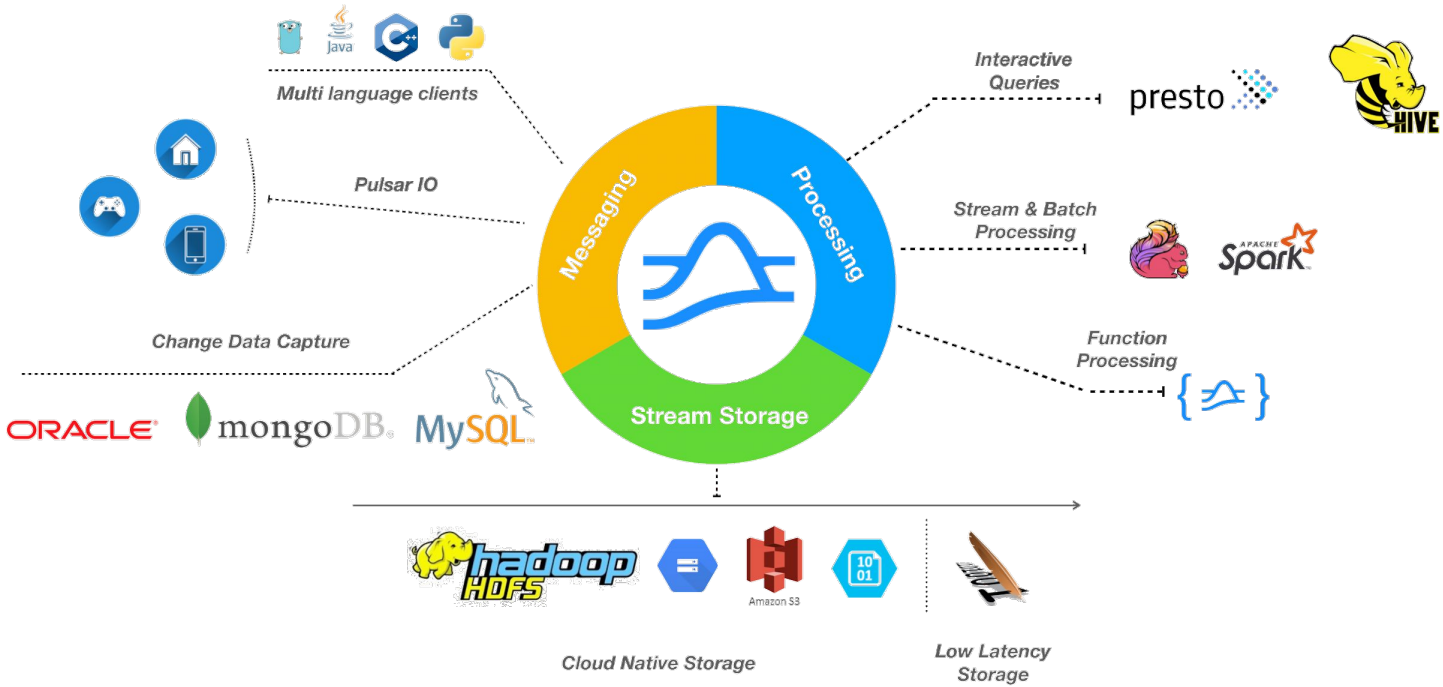
Scalability

Geo-Replication

Unified Messaging Model

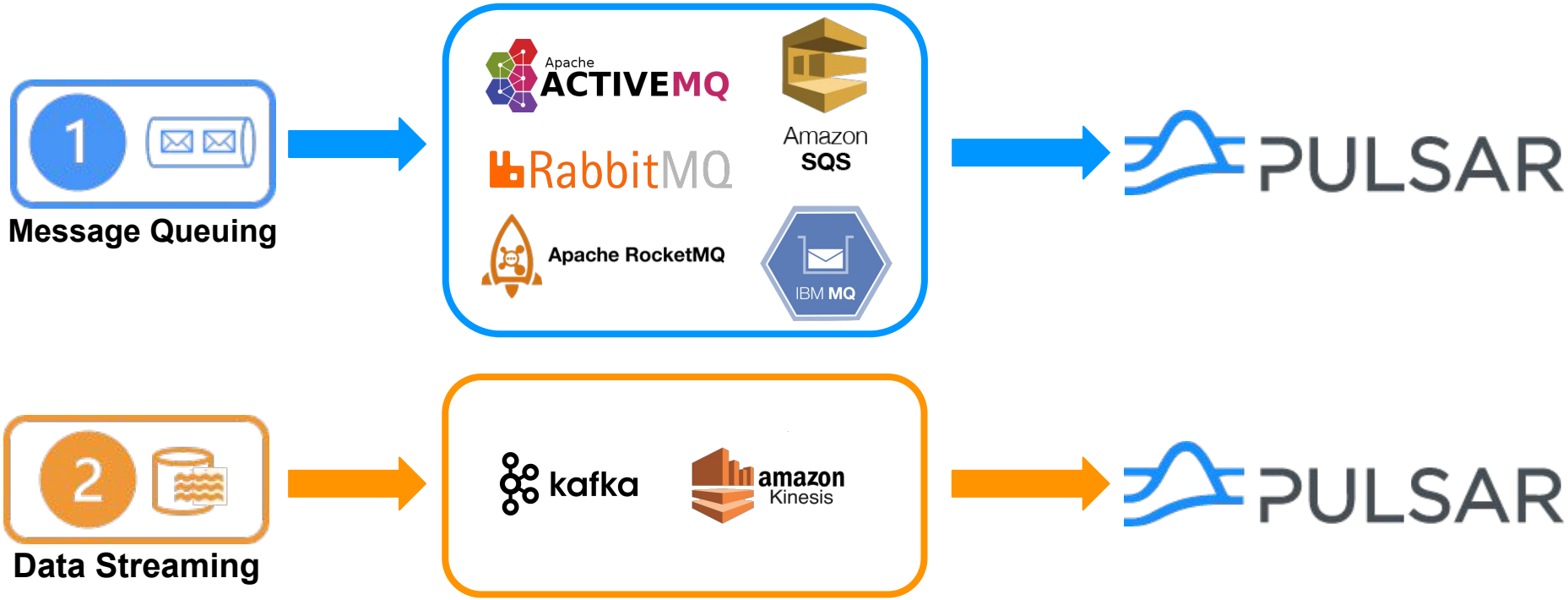
Data Durability

Apache Pulsar



StreamNative

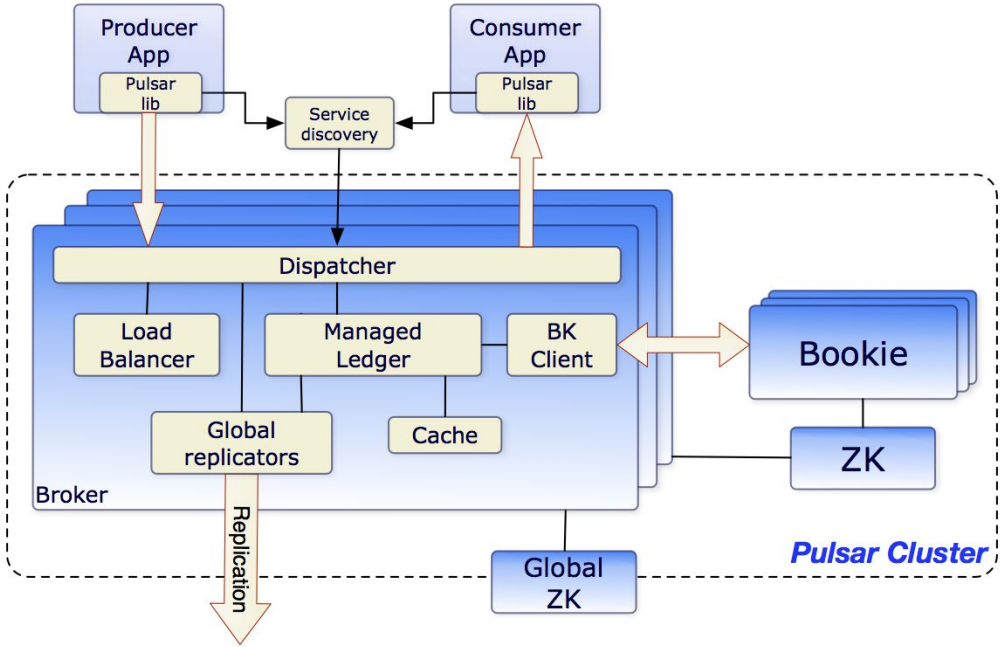
A Unified Messaging Platform



Apache Pulsar Overview

Enable Geo-Replicated Messaging

- Pub-Sub
- Geo-Replication
- Pulsar Functions
- Horizontal Scalability
- Multi-tenancy
- Tiered Persistent Storage
- Pulsar Connectors
- REST API
- CLI
- Many clients available
- Four Different Subscription Types
