Hortonworks Streaming Analytics Manager 3

Getting Started with Streaming Analytics

Date of Publish: 2018-12-18



https://docs.hortonworks.com/

Contents

Building a	an End-to-End Stream Application	
Unders	tanding the Use Case	
Referen	ice Architecture	5
Prepare Y	our Environment	5
Deploy	ing Your Cluster	6
Registe	ring Schemas in Schema Registry	6
-	Create the Kafka Topics	6
	Register Schemas for the Kafka Topics	6
Setting	up an Enrichment Store, Creating an HBase Table, and Creating an HDFS Directory	7
Creating	a Dataflow Application	8
	a Dutation Application Generates Events	o
	Trate a Deteflow Application	
NIFI. C	NEE Controller Services	10
	NET Ingests the Daw Sensor Events	10
	NIFI Ingests the Kaw Sensor Events.	11
	Fublish Enficience Events to Karka for Consumption by Analytics Applications	12
Pick your	Streaming Engine	14
Pick your	Streaming Engine	14
Pick your	Streaming Engine	14
Pick your Creating a	Streaming Engine	14
Pick your Creating a Creatin	Streaming Engine a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM	14 16
Pick your Creating a Creatin	Streaming Engine	14
Pick your Creating a Creatin	Streaming Engine a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment	14 16
Pick your Creating a Creatin	Streaming Engine a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application	14 16
Pick your Creating a Creatin	Streaming Engine a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream	14 16 16 16 17
Pick your Creating a Creatin	Streaming Engine a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components	14 16 16 16 16 17
Pick your Creating a Creatin	Streaming Engine a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components Joining Multiple Streams	14 16 16 16 17 18 19 20 21
Pick your Creating : Creatin	Streaming Engine a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components Joining Multiple Streams Filtering Events in a Stream using Rules	14 16 16 16 16 17 18 20 21 23
Pick your Creating a Creatin	Streaming Engine a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components Joining Multiple Streams	14 16 16 16 17 18 19 20 21 23 24
Pick your Creating a Creatin	Streaming Engine a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications. Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components Joining Multiple Streams Filtering Events in a Stream using Rules Using Aggregate Functions over Windows. Implementing Business Rules on the Stream	14 16 16 16 17 18 19 20 21 23 24 24
Pick your Creating a Creatin	Streaming Engine a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components Joining Multiple Streams Filtering Events in a Stream using Rules Using Aggregate Functions over Windows Implementing Business Rules on the Stream Transforming Data using a Projection Processor	14 16 16 16 16 17 18 19 20 21 23 24 24 24 26
Pick your Creating a Creatin	Streaming Engine a Streaming Analytics Application with SAM	14 16 16 16
Pick your Creating a Creatin	Streaming Engine a Streaming Analytics Application with SAM	14 16 16 16 16 17 18 19 20 21 21 23 24 24 24 24 24 24 24
Pick your Creating a Creatin	Streaming Engine a Streaming Analytics Application with SAM	14 16 16 16 16 17 18 19 20 21 23 24 24 24 24 26 29 30 32
Pick your Creating a Creatin	Streaming Engine	14 16 16 16 17 18 19 20 21 23 24 24 24 24 26 29 30 32 35
Pick your Creating a Creatin	Streaming Engine	14 16 16 16 17 18 19 20 21 23 24 24 26 30 32 35 35
Pick your Creating a Creatin	Streaming Engine	14 16 16 16 16 17 18
Pick your Creating a Creatin	Streaming Engine	14 16 16 16 16 17 18
Pick your Creating a Creatin	Streaming Engine	14 16 16 16 16 17 18
Pick your Creating a Creatin Deploy Advance	Streaming Engine	14 16 16 16 17 18 19 20 21 23 24 24 24 24 24 24 24 30 30 35 35 36 38 36 38 36 38 36 38 39 30 39 30 39 30
Pick your Creating a Creatin Deploy Advance	Streaming Engine	14 16 16 16 17 18 19 20 21 23 24 24 24 24 24 24 24 24 30 30 32 35 36 38 38 39 40 41
Pick your Creating a Creatin Deploy Advanc	Streaming Engine	14

Streaming Split Join Pattern	51
Score the Model Using the PMML Processor and Alert	
Creating Visualizations Using Superset	61
Creating Insight Slices	61
Adding Insight Slices to a Dashboard	
SAM Test Mode	
Four Test Cases using SAM's Test Mode	
Creating Custom Sources and Sinks	
Cloud Use Case: Integration with AWS Kinesis and S3	
Registering a Custom Source in SAM for AWS Kinesis	
Registering a Custom Sink in SAM for AWS S3	
Implementing the SAM App with Kinesis Source and S3 Sink	
Stream Operations	
My Applications View	
Application Performance Monitoring	
Exporting and Importing Stream Applications	89
Troubleshooting and Debugging a Stream Application	91
Monitoring SAM Apps and Identifying Performance Issues	91
Identifying Processor Performance Bottlenecks	
Debugging an Application through Distributed Log Search	
Debugging an Application through Sampling	
Spark Streaming	
Running the Stream Simulator	103
Managing Kafka with Streams Messaging Manager	
SMM Overview	
Installing DataPlane Streams Messaging Manager	
Enabling Reference Application Cluster for SMM	105
Monitoring Katka with SMM	107

Building an End-to-End Stream Application

A good way to get started using Hortonworks DataFlow (HDF) and all of its services like NiFi, Streams Messaging Manager, Streaming Analytics Manager, Schema Registry, etc is to imagine a real life use case, and to learn about the common HDF stream processing tasks and concepts through this use case. This guide sets up a fictional use case, and walks you through the deployment and common tasks you would perform while engaging in many of HDF's stream processing use cases.

Use this guide as a tutorial to get you started with NiFi, SMM, SAM and Schema Registry and Spark Streaming. All the resources required to complete the tasks are provided in line.

Understanding the Use Case

To build a complex streaming analytics application from scratch, we will work with a fictional use case. A trucking company has a large fleet of trucks, and wants to perform real-time analytics on the sensor data from the trucks, and to monitor them in real time. Their analytics application has the following requirements:

Procedure

- 1. Outfit each truck with two sensors that emit event data such as timestamp, driver ID, truck ID, route, geographic location, and event type.
 - The geo event sensor emits geographic information (latitude and longitude coordinates) and events such as excessive braking or speeding.
 - The speed sensor emits the speed of the vehicle.
- 2. Stream the sensor events to an IoT gateway. The data producing app (e.g: a truck) will send CSV events from each sensor to one of three gateway topics (gateway-west-raw-sensors, gateway-east-raw-sensors or gateway-central-raw-sensors). Each event will pass the schema name for the event as a Kafka event header.
- **3.** Use NiFi to consume the events from the Kafka topic, and then route, transform, enrich, and deliver the data from the gateways to two syndication topics (e.g: syndicate-geo-event-avro, syndicate-speed-event-avro, syndicate-geo-event-avro, syndicate-speed-event-json, syndicate-speed-event-json) that various downstream analytics applications can subscribe to.
- 4. Connect to the two streams of data to perform analytics on the stream.
- **5.** Join the two sensor streams using attributes in real-time. For example, join the geo-location stream of a truck with the speed stream of a driver.
- 6. Filter the stream on only events that are infractions or violations.
- 7. All infraction events need to be available for descriptive analytics (dash-boarding, visualizations, or similar) by a business analyst. The analyst needs the ability to perform analysis on the streaming data.
- **8.** Detect complex patterns in real-time. For example, over a three-minute period, detect if the average speed of a driver is more than 80 miles per hour on routes known to be dangerous.
- 9. When each of the preceding rules fires, create alerts, and make them instantly accessible.
- **10.** Execute a logistical regression Spark ML model on the events in the stream to predict if a driver is going to commit a violation. If violation is predicted, then generate an alert.
- 11. Monitor and manage the entire application using Streams Messaging Manager and Stream Operations.

What to do next

The following sections walk you through how to implement all ten requirements. Requirements 1-3 are performed using NiFi and Schema Registry. Requirements 4 through 10 are implemented using the new Streaming Analytics Manager.

Reference Architecture

This reference architecture diagram gives you a general idea of how to build an HDF cluster for your trucking use case. Review this suggested architecture before you begin deployment.



Prepare Your Environment

Deploying Your Cluster

Now that you have reviewed the reference architecture and planned the deployment of your trucking application, you can begin installing HDF according to your use case specifications. To fully build the trucking application as described in this *Getting Started with Stream Analytics* document, use the following steps.

Procedure

- 1. Install Ambari 2.7.0.
- 2. Install HDP 3.0.0.
- 3. Install the HDF 3.2.0 Management Pack onto the HDP cluster.

What to do next

You can find more information about your HDF and HDP cluster deployment options in Planning Your Deployment.

You can find more information about which versions of HDP and Ambari you should use with your version of HDF in the *HDF Support Matrices*.

Registering Schemas in Schema Registry

The trucking application streams CSV events from the truck sensors to a IOT gateway powered by a set of Kafka gateway topics. NiFi consumes the events from these topics, and then routes, enriches, and delivers them to a set of syndication Kafka topics.

To do this, you must perform the following tasks:

- Create the Kafka gateway and syndication topics
- Register a set of Schemas in Schema Registry

Create the Kafka Topics

Kafka topics are categories or feed names to which records are published.

Procedure

- 1. Log into the node where Kafka broker is running.
- 2. Download script createKafkaTopics.sh from here: https://raw.githubusercontent.com/georgevetticaden/sam-trucking-data-utils/hdf-3-2/src/main/resources/scripts/smm/createKafkaTopics.sh
- **3.** Execute the script by passing as arg ZK:Port

For example:

./createKafkaTopics.sh ZK_HOST:2181

Register Schemas for the Kafka Topics

Register schemas for the IOT gateway and syndication Kafka topics .Registering the Kafka topic schema is beneficial in several ways. Schema Registry provides a centralized schema location, allowing you to stream records into topics without having to attach the schema to each record.

Procedure

- 1. Log into the node where you have access to the Schema Registry Server.
- 2. Download the Data-Loader and unzip the contents.

3. Navigate to the Data-Loader directory:

cd Data-Loader

4. Execute the following:

```
java -cp \
stream-simulator-jar-with-dependencies.jar \
hortonworks.hdf.sam.refapp.trucking.simulator.schemaregistry.TruckSchemaRegistryLoade
\
$SCHEMA_REGISTRY_ENDPOINT_URL
E.g: SCHEMA_REGISTRY_ENDPOINT_URL = http://SR_HOST::7788/api/v1
```

Setting up an Enrichment Store, Creating an HBase Table, and Creating an HDFS Directory

To prepare to perform predictive analytics on streams, you need some HBase and Phoenix tables. This section gives you instructions on setting up the HBase and Phoenix tables timesheet and drivers, loading them with reference data, and downloading the custom UDFs and processors to perform the enrichment and normalization.

Install HBase/Phoenix and download the sam-extensions

- 1. If HBase is not installed, install/add an HBase service.
- 2. Ensure that Phoenix is enabled on the HBase Cluster.
- 3. Download the Sam-Custom-Extensions.zip and save it to your local machine.
- 4. Unzip the contents. Name the unzipped folder \$SAM_EXTENSIONS.

Steps for Creating Phoenix Tables and Loading Reference Data

- 1. Copy the \$SAM_EXTENSIONS/custom-processor/scripts.tar.gz to a node where HBase/Phoenix client is installed.
- 2. On that node, untar the scripts.tar.gz. Name the directory \$SCRIPTS.

tar -zxvf scripts.tar.gz

3. Navigate to the directory where the phoenix script is located which will create the phoenix tables for enrichment and load it with reference data.

cd \$SCRIPTS/phoenix

- 4. Open the file phoenix_create.sh and replace <ZK_HOST> with the FQDN of your ZooKeeper host.
- 5. Make the phoenix_create.sh script executable and execute it. Make sure you add the script to JAVA_HOME.

./phoenix_create.sh

Steps for Verifying Data has Populated Phoenix Tables

1. Start up the sqlline Phoenix client.

cd /usr/hdp/current/phoenix-client/bin
./sqlline.py \$ZK_HOST:2181:/hbase-unsecure

2. List all the tables in Phoenix.

!tables

3. Query the drivers and timesheet tables.

select * from drivers;

select * from timesheet;

Steps for Starting HBase and Creating an HBase Table

- 1. This can be easily done by adding the HDP HBase Service using Ambari.
- 2. Create a new HBase table by logging into an node where Hbase client is installed then execute the following commands:

```
cd /usr/hdp/current/hbase-client/bin
/hbase shell
create 'violation_events', {NAME=> 'events', VERSIONS => 3};
```

Steps for Creating an HDFS Directory

Create the following directory in HDFS and give it access to all users.

- 1. Log into a node where HDFS client is installed.
- 2. Execute the following commands:

```
su hdfs
hadoop fs -mkdir /apps/trucking-app
hadoop fs -chmod 777 /apps/trucking-app
```

Creating a Dataflow Application

Data Producer Application Generates Events

The following is a sample of a raw truck event stream generated by the sensors.

Geo Sensor Stream



Speed Sensor Stream



The date producing application or data simulator publishes CSV events with schema name in the Kafka event header (leveraging the Kafka Header feature in Kafka 1.0). The following diagram illustrates this:



Use NiFi's Kafka 1.X ConsumeKafkaRecord and PublishKafkaRecord processors using record-based processing to do the following:

- 1. Grab the schema name from Kafka Header and store in flow attribute called schema.name
- 2. Use the schema name to look up the Schema in HWX SR
- 3. Use the schema from HWX SR to convert to ProcessRecords
- 4. Route events by event source (speed or geo) using SQL
- 5. Filter, Enrich, Transform

6. Deliver the data to downstream syndication topics

The below diagram illustrates the entire flow and the subsequent sections walks through how to setup this data flow.



NiFi: Create a Dataflow Application

To make things easier to setup, import the NiFi Template for this flow by downloading it to this Github location. After importing, select IOT Trucking Fleet - Data Flow process group. The following instructions are with respect to that flow.

NiFi Controller Services

Procedure

- 1. Click on Flow Configuration Settings icon, select Controller Services tab, and select Hortonworks Schema Registry Controller Service.
- 2. Click on Flow Configuration Settings icon and select Controller Services tab.

You will see the HWX Schema Registry controller service. Edit the properties to configure the Schema Registry URL based on your environment. You can find this value in the Streaming Analytics Manager Service in Ambari for the configuration property called registry.url. An example of what the URL looks similar to http:// \$REGISTRY_SERVER:7788/api/v1.

3. Edit the properties to configure the Schema Registry URL based on your environment.

You can find this value in the Streaming Analytics Manager Service in Ambari for the configuration property called registry.url. The URL should look similar to the following: http://\$REGISTRY_SERVER:7788/api/v1.

4. Enable the HWX Schema Registry and the other controller services. You should have 5 controller services enabled like the following.

	Name	Туре	Bun
8	HWX Schema Registry	HortonworksSchema	org
8	Enrich-ReverseGeoCodeLookupService	ScriptedLookupServic	orç
8	CSV Writer - HWX SR Registry	CSVRecordSetWriter 1	orç
8	CSV Reader - HWX SR Registry	CSVReader 1.5.0.3.1.0	orç
8	Avro Writer - HWX SR Registry - HWX Content Enc	AvroRecordSetWriter	orç

NiFi Ingests the Raw Sensor Events

In the IOT Trucking Fleet - Data Flow process group, go into the "Acquire Events from Kafka IOT Gateways" process group. The first step in the NiFi flow is to ingest the csv events from the three IOT gateway topics. We will use the new Kafka 1.0 ConsumerKafkaRecord processor for this.

Upstream app is sen Use ConsumeKafkaf 1. Grab the schema n 2. Use the schema n 3. Use the schema fi 4. Route events by e	Record do the name from Ka ame to look u om HWX SR f vent source (s	followi afka He ip the S to conv speed c	n schema name ng ader and store in chema in HWX S ert to ProcessRe or geo) using SQ	in Kafka Messag n Attribute called SR ecords L	ge Hea d schei
US West Truck Fleet - Kafk ConsumeKafkaRecord_1_0 1.7 org.apache.nifi - nifi-kafka-1-0-nar	a Gateway 0.3.2.0.0-485		US Central Truck Fleet ConsumeKafkaRecord_1_ org.apache.nifi - nifi-kafka-1-0-n	- Kafka Gatew 0 1.7.0.3.2.0.0-485	0
In 0 (0 bytes)	5 min	In	0 (0 bytes)	5 min	In
Read/Write 0 bytes / 40.35 KB	5 min	Read/Wi	ite 0 bytes / 27.82 KB	5 min	Read
Out 326 (40.35 KB) Tasks/Time 1,052 / 00:00:12.521	5 min 5 min	Out Tasks/Ti	230 (27.82 KB) me 1,000 / 00:00:11.451	5 min 5 min	Out Task
	Name succe	955	success	Name success	
	Queued 0 (0	bytes)	d 0 (0 bytes)	Queued 0 (0 bytes)	
			Truck Fleet Sensor Streams		

Configure the 'Kafka Brokers' value for each of the three ConsumeKafkaRecord_1_0 processors based on your cluster.

Publish Enriched Events to Kafka for Consumption by Analytics Applications

After NiFi completes the routing, transforms, and enrichment, NiFi publishes the enriched events into a set of syndication Kafka topics. This is done in the Publish Enriched Streams Kafka Syndication Services process group. This process group uses the PublishKafkaRecord processor to publish the events into 4 syndication topics.



Ensure that for the PublishKafkaRecord, you change the Kafka Brokers property value to your cluster settings.

Start the NiFi Flow

Start the Process Group called IOT Trucking Fleet - Data Flow.

Pick your Streaming Engine

Hortonworks supports a number of powerful Streaming Engines including:

- Spark Streaming (Currently Supported in HDP)
- Streaming Analytics Manager (SAM) using Apache Storm (Currently Supported in HDF)
- Apache Kafka Streams (Will be supported in future HDP/HDF release)

Hortonworks provides you the flexibility to pick the streaming engine of your choice to build streaming analytics application.



If your organization has not standardized on a streaming engine and is looking for guidance on choosing an engine, use the below table to help guide your selection.

Requirement	Streaming Engine to Use
You want to build streaming applications with as little code as possible and want to use ETL like Tooling with a drag and drop paradigm to build streaming apps	Streaming Analytics Manager (SAM) with Storm
You want the ability to build an app that you can deploy in batch and/ or streaming mode	Spark Streaming
You plan to develop streaming applications using scala and java and want a clean easy to use API	Spark Streaming
You want the ability to execute SQL against the stream	Spark Streaming
You want process event one at a time (no microbatching)	Streaming Analytics Manager (SAM) with Storm

The below sections walk you through implementing the streaming analytics requirements with these different tools/ engines.

Creating a Streaming Analytics Application with SAM

Creating a Stream Analytics Application with SAM

Two Options for Creating the Streaming Analytics Applications

Over the next few sections, we walk through building up the stream analytics app to implement all the requirements of this use case. This will entail actions such as:

- Uploading Custom UDFs
- Uploading Custom Sources
- Uploading Custom Sinks
- Uploading a PMML model into Model Registry
- Uploading Custom Processors
- Creating Service Pools for HDP and HDF
- Create SAM Environment required for the Reference App
- Building the Reference application with streaming joins, filtering, aggregations over windows, dashboarding, execute PMML models, doing streaming split pattern, etc..
- Setting up Test Cases for the Reference App

There are two options to performing these actions:

- 1. Doing all of these steps manually as the subsequent sections will walk you through. This is recommended if you are new to SAM and want to build a complex app from scratch.
- 2. Running a utility that performs all of these actions for you.

Setting up the Stream Analytics App using the TruckingRefAppEnvEnviornmentBuilder

Follow the below instructions if you want to set up the reference application using an utility. If not, skip this section and go through the next set of sections. Perform the below steps on the host where SAM is running.

Procedure

1. Download the SAM_EXTENSIONS zip file. Unzip the contents. Call the unzipped folder \$SAM_EXTENSION

2. Navigate to the unzip location:

cd \$SAM_EXTENSION/ref-app

- 3. Modify the trucking-advanced-ref-app.properties file based on your environment:
 - sam.rest.url = the REST url of the SAM instance in your env (e.g.: http://<SAM_HOST>:<SAM_PORT>/api/ v1)
 - sam.service.pool.hdp.ambari.url = The rest endpoint for the HDP/HDF cluster you installed (e.g.: http:/ <HDP_AMBARI_HOST>:<PORT>/api/v1/clusters/<cluster_name>)
 - sam.service.pool.hdp.ambari.username = the username to log into Ambari
 - sam.service.pool.hdp.ambari.passwd = the password to log into Ambari
 - sam.schema.registry.url = The url of the Schema Registry service in SAM you installed as part of the HDF cluster (e.g.: http://SR_HOST:SR_PORT/api/v1)
- **4.** Run the following command:

```
java -cp sam-trucking-ref-app-shaded.jar
hortonworks.hdf.sam.refapp.trucking.env.SMMTruckingRefAppEnviornmentBuilderImpl
trucking-advanced-ref-app.properties
```

Results

If script ran successfully, you should see output like the following (it will take about 3-5 minutes to finish):

```
Trucking Ref App environment creation time[367 seconds]
Trucking Ref App SAM URL: http://SAM_HOST:SAM_PORT/#/applications/78/view
```

Configuring and Deploying the Reference Application

In SAM, go into the edit mode the Trucking Ref App using the url outputted in the logs and then follow the below steps to configure and deploy the reference application:

Procedure

- 1. Double click on the TruckGeoEvent Kafka Source and configure it.
 - a. Select a cluster.
 - **b.** Select truck_events_avro for the kafka topic.
 - c. Select 1 for the Reader Schema Version.
 - **d.** Put an unique value for the consumer group id.
- 2. Double click on the TruckSpeedEvent Kafka Source and configure it.
 - **a.** Select a cluster.
 - **b.** Select truck_speed_events_avro for the kafka topic.
 - c. Select 1 for the Reader Schema Version.
 - **d.** Put an unique value for the consumer group id.
- **3.** Open the configuration for the two Druid sinks (Alerts-Speeding-Driver-Cube and Dashboard-Predictions) and configure each one by selecting a cluster.
- **4.** Open the configuration for the ENRICH-HR and ENRICH-Timesheet configure the 'Phoenix ZooKeeper Connection Url" based on your cluster (e.g.: ZK_HOST:ZK:PORT).
- 5. Open the configuration for the Hbase and HDFS sink and ensure that the cluster is selected.

Creating a Service Pool and Environment

Before you create an application, you have to create a Service Pool and then an Environment that you associate with an application.

Creating Your First Application

The Streaming Analytics Manager provides capabilities to the application developer for building streaming applications. You can go to the Stream Builder UI by selecting the **Streaming Analytics Manager** service in Ambari and under **Quick Links** select **SAM UI**.

About this task

Creating a new stream application requires two steps: clicking the + icon, and then providing a unique name for the stream application and associating the application with an Environment.

Procedure

- 1. Click the + icon on the My Applications dashboard and choose New Application.
- 2. Specify the name of the stream application and the environment that you want it to use to stream.



Note:

The name of the stream application cannot have any spaces.

