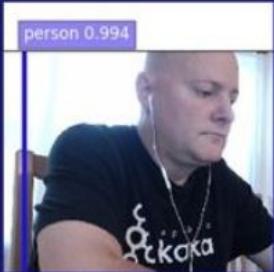




**Stream
Native**

Real-Time Streaming in Azure

Timothy Spann | Developer Advocate



Real-Time Streaming in Azure

with Timothy Spann

13 October 2021

18:00 (GMT+1)



www.cloudlunchlearn.com

In collaboration with





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/>





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



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

What are the Benefits of Pulsar?



Multi-Tenancy

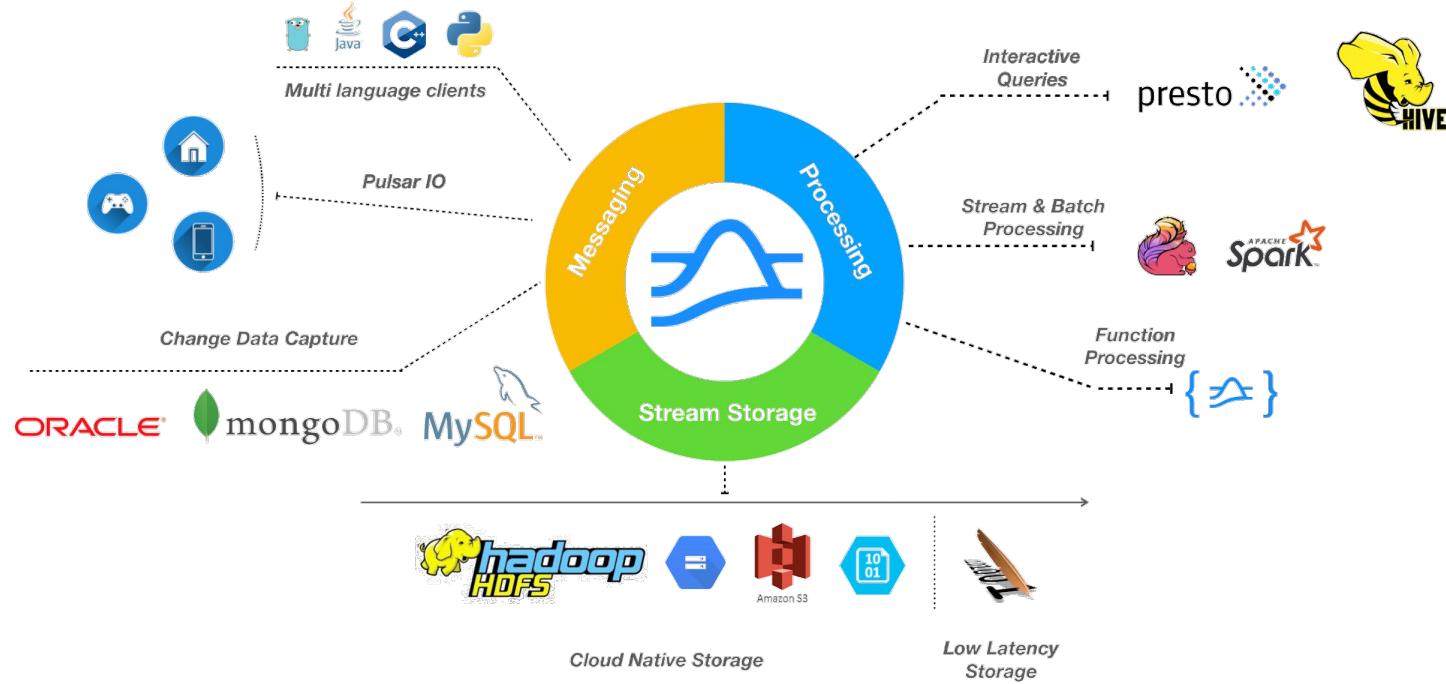
Scalability

Geo-Replication

Unified Messaging
Model

Data Durability

Apache Pulsar



A Unified Messaging Platform



The Difference Between Messaging & Streaming

Messaging

Messaging systems are ideal for work queues that do not require tasks to be performed in a particular order—for example, sending one email message to many recipients.

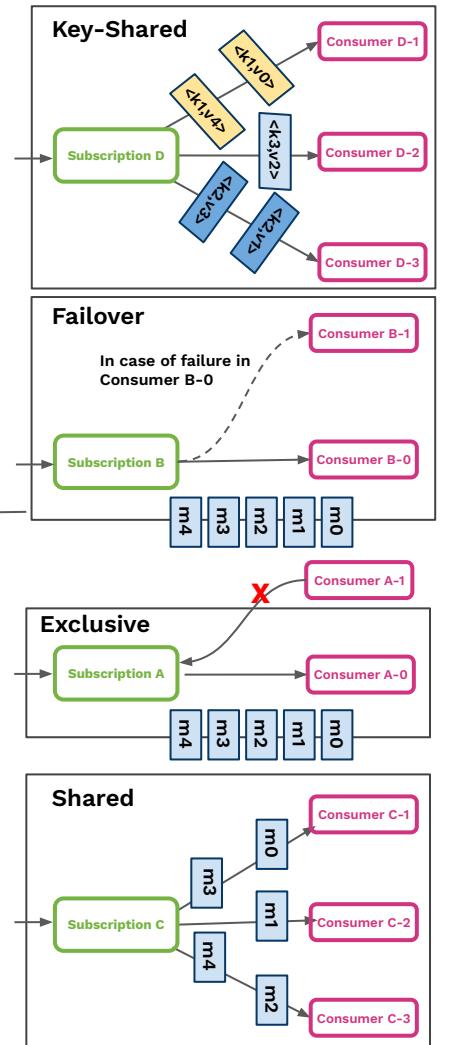
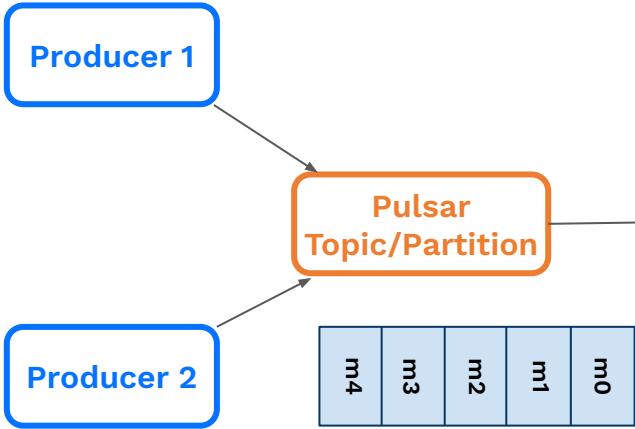
[RabbitMQ](#) and [Amazon SQS](#) are examples of popular queue-based message systems.

Streaming

Streaming works best in situations where the order of messages is important—for example, data ingestion.

[Kafka](#) and [Amazon Kinesis](#) are examples of messaging systems that use streaming semantics for consuming messages.

Unified Messaging Model

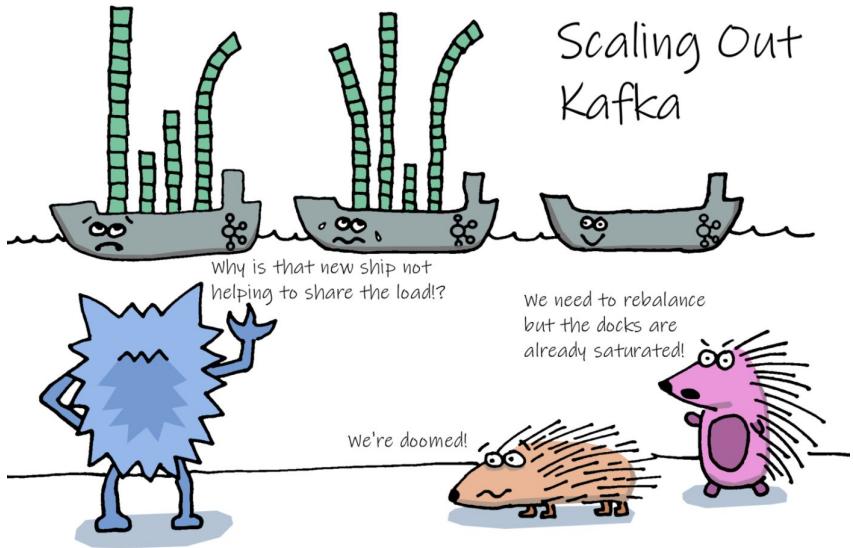


Streaming

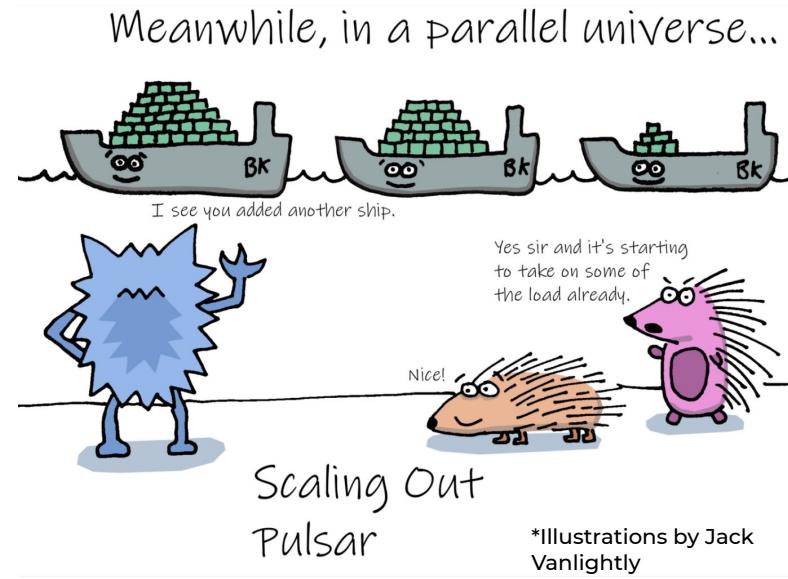
Messaging

Pulsar is Cloud Native

- Rebalance free architecture via separate compute and storage for easy scalability
- Built for Kubernetes, with production ready helm charts and Kubernetes operators
- Infinite streams by leveraging cost effective cloud-storage like S3, GCS, etc
- Supports a broad range of workloads and teams through unified messaging model



Scaling Out Kafka



*Illustrations by Jack Vanlightly

Pulsar is built for easy scale-out.

Pulsar Global Adoption



splunk®

Flipkart



verizon
media

YAHOO!
JAPAN

Tencent 腾讯

OVHcloud

NUTANIX™

COMCAST

narvar

INSTRUCTURE



BestPay



ITERABLE

overstock.