- Multi-Protocol Support
  - MQTT
  - AMQP
  - JMS
  - Kafka
  - ...



What is the Pulsar Ecosystem?

- **Functions and Connectors**
  - Functions: Lightweight stream processing
  - Connectors: Part of “Pulsar IO”, includes “Source” and “Sink” APIs
    - Files, Databases, Data tools, Cloud Services, etc
- **Protocol Handlers**
  - Allows Pulsar to handle additional protocols by an extendable API running in the broker
    - AoP (AMQP), KoP (Kafka), MoP (MQTT)

What is the Pulsar Ecosystem? (cont'd)

- **Processing Engines**
  - Supports modern processing engines
    - Flink and Spark, as well as Pulsar SQL (Presto/Trino)
- **Offloaders**
  - Allows data to be offloaded to cloud storage and used with existing Pulsar APIs
    - S3, GCP Cloud Storage, HDFS, File (NFS), Azure Blob Storage (in Pulsar 2.7.0)

Pulsar Functions

Provides a simple API to:

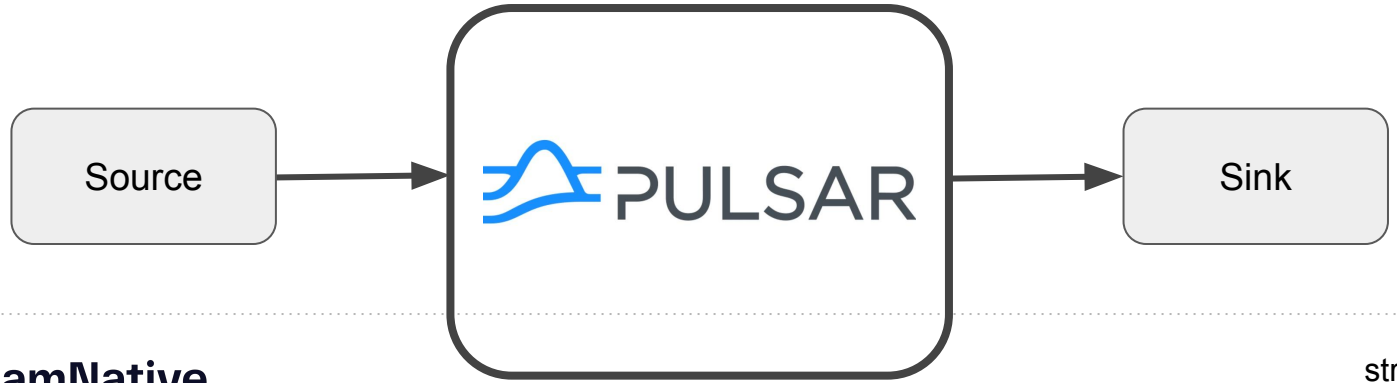
- Receive a message (consume)
- Process the message using your own code
- Send a message (produce)

Takes care of the boilerplate code so there is no need to create producers and consumers.