Add Stream	×
NAME •	
Trucking-IOT-Stream-Analytics	
ENVIRONMENT .	
Dev	*
	Cancel Ok

Results

SAM displays the Stream Builder canvas. Builder components on the canvas palette are the building blocks used to build stream applications. Now you are ready to start building the streaming application.

S	My Appli	lications Sample Application Edit and name the stream application				<u>^</u>
	Q 🕜		Last Change:0s ago	Version:CURRENT	DEV	ତ୍ତ 😅
	SOURCE					-
ති	EVENT					
ு	HUBS	Processor, source, and sink palette contains builder components			Applicatio	on and
	1029 HDES				deployn	nent
Ø	& 8				configur	ation
x	KAFKA					
	PROCESSOR					
	Σ					
	BRANCH					
	>+					
	JOIN					
	PROJECTION			_		
	9 <u>0</u> 0		Deploy button to			
	RULE		applications to the	→	Status: NOT RUNNING	1
	<u> </u>		streaming engine			
	ENRICH	, and the second se				

Creating and Configuring the Kafka Source Stream

The first step in building a stream application is to drag builder components to the canvas. As described in the *Hortonworks DataFlow Overview*, Stream Builder offers four types of builder components: sources, processors, sinks, and custom components.

About this task

Every stream application must start with a source.

Complete the following instructions to start building a stream application. Use these steps to implement Requirement 4 of the use case.

Procedure

1. Drag the Kafka builder component onto the canvas, creating a Kafka tile:



- 2. Set the number of run-time instances for your Kafka tile component by clicking the up arrow on the tile.
- **3.** Double-click on the tile to begin configuring Kafka. After you specify a Kafka topic name, SAM communicates with the Schema Registry and displays the schema:

TruckGeoEvent Kafka connection settings are populated by SAM, based on the Kafka service in Environment from the Service Pool					
REQUIRED OPTIONAL NOTES					
streamanalytics_AUTOCREATED -	Output				
SECURITY PROTOCOL *	eventSource* STRING				
PLAINTEXT -	truckId* INTEGER				
	driver1d*				
BOOTSTRAP SERVERS * driverName*					
hdf-3-1-build4.field.hortonworks.com:6667,hdf-3-1-buil					
KAFKA TOPIC *	route*				
truck_events_avro *	eventType*				
READER SCHEMA VERSION *	latitude*				
1	longitude*				
CONSUMER GROUP ID *	correlationId*				
ref-geo-101	geoAddress strang				
After you select a Kafka topic, SAM fetches the topic schema from Schema Registry	Cancel Ok				

Results

Once you have configured your Kafka component correctly, the tile component displays a green dot.

Connecting Components

To pass a stream of events from one component to the next, create a connection between the two components. In addition to defining data flow, connections allow you to pass a schema from one component to another.

Procedure

1. Click the green dot to the right of your source component.



2. Drag your cursor to the component tile to which you want to connect.

Example

The following example shows two connections: a connection from Kafka sink TruckGeoStream to the join processor, and a connection from the Kafka sink TruckSpeedStream to the same join processor.

5	My Applications / IOT-Trucking-Ref-App							
	Q 🕜		Last Change Os ago	Version:CURRENT	• •			
, w	SOURCE	Two Kafka components are	Last onlange.oo ago	VOI BIOIN BOUNDEN	~ ~ ~			
æ	×	Both streams and the respective						
-	EVENT HUBS	schemas are passed to the Join processor.						
Ş	1							
	HDFS	Re TruckCooFy						
Ŷ	*							
	KAFKA							
æ	PROCESSOR	KAFKA TruckSpeed ∢ 01 ▶ •						
	Σ							
	AGGREGATE	•						
	~	You can connect components by clicking the green dot on a						
	BRANCH	component tile and dragging to the						
	>+	you want to connect						
	JOIN							

Joining Multiple Streams

Joining multiple streams is an important SAM capability. You accomplish this by adding the Join processor to your stream application. You can join on multiple streams using one of two concepts available in the **Window Type** filed:

About this task

- Processing Time Represents window based on when the event is processed by SAM
- Event Time Represents the window based on the timestamp of the event.

JOIN	N4LOO				×
CONFIGURATION NOTES					
Input	kafka_stream_104 👻	driverId -			Output
kafka_stream_104 🔹	JOIN TYPE*	SELECT STREAM*	SELECT FIELD*	WITH STREAM*	truckld* ⊪reger driverld*
eventTime* strang	INNER -	kafka_stream_105 -	driverId -	kafka_stream_104 🛛 👻	INTEGER driverName* STRING
eventsource* string truckld*	WINDOW TYPE*				routeld* INTEGER
INTEGER driverId*	Processing Time	~			route* string
driverName*	WINDOW INTERVAL*		SLIDING INTERVAL		event I ype* STRING
routeld* INTEGER	3	Seconds 💌	3	Seconds 👻	bouble longitude*
route* STRING	OUTPUT FIELDS* ① ALIAS FOR	OUTPUT FIELDS ARE MANDATOR	Y	SELECT ALL	correlationId*
string	× 1 eventTime as eventTime	× 1 eventSou	Irce as eventSource		geoAddress STRING
	× 1 truckld as truckld	× 1 driverId as	driverId		speed* INTEGER
correlationId*					Cancel Ok

JOIN					×
CONFIGURATION NOTES					
Input	SELECT STREAM*	SELECT FIELD WITH*			Output
kafka_stream_104 🔹	kafka_stream_104 🗸	driverId 👻			eventTime* string
eventTime* string	JOIN TYPE*	SELECT STREAM*	SELECT FIELD*	WITH STREAM*	eventSource* string truckld*
eventTimeLong*	INNER -	kafka_stream_105 v	driverId -	kafka_stream_104 v	INTEGER driverId*
eventSource*	WINDOW TYPE*		TIMESTAMP FIELDS*	LAG MILLISECONDS*	INTEGER driverName*
truckld* INTEGER driverld* INTEGER	Event Time	Ŧ	eventTimeLong	0	routeld*
driverName* ^{STRING} routeld*	WINDOW INTERVAL*		kafka_stream_104		string eventType* string
INTEGER route* STRING	3	Seconds 👻	CORRELATIONID	Seconds	latitude* DOUBLE longitude*
eventType*	OUTPUT FIELDS* 19 ALIAS FOR	OUTPUT FIELDS ARE MANDATORY	eventtimeLong	SELECT ALL	correlationId*
latitude* DOUBLE longitude* DOUBLE					Cancel Ok

For the purposes of this reference application, use processing time.

This section shows you how to configure a Join processor that joins the truck geo-event stream with the speed event stream, based on Requirement 5 of the use case.

Procedure

- 1. Drag a Join processor onto your canvas and connect it to a source.
- 2. Double-click the Join tile to open the Configuration dialog.
- **3.** Configure the Join processors according to the example below.

Example

JOIN	Join stream_1 on fi	eld driverld			Wait 5 seconds for
CONFIGURATION NOTES	↓		Inner join with stu	ream_2 on	streams to catch up before the join occurs
Input	kafka_stream_1 🔍	driverId -	L		Output
kafka_stream_1 📼	JOIN TYPE	SELECT STREAM	SELECT FIELD	WITH STREAM	INTEGER driverld* INTEGER
eventTime*	INNER -	kafka_stream_2 🔹	driverId 👻	kafka_stream_1 🗸 👻	driverName* STRING
eventsource* string truckld* integer	WINDOW INTERVAL TYPE*	-			routeld*
driverId* INTEGER driverName*	WINDOW INTERVAL*		SLIDING INTERVAL	*	eventType*
STRING routeld* INTEGER	05	Seconds -	5	Seconds	latitude* DOUBLE longitude*
route* string	OUTPUT FIELDS*			SELECT ALL	- correlationId*
eventType* STRING latitude* DOUBLE	× eventTime × eventSour × latitude × longitude	ce × truckld × driverld × × correlationId × geoAddress	driverName × routeld ×	route × eventType × •	geoAddress STRING speed* INTEGER
longitude* DOUBLE correlationId* LONG			1	The output of the joins	Cancel Ok

Filtering Events in a Stream using Rules

SAM provides powerful capabilities to filter events in the stream. It uses a Rules Engine, which translates rules into SQL queries that operate on the stream of data. The following steps demonstrate this, implementing Requirement 6 of the use case.

Procedure

1. Drag the Rule processor to the canvas and connect it from the Join processors.



- 2. Double-click the Rule processor, click the Add new Rules button, and create a new rule.
- 3. Click OK to save the new rule.

Example

EventType	Dot-Ann			×		
CONFIGURATION NO	TES	Click to add rules whi	ich get			
Input	+Add New Rules	translated into SQL on the and allows filtering of	ne stream stream	Output		
eventTime*		events		eventTime*		
eventSource*	Name	Condition	Actions	eventSource*		
truckId* INTEGER	Violation Event	eventType <> 'Normal'	e 🖉	truckId*		
driverId*		†		driverId*		
driverName*				driverName*		
routeld*		A rule that is translated into SQ	L	routeld*		
route* STRING		that looks for any event in the stream with an event type not				
eventType* string		equal to Normal, which represents a Violation Event				
latitude*				latitude*		
longitude*				longitude*		
correlationId*				correlationId*		
				Cancel Ok		

Using Aggregate Functions over Windows

Windowing is the ability to split an unbounded stream of data into finite sets based on specified criteria such as time or count, so that you can perform aggregate functions (such as sum or average) on the bounded set of events. SAM exposes these capabilities using the Aggregate processor. The Aggregate processor supports two window types, tumbling and sliding windows. The creation of a window can be based on time or count. The following images show how to use the Aggregate processor to implement Requirement 8 of the use case.

Procedure

1. Drag the Aggregate processor to the canvas and connect it to the Rule processor.

5	My Applic	ations / IOT-Trucking-Ref-App
	Q 07	Last Change:9s ago Version:CURRENT 🤨 🔾 🔅
f	SOURCE	
æ	E	
	EVENT HUBS	
\$	1	
	HDFS	20 TruckGeoEy
Ŷ	%	80 KAFKA
c	KAFKA	
10	PROCESSOR	& TruckSpeed (01) Drag the Aggregate
	Σ	processor onto the canvas and connect it to
	AGGREGATE	Violent Events Rule
	~	
	BRANCH	
	→	
	JOIN	

2. Double-click on the Aggregate processor, and configure it to calculate the average speed of driver over a threeminute duration.

CONFIGURATION NOTES	The fields to group by	At the end of the windo is the new schema that output to the stream:	w, this will be the		×
Input	SELECT KEYS*	average speed of every	unver	Output	
truckld*	× driverId × driverName × route		× 👻	driverId*	
driverId*	WINDOW INTERVAL TYPE*			driverName* STRING	
driverName* STRING	Time		-	route* STRING	
routeld* INTEGER	WINDOW INTERVAL*			speed_AVG*	
route* STRING	3	Minutes			
eventType* string latitude*	SLIDING INTERVAL				
DOUBLE longitude*	3	Minutes			
correlationId*	TIMESTAMP FIELD				
geoAddress* string	processingTime × 👻				
speed* INTEGER	Output Fields				
				Cancel	Ok

Implementing Business Rules on the Stream

This section shows you how to implement the business rule you created above to detect high speeding drivers. "High speed" is defined as greater than 80 miles per hour over a three-minute time window. This step partially implements Requirement 8 of the use case.

Procedure

1. Drag the Rule processor onto the canvas and connect to it to the DriverAvgSpeed Aggregate processor:

5	My Applic	ations / IOT-Trucking-Ref-App
	Q 🕼	Last Change: 0s ago Version: CURRENT 🤤 🔍 🌣
Ð	EVENT HUBS	
Ø	100	
Ŷ	HDFS	% TruckGeoEV ∢ 01 » MARKA → → → JOIN ≪ 10 1 » → ● ♥ ® EvenType < 01 » ●
r	PROCESSOR	8e TruckSpeed ∢01 ▶ ●
	Σ	Add the Rule processor to the
	AGGREGATE	DriverAvgSpeed Aggregate
	~	processor
	BRANCH	
	>≁	
	JOIN	

2. Configure the business rule as follows:

Add New Rule					×
RULE NAME*					
Speeding Driver					
DESCRIPTION*					
Driver who is speedir	ng excessively				
CREATE QUERY*	speed_AVG	× -	AN × –	80	× • +
QUERY PREVIEW:					
<pre>speed_AVG > 80</pre>					
					Cancel Ok

Results

The fully configured business rule should look similar to the following. Only high speed events continue on in the stream.

IsDriverSpeeding		Only bich	speed	×					
CONFIGURATION NOTES		events continue on in the stream							
Input	♣Add New Rules			Output					
driverId*				driverId*					
INTEGER driverName*	Name	Condition	Actions						
route* STRING	Speeding Driver	speed_AVG > 80	e 🖉	route* STRING					
speed_AVG* DOUBLE				speed_AVG* DOUBLE					
				Cancel Ok					

Transforming Data using a Projection Processor

It is common to do transformations on the events in the stream. In our case, before we alert on the speeding driver, we want to convert the average speed we calculated in the aggregate processor into a integer from a double so it is easier to display in the alert. The projection processor allows you to do these transformations.

Procedure

1. Drag the Projection processor onto the canvas and connect to it to the IsDriverSpeeding Rule processor:

5	My Applic	ations / IOT-Trucking-Ref-App						
.#.	Q 🕜					Last Change:23s ago	Version:CURRENT	ତ୍ତ୍ତ
æ	EVENT HUBS							
¢3	10°			► ∑ DriverAvgS	··· ∢01 ▶ • → • • • BDriverSp • RULE	 € 01 ► ● ● ● ● PROJECTION PROJECTION 	401▶●	
Ø	HDFS	KAFKA						
	KAFKA		4 01 ▶ • - ▶ • ● ● ■ EventType RULE	4 01 ▶ ●				
r	PROCESSOR	KAFKA € 01						
	Σ							
	AGGREGATE							
	~							
	BRANCH							
	>+							

2. When you double-click on the projection processor, you see a number of out-of-the-box functions, however a Round function does not exist.

PROJECTION						×
CONFIGURATION	NOTES					
Input		PROJECTION FIELDS*				Output
driverId*		Select			$\overline{\mathbf{v}}$	
driverName* straing		FUNCTION	ARGUMENTS	FIELDS NAME		
route* string		Select × 🔺	Select	•	+	
speed_AVG*		UPPER				
		LOWER				
		INITCAP				
		SUBSTRING				
		CHAR_LENGTH				
		CONCAT				
						Cancel Ok

- **3.** Adding UDFs (User Defined Functions) is easy to do within SAM. Follow the below steps to add Round UDF function to SAM.
 - a. From the left-hand menu, click Configuration, then Application Resources.
 - **b.** Select the **UDF** tab and click the + sign to create the ROUND UDF. The jar for this UDF can be downloaded from here, located in the custom-udf folder. The simple java class used to implement this Round function using the SAM SDK can be found here. Unzip the downloaded artifact and use the jar called sam-custom-udf-0.0.5.jar. Configure the UDF with the following values:

Edit UDF	×
NAME *	
ROUND_AUTOCREATED	
DISPLAY NAME *	
ROUND_AUTOCREATED	
DESCRIPTION *	.r)
Rounds a double to integer using Math.Round	tr
TYPE *	
FUNCTION	-
CLASSNAME *	tr
hortonworks.hdf.sam.custom.udf.math.Round	
UDF JAR *	У
Browse sam-custom-udf.jar	tr
Canc	el Ok o

c. After uploading the UDF, you should see the new Round UDF created.

Custom Processor UDF			Notifiers				
ROUND	Q						
Name	Description	Туре	Class Name	Argument Types	Return Type	Actions	

- **4.** After creating the UDF, go back to your Application and double-click Projection Processor you added to the canvas. You will see ROUND_AUTOCREATE in the FUNCTION drop down list.
- **5.** Configure the ROUND_AUTOCREATE function as the following:

Results

Round									×
CONFIGURATION	NOTES								
Input		PROJECTION FIELDS*					SELECT ALL	Output	
driver1d*	<i>x</i>	× driverId × driverName	× route				× 👻	driverId*	
driverName*		FUNCTION	ARGUMENTS		FIELDS NAME			driverName*	
route*		ROUND_AUTOCREATE	× speed_AVG	× -	speed_AVG_Round	+		route* string	
speed_AVG*								speed_AVG_Round*	
								Cancel	OK

Streaming Alerts to an Analytics Engine for Dashboarding

In addition to creating notification alerts, a common use case requirement is to send these alerts to a dashboard so they can be displayed and visualized. SAM offers this capability by allowing you to stream data into DRUID and then using Superset to create dashboards and visualizations.

Procedure

1. Drag the Druid sink to the canvas and connect to it to the Round Projection.

-d: Tru	ckGeoEv ∉ 01 ▶● SIS_AU						
		4 01 ▶• →•	40110	DriverAvgS <01	•••• isDriverSp RULE	€01 ► • • • Round PROJECTION	401 ▶ → S3-Speedin S3_AUTOCREATED
	ckSpeed ∢01)● sis_AU						

2. Stream these events into a Druid cube called alerts-speeding-drivers-cube by configuring the Druid processor like the following:

Alert-Speeding-Driver-	Cube
REQUIRED OPTIONAL N	OTES
Input driverId* INTEGER driverName* STRING speed_AVG* DOUBLE speed_AVG_Round* LONG	DATASOURCE NAME * alerts-speeding-drivers-cube ZOOKEEPER CONNECT STRING * secure-fenton-hdf1.field.hortonworks.com:2181,secure-f DIMENSIONS * * driverId * driverName * route * speed_AVG * speed_AVG_Round TIMESTAMP FIELD NAME * processingTime * WINDOW PERIOD * PT10M Cancel Ok

3. In the Creating Visualization Section, describe how to create dashboards for the alerts-speeding-drivers-cube.

Streaming Violation Events to an Analytics Engine for Descriptive Analytics

All infraction events need to be available for descriptive analytic (dash-boarding, visualizations, etc.) by a business analyst. The analyst needs the ability to do analysis on the streaming data. The analytics engine in SAM is powered by Druid. The following steps show how to stream data into Druid, so that a business analyst can use the Stream Insight Superset module to generate descriptive analytics.

Procedure

1. Drag the Druid processor to the canvas and connect it to the ViolationEvents Rule processor.

5	My Applio	cations / IOT-Trucking-Ref-App						
ф.	Q 🗭					Last Change:1s ago	Version:CURRENT	ତ୍ତ୍ର 🕈
æ	EVENT HUBS					→• <u>/</u>		
ⅆ	CP HDFS				••• • • • • • • • • • • • • • • • • •	Round 4 01 PROJECTION		
Ø	% Kafka	% TruckGeoEv ∢o1 ▶ ● KAFKA JOIN	∢01 ▶ • → ● ⊕ BeventType	4 01▶●		L.	∋ Dashboard ∢01 ►	
r	PROCESSOR	8 TruckSpeed ∉01 ▶ ●			401▶			
	AGGREGATE							
	BRANCH							

2. Configure the Druid processor.

You can edit the ZooKeeper connect string in the advanced section of the Druid Service in Ambari, under the property druid.zk.service.host.

Violation-Events-Cu	ıbe	The name of the insight data source/cube to which you want to stream data.
REQUIRED OPTIONAL	NOTES	Business analysts use these data sources to query the data
Input	DATASOURCE	IAME *
eventTime* string	violation-ev	vents-cube
eventSource* STRING	ZOOKEEPER CO	DNNECT STRING *
truckld* INTEGER	secure-fent	on-hdf1.field.hortonworks.com:2181,secure-f
driverld* INTEGER driverName* STRING		v suastSauraa v truskid
routeld* INTEGER	× driverId	× driverName × routeld × route
route* STRING	× eventType	× latitude × longitude
eventType* string	× correlatio	nld × geoAddress × speed
latitude* DOUBLE	TIMESTAMP FI	ELD NAME *
longitude*	processing	Time 👻
correlationId*	WINDOW PERIO)D *
		Cancel Ok

3. Configure the Aggregator Info settings, under the OPTIONAL menu

Violation-Events-Cub	e	×
REQUIRED OPTIONAL	NOTES	
Input eventTime* string eventSource* string truckId* integer driverId* integer driverName* string routeId* integer routeld* integer string eventType* string latitude* DOUBLE	DRUID PARTITIONS 1 PARTITION REPLICATION 1 Aggregator Info + Aggregator Info (Count Aggregator) NAME * Cnt	
correlationId*	Add a Count Aggregator and give it a name Cancel C	lk

Streaming Violation Events into a Data Lake and Operational Data Store

Another common requirement is to stream data into an operational data store like HBase to power real-time web applications as well as a data lake powered by HDFS for long term storage and batch ETL and analytics.

Procedure

1. Drag the HBase sink to the canvas and connect it to the ViolationEvents Rule processor.

5	My Applic	cations / IOT-Trucking-Ref-App							
#	Q 🕼					Last	Change:1s ago	Version:CURRENT	ତ୍ତ୍ର 🕈
ති	EVENT HUBS						└ ▶• ①	NOTIFICATION 401	
ß	OP HDFS			DriverAvgS.	•• ∢01▶•→• ⊕® isDriverSp	€01 ₽ ●●● ■< Round PROJECTION	N 401 P		
ک مر	KAFKA	Se TruckSpeed 401 - Join	∢01 ▶ • → ● [●] ® EventType _® ® _{RULE}	 € 01 ● ● ● ○ Violation 	401.		↓ ••	Dashboard 401	
		8º KARKA		HBASE	4 01 1				
	BRANCH								

2. Configure the Hbase Sink using the following parameters.

Operational-Store-Violation-Events					
	NOTES				
Input eventTime*	WRITE TO WAL?				
string eventSource* string truckId*	eventTime	•			
INTEGER driverId* INTEGER driverName*					
routeld* INTEGER route*					
eventType* STRING latitude* DOUBLE					
longitude* DOUBLE correlationId* LONG					
	Cancel	Ok			

Operational-Store-Violation-Events ×						
REQUIRED OPTIONAL N	OTES					
eventTime* string eventSource* string truckId* integer driverId* integer driverName* string routeId* integer routed*	CLUSTER NAME * streamanalytics HBASE TABLE * default:violation_events COLUMN FAMILY * events BATCH SIZE *					
eventType* STRING latitude* DOUBLE longitude* DOUBLE correlationId* LONG	100 Cancel Ok					

3. Drag the HDFS sink to the canvas and connect it to the ViolationEvents Rule processor.

5	My Applic	ations / IOT-Trucking-Ref-App							
	Q 🕼					Last C	hange:0s ago	Version:CURRENT	ତ୍ତ୍ର 🕈
æ	EVENT HUBS						↓ •	NOTIFICATION 401	
6	HDFS	% TruckGeoFy		► ∑ DriverAvgS. AGGREGATE		401 PROJECTION	401▶●		
ي بر	KAFKA	% KAFKA	∢ 01 ▶ • → ● ● EventType	∢01 ▶ • • • → Violation	≪ 01 ▶		L	Dashboard 401 b	
	\sum	20 KARKA		->- Derationa	… ∢01▶				
	BRANCH			HBASE	€01.▶				
	PMML								

4. Configure HDFS as below.

Make sure you have permission to write into the directory you have configured for HDFS path.

