Fast-Track Machine Learning Operationalization Through Streaming Integration

May 2018

Speakers and Agenda

Changsha Ma

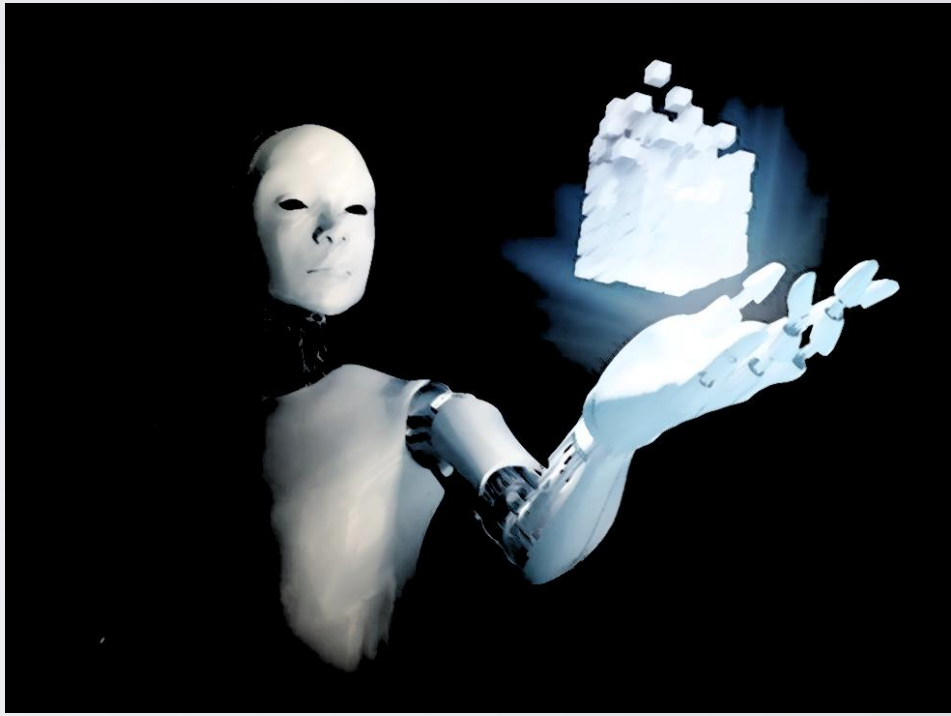
Codin Pora

Steve Wilkes

- Background
- Introduction To Striim
- Lab : Network Intrusion Detection System
- Q&A

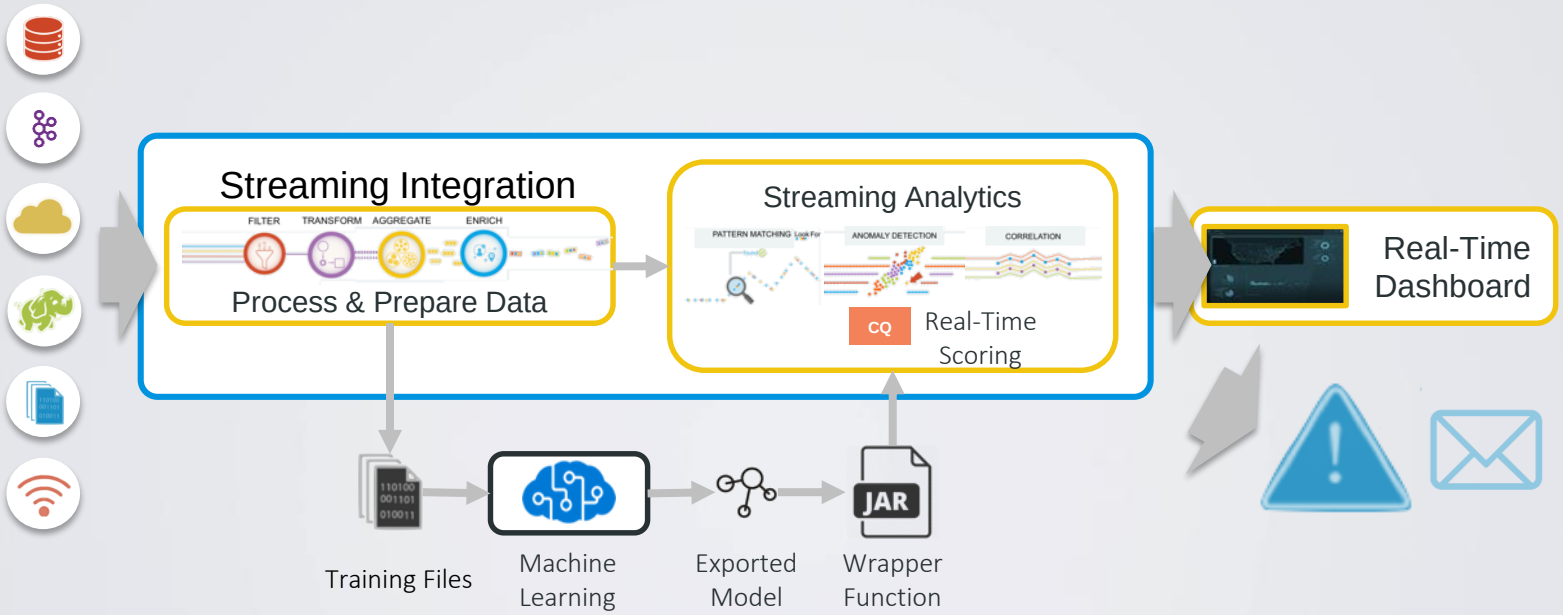
Operational Machine Learning Facts

- The value of ML is based on the real-time handling of high data volume, velocity, and variety at scale
- Intensive real-time pre-processing and feature extraction is required before feeding raw data into models
- Static models cannot fit dynamic data in operational systems even they are fine-tuned offline
- Operational systems demand continuous insights from model serving and minimal human intervention



striim

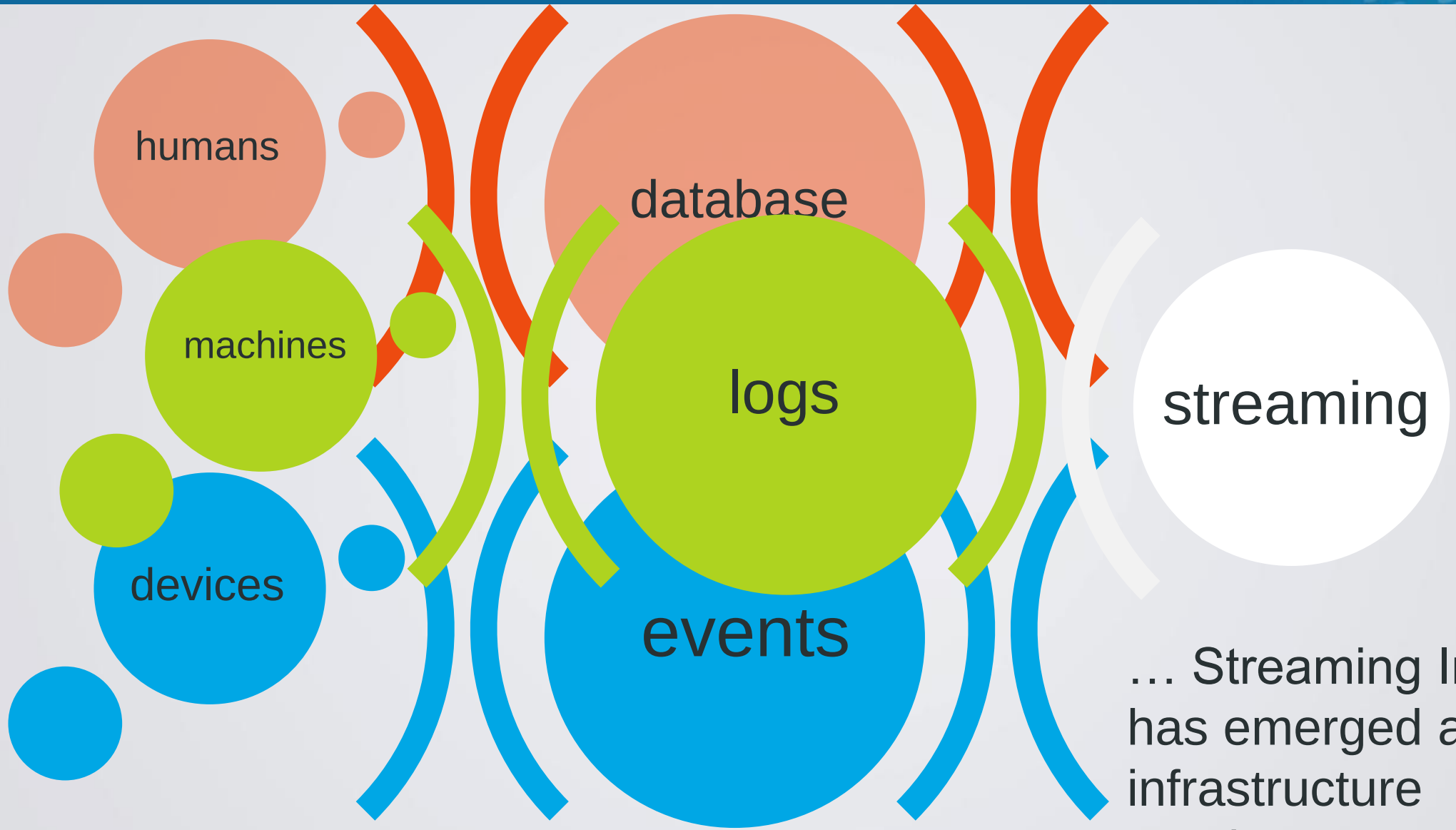
Striim Solution of Fast-Track ML Operationalization



- Filter, enrich and otherwise prepare streaming data
- Land data continuously, in an appropriate format for training a machine learning model
- Handle model lifecycles, enabling retraining if the model no longer fits the data
- Integrate a trained model into the real-time data stream to make continuous predictions
- Visualize the real-time data and associated predictions, and alert on issues

striim

Data Arrives in Streams - Not in Batches



… Streaming Integration has emerged as a major infrastructure requirement

striim

Striim Is Built For Streaming Integration

Handling extreme volumes of data at scale with high throughput

Processing, analyzing, and correlating data in flight

Making data valuable, verifiable, and visible in real time

striim

Why Are Companies Using Streaming Integration?

	**Data Distribution and Consistency**	• Kafka Integration and Processing • RDMS Integration • Data Grid Cache Consistency
	**Cloud Adoption**	• Zero Downtime Migration to Cloud • Continuous Real-Time Data Integration
	**Integration for Analytics**	• Hadoop Integration • Operational DS/DW Integration • Offloading Reporting
	**Real-Time Analytics**	• Next-Generation Analytics • Real-Time Alerts • Data Visualization
	**Internet of Things**	• IoT Edge Processing and Analytics

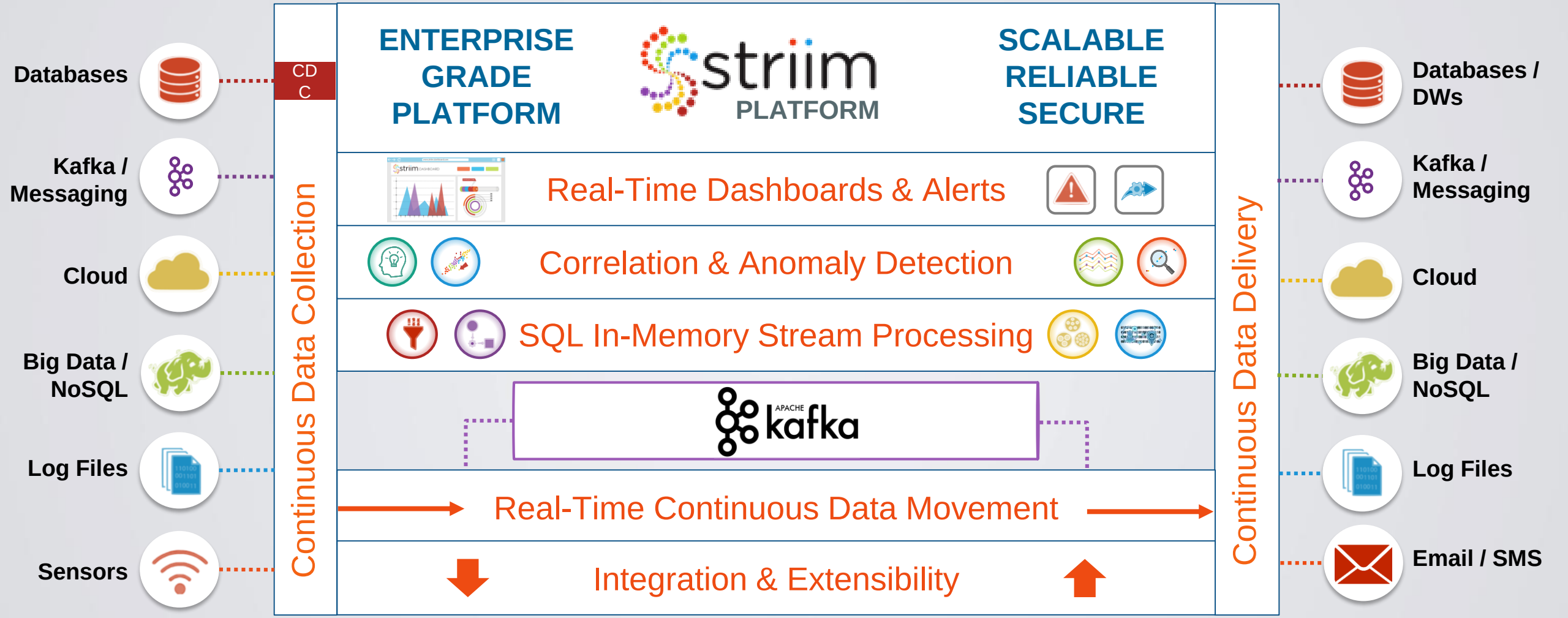
Customers Are Modernizing

Moving to a “streaming first” data architecture, supporting cloud, streaming analytics, and IoT.