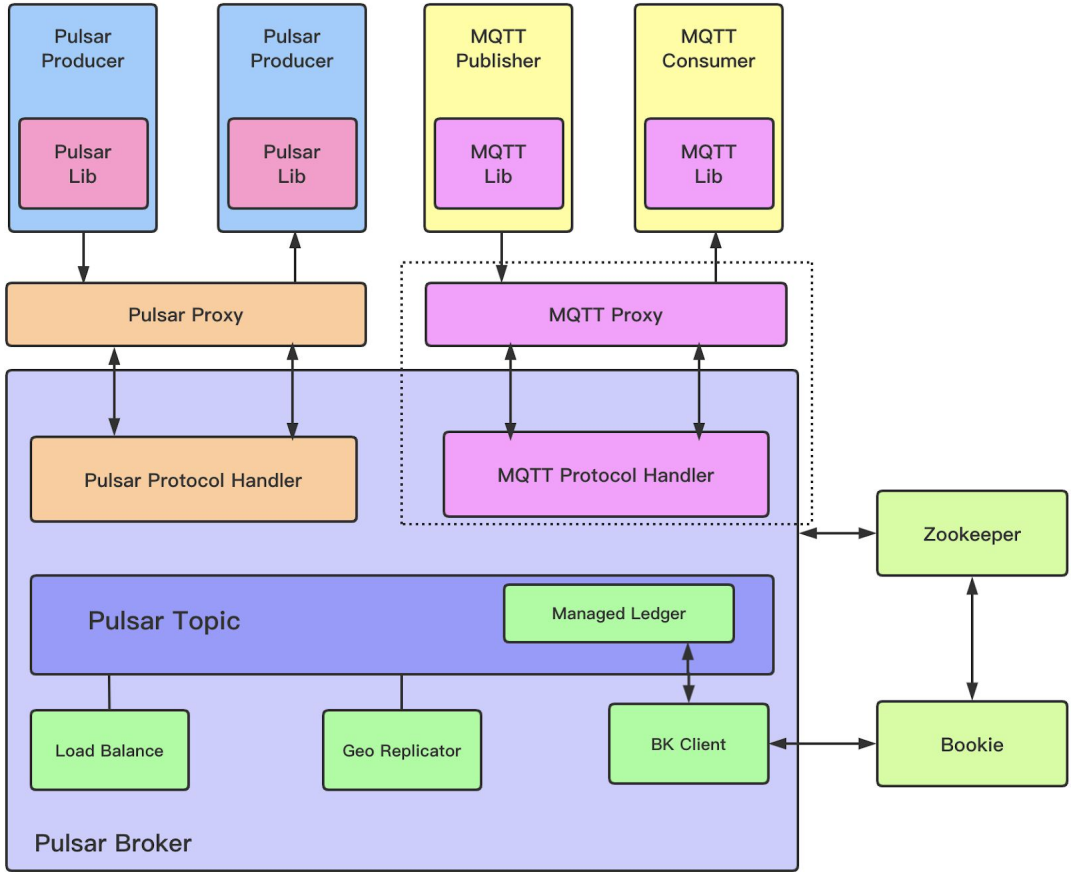
Moving Data In and Out of Pulsar

IO/Connectors are a simple way to integrate with external systems and move data in and out of Pulsar.