Data-Lake-HDFS	×
REQUIRED OPTIONAL NO	DTES
Input eventTime* string eventSource* string truckId* integer driverId* integer driverName* string routeId* integer routed* string eventType*	PATH * /apps/trucking-app FLUSH COUNT * 1000 ROTATION POLICY Time Based Rotation ROTATION INTERVAL MULTIPLIER * 3
STRING latitude* DOUBLE longitude* DOUBLE correlationId* LONG	ROTATION INTERVAL UNIT * MINUTES OUTPUT FIELDS * Cancel Ok

Deploy a SAM Application

Configure Deployment Settings

Before deploying the application, you must configure deployment settings such as JVM size, number of ackers, and number of workers. Because this topology uses a number of joins and windows, you should increase the JVM heap size for the workers.

Procedure

- 1. Click the gear icon at the top right corner of the canvas to display the Application Configuration dialog.
- 2. Increase Number of Workers to 5.
- 3. Set Topology Worker JVM Options to -Xmx3072m.

Example

Application Configuration				
GENERAL ADVANCED				
NUMBER OF WORKERS				
3				
NUMBER OF ACKERS				
1				
TOPOLOGY MESSAGE TIMEOUT (SECONDS)				
40				
TOPOLOGY WORKER JVM OPTIONS				
-Xmx3072m				
Cancel)k			

Deploy the App

After you have configure the application's deployment settings, click the **Deploy** button at the lower right of the canvas.
Application Configuration	×
GENERAL ADVANCED	
NUMBER OF WORKERS	
3	
NUMBER OF ACKERS	
1	
TOPOLOGY MESSAGE TIMEOUT (SECONDS)	
40	
TOPOLOGY WORKER JVM OPTIONS	
-Xmx3072m	
Cancel	

During the deployment process, Streaming Analytics Manager completes the following tasks:

- 1. Construct the configurations for the different big data services used in the stream app.
- 2. Create a deployable jar of the streaming app.
- **3.** Upload and deploy the app jar to streaming engine server.

As SAM works through these tasks, it displays a progress bar.



Building Application Jars

The stream application is deployed to a Storm cluster based on the Storm Service defined in the Environment associated with the application.

After the application has been deployed successfully, SAM notifies you and updates the button to red to indicate it is deployed. Click the red button to kill/undeploy the app.

5	My Applic	ations / IOT-Trucking-Ref-App		Application Deployed Successfully
#	Q 🕼		Last Change	:23s ago Version:CURRENT 🔍 Q 🔅
æ	EVENT HUBS			NOTIFICATION 401
đ	CP HDFS		► ∑ DriverAvgS <01 ► → S isDriverSp <01 ► → RULE ROLLE	4 01 1-
Ø	% KAFKA	KAFKA Solution KAFKA Solution Solut	401 b 0 * → Wolation 401 b	→• ⇒ Dashboard ∢01 ▶
۶	PROCESSOR	8 TruckSpeed ∉ 01 ▶ •		
			Operationa (01)	
	REANCH		→ c Data-Lake 401 >	
	>+			
	JOIN			
				Status:
»	PROJECTION			

Advanced: Performing Predictive Analytics on the Stream using SAM

Requirement 10 of this use case states the following:

Execute a logistical regression Spark ML model on the events in the stream to predict if a driver is going to commit a violation. If violation is predicted, then alert on it.

HDP, the Hortonworks data at rest platform provides a powerful set of tools for data engineers and scientists to build powerful analytics with data processing engines like Spark Streaming, Hive, and Pig. The following diagram illustrates a typical analytics life cycle in HDP.



Once the model has been trained and optimized, you can create insights by scoring the model in real-time as events are coming in. The next set of steps in the life cycle score the model in real-time using HDF components.



In the next few sections we will walk through how to do steps 5 through 9 in SAM.

Logistical Regression Model

In steps 1-4 with HDP, we were able to build a logistical regression model. The model was then exported into PMML. The following diagram illustrates the features, coefficients, and output of the model.



Export the Model into SAM's Model Registry

SAM provides a registry where you can store PMML models. To get started with predictive analytics, upload this logistical regression model.

Procedure

- 1. Download this PMML model and save it locally with an .xml extension.
- 2. Select the Model Registry menu item.
- 3. Click the + icon.
- **4.** Give your PMML model a name.
- 5. From Upload PMML File, select the PMML file you just downloaded.

×
_
)k

6. Click Ok.

Results

The model is saved in the Model Registry.

5	Model Regis	try		
#			Search by name	Q
æ		Model Name	PMML File Name	Actions
Ø		DriverViolationPredictionModel	$\label{eq:constraint} DriverViolationLogisticalRegessionPredictionModel-pmml.xml$	â
Ŷ				
ŗ				

Enrichment and Normalization of Model Features

Now that the model has been added to the model registry, you can use it in the streaming application by the PMML processor. Before the model can be executed, you must enrich and normalize the streaming events with the features required by the model. As the above diagram illustrates, there are seven features in the model. None of these features come as part of the stream from the two sensors. So, based on the driverId and the latitude and longitude location, enrich the streaming event with these features and then normalize it required by the model. The table below describe each feature, enrichment store, and the normalization required.

Feature	Description	Enrichment Store	Normalization
Model_Feature_Certification	Identifies if the driver is certified or not	HBase/Phoenix table called drivers	"yes" # normalize to 1 "no" # normalize to 0
Model_Feature_WagePlan	Identifies if the driver is on an hourly or by miles wage plan	HBase/Phoenix table called drivers	"Hourly" # normalize to 1 "Miles" # normalize to 0
Model_Feature_Fatigue ByHours	The total number of hours driven by the driver in the last week	HBase/Phoenix table called timesheet	Scale by 100 to improve algorithm performance (e.g., hours/100)

Model_Feature_Fatigue ByMiles	The total number of miles driven by the driver in the last week	HBase/Phoenix table called timesheet	Scale by 1000 to improve algorithm performance (e.g.,miles/1000)
Model_Feature_Foggy Weather	Determines if for the given time and location, if the conditions are foggy	API to WeatherService	if (foggy) # normalize to 1 else 0
Model_Feature_Rainy Weather	Determines if for the given time and location, if the conditions are rainy	API to WeatherService	if (raining) -> normalize to 1 else 0
Model_Feature_Windy Weather	Determines if for the given time and location, if the conditions are windy	API to WeatherService	if (windy) # normalize to 1 else 0

Upload Custom Processors and UDFs for Enrichment and Normalization

To perform the above enrichment and normalization, upload the custom UDFs and processors you downloaded in the previous section.

Upload Custom UDFs

Steps for Uploading the Timestamp_Long UDF

- 1. From the left-hand menu, click **Configuration**, then **Application Resources**.
- 2. Select the UDF tab and click the + sign to create the TIMESTAMP_LONG UDF. This UDF will convert a string date time to a Timestamp long. The simple class for this UDF using the SAM SDK can be found here.

The jar for this UDF is located in SAM_EXTENSIONS/custom-udf.

3. Configure the UDF with the following values:

Edit UDF	×
NAME *	
TIMESTAMP_LONG_AUTOCREATED	
DISPLAY NAME *	
TIMESTAMP_LONG_AUTOCREATED	
DESCRIPTION *	
Converts a string date time to a Timestamp Long	
TYPE*	
FUNCTION	~
CLASSNAME *	
hortonworks.hdf.sam.custom.udf.time.ConvertToTimestampLong	
UDF JAR *	
Browse sam-custom-udf.jar	
	Cancel Ok

Steps for Configuring the Get_Week UDF

1. Select the UDF tab and click the + sign to create the GET_WEEK UDF.

The jar for this UDF is located in SAM_EXTENSIONS/custom-udf. This UDF will convert a timestamp string into the week of the year which is required for an enrichment query. The simple class for this UDF using the SAM SDK can be found here.

2. Configure the UDF with the following values:

Edit UDF	×
NAME *	
GET_WEEK_AUTOCREATED	
DISPLAY NAME *	
GET_WEEK_AUTOCREATED	
DESCRIPTION *	
For a given date time string, the functions returns the week of the date/time	
TYPE*	
FUNCTION	*
CLASSNAME *	
hortonworks.hdf.sam.custom.udf.time.GetWeek	
UDF JAR *	
🕞 Browse sam-custom-udf.jar	
Cancel	Ok

Upload Custom Processors

Steps for Uploading the ENRICH-PHOENIX Custom Processor

- 1. From the left-hand menu, click Configuration, then Application Resources.
- 2. Select Custom Processor and click the + sign to create the ENRICH-PHOENIX processor.

Configure the processor with the following values. This processor can be used to enriched streams with data from Phoenix based on a user supplied SQL statement. The java class for this processor using the SAM SDK can be found here.

BIGGRIPPINA CLISIMULA CLISIMU	NAME*	ENRICH-PHOEM	NIX_AUTOCREATED				_	
CUSADANE* Internote Industry processes existing homes. Proceeding uses CUMO LA* Control Contro Contro Control Control Contro Control Control Control Control Co	DESCRIPTION*	Enriches the inj	out schema with data from P	hoenix base	ed on us	er supplied		
UNUAD LAI Existent Beater Autor CONFORMER Existent LAI Autor Option Top of an top of a	CLASSNAME*	hortonworks.hd	lf.sam.custom.processor.en	rich.phoeni>	.Phoeni	ixEnrichmen		
CONTR FIELD Action/URL Image: Several to the control of Line of the control of Line of the control of SEQUE/LX-(HOST):SZC-(FORT) 0 ASSNICH to the control of Line of the control of SEQUE/LX-(HOST):SZC-(FORT) 0 Assnice to the control of Line of the control of SEQUE/LX-(HOST):SZC-(FORT) 0 Assnice to the control of Line of the control of SEQUE/LX-(HOST):SZC-(FORT) 0 Assnice to the control of Line of the control of Line of the control of SEQUE/LX-(HOST):SZC-(FORT) 0 Assnice to the control of Line of the control of Line of the control of SEQUE/LX-(HOST):SZC-(FORT) 0 Assnice to the control of Line of the control of Line of the control of Line of the control on a solute Hillson/Phonenic Cluster 0 Assnice to the control of Line of the control on a solute Hillson/Phonenic Cluster 0 0 Assnice to the control of Line of the control on a solute Hillson/Phonenic Cluster 0 0 Assnice to the control of Line of the control on a solute Hillson/Phonenic Cluster 0 0 Assnice to the control of Line of the control on a solute Hillson/Phonenic Cluster 0 0 Assnice to the control of Line of the control on a solute Hillson/Phonenic Cluster 0 0 Assnice to the control of Line of the control on a solute Hillson/Phonenic Cluster 0 0 Assnice to the control of Line of the control on a solute Hillson/Phonenic Cluster 0 0 Assnice to the c	UPLOAD JAR*	🗁 Browse 🛛 📢	am-custom-processor-jar	-with-depe	ndenci	es.jar		
Field Name Optional Type Paradite Tooltip Action 26ServerUit Phoneiri Zookeeper Com false atin a SQL to execute of the enriched values a enrichmentSQL Enrichment Output Fields false atin Q SQL to execute for the enriched values a enrichedOutput Enrichment Output Fields false atin Q The output field names to store new enriched values a execureCluster Secure Cluster false false atin Q The output field names to store new enriched values a kerberoacCluster/ Fields Secure Cluster false false atin Q The output field names to store new enriched values a kerberoacCluster/ Fields KerberoacKeyTab KerberoacKeyTab KerberoacKeyTab KerberoacKeyTab KerberoacKeyTab Secure Cluster atin a NEVUT SCHEMA © 1 Image: Secure Cluster Secure Cluster atin atin a Secure Cluster atin a NEVUT SCHEMA © 1 Image: Secure Cluster Image: Secure Cluster Secure Cluster Secure Cluster	CONFIG FIELDS	Add Config Fields						
zkServerLid Plocesiz Zookkeeper Com false strin Cookkeeper server uni in the format of \$F2QNLZK_HOST \$ZK_PORT \$\$ errichmentSQL false strin si \$\$		Field Name	UI Name	Optional	Туре	Default Value	Tooltip	Actions
enrichmentSQL false strin Q SQL to execute for the enriched values \$		zkServerUrl	Phoenix Zookeeper Conn ection URL	false	strin g		Zookeeper server url in the format of \$FQDN_ZK_HOST:\$ZK_PORT	/ 0
enrichedOutput Enrichment Output Fields false etrin g The output field names to store new enriched values false secureCluster Secure Cluster false bood false bood false Oheck if connecting to a secure HBase/Phoenix Cluster. Requi false kreberoacKeyTab Kreberoa KeyTab File true gtin Q The principal uses to connect to secure HBase/Phoenix Cluster. Requi false NEWT SOHEMA O Kreberoa KeyTab File true gtin Q Kerberoa Key Tab File Location on each of the worker nodes for thee principal uses to connect to secure HBase/Phoenix Cluster. Requi false NEWT SOHEMA O V V true gtin Q Kerberoa Key Tab File Location on each of the worker nodes for thee principal uses to connect to secure HBase/Phoenix Cluster. Requi false NEWT SOHEMA O V V true gtin false true gtin false true gtin false		enrichmentSQL	Enrichment SQL	false	strin g		SQL to execute for the enriched values	/ 0
secureCluster false hoo false chck if connecting to a secure HBase/Phoenix Cluster i kerberos Client Pincipal twe g trin g che The principal uses to connect to secure HBase/Phoenix Cluster. Requil i kerberos KeyTab kerberos KeyTab File twe g trin g che reprincipal uses to connect to secure HBase/Phoenix Cluster. Requil i NRUT SCHEAL © kerberos KeyTab File twe g trin g che reprincipal uses to connect to secure HBase/Phoenix Cluster. Requil i NRUT SCHEAL © kerberos KeyTab File twe g trin g che reprincipal uses to connect to secure HBase/Phoenix Cluster. Requil i NRUT SCHEAL © kerberos KeyTab File twe g trin g che reprincipal configured reprincipal configured i i NRUT SCHEAL © 1 strin g trin strin g reprincipal configured strin g repri		enrichedOutput Fields	Enrichment Output Fields	false	strin g		The output field names to store new enriched values	/ 0
Kerberos Client Principal True strin The principal uses to connect to secure HBase/PHoenix Cluster. Requit Image: Cluster is checked Kerberos Key Tab File true strin g Kerberos Key Tab File location on each of the worker nodes for thee principal uses to connect to secure HBase/PHoenix Cluster. Requit Image: Cluster is checked INPUT SCHEMA O Image: Cluster is checked Image: Cluster is c		secureCluster	Secure Cluster	false	bool ean	false	Check if connecting to a secure HBase/Phoenix Cluster	/ 0
kerberos KeyTab File tue strin c Kerberos Key Tab File location on each of the worker nodes for thee pri opal configured C INPUT SCHEMA • •<		kerberosClientP rincipal	Kerberos Client Principal	true	strin g		The principal uses to connect to secure HBase/PHoenix Cluster. Requi red if secureCluster is checked	/ 0
INPUT SCHEMA		kerberosKeyTab File	Kerberos Key Tab File	true	strin g		Kerberos Key Tab File location on each of the worker nodes for thee pri ncipal configured	/ 0
	OUTPUT SCHEMA							
				С	ancel	Save		

ENRICH-PHOENIX Configuration Values

- Streaming Engine Storm
- Name ENRICH-PHOENIX
- Description Enriches the input schema with data from Phoenix based on user supplied SQL
- ClassName hortonworks.hdf.sam.custom.processor.enrich.phoenix.PhoenixEnrichmentSecureProcessor
- Upload Jar The jar for this custom processor can be found under SAM_EXTENSIONS/custom-processor/ sam-custom-processor-jar-with-dependencies.jar

Click the Add Config Fields button and the following three configuration fields:

• Add a config field called **zkServerUrl** with the following values:

- a. Field Name zkServerUrl
- b. UI Name Phoenix ZooKeeper Connection URL
- c. Optional false
- d. Type string
- e. ToolTip ZooKeeper server URL in the format of \$FQDN_ZK_HOST:\$ZK_PORT
- Add a config field called **enrichmentSQL** with the following values:
 - a. Field Name enrichmentSQL
 - b. UI Name Enrichment SQL
 - c. Optional false
 - **d. Type** string
 - e. ToolTip SQL to execute for the enriched values
- Add a config field called **enrichedOutputFields** with the following values:
 - a. Field Name enrichedOutputFields
 - b. UI Name Enrichment Output Fields
 - c. Optional false
 - d. Type string
 - e. ToolTip The output field names to store new enriched values
- Add a config field called **secureCluster** with the following values:
 - a. Field Name secureCluster
 - b. UI Name Secure Cluster
 - **c. Optional** false
 - d. Type boolean
 - e. ToolTip Check if connecting to a secure HBase/Phoenix Cluster
- Add a config field called kerberosClientPrincipal with the following values:
 - a. Field Name kerberosClientPrincipal
 - b. UI Name Kerberos Client Principal
 - c. Optional true
 - d. Type string
 - e. ToolTip The principal uses to connect to secure HBase/Phoenix Cluster. Required if secureCluster is checked.
- Add a config field called **kerberosKeyTabFile** with the following values:
 - a. Field Name kerberosKeyTabFile
 - **b.** UI Name Kerberos Key Tab File
 - c. Optional true
 - **d. Type** string
 - e. ToolTip Kerberos Key Tab File location on each of the worker nodes for the configured principal

Steps for Uploading the ENRICH-WEATHER Custom Processor

- 1. Select **Custom Processor** and click the + sign to create the ENRICH-WEATHER processor. This processor can be used to enrich streams with weather data based on time and lat/long location. The java class for this processor using the SAM SDK can be found here.
- **2.** Configure the processor with the following values.

Configuration / /	Application Resources							
	Custom Processor UDF	Notifiers						
	STREAMING ENGINE*	STORM						
	NAME*	ENRICH-WEATHER_AUTOC	REATED					
	DESCRIPTION*	Enrichment with normalize	d weather data required for the model					
	CLASSNAME*	hortonworks.hdf.sam.cust	om.processor.enrich.weather.WeatherEnric	chmentProcesso				
	UPLOAD JAR*	Browse CustomProc	essor.jar					
	CONFIG FIELDS	Add Config Fields						
		Field Name	UI Name	Optional	Туре	Default Value	Tooltip	Actions
		weatherServiceURL	Weather Web Service URL	false	string	http://weather.com/api?lat=\${latitude}&lng=\${longitude}	The URL to the Weather Web Service	/ 0
	Revi Schema O	1 [2 { 3 'name': 'dr. 4 'type': 'lr. 5 'optional': 6 }, 7 { 8 'name': 'la 9 'type': 'Do 10 'optional': 1 }, 12 { 13 'type': 'Do 15 'optional': 15 'optional': 16 'type': 'Do 17 'type': 'Do 18 'type': 'Do 18 'type': 'Do 18 'type': 'Do 18 'type': 'Do 19 'type': 'Do 10 'type:''	verId', TGDR', false titude', BLC', gitude', BLC', false			(*) 0.1244		
	OUTPUT ISOBAA 0	<pre>1 [2 { 3</pre>	<pre>lal Feature_FoggyWeather", MID", false kel_Feature_RainyWeather", MID", false kel_Feature_NindyWeather", false</pre>			↓ I CLEAR		

ENRICH-WEATHER Configuration Values

- Streaming Engine Storm
- Name ENRICH-WEATHER
- **Description** Enrichment with normalized weather data for a geo location
- ClassName hortonworks.hdf.sam.custom.processor.enrich.weather.WeatherEnrichmentProcessor
- Upload Jar The jar for this custom processor can be found under SAM_EXTENSIONS/custom-processor/samcustom-processor.jar

Click the **Add Config Fields** button and a configuration field with the following values:

- **Field Name** weatherServiceURL
- **UI Name** Weather Web Service URL
- **Optional** false
- **Type** string
- **Tooltip** The URL to the Weather Web Service

Input and Output Schema for ENRICH-WEATHER

- Copy this input schema and paste into the INPUT SCHEMA text area box
- Copy this output schema and paste into the OUTPUT SCHEMA text area box

Steps for Uploading the NORMALIZE-MODEL-FEATURES Custom Processor

- **1.** Select the **Custom Processor** tab and click the + sign to create the NORMALIZE-MODEL-FEATURES processor. This processor normalizes the enriched fields to a format that the model is expecting.
- **2.** Configure the processor with the following values:

Configuration / Application	Resources								
STREAMING	S ENGINE*	STORM							
NAME*		NORMALIZE-MODEL-FEATURES	DELAY_AUTOCREATED						
DESCRIPTIO	*אמ	Normalize the features of the me	odel before passing it to model with option to cause latency						
CLASSNAM	IE•	hortonworks.hdf.sam.custom.pr	ocessor.enrich.driver.predictivernodel.FeatureNormalizationWithDelay	Processor					
UPLOAD JA	R*	Browse CustomProcesso	jar						
CONFIG FIE	LDS	Add Config Fields							
		Field Name	UI Name		Optional	Туре	Default Value	Tooltip	Actions
		delayTimeOutSecs	Timeout Delay for Monitoring Use Case (Seconds)		true	number	0	timeout delay in seconds	/ 0
NPUTSON	0MA 0	• 1			2	CLEAR			
output se	HEMA O	<pre>1 { 2 { 3</pre>	Teature_FoggyWeather", se Teature_RainyWeather", se Teature_MindyWeather", se		21	CLEAR			

NORMALIZE-MODEL-FEATURES Configuration Values

- Streaming Engine Storm
- Name NORMALIZE-MODEL-FEATURES
- Description Normalize the features of the model before passing it to model
- ClassName -

hortonworks.hdf.sam.custom.processor.enrich.driver.predictivemodel.FeatureNormalizationProcessor

• Upload Jar – The jar for this custom processor can be found under SAM_EXTENSIONS/custom-processor/samcustom-processor.jar

Input and Output Schema for NORMALIZE-MODEL-FEATURES

• Copy this output schema and paste into the OUTPUT SCHEMA text area box

Result

You have uploaded three custom processors required to do enrichment of the stream and normalization of the enriched values to feed into the model.

Configuration	n / Application Resources			
[Custom Processor	UDF Notifiers		0
	Search by name	Q		
	Name	Description	Jar File Name	Actions
	ENRICH-PHOENIX	Enriches the input schema with data from Phoenix based on user supplied SQL	sam-custom-processor-0.0.5-jar-with-dependencies.jar	e 🖉
	ENRICH-WEATHER	Enrichment with normalized weather data for a geo location	sam-custom-processor-0.0.5.jar	e 🖉
	NORMALIZE-MODEL-FEATURES	Normalize the features of the model before passing it to model	sam-custom-processor-0.0.5a.jar	e 🖉
	Configuratio	Configuration / Application Resources Custom Processor Search by name ENRICH-PHOENIX ENRICH-WEATHER NORMALIZE-MODEL-FEATURES	Configuration / Application Resources Custom Processor UDF Notifiers Search by name Q Name Description ENRICH-PHOENIX Enriches the input schema with data from Phoenix based on user supplied SQL ENRICH-WEATHER Enrichment with normalized weather data for a geo location NORMALIZE-MODEL-FEATURES Normalize the features of the model before passing it to model	Configuration Resources UDF Notifiers Search by name Q Name Description Jar File Name Colspan="2">Colspan="2"Colspan=""2"Colspan="2"Colspan="2"Colspan

If you go back to the Stream Builder, you will see three new custom processors on the palette.