Bridge the old and new worlds of data.

Streaming Integration is the foundation for data modernization initiatives.

Striim Platform for Streaming Integration with Intelligence



DATA IS BORN → **REAL-TIME, CONTINUOUS, HETEROGENEOUS INTEGRATION** → **CONTINUOUS DELIVERY**

Streaming Integration with Intelligence

CONTINUOUS DATA INGESTION

Parsers

- Delimited
- JSON
- XML
- Free Text
- Binary
- Name/Value
- Zipped
- AVRO
- GoldenGate
- Apache Log
- Sys Log
- MS Event Log
- Mail Log
- SNMP
- CollectD
- CEF
- DHCP Log
- WCF
- +Others

DBs
- Oracle CDC
- MS SQL CDC
- MySQL CDC
- HPE NSK CDC
- Maria DB CDC
- JDBC/SQL

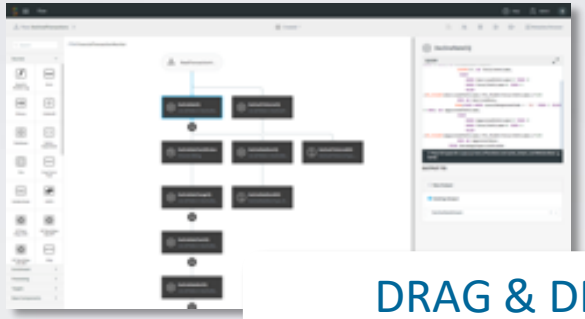
Cloud
- Amazon S3
- AWS RDS
- Salesforce

Files
- Log Files
- System Files
- Batch Files

Network
- TCP
- UDP
- HTTP
- MQTT
- Netflow
- PCAP
- OPC-UA

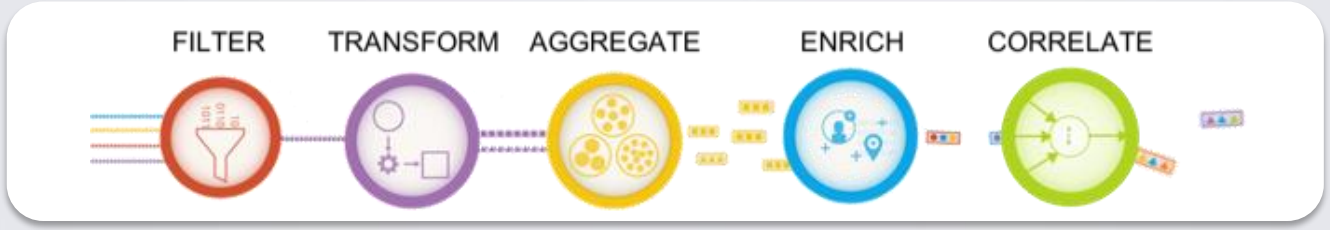
Messaging
- Kafka
- Flume
- JMS
- AMQP

Big Data
- HDFS
- HBase
- Hive



DRAG & DROP UI

STREAM PROCESSING WITH INTELLIGENCE



REAL-TIME VISUALIZATION

CONTINUOUS DATA DELIVERY

DBs/DWs
- Oracle
- MS SQL
- MySQL
- Teradata
- PostgreSQL
- MemSQL
- JDBC/SQL

Files

Network
- MQTT
- OPC-UA

Messaging
- Kafka
- JMS
- AMQP

Big Data
- HDFS
- HBase
- Hive
- Hazelcast

Cloud
- Azure Blob
- Azure SQL DB
- Azure HDInsight
- Amazon S3
- Amazon Redshift
- Amazon Kinesis
- Google Big Query

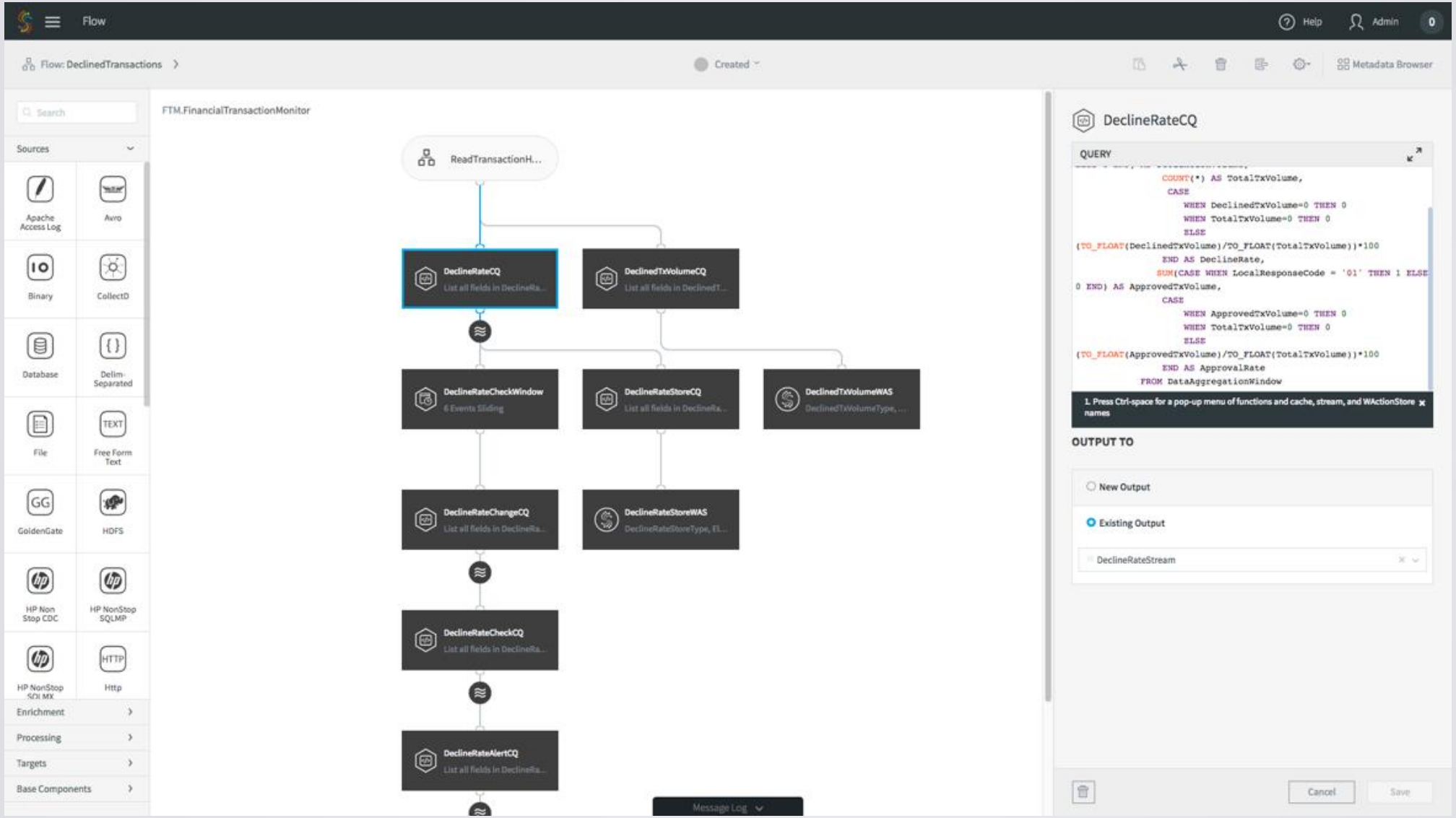
Alerting
- Email
- SMS

Formats

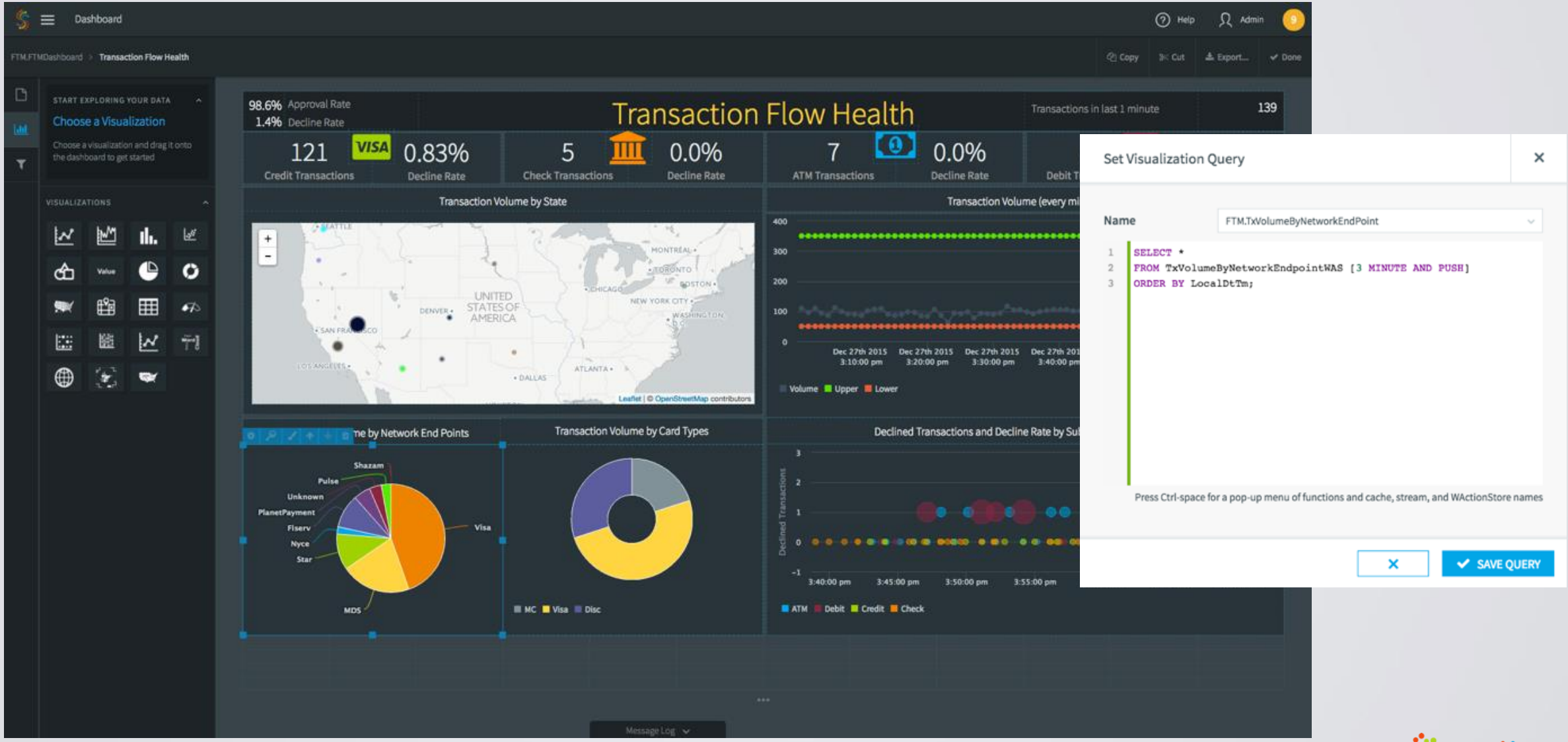
- Delimited
- JSON
- XML
- AVRO
- Template