- Built on top of Pulsar Functions
- Built-in connectors - hub.streamnative.io



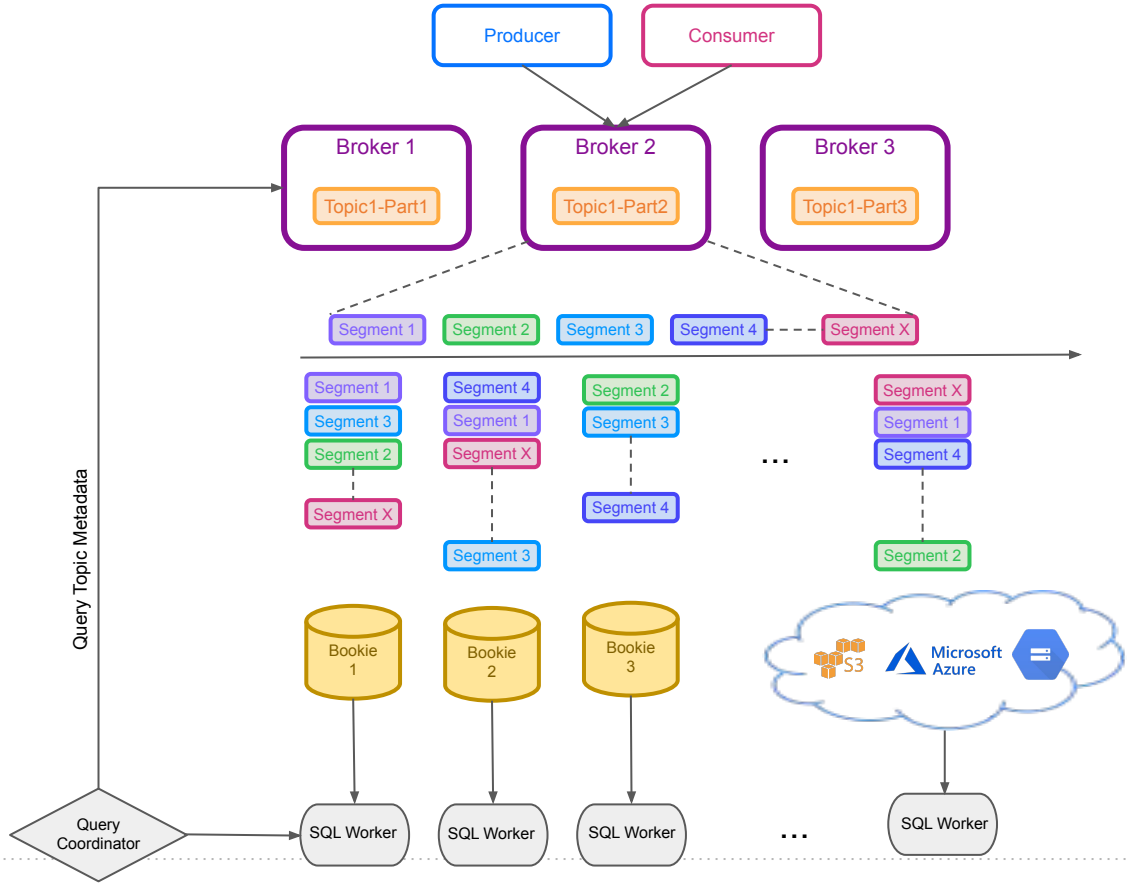
MQTT on Pulsar (MoP)



StreamNative

Pulsar SQL

Presto/Trino workers can read segments directly from bookies (or offloaded storage) in parallel.



trino

StreamNative

Ingesting IoT Data via Java Pulsar

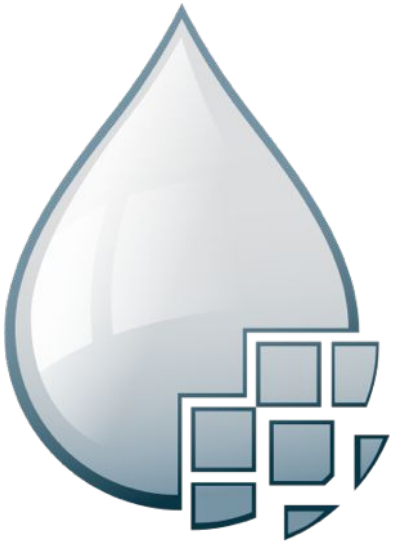
```java
UUID uuidKey = UUID.randomUUID();
String pulsarKey = uuidKey.toString();
String OS = System.getProperty("os.name").toLowerCase();
String message = "" + jct.message;
IoTMessage iotMessage = parseMessage("" + jct.message);
String topic = DEFAULT_TOPIC;
if ( jct.topic != null && jct.topic.trim().length()>0) {
    topic = jct.topic.trim();
}
ProducerBuilder<IoTMessage> producerBuilder = client.newProducer(JSONSchema.of(IoTMessage.class))
    .topic(topic)
    .producerName("jetson")
    .sendTimeout(5, TimeUnit.SECONDS);

Producer<IoTMessage> producer = producerBuilder.create();

MessageId msgID = producer.newMessage()
                .key(iotMessage.getUuid())
                .value(iotMessage)
                .send();
```