Scoring the Model in the Stream using a Streaming Split Join Pattern

Now that you have created the enrichment store, loaded the enrichment data, and uploaded the custom UDFs and processors to SAM, build the stream flow to score the model in real-time. In this case, you want to predict violations for events that are not blatant infractions.

Procedure

- **1.** Click into the Trucking IOT application you built.
- 2. Double-click the Event Type rule processor to display the Add New Rule dialog.
- **3.** Configure the new rule with the following values:

Add New Rule				2
RULE NAME*				
Non Violation Eve	ents			
DESCRIPTION*				
Events that are no	ot violations that we want	t to do predictions on		
CREATE QUERY*	eventType	× 👻 EQUALS	× 💌 'Normal'	× -
QUERY PREVIEW:				
eventType = 'N	lormal'			
				Cancel Ok

Results

Your new rule is added to the Event Type processor.

EventType	n			×
CONFIGURATION NOTES				
Input	+Add New Rules			Output
				eventTime*
eventSource*	Name	Condition	Actions	eventSource*
truckld*	ViolationEvents	eventType <> 'Normal'	۵ 🖉	truckld*
driverId*	Non Violation Events	eventType = 'Normal'	e 🖉	driverId*
driverName* STRING				driverName* string
routeld*				routeld*
route* STRING				route* STRING
eventType* STRING				eventType* string
latitude*				latitude* DOUBLE
longitude*				longitude*
correlationId*				correlationId*
				Cancel Ok

Streaming Split Join Pattern

About This Task

Your objective is to perform three enrichments:

- Retrieve a driver's certification and wage plan from the driver's table.
- Retrieve the driver's hours and miles logged from the timesheet table.
- Query weather information for a specific time and location.

To do this, use the split join pattern to split the stream into three, perform the enrichment in parallel, and then re-join the three streams.

Steps for Creating a Split Join Key

1. Create a new split key in the stream which allows you to join in a common field when you join the three stream.

To do this, drag the projection processor to the canvas and create a connection from the EventType rule processor to this projection processor.

When configuring the connection, select the Non Violation Events Rule which tells SAM to only send non-violation events to this project processor.

Event	Type-PROJECTION	×
STREAM	ID*	
rule_tr	ansform_stream_3	-
FIELDS		
1	[
2	{	
3	"name": "eventTime",	
4	"type": "STRING",	
5	"optional": false	
б	},	
7	{	
8	"name": "eventSource",	
9	"type": "STRING",	
10	"optional": false	
11	},	
12	{ 	
13	"name": "truckld",	
14	"type": "INTEGER",	
	ODTIONAL : TAISE	
RULES*		
Non V	iolation Events	*
GROUPIN	IG*	
SHUF	E	-
	Cancel	Ok

2. Configure the projection processor to create the split join key called splitJoinValue using the custom UDF you uploaded earlier called "TIMESTAMP_LONG".

You will also do a transformation which calculates the week based on the event time which is required for one of the enrichments downstream. Configure the processor with the following parameters:

plit					1
ONFIGURATION NOTE	S				
nput	PROJECTION FIELDS*				Output
eventTime* STRING eventSource* STRING	× eventTime × eventSource × latitude × longitude ×	× truckId × driverio correlationId × geoAdd	d × driverName × routeld × route dress × speed	× eventType	driverName* STRING routeld* INTEGER
truckId*	FUNCTION	ARGUMENTS	FIELDS NAME		route* STRING
driverld*	TIMESTAMP_LONG × \neg	× eventTime	✓ splitJoinValue	+	eventType*
driverName*		X quantTime			latitude*
routeld* NTEGER	GEI_WEEK A V	 eventrime 	week	•	longitude*
oute*					correlationId*
eventType*					geoAddress*
atitude*					speed*
ongitude*					splitJoinValue*
correlationId*					week*

Steps for Splitting the Stream into Three to Perform Enrichments in Parallel

1. With the split join key created, you can split the stream into three to perform the enrichments in parallel.

To do the first split to enrichment the wage and certification status of driver, drag the "ENRICH-PHOENIX" processor to the canvas and connect it from the Split project processor.

- 2. Configure the enrich processor with the following parameters:
 - a. ENRICHMENT SQL: select certified, wage_plan from drivers where driverid=\${driverId}
 - b. ENRICHMENT OUTPUT FIELDS: driverCertification, driverWagePlan
 - c. SECURE CLUSTER: false
 - d. OUTPUT FIELDS: Click Select All.
 - e. NEW OUTPUT FIELDS: Add new output fields for the two enriched values: driverCertification and driverWagePlan.

After this processor executes, the output schema will have two fields populated called driverCertification and driverWagePlan.

ONFIGURATION	NOTES					
Input	1	KERBEROS KEY TAB FILE				Output
eventTime*	l					eventTime* STRING
eventTimeLong*	(OUTPUT FIELDS			SELECT ALL	eventSource* STRING
eventSource*		× eventTime × eventS	ource × truckld	× driverId		truckId*
truckld*		× driverName × routel	d × route ×	eventType × la	atitude 🗸 👻	driverId*
driverId*		× splitJoinValue × wee	ek × eventTime	Long		driverName* STRING
driverName*						routeld*
routeld*	1	NEW OUTPUT FIELDS +				route* STRING
route* string		Field Name	Туре	Optional	Actions	eventType* STRING
eventType* STRING		driverCertification	STRING	false	/ ĉ	latitude* DOUBLE
latitude*		driverWagePlan	STRING	false	2 A	longitude*
		anverwager lan	0111110	Tuise		correlationId*

3. Create the second stream to enrich the drivers hours and miles logged in last week by dragging another "ENRICH-PHOENIX" processor to the canvas and connecting it from the Split projection processor.

5	My Applic	cations / IOT-Trucking-Ref-App						
*	Q 🕜					Last Change:0s ago	Version:CURRENT	ତ୍ତ୍ ପ୍ 🌣
æ	EVENT HUBS					 *•	NOTIFICATI ◀ 01 ▶ NOTIFICATION	
đ	COP HDFS				 € IsDriverSp € RULE 	Round PROJECTION 401		
Ø	%	& TruckGeoEv ∢01 ▶ ●	∢01▶•→•• 🐜 EventType	401 ▶ ••• → Violation	401 >	L,	Dashboard 401	
×	PROCESSOR	& TruckSpeed ∉01 ►	- NULL					
				HBASE	401>			
	BRANCH			Data-Lake	401			
	>+							
	JOIN							
	PMML				ENRICH-PHOENIX 01 P			
	PROJECTION							
	20							

- 4. Configure the enrich processor with the following parameters:
 - a. ENRICHEMNT SQL: select hours_logged, miles_logged from timesheet where driverid= \${driverId} and week=\${week}
 - b. ENRICHMENT OUTPUT FIELDS: driverFatigueByHours, driverFatigueByMiles
 - c. SECURE CLUSTER: false
 - d. OUTPUT FIELDS: Select the splitJoinValue field.
 - e. NEW OUTPUT FIELDS: Add new output fields for the two enriched values driverFatigueByHours and driverFatigueByMiles.

×

EN	RI	Cŀ	1-1	Гiт	iesł	nee	t
							_

CONFIGURATION	NOTES					
Input						Output
eventTime*			splitJoinValue*			
eventTimeLong*		KERBEROS KEY TAB FILE		driverFatigueByHours* STRING		
eventSource*						driverFatigueByMiles*
truckld* INTEGER		OUTPUT FIELDS			SELECT ALL	
driverId*		× splitJoinValue			× -	
driverName*		NEW OUTPUT FIELDS +				
routeld*						
route* string		Field Name	Туре	Optional	Actions	
eventType* string		driverFatigueByHours	STRING	false	e 🖉	
latitude*		driverFatigueByMiles	STRING	false	/ 1	
longitude*						
						Cancel Ok

After this processor executes, the output schema will have two fields populated called driverFatigueByHours and driverFatigueByMiles.

5. Create the third stream to do weather enrichment by dragging the custom processor you uploaded called "ENRICH-WEATHER" processor to the canvas and connect it from the Split project processor.

5	My Applic	ations / IOT-Trucking-Ref-App						
4	Q 🕼				Last C	hange:0s ago V	ersion:CURRENT	@ Q \$
æ	EVENT HUBS					₽• <u></u> ;	NOTIFICATI 4 01 NOTIFICATION	
đ	CP HDFS			DriverAvgS	€01 ► • • • • • • • • • • • • • • • • • •	ON ≪01▶●		
Ŷ	& KAFKA	Se TruckGeoEV ∉ 01 ▶ ● KAFKA	∢ 01 ▶ •──●● 🐑 EventType RULE	4 01 ▶ ••• ⊖ Violation	401	-	Dashboard	
r	PROCESSOR	KAFKA KAFKA						
				HBASE	401 ▶			
	×°			Data-Lake	401▶			
	>+				ENRICH-HR			
	JOIN			Split PROJECTION	4 01 ▶ ● ■ ENRICH-Tim 4 01 ▶ ● ENRICH-PHOENIX			
					ENRICH-WEATHER			
	PROJECTION							

- **6.** Configure the weather process with the following parameters (currently the weather processor is just a stub that generates random normalized weather info).
 - **a.** WEATHER WEB SERVICE URL: http://weather.com/api?lat=\${latitude}&lng=\${longitude}
 - **b.** INPUT SCHEMA MAPPINGS: Leave defaults
 - c. OUTPUT FIELDS: Select the splitJoinValue and the three model enriched features

ENRICH-WEATHER				3
CONFIGURATION NOT	TES			
Input	WEATHER WEB SERVIC	E URL *		Output
eventTime*	http://weather.com	m/api?lat=\${latitude}&lng=\${long	gitude}	splitJoinValue*
eventTimeLong*	INPUT SCHEMA MAPP	ING		Model_Feature_FoggyWeather
eventSource* STRING	driverId	driverId	× 👻	Model_Feature_RainyWeather
truckId*	latitude	latitude	× -	Model_Feature_WindyWeathe
driverId*	la se de			
driverName* string	longitude	longitude	X 👻	
routeld*	OUTPUT FIELDS*		SELECT ALL	
route* string	× splitJoinValue	× Model_Feature_FoggyWeather	× -	
eventType* string	× Model_Feature_Ra	ainyWeather Model_Feature_W	indyWeather	
latitude*				
longitude*				
				Cancel Ok

After this processor executes, the output schema will have three fields populated called Model_Feature_FoggyWeather, Model_Feature_RainyWeather, and Model_Feature_WindyWeather.

Steps for Rejoining the Three Enriched Streams

1. Now that you have done the enrichment in parallel by splitting the stream into three, you can now join the three streams by dragging the join processor to the canvas and connecting the join from the three streams.

5	My Applic	cations / IOT-Trucking-Ref-App					
т	Q 🕼				Last Ch	ange:0s ago Version:CURRENT	ତ୍ତ୍ 🗙
æ	EVENT HUBS					NOTIFICATION 401	
đ	60 HDFS			► ∑ DriverAvgS	4 01 Demok StriverSp 4 01 Demok Round Projection	≪01⊁●	
Ø	%	& TruckGeoEv ∉ 01 ▶ • KAFKA	∢01 ▶ • → • 🐑 EventType	 € • • • • • • • • • • • • • • • • • • •	401	Dashboard 401	
ŗ	PROCESSOR	& TruckSpeed ∢01 ►					
				HBASE	≪ 01 ≥		
	BRANCH			Data-Lake	≪01 ▶		
	>				ENRICH-PHOENIX 401		
				Split PROJECTION	4 01 P SINCH-TIM 4 01 P JOIN-1 JOIN	401⊁●	
	PMML				THE ENRICH-WEA. doub		
	PROJECTION				ENRICH-WEATHER		

Configure the join processor like the following where you use the joinSplitValue to join all three streams.
 For the Output field, click SELECT ALL to select all the fields across the three streams.

JOIN-ENRICHMENTS					×
CONFIGURATION NOTES					
	SELECT STREAM*	SELECT FIELD WITH*			Output eventTime*
splitJoinValue*	JOIN TYPE*	SELECT STREAM*	SELECT FIELD*	WITH STREAM*	eventSource* string truckld*
Model_Feature_FoggyWeather* DOUBLE Model_Feature_RainyWeather*	INNER -	custom_processor_strem	splitJoinValue	custom_processor_stre.#	INTEGER driverId* INTEGER
Model_Feature_WindyWeather*	INNER -	custom_processor_strem	splitJoinValue	custom_processor_stre.	oriverName* STRING routeId* INTEGER
	Processing Time	•			route* string eventType*
	WINDOW INTERVAL*		SLIDING INTERVAL		STRING latitude* DOUBLE
	4	Seconds -	4	Seconds -	longitude*
			,	SELECT ALL	correlationId*

3. Now that you have joined three enriched streams, normalize the data into the format that the model expects by dragging the "NORMALIZE-MODEL-FEATURES" custom processor that you added to the canvas.

For the output fields, select all the fields and leave the mapping as defaults.

ONFIGURATION NO	DTES	
eventTime* STRING eventSource*	TIMEOUT DELAY FOR MONITORING USE CASE (SECONDS)	Output eventTime* strang eventSource*
STRING truckld* INTEGER driverld* INTEGER driverName* STRING routeld* INTEGER route* STRING eventType*	× eventTime × eventSource × truckId × driverId × driverName × routeld × route × eventType × latitude × longitude × correlationId × geoAddress × speed × splitJoinValue × week × driverCertification × driverWagePlan × driverFatigueByHours × driverFatigueByMiles × × Model_Feature_FoggyWeather × Model_Feature_RainyWeather × Model_Feature_WindyWeather × eventTimeLong	STRING truckId* INTEGER driverId* INTEGER driverName* STRING routeId* INTEGER routed* STRING eventType*
STRING latitude* DOUBLE longitude* DOUBLE correlationId*	× Model_Feature_Certification × Model_Feature_WagePlan × Model_Feature_FatigueByHours × Model_Feature_FatigueByMiles	STRING latitude* DOUBLE longitude* DOUBLE correlationId*

Result

Your flow looks similar to the following.



Score the Model Using the PMML Processor and Alert

Now you are ready to score the logistical regression model.

Procedure

1. Drag the PMML processor to the canvas and connect it to the Normalize processor.



2. Configure the PMML processor like the following by selecting the DriverViolationPredictionModel that you uploaded earlier to the **Model Registry**.

After this processor executes, a new field called **ViolationPredicted** is added to stream for the result of the prediction. In output fields, select all the contextual fields you want to pass on including the model value result.

Predict		×
CONFIGURATION NOTES		
Input eventTime* string eventSource* string truckld* integer driverld* integer driverName* string routeld* integer routeld*	MODEL NAME* DriverViolationPredictionModel OUTPUT FIELDS* SELECT ALL * eventTime * eventSource * truckId * driverId * driverName * routeId * route * latitude * longitude * geoAddress * speed	Output ViolationPredicted* strang eventTime* strang eventSource* strang truckId* INTEGER driverName* strang routeId* INTEGER
eventType* sTRING latitude* DOUBLE correlationId* LONG		route* strand latitude* DOUBLE longitude* DOUBLE geoAddress* strand Cancel Ok

3. Determine if the model predicted if the driver will commit a violation by dragging a rule processor to the canvas and configuring a rule like the following:

Edit Rule				×
RULE NAME*				
Violation Predict	ed			
DESCRIPTION*				
model returned a	prediction			h
CREATE QUERY*	ViolationPredicted	× 👻 EQUALS	× 👻 'yes'	× 👻 🕇
QUERY PREVIEW:				
ViolationPredi	cted = 'yes'			
				Cancel Ok

- **4.** If a violation is predicted, send it to a Druid to display on a dashboard. Drag the Druid processor to canvas and configure.
- 5. Stream the events into a cube called **alerts-violation-predictions-cube**.

Dashboard-Predictions	3 X
REQUIRED OPTIONAL N	OTES
Input	DATASOURCE NAME *
ViolationPredicted*	alerts-violation-predictions-cube
eventTime* string	ZOOKEEPER CONNECT STRING *
eventSource* string	secure-sam-hdf2.field.hortonworks.com:2181,secure-sar
truckld* INTEGER	DIMENSIONS *
driverld* INTEGER	× ViolationPredicted × eventTime
driverName* string	× eventSource × truckId × driverId
routeld*	× driverName × routeld × route × -
route* STRING	× eventType × latitude × longitude
eventType* string	georialized speed
latitude*	TIMESTAMP FIELD NAME *
longitude*	processingTime -
	Cancel Ok

Results

The final flow looks like the following:



Creating Visualizations Using Superset

A business analyst can create a wide array of visualizations to gather insights on streaming data. The platform supports over 30+ visualizations the business analyst can create. For visualization examples, see the Gallery of Superset Visualizations.

The general process for creating and viewing visualizations is as follows:

- 1. Whenever you add new data sources to Druid via a Stream App, perform the Refresh Druid Metadata action on the **Superset** menu.
- 2. Using the Superset Stream Insight UI, create one or more "slices". A slice is one business visualization associated with a data source (for example, Druid cube).
- 3. Using the Dashboard menu, add the slices to your dashboard and organize their layout.



When a SAM app streams data to a new cube using the Druid processor, it will take about 30 minutes for the cube to appear in Superset. This is because Superset has to wait for the first segment to be created in Druid. After the cube appears, users can analyze the streaming data immediately as it is streaming in.

Creating Insight Slices

Note:

The following steps demonstrate a typical flow for creating a slice:

Procedure

- 1. Choose Slices on the Menu.
- 2. Click + to create a new Slice.
- 3. Select the Druid Data Source that you want to use for the new visualization:

rse	nt of Secur	ity 🗸 🥻 Manag	e 🗸 🐻 Sources 🗸 🕍	Slices 🚯 Dashboards 👗 SQL Lab	*	0
c	lick on a druid I	ink to create a Slice				
L	ist Druid Data	source				
S	earch 🛩					
I	Actions	× •			Record	Count: 2
		Data Source 1	Cluster	Changed By	Changed On I	Time Offset
	Q 2 #	Alerts-High- Speed-Cube-V2	Streaming Analytics Manager - Stream Insight	George Vetticaden</a 	2017-02-07 15:50:59.807995	0

4. Select a Chart Type from the menu.

The following example creates a "Sunburst" visualization of rolling up multiple dimensions like route, eventType, and driver info..

Configure the chart and click Execute Query

Query Save as Datasource & Chart Type	Violation Details Brea	kdown ☆œ	0.38 sec 9
	Laws Demonstrate	Des Maises to Okiases	7.50%
[druid-ambari].[violation-events-cube] *	Lane Departure	Des moines to Chicago	7.58%
Sunburst *			
Time 🕐			
Time Granularity Origin ® 1 hour * Since ® Until 7 days ago now Hierarchy ® ¥ route ¥ eventType Primary Metric ®	7	7.58% of tota 32.3% of parent m1: 220	al
COUNT(*) *			
Secondary Metric ®			
COUNT(*) *			
Row limit			
5000 *			
Filters ⑦			
+ Add filter			

5. Another visualization could be integration with MapBox Here we are mapping where violations are occurring the most based on the lat/long location of the event

Superset of Security v F Manage v	≣ Sources ✔ 🕍 Slices 🍙 Dashboards 👗 SQL Lab ✔	∩ <i>⊪</i> ≛∖
Query O Save as	Route Violations Map 👷	9 🗣 🚸 🔯 json 🖻 .csv Query
Datasource & Chart Type	Rochester	
[druid-ambari].[violation-events-cube] *	Sigur Fills	Toronto
Mapbox *		ng Detroit Butte
	NEBRASKA Des Moines 7 12 10 Chicago	
Time 🕲	North Platte S	PENN
Time Granularity 🐑 🛛 Origin 🟵		
1 hour * *	9ULLINOIS INDIANA	Columbus
Since 🏵 Until	United States12	
7 days ago * now *	16 Linuis	WEST
	KANSAS MISSOURI 4 20 Frenklar	t VIRGINIA
Longitude 1 Latitude 1	Weta KENTINGY	VIRI
longitude + latitude +	Spring 6:	for the contraction
Clustering Radius ®	Ciarkeville	
60 *	OKLAHOMA 10 Fayetteville TENNESSEE	Knowline Contraction
Row limit	Oklahoma City 11	Charlotte CAROLIN.
5000 v	ARKANSAS Memphis	FRANK (
Group by ®		
X latitude X longitude	a . The All	enta SOUTH CAROLINA
Live render 🕐 🖉	· · · · · · · · · · · · · · · · · · ·	V La Xor
	Abilene Dallas MISSISSIPPI ALABAMA	N VY
Viewport		
Default longitude ®		
-90.1994		
Default latitude ®		
38.627		
Zoom 🕲		
5		

6. To save the slice, specify a name and name and click Save.

Save as Driver V	fiolations Break	
Do not add to a d	lashboard	
⊖Add slice to exist	ting dashboard *	
⊖Add to new dash	board [dashboard name]	

Adding Insight Slices to a Dashboard

After you create slices, you can organize them into a dashboards:

Procedure

0

- 1. Click the **Dashboard** menu item.
- **2.** Click + to create a new Dashboard.
- 3. Configure the dashboard: specify a name and the slices to include in the Dashboard.

Add Dasł	iboard
Title	Trucking IOT Dashboard
Slug	trucking-iot-dashboard
	To get a readable URL for your dashboard
Slices	* Total Violations in Last Hour
	* Top Violation Drivers
	# Driver Violations Breakdown
	R Direction Infraction Details
	Routes with Infractions
Owners	M George Vetticaden
	Owners is a list of users who can alter the dashboard.
Position JSON	Position JSON
	This json object describes the positioning of the widgets in the dashboard. It is dynamically generated when adjusting the widgets size
	positions by using drag & drop in the dashboard view
CSS	CSS
	The css for individual dashboards can be altered here, or in the dashboard view where changes are immediately visible
JSON Metadata	JSON Metadata
	This JSON object is generated dynamically when clicking the save or overwrite button in the dashboard view. It is exposed here for reference and for power users who may want to alter specific parameters.

4. Arrange the slices on the dashboard as desired, and then click Save.

Dashboards for the Trucking IOT App

The IOT Trucking application that we implemented using the Stream Builder streams violation events, alerts, and predictions into three cubes:

- violation-events-cube
- alerts-speeding-drivers-cube
- alerts-violation-predictions-cube

Based on the powerful visualizations that SuperSet offers, you can create the following powerful dashboards in minutes.

