MUSIC CITY TECH
VIRTUAL CONFERENCE

Timothy Spann

Developer Advocate

Hail Hydrate! From Stream to Lake

Progress®

NASHVILLE SOFTWARE SCHOOL

TVA TENNESSEE VALLEY AUTHORITY


Tim
SPANN
https://github.com/tspannhw
https://www.datainmotion.dev/

Tim Spann, Developer Advocate

DZone Zone Leader and Big Data MVB
Data DJay

https://github.com/tspannhw
https://www.datainmotion.dev/
https://github.com/tspannhw/SpeakerProfile
https://dev.to/tspannhw
https://sessionize.com/tspann/
https://www.slideshare.net/bunkertor

Stream Native

@PaasDev

AGENDA

Use Case - Populate the Data Lake

Key Challenges

- Their Impact
- A Solution
- Outcome

Why Apache NiFi and Apache Pulsar?

Successful Architecture

Demo

USE CASE

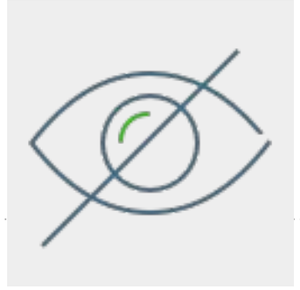
IoT Ingestion: High-volume streaming sources, multiple message formats, diverse protocols and multi-vendor devices creates data ingestion challenges.

MUSIC CITY TECH

KEY CHALLENGES

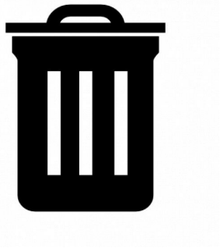
Data Ingestion: High-volume streaming sources, multiple message formats, diverse protocols and multi-vendor devices creates data ingestion challenges.

Real-time Insights: Analyzing continuous and rapid inflow (velocity) of streaming data at high volumes creates major challenges for gaining real-time insights.



Visibility: Lack visibility of end-to-end streaming data flows, inability to troubleshoot bottlenecks, consumption patterns etc.

IMPACT



Code Sprawl: Custom scripts over various qualities proliferate across environments to cope with the complexity.

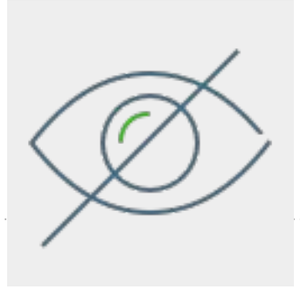
Costs: Increasing costs of development and maintenance. Too many tools, not enough experts, waiting for contractors or time delays as developers learn yet another tool, package or language. Maintaining multiple messaging clusters.

Delays: Decreasing user satisfaction and delay in project delivery. Missed revenue and opportunities.

SOLUTION

Data Ingestion: Apache Pulsar and Apache NiFi provide tools to handle high-volume streaming sources, multiple message formats, diverse protocols and multi-vendor devices.

Variety of Data: Apache Pulsar and Apache NiFi offer many OOTB connectors for sinks and sources.



Visibility: Apache Pulsar and Apache NiFi provenance provides insights, metrics and control over the entire end-to-end stream across clouds.

OUTCOME

New Applications: Enablement of new innovative use cases in compressed timeframe. No more waiting for data to arrive, Data Analysts and Data Scientists focus on innovation.

Savings: Cost reduction thanks to technologies offload, reduced consultant costs and simplification of ingest processes.

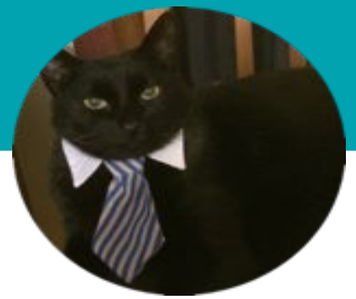
Agility: Reduction of new data source onboarding time from weeks to days. More data in your data warehouse now.

FLiPN Stack for Cloud Data Engineers - Events

Multiple users, protocols, frameworks, languages, clouds, data sources & clusters

CLOUD DATA ENGINEER

- Experience in ETL/ELT
- Coding skills in Python or Java
- Knowledge of database query languages such as SQL
- Experience with Streaming
- Knowledge of Cloud Tools



CAT

- Expert in ETL (Eating, Ties and Laziness)
- Edge Camera Interaction
- Typical User
- No Coding Skills
- Can use NiFi
- Questions your cloud spend

AI / Deep Learning / ML / DS

- Can run in Apache NiFi
- Can run in Apache Pulsar Functions
- Can run in Apache Flink
- Can run in Apache Flink SQL
- Can run in Apache Pulsar Clients
- Can run in Apache Pulsar Microservices
- Can run in Function Mesh

Function Mesh

https://functionmesh.io/

StreamNative Solution

APP Layer

Application Messaging

Micro Service

Payment

Notification

Data Pipelines

ETL

Real-time Contextual Analytics

Dashboard

Risk Control

Auditing

StreamNative Platform

Computing Layer

Storage Layer

Tiered Storage

IaaS Layer

kubernetes

Google Cloud Platform

Stream Native Platform

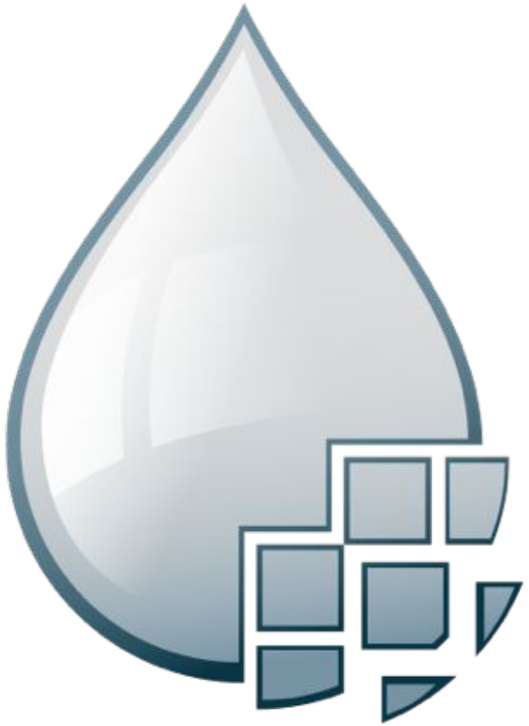
Stream Native Cloud

StreamNative

What is Apache NiFi?

Apache NiFi is a scalable, real-time streaming data platform that collects, curates, and analyzes data so customers gain key insights for immediate actionable intelligence.