中国电信
CHINA TELECOM

tuya.com

toast

clever cloud

IoTium

</n^dustrial.io>

mercado
libre

OKCOIN

360
www.360.cn

proxyclick

BrandsEye

Huya

THE HUT GROUP®

Giggso

Kingsoft Cloud
KS CLOUD

Max Kelsen

环球易购
Globalegrow E-Commerce

VIP KID

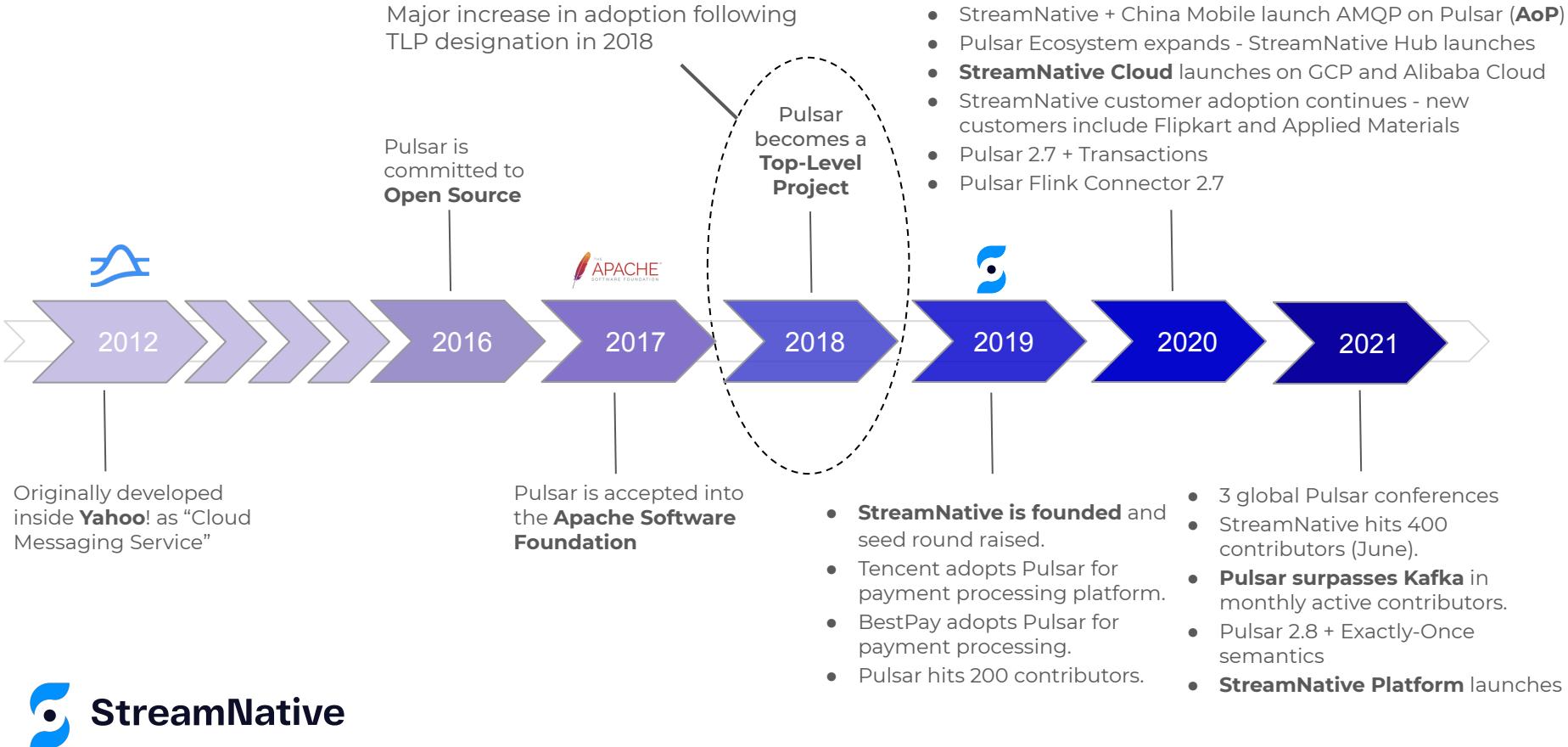
EMQ

5G+

中国移动
China Mobile

StreamNative

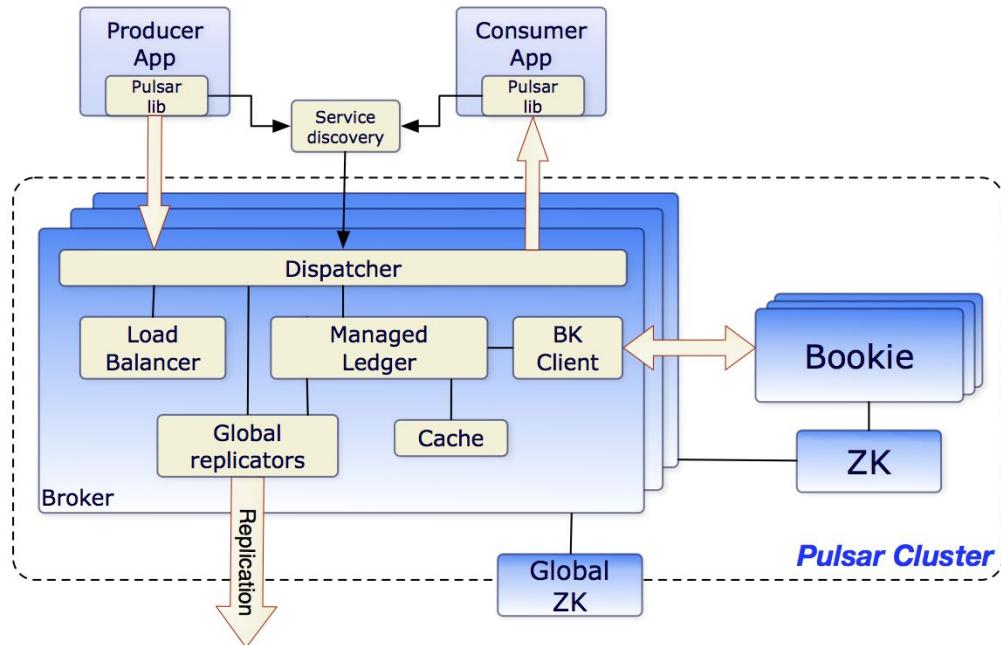
Key Milestones



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

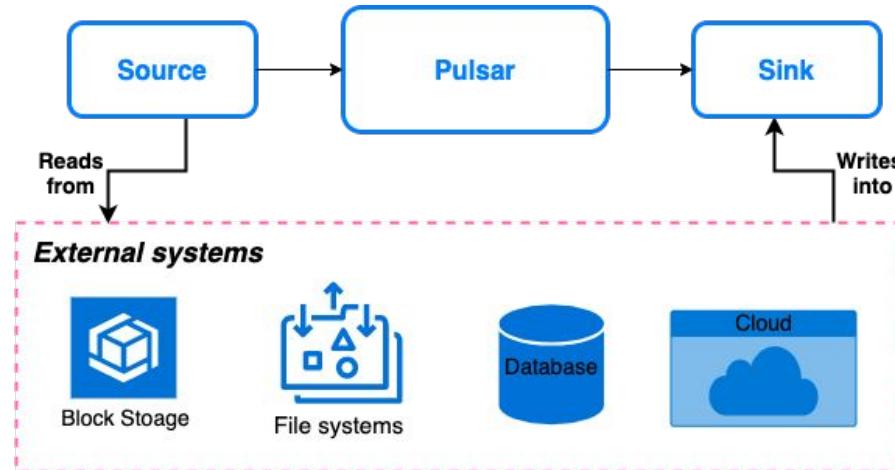
IO/Connectors are a simple way to integrate with external systems and move data in and out of Pulsar.

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



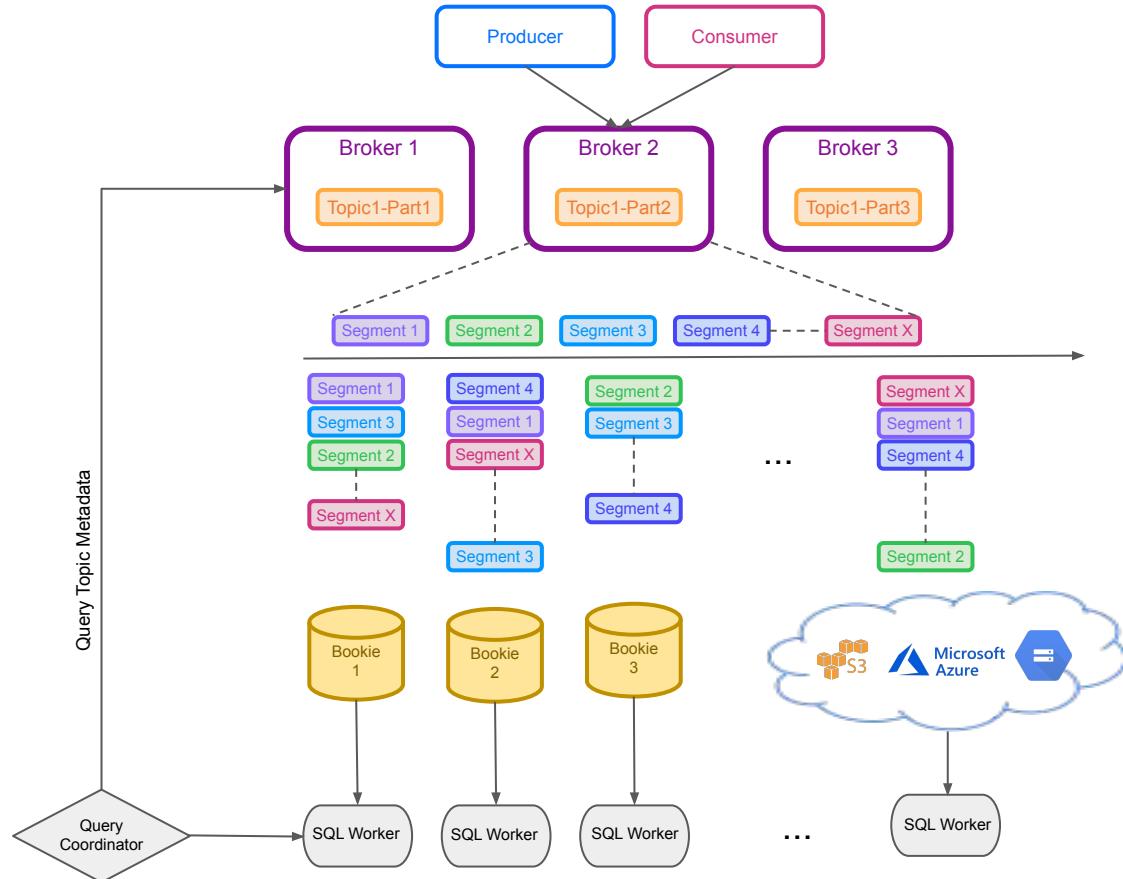
Use Azure BlobStore offloader with Pulsar

<https://pulsar.apache.org/docs/en/tiered-storage-azure/>



Pulsar SQL

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



Query Your Topics with Pulsar SQL (Trino)

```
presto> select camera, cpu, cputempf, gputempf, memory, top1, top1pct, uuid, __publish_time__, __message_id__, __key__ from pulsar."public/default".iotjetsonjson;
          camera |   cpu | cputempf | gputempf | memory |      top1 |    top1pct |        uuid | __publish_time__ | __message_id__ | __key__
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
/dev/video0 |  8.7 |   82 |    82 |   33.5 | microphone, mike | 18.85986328125 | xav_uuid_video0_lgl_20211001183019 | 2021-10-01 14:30:30.657 | (564,3,0) |
/dev/video0 |  8.7 |   82 |    82 |   33.6 | microphone, mike | 19.22607421875 | xav_uuid_video0_kpt_20211001183033 | 2021-10-01 14:30:44.380 | (564,4,0) |
/dev/video0 | 12.0 |   80 |    81 |   33.5 | microphone, mike | 12.53662109375 | xav_uuid_video0_gzd_20211001182930 | 2021-10-01 14:29:48.756 | (564,0,0) |
/dev/video0 |  8.5 |   82 |    82 |   33.6 | microphone, mike |       14.0625 | xav_uuid_video0_wlw_20211001182951 | 2021-10-01 14:30:02.919 | (564,1,0) |
/dev/video0 |  8.5 |   82 |    82 |   33.5 | microphone, mike |     29.8828125 | xav_uuid_video0_ulq_20211001183005 | 2021-10-01 14:30:16.787 | (564,2,0)
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
[5 rows]
[END]
```

```
presto> show tables in pulsar."public/default";
          Table
-----
generator_test
iotjetsonjson
mqtt-2
(3 rows)

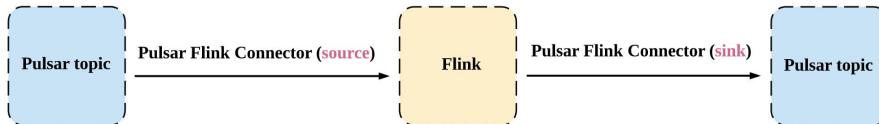
Query 20211001_054538_00008_s8x23, FINISHED, 1 node
Splits: 19 total, 19 done (100.00%)
0:00 [3 rows, 105B] [14 rows/s, 493B/s]
```

Apache Flink

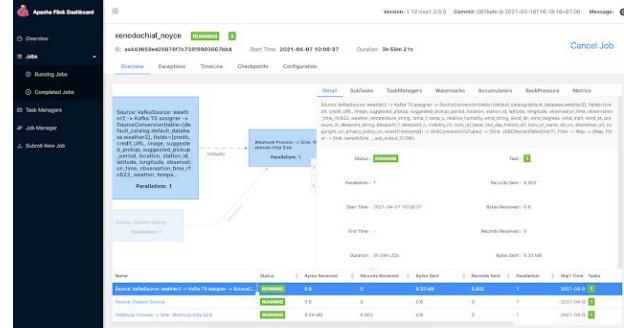




Why Apache Flink?



- Unified computing engine
- Batch processing is a special case of stream processing
- Stateful processing
- Massive Scalability
- Flink SQL for queries, inserts against Pulsar Topics
- Streaming Analytics
- Continuous SQL
- Continuous ETL
- Complex Event Processing
- Standard SQL Powered by Apache Calcite



Apache Flink

- Apache Flink is a distributed stream processing system.
- It is capable of providing high throughput, near real-time processing of streams from Pulsar.
- It is ideal for *ambitious* Stream Processing compared to Pulsar's model of lightweight Stream Processing.