striim

Integration and Analytics Through Data Flows

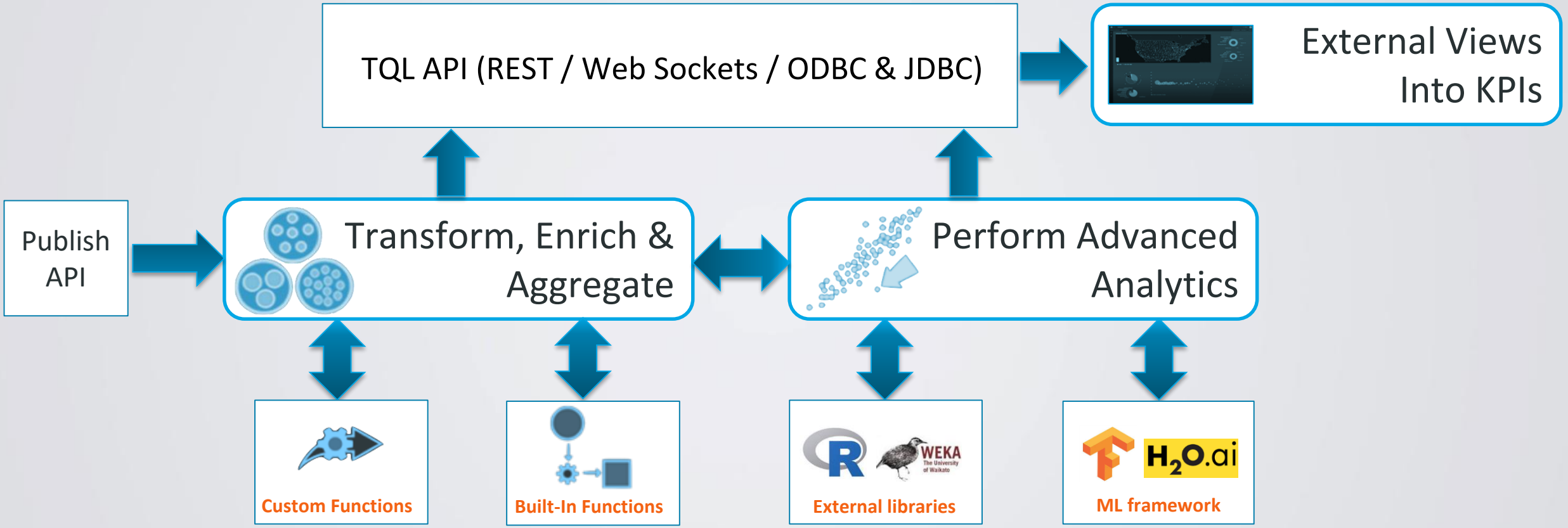


Visualization Through Streaming Dashboards



striim

Striim – Integration Overview

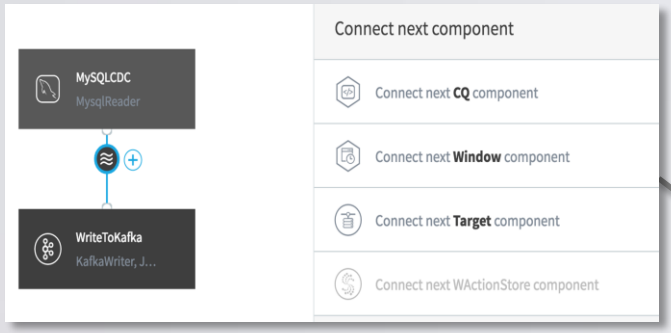


Enterprise Grade

Clustered, Distributed, Scalable, Reliable and Secure

Striim Application

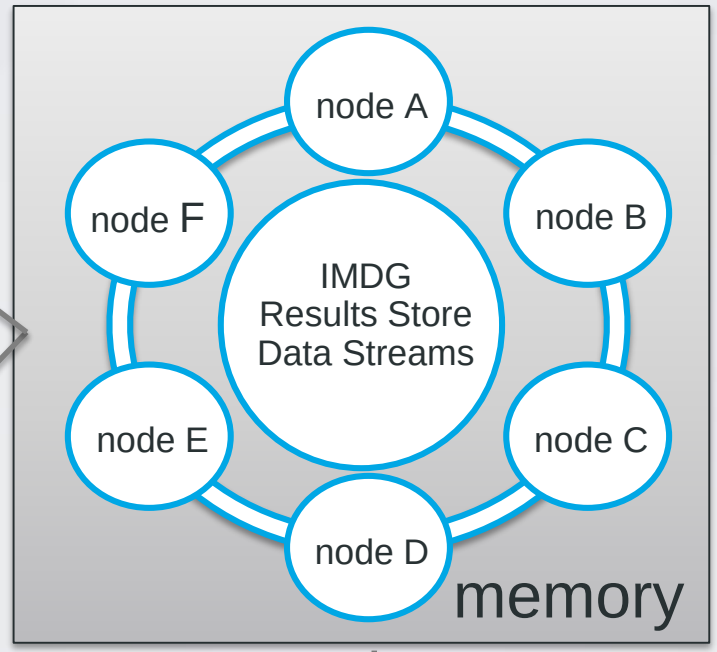
Flow Designer UI

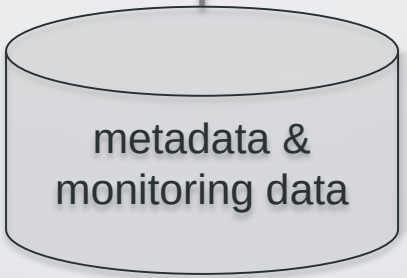


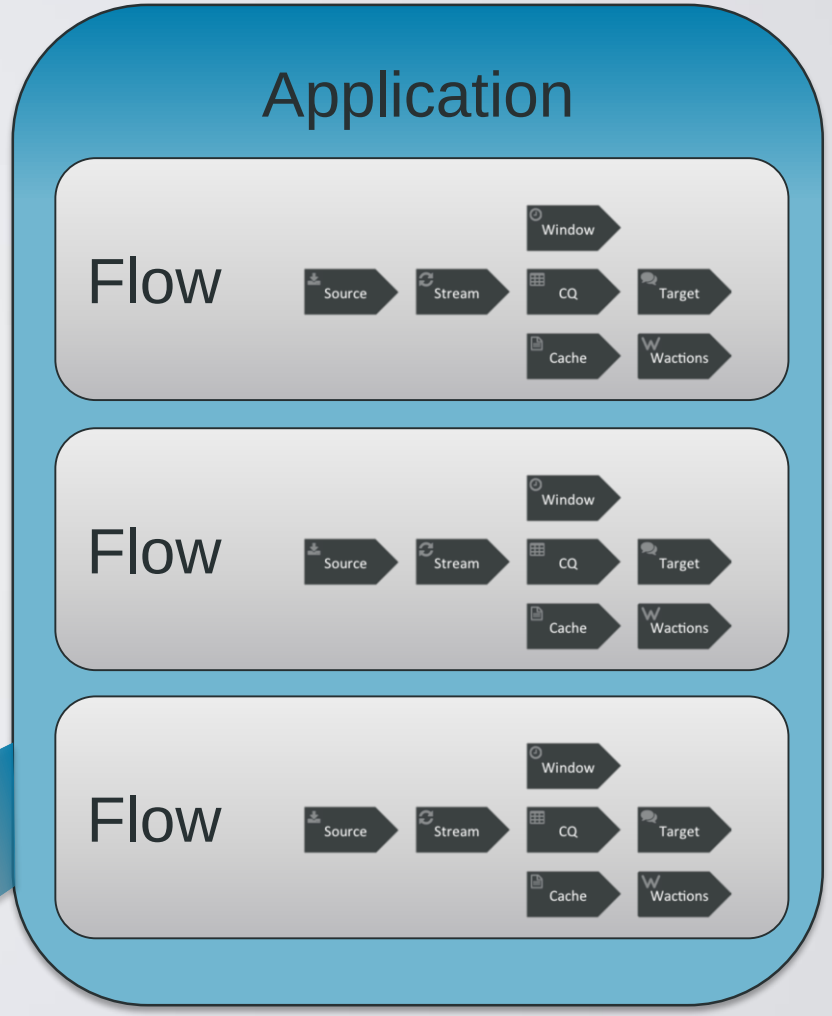
Tungsten console

```
CREATE APPLICATION MultiLogApp;
CREATE FLOW MonitorLogs;
CREATE SOURCE AccessLogSource USING…
CREATE TYPE AccessLogEntry …
CREATE STREAM AccessStream OF…
CREATE CQ ParseAccessLog …
W >
```

Striim cluster







striim

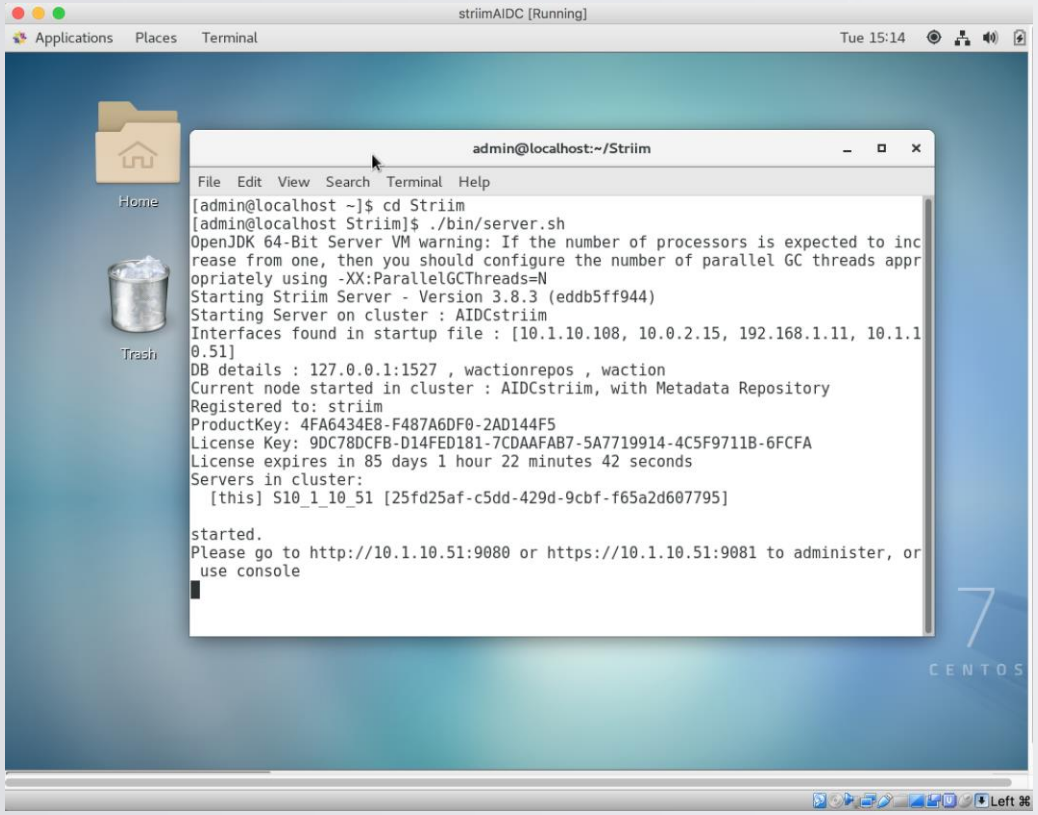
Core Striim Components

 Source	Source accesses external data and provides real-time continuous events into streams
 Stream	Stream carries data between components and nodes
 Type	Type is a named set of fields, each has a name and a data type, such as *Integer* or *String*
 Window	Window provides moving snapshot/collection of events for aggregates and models
 Cache	Cache is external contextual data made available using distributed in-memory grid
 CQ	A Continuous Query emits big data records after processing real-time streaming events (can process data from streams, windows, caches, event tables, and stores)
 Target	Target outputs real-time big data records to external systems
 Results	A Results Store contains results of processing within the Striim server, defines the data allowed in the store and how it is persisted to a backend system (default ES)

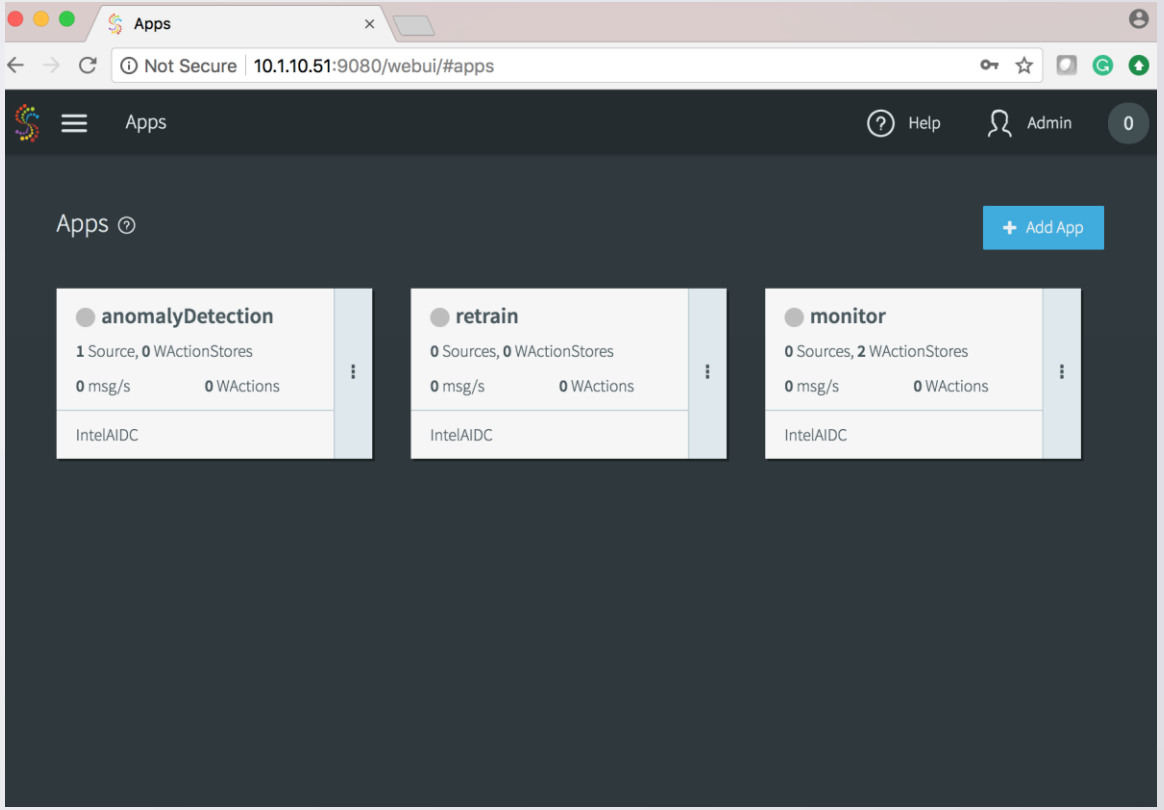
striim

Lab: Network Intrusion Detection System (NIDS)

Start Striim server in virtual machine striimAIDC



Navigate to Striim Web UI in your host/VM browser



Note: if the URL is not accessible from you host browser, go to *Applications -> Sundry -> Firewall -> Options -> Change default zone* to *trusted*.