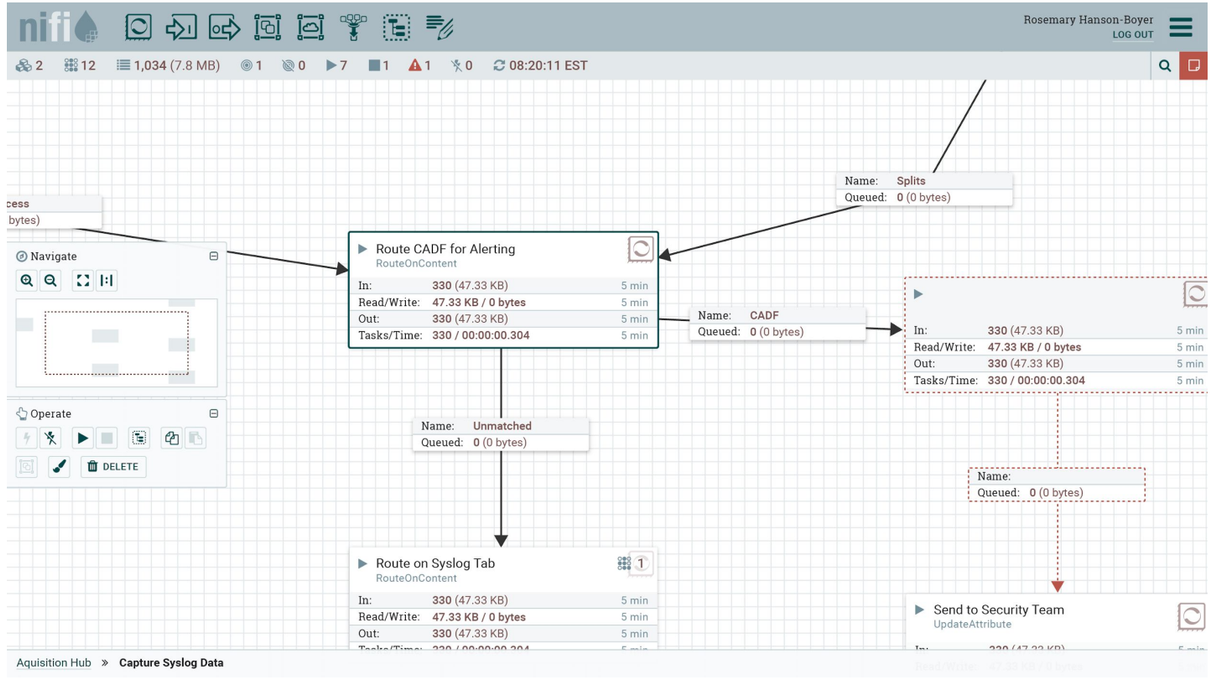
Why Apache NiFi?



- 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

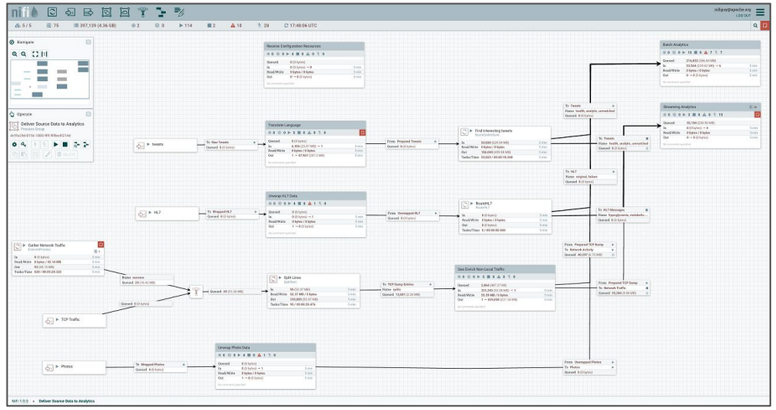
Apache NiFi High Level Capabilities

- Scale horizontal and vertically
  - Scale your data flow to millions event/s
  - Ingest TB to PB of data per day
- Adapt to your flow requirements
  - Back pressure & Dynamic prioritization
  - Loss tolerant vs guaranteed delivery
  - Low latency vs high throughput
- Secure
  - SSL, HTTPS, SFTP, etc.
  - Governance and data provenance
- Extensible
  - Build your own processors and Controller services (providers)
  - Integrate with external systems (Security, Monitoring, Governance, etc)



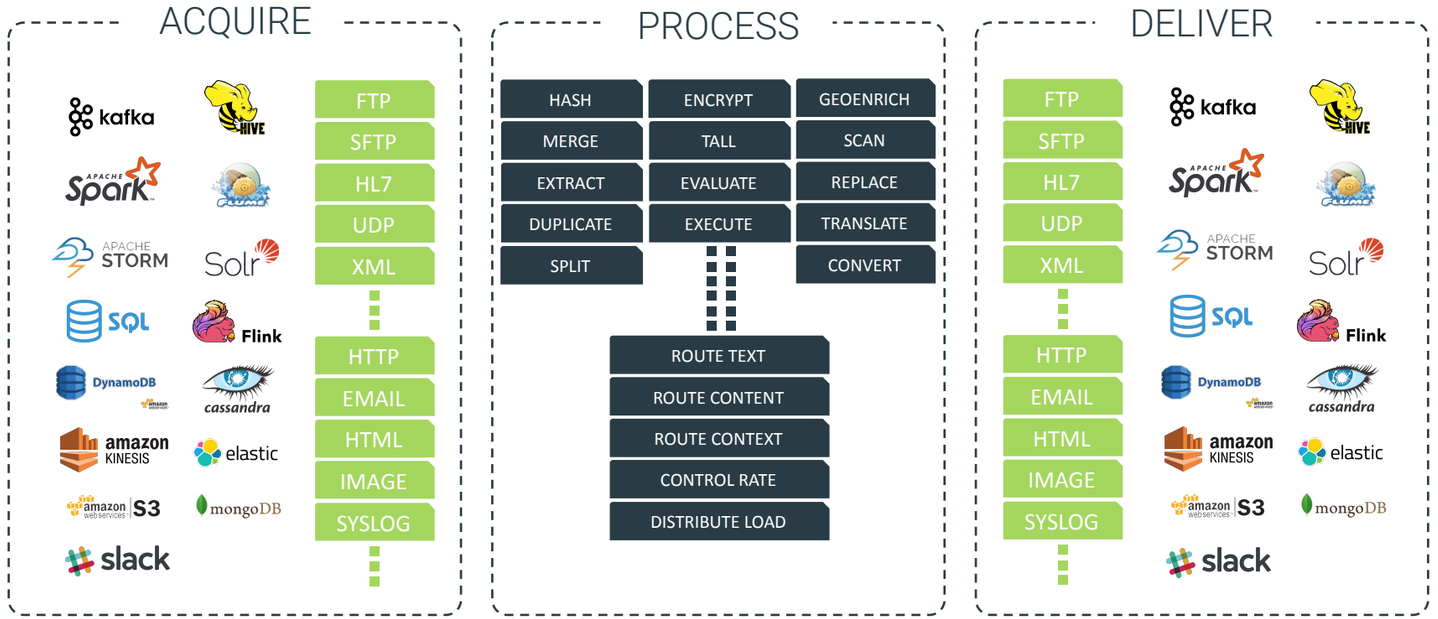
Apache NiFi

Enable easy ingestion, routing, management and delivery of any data anywhere (Edge, cloud, data center) to any downstream system with built in end-to-end security and provenance