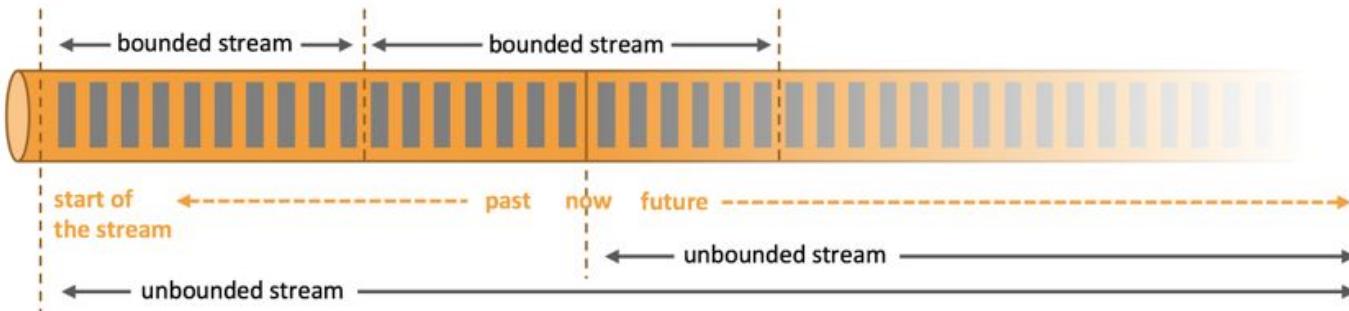
Flink

Some Apache Flink Users



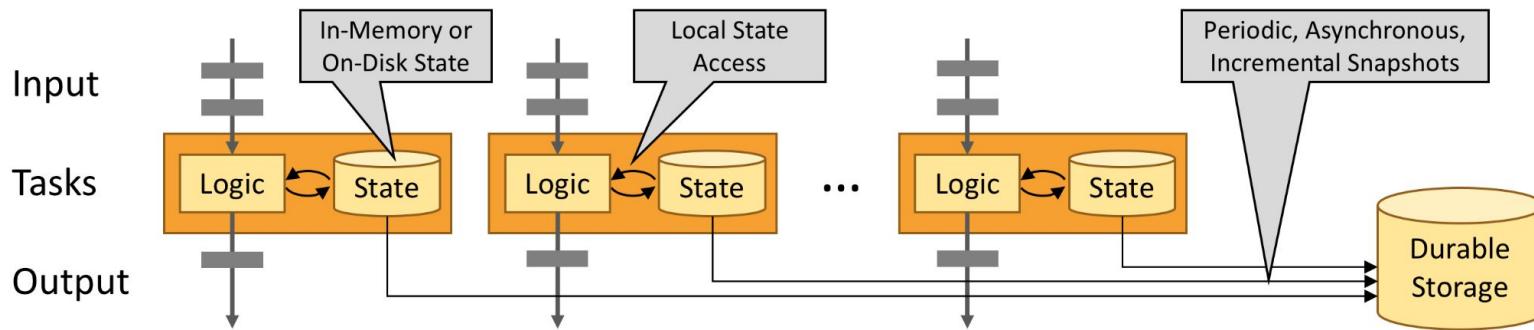
Everything Is a Stream

- Stream: Sequence of data which is made available over time
- All computation processes chunks of data over time producing results over time → Stream processing
 - E.g. reading data from disks is done in streaming fashion
- Events can be of various forms
- Decisive difference: Is my stream bounded or not?



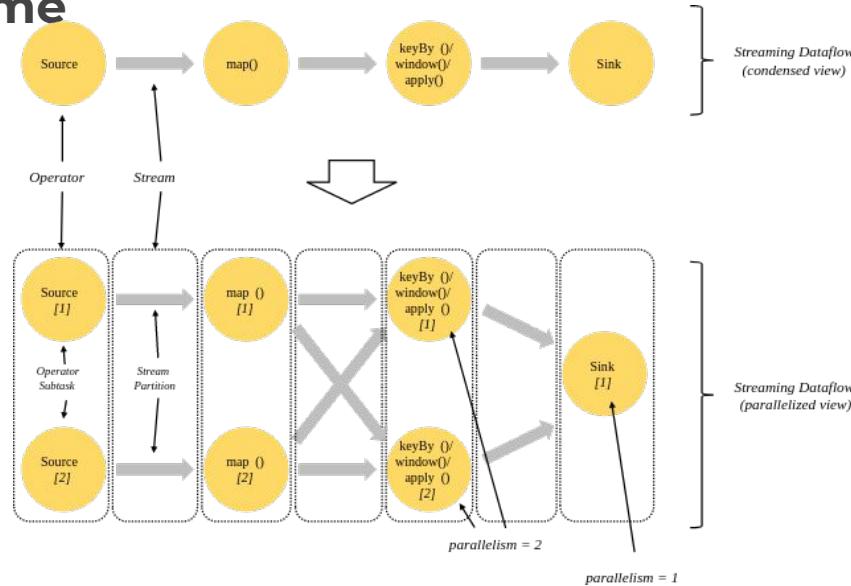
How Flink Works

Flink Provides an API for **handling events** and **intermediate state** and **periodic snapshots** to ensure consistency



How Flink Works

Flink apps are composed of a **graph of tasks** which is executed in a **distributed runtime**



How Flink Works

Flink provides **multiple APIs at different layers of abstraction**

High-level
Analytics API

Stream- & Batch
Data Processing

Stateful Event-
Driven Applications

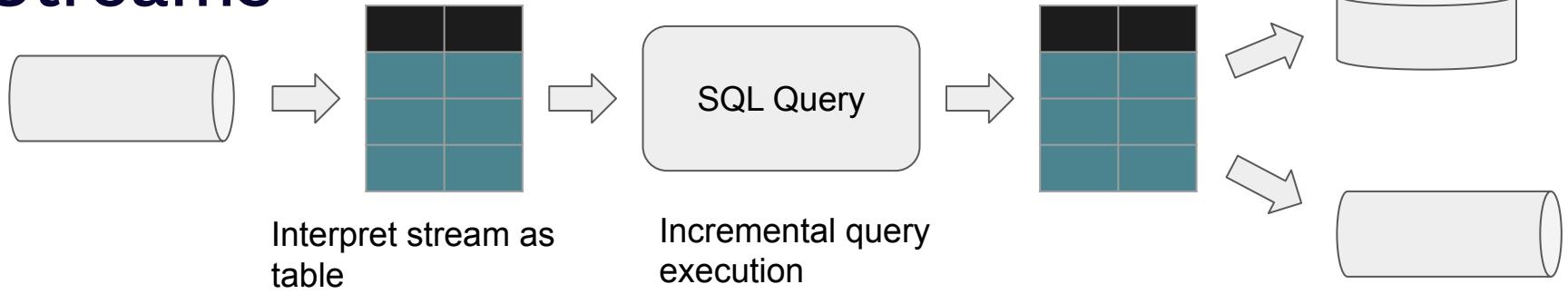
SQL / Table API (dynamic tables)

DataStream API (streams, windows)

ProcessFunction (events, state, time)

- Conciseness +
+ Expressiveness -

SQL / Table API: Running The Same Query On Streams



```
SELECT
    room,
    TUMBLE_END(rowtime, INTERVAL '1' HOUR),
    AVG(temperature)
FROM
    sensors
GROUP BY
    TUMBLE(rowtime, INTERVAL '1' HOUR), room
```

StreamNative Cloud



StreamNative Cloud

Powered by Apache Pulsar, StreamNative provides a cloud-native, real-time messaging and streaming platform to support multi-cloud and hybrid cloud strategies.



Cloud Native



kubernetes

Built for Containers



Flink

Flink SQL



StreamNative

StreamNative

Powered by Apache Pulsar, StreamNative provides a cloud-native, unified messaging and streaming platform to support multi-cloud and hybrid cloud strategies.



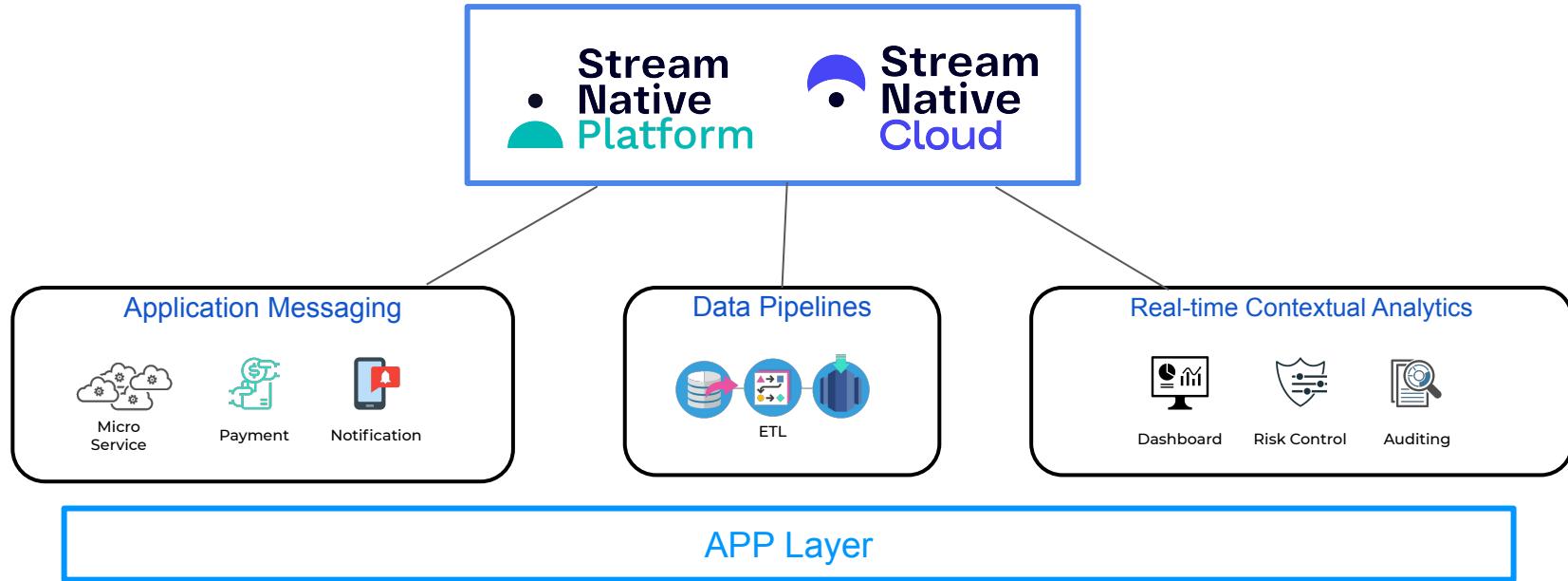
Cloud Native



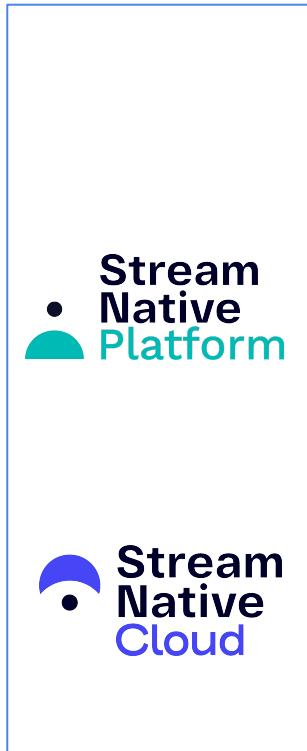
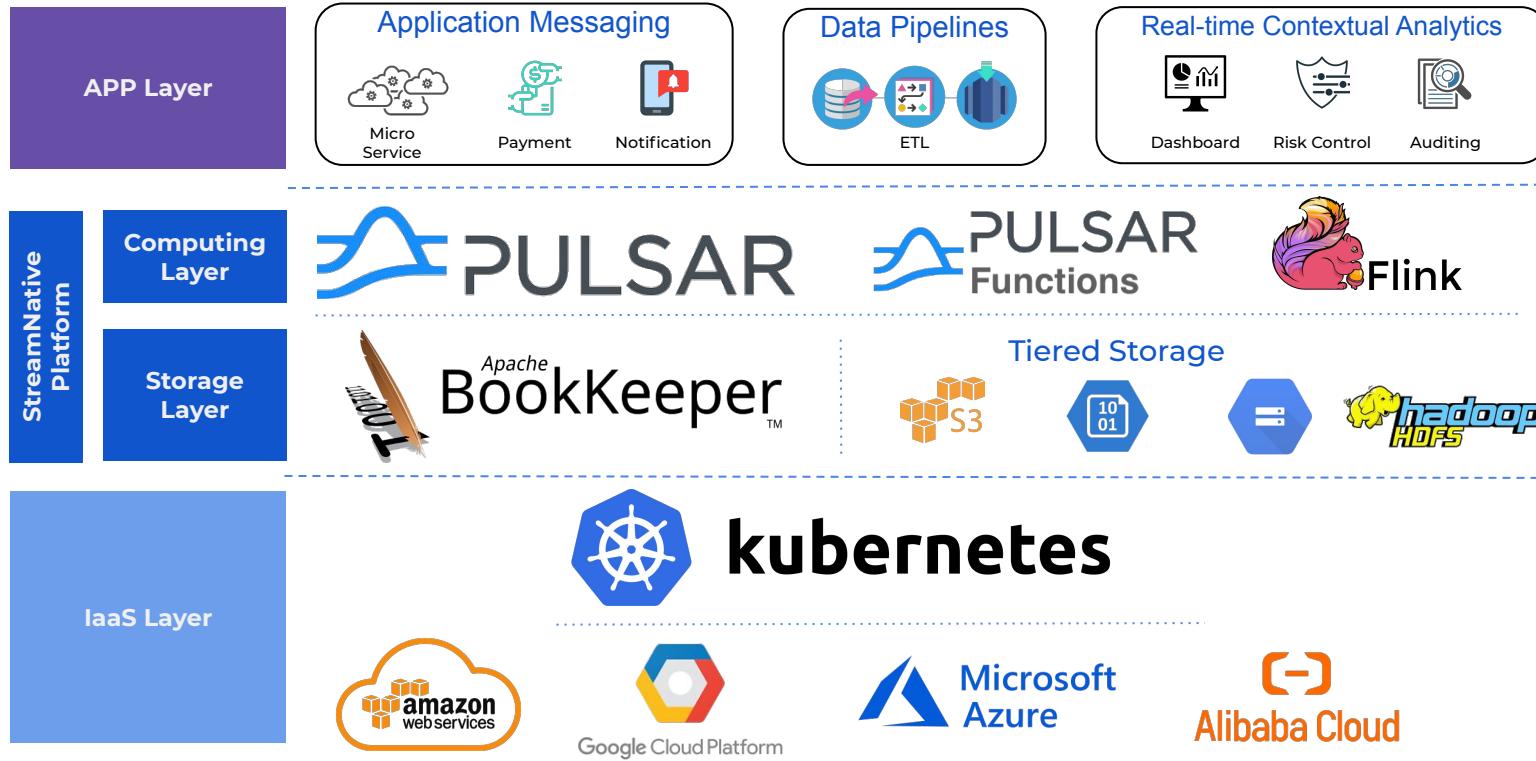
kubernetes