IoT Dashboard

🍽 Superset of Security 🗸 🥕 Manage 🗸 🛢 Sources 🗸 🕍 Slices 🍙 Dashboards 🕹 SQL Lab 🗸			n <i>e</i> a
Trucking IOT Dashboard 🌣		٥	* 0 T 3 8 5
Vielations is Last Hour	Tap Boutes with Vielations		
22.0	Saint L Springtiet	Saint Louis to Tr OUIS to Chicago Rou d to KC Via Hanibal Route 2 Saint Louis to Chicago _{Saint}	ulsa Jte2 nots ^{to tutsa} notech source on tutsa notech source on the source of the s
Vicialize Notelle Banddowe	Dente Detaile		
Violation Details Breakdown	Route Details	a such	
	eventType	Des Moines to Chicago	count v
	Unsafe tail distance	Des Moines to Chicago	182
	Unsafe following distance	Des Moines to Chicago	172
	Lane Departure	Des Moines to Chicago	168
	Unsafe tail distance	Saint Louis to Memphis	109
	Lane Departure	Saint Louis to Memphis	109
	Unsafe following distance	Saint Louis to Memphis	108
	Overspeed	Saint Louis to Memphis	94.0
	Overspeed Upsafe tail distance	Saint Louis to Chicago Route2	67.0
	Unsafe following distance	Saint Louis to Chicago Route2 Saint Louis to Chicago Route2	65.0
	Unsafe tail distance	Des Moines to Chicago Route 2	52.0
	Lane Departure	Des Moines to Chicago Route 2	51.0
	Unsafe following distance	Des Moines to Chicago Route 2	48.0
	Unsafe tail distance	Memphis to Little Rock Route 2	47.0
	Overspeed	Des Moines to Chicago Route 2	47.0
	Lane Departure	Saint Louis to Chicago Route2	45.0
	Unsafe following distance	Memphis to Little Rock Route 2	44.0
	Lane Departure	Memphis to Little Rock Route 2	41.0
	Overspeed Upsafe following distance	Memphis to Little Rock Route 2 Decrip to Coder Panida	39.0
Route Violations Map Semiciar UTAR Draw Juncis Col.08A00 Dorme	3 Chage ILLINDIS INDIANA O Louis Deventia Communic Commun	PENNSYLVANIA Original Marina M	colin, R.
Routes with Infractions Over Time Cost Mores to Chag_ Cost More to Chag_ Cost Mores to Chag_ Cost More to Chag_	Little Ro @ Peoria to Ceder Rapt @ Peoria to x KC VI @ Wehita to Little Ro @ Wehita to	Ceder Ragi Start Louis to Chica Start Louis to Chica Start Louis to Little Ra	Hemph. Start Louis to Memph.
1.00 Wes Jun 07, 03 PM Wes Jun 07, 05 PM Wed Jun 07, 09 PM Thu Jun 08	Thu Jun 08, 03 AM	Thu Jun 08, 06 AM Thu Jun 08, 09 AI	M Thu Jun 08. 1

Alerts Dashboard



SAM Test Mode

In a typical SDLC (Software development lifecycle), you want to test the streaming analytics app locally before deploying the SAM app to a cluster. SAM's "Test Mode" allows you to test the app locally using test data for the sources. SAM's Test Mode allows you to do the following:

- Create a Named Test Case
- Mock out the sources of the app and configure test data for each test source. SAM validates the test data using the configured Schema in the Schema Registry for each source
- Execute the Test Case and visually see how the data looks like at each component/processor in the app as flows across your application.
- Download the output of the test which represents the state of the data at each processor and sink.

In the following sections, we will walk through creating Test Cases for different test scenarios for the reference app. If you ran the test utility, these 4 test cases will already be created for you.

Four Test Cases using SAM's Test Mode

Test Case 1: Testing Normal Event with No Violation Prediction

The Assertions of this test case are the following:

- Assertion 1: Validate test data for geo steam and speed stream that are non violations
- Assertion 2: Validate the Join of data between geo stream and speed stream
- Assertion 3: Validate that the filter "EventType" detects that this is a "Non Violation Event"
- Assertion 4: Validate the joined event gets split into three events by the "Split" projection.
- Assertion 5: Validate that the three enrichments are applied: weather enrichments, timesheet enrichment and HR enrichment.
- Assertion 6: Validate the three enrichment streams are joined into a single stream.
- Assertion 7: Validate that data after normalization for the model
- Assertion 8: Validate the output of the Prediction model is that no violation is predicted
- Assertion 9: Validate the filter "Prediction" detects that it is non violation.

Follow the below steps to create this test case for the reference app in Edit Mode. (If you ran the test utility, these 4 test cases will already be created for you.)

1. Click "TEST" Mode



2. Click Add Test Case



3. Provide Test Case details. Provide a name for test case, test data for TruckGeoEvent and test data for TruckSpeedEvent.

IAME*				
Test-Normal-Event-No-Vi	olation-Predicti	on		
Sources	1 [2	{		Output
 Test-TruckSpeed Kafka Test-TruckGeoE Kafka 	3 4 5 6 7 8 9 10 11 12 13 14 15	<pre>"eventTime":"20 "eventSource":" "truckId":54, "driverId":23, "driverName":"3 "routeId":1, "route":"Peoria "eventType":"No "latitude":40.7 "longitude":-88 "correlationId" "geoAddress":"" }</pre>	<pre>117-09-26 14:54:32.64", truck_geo_event", Teff Markham", to Ceder Rapids Route 2" ormal", </pre>	eventTime* STRING eventSource* STRING truckld* INTEGER driverId* INTEGER routeld* INTEGER routed* STRING eventType* STRING Latiude*
	1 SLEEP TIME*		times	longitude* DOUBLE correlationId*

4. Execute the Test Case.



5. You should see the result of the test case as the following.



6. Download the test case results.



Analyzing Test Case 1 Results

The key to reading the test case results is to keep in mind that when you look at the results of the component, you are viewing the input into that component.

• Assertion 1 is to Validate test data for geo steam and speed stream that are non violations. For this assertion, you would look at the downstream component after the sources. So in this case, it would be the Join component. Use the paging features to see the inputs to the join processor.



• Assertion 2 is to validate the Join of data between geo stream and speed stream. For this assertion, you would look at the downstream component after the Join. So in this case, it would be the EventType component. Note that you see speed and geo information.

e Ever	ntType ◀ 01 ▶
route*	Peoria to Ceder Rap
eventType*	Normal
latitude*	40.7
longitude*	-89.52
correlation	1
geoAddres	
speed*	58

• Assertion 3 is to validate that the filter "EventType" detects that this is a "Non Violation Event". View the Split Component.

	C Split	ECTION				
••	eventTime*	2017-09-26 14:54:3				
	eventSour	truck_geo_event				
	truckId*	54 .				
	driverId*	23				
	driverNam	Jeff Markham				
	routeld*	1 Peoria to Ceder Rap				
	route*					

• Assertion 4 is to Validate test data for geo steam and speed stream that are non violations. View the JOIN-ENRICHMENT component

ENRICH-HR ENRICH-PHO 4 01		ENRICH-Tim • 01 •		ENRICH-WEA 01 ENRICH-WEA		
splitJoinV 1506 week* 39 eventTime* 2017 eventSour truck truckId* 54 driverId* 23 driverNam Jeff 1	437672640 -09-26 14:54:3: _geo_event Markham	splitJoinV week* eventTime* eventSour truckId* driverId* driverNam	1506437672640 39 2017-09-26 14:54:3: truck_geo_event 54 23 Jeff Markham	spli wee eve true driv driv	itJoinV ek* entTime* entSour ckId* verId* verNam	1506437672640 39 2017-09-26 14:54:3 truck_geo_event 54 23 Jeff Markham

• Assertion 5 is to validate that the three enrichments are applied: weather enrichments, timesheet enrichment and HR enrichment. Use the paging features to page through the three enrichment outputs.

→ JOIN-ENRIC ↓ 01 ►			→ JOIN-ENRIC ↓ 01 ►			→ JOIN-ENRIC ◀ 01 ►			
driverFatig driverFatig	48 2796		Model_Fea Model_Fea	0 0		truckld* week*	54 39		
splitJoinV	15064376726	40	Model_Fea splitJoinV	0 15064376726	540	driverCerti latitude*	Y 40.7	1	
						eventSour driverWag	truck_geo_even hours	nt	
« <	1 Of 3	> »	« <	2 Of 3	> »	eventTvpe*	Normal 3 Of 3	>	

• Assertion 6 is to validate the three enrichment streams are joined into a single stream. View the NORMALIZE component.



• Assertion 7 is to validate that data after normalization for the model. View the Predict component.
•

	Pred PMM	lict	€01 ▶
	week*	39	
	driverCerti	Y	
,	Model_Fea	0	
	driverFatig	48	- 1
	latitude*	40.7	- 1
	eventSour	truck_geo_ev	ent
•	driverWaa	hours	- 1

Assertion 8 is to validate that the output of the Prediction model is that no violation is predicted. View the Prediction component.

Prec	liction ∢01 ▶	1
ViolationP truckId* eventTime latitude* eventSour eventType* speed*	no 54 1506455684451 40.7 truck_geo_event Normal 58	•-
	Precedent Rules	Prediction RULE ViolationP No truckId* 54 eventTime 1506455684451 latitude* 40.7 eventSour truck_geo_event speed* 58

Test Case 2: Testing Normal Event with Yes Violation Prediction

In this test, we are validating all the same assertions as previous test but in this test case the violation prediction model should return true and be . Similar to above, Create a test named "Test-Normal-Event-Yes-Violation-Prediction", use the following test data for TruckGeoEvent and use the following test data for TruckSpeedEvent.

Analyzing Test Case 2 Results

1. Analyzing the Test Case Results.

The output of the test case should be the following:



To validate the PMML processor returns a violation prediction and sent to the sink, view the Prediction and Druid component.

● Prediction ↓ 01 ►	→ TEST-SINK
RULE	DRUID
ViolationPyestruckId*48eventTime1506455684451latitude*38.64eventSourtruck_geo_eventeventType*Normalspeed*60	ViolationPyestruckId*48eventTime1506455684451latitude*38.64eventSourtruck_geo_eventeventType*Normalspeed*60

Test Case 3: Testing Violation Event

The Assertions of this test case are the following:

- Assertion 1: Validate test data for geo steam and speed stream that are "violation" events
- Assertion 2: Validate the Join of data between geo stream and speed stream
- Assertion 3: Validate that the filter "EventType" detects that this is a "Violation Event"

- Assertion 4: Validate that the inputs to the aggregate speed processor. There should only be 1 in the window
- Assertion 5: Validate the result of the DriverAvgSpeed aggregate process is average speed of 79 since there only 1 event
- Assertion 6: Validate the isDriverSpeeding rule recognized it was not speeding since the speed wasn't greater than 80. The event should stop.

Create a test named "Test-Violation-Event", use the following test data for TruckGeoEvent and use the following test data for TruckSpeedEvent.

Analyzing Test Case 3 Results

The output of the test case should look something like the following:



• Assertion 3 is to validate that the filter "EventType" detects that this is a "Violation Event" and Assertion 4 is to validate that the inputs to the aggregate speed processor should be 1 event within the window. View the DriverAvSpeed component to validate these assertions:



• Assertion 5 is to validate the result of the DriverAvgSpeed aggregate process is average speed of 79 since there only 1 event. View the isDriverSpeeding component:

● isDri ● RULE	iverSp ∢ 01 ▶
speed_AVG*	79
driverId*	13
driverNam	Suresh Srinivas 🛛 💧
route*	Des Moines to Chic

• Assertion 6 is to validate the isDriverSpeeding rule recognized it was not speeding since the speed wasn't greater than 80. The event should stop. See the downstream components after isDriverSpeeding.

Round PROJECTION	101		€01
No Records		No Records	_

Test Case 4: Testing Multiple-Speeding-Events

The assertions of this test case are the following:

- Assertion 1: Validate that there are two geo events both of which are violations (Overspeed, Excessive Breaking) in source. Validate there are two speeding events both of which are speeding (96, 83)
- Assertion 2: Validate the Join of data between geo stream and speed streams
- Assertion 3: Validate that the filter "EventType" detects that this is a "Violation Event"
- Assertion 4: Validate the inputs of the window should be two events (geo/speed 1 with speed of 83, geo/speed 2 with speed of 96)
- Assertion 5: Validate the result of the DriverAvgSpeed aggregate processor should be one event that represents the average of 83 and 96...89.5
- Assertion 6: Validate the isDriverSpeeding rule recognizes it as speeding event (89.5) since it is greater than 80 and continue that event to custom round UDF
- Assertion 7: Validate the output of the round UDF event should change the speed from 89.5 to 90.0 and that is the final event that goes to the sink.

Create a test named "Test-Multiple-Speeding-Events", use the following test data for TruckGeoEvent and use the following test data for TruckSpeedEvent.

Analyzing Test Case 4 Results

The output of the test case should look something like the following:

577	My Applica	ations / streaming-ref-app-ad	vanced_AUTOCREATED						
4	TEST CASES					Last Change:2h 46m 42s a	ago Version:CURRENT Mode:		•
88 @ \$	Later Specing Events AUDOREATED Test Violation Events AUDOREATED AUDOREATED Violation- Preservers Violation- Reververs Violation- Violation- Violation- Reververs Violation- Violat	TEST-SOURCE (01) trate trate	→ JOIN 401 muth 23 muth 24 muth 24 muth 24 muth 24 muth 25 muth 24 muth 25 muth 25	EventType {01} RILE eventTwat 2017092615455 eventSize toxks.gec.tvent oxida 22 divate 13	DivertAvgS 401 Automatical exertinary tracksportation matter 23 divertinary 23 divertinary 23 divertinary 24 divertinary 24 divertinary 24 divertinary 24 divertinary 25 divertinary 24 divertinary 25 divertinary 2	BDTVerSp (01) INSE Send, AVX 89.5 Short 13 Note: Des Menes to Cho Des Menes to Cho Note: C	Round ←01 → □ TEST- PROJECTION +01 → □ CRAID Send Lave H5 sender 1 Sender 1 Sender 1 Sender 4 Sender 4	SINK 4 01 b 0 a aurech Strinnas en Monres to Chic	
		abundt 13 4 10/2 20 ESERCOUNCE { 01 > 20 ESERCOUNCE { 01 > 21 ESERCOUNCE { 01 > 22 ESERCOUNCE { 01 > 24 DE Mones to Diac 40447 31	eventTime* 2017/20/2018/45/5 eventTime* tuck.peed.vent diventum. Streets Streivas < < 10/4 >>>	divertua Suretà Sinivas moteté 4 note De Mones to Chic ≪ C 10/4 > 3	No Records	No Records	JOIN-ENRIC 401 Mo Records	NORMALIZE	
								No Records	
								~	

• Assertion 4 is to validate the inputs of the window should be two events (geo/speed 1 with speed of 83, geo/speed 2 with speed of 96). View the two events in the Join processor (use the paging feature to see the events)

∑ Driv AGGR	erAvgS	DriverAvgS . AGGREGATE	·· ∢01▶
eventType*	Lane Departure	eventType* Lane Dep	oarture
latitude*	41.62	latitude* 41.62	
longitude*	-93.58	longitude* -93.58	
correlation	1	correlation 1	l li
geoAddres		geoAddres	
speed*	83	speed* 96	
eventTime	1506455684451	eventTime 1506455	684451
« <	1 Of 2 > >>	« < 2 Of 4	> »

• Assertion 5 is to validate the result of the DriverAvgSpeed aggregate processor should be one event that represents the average of 83 and 96...89.5. View the Round Projection processor.



Assertion 6 is to validate the isDriverSpeeding rule recognizes it as speeding event (89.5) since it is greater than 80 and continue that event to custom round UDF. View the Round Projection processor



• Assertion 7 is to validate the output of the round UDF event should change the speed from 89.5 to 90.0 and that is the final event that goes to the sink. View the Druid Test Sink component.



Running SAM Test Cases as Junit Tests in CI Pipelines

Using SAM's Test Mode provides a quick and effective way to test your applications locally visualizing the output within each component of the app without deploying to a cluster. Since all of SAM's capabilities is backed by REST APIs, you can execute the these SAM Test Cases as part of your Junit Tests. This provides the power of using Junit assertions to validate the results of the test and incorporating them in automated tests as part of your continuous integration and delivery pipelines.

Examples of incorporating SAM Test Cases as part of unit tests can be found in the following artifacts:

- Trucking Ref App Git Hub Project
- TruckingRefAppAdvancedApp Junit Test Case

The Junit Test case above uses the SAM SDK project to setup a self contained Junit test that executes the SAM test cases and validates the result. The test case performs the following on setup of the this Junit Test Case:

- Create SAM Service Pool
- Create SAM Environment
- Import the Trucking Ref App

Then the following 4 test cases are executed:

- testNormalEventNoViolationPrediction
- testNormalEventYesViolationPrediction
- testViolationTruckingEvents
- testMultipleSpeedingEvents

Each of these test cases will do the following:

- Create the SAM Test Case
- Setup test data for each of the sources for each test case.
- Execute the SAM Test Case using SAM Test Mode and wait for test to complete.
- Download the results of the test case.

• Validate the results of the Test Case.

SAM Test Mode execution via Junit Tests allows you to integrate these tests as part of your continuous integration / delivery pipeline.



没 Jenkins	3 Q, search	?	George Vetticader	n ∣log out
Jenkins → SAM Ref App Pipeline → #35 → Test	Results > hortonworks.hdf.sam.refapp.trucking.app > TruckingRefAdvancedAppTest		ENABLE	AUTO REFRESH
Filstory Open Blue Ocean	Test Result : TruckingRefAdvancedAppTest			
Oit Build Data No Tags Docker Fingerprints			Za	4 tests (±0) Took 16 min. dd description
Test Result	All Tests			
한 Replay	Test name		Duration	Status
Pipeline Steps	testMultipleSpeedingEvents		4 min 0 sec	Passed
🐗 Previous Build	testNormalEventNoViolationPrediction		4 min 0 sec	Passed
Next Build	testNormalEventYesViolationPrediction		4 min 0 sec	Failed
- Hore Build	testViolationTruckingEvents		4 min 0 sec	Passed

Creating Custom Sources and Sinks

Throughout the getting started doc with the trucking reference application, we have showcased the powerful extensibility features of SAM including:

- Uploading custom UDFs
- Uploading custom processors

In this section, we walk through how to register custom sources and sinks in SAM integrated with Schema Registry.

Cloud Use Case: Integration with AWS Kinesis and S3

To showcase registering custom sources and sink, lets modify our Trucking Reference Application Use Case Requirements with the following:

- The Trucking company wants to deploy the Trucking Application on AWS
- The Trucking company wants to streams the sensor data into AWs Kinesis instead of Apache Kafka.
- The trucking company wants to use SAM for streaming analytics.
- The insights generated by SAM should be stored into AWS S3 instead of Druid.

The below diagram illustrates this Cloud architecture:



Registering a Custom Source in SAM for AWS Kinesis

To register any custom source in SAM, there are three artifacts you need:

- 1. Artifact 1: Code for the custom source using the underlying streaming engine. Since SAM today supports Storm as the Streaming engine, you can refer to the following artifacts for the custom source:
 - Git Project for Storm Kinesis
 - AWS Storm Kinesis Spout
- **2.** Artifact 2: Code for mapping the SAM configs to the custom source/spout. Refer to the following artifacts for this mapping code:
 - Git Project for SAM Storm Kinesis Mapping
 - SAM Kinesis Flux Mapping Class
- 3. Artifact 3: Flux mapping file to map the SAM config to the Kinesis Spout. Refer to the following artifacts
 - SAM Kinesis Flux Mapping Config

More Details on implementing a custom source and registering with SAM can be found here: https://github.com/ hortonworks/streamline/tree/master/examples/sources

To register the custom Kinesis Source in SAM using the above three artifacts, perform the following steps:

- 1. Download the Sam-Custom-Extensions.zip to the host where SAM is installed (if you haven't done it in a past step)
- 2. Unzip the contents. We will call the unzipped folder \$SAM_EXTENSIONS
- **3.** Switch to user streamline:

sudo su streamline

4. Install Artifact 1 (the custom source code) on host's local maven repo

```
cd $SAM_EXTENSIONS/custom-source/kinesis/
mvn install:install-file -Dfile=storm-kinesis-1.1.0.5.jar \
-DgroupId=org.apache.storm \
-DartifactId=storm-kinesis \
-Dversion=1.1.0.5 \
-Dpackaging=jar
```

5. Register the custom source via SAM REST call. Replace SAM_HOST and SAM_PORT.

```
curl -sS -X POST -i -F \
topologyComponentBundle=@config/kinesis-source-topology-component.json -F
\
bundleJar=@sam-custom-source-kinesis.jar \
http://SAM_HOST:SAM_PORT/api/v1/catalog/streams/componentbundles/SOURCE
```

6. If the registration was successful, you should see a message like the following by the REST response:

```
HTTP/1.1 201 Created
Date: Wed, 03 Jan 2018 20:26:22 GMT
Content-Type: application/json
Content-Length: 4569
```

7. On the SAM Application Canvas Palette, you should now see KINESIS source.



8. Dragging the kinesis source onto the canvas and double clicking it, you should see the following kinesis dialog. The dialog properties comes from the topologyComponentBundle flux config you used to register the custom source.

KINESIS 🗸 🗙	×
REQUIRED OPTIONAL NOTES	
AWS ACCESS KEY ID * AWS ACCESS KEY SECRET * AWS REGION * US_WEST_2 KINESIS STREAM * Truck_events_avro READER SCHEMA VERSION * 1 SHARD ITERATOR TYPE *	Cutput eventTime straws eventSource straws truckId intrecer driverId intrecer driverName straws routeId intrecer straws routeId intrecer straws coute straws
	Cancel Ok

Registering a Custom Sink in SAM for AWS S3

To register any custom sink in SAM, there are three artifacts you need:

- 1. Artifact 1: Code for the custom sink using the underlying streaming engine. Since SAM today supports Storm as the Streaming engine, you can refer to the following artifacts for the custom sink:
 - Git Project for Storm S3
 - Storm S3 Sink
- **2.** Artifact 2: Code for mapping the SAM configs to the custom/spout. Refer to the following artifacts for this mapping code:
 - Git Project for SAM Storm S3 Mapping
 - SAM S3 Flux Mapping Class
- 3. Artifact 3: Flux mapping file to map the SAM config to the S3 Sink. Refer to the following artifacts
 - SAM S3 Flux Mapping Config

To register the custom S3 Sink in SAM using the above three artifacts, perform the following steps:

- 1. Download the Sam-Custom-Extensions.zip to the host where SAM is installed (if you haven't done it in a past step).
- 2. Unzip the contents. We will call the unzipped folder \$SAM_EXTENSIONS.

3. Switch to user streamline.

sudo su streamline

4. Install Artifact 1 (the custom sink/bolt code) on host's local maven repo.

```
cd $SAM_EXTENSIONS/custom-sink/s3
mvn install:install-file \
    -Dfile=storm-s3-0.0.1-SNAPSHOT.jar \
    -DgroupId=hortonworks.storm.aws \
    -DartifactId=storm-s3 \
    -Dversion=0.0.1-SNAPSHOT \
    -Dpackaging=jar
```

5. Register the custom sink via SAM REST call. Replace SAM_HOST and SAM_PORT.

```
curl -sS -X POST -i -F \
topologyComponentBundle=@config/s3-sink-topology-component.json -F \
bundleJar=@sam-custom-sink-s3.jar \
http://SAM_HOST:SAM_PORT/api/v1/catalog/streams/componentbundles/SINK
```

6. On the SAP App Canvas Palette, you should now see S3 sink.



7. Dragging the S3 sink onto the canvas and double clicking it, you should see the following s3 dialog. The dialog properties comes from the topologyComponentBundle flux config you used to register the custom sink.