APACHE NIFI registry

Advanced tooling to industrialize flow development (Flow Development Life Cycle)



- Over 300 Prebuilt Processors
- Easy to build your own
- Parse, Enrich & Apply Schema
- Filter, Split, Merger & Route
- Throttle & Backpressure

- Guaranteed Delivery
- Full data provenance from acquisition to delivery
- Diverse, Non-Traditional Sources
- Eco-system integration

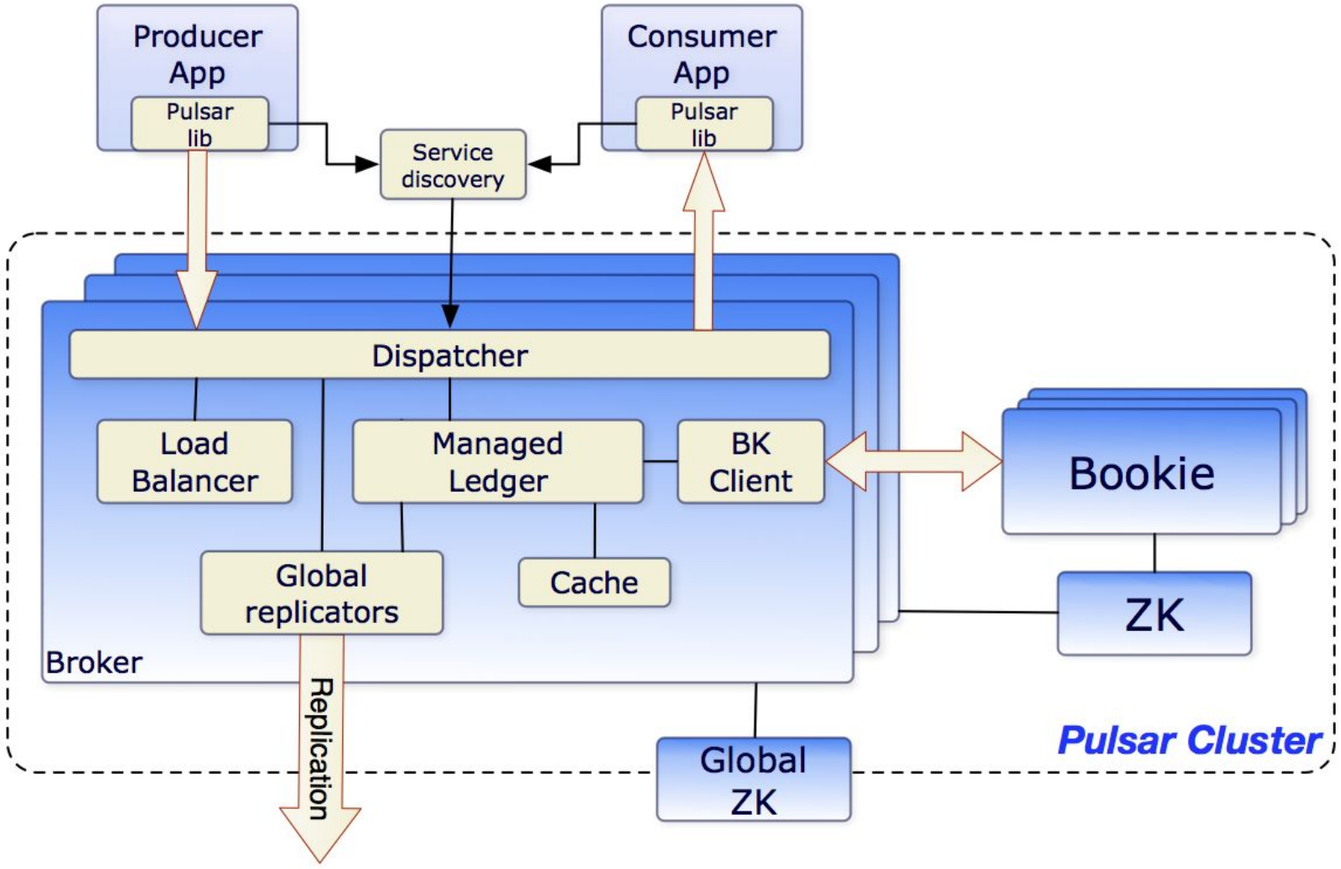
What is Apache Pulsar?

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

Apache Pulsar

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**
  - ...



Apache Pulsar: Key Features (1)

Multi-tenancy

- ✓ Data is stored in one system and shared by multiple organizations
- ✓ Apply access control policy to ensure data stay compliant

Geo-replication

- ✓ Pulsar supports multi-datacenter (n-mesh) replication with both asynchronous and synchronous replication for built-in disaster recovery

Tiered storage

- ✓ Enable historical data to be offloaded to cloud-native storage
- ✓ Effectively store event streams for indefinite periods of time

Cloud-Native Architecture

- ✓ Separate computing layer from storage layer
- ✓ Instant elasticity and scalability
- ✓ Rebalance-free to save labor cost
- ✓ Streamlined operations

Apache Pulsar: Key Features (2)

Converged Messaging

- ✓ Support both application messaging and data pipelines
- ✓ Store one copy of data
- ✓ Consume with different subscriptions

Pluggable Protocols

- ✓ Support popular messaging protocols: Kafka, AMQP, MQTT
- ✓ Provide full protocol compatibility
- ✓ Zero migration cost

Serverless Streaming

- ✓ Pulsar Functions provides an easy-to-use stream processing framework to process streams in a serverless way