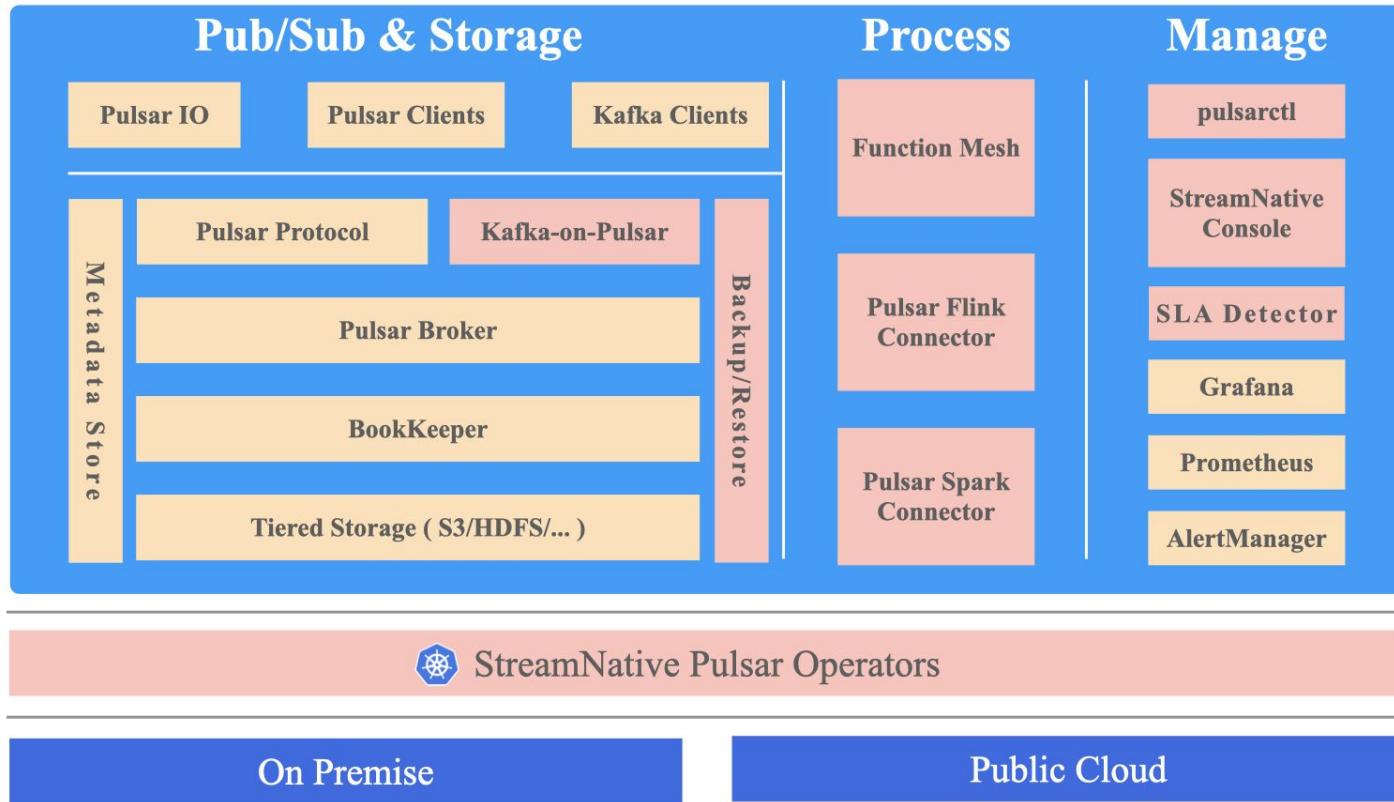
Built for Containers

The StreamNative Offering



StreamNative Solution



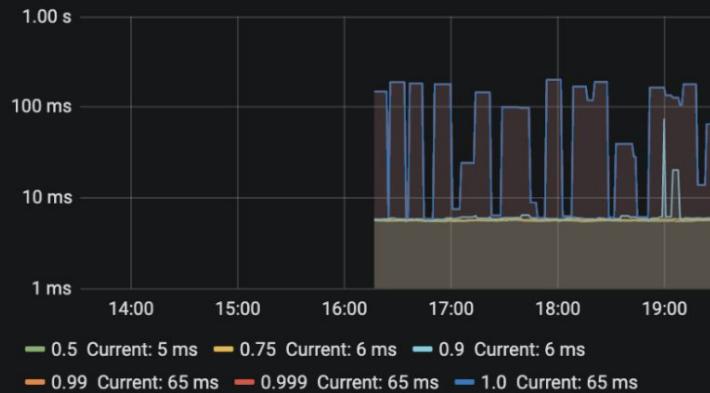


StreamNative

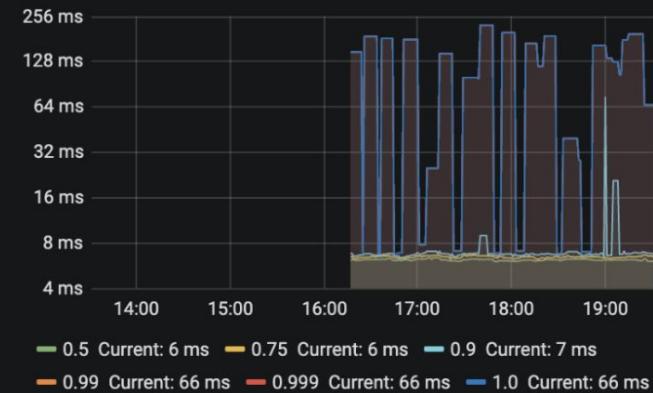
e2e failure count



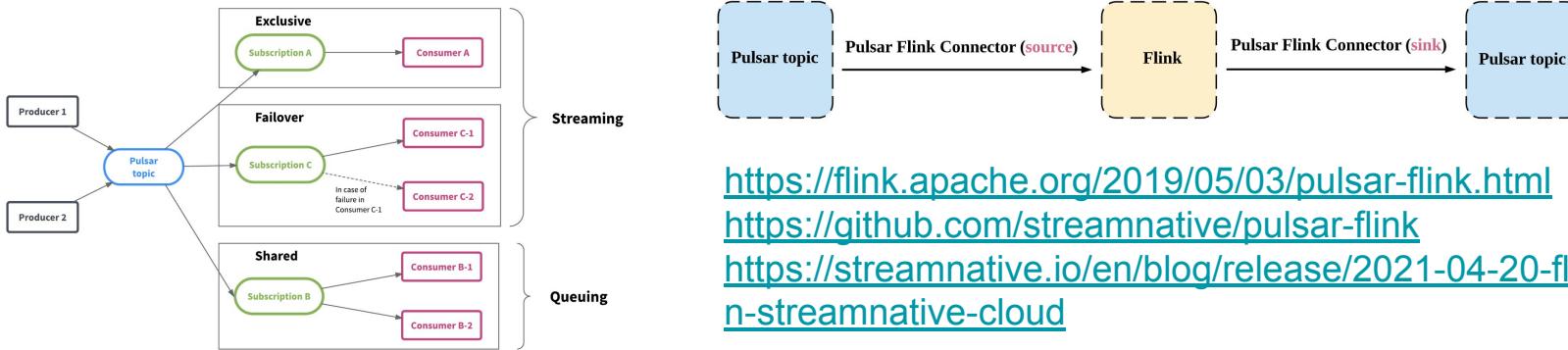
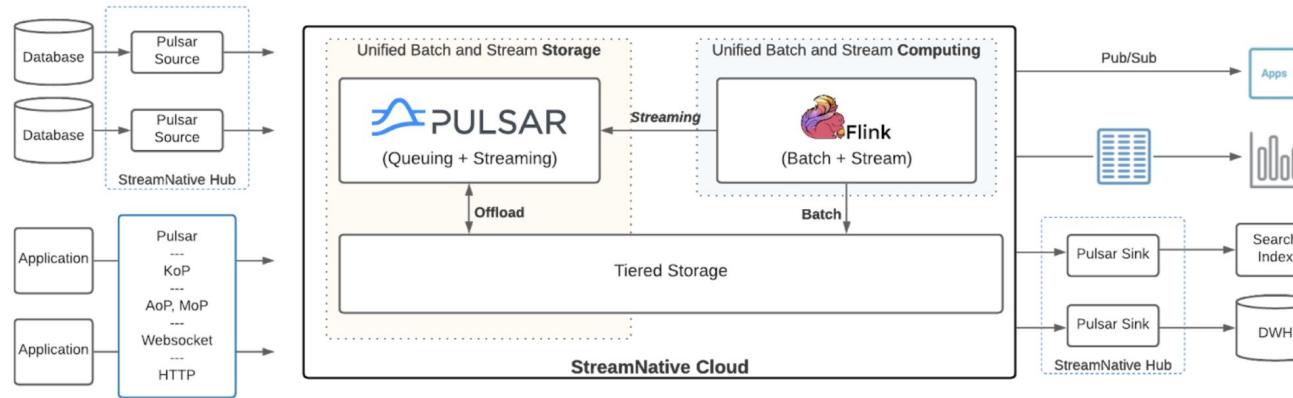
publish_latency



e2e_latency



Flink + Pulsar



<https://flink.apache.org/2019/05/03/pulsar-flink.html>

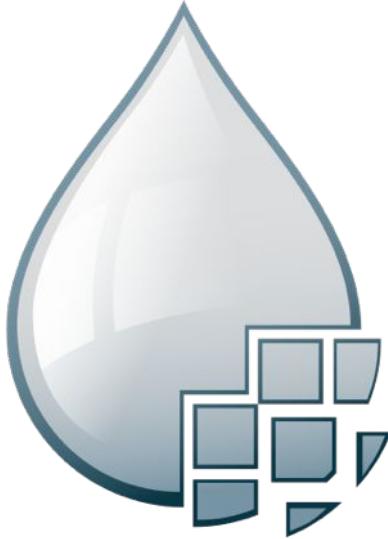
<https://github.com/streamnative/pulsar-flink>

<https://streamnative.io/en/blog/release/2021-04-20-flink-sql-on-streamnative-cloud>

Apache NiFi

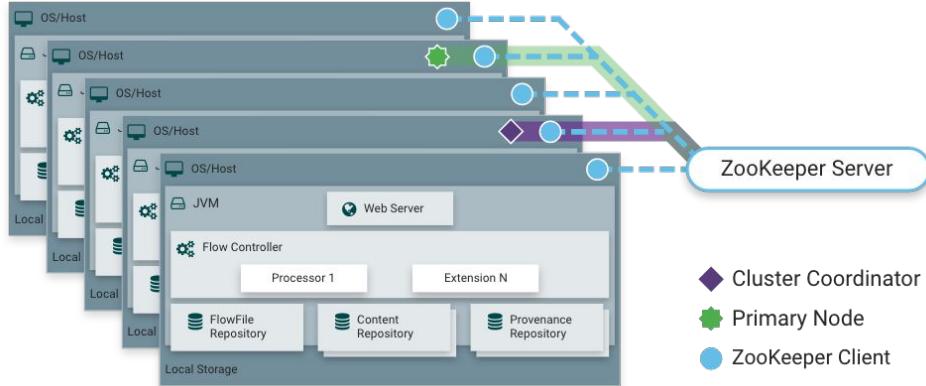
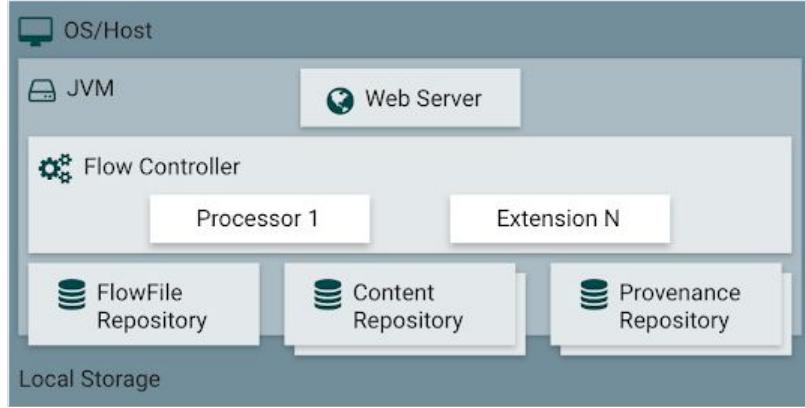


Why Apache NiFi?