Network Intrusion Detection System (NIDS)

- Data:
  - Network flows with robust features from tcpdump analyzer
- Tasks:
  - Detect abnormal flows with low false positive rate
  - Automatically adapt model serving to data evolution
  - Continuously monitor system and alert on issues in real time
- Algorithms
  - One-Class SVM (Weka LibSVM)
  - Time series spike detection

striim

NIDS Data

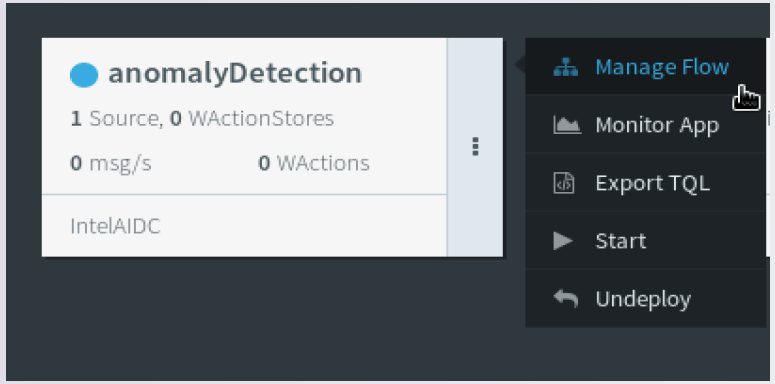
Name	Type	Description
srcip	nominal	Source IP address
sport	integer	Source port number
dstip	nominal	Destination IP address
dsport	integer	Destination port number
proto	nominal	Transaction protocol
state	nominal	Indicates to the state and its dependent protocol
dur	Float	Record total duration
sbytes	Integer	Source to destination transaction bytes
dbytes	Integer	Destination to source transaction bytes
sttl	Integer	Source to destination time to live value
dttl	Integer	Destination to source time to live value
sloss	Integer	Source packets retransmitted or dropped
dloss	Integer	Destination packets retransmitted or dropped
service	nominal	http, ftp, smtp, ssh, dns, ftp-data ,irc
Sload	Float	Source bits per second
Dload	Float	Destination bits per second
Spkts	integer	Source to destination packet count
Dpkts	integer	Destination to source packet count
swin	integer	Source TCP window advertisement value
dwin	integer	Destination TCP window advertisement value
stcpb	integer	Source TCP base sequence number
dtcpb	integer	Destination TCP base sequence number
smeansz	integer	Mean of the packet size transmitted by the src
dmeansz	integer	Mean of the packet size transmitted by the dst
trans_depth	integer	Represents the pipelined depth into the connection
Sjit	Float	Source jitter (mSec)

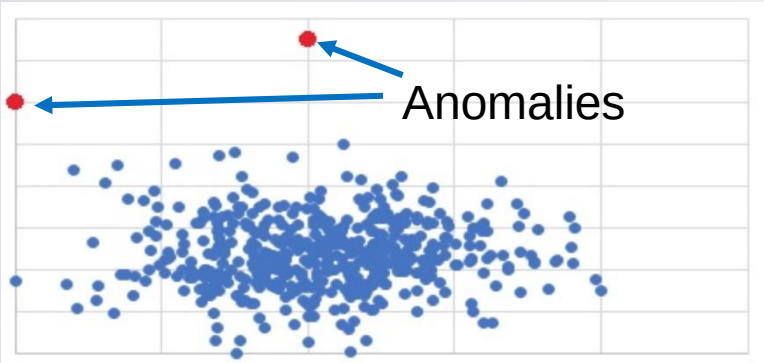
Name	Type	Description
Djit	Float	Destination jitter (mSec)
res_bdy_len	integer	Actual uncompressed content size of the data transferred from the server's http service.
Sintpkt	Float	Source interpacket arrival time (mSec)
Dintpkt	Float	Destination interpacket arrival time (mSec)
tcprtt	Float	TCP connection setup round-trip time, the sum of 'synack' and 'ackdat'.
synack	Float	TCP connection setup time, the time between SYN and SYN_ACK.
ackdat	Float	TCP connection setup time, the time between SYN_ACK and ACK.
is_sm_ips_ports	Binary	If source and destination IP addresses equal and port numbers equal then, this variable takes value 1 else 0
ct_state_ttl	Integer	No. for each state according to specific range of values for source/destination time to live.
ct_flw_http_mthd	Integer	No. of flows that has methods such as Get and Post in http service.
is_ftp_login	Binary	If the ftp session is accessed by user and password then 1 else 0.
ct_ftp_cmd	integer	No of flows that has a command in ftp session.
ct_srv_src	integer	No. of connections that contain the same service and source address in 100 connections according to the last time.
ct_srv_dst	integer	No. of connections that contain the same service and destination address in 100 connections according to the last time.
ct_dst_ltm	integer	No. of connections of the same destination address in 100 connections.
ct_src_ ltm	integer	No. of connections of the same source address in 100 connections.
ct_src_dport_ltm	integer	No. of connections of the same source address and destination port in 100 connections.
ct_dst_sport_ltm	integer	No. of connections of the same destination address and the source port in 100 connections.
ct_dst_src_ltm	integer	No. of connections of the same source and destination address in in 100 connections.

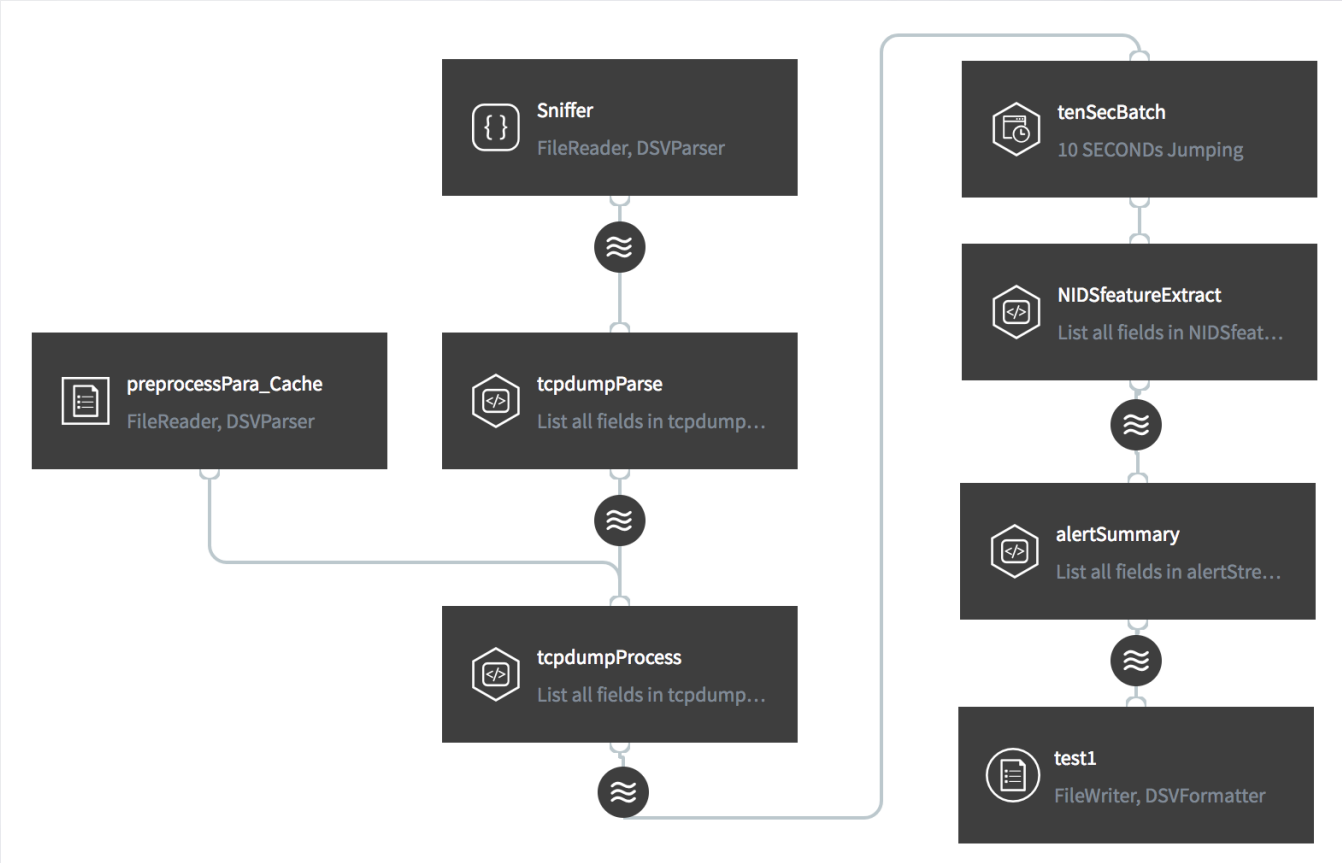
striim

NIDS Task 1: Anomaly Detection on Network Flows

Deploy anomalyDetection application, and view Flow

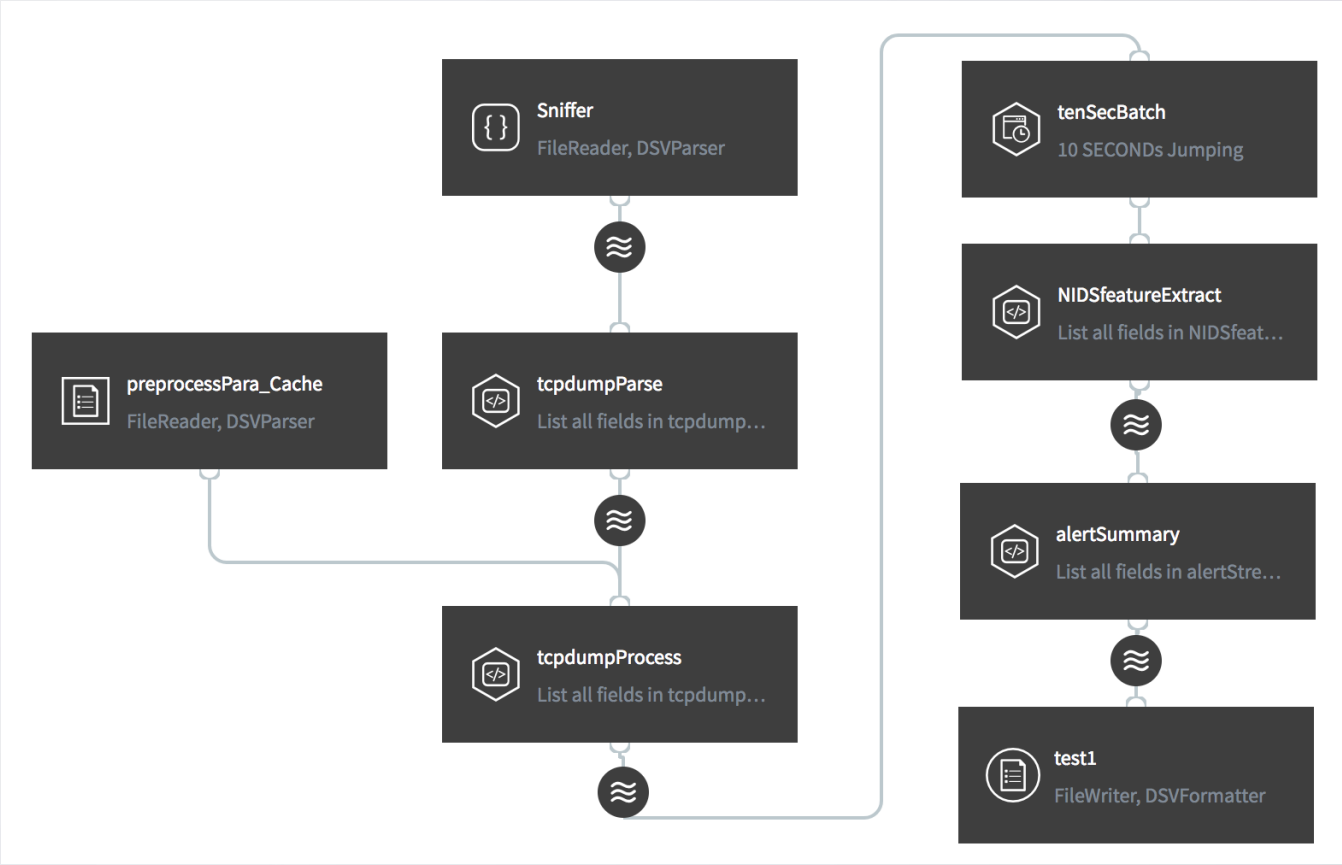






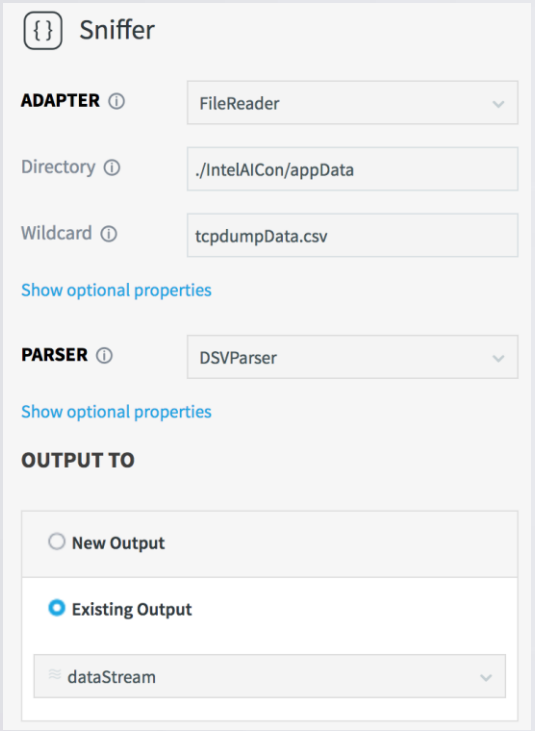
NIDS Task 1: Anomaly Detection on Network Flows