Unified Batch and Stream Storage

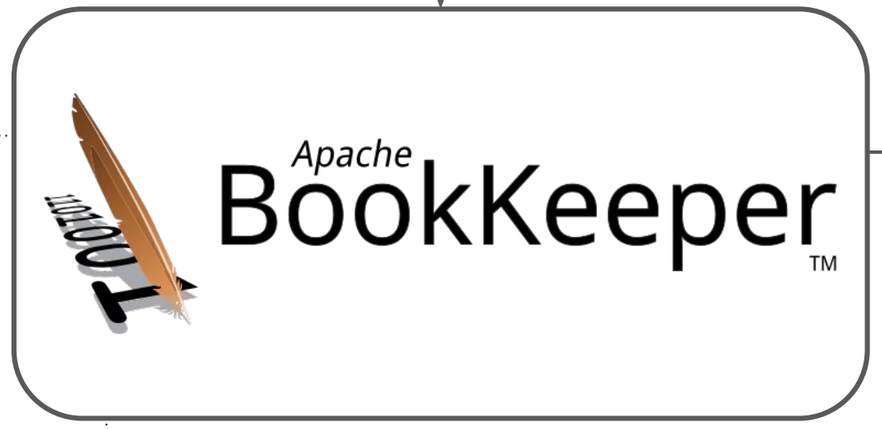
- ✓ Tiered storage enables Pulsar to store real-time data and historic data in one system
- ✓ Tightly integrated with Flink for unified batch and stream processing

Pulsar Cluster

- **“Brokers”**
- Handles message routing and connections
- Stateless, but with caches
- Automatic load-balancing
- Topics are composed of multiple segments

Metadata & Service Discovery

Store Messages

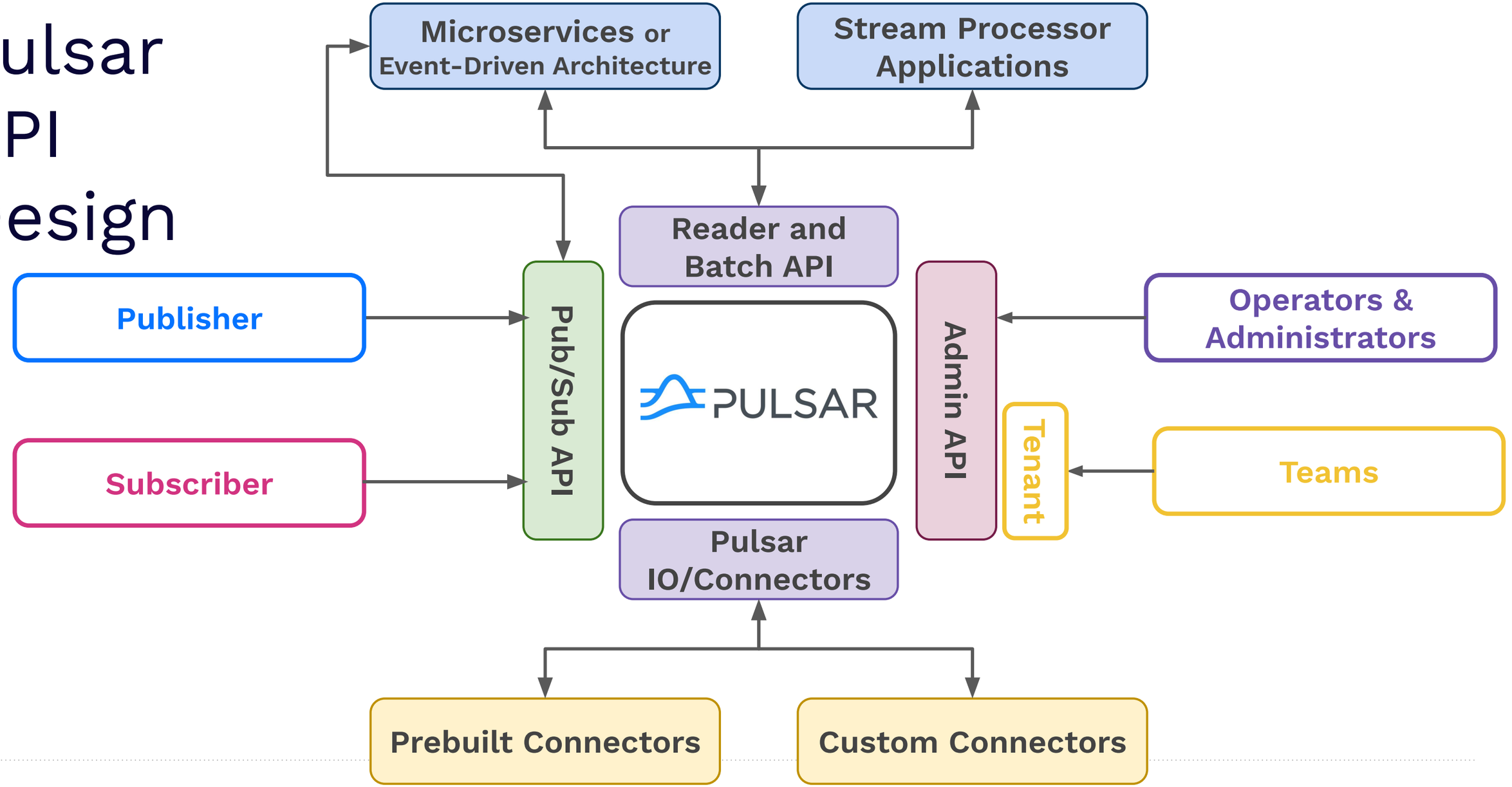


Metadata & Service Discovery

- Stores metadata for both Pulsar and BookKeeper
- Service discovery

- **“Bookies”**
- Stores messages and cursors
- Messages are grouped in segments/ledgers
- A group of bookies form an “ensemble” to store a ledger