\$3	×
REQUIRED OPTIONAL	NOTES
Input eventTime* straws eventTimeLong* Lows eventSource* straws truckid* att	AWS ACCESS KEY ID *
	Cancel Ok

Implementing the SAM App with Kinesis Source and S3 Sink

Now that we have registered teh custom Kinesis and S3 sources and sink, we can now build the streaming application in SAM to implement the cloud use case requirements.

Procedure

1. Clone the trucking reference application.



- 2. Rename the clone app to streaming-ref-app-advanced-cloud
- 3. Delete the Kafka sources and druid sinks from the SAM App
- 4. Add Kinesis sources for the deleted the Kafka Topics. Make sure to create the create the kinesis streams in AWS with the same names as the schemas you defined SAM's SR. You you will need to reevaluate the config for other components that are marked as yellow.

uckGeoEvent	× TruckSpeedEvents
QUIRED OPTIONAL NOTES	REQUIRED OPTIONAL NOTES
AS REGION • Cutput JS_WEST_2 Cutotic for the second secon	AWS REGION * US_WEST_2 US_WEST_2 KINESIS STREAM * Cee* READER SCHEMA VERSION * Select SHARD ITERATOR TYPE * LATEST SCHEMA REGISTRY URL * SCHEMA REGISTR

- **5.** Add S3 sink for the deleted druid sinks. Make sure to create the S3 buckets in AWS. If you can't connect to the S3 from the Round projection, try deleting the Round projection, adding it back in and then connecting it to the S3.
- 6. Remove any HDFS or HBase Sinks that you have in the app.

Results

The SAM App should look like the following:



Stream Operations

The Stream Operation view provides management of the stream applications, including the following:

- Application life cycle management: start, stop, edit, delete
- Application performance metrics
- Troubleshooting, debugging
- Exporting and importing applications

My Applications View

Once a stream application has been deployed, the Stream Operations displays operational views of the application.

One of these views is called My Application dashboard.

To access the application dashboard in SAM, click **My Application** tab (the hierarchy icon). The dashboard displays all applications built using Streaming Analytics Manager.

Each stream application is represented by an application tile. Hovering over the application tile displays status, metrics, and actions you can perform on the stream application.



Application Performance Monitoring

To view application performance metrics (APM) for the application, click the application name on the application tile.

The following diagram describes elements of the APM view.



Exporting and Importing Stream Applications

Service pool and environment abstractions combined with import and export capabilities allow you to move a stream application from one environment to another. This task provides instructions about importing a stream application that was exported in JSON format.

About this task

To export a stream application, click the Export icon on the **My Application** dashboard. This downloads a JSON file that represents your streaming application.



Procedure

1. Click on the + icon in My Applications View and select import application:



2. Select the JSON file that you want to import, provide a unique name for the application, and specify which environment to use.

Import Stream	×
SELECT JSON FILE *	
Choose File Trucking-IOT-Streaming-Analtyics.json	
TOPOLOGY NAME	
Trucking-IOT-Streaming-Analtics-App-Import	
ENVIRONMENT *	
Dev	~
	Cancel Ok

Troubleshooting and Debugging a Stream Application

Once we have deployed the streaming app, common actions performed by users such as DevOps, Developers, and Operations teams are the following:

- Monitoring the Application and troubleshooting and identifying performance issues
- Troubleshooting an application through Log Search
- Troubleshooting an application through Sampling

SAM makes performing these tasks easier by using the same visual approach as users have when developing the application. We will walk through these common use cases in the below sections.

Monitoring SAM Apps and Identifying Performance Issues

After deploying SAM and running the test generator for about 30 mins, your Storm Operation Mode of the app renders important metrics within each component on the canvas like below.

All Components Log: None Sampling: 0% •	Model CVERVERV METRICS SAMPLE	🧇 💿 10 minutes - 🧧 🖉
See Overview Metrics Directly on the SAM App within each Component	District/NGS (c)} (c) (c)	Q Q
A DOAL And Alase Sh. 242-0 2.5 Some of that Alase Sh. 242-0 2.5 Some of the Alase Sh. 252-0 2.5 Some of the Alase So	Control Transmit Datase Alla Alla Alla Alla Alla Alla Alla All	(01) ************************************
	ENRICH-INVEAL Control Provent Description	Dashboard 401 b D

You can click on **Show Metrics** to get more details on the metrics and drill down on individual metrics. Note the detailed level metrics for **All Components**, **TruckGeoEvent Kafka** source, and **Dashboard-Predictions** Druid Sink.

All Components -	Emitted 🎽 94k -0.0k	Acked ≌ 99k -0.0m	Latency 7 34.0sec +15.4s	Failed ec 0 0	Workers 3	Executors 19	Hide Metrics 🗸
Input/Output							
Acked Tuples							
Failed Tuples							
Queue							
Latency	5						

All Components -	Emitted 🎽 94k -0.0k	Acked 🎽 99k -0.0m	Latency 7 34.0sec +15.4sec	Failed 00	Workers 3	Executors 19	Hide Metrics 🗸
Input/Output							
Acked Tuples							
Failed Tuples							
Queue							
Latency	5						

Dashboard-Predictions - Druid	Emitted № Acked № 730.0 -290.0 350.0 -160.0 Workers Executors 3 19	Process Latency 9.4ms -508.7ms	Execute Latency 凶 1.5ms −1.4ms Hid	Failed 0 o de Metrics ❤
Input/Output				
Acked Tuples				
Failed Tuples				
Queue				
Process Latency				
Execute Latency				

Key metrics include the following:

Metric Name	Description	
Execute Latency	The average time it takes an event to be processed by a given component	
Process Latency	The average time it takes an event to be acked. Bolts that join, aggregate or batch may not Ack a tuple until a number of other Tuple have been received	
Complete Latency	How much time an event from source takes to be fully processed and acked by the topology. This metrics is only available for sources (e.g.: Kafka Source)	
Emitted	The number of events emitted for the given time period. For example for a Kafka Source, it is the number of events consumed for the give time period	
Acked	The number of events acked for the given time period. For example, for a Kafka Source, it is the number of events consumed and then acked.	

Identifying Throughput Bottlenecks

Looking through the metrics the Source and Sink metrics, we want to increase the throughput such that we emit/ consume more events from the Kafka Topic and send more events to Druid sink over time. We make some changes to the app to increase throughput.

Increase the parallelism of TruckGeoEvent (kafka topic: truck_events_avro) and TruckSpeedEvent (kafka topic: truck_speed_events_avro) from 1 to 3. Note that each of these kafka topics have three partitions.



Increase the parallelism of the Join from 1 to 3. Since the join is grouped by driverId, we can configure the connection to use fields grouping to send all events with driverId to the same instance of the Join.

KAFKA	Configure each connection to do a gro by driverld so that a events with the sam driverld go to the san instance of the Join	Sup II e ne 1 GROUPING*	
		FIELDS	-
		SELECT FIELDS*	
	-	× driverId	× -
KAFKA			

Increase the parallelism of the DriverAvgSpeed aggregate window from 1 to 3. Since the window groups by driverId,driverName and route, we can configure the connection to use fields grouping to send all events with those field values to the same instance of the window.

	- DriverAvgS 4 03 P-		
		GROUPING*	
		FIELDS	-
		SELECT FIELDS*	
-▶● 💬 EventType 🔹 🕫 👘	-	× driverld × driverName × route	× -

Increase the parallelism of the Dashboard-Predictions Druid sink from 1 to 3 so we can have multiple JVM instances of Druid writing to the cube.



After making these changes, we re-deploy the app using SAM and run the data generator for about 15 minutes and view seeing the following metrics.

SAM's overview and detailed metrics makes it very easy to verify if the performance changes we made had the desired effect.

My Applications / View: streaming-ref-app-advanced_AUTOCREATED		
All Components Log: None Sampling: 0% -	Mode: OVERVIEW METRICS SAMPLE	🤧 💿 30 minutes - 😰 🖉
3e Tuckleesty, gast	DirectionS- Learning (a) (b) (b)	40) a Yata Mali a 110 berning Ya
Bit OZA Bit OZA <t< td=""><td>Terretorme, st., Loss Loss Loss Loss Loss Loss Loss Loss</td><td>ZE(01)</td></t<>	Terretorme, st., Loss Loss Loss Loss Loss Loss Loss Loss	ZE(01)
All Components . Emitted 71 Acked 71 Latency 31 Failed W	views Executors	Dathbard- (0) Dath Constraint (0) Inter Frees Teners that Adust Less Ses 22-0 13. Less we see we

Throughput Improvements for the Kafka Source

The below is the before and after metrics for the TruckGeoEvent Kafka Sink:

BEFORE			After
Image: Second	By increasing from 1 to 3 substantial i throughput wit events consun and ac	y parallelism 3, we see increase in th respect to ned (emitted) cked	Image: Second
TruckGeoEvent - Emitted M Acked M Complete Latency 77 Failed Workers Executors Faile 5.1K -1.1k 2.5K -500.0 24.2sec +2.2sec 0 0 3 19 Hidd	e Metrics 🗸	🌸 TruckG Kafka	leoEvent - Emented 71 Acked 71 Complete Latency 71 Failed Workers Executors 15k +3k 7.4k +1.4k 23.65eC +1.8sec 0 0 6 33 Hide Metrics ↓
Input/Output		Input/Output	
Acked Tuples		Acked Tuples	
Failed Tuples		Failed Tuples	
Queue		Queue	······································
Complete Latency		Complete Late	

The below is the before and after metrics for the Dashboard-Predictions Druid Sink:

BEFORE	After
Debug Decay 2000 Constant Private Exercise Failed Failed Constant Private Constant Constant Constant Private Constant Co	By increasing parallelism from 1 to 3, we see substantial increase in throughput of the events written to Druid
Dashboard-Predictions - Entered M Acked M Process Latency M Execute Latency M Face 0 Doub 730.0 200.0 350.0 160.0 9.4ms -508.7ms 1.5ms 1.4ms 0 Workers Executors 19 Hide Input/Output Acked Tueles	ed Dashboard-Predictions - Emitted 7 Acked 7 Process Latency 2 Execute Latency 7 Failed 2.0K +40.0K 1.3K +2200 9.5ms 463.3ms 2.3ms +0.2ms 0 0 Worker Executer 6 33 Hout/Output Executer
Failed Tuples	Failed Tuples
Process Latency Execute Latency	Cueve Process Latency Execute Latency

Identifying Processor Performance Bottlenecks

In this scenario, we identify a custom processor that has high latency. After running the data simulator for 30 mins, we view the Overview Metrics of the topology.



Scanning over the metrics, we see that the NORMALIZE-MODEL-FEATURES custom processor has high execute latency of 2 seconds. This means that over the last 30 minutes the average time an event spends in this component is 2 seconds.



After making changes to the custom processor to address the latench, we re-deploy the app via SAM and run the data generator for about 15 minutes and view seeing the following metrics.



SAM's overview and detailed metrics makes it very easy to verify if the performance changes we made had the desired effect.

Latency Improvements

The below is the before and after metrics for the NORMALIZE-MODEL-FEATURES custom processor.

BEFORE			AFTER
NORMALIZE 4 01 NORMALIZE Emitted Process Execute Failed Acked Latency 2.0sec 0 240.0 Log: None Sampling 0%	After refactor processor, we se decrease sub correlates to inc (emitted and a	ing the custom ee execute latency stantially which reased throughput cked increases)	NORMALIZE Image: 01 minipage Emitted Process Execute Failed Latency Latency 28k 112.5mm Acked 114.7ms
NORMALIZE-MODEL-FEATURES-DELAY - Emitted Acked P Custom Execute Later of 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Process Latency 2.0sec 2.0sec s Executors 33 Hide Metrics V		9.2.2. DEL-FEATURES-DELAY. 286 vote 9 2A vote 112.5ms 1992 5ms Events Latercy 3J 114.7ms 1888 3ms 0 0 6 33 Hide Metrics ↓
Input/Output		Input/Output	
Acked Tuples		Acked Tuples	
Failed Tuples		Failed Tuples	
Queue		Queue	
Process Latency		Process Latency	
Execute Latency		Execute Latency	

In the metric details view, the graphs provides an easy way to compare metrics before and after the code change.

8	NORMALIZE-MODEL-FEATURES-DELA	Y ▲ Emitted 7 Acked 7 Process Latency ¥ 28k +26k 9.2k +8.4k 112.5ms -1892.5ms Execute Latency ¥ Failed Workers Executors 114.7ms -1888.3ms 0 0 6 33 Hide Metrics ✔
The time before code change was done:	Input/Output	
1/23/18 12:31:30	1/23/2018 012:31:30 РМ Acke 012:31:30 РМ Acked 140 Failed Tuples 1/23/2018 012:31:30 РМ	
Events being queued up	Failed 0 Queue 1/23/2018 0 12:31:30 PM Wait 1024 Process Latency 1/23/2018 0 12:31:30 PM ProcessTime 2001.57	
2 sec Execute Latency	Execute Latency	
	NORMALIZE-MODEL-FEATURES-DE	LAY - Emitted - Acked - Process Latency - Emitted - Acked - Process Latency - Execute 28k + 26k 9.2k + 8.4k 112.5ms - 1892.5ms Execute Latency - Failed Workers Executors 114.7ms - 1888.3ms 0 0 6 33 Hide Metrics - Hide Metrics - Hid
The time after the app was re-deployed with updated custom processor: 1/23/2018 12:38	Input/Output Acked Tuples Failed Tuples	
Event queue dropped from 1024 to 1	Queue Queue Pailed 0 1/23/2018 0 12.38.00 PM 1/23/2018 0 12.38.00 PM Wait 1 Process Latency 1/23/2018 0 12.38.00 PM Wait 1	
Execute time dropped from	Execute Latency	

You can also select the Metrics tab to validate the performance improvement.

.

My Applications / View: streaming-ref-app-advanced_AUTOCREATED-normalize-delay



If you zoom in on the NORMALIZE-MODEL-FEATURES component, you will see that after the code change is made, throughput increases and the wait drops to 0.

	NORMALIZE ◀ 01 ►
	Emitted Process Execute Failed Latency Latency 28k 112.5ms 114.7ms 0
	Acked 9.2k
The time when app was re-reployed	QUEUE
with changes	

Debugging an Application through Distributed Log Search

In a distributed system, searching for logs on different hosts for different components can be extremely tedious and time consuming. With SAM, all the application logs are indexed via the Ambari Log Search Server via Solr. SAM makes it easy to drill into and search for logs for specific components directly from the DAG view. Follow the below steps to use distributed log search:

Procedure

- **1.** To enable Log Search in SAM, perform the following actions in Ambari.
 - a. In Ambari, select the Log Search service and select 'Log Search UI' from Quick Links.
 - **b.** Select the filter icon on the top right menu.
 - **c.** For the storm_worker component, configure the filter like the following and click Save.

Log Feeder Log Levels Filter

Components	Override	FATAL	VARN	INFO	DEBUG	
storm_worker						

2. In SAM, you can dynamically change the logging level. For example, in SAM view mode of an application, click on the Log link, select the log level and the duration you want that log level.



3. Then click on the component you want to search logs for and under Actions select Logs.

			Selee wa	ct the nt to s	comp earch	oner logs	nt yo s for	u										
	→ JOIN-ENRIC	€ 01 ►		IORMALIZ	, LIZE		€01 ►	1	P	redict			€01	0_0	Predicti RULE	on		€03
- *•	Emitted Process Execut Latency Latence 52k 5.9sec 0.0ms	e Failed Acked	Emitter	d Process Latency 1.4ms	Execute Latency 1.0 ms	Failed	Acked		Emitted 22k	Process Latency 0.2ms	Execute Latency 0.2 ms	Faile 0	d Acked (Emitte 16k	ed Process Latency 0.0ms	Latency 0.0ms	Failed 0	1 Acked
		Sanjung ou SAI	MPLING F	PERCENT	AGE BE	ETWEEN (o TO 100 C	DNLY	y, nu o		Actions View L	ogs					54	
									0	Click	View	Log	S					

4. This brings you to the Log Search page where you can search by component (s), log level(s) and search for strings using wildcard notation.

My Applications / View: streaming-ref-app-advanced_AUTOCREATED-normalize-delay / Log Search

COMPONENT					LOG LEVEL	
× NORMAL	IZE-MODEL-F	EATURES-DELAY	× 🔻		Select Log Level	
SEARCH						
Search						Ø 3 hours - Q
Date/Time	Log Level	Component Name	Log Message			-
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	Preparing bolt 52-NORMALIZE-1	ЛС	DEL-FEATURES-DELAY:(31)	
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	Initialzing FeatureNormalization	р	rocessor	
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	Configured Delay timeout is (ne	w)	2	
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	Finished Initialzing FeatureNorn	nal	ization processor	
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	Prepared bolt 52-NORMALIZE-N	10	DEL-FEATURES-DELAY:(31)	
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	About to do feature normalizatii entTime'."2018-01-23 18:11:11. emiee',"routeld':6;"route':'Memp ond''.1',"geoAddress':"No Addre n":"Y";"driverWagePlan:''hours'',v ather':0.0,"Model_Feature_Rainy 9),"auxiliaryFieldsAndValues''.0; 1-4666-a4e4-e046ab3bb2/8'',"7f 2772c675219'',"c9abc1e7-17f4- cb0-1524-46de-8993-14fdf230a	on 17 hi: ss driv W he ee 1a be	event: StreamlineEvent{"dataSourceld": "multiple s: 9", "eventSource": "truck_geo_event", "truckld":84, "dri s to Little Rock", "eventType": "Normal", "latitude":35. Available", "speed": 67, "splitJoinValue": 1516731071 verFatigueByHours": "51", "driverFatigueByMiles": "27 eather": 0.0, "Model_Feature_WindyWeather": 1.0, "ev ader": ("sourceComponentName": "JOIN-ENRICHMI d3d0-6b40-4e68=ac3a-cce94e040a9b"), "parentIds" e3-ba99-6180405d7806", "318ffe99-00a5-4bf4-936 5", "sourceStream": "default").	verces", 'fieldsAndValues'. (' verld": 15, 'driverName": 'Joe I 19, 'longitude": -90.04, 'correla 179, 'week": 4, 'driverCertifica 01", 'Model_Feature_FoggyV 01", 'Model_Feature_FoggyV 01", 'TimeLong": 15167310711 ENTS", 'rootlds": ['4a149dff-5 ['688aa81-2375-413c-a166 b-b2f91e661375"]), ''d": '9011
3 hours ago	INFO	NORMALIZE- MODEL-	Normalized Feautres are: {Mode ature_Certification=1, Model_Fe	el_i ati	Feature_FatigueByHours=0.51, Model_Feature_Fat Jre_WagePlan=0}	igueByMiles=2.701, Model_

Debugging an Application through Sampling

For troubleshooting, a convenient tool is to turn on sampling for a component or for the entire topology based on a sample percentage. Sampling allows you to log the emitted values of a component in the SAM App.

Procedure

- 1. To enable Log Search in SAM, perform the following actions in Ambari.
 - a. In Ambari, select the Log Search service and select 'Log Search UI' from Quick Links.
 - **b.** Select the filter icon on the top right menu.
 - c. For the storm_worker_event component, configure the filter like the following and click Save.

Log Feeder Log Levels Filter

Components	Override	S FATAL	VARN WARN	DEBUG	
storm_worker_event					

2. In SAM view mode of the App, click on the component you want to turn on sampling for and enter a sampling percentage.

	17k 33.6s	ece Palled Sy sec 0	8.1k			
SAM	Log: None	Sam	pling: 0%	0 ONLY	Actions	

3. Click the 'SAMPLE' Tab .



4. Use the Sample Search UI to search for different events that were logged.

SELECT COM	PONENT :			C	DATE / TIME :				
× TruckGe	oEvent		× •		2018-01-23 14:54:08 - 2018-01-23 15:24:08	@ 30 minu	tes 🕶		
SEARCH BY K	EY:			S	SEARCH BY ID :				
Search by	/ Key Values, Hea	ders, Aux Key Values	Q		Search by Event Id, Root Id, Parent Id		Q		
Date/Time	Component	Key Values				1	<u>i</u> -¢-		
8 minutes ago	TruckGeoEvent	"{eventTime=2018-01-23 21:21:13.616, eventTimeLo routeld=2, route=Memphis to Little Rock, eventType=	ng=151 Lane De	674: epar	2473616, eventSource=truck_geo_event, truckId=14, driverId=13, d ture, latitude=34.8, longitude=-92.09, correlationId=1, geoAddress	riverName=Suresh No Address Availab	Srinivas, ole}"		
8 minutes ago	TruckGeoEvent	eventTime=2018-01-23 21:21:20.486, eventTimeLong=1516742480486, eventSource=truck_geo_event, truckid=106, driverid=11, driverName=Jamie Engess routeld=12, route=Springfield to KC Via Hanibal, eventType=Normal, latitude=39.78, longitude=-93.13, correlationId=1, geoAddress=No Address Available)*							
8 minutes ago	TruckGeoEvent	*(eventTime=2018-01-23 21:21:30.056, eventTimeLong=1516742490056, eventSource=truck_geo_event, truckId=56, driverId=10, driverName=George Vetticac en, routeId=0, route=Peoria to Ceder Rapids Route 2, eventType=Normal, latitude=42.23, longitude=-91.78, correlationId=1, geoAddress=No Address Availabl e)*							
8 minutes ago	TruckGeoEvent	"{eventTime=2018-01-23 21:21:31.546, eventTimeLo teld=5, route=Memphis to Little Rock Route 2, event	ng=151 Type=No	674: orma	2491546, eventSource=truck_geo_event, truckId=101, driverId=21, al, latitude=34.78, longitude=-92.31, correlationId=1, geoAddress=N	driverName=Ajay Si Io Address Available	ngh, rou }"		
7 minutes ago	TruckGeoEvent	"{eventTime=2018-01-23 21:21:42.586, eventTimeLo outeId=3, route=Joplin to Kansas City Route 2, event	ng=151 Type=No	674: orm	2502586, eventSource=truck_geo_event, truckId=104, driverId=14, al, latitude=37.31, longitude=-94.31, correlationId=1, geoAddress=I	driverName=Paul C No Address Availabl	odding, r e}"		
7 minutes ago	TruckGeoEvent	"{eventTime=2018-01-23 21:21:45.086, eventTimeLo teld=1, route=Saint Louis to Memphis, eventType=No	ong=151 ormal, la	674: tituo	2505086, eventSource=truck_geo_event, truckId=38, driverId=26, d de=38.43, longitude=-90.35, correlationId=1, geoAddress=No Addre	riverName=Don Hilb ess Available}*	oorn, rou		
7 minutes ago	TruckGeoEvent	*{eventTime=2018-01-23 21:21:48.166, eventTimeLo outeld=10, route=Joplin to Kansas City, eventType=N	ong=151 Normal, I	674: latitu	2508166, eventSource=truck_geo_event, truckId=64, driverId=28, d ude=37.66, longitude=-94.3, correlationId=1, geoAddress=No Addre	riverName=Michael ess Available}"	Aube, r		
7 minutes ago	TruckGeoEvent	"(eventTime=2018-01-23 21:21:57.636, eventTimeLong=1516742517636, eventSource=truck_geo_event, truckId=92, driverId=22, driverName=Chris Harris, rou teld=7, route=Saint Louis to Chicago, eventType=Normal, latitude=38.65, longitude=-90.2, correlationId=1, geoAddress=No Address Available}"							
7 minutes ago	TruckGeoEvent	"(eventTime=2018-01-23 21:21:58.666, eventTimeLong=1516742518666, eventSource=truck_geo_event, truckId=17, driverId=29, driverName=Mark Lochbihler, routeId=10, route=Springfield to KC Via Hanibal Route 2, eventType=Normal, latitude=39.71, longitude=-92.07, correlationId=1, geoAddress=No Address Availa ble)*							

Spark Streaming

Information on how to create a Streaming Analytics Application with Spark Streaming is under development and will be available after the HDF 3.2.0 release is complete.