- 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



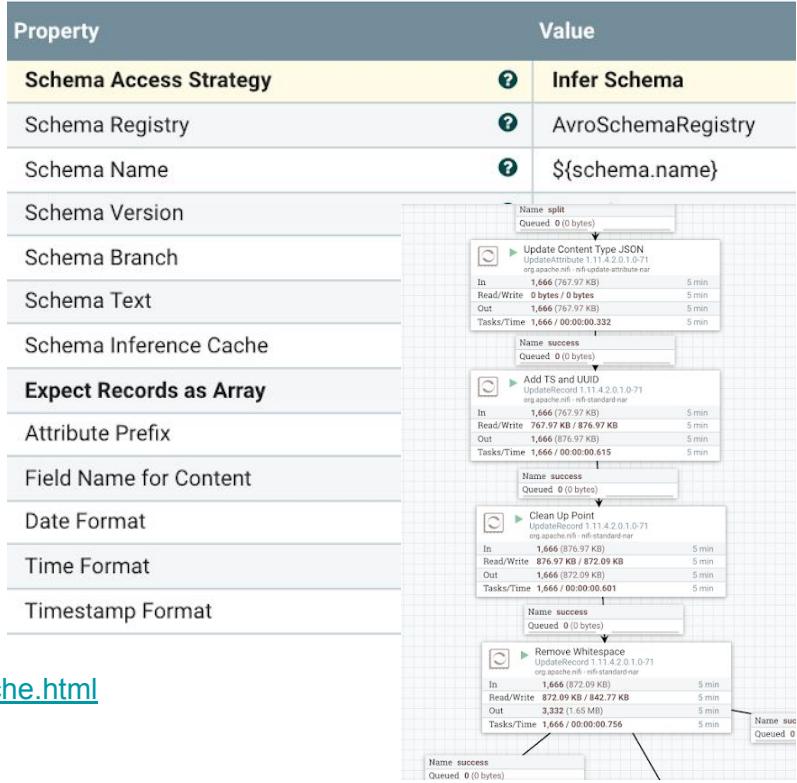
<https://nifi.apache.org/docs/nifi-docs/html/overview.html>

Record Processors

- XML, CSV, JSON, AVRO and more
- Schemas or Inferred Schemas
- Easily convert between them
- Support SQL with Apache Calcite

Property	Value
Record Reader	XMLReader
Record Writer	JsonRecordSetWriter
Include Zero Record FlowFiles	false
Cache Schema	true
query1	SELECT * FROM FLOWFILE

<https://www.datainmotion.dev/2019/03/advanced-xml-processing-with-apache.html>



Provenance

NiFi Data Provenance

Displaying 165 of 165
Oldest event available: 12/21/2020 16:55:33 UTC

Filter by component name ▾

Date/Time	Type	Flowfile Uuid	Size
12/22/2020 16:54:17.193 UTC	ATTRIBUTES_MODIFIED	6fbbae84f6ba4-47c3-ba03-8830ec7cd3db	89 byte
12/22/2020 16:54:17.192 UTC	ATTRIBUTES_MODIFIED	1233e8d4e84d-4218-b3d0-2598e7f901e	87 byte
12/22/2020 16:54:14.194 UTC	ATTRIBUTES_MODIFIED	37fbca2153-4185-4460-b633-7eb474ed5718	81 byte
12/22/2020 16:54:05.297 UTC	ATTRIBUTES_MODIFIED	699d6fc9-9d71-4cf6-b733-b5c4e05d362	83 byte
12/22/2020 16:53:59.296 UTC	ATTRIBUTES_MODIFIED	df43c05c-5aae-44c2-9edc-c20a8148604	84 byte
12/22/2020 16:53:59.295 UTC	ATTRIBUTES_MODIFIED	4b1db81c-1f83-4a99-b309-2da1277cd7c6	84 byte
12/22/2020 16:53:58.296 UTC	ATTRIBUTES_MODIFIED	45fe8edd-cx55-431e-82b9-436c4a4092e	81 byte
12/22/2020 16:53:57.298 UTC	ATTRIBUTES_MODIFIED	b07034b-6361-4c34-b	
12/22/2020 16:53:57.297 UTC	ATTRIBUTES_MODIFIED	d12601a-7974-4c16-b	
12/22/2020 16:53:57.297 UTC	ATTRIBUTES_MODIFIED	29966d80-4153-41bc-a	
12/22/2020 16:53:43.753 UTC	ATTRIBUTES_MODIFIED	1ca5c744-1cb4-4ff1-bb	
12/22/2020 16:53:37.747 UTC	ATTRIBUTES_MODIFIED	faf647db-9e65-48c0-a	
12/22/2020 16:53:21.646 UTC	ATTRIBUTES_MODIFIED	df1f60ff-6d65-460e-99	
12/22/2020 16:53:05.515 UTC	ATTRIBUTES_MODIFIED	964695fc-d953-44c0-b	
12/22/2020 16:52:43.374 UTC	ATTRIBUTES_MODIFIED	79fcfa90-b160-4fc4-8a	
12/22/2020 16:52:29.308 UTC	ATTRIBUTES_MODIFIED	3453eeb9-953c-4952-a	
12/22/2020 16:52:29.307 UTC	ATTRIBUTES_MODIFIED	a166e297-118a-4262-9	
12/22/2020 16:52:29.307 UTC	ATTRIBUTES_MODIFIED	bd2946fd-5a99-42d7-b	
12/22/2020 16:52:29.307 UTC	ATTRIBUTES_MODIFIED	a16841bc-2505-4c8c-b	
12/22/2020 16:52:29.306 UTC	ATTRIBUTES_MODIFIED	578506fe-e449-471f-a	
12/22/2020 16:52:29.306 UTC	ATTRIBUTES_MODIFIED	3d44c5fb-4737-4a9e-82	
12/22/2020 16:52:29.306 UTC	ATTRIBUTES_MODIFIED	4dc93a17-7059-424e-9	
12/22/2020 16:52:29.306 UTC	ATTRIBUTES_MODIFIED	9fb9d9c1-f394-4c11-93	

Provenance Event

DETAILS ATTRIBUTES

Attribute Values

lastprice

123.66

No value set

symbol

IBM

No value set

timestamp

1608654962884

No value set

volume

100

No value set

<https://www.datainmotion.dev/2021/01/automating-starting-services-in-apache.html>

Backpressure & Prioritizers

Configure Connection

DETAILS SETTINGS

Name	Available Prioritizers <small>?</small>		
	FirstInFirstOutPrioritizer		
	NewestFlowFileFirstPrioritizer		
	OldestFlowFileFirstPrioritizer		
	PriorityAttributePrioritizer		
Id 3ca22430-cba4-3347-b45b-7bdc3530bd7e	Selected Prioritizers	<small>?</small>	
FlowFile Expiration <small>?</small> 0 sec			
Back Pressure Object Threshold <small>?</small> 10000	Size Threshold <small>?</small> 1 GB	Selected Prioritizers <small>?</small>	
Load Balance Strategy <small>?</small> Do not load balance	<small>▼</small>		

<https://www.datainmotion.dev/2019/11/exploring-apache-nifi-110-parameters.html>

System Diagnostics

NiFi Summary

Processors	Input Ports	Output Ports	Remote Process Groups	Connections	Process Groups	NiFi Summary									
Displaying 1,338 of 1,338															
Filter by name															
Name	Type	Process Group	Run Status	In (Size) 5 min	Read Write 5 min	Out (Size) 5 min	Tasks Time 5 min								
PublishPulsar	PublishPulsar	Status Pulsar	Running (1)	61 (50.13 KB)	50.13 KB 0 bytes	0 (0 bytes)	5,504,778 00:00:23.264	—	—	—					
Acquire Satellite Data	GenerateFlowFile	Satellite Data	Disabled	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	—	—	—					
Acquire Satellite Data	GenerateFlowFile	Satellite Data	Disabled	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	—	—	—					
Analyze Data in Stream	QueryRecord	Fresh Food	Disabled	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	—	—	—					
App Data	PublishKafkaRecord_2_0	Mobile Ingest	Disabled	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	—	—	—					
App Data	PublishKafkaRecord_2_0	Mobile Ingest	Disabled	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	—	—	—					
AttributeCleanerProcessor	AttributeCleanerProcessor	Mobile Ingest	Disabled	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	—	—	—					
AttributeCleanerProcessor	AttributeCleanerProcessor	Mobile Ingest	Disabled	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	—	—	—					
Attributes Grab	EvaluateJsonPath	Predict Temperature	Disabled	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	—	—	—					

Flow File

Flow Files are content and key/value pairs for attributes that are each event/message/file that has been introduced into NiFi.



A screenshot of a web-based interface for viewing FlowFiles. At the top left is a blue water droplet icon. Below it, a dropdown menu shows "View as: formatted". The main area displays a JSON object:

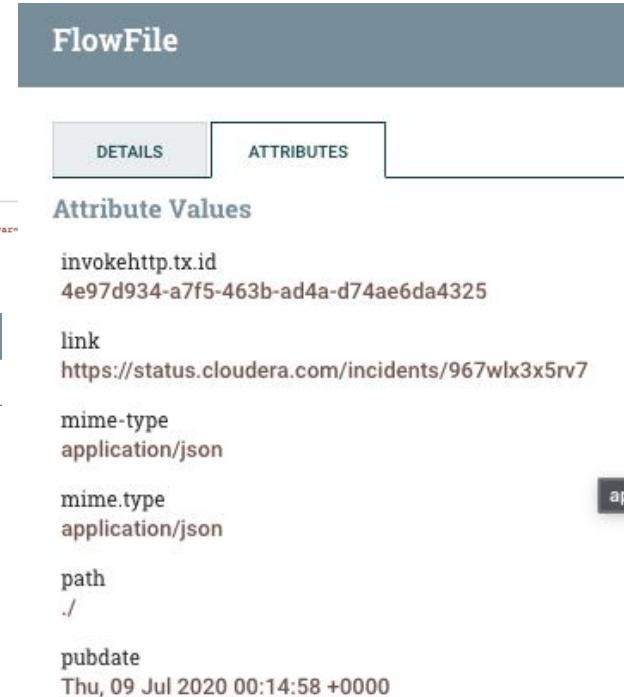
```
1+ | {  
2+ |   "title" : "CDP Control-plane network maintenance (low-impact)",  
3+ |   "description" : "<p><small>Jul 09 2020 00:14:58 +0000</small><br><strong>Completed</strong> - This operation is complete.</p><p><small>Jul 09 2020 00:14:58 +0000</small><br><strong>Completed</strong> - This operation is complete.</p>",  
4+ |   "published" : "Thu, 09 Jul 2020 00:14:58 +0000",  
5+ |   "link" : "https://status.cloudera.com/incidents/967wlx3x5rv7",  
6+ |   "guid" : "https://status.cloudera.com/incidents/967wlx3x5rv7",  
7+ | } }
```



A screenshot of a "FlowFile" details view. At the top is a blue header bar with the word "FlowFile". Below it is a navigation bar with "DETAILS" and "ATTRIBUTES" tabs, where "DETAILS" is selected. The main content area is divided into two columns: "FlowFile Details" and "Content Claim".