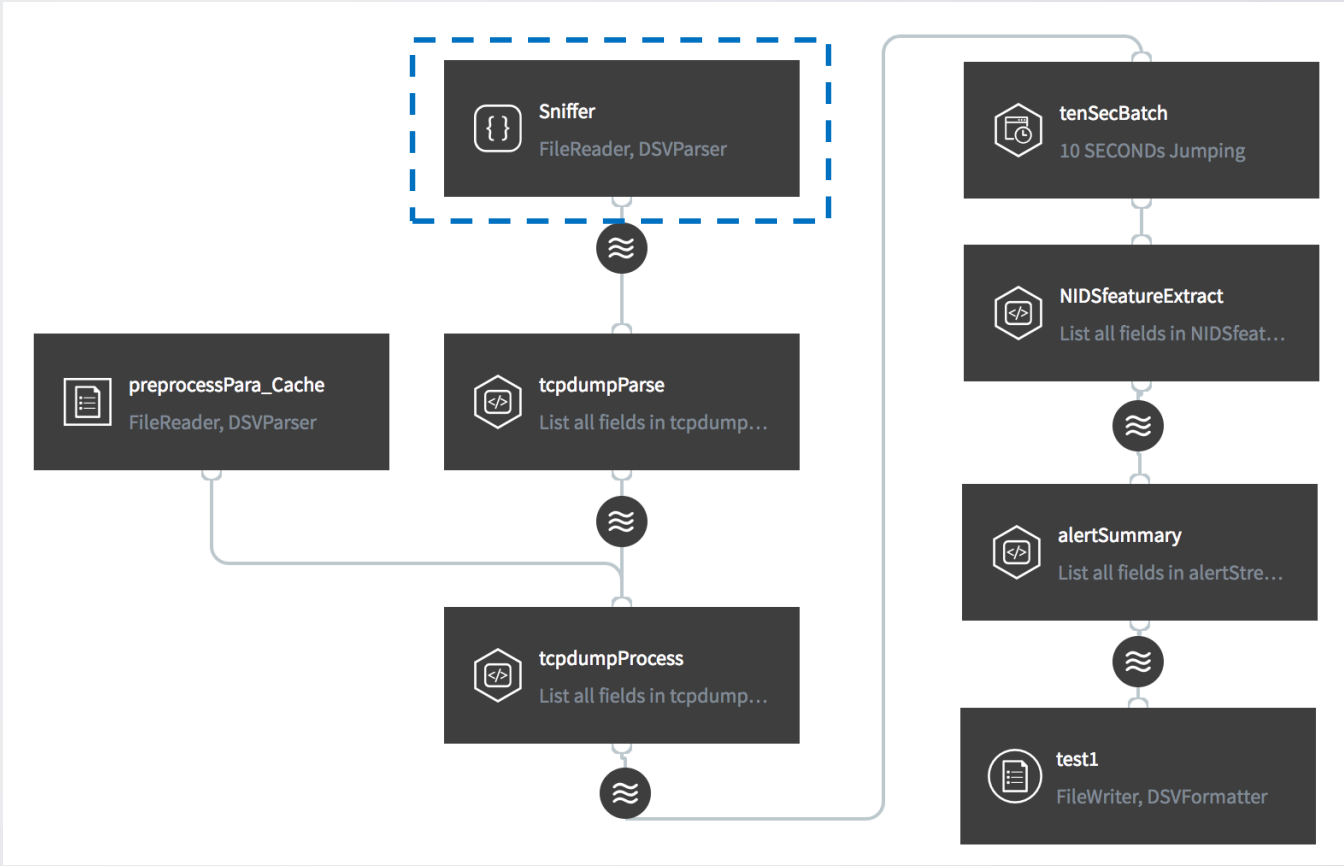
- Ingest data
- Filter data fields
- Preprocess raw data
- Aggregate events
- Extract features
- Detect anomalies
- Persist results



NIDS Task 1: Anomaly Detection on Network Flows

- Ingest data

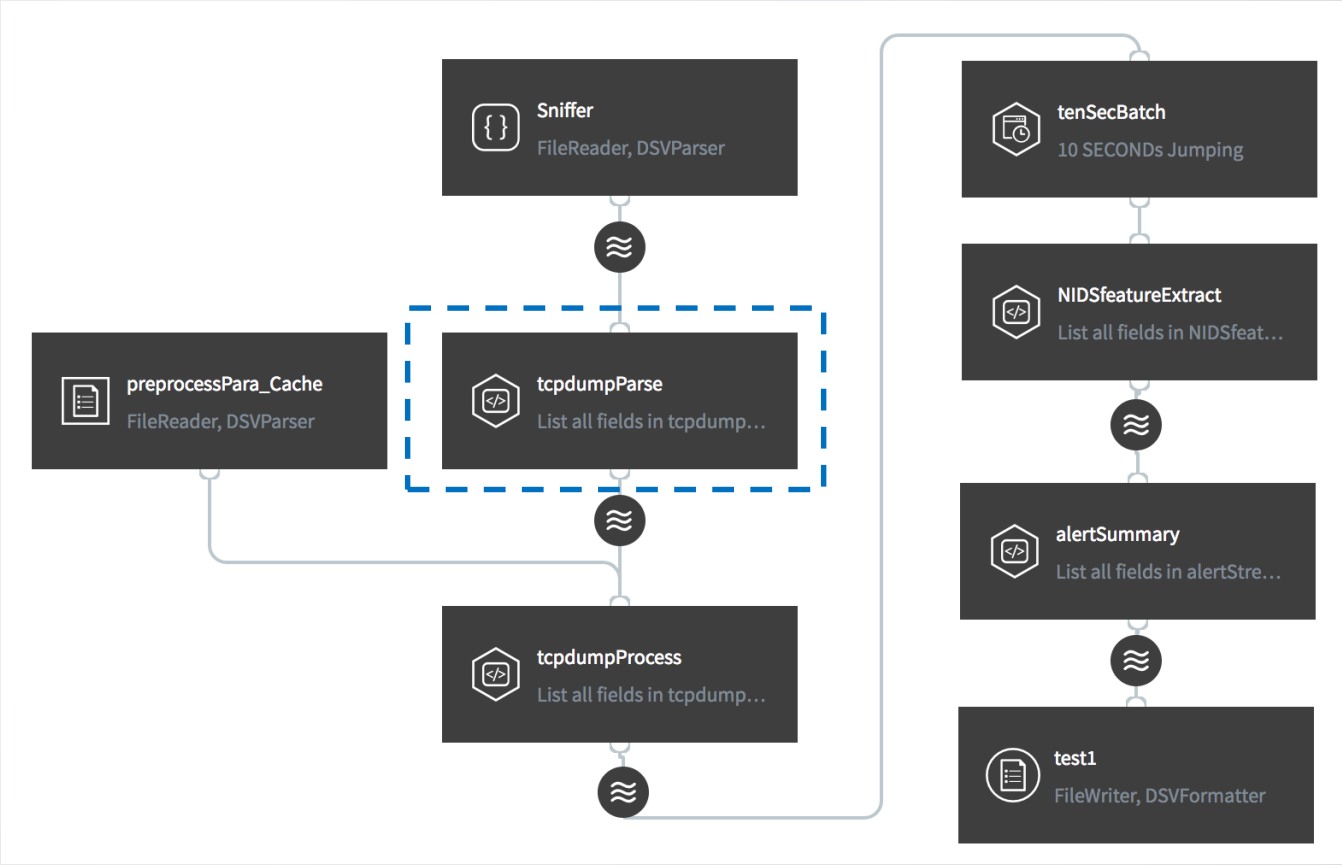




NIDS Task 1: Anomaly Detection on Network Flows

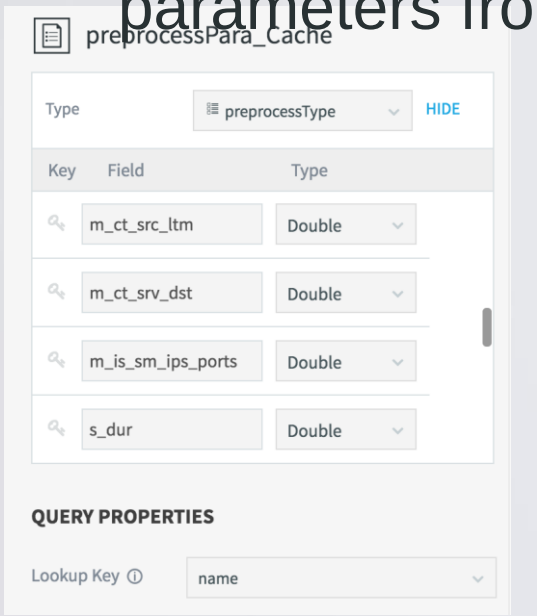
- Filter data fields

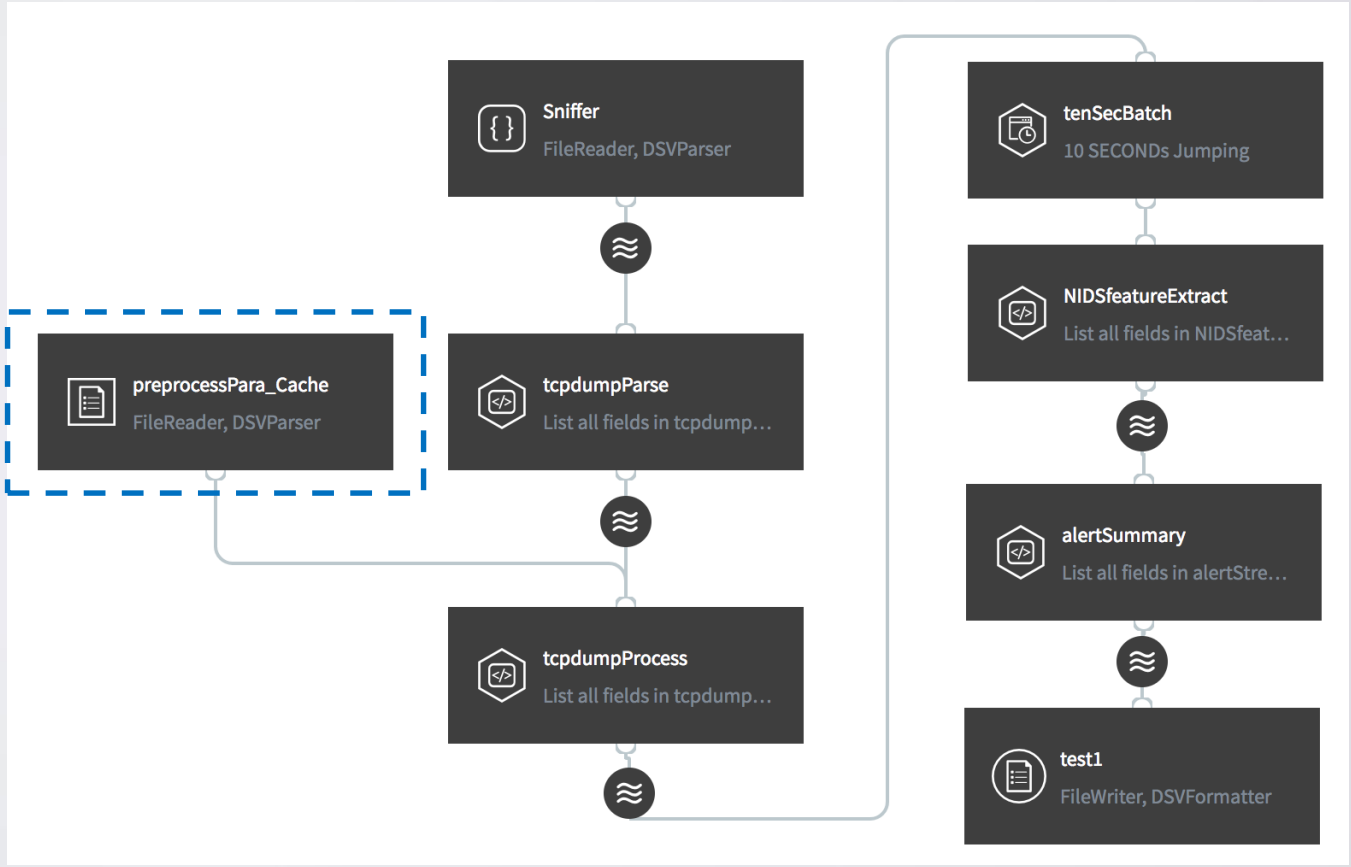
```
SELECT  "NIDS",TO_DATE(TO_LONG(data[0])*1000),
TO_STRING(data[1]), TO_STRING(data[2]),
TO_Double(data[3]),TO_STRING(data[4]),TO_STRING(data[5]),
TO_STRING(data[6]),TO_Double(data[7]),TO_Double(data[8]),
TO_Double(data[9]),TO_Double(data[10]),TO_Double(data[11]),
TO_Double(data[12]),TO_Double(data[13]),TO_Double(data[14]),
TO_Double(data[15]),TO_Double(data[16]),TO_Double(data[17]),
TO_Double(data[18]),TO_Double(data[19]),TO_Double(data[20]),
TO_Double(data[21]),TO_Double(data[22]),TO_Double(data[23]),
TO_Double(data[24]),TO_Double(data[25]),TO_Double(data[26]),
TO_Double(data[27]),TO_Double(data[28]),TO_Double(data[29]),
TO_Double(data[30]),TO_Double(data[31]),TO_Double(data[32]),
TO_Double(data[33]),TO_Double(data[34]),TO_Double(data[35]),
TO_Double(data[36]),TO_Double(data[37]),TO_Double(data[38]),
TO_Double(data[39]),TO_Double(data[40]),TO_Double(data[41]),
TO_Double(data[42]),TO_Double(data[43]),TO_Double(data[44])
FROM dataStream c WHERE PAUSE(15000L, c)
```