https://github.com/tspannhw/StreamingAnalyticsUsingFlinkSQL/

Ingesting IoT Data via Java Pulsar

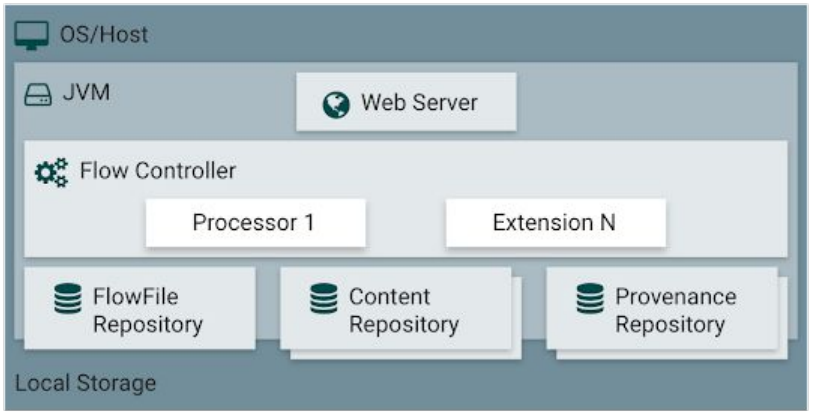
```
private static IoTMessage parseMessage(String message) {

    IoTMessage iotMessage = null;

    try {
        if ( message != null && message.trim().length() > 0) {
            ObjectMapper mapper = new ObjectMapper();
            iotMessage = mapper.readValue(message, IoTMessage.class);
            mapper = null;
        }
    }
    catch(Throwable t) {
        t.printStackTrace();
    }

    if (iotMessage == null) {
        iotMessage = new IoTMessage();
    }
    return iotMessage;
}
```

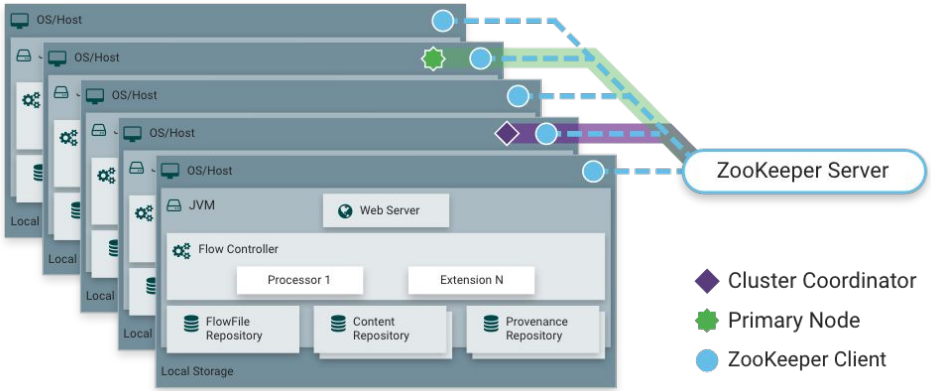
Why Apache NiFi?