Running the Stream Simulator

Now that you have developed and deployed the NiFi Flow Application and the Stream Analytics Application, you can run a data simulator that generates truck geo events and sensor events for the apps to process.

About this task

To generate the raw truck events serialized into Avro objects using the Schema registry and publish them into the raw Kafka topics, do the following:

Procedure

- **1.** Download the Data-Loader.
- **2.** Unzip the Data Loader file and copy it to the cluster. Lets call the directory to which you unzipped the file as \$DATA_LOADER_HOME.

3. Change into the Data Loader directory:

cd \$DATA_LOADER_HOME

4. Untar the route.tar.gz file:

tar -zxvf \$DATA_LOADER_HOME/routes.tar.gz

- 5. Open the file startTruckGenerators.sh and make the following changes:
 - a. Modify the kafkaBrokers value based on your cluster
 - **b.** If your cluster is not secure, set the value of SECURE_MODE to NONSECURE and set JAAS_CONFIG to an empty space
 - **c.** Set the value of ROUTES_LOCATION to the location where you untar the routes in step 4 and then (e.g: \$DATA_LOADER_HOME/routes/midwest)
- 6. Run the simulator/data generator:

```
./startTruckGenerators.sh
```

Results

You should see events being sent to the gateway kafka topics. NiFi should consume the events, enrich them, and then push them into the syndicate topics. SAM, Spark Streaming and other apps should consume from these syndicate topics

Managing Kafka with Streams Messaging Manager

SMM Overview

A key part of this streaming application has been the use of Kafka which powers the IOT Gateway and the Syndication Services. In this architecture, Kafka is everywhere.



Kafka is Everywhere. Critical Component of Streaming Architectures

As a result, it becomes critical to be able to monitor and understand what is going on in the cluster to cure the Kafka blindness. To accomplish this, we will use Hortonworks Streams Messaging Manager to monitor the Kafka components of this reference application.

Installing DataPlane Streams Messaging Manager

Follow the *SMM Installation* documentation to install SMM which requires DataPlane Service (DPS) platform as well as certain prerequisites required on the HDP/HDF cluster where Kafka is running.

Enabling Reference Application Cluster for SMM

After installing DPS with the SMM application, you need to register the cluster you have created for the trucking reference application. The below steps walks you through this registration process.

Procedure

1. Log into DataPlane that was installed and click Add Cluster.



2. Provide Ambari endpoint and details for the HDP/HDF cluster that is being added to DataPlane.

🜍 DATAPLA	NEADMIN V Adm	nin / Clusters / Add	<u>.</u>
& Cluetere	📑 Ad	dd Your Cluster	
eat	Ple	see add Ambari managed cluster to DataPlane. DataPlane will auto-discover cluster information such as the cluster name, active services & important metrics running.	
👕 Users			
Services	Che	Amban and ulaste services beining Antox autoway @ kothis option if Amban and your cluster services are accessed via a Knox Gateway.	
Settings	Che	Validate the SSL certificate and only allow trusted connections 0	
	Am	Step 1	
	h	ttp://c-dpe-connected dp0.field.hortonworks.com:8080 Provide Ambari	
		endpoint for cluster	
		Cluster Name orlandostreamcluster	
		IP Address Step 2	
		Services DP will discover the	
		HIVE O HDFS O ATLAS O ZOOKEEPER O KAFKA Services unning on	
		v All Services	
		O Datalake	
		This cluster has been identified as a Datalake since it has data management services, ATLAS running.	
		Cluster Location Geographical location of the cluster	
		Orlando, Florida, United States of America Step 3	
		Data Center* Enter details of the	
		Data Center of the cluster Cluster being added	
		Uranoo Uata Center - Green Truck Company	· · · ·
		Taga Taga Taga Taga Taga Taga Taga Taga	
		Description	Sm.
DataPlane V	ersion 1.2	Description of the cluster	S P

3. After the cluster has been added, go to the cluster details page and enable the SMM application.

õ	Admin / Clusters / Det	tails										å
B	orlandostreamcluster											C
	LOCATION Orlando, United States of America	DATA CENTER Orlando Data Center - Green Truck Company	CLUST HDP-5	TER VERSION 3.0	NODES 16/16		TAGS		REGISTE Thu Au	ERED AT 9 02 2018	REGISTERED BY gvetticaden	
•	INFORMATION							1		Cluster Services		
	Description			Services						ATLASHDFS	0.7.0.3.0 3.1.0	5
	Connectivity		Security Type	Do you want to	o enable Stream	s Messaging Manager	on cluster orlandostre	amcluster?		HIVE KAEKA	3.0.0.3.	ο O
	Reachable		KERBEROS				CANCEL	ок		RANGER	1.0.0.3.0	0
	DataNodes		NameNode Heap S	Size 12.49%		HDFS Disk Space	- 25.71%	_		 STREAMS MESSAGING MANAGER 	G 1.0.0	
			125 MB/1004 MB			61 GB/237 GB				ZOOKEEPER	3.4.9.3.	D
	NodeManagers		ResourceManager	r Heap Size		ResourceManager	e					
	● 3 ● 0		57 MB/911 MB	6.21%		2 days						
						Click Confir	Enable and m enabling of					
	SERVICES					the	SMM App	Show All Service				
l	Enable DPS Services that co corresponding service.	an access and manage your cl	luster. Below are the	e services that are available in	DataPlane. Click	on Enable against a se	br associating this c	luster to the				
	• STREAMS MESS	AGING MANAGER					ENABLE					

4. Once the SMM app is enabled, you should see the SMM Icon from the app picker. Click on the SMM App to start monitoring the Kafka brokers in the cluster you registered.

۵Ì	Admin / Clusters / Deta	ails							<u>*</u> *
	er								C
Datal	Plane Streams	DATA CENTER Orlando Data Center - Green Truck Company	CLUSTER VERSION HDP-3.0	NODES 16/16		TAGS	REGISTERED AT Thu Aug 02 2018	REGISTERED B gvetticaden	Ŷ
Ādī	Minini Messaging Manager Description Connectivity • Reachable DataNodes • 3 • 0 NodeManagers • 3 • 0 SERVICES I Enable DPS Services that car corresponding service.	After the App has been enabled, use should see SMM App in the app picker list. Select if KERBERO: NameNod 125 MB/11 Resourced 57 MB/91	r ppe 3 Heap Size 12.49% Annager Heap Size MB 6.21%	DataPiene, Click	No of Services 26 HDFS Disk Space 61 GB/237 GB ResourceManager Up 2 days	25.71% ime Show All Servic ce for associating this cluster to the	E Cluster Servic ATLAS HDFS HIVE KAFKA RANGER STREAMS MANAGER ZOOKEEPER	VESSAGING 1	17.0.3.0 11.0 10.0.3.0 10.0.3.0 10.0.3.0 10.0 10
	🝿 🏾 🛛 STREAMS MESSA	GING MANAGER				ENABLED			
V 1.2									

5. In the SMM App, every HDP/HDF cluster you enabled with SMM shows up in the cluster dropdown. Hence, a single SMM App can monitor multiple clusters. Select the cluster you want to monitor.

STREAMS MESSAGING	Overview							Clu	uster: <u>orlandostreamcluster</u> .•	
🍄 Overview	Producers 71	Brokers 5	-		Every HD enabled	P/HDF cluster y with SMM show	vou ws	Consum	orlandostreamcluster ne chicagostreamcluster 26	Clear
Brokers	TOPICS (27) BROKERS (5)				box. F	Pick a cluster to			1 3 30 mir	nutes 🗸
Topics		NAME	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS				
Producers	Producers (71) ACTIVE (71) PASSIVE (0) ALL	syndicate-transmission		DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>†</u> 600,≣	~	ACTIVE (5) PASSIVE (21)) ALL
Consumer Groups	MESSAGES minifi-eu-i1 37k		041ND	TUKB	эк	0			nifi-truck-sensors-east	LAG 4
	load-optimizer-apps 35k	syndicate-speed-event-j	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>†</u> 600 ≡	~	nifi-truck-sensors-west	2
	geo-critical-event-collec 25k	•	513KB	OB	1.8K	U			nifi-truck-sensors-cent	1
	minifi-eu-i2 19k	and instants of summers	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS			ranger_entities_consu	1
	supply-chain-apps 12k	syndicate-speed-event-a	317KB	41MB	1.8k	3	ଷୁଢୁଢୁ©	~	atlas	0
	minifi-eu-i4 9.5k									
	geo-critical-event-collec 9.5k	syndicate-oil	DATA IN 757KB	DATA OUT OB	MESSAGES IN 3.5k	CONSUMER GROUPS	<u>∲</u> @Q ≡	~		
	predictive-apps 8.9k									
	geo-critical-event-collec 7.6k	syndicate-geo-event-json	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u> 690 </u>	~		
	minifi-eu-i5 7.6k		695KB	OB	1.8K	0				
	geo-critical-event-collec 6.9k		DATA IN	ράτα ομτ	MESSAGES IN	CONSUMER GROUPS				
	minifi-eu-i6 6.4k	syndicate-geo-event-avro	397KB	39GB	1.8k	8	<u>†</u> ©Q ≡	~		
	geo-critical-event-collec, 6.3k									
	audit-apps 5.9k	syndicate-battery	DATA IN 703KB	DATA OUT OB	MESSAGES IN 3.2k	CONSUMER GROUPS	🧑 🖗 Q 🔳	~		
	geo-critical-event-collec 5.9k									
	minifi-eu-i7 5.5k	syndicate-all-geo-critical	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u> 690 </u>			
	geo-critical-event-collec 5.4k	-jinaioato an goo ontioum	26MB	OB	0.1m	0	₩ € ~ Ш			
	nee eritiest sucht selles									

Monitoring Kafka with SMM

SMM helps address the operational, management and monitoring needs of Kafka for two distinct teams: the Platform Operations and the DevOps / AppDev teams. Each of these teams have a different lens through which they monitor Kafka and hence have different needs for monitoring Kafka.

SMM Platform Operations Persona

A Platform Operations user is less concerned about the individual performance for a given consumer and/or producer application but rather more focused on the Kafka cluster holistically and the infrastructure that it runs on. Some specific needs, requirements, and questions from a Platform Operator may include the following:

Platform Ops Use Case	Description
Use Case 1	I would like a single platform to monitor all the Kafka clusters within my organization. I want to be able to quickly switch from one Kafka cluster to another.
Use Case 2	I would like to get quick current snapshot of my cluster: number of producers, number of brokers, number of topics, number of consumers.
Use Case 3	Across the entire cluster, which producers are generating the most data right now?
Use Case 4	Across the entire cluster, which of my consumer groups and consumer instances are falling behind with respect to reading from a topic or partition?
Use Case 5	I would like to see a snapshot view of all the Kafka brokers in my cluster with information including the hosts on which the broker is running, throughput in, messages in, number of partitions, and number of replicas.
Use Case 6	Are any of my brokers running hot? Which broker has the highest throughput in and out rates?
Use Case 7	Which topic partitions are on a given Kafka broker?
Use Case 8	Are there any skewed partitions for a broker? What is the throughput in and out for a given partition on that broker?
Use Case 9	For a given broker, topic, or partition, which producers are sending data to it, and which consumer groups are consuming from it.
Use Case 10	View detailed level metrics of a broker across time to see trends and patterns.
Use Case 11	Whats are host metrics on the host where my broker is running? What are the other services running on my broker host?

Let's walk through how SMM can answer these questions for a Platform Operations user:

1. Select the Kafka Cluster you want to monitor with SMM. This takes you to the main dashboard view for that cluster selected. This view gives you a powerful snapshot of the workings of the cluster displaying: total number of active and inactive producers and consumers, all topics with summary metrics. This view also provides the ability to filter on four key Kafka entities: producers, brokers, topics, and consumer groups. This addresses
Use Case 1 which is the ability for SMM to manage multiple Kafka clusters in the organization and Use Case

STREAMS MESSAGING	Overview				The or	Kafka cluster landostreamc selected	called luser	Cluster: orlandostreamcluste	ar . ≜ -
Overview	Producers 59	Brokers 5		-	Topics 27	÷	Cons	umer Groups	Clear
Brokers	TOPICS (27) BROKERS (5)							1 0 20	minutes *
🛢 Topics	Producers (50)	NAME	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS		Consumer Groups (26	6)
Producers	ACTIVE (59) PASSIVE (0) ALL MESSAGES	syndicate-transmission	data in 647KB	data out 10KB	MESSAGES IN 3k	CONSUMER GROUPS	<mark>७</mark> 00≣ ∨	ACTIVE (18) PASSIVE (8)	ALL
	minifi-eu-i1 39k load-optimizer-apps 36k	syndicate-speed-event-j	DATA IN 523KB	DATA OUT OB	MESSAGES IN 1.8k	CONSUMER GROUPS	<u></u> ©@Q≣ ∨	route-micro-service load-optimizer-micro-se	97k e 5.1k
	minifi-eu-i2 20k minifi-eu-i4 9.8k	syndicate-speed-event	DATA IN 319KB	DATA OUT 41MB	MESSAGES IN 1.8k	CONSUMER GROUPS	<u></u> ©Q⊡ ∨	fuel-micro-service supply-chain-micro-service	2.5k v 1.7k
	geo-critical-event-collec 7.7k geo-critical-event-collec 7.1k	syndicate-oil	DATA IN 779KB	DATA OUT	MESSAGES IN 3.6k	CONSUMER GROUPS	<u></u> ©Q≣ ∨	energy-micro-service audit-micro-service	984 812
	geo-critical-event-collec 6.5k minifi-eu-i6 6.5k	syndicate-geo-event-json	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>6</u> @Q≣ ∨	compliance-micro-servi adjudication-micro-serv	i 698 /i 599
	audit-apps 6k geo-critical-event-collec 6k		703KB	DATA QUIT	I.8K			approval-micro-service flink-analytics-geo-even	542 1t 224
	geo-critical-event-collec 5.5k	syndicate-geo-event-avro	400KB	39GB	1.8k	8	<u>†</u> 8Q ■ ∨	spark-streaming-analytics	i 224
	minifi-eu-i8 4.8k	syndicate-battery	data in 706KB	data out 0B	MESSAGES IN 3.2k	CONSUMER GROUPS	<mark>8</mark> 0Q≣ ∨	nifi-truck-sensors-east	2
	geo-critical-event-collec 4.8k	syndicate-all-geo-critica	DATA IN 16MB	DATA OUT OB	MESSAGES IN 74k	CONSUMER GROUPS	<mark>©</mark> @Q≣ ∨	nifi-truck-sensors-centra	ai T n 1
	adjudication-apps 4.5k minifi-eu-i9 4.3k	aunnhu abain	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS		atlas	0
	approval-apps 4k	Supply-chain	299MB	530KB	1.4m	1			

2. From the Producers panel on the left hand side of the screen, select the Messages header to sort on messages sent by all the producers in the system. We see that the Kafka producer called minifieui1 is the most active producer, sending 39K messages in the last 30 minutes. This addresses Use Case

STREAMS MESSAGING OV	verview							CI	uster: orlandostreamcluster 👻 🛔 🛩
28 Overview	Click or on me produ	n Message ssages sei cers in the	to sort Brokers Brokers Bast 30	Ŧ		Topics 27	•	Consur	mer Groups 26
Brokers	TOPICS (27)	mins							"⊃ 30 minutes ◄
🛢 Topics	Producers (59)		NAME	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS		Consumer Groups (26)
Producers	ACTIVE (59) PASSIVE (0)		syndicate-transmission	DATA IN 647KB	data out 10KB	MESSAGES IN	CONSUMER GROUPS	<u></u> ©@Q≣ ∨	ACTIVE (18) PASSIVE (8) ALL
*) Consumer Groups	minifi-eu-i1 load-optimizer-apps	39k 36k	syndicate-speed-event-j	DATA IN 523KB	DATA OUT OB	MESSAGES IN 1.8k	CONSUMER GROUPS	© @Q≣ ∨	route-micro-service 97k load-optimizer-micro-se 5.1k
A Kafka producer called minfi-eu-i1 is the most	ninifi-eu-i2 ninifi-eu-i4	20k 9.8k	syndicate-speed-event	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>6</u> @Q≡ ∨	fuel-micro-service 2.5k supply-chain-micro-serv 1.7k
active producer sending 39K messages in the last 30 mins	peo-critical-event-collec	9.8k 7.7k		DATA IN	4 I MB	I.8K	3 CONSUMER GROUPS		predictive-micro-service 1.3k energy-micro-service 984
	geo-critical-event-collec	6.5k	syndicate-oil	779KB	0B	3.6k	0	<u></u> ©Q≡ ∨	compliance-micro-servi 698
	minifi-eu-i6 audit-apps	6.5k 6k	syndicate-geo-event-json	data in 703KB	data out OB	MESSAGES IN 1.8k	CONSUMER GROUPS	<u>†</u> 90	adjudication-micro-servi 599 approval-micro-service 542
	geo-critical-event-collec geo-critical-event-collec	6k 5.6k	syndicate-geo-event-avro	data in 400KB	data out 39GB	MESSAGES IN 1.8k	CONSUMER GROUPS	<mark>©</mark> @Q≣ ∨	flink-analytics-geo-event 224 kafka-streams-analytics 224
	compliance-apps minifi-eu-i8	5.2k 4.8k	syndicate-battery	DATA IN 706KB	DATA OUT	MESSAGES IN 3.2k	CONSUMER GROUPS	<u>ö</u> @Q≣ ∨	nifi-truck-sensors-east 4 nifi-truck-sensors-west 2
	geo-critical-event-collec geo-critical-event-collec	4.8k 4.6k	syndicate-all-geo-critica	DATA IN 16MB	DATA OUT	MESSAGES IN 74k	CONSUMER GROUPS	<u>¢</u> @Q≣ ~	nifi-truck-sensors-central 1 ranger_entities_consum 1
	adjudication-apps minifi-eu-i9 approval-apps	4.5k 4.3k 4k	supply-chain			MESSAGES IN	CONSUMER GROUPS	<u>ö</u> @Q≣ ~	atlas 0
«				733IAID	JJUND	1.4111	1		

3. On the Consumer Groups panel on the right hand side, select the column LAG to sort on consumer group lag. This lag is defined as the summation of all consumer instances lag in the consumer group. This addresses

Use Case 4 because we can easily see that we have a consumer group called micro-alert-service that has a lag of 97K over the last 30 minutes which is significantly more than any other consumer group in the

MESSAGING	Overview						c	Cluster: orlandostreamcluster - 🔺 -
a Overview	Producers 59	• Brokers 5	-		Topics 27	•	Const	26 Click on LAG to sort on consumer lag across all consumers in the last 30
Brokers	TOPICS (27) BROKERS (5)							mins
Topics		NAME	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS		
Producers	Producers (59)	syndicate-transmission	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	600 T	Consumer Groups (26)
	ACTIVE (59) PASSIVE (0) ALL MESSAGES		647KB	10KB	3k	0		
Consumer Groups	minifi-eu-i1 39k							route-micro-service 97k Consumer group named
	load-optimizer-apps 36k	syndicate-speed-event-j	523KB	DATA OUT OB	MESSAGES IN 1.8k	0	<u>†</u> 800≣ ∨	load-optimizer-micro-se 5.1k
	minifi-eu-i2 20k							fuel-micro-service 2.5k (97K) than any another
	minifi-eu-i4 9.8k	syndicate-speed-event-	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	****	supply-chain-micro-serv 1.7k consumer in the cluster
	geo-critical-event-collec 9.8k		319KB	41MB	1.8k	3	O G C C	predictive-micro-service 1.3k
	geo-critical-event-collec 7.7k							energy-micro-service 984
	geo-critical-event-collec 7.1k	syndicate-oil	779KB	DATA OUT OB	MESSAGES IN 3.6k	CONSUMER GROUPS	© Q⊡ ∨	audit-micro-service 812
	geo-critical-event-collec 6.5k							compliance-micro-servi 698
	minifi-eu-i6 6.5k	syndicate-rec-event-ison	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	****	adjudication-micro-servi 599
	audit-apps 6k	of the ofference geo event joon	703KB	0B	1.8k	0	O G C C C	approval-micro-service 542
	geo-critical-event-collec 6k							flink-analytics-geo-event 224
	geo-critical-event-collec 5.6k	syndicate-geo-event-avro	400KB	DATA OUT 39GB	MESSAGES IN 1.8k	CONSUMER GROUPS	<mark>6</mark> @Q≣ ∨	kafka-streams-analytics 224
	minifi-eu-i7 5.5k							spark-streaming-analyti 224
	compliance-apps 5.2k	aundicate-batten/	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS		nifi-truck-sensors-east 4
	minifi-eu-i8 4.8k	Syndicate-Dattery	706KB	0B	3.2k	0	Ø⊌Q⊞ ∨	nifi-truck-sensors-west 2
	geo-critical-event-collec 4.8k							nifi-truck-sensors-central 1
	geo-critical-event-collec, 4.6k	syndicate-all-geo-critica	16MB	DATA OUT OB	MESSAGES IN 74k	CONSUMER GROUPS	© Q⊡ ∨	ranger_entities_consum 1
	adjudication-apps 4.5k		16MB 0E					atlas 0
	minifi-eu-i9 4.3k	aunaly shain	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	****	
*	approval-apps 4k	suppry-chain	299MB	530KB	1.4m	1	<u>⊽</u> ⊌ų	

4. Click the Brokers tab to see a broker centric view of the dashboard. You can view important metrics for each broker in the cluster including: hosts the broker is running on, throughput in, messages in, number of partitions, and number of replicas. Click the THROUGHPUT column to sort by throughput across all brokers. You can easily see that broker 1001 has the highest rate of data in over the last 30 minutes: 80K messages totaling 17MB. This addresses Use Cases 5 and