NIDS Task 1: Anomaly Detection on Network Flows

- Preprocess raw data
  - Enrich stream with standardization parameters from cache





NIDS Task 1: Anomaly Detection on Network Flows

- Preprocess raw data

tcpdumpProcess

QUERY

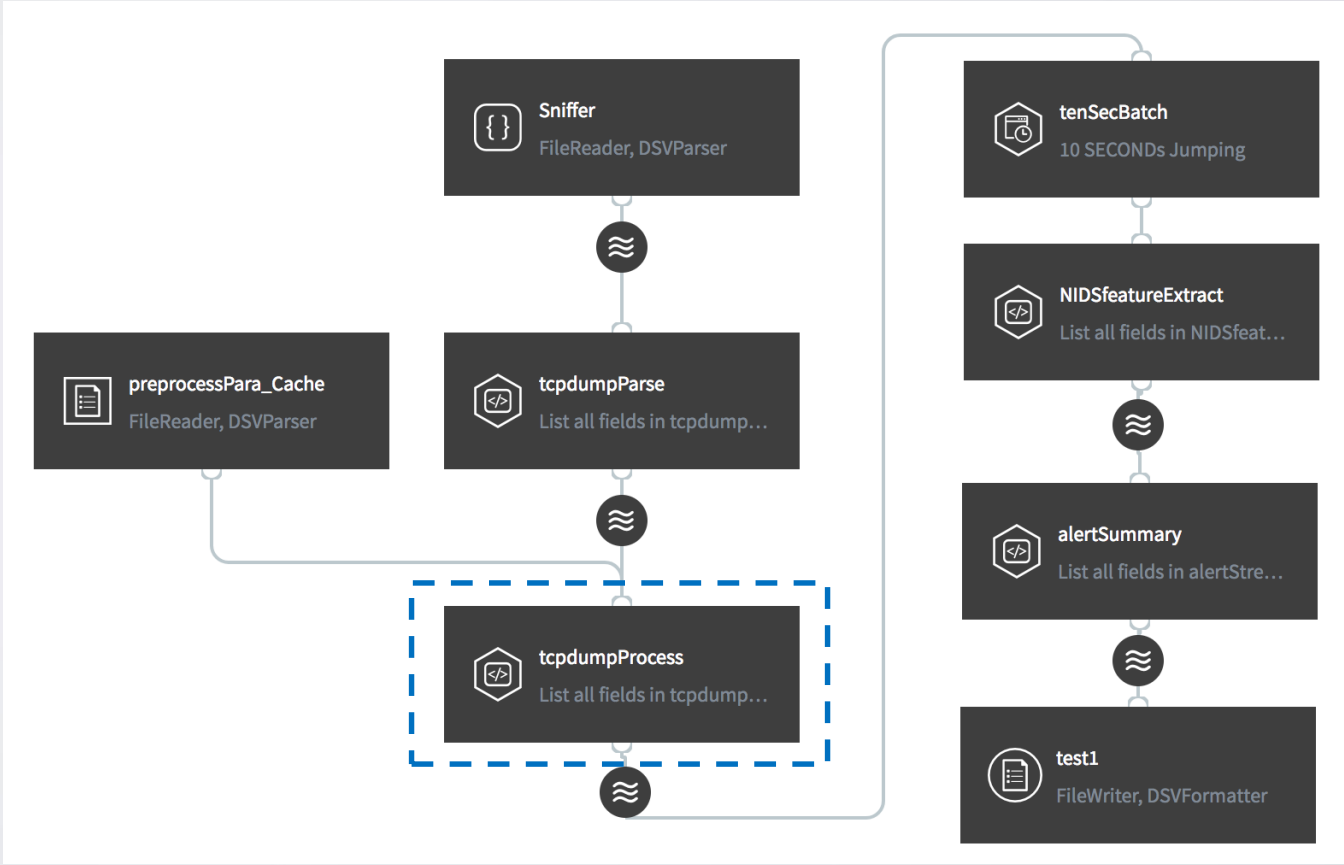
```
select
s.name, s.datetime, s.src,s.dst,
(s.dur-p.m_dur)/p.s_dur,
s.proto, s.service, s.state,
(s.spkts-p.m_spkts)/p.s_spkts,
(s.dpkts-p.m_dpkts)/p.s_dpkts,
(s.sbytes-p.m_sbytes)/p.s_sbytes,
(s.dbytes-p.m_dbytes)/p.s_dbytes,
(s.rate-p.m_rate)/p.s_rate,
(s.sttl-p.m_sttl)/p.s_sttl,
(s.dttl-p.m_dttl)/p.s_dttl,
(s.sload-p.m_sload)/p.s_sload,
(s.dload-p.m_dload)/p.s_dload,
(s.sloss-p.m_sloss)/p.s_sloss,
(s.dloss-p.m_dloss)/p.s_dloss,
(s.sinpkt-p.m_sinpkt)/p.s_sinpkt,
```

OUTPUT TO

New Output

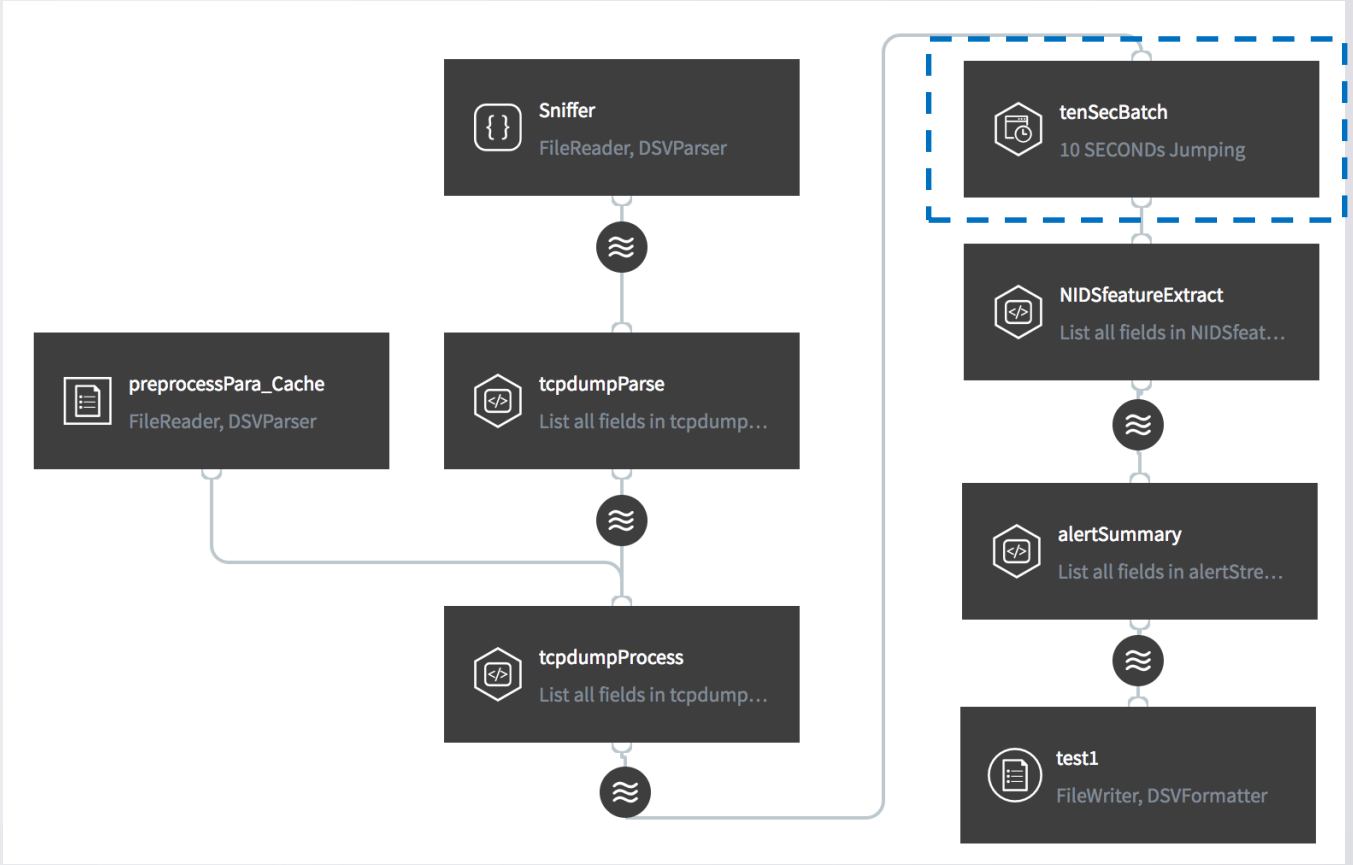
Existing Output

tcpdumpProcessStream



NIDS Task 1: Anomaly Detection on Network Flows

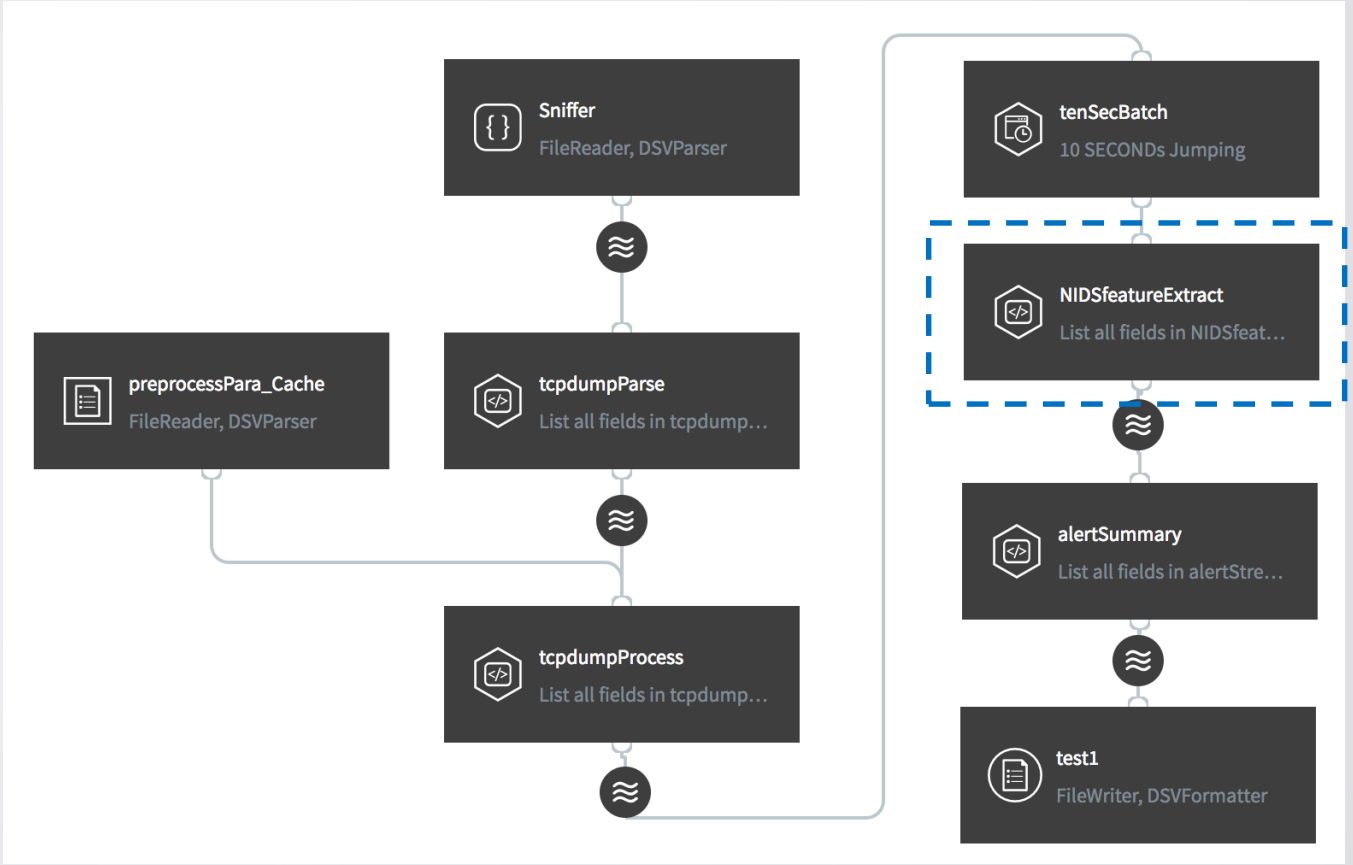
- Aggregate events
  - 10 seconds as observation interval
  - Two adjacent points in time series have 10-second time difference