- Guaranteed delivery
- Data buffering
  - Backpressure
  - Pressure release
- Prioritized queuing
- Flow specific QoS
  - Latency vs. throughput
  - Loss tolerance
- Data provenance
- Supports push and pull models
- Hundreds of processors
- Visual command and control
- Over a sixty sources
- Flow templates
- Pluggable/multi-role security
- Designed for extension
- Clustering
- Version Control

Architecture

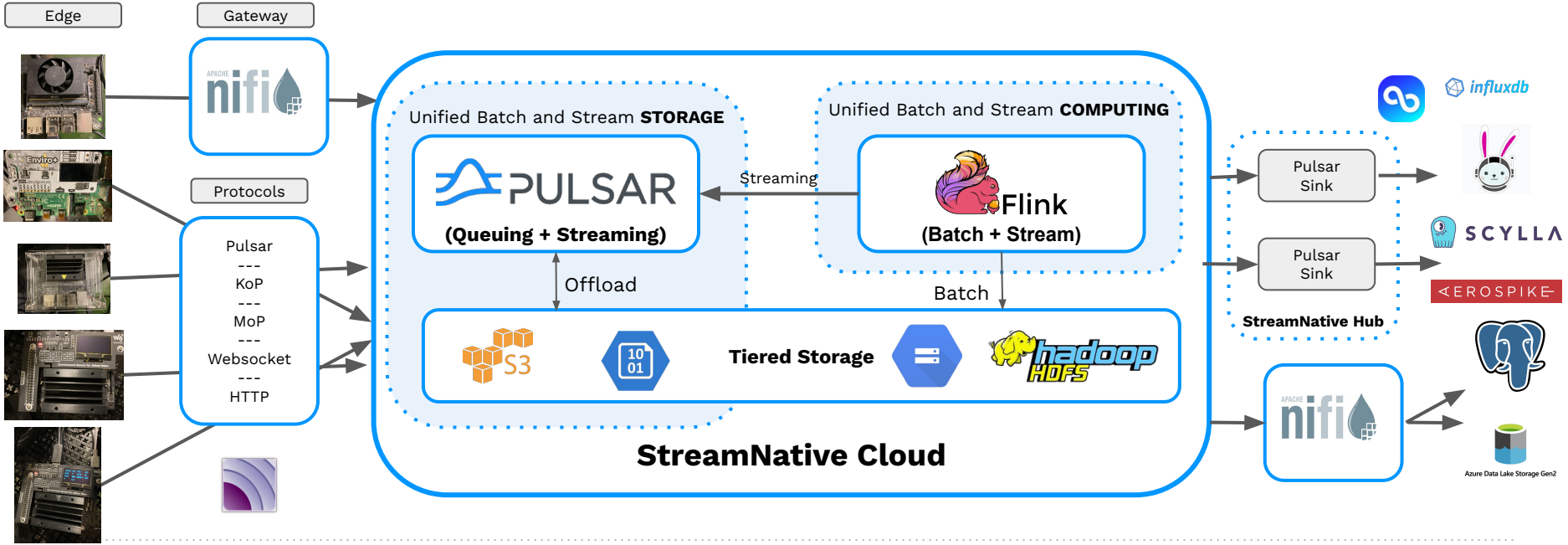




https://nifi.apache.org/docs/nifi-docs/html/overview.html

End-to-End Streaming FLiP(N) IoT Apps

Apache Pulsar - Apache NiFi - MiNiFi <-> Events/Messages <-> Data Stores



StreamNative

Demo

StreamNative

Wrap-Up

StreamNative

Interested In Learning More?

Resources

[Flink SQL Cookbook]

[The Github Source for Flink SQL Demo]

[The GitHub Source for Demo]

Free eBooks

[Manning's Apache Pulsar in Action]

[O'Reilly Book]

Upcoming Events

[11/8] [PASS Data Community]

[11/18] [Developer Week Austin]

[11/19] [Porto Tech Hub Con]

[12/3] [Data Science Camp]

Let’s Keep in Touch!

Timothy Spann
Developer Advocate

@PaasDev

https://www.linkedin.com/in/timothyspann

https://github.com/tspannhw

StreamNative