Click on t to see a	Step 1 the Brokers tab broker centric the Dashboard	ers		St Click on Th sort on dat	ep 2 nroughput to a in across a		Topics			Cons	Cluster: orlandostreamcluster	r~ &.
	59			bro	okers		27	Ť			26	
Brokers	TOPICS (27) BROKERS	(5)				_					n 08 C	ninutes
Topics	Dradwaara (50)		NAME		THROUGHPUT -	MESSAGES IN	PARTITIONS	REPLICAS			Consumer Crowns (26	
Producers	ACTIVE (50) DASSINE (0)	ALL	1001		THROUGHPUT	MESSAGES IN	PARTITIONS	REPLICAS	18 D 🗖	~	ACTIVE (19) DASSIVE (20))
í <u> </u>	ACTIVE (59) PASSIVE (0)	ESSAGES	c-dps-connected-c	p13.field.hortonwor	17MB	80k	21	36	V &		ACTIVE (18) PASSIVE (8)	LAG
Consumer Gro	Analysis	396									route-micro-service	97k
Bro	ker 1001 has the highes	t 🦯	1002	n12 field hortonwor	THROUGHPUT 16MB	MESSAGES IN	PARTITIONS 16	REPLICAS 34	🎋 🔕 🔲	~	load-optimizer-micro-se.	5.1k
rate	of data in over the last 3	20k									fuel-micro-service	2.5k
min	17MB	.8k	1005		THROUGHPUT	MESSAGES IN	PARTITIONS	REPLICAS			supply-chain-micro-serv.	1.7k
		.8k	c-dps-connected-c	p11.field.hortonwor	14MB	63k	15	31	🥸 🚯 🔳	~	predictive-micro-service	1.3k
	geo-critical-event-collec	7.7k									energy-micro-service	984
	geo-critical-event-collec	7.1k	1003	at 4 Gold bootservice	THROUGHPUT	MESSAGES IN	PARTITIONS	REPLICAS	🧔 🔕 🔁	~	audit-micro-service	812
	geo-critical-event-collec	6.5k	c-aps-connected-c	p14.neid.nortonwor	TONID	721	10	50			compliance-micro-servi.	698
	minifi-eu-i6	6.5k	1004		THROUGHPUT	MESSAGES IN	PARTITIONS	REPLICAS			adjudication-micro-servi	i 599
	audit-apps	6k	c-dps-connected-c	p15.field.hortonwor	7MB	33k	14	30	🧔 🕲 🥵	~	approval-micro-service	542
	geo-critical-event-collec	6k									flink-analytics-geo-event	t 224
	geo-critical-event-collec	5.6k									kafka-streams-analytics	224
	minifi-eu-i7	5.5k									spark-streaming-analyti.	224
	compliance-apps	5.2k									nifi-truck-sensors-east	4
	minifi-eu-i8	4.8k									nifi-truck-sensors-west	2
	geo-critical-event-collec	4.8k									nifi-truck-sensors-centra	ป 1
	geo-critical-event-collec	4.6k									ranger_entities_consum	1
	adjudication-apps	4.5k									atlas	0
	minifi-eu-i9	4.3k										
	approval-apps	4k										
	minifi ou i10	2.01/										

5. Click on the broker panel to expand it and see more details and metrics for that broker. The expanded panel shows all partitions for different topics that are stored on that broker. For each partition, we see the throughput in and out relative to other partitions. We can click on a given partition and see all the

consumers currently sending data to that broker, topic, or partition and all consumer groups consuming from that broker, topic, or partition. We can easily see that partition 0 of topic syndicate-speed on broker 1001 has considerably higher throughput out than any of the other partitions on that broker. By viewing how data flows in and out of that partition, we see that the partition has 1 producer but 3 consumer groups which explains the high throughput out relative to the throughput in. This addresses Use Case 7, 8 and



6. Click the Grafana icon on the broker panel to see more detailed metrics within Grafana for that broker. This addresses Use Case

IO02 edpe-connected dp12.feld.hortownor THROUGHPUT MESSAGES IN 16MB PARTMINNS REFUGAS 34 Image: Connected dp12.feld.hortownor displayed providing more broker metrics graphed across time	
1005 cdps-connected dp11.field.hortonwor 14MB 63k 12 0 14Kdua-Hools c	Zoom Out 🛛 Last 6 hours 🏾 🕱
1003 THROUGHPUT MESSAGES IN PA edge connected dp14 field hortomoor. 10MB 42k 1 Benetic (1) per broker level. Click on each row tille to expand on demand to look at various metrics. Al	
THROUGHPUT MESSAGES IN PA cdps connected dp15 field hortomorka.com TMB S32k S TMB S32k S TMB S32k S	icated Partitions
	120 1200 1200 1400
	00 12-20 13:00 13:20 14:00
REFLICA MANAGER	
	ag Replice

7. Click the Ambari icon on the broker panel and view Broker Host detail metrics. This addresses Use Case

NAME 1001 c-dps-connected-dp13.field.hortonwor	THROUGHPUT • THROUGHPUT 17MB	MESSAGES IN MESSAGES IN 80k	PARTITIONS PARTITIONS 21	REPL REPL	ICAS O Trun	anel and the Ambari host view for that broker is edp providing host level ad a view of other services nning on that host
1002 c-dps-connected-dp12.field.hortonwor	THROUGHPUT	MESSAGES IN 75k	partitions 16	8 - 1	↑ / Hosts / c-dps-connected-dp13.field.hortonworks.com / 3	/ Summary orlandostr 🔅 🔘 斗 🌖 🏢 🗘 gvetticade
1005 c-dps-connected-dp11.field.hortonwor	THROUGHPUT 14MB	MESSAGES IN 63k	partitions 15	RS	Host: c-dps-connected-dp13.field.hortonworks.com o	HOST ACTION
1003 c-dps-connected-dp14.field.hortonwor	THROUGHPUT 10MB	MESSAGES IN 42k	partitions 16	RS	Components Status Name Type	+ ADD Host Metrics LAST 1 HOUR
1004 c-dps-connected-dp15.field.hortonwor	THROUGHPUT 7MB	MESSAGES IN	partitions 14	RS	Kafka Broker / Kafka Master HST Agent / SmartGenee Slave Log Feeder / Log Search Slave Metrics Monitor / Ambain Merrics Slave Known Crief / Knownee Client	
					Summary	1
					Hostmanne: c-dps-connected-dp13.field hortonworks.com IP Adfress: 172.26.254.59 Rate: / zedu/arcack // OS: central? (zd6,54) Corres (CPU): 8 (0) Disk: C.3460/73.99068 (7.39% used) Memory: 14.5300 Load Arg: 0.02 Heartheat: less than a minute apo Ourrer Versite: 30.0-19.54	250.5 00 Lad Memory Usage 250.5 00 Lad Memory Usage 150 100 100 100 100 100 100 100

DevOps and Application Developer Persona

Unlike a Platform Operations persona, the DevOps / AppDev persona is most interested in the entities (producers, topics, consumers) specific to their application. So let's assume we are on the DevOps team responsible for monitoring the Trucking Reference Application that is deployed in production, based on this architecture.



Some specific needs/requirements/questions as a DevOps member for the trucking ref app might be the following.

DevOps/App Dev Use Case	Description
Use Case 1	I want to quickly find all entities (producer, consumers, topics) associated with my application.

Use Case 2	For all the topics associated with my applications, I want to easily see important metrics such as throughput in/out, number of consumer groups, number of messages across a period of time.
Use Case 3	Which of my topics is being sent the most amount of data over a certain period of time? In other words, which regional/geo truck fleet is sending the most amount of data?
Use Case 4	For a given Kafka topic that is part of my application, which are all the connected producers sending data? In other words, which truck fleets are sending data to a gateway topic?
Use Case 5	Are there any topics who have producers but no consumer groups connected to it? In other words, are trucks sending data to a topic but no analytics or processing is being done?
Use Case 6	For a given topic, how many partitions are there? Where are the partitions located? How is data distributed across the partitions? Are there any partition skews?
Use Case 7	Which consumer groups are connected to a given topic and are actively consuming data from it?
Use Case 8	For a given topic, I want to be able to explore data in the topic searching using offsets and/or partition.
Use Case 9	I want to find Metadata and Lineage of the Topic across producers, consumers, and multiple Kafka hops.

Let's walk through how SMM can answer these questions for a DevOps/App Dev user.

1. Select the Topic filter and select all the IoT gateway topics by searching for all topics that start with gateway.

MESSAGING	Overview									Cluster: orlandostreamcluste	H+ ≜+
Overview	Producer: 83	s	- Broken	'S	-	Topics 27			Cons	umer Groups	Clear
	TOPICS (27) BROKERS (5)					1		Us	e the	Filter to filter on	minutes -
Topics			NAME	DATA IN -	DAT	Norma	Q	copi	gai	teway topics	initiates •
	Producers (83) ACTIVE (83) PASSIVE (0)	ALL	syndicate-all-geo-critical-eve	DATA IN 26MB	OB OB	name gateway-europe-raw-sense	ors 📼	و و بر س		ACTIVE (18) PASSIVE (8)	ALL
 Consumer Groups 	route-apps	essages 0.1m				gateway-west-raw-sensor:				route-micro-service	LAG 0.2m
	minifi-eu-i1	25k	route-planning	DATA IN 26MB	DAT 0	gateway-central-raw-sense	ors 🔲	6 e q 📼	~	load-optimizer-micro-service	e 12k
	load-optimizer-apps	23k	T	20110	20	gateway-east-raw-sensors				fuel-micro-service	6k
	geo-critical-event-collector-i1	23k		DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS			supply-chain-micro-service	4k
	minifi-eu-i2	12k	gateway-europe-raw-sensors	18MB	0B	88k	0	🧑 😡 Q 🔳	ř	predictive-micro-service	3k
	geo-critical-event-collector-i2	11k								energy-micro-service	2.3k
	geo-critical-event-collector-i3	7.7k	load-optimization	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	🧑 😔 Q 🕅	~	audit-micro-service	1.9k
	minifi-eu-i4	6.3k	T	UNID	2010	205				compliance-micro-service	1.7k
	geo-critical-event-collector-i8	6.3k		DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS			adjudication-micro-service	1.4k
	geo-critical-event-collector-i4	5.8k	fuel-logistics	1MB	28KB	6.5k	1	🧑 😡 Q 🕅	×	approval-micro-service	1.3k
	fuel-apps	5.3k								flink-analytics-geo-event	525
	geo-critical-event-collector-i10	5k	supply-chain	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	👩 😡 Q 🔳	~	kafka-streams-analytics-geo	ə 525
	geo-critical-event-collector-i5	4.6k		003ND	ZOND	4.3K	'			spark-streaming-analytics-g	j 525
	geo-critical-event-collector-i11	4.6k		DATA IN	DATA OUT	LECOADED IN	CONCLINED COOLIDE			nifi-truck-sensors-west	2
	minifi-eu-i6	4.2k	audit-events	804KB	27KB	3.9k	1	🧑 😡 Q 🕅	×	nifi-truck-sensors-east	2
	geo-critical-event-collector-i12	4.2k								nifi-truck-sensors-central	1
	audit-apps	3.9k	compliance	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	6 Q Q 🗐	~	ranger_entities_consumer	1
	geo-critical-event-collector-i13	3.9k		697KB	29KB	3.4K	1			atias	0
	minifi-eu-i3	3.8k	1	0.171.01			000000000000000000000000000000000000000				
	geo-critical-event-collector-i6	3.8k	predictive-alerts	648KB	28KB	3 3k	CONSUMER GROUPS	🙇 😔 Q 📼	~		

2. When the filter is applied, SMM provides intelligent filtering by showing only the producers that are sending data to the 4 gateway topics and the consumer groups only consuming data from those topics. So, when the user selects the 4 gateway topics, SMM displays 34 of the 83 producers sending data to those topics and 3 of the 26 consumer groups consuming data from it. Key metrics for the selected topics are shown including

data-in and out, number of messages, number of consumer groups, etc. This addresses Use Cases 1 and

Intelligent Filte	ering filters the										Intellig SMM auton consumers a selected to consum	gent Filtering natically filte associated wi opics. 3 of th ners have bee	rs the ith th e 26 en
producers associate selected topics. 34 producers have been as sending data to the	d with the of the 84 identified he 4 topics	Producers 34 of 83		Broker 5 of	s 5	-	Topics 4 of 2	7 *		Clust Consumer Gro 3 of 26	er identified a from the a	4 topics select	data ted
Brokers	55 (4)	BROKERS (5)	NAME		DATA IN 🗸	DAT	User Action	ROUPS			30	minutes -	
	Producers (ACTIVE (34)	34) PASSIVE (0) ALL MESSAGES	gateway-euro	pe-raw-sensors	DATA IN 18MB	A IO DAT OB	been selecte	d Roups	6 9 Q 📼	Con All	SUMER GROUPS (3)	ALL	
 Consumer Groups 	minifi-eu-i1 minifi-eu-i2 minifi-eu-i4	25k 12k 6.3k	gateway-wes	t-raw-sensors	DATA IN 225KB	DATA OUT 225KB	MESSAGES IN 1.1k	CONSUMER GROUPS	<mark>6</mark> 9 Q 📼	✓ nifi nifi nifi	truck-sensors-west truck-sensors-east truck-sensors-central	2	
	minifi-eu-i6 minifi-eu-i3	4.2k 3.8k	gateway-cent	ral-raw-sensors	DATA IN 146KB	DATA OUT 146KB	MESSAGES IN 731	CONSUMER GROUPS	<u>6</u> e q m	~			
	minifi-eu-i7 minifi-eu-i8 minifi-eu-i9	3.6k 3.1k 2.8k	gateway-east	-raw-sensors	DATA IN 111KB	DATA OUT 111KB	MESSAGES IN 551	CONSUMER GROUPS	<mark>6</mark> e q m	~			
	minifi-eu-i10 minifi-eu-i5	2.5k											
	minifi-eu-i12 minifi-eu-i13	2 2.1k 8 1.9k											
	minifi-eu-i14 minifi-eu-i15 minifi-eu-i16	5 1.7k											
	minifi-eu-i12 minifi-eu-i18	7 1.5k 8 1.4k											
	minifi-eu-i19 minifi-eu-i20 minifi-eu-i22	2 1.3k 2 1.3k											

3. Click DATA IN to sort on data throughput-in across all topics. We see that gateway-europe-raw-sensor has significantly more data coming in than any other topic: 18 MB totaling 88K in the last 30 minutes. This addresses Use Case 3.

s. M	TREAMS Essaging	Overview			Step	01					Cluster: orlandostreamcluster - 🎄 -
🛛 🔹 Overview		3	Producers 34 of 83	• ⁸⁴ 5	Click on DAT	A IN to sort across all last 30 mins	Topic 4 of 2	27 ×		Consu 3	mer Groups of 26
III Brokers		TOPICS (4) BROKER	RS (5)								S 30 minutes -
Topics		Desducers (2.4)		NAME	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS			0
6 Producers	n	ACTIVE (34) PASSIV	/E (0) ALL	gateway-europe-raw-sensor	s DATA IN 18MB	DATA OUT OB	MESSAGES IN 88k	CONSUMER GROUPS	🧑 🛛 ପ୍ 🕅	~	ACTIVE (3) PASSIVE (0) ALL
Consumer		Analysis	MESSAGES								nifi-truck-sensors-west 2
	Kafka topic	called gateway-e	europe-	gateway-west-raw-sensors	DATA IN 225KB	DATA OUT 225KB	MESSAGES IN 1.1k	CONSUMER GROUPS	🧑 \varTheta Q 📼	~	nifi-truck-sensors-east 2
	raw-senso sent to it ti	ors has more data han any other top	being bic: 88K								nifi-truck-sensors-central 1
	messages t	otaling 18 MB in 1 30 mins	the last	gateway-central-raw-sensor	s DATA IN 146KB	DATA OUT 146KB	MESSAGES IN 731	CONSUMER GROUPS	🧑 🛛 Q 📼	~	
		minifi-eu-i7	3.6k								
		minifi-eu-i8	3.1k	gateway-east-raw-sensors	DATA IN 111KB	DATA OUT 111KB	MESSAGES IN 551	CONSUMER GROUPS	🧑 \varTheta Q 📼	~	
		minifi-eu-i9	2.8k								
		minifi-eu-i10	2.5k								
		minifi-eu-i5	2.3k								
		minifi-eu-i12	2.1k								
		minifi-eu-i13	1.9K								
		minifi-eu-i15	1.0K								
		minifi-eu-i16	1.6k								
		minifi-eu-i17	1.5k								
		minifi-eu-i18	1.4k								
		minifi-eu-i19	1.3k								
		minifi-eu-i20	1.3k								
	«	minifi-eu-i22	1.2k								

4. Expand the topic panel for gateway-europe-raw-sensor to get more details for the topic, including like partition layout. Click the topic to see which producers are sending data to each partition of that topic. We see that all the producers are trucks from the EU fleet. Also note that all five partitions for that topic have 0B going out and we see no data flowing from the topic to any consumer group. This could be worth investigating to

He								C	Step 1
æ	^{Pr} 34 Who are	Step 2 the Topic to see all the producers					-*	Con	Expand topic panel to see more details of the topic that has high data-in rates
	TOPICS (4) BRO								℃ 30 minutes -
	Producers (34)			DATA IN 🔫					Consumer Groups (3)
	ACTIVE (34) PASSIVE (0) ALL	gateway-e	europe-raw			MESSAGES IN 35m		<mark>⊚</mark> @Q≣ ^	ACTIVE (0) PASSIVE (3) ALL
-+3	MESSAGES								
	minifi-eu-i3 4.8m	Replication F							nifi-truck-sensors-e 2
	minifi-eu-i5 2.9m	1001	P0 268 in	0B out				1000	nifi-truck-sensors-c., 1
	minifi-eu-i9 1.6m	1001	PU 200 III	00.001				/	
	minifi-eu-i8 1.3m	1002	P1 658MB in	0B out					
	minifi-eu-i10 1m			_					
	minifi-eu-i15 1m	1003	PZ 3GB in	0B out			/	1001	
	minifi-eu-i12 0.8m	2 21001	P3 2GB in	0B out			Analy	sis	
	minifi-eu-i14 0.7m					No	te that for each	partition the	re
	minifi-eu-i21 0.7m	1002	P4 339MB in	0B out		is is see	no data going or no data going to	ut (OB) and w o any consun	re ner
	minifi-eu-i24 0.6m					gro	ups. This means	that while t	he
	minifi-eu-i18 0.6m					MESS	s no consumers	which could	
	minifi-eu-i20 0.5m	gateway-v	vest-raw-se	136MB	136MB	0.6n	indicate a p	robiem	
	minifi-eu-i22 0.5m								
5.	minifi-eu-i23 0.4m	gateway-c	entral-raw-s	DATA IN 86MB	DATA OUT 86MB	MESSAGES IN 0.4m	CONSUMER GROUPS	© @Q≡ ~	

identify why a topic with the most amount of producers has no consumers. This addresses Use Cases 4 and

5. Click another topic called gateway-west-raw-sensors that has consumers. This topic has three producers sending data to it and a NiFi consumer consuming from it. We also see that two of the four partitions for this topic have no data in them, which indicates that there is partition skew issue. This addresses Use Cases 6 and



6. Click the explorer/magnifying glass icon to search for events in the selected Kafka topic. This addresses Use Case

Producers 34 of 83		+	Brokers 5 of 5		- Topics - X 4 of 27		Consumer Groups 3 of 26		- Clear			
OPICS (4) BROKERS	S (5)									ත 30 minutes +	Click on the	explorer icon
roducers (34)		NAME		DATA IN 💌	DATA OUT	MESSAGES IN	CONSUMER GROUPS		Consumer Gro	ups (3)	the Ka	fka Topic
ACTIVE (34) PASSIVE	E (0) ALL	gateway-europe-	raw-sensors	18MB	OB	MESSAGES IN 88k	CONSUMER GROUPS	6 Q 🖬 🗸	ACTIVE (3)	PASSIVE (0)		
minifi-eu-i1	MESSAGES 25k									s-west 2		
minifi-eu-i2	12k	gateway-west-raw-sensors		DATA IN 225KB	DATA OUT	MESSAGES IN CONSU	CONSUMER GROUPS	6 9 Q III ^	 nifi-truck-senso 	rs-east 2	1	
minifi-eu-i4	6.3k	-	22360		1.16			nifi-truck-sense	rs-central 1			
minifi-eu-i6	4.2k	Replication Factor: (2	() InSync Replicas	8 Of 8 Total mess								
minifi-eu-i3	3.8k	Data Explorer										
minifi-eu-i7	3.6k	1001 P0	<u>uum</u>	UB COL							n Marrielan	Maluary Otalan
minifi-eu-i8	3.1k	☑1002 P1	08 in	06 out						DESERIALIZE	R: Keys: String *	values: string
minifi-eu-i9	2.8k	1000			1		FROM OFFSET					TO OFFSI
minifi-eu-i10	2.5k	V1003 P2	88K8 in	88K8 out	Partition	2 •	275851					27586
minifi-eu-i5	2.3k	21004 P3	137K8 in	137KB out			0	45976	91952	137928	183904 229880	275860
minifi-eu-i12	2.1k				Offset -	Timestamp	Value					
minifi-eu-i13	1.9k				275850	Tue, Jul 31 2018,	5:59:32 2018-07-31	2:59:32.679 153307	7972679 truck_geo_e	vent 455 10 George Ve	tticaden 10 Saint Louis to Tulsa Nor	nal 37.8 -92.48 1
minifi-eu-i14	1.8k	gateway-central-raw-sensors		146KB	275851 Tue, Jul 31 2018, 5:59:32 2018-07-31 22:59:32.68(1533077972680(truck_speed_event)455(10)George Vetticaden(10)Saint Louis to Tulsa(72							
minifi-eu-i16	1.78		275852 Tue, Jul 31 2018, 6:59:37 2018-07-31 22:59:37.63(1533077977630(truck_geo_event)455)10(George Vetticaden(10)Saint Louis to Tulsa)Normal(37.81)-92.31(1									
minifi-eu-i17	1.5k	gateway-east-raw-sensors		DATA IN	275853 Tue, Jul 31 2018, 5:59:37 2018-07-31 22:59:37.631/1533077977631/truck_speed_event[455/10]George Vetticaden[10]Saint Louis to Tulsa]68							1
minifi-eu-i18	1.4k			111KB	275854 Tue, Jul 31 2018, 5:59:41 2018-07-31 22:59:41.469(1533077981469)truck_geo_event/455(10)George Vetticaden(10)Saint Louis to Tulsa(Normal)37.81(-92.08)1							
minifi-eu-i19	1.3k				275855	Tue, Jul 31 2018,	5:59:41 2018-07-31 2	2:59:41.471 153307	7981471 truck_speed	_event 455 10 George1	Vetticaden(10)Saint Louis to Tulsa(6)	j
minifi-eu-i20	1.3k				275856	Tue. Jul 31 2018	5:59:45 2018-07-31 1	2:59:45.76/1533077	985760ltruck geo ev	enti455i10iGeorge Vett	icadeni10(Saint Louis to TulsalNorm	ali37.94i-91.99i1
minifi-eu-i22	1.2k				275857	Tue, Jul 31 2018,	5:59:45 2018-07-31	2:59:45.76 1533077	985760[truck_speed_	event 455 10 George Vi	etticaden 10 Saint Louis to Tulsa 73	
					275858	Tue, Jul 31 2018,	5:59:50 2018-07-31	2:59:50.669 153307	7990669 truck_geo_e	vent 455 10 George Ve	tticaden 10 Saint Louis to Tulsa Nor	nal 37.99 -91.69 1

7. Click the Atlas icon to see the metadata and lineage of the Kafka topic in Atlas. This addresses Use Case



9.