NIDS Task 1: Anomaly Detection on Network Flows

- Extract features

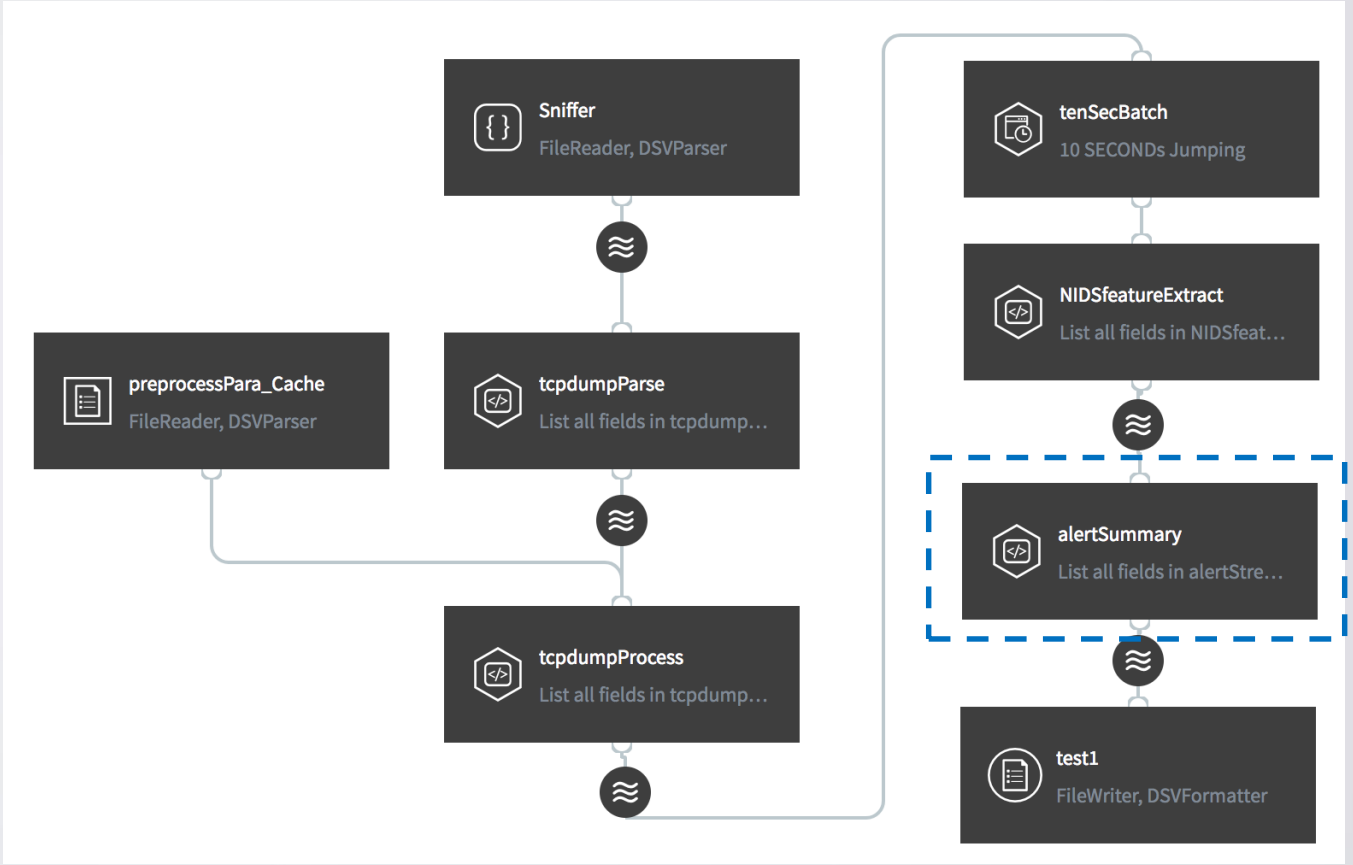
```
select last(datetime) as datetime, last(name) as name,
count(datetime) as total, list(src) as src, list(dst) as dst, list(dur) as f1,
list(proto) as f2, list(service) as f3, list(state) as f4, list(spkts) as f5,
list(dpkts) as f6, list(sbytes) as f7, list(dbytes) as f8, list(rate) as f9,
list(sttl) as f10, list(dttl) as f11, list(sload) as f12, list(dload) as f13,
list(sloss) as f14, list(dloss) as f15, list(sinpkt) as f16, list(dinpkt) as
f17, list(sjit) as f18, list(djit) as f19, list(swin) as f20, list(stcpb) as f21,
list(dtcpb) as f22, list(dwin) as f23, list(tcprtt) as f24, list(synack) as
f25, list(ackdat) as f26, list(smean) as f27, list(dmean) as f28,
list(trans_depth) as f29, list(response_body_len) as f30,
list(ct_srv_src) as f31, list(ct_state_ttl) as f32, list(ct_dst_ltm) as f33,
list(ct_src_dport) as f34, list(ct_dst_sport) as f35, list(ct_dst_src_ltm)
as f36, list(is_ftp_login) as f37, list(ct_ftp_cmd) as f38,
list(ct_flw_http_mthd) as f39, list(ct_src_ltm) as f40, list(ct_srv_dst) as
f41, list(is_sm_ips_ports) as f42
from tenSecBatch
```



NIDS Task 1: Anomaly Detection on Network Flows

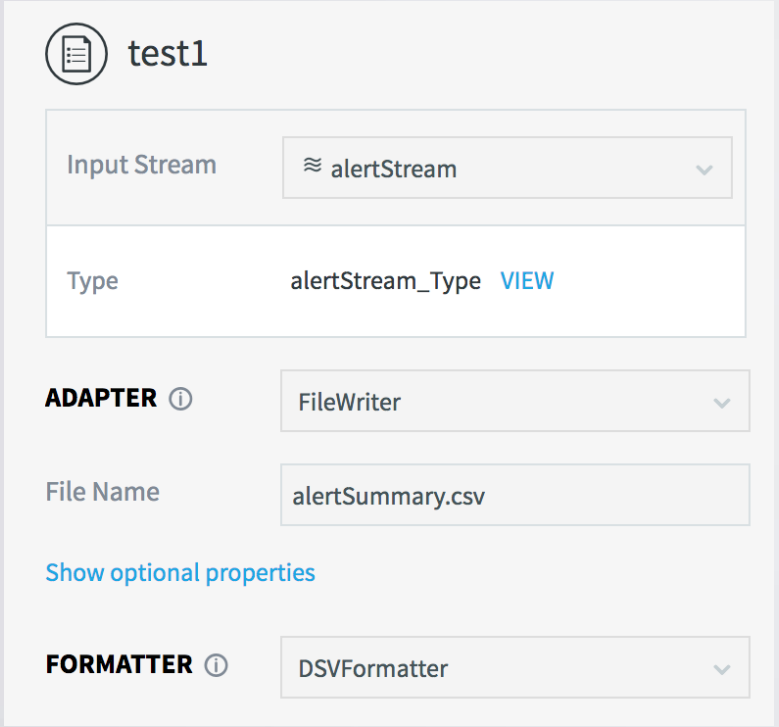
- Detect anomalies
  - One-class SVM

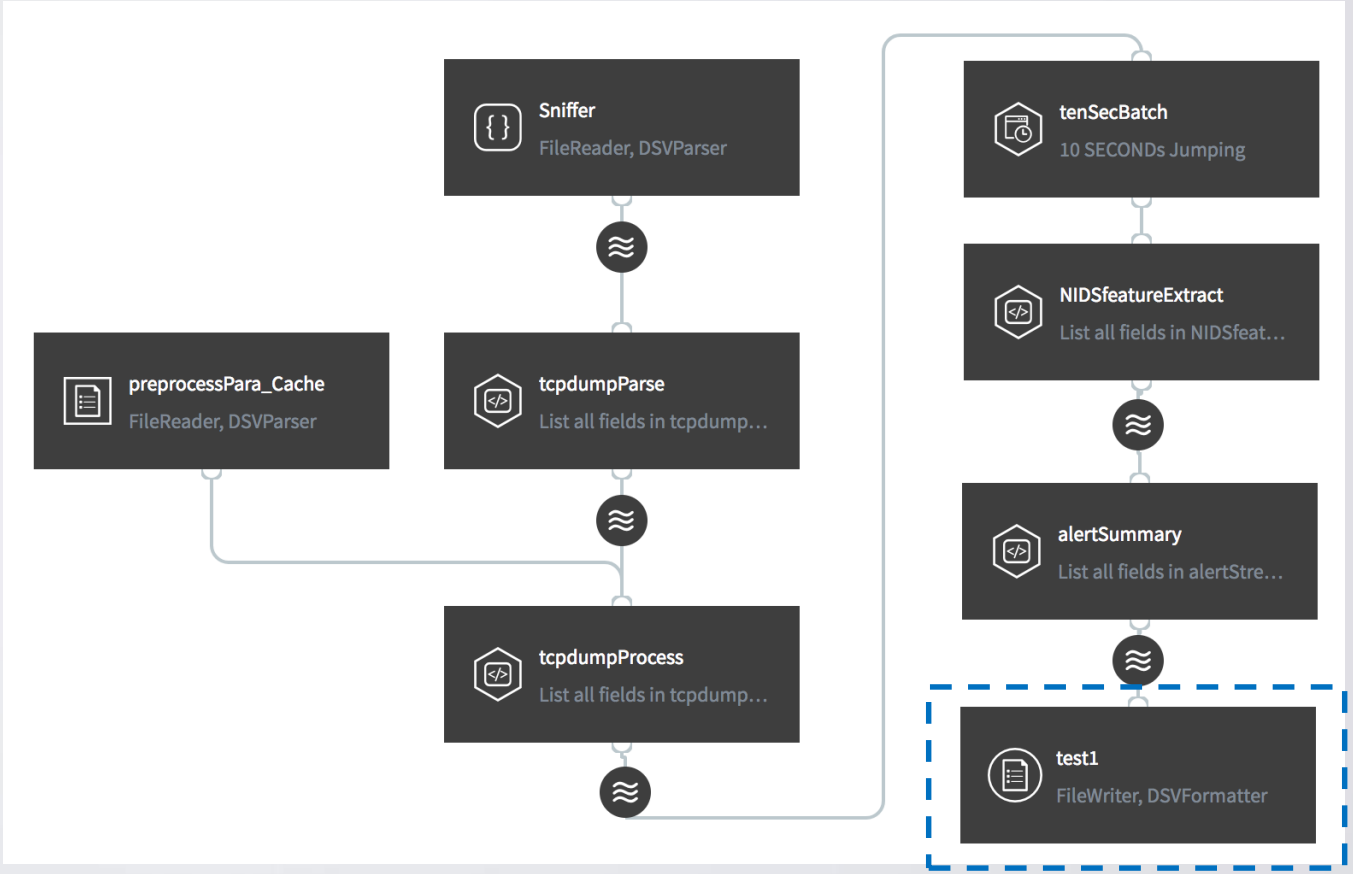
```
select datetime, name, total, AIDCSVMscore(f1, f2, f3, f4,
f5, f6, f7, f8, f9, f10, f11, f12, f13, f14, f15, f16, f17, f18,
f19, f20, f21, f22, f23, f24, f25, f26, f27, f28, f29, f30, f31,
f32, f33, f34, f35, f36, f37, f38, f39, f40, f41, f42,
'/home/admin/Striim/IntelAICon/appData/arffHeader.txt',
'/home/admin/Striim/IntelAICon/appData/testData.arff',
'/home/admin/Striim/IntelAICon/idsOneClass.model') as
anomalySum, anomalySum.size() as anomalyNum,
anomalyExtract(anomalySum,src) as anomalySrc,
anomalyExtract(anomalySum, dst) as anomalyDst from
NIDSfeatureStream;
```