Pulsar API Design



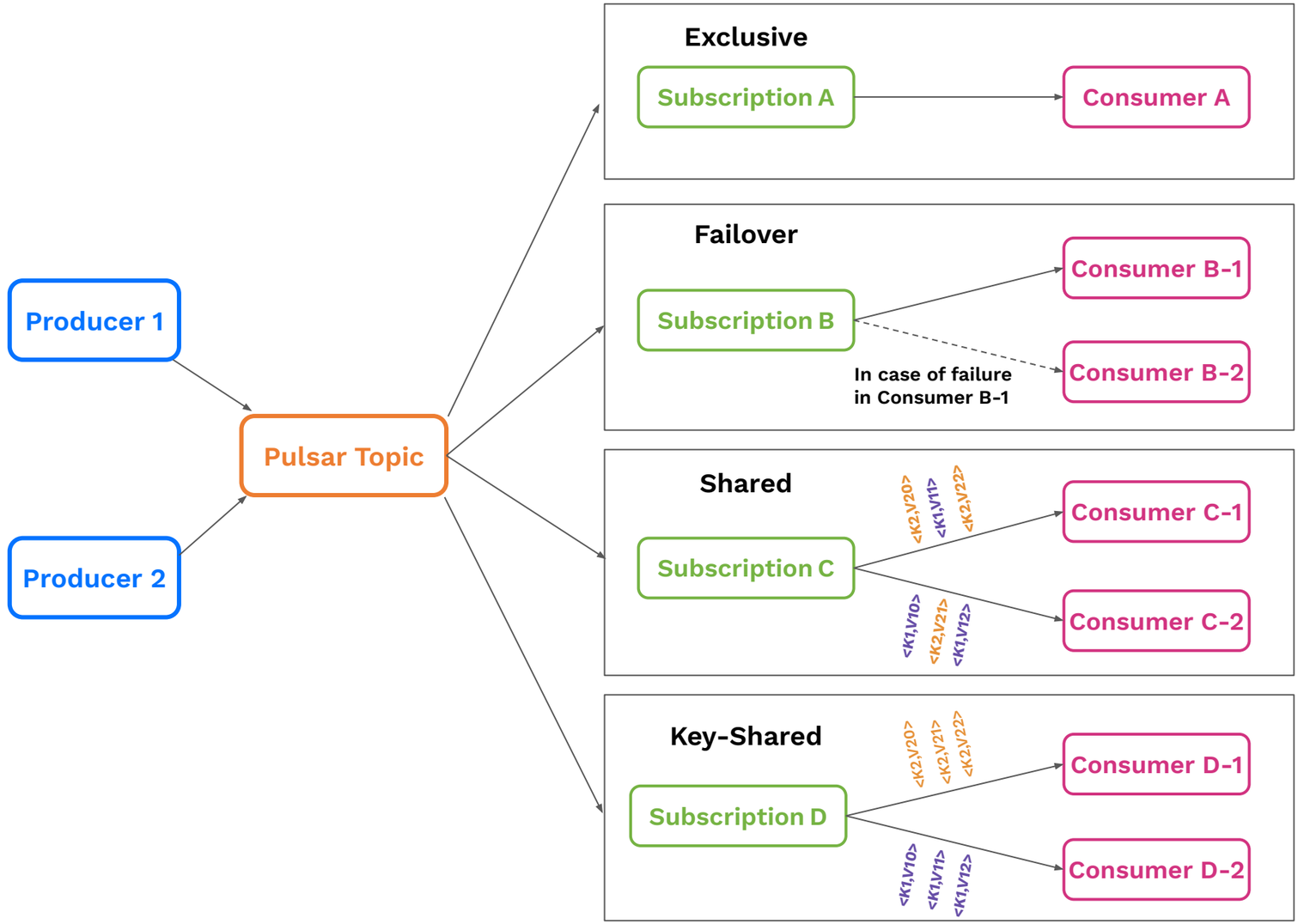
Subscription Modes

Different subscription modes have different semantics:

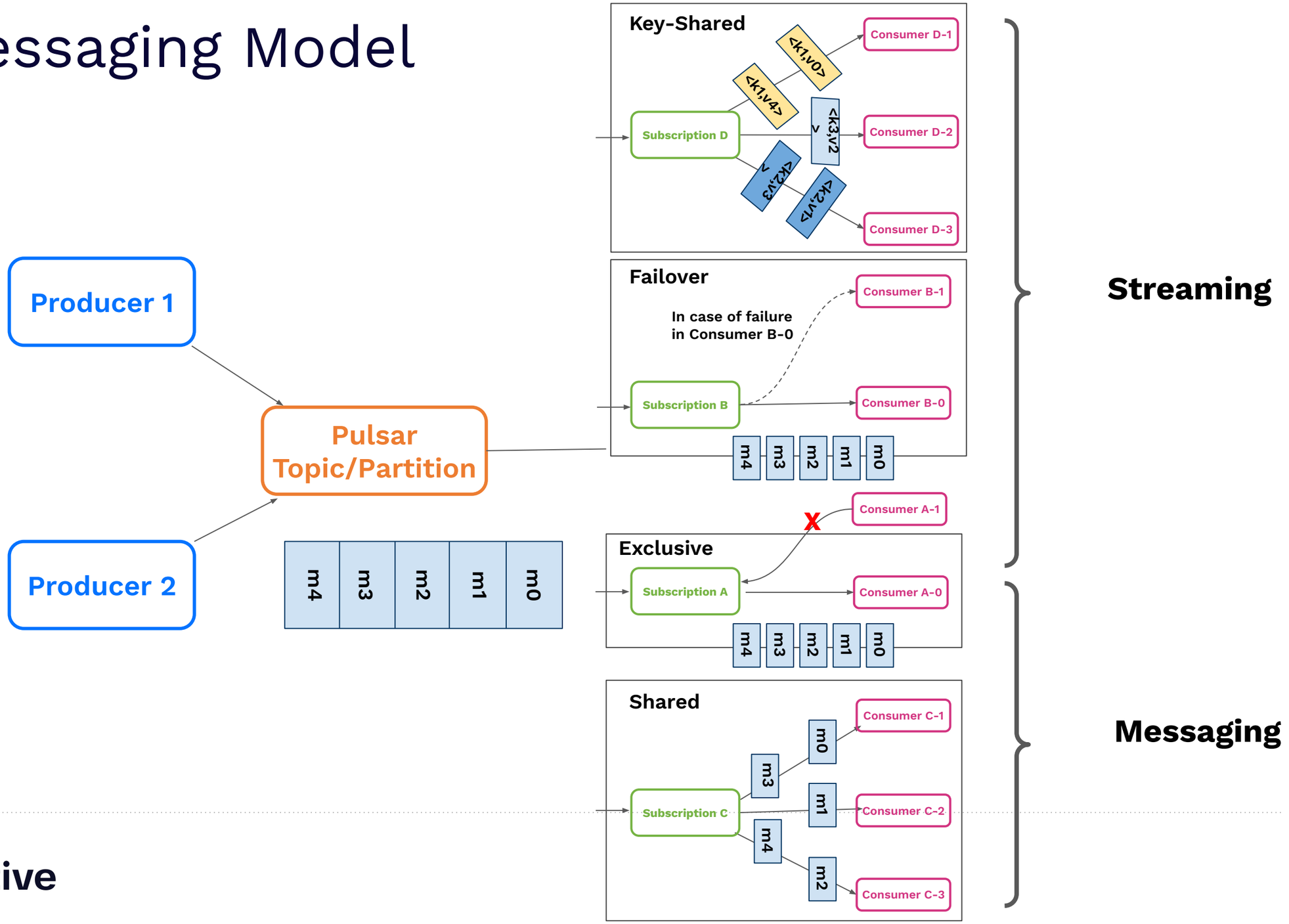
Exclusive/Failover - guaranteed order, single active consumer