FlowFile Details	Content Claim
UUID 29235844-0576-432b-9078-b477f9ff6ff5	Container default
Filename 35714351-11c8-48e0-b52c-6e295edbbb69	Section 2
File Size 1.15 KB	Identifier 1631110638665-2
Queue Position No value set	Offset 50840
Queued Duration 1 days and 03:58:04.364	Size 1.15 KB
Lineage Duration 1 days and 04:01:15.707	
Penalized No	

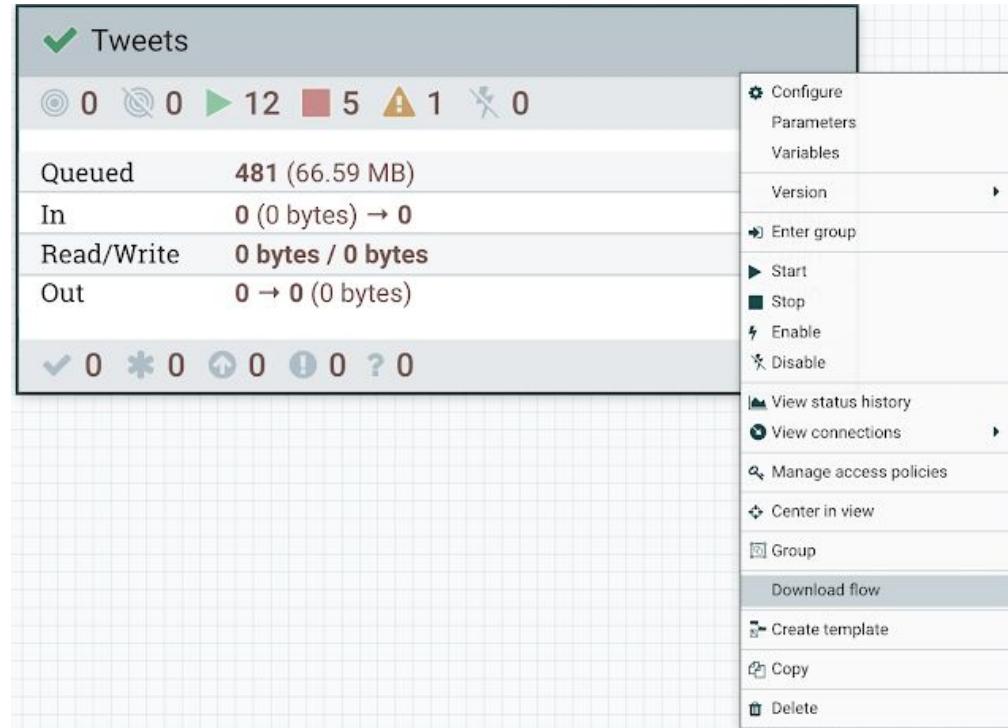
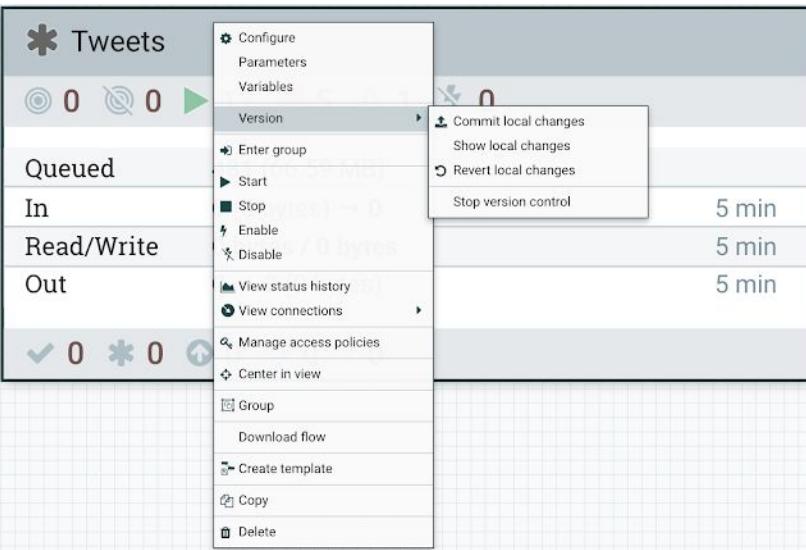
At the bottom right are two buttons: "DOWNLOAD" and "VIEW".



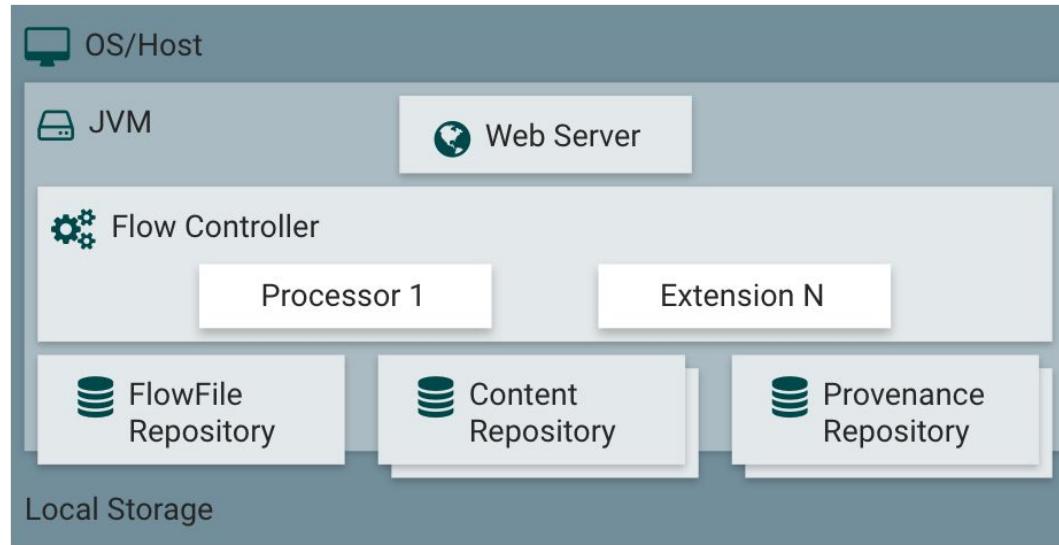
A screenshot of a "FlowFile" details view. At the top is a dark blue header bar with the word "FlowFile". Below it is a navigation bar with "DETAILS" and "ATTRIBUTES" tabs, where "ATTRIBUTES" is selected. The main content area lists various attributes:

- invokehttp.tx.id
4e97d934-a7f5-463b-ad4a-d74ae6da4325
- link
<https://status.cloudera.com/incidents/967wlx3x5rv7>
- mime-type
application/json
- mime.type
application/json
- path
./
- pubdate
Thu, 09 Jul 2020 00:14:58 +0000
- record.count

Version Control (Github and Beyond)

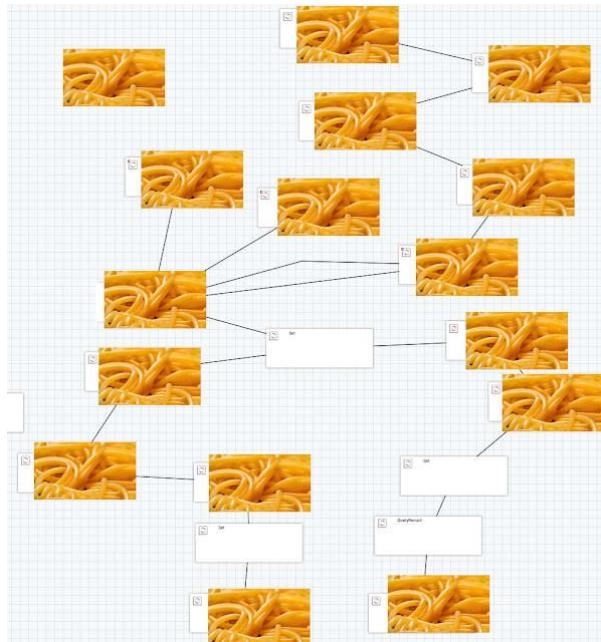


Repositories



<https://nifi.apache.org/docs/nifi-docs/html/nifi-in-depth.html#repositories>

No More Spaghetti Flows - DO NOT

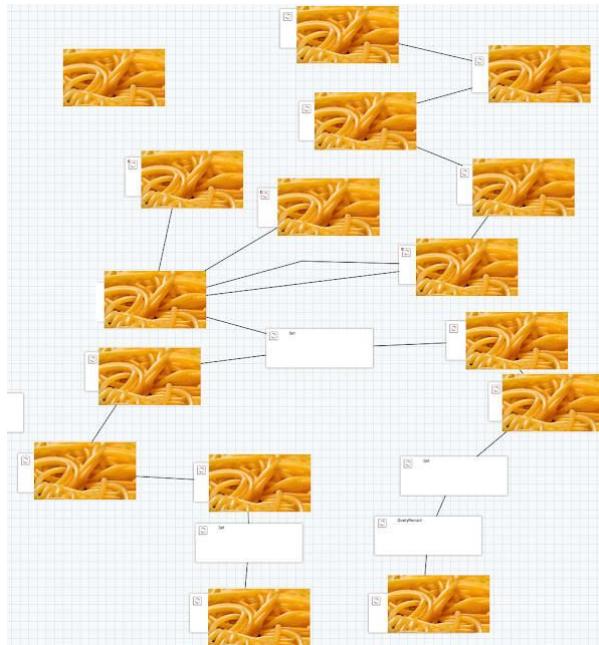


Do Not

- Do not Put 1,000 Flows on one workspace.
- If your flow has hundreds of steps, this is a Flow Smell. Investigate why.
- Do not Use ExecuteProcess, ExecuteScripts or a lot of Groovy scripts as a default, look for existing processors
- Do not Use Random Custom Processors you find that have no documentation or are unknown.
- Do not forget to upgrade, if you are running anything before Apache NiFi 1.14, upgrade now!
- Do not run on default 512M RAM.
- Do not run one node and think you have a highly available cluster.
- Do not split a file with millions of records to individual records in one shot without checking available space/memory and back pressure.
- Use Split processors only as an absolute last resort. Many processors are designed to work on FlowFiles that contain many records or many lines of text. Keeping the FlowFiles together instead of splitting them apart can often yield performance that is improved by 1-2 orders of magnitude.

<https://dev.to/tspannhw/no-more-spaghetti-flows-2emd>

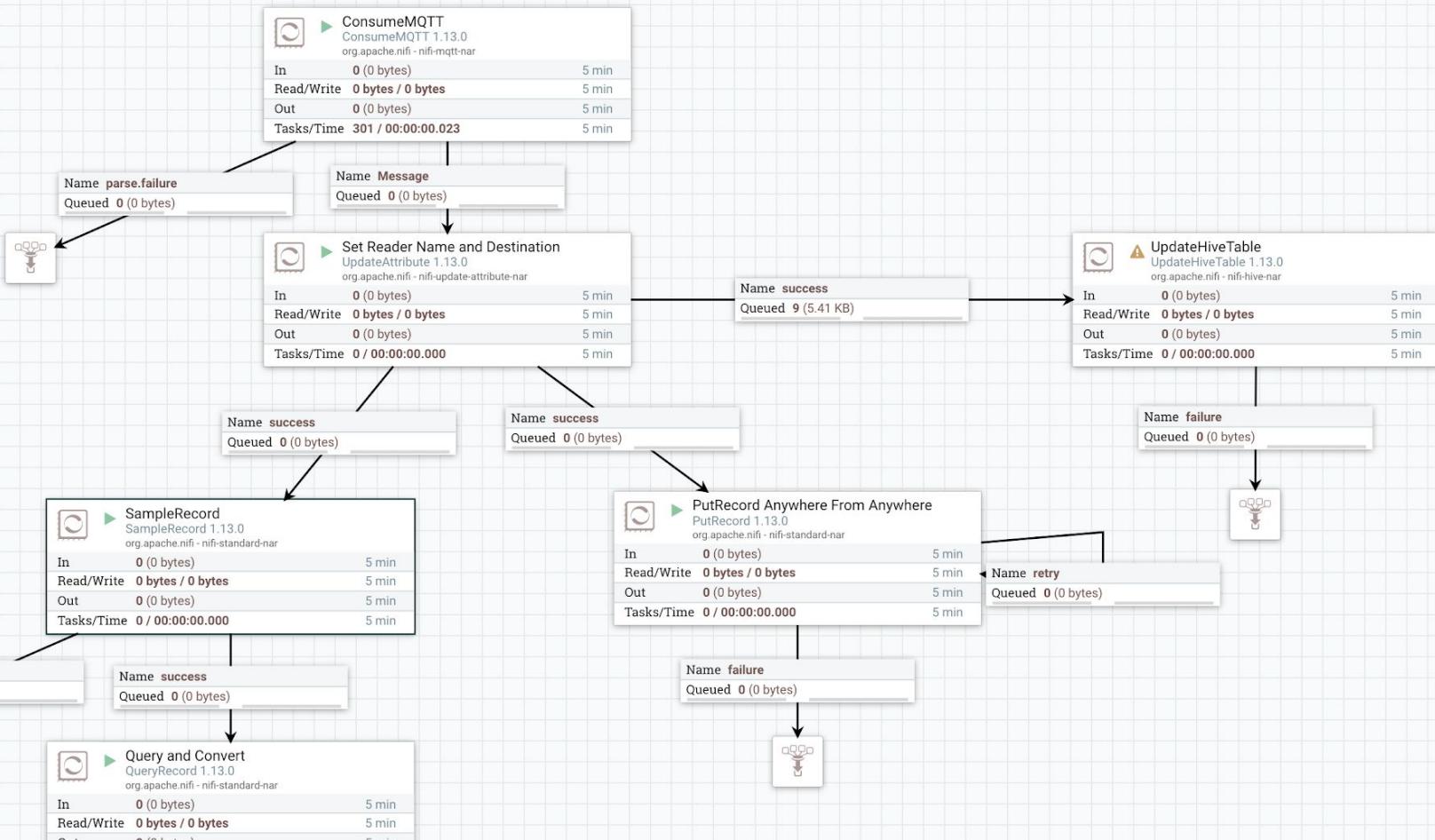
No More Spaghetti Flows - DO



Do

- Reduce, Reuse, Recycle. Use Parameters to reuse common modules.
- Put flows, reusable chunks (write to Slack, Database, Kafka) into separate Process Groups.
- Write custom processors if you need new or specialized features
- Use Record Processors everywhere
- Read the Docs!
- Use the NiFi Registry for version control.
- Use NiFi CLI and DevOps for Migrations.
- Walk through your flow and make sure you understand every step and it's easy to read and follow. Is every processor used? Are there dead ends?
- Do run Zookeeper on different nodes from Apache NiFi.
- Use routing based on content and attributes to allow one flow to handle multiple nearly identical flows is better than deploying the same flow many times with tweaks to parameters in same cluster.
- Use the correct driver for your database. There's usually a couple different JDBC drivers.