NIDS Task 1: Anomaly Detection on Network Flows

- Persist results

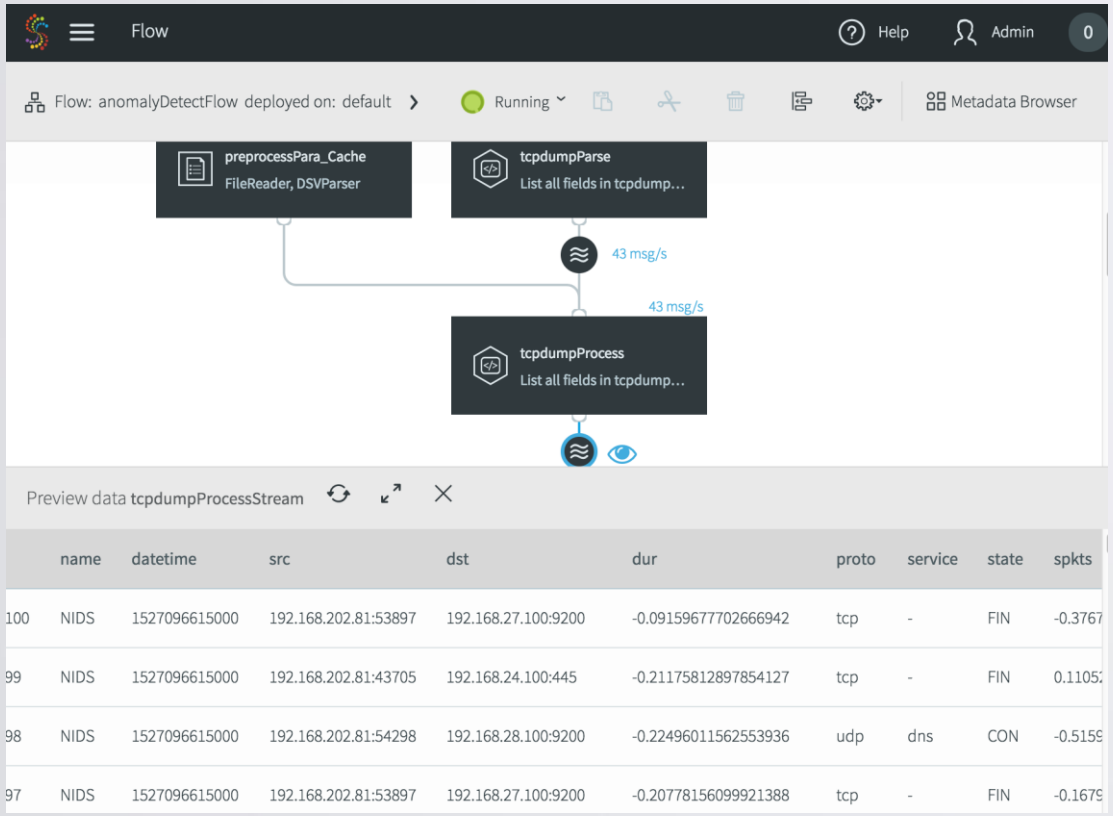




striim

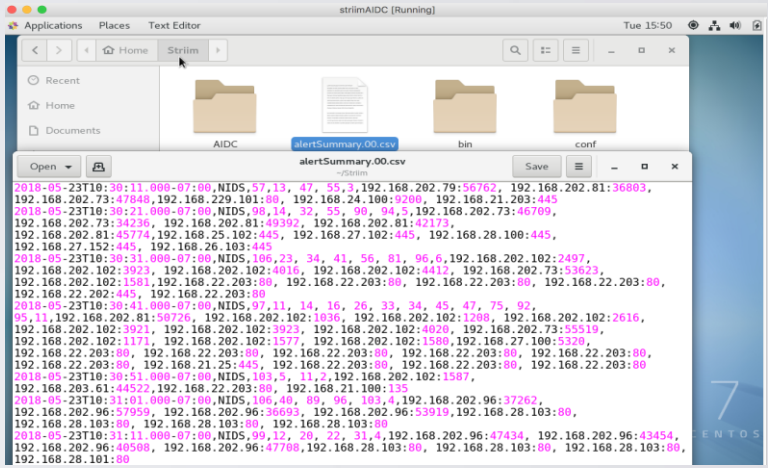
NIDS Task 1: Anomaly Detection on Network Flows

1. Run application to see streaming processing results at each step (Click one stream and then click *Preview on Run*).

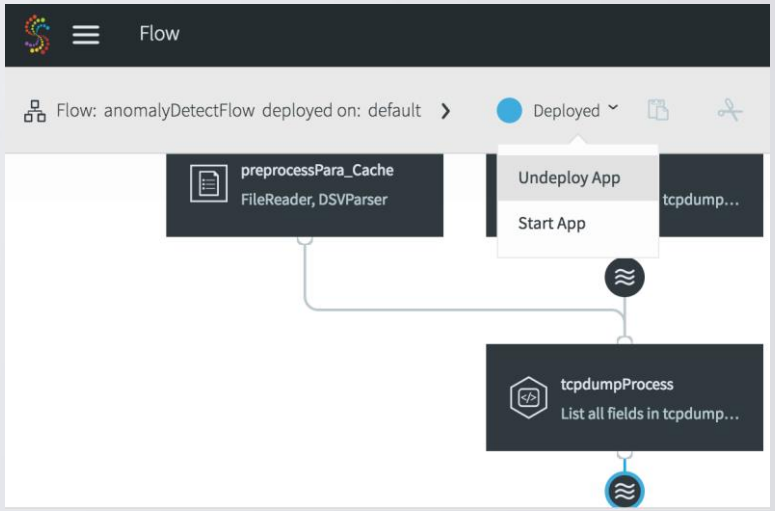


	name	datetime	src	dst	dur	proto	service	state	spkts
100	NIDS	1527096615000	192.168.202.81:53897	192.168.27.100:9200	-0.09159677702666942	tcp	-	FIN	-0.3767
99	NIDS	1527096615000	192.168.202.81:43705	192.168.24.100:445	-0.21175812897854127	tcp	-	FIN	0.1105
98	NIDS	1527096615000	192.168.202.81:54298	192.168.28.100:9200	-0.22496011562553936	udp	dns	CON	-0.5159
97	NIDS	1527096615000	192.168.202.81:53897	192.168.27.100:9200	-0.20778156099921388	tcp	-	FIN	-0.1679

2. Go to /home/admin/Striim to see the persisted results in alertSummary.csv

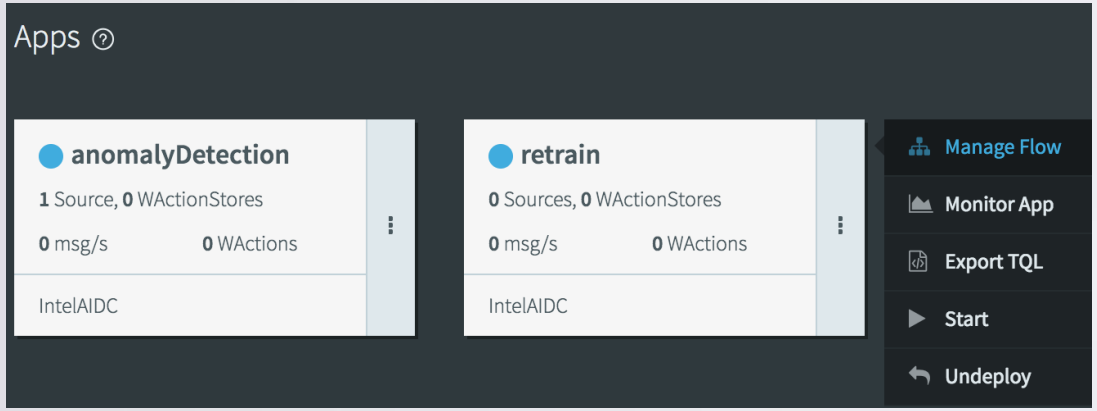


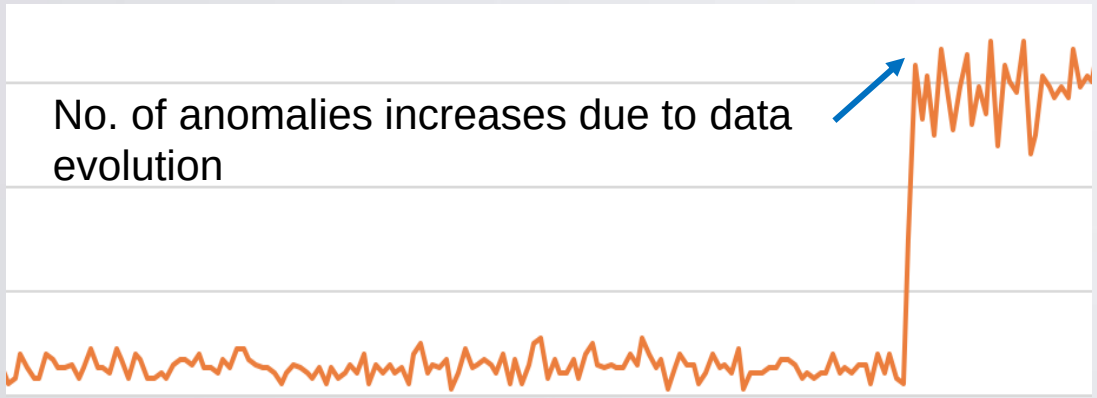
3. **Stop** and **undeploy** the application

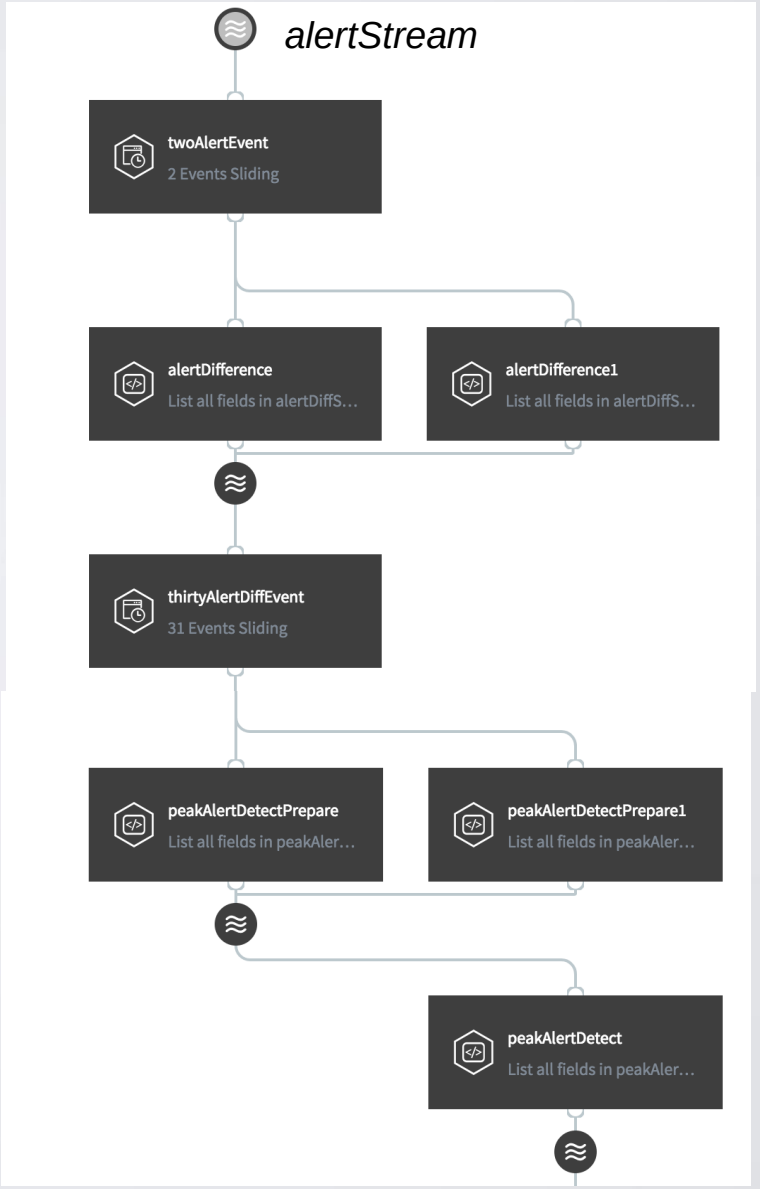


NIDS Task 2 (1): Detect Data Pattern Change

Deploy *anomalyDetection* and then deploy *retrain*
View Flow of *retrain*

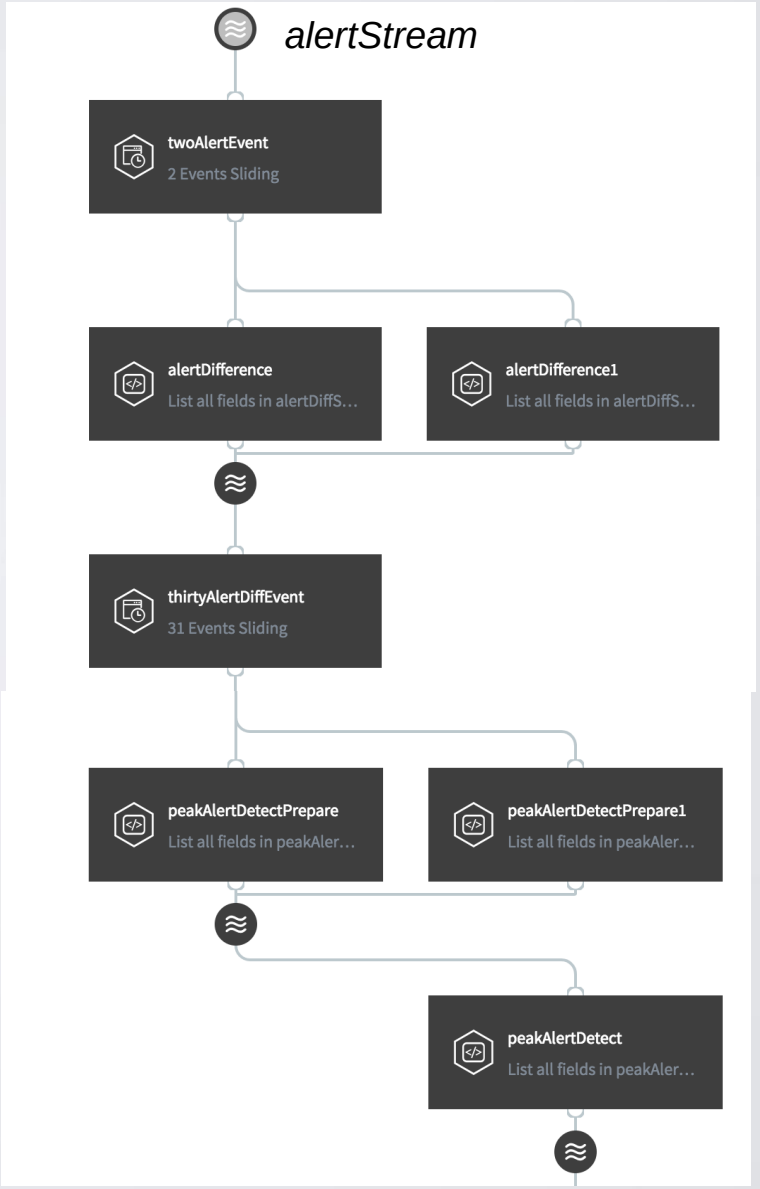






NIDS Task 2 (1): Detect Data Pattern Change

- Difference time series
- Aggregate recent time events and prepare algorithm input
- Detect time series peaks



NIDS Task 2 (1): Detect Data Pattern Change

- Difference time series

twoAlertEvent

Mode ⓘ Sliding

alertDifference