Shared - multiple active consumers, no order

Key_Shared - multiple active consumers, order for given key

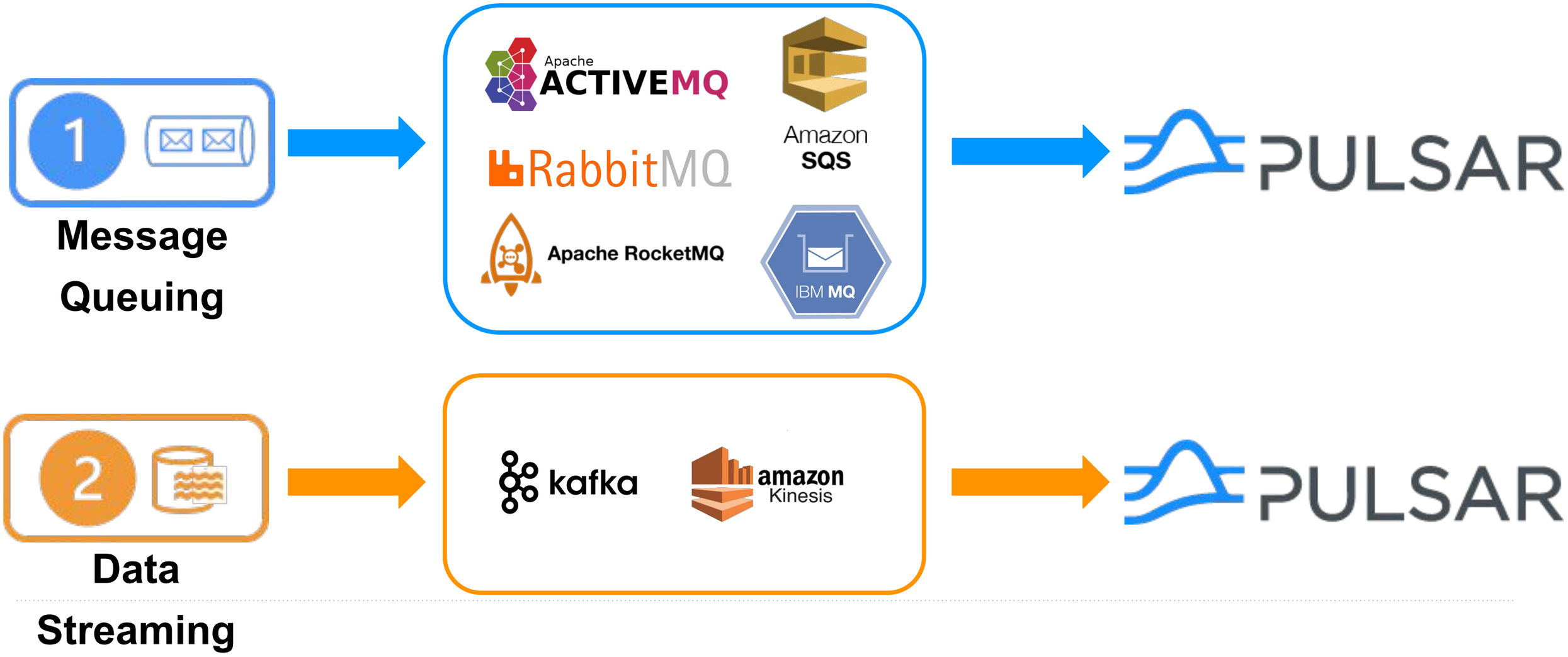


Unified Messaging Model

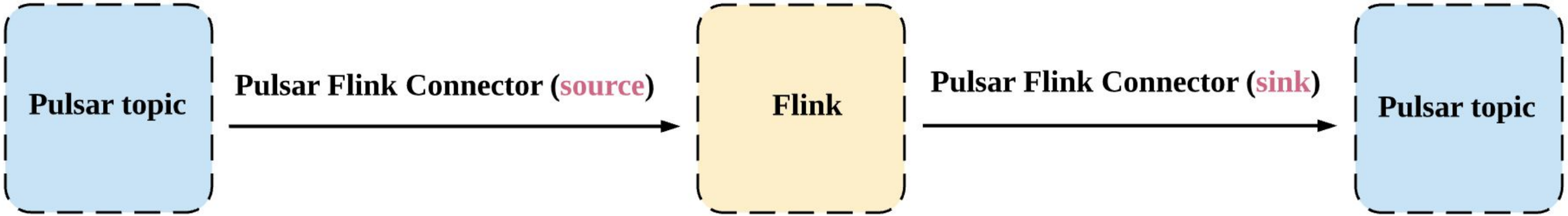


StreamNative

A Unified Messaging Platform



FLiP Stack (FLink -integrate- Pulsar)



https://hub.streamnative.io/data-processing/pulsar-flink/2.7.0/

Stream Native Cloud

A cloud-native, real-time messaging and streaming platform to support multi-cloud and hybrid cloud strategies.

Powered by Pulsar

Built for Containers

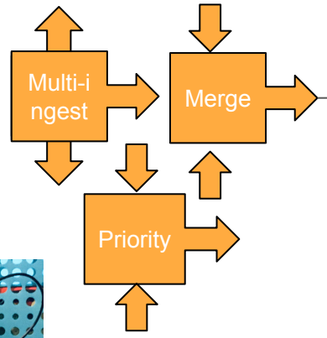
Cloud Native

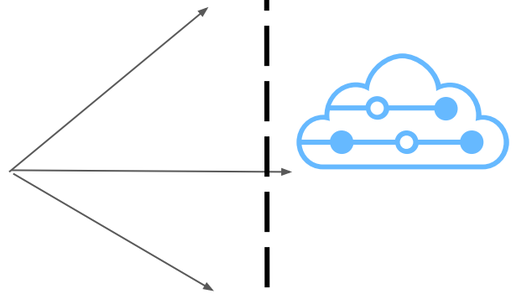
Flink SQL

All Data - Anytime - Anywhere - Any Cloud

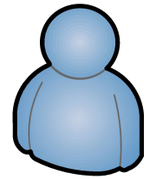
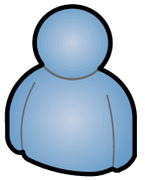
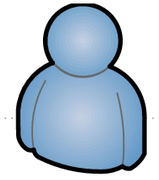


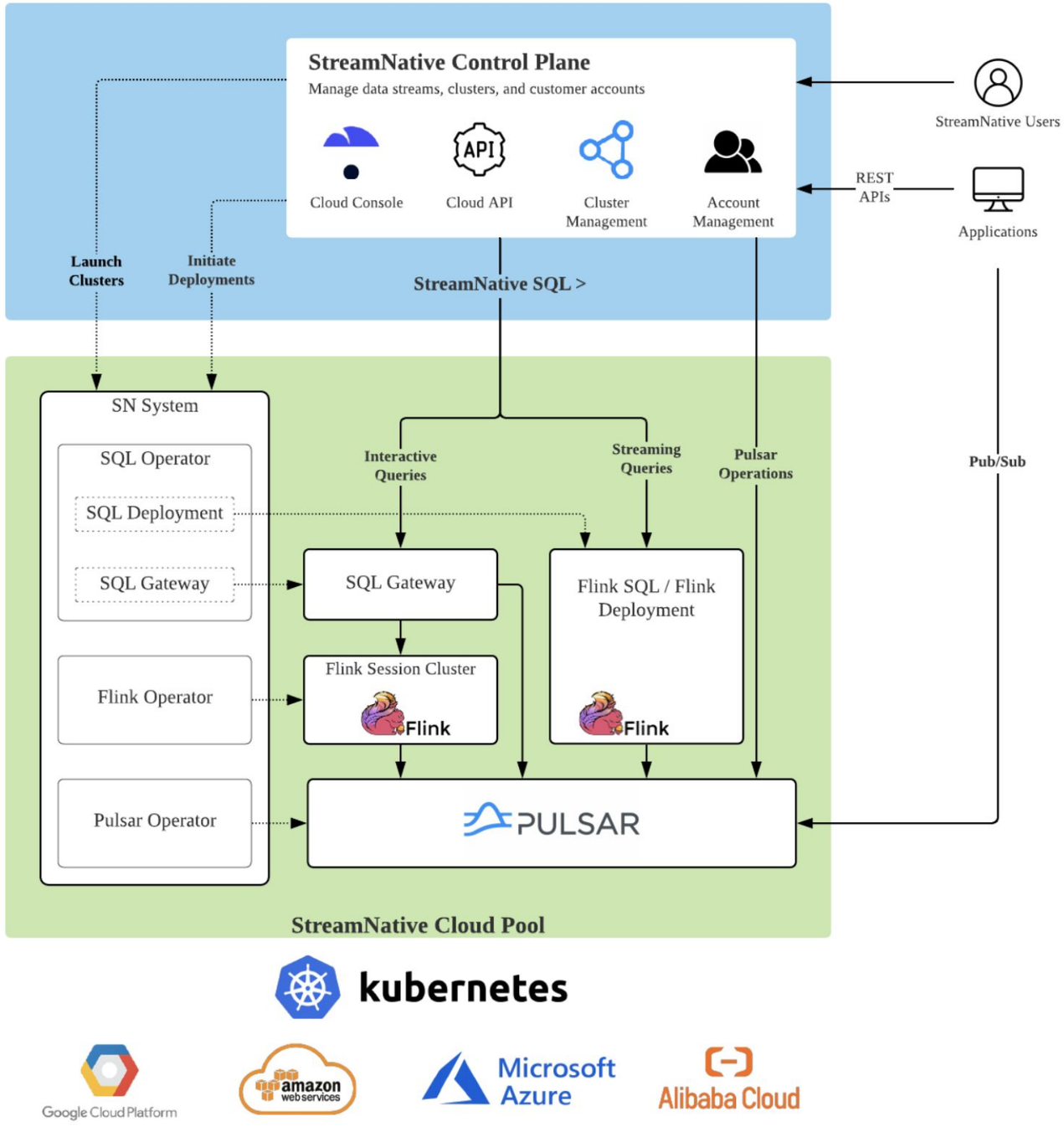







Data Analyst

Data Analyst

Data Scientist




```
|    mqtt-python |
|       mqtt-rp4 |
|       my-topic |
| nvidia-kafka-1 |
|    rp4-kafka-1 |
|           rwar |
|          scada |
|         stocks |
|         stonks |
|        stonkss |
|  topic82547611 |
+----------------+
28 rows in set

Flink SQL> use catalog default_catalog;
[INFO] Execute statement succeed.

Flink SQL> show tables;
+------------+
| table name |
+------------+
|     scada2 |
|     stocks |
+------------+
2 rows in set

Flink SQL> exit;
[INFO] Exiting Flink SQL CLI Client...
```

Demo Walk Through

Create Apache Pulsar Tenants and Namespaces

bin/pulsar-admin tenants create stocks
bin/pulsar-admin namespaces create stocks/inbound
bin/pulsar-admin topics create persistent://stocks/inbound/stocks
bin/pulsar-admin topics create persistent://stocks/inbound/stocks2
bin/pulsar-admin topics list stocks/inbound/
bin/pulsar-client consume -n 0 -s "subs" -p Earliest persistent://stocks/inbound/stocks

```
 // Example Java Code

ProducerBuilder<byte[]> producerBuilder =
client.newProducer().topic(topic)
                .producerName("jetson");
Producer<byte[]> producer =
producerBuilder.create();

String message = "" + jct.message;
MessageId msgID =
producer.newMessage().key(pulsarKey).value(message.
getBytes())

.property("device",OS).send();
```

A cloud data lake that is empty is not useful to anyone.

How can you quickly, scalably and reliably fill your cloud data lake with diverse sources of data you already have and new ones you never imagined you needed. Utilizing open source tools from Apache, the FLaNK stack enables any data engineer, programmer or analyst to build reusable modules with low or no code. FLaNK utilizes Apache NiFi, Apache Kafka, Apache Flink and MiNiFi agents to load CDC, Logs, REST, XML, Images, PDFs, Documents, Text, semistructured data, unstructured data, structured data and a hundred data sources you could never dream of streaming before.

I will teach you how to fish in the deep end of the lake and return a data engineering hero. Let's hope everyone is ready to go from 0 to Petabyte hero.

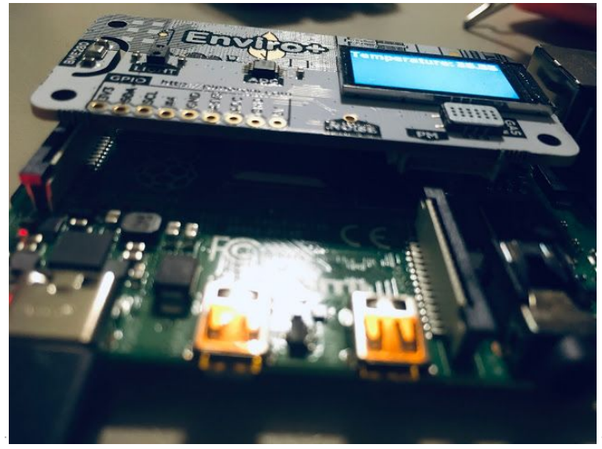
- https://github.com/tspannhw/FLiP-IoT
- https://github.com/tspannhw/FLiP-SQL