<https://dev.to/tspannhw/no-more-spaghetti-flows-2emd>



StreamNative

Apache MXNet Native Processor through DJL.AI for Apache NiFi



#workshop

11:30 AM =====

Deep Learning Class Label: person
File: cc0a469f-c108-42c7-95c6-10e5fda95006.person.png
Probability: 0.96
UUID: 32ef65a3-0650-42cd-965c-ba25597eb1ad
Rank: 1

Bounding Box (Height/Width, X,Y)
0.74 / 0.69
0.27, 0.25

Image (Height/Width, X,Y)
480 / 640
0, 0

=====

tspann 11:30 AM
371bdb8f-35bc-4a2a-919c-bdeb609b726c.person.png

```
private void runModelUserDeploy() {
    testRunner.setWorkflowExpressionUsage(false);
    testRunner.run();
    testRunner.assertValid();
}

testRunner.assertAllFlowFilesTransferred(DeepLearningProcessor.REL_SUCCESS);
List<MockFlowFile> successfuls = testRunner.getFlowFilesForRelationship(DeepLearningProcessor.REL_SUCCESS);

for (MockFlowFile mockFile : successfuls) {
    assertEquals("car", mockFile.getAttribute("category"));
    assertEquals("1.0", mockFile.getAttribute("confidence"));

    System.out.println("MockFile[" + mockFile +
        "]\nMapString, String-attributes = " + mockFile.getAttribute("mapString"));
    for (String attribute : attributes.keySet()) {
        System.out.println("Attribute[" + attribute + "] = " + attributes.get(attribute));
    }
}

@test
public void testProcessor() throws Exception {
    java.io.File resourcesDirectory = new java.io.File(FileUtil.getTestResourcesDirectory().getAbsolutePath());
    System.out.println(resourcesDirectory.getAbsolutePath());
    testRunner.setProperty(DeepLearningProcessor.MACROS, resourcesDirectory.getAbsolutePath());
    testRunner.setProperty(DeepLearningProcessor.DATASET, DeepLearningProcessorTest.testProcessor());
}
```

=====
Tests passed: 1 of 1 (test - 4 s 018 ms)
Size:17632B
Attribute:bounding_box_height_1 = 0.35
Attribute:probability_1 = 1.00
Attribute:bounding_box_x_1 = 0
Attribute:bounding_box_y_1 = 0
Attribute:class_1 = car
Attribute:rank_1 = 1
Attribute:uuid = e9993c52-f5ab-4849-8876-a25796714984
Attribute:bounding_box_width_1 = 0.24

Attribute Values

boundingbox_height_1

0.99

No value set

boundingbox_width_1

0.90

No value set

boundingbox_x_1

0.09

No value set

boundingbox_y_1

0.01

No value set

class_1

tvmonitor

No value set

filename

2020-08-26_1330.jpg.tvmonitor.png

2020-08-26_1330.jpg (previous)

This processor uses the DJL.AI Java Interface

<https://github.com/tspannhw/nifi-djl-processor>

<https://dev.to/tspannhw/easy-deep-learning-in-apache-nifi-with-djl-2d79>

Use Cases

USE CASE

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



REST and Websocket JSON “stonks”



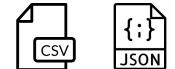
I Can Haz Data?

```
{"symbol":"MSFT",
"uuid":"10640832-f139-4b82-8780-e3ad37b3d0
ce",
"ts":1618529574078,
"dt":1612098900000,
"datetime":"2021/01/31 08:15:00",
"open":122.24500,
"close":122.25500,
"high":122.25500,
"volume":12353,
"low":12.24500}
```

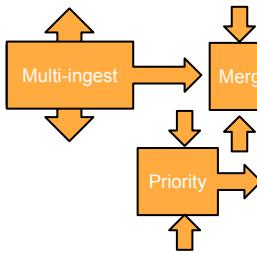


Best Practice Architectures

All Data - Anytime - Anywhere - Multi-Cloud - Multi-Protocol

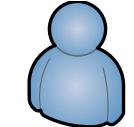


APACHE
nifi



Stream
Native

PULSAR



Data Analyst

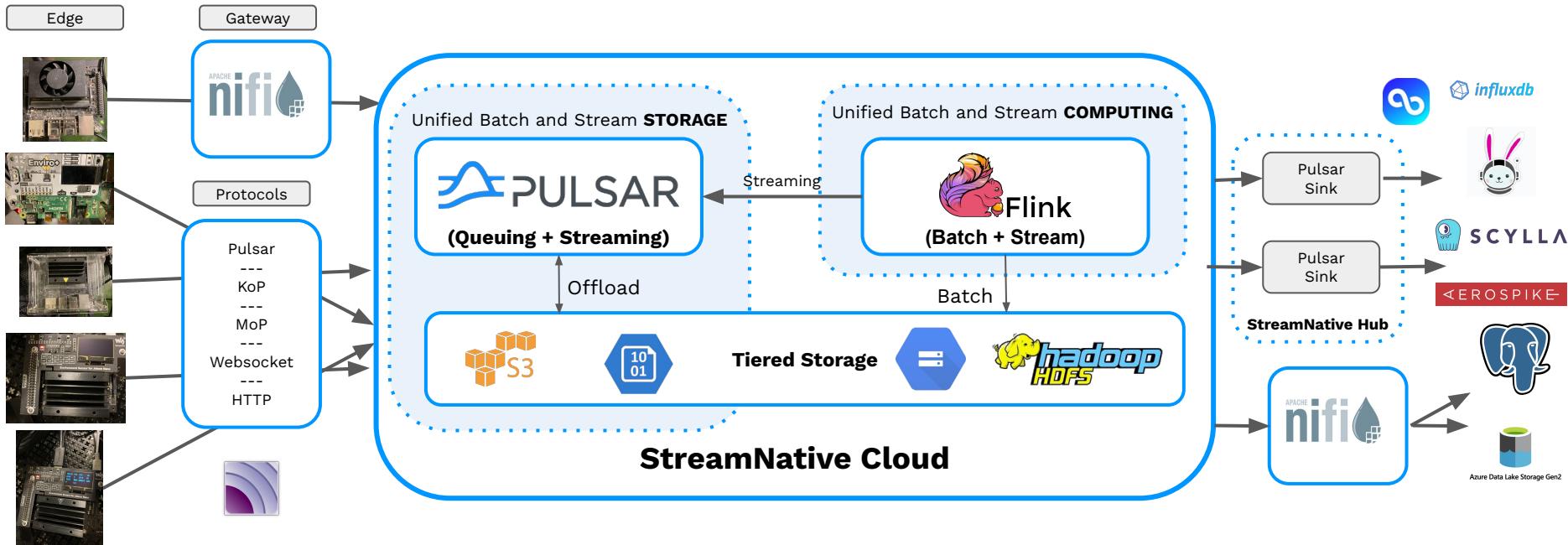
Data Scientist



StreamNative

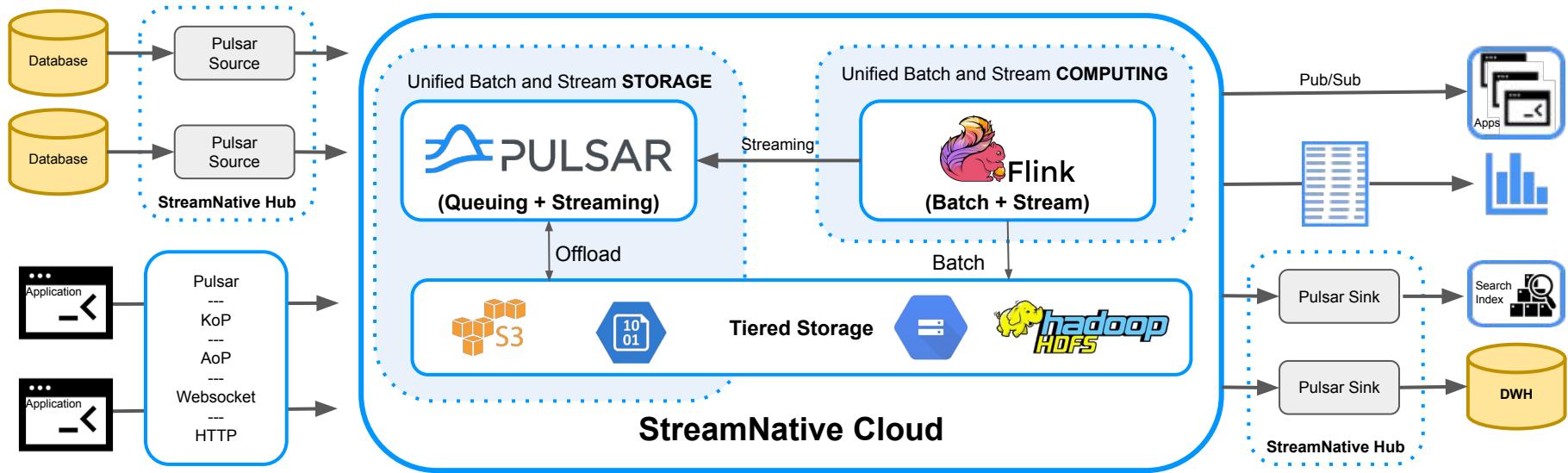
End-to-End Streaming FLiP(N) Apps

Apache Flink - Apache Pulsar - Apache NiFi <-> Events <-> Azure Data Stores



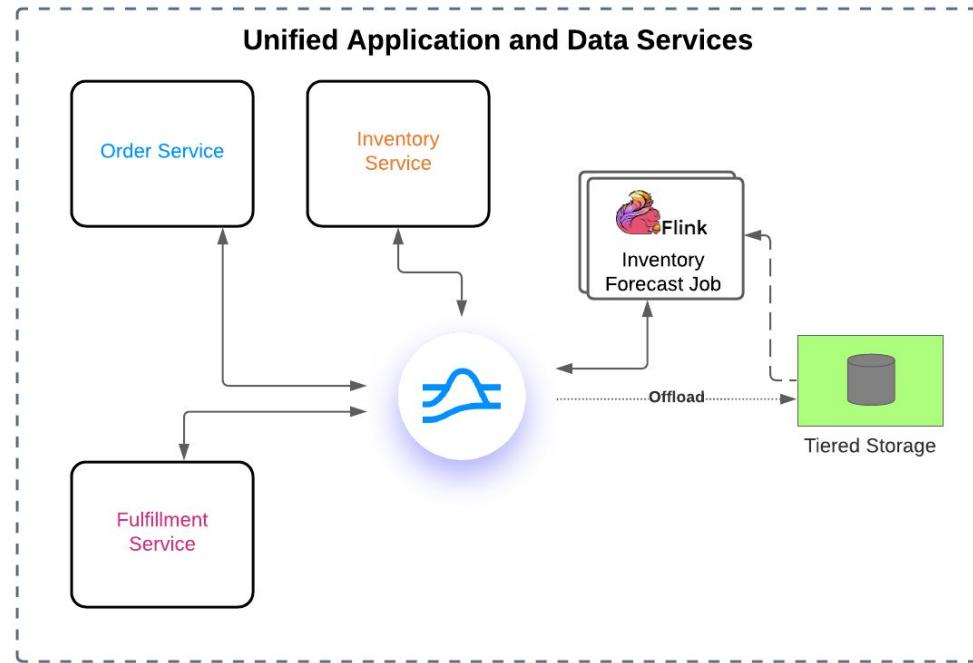
End-to-End Streaming Applications with Flink

With Pulsar and Flink, StreamNative offers both **stream storage** and **stream compute** for a complete streaming solution.