WAITING FOR DATA FROM YOUR PIPELINE

Sometimes data from your pipeline never arrives
Sometimes it’s late
Trying to debug hybrid cloud data streams can be hairy
Apache Pulsar and Apache NiFi make this process transparent
Apache NiFi shows logs, errors and provenance via UI, REST and CLI
NiFi and Pulsar have many metrics available via CLI and REST and streamed to Grafana, Prometheus, …
Use StreamNative Cloud for Easy Visibility

SHOW ME SOME DATA


```
{"uuid": "rpi4_uuid_jfx_20200826203733", "amplitude100": 1.2, "amplitude500": 0.6,
"amplitude1000": 0.3, "lownoise": 0.6, "midnoise": 0.2, "highnoise": 0.2, "amps": 0.3,
"ipaddress": "192.168.1.76", "host": "rp4", "host_name": "rp4", "macaddress":
"6e:37:12:08:63:e1", "systemtime": "08/26/2020 16:37:34", "endtime": "1598474254.75",
"runtime": "28179.03", "starttime": "08/26/2020 08:47:54", "cpu": 48.3, "cpu_temp":
"72.0", "diskusage": "40219.3 MB", "memory": 24.3, "id":
"20200826203733_28ce9520-6832-4f80-b17d-f36c21fd8fc9", "temperature": "47.2", "adjtemp":
"35.8", "adjtempf": "76.4", "temperatureF": "97.0", "pressure": 1010.0, "humidity": 8.3,
"lux": 67.4, "proximity": 0, "oxidising": 77.9, "reducing": 184.6, "nh3": 144.7, "gasKO":
"Oxidising: 77913.04 Ohms\nReducing: 184625.00 Ohms\nNH3: 144651.47 Ohms"}
```

SHOW ME SOME DATA

1378:0	{"uuid":"xav_uuid_video0_udz_20210818200243","camera":"/dev/video0","ipaddress":"192.168.1.175","networktime":4.455071926116943,"top1pct":12.9638671875,"top1":"tow truck, tow car, wrecker","cputemp":"39.0","gputemp":"38.5","gputempf":"101","cputempf":"102","runtime":"4","host":"nvidia-desktop","filename":"/home/nvidia/nvme/images/out_video0_qhu_20210818200243.jpg","host_name":"nvidia-desktop","macaddress":"70:66:55:15:b4:a5","end":"1629316967.7859843","te":"4.153049945831299","systemtime":"08/18/2021 16:02:47","cpu":7.9,"diskusage":"33179.4 MB","memory":33.5,"id":"video0_20210818200243_2e3e0dff-41e6-4b90-8782-2aa7fdd92ed4","imageinput":"/home/nvidia/nvme/images/img_video0_jwr_20210818200243.jpg"}
1378:1	{"uuid":"xav_uuid_video0_udz_20210818200243","camera":"/dev/video0","ipaddress":"192.168.1.175","networktime":4.455071926116943,"top1pct":12.9638671875,"top1":"tow truck, tow car, wrecker","cputemp":"39.0","gputemp":"38.5","gputempf":"101","cputempf":"102","runtime":"4","host":"nvidia-desktop","filename":"/home/nvidia/nvme/images/out_video0_qhu_20210818200243.jpg","host_name":"nvidia-desktop","macaddress":"70:66:55:15:b4:a5","end":"1629316967.7859843","te":"4.153049945831299","systemtime":"08/18/2021 16:02:47","cpu":7.9,"diskusage":"33179.4 MB","memory":33.5,"id":"video0_20210818200243_2e3e0dff-41e6-4b90-8782-2aa7fdd92ed4","imageinput":"/home/nvidia/nvme/images/img_video0_jwr_20210818200243.jpg"}

Message Properties

{"uuid":"xav_uuid_video0_udz_20210818200243","camera":"/dev/video0","ipaddress":"192.168.1.175","networktime":4.455071926116943,"top1pct":12.9638671875,"top1":"tow truck, tow car, wrecker","cputemp":"39.0","gputemp":"38.5","gputempf":"101","cputempf":"102","runtime":"4","host":"nvidia-desktop","filename":"/home/nvidia/nvme/images/out_video0_qhu_20210818200243.jpg","host_name":"nvidia-desktop","macaddress":"70:66:55:15:b4:a5","end":"1629316967.7859843","te":"4.153049945831299","systemtime":"08/18/2021 16:02:47","cpu":7.9,"diskusage":"33179.4 MB","memory":33.5,"id":"video0_20210818200243_2e3e0dff-41e6-4b90-8782-2aa7fdd92ed4","imageinput":"/home/nvidia/nvme/images/img_video0_jwr_20210818200243.jpg"}

Message **Properties**


```
{
 "X-Pulsar-num-batch-message": "1",
 "device": "linux",
 "publish-time": "2021-08-18T21:19:43.266Z",
 "uuid2": "0c10faf1-8b8f-400c-b37c-1929e1b9ed2e"
}
```

DEEPER CONTENT

- https://www.datainmotion.dev/2020/10/running-flink-sql-against-kafka-using.html
- https://www.datainmotion.dev/2020/10/top-25-use-cases-of-cloudera-flow.html
- https://github.com/tspannhw/EverythingApacheNiFi
- https://github.com/tspannhw/CloudDemo2021
- https://github.com/tspannhw/StreamingSQLExamples
- https://www.linkedin.com/pulse/2021-schedule-tim-spann/
- https://github.com/tspannhw/StreamingSQLExamples/blob/8d02e62260e82b027b43abb911b5c366a3081927/README.md

Connect with the Community & Stay Up-To-Date

- Join the Pulsar Slack channel - *Apache-Pulsar.slack.com*
- Follow @streamnativeio and @apache_pulsar on Twitter
- Subscribe to Monthly Pulsar Newsletter for major news, events, project updates, and resources in the Pulsar community

Interested In Learning More?

Resources

[Function Mesh - Simplify Complex Streaming Jobs in Cloud]

[The GitHub Source Code for Demo]

Free eBooks

[Manning's Apache Pulsar in Action]

[O’Reilly Book]

Upcoming Events

[[10/6] Pulsar Summit Europe]

The Stack

Home About Team Leadership + Follow

Home / Featured / One to Watch: StreamNative raises $23 million to take Apache Pulsar mainstream

Featured

One to Watch: StreamNative raises $23 million to take Apache Pulsar mainstream

Sijie Guo of Streamlio by SRK Headshot Day

Whether you're a utility pulling performance data from the sensor-laden turbines of an offshore windfarm, an ecommerce provider responding to a spike in orders, or a bank seeking intelligence on sudden surge in withdrawals, real-time intelligence from streamed data has never been more important. Yet often the behind-

Source: Data streaming service StreamNative takes in $23.7M

Q&A