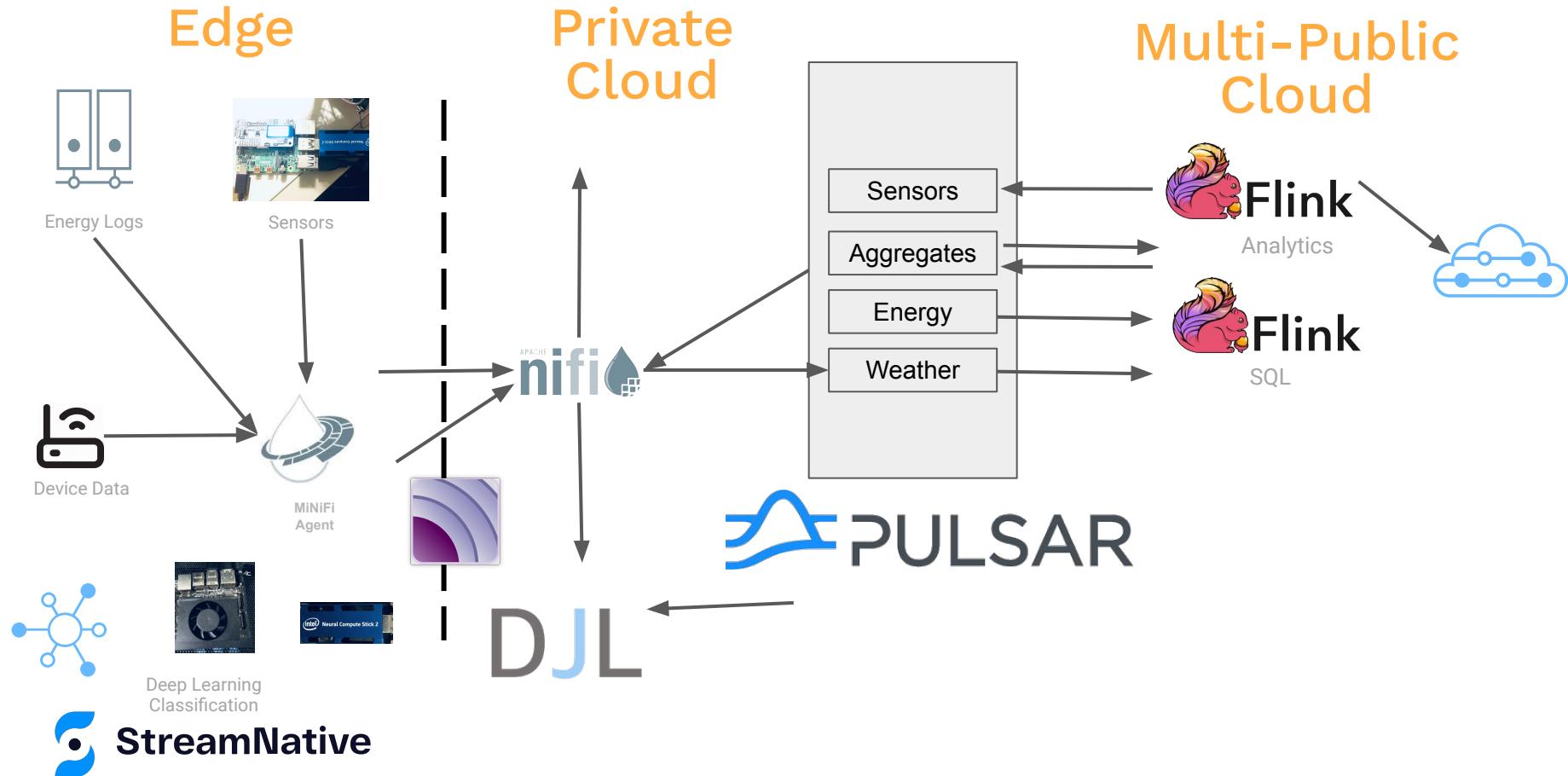


Example: E-Commerce with Pulsar

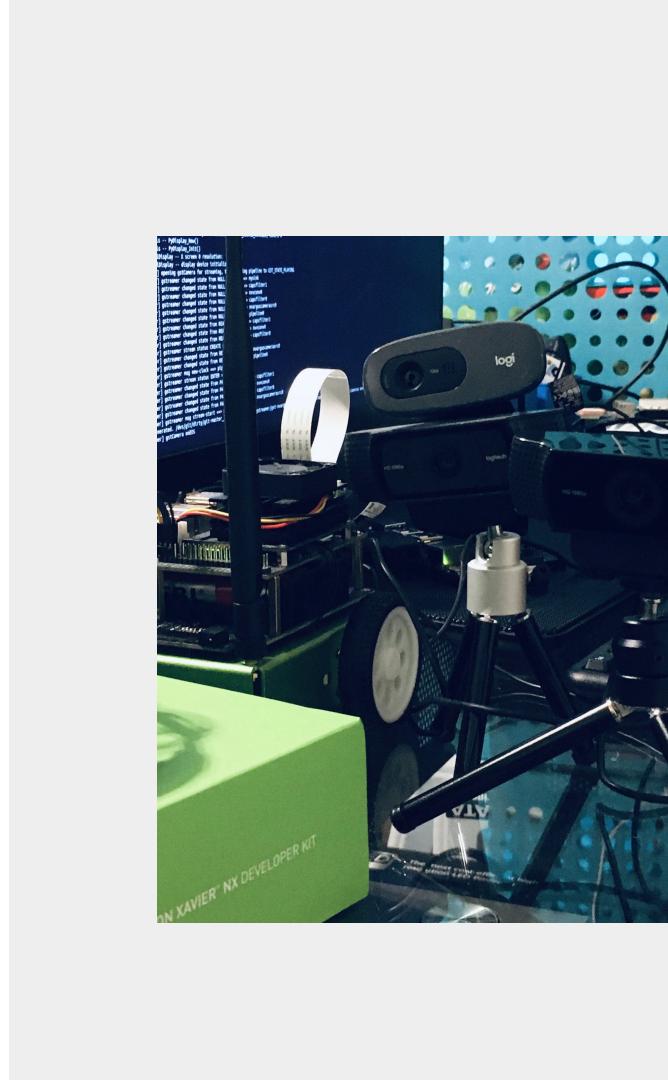
- Unified storage with access to underlying data
- Native tiered storage
- Single system to exchange data
- Teams share toolset



Edge AI to Cloud Streaming Pipeline



Demo



Demo Walkthrough

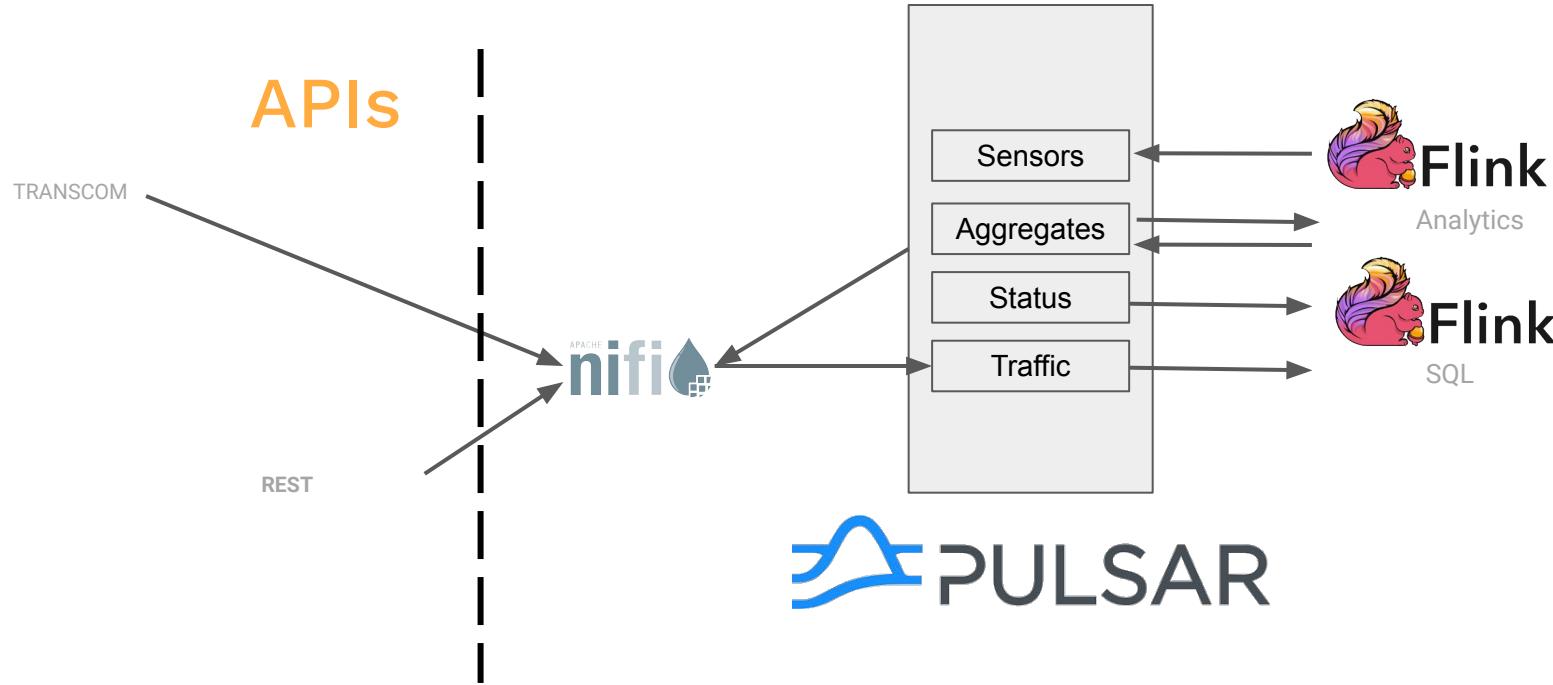
```
Flink SQL> use catalog pulsarcatalog;
[INFO] Execute statement succeed.

Flink SQL> show tables;
+-----+-----+
| table name | 
+-----+-----+
| # | 
| click_events | 
| hello_world | 
| kafka-1 | 
| kafka-2 | 
| kafka-3 | 
| kafka-4 | 
| kafka-5 | 
| mqtt-1 | 
| mqtt-2 | 
| mqtt-3 | 
| mqtt-4 | 
| mqtt-5 | 
| mqtt-go | 
| mqtt-mac | 
| mqtt-nifi | 
| mqtt-nvidia | 
| mqtt-python | 
| mqtt-rp4 | 
| my-topic | 
| nvidia-kafka-1 | 
| rp4-kafka-1 | 
| rwar | 
+-----+-----+
23 rows in set

Flink SQL>
```

```
{"entriesAddedCounter":1,"numberOfEntries":1,"totalSize":651,"currentLedgerEntries":1,"currentLedgerSize":651,"lastLedgerCreatedTimestamp":"2021-09-13T16:13:06.6-04:00","waitingCursorsCount":0,"pendingAddEntriesCount":0,"lastConfirmedEntry":"7076:0","state":"LedgerOpened","ledgers":[{"ledgerId":7076,"entries":0,"size":0,"offloaded":false,"underReplicated":false}],"cursors":{},"schemaLedgers":[]}, "compactedLedger":{"ledgerId":-1,"entries":-1,"size":-1,"offloaded":false,"underReplicated":false}}
```

Real-Time Cloud Streaming Pipeline



Wrap-Up

Connect with the Community & Stay Up-To-Date

- Join the Pulsar Slack channel - Apache-Pulsar.slack.com
- Follow [@streamnativeio](https://twitter.com/streamnativeio) and [@apache_pulsar](https://twitter.com/apache_pulsar) on Twitter
- [Subscribe](#) to Monthly Pulsar Newsletter for major news, events, project updates, and resources in the Pulsar community

Deeper Content

- <https://www.datainmotion.dev/2020/04/building-search-indexes-with-apache.html>
- <https://github.com/tspannhw/nifi-solr-example>
- <https://github.com/streamnative/pulsar-flink>
- <https://www.linkedin.com/pulse/2021-schedule-tim-spann/>
- https://github.com/tspannhw/SpeakerProfile/blob/main/2021/talks/20210729_HailHydrate!FromStreamtoLake_TimSpann.pdf
- <https://streamnative.io/en/blog/release/2021-04-20-flink-sql-on-streamnative-cloud>
- <https://docs.streamnative.io/cloud/stable/compute/flink-sql>



@PaasDev timothyspann

<https://www.pulsardeveloper.com/>

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

[\[10/21\] Trino Summit](#)



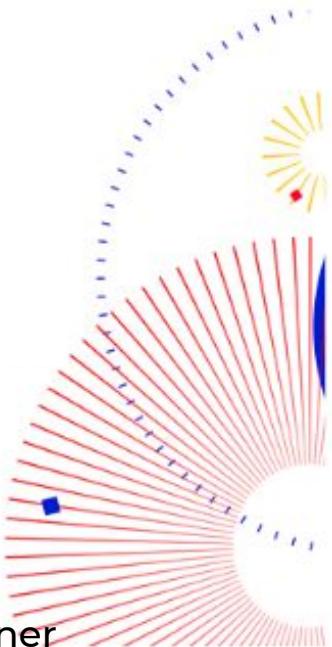
StreamNative



Pulsar Summit Asia

November 20-21, 2021

Contact us at partners@pulsar-summit.org to become a sponsor or partner





trino SUMMIT

10:45 - 11:25am EST
7:45 - 8:25am PST
3:45 - 4:25pm BST
8:15 - 8:55pm IST

FLiP Into Trino



Remember the days when you could wait until your batch data load was done and then you could run some simple queries or build stale dashboards? Those days are over, today you need instant analytics as the data is streaming in real-time. You need universal analytics where that data is. I will show you how to do this utilizing the latest cloud native open source tools. In this talk we will utilize Trino, Apache Pulsar, Pulsar SQL and Apache Flink to analyze instantly data from IoT, sensors, transportation systems, Logs, REST endpoints, 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 how to use Pulsar SQL to run analytics on live data.



Tim Spann

Developer Advocate
StreamNative



David Kjerrumgaard

Developer Advocate
StreamNative



StreamNative

Wednesday, October 27, 2021

9:00 AM - 9:25 AM PDT

AI DEVWORLD

AI OPEN TALKS

AI FOR THE ENTERPRISE

Add

OPEN TALK (AI): Utilizing Apache  Pulsar, Apache NiFi and MiNiFi for EdgeAI IoT at Scale



Timothy Spann

StreamNative, Developer Advocate

Thursday, October 28, 2021

9:00 AM - 9:25 AM PDT

API INNOVATION

EMERGING APIs

Add

PRO TALK (API): Apache NiFi 101: Introduction and Best Practices



Timothy Spann

StreamNative, Developer Advocate

Let's Keep in Touch!



Tim Spann

Developer Advocate



@PassDev



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



<https://github.com/tspannhw>

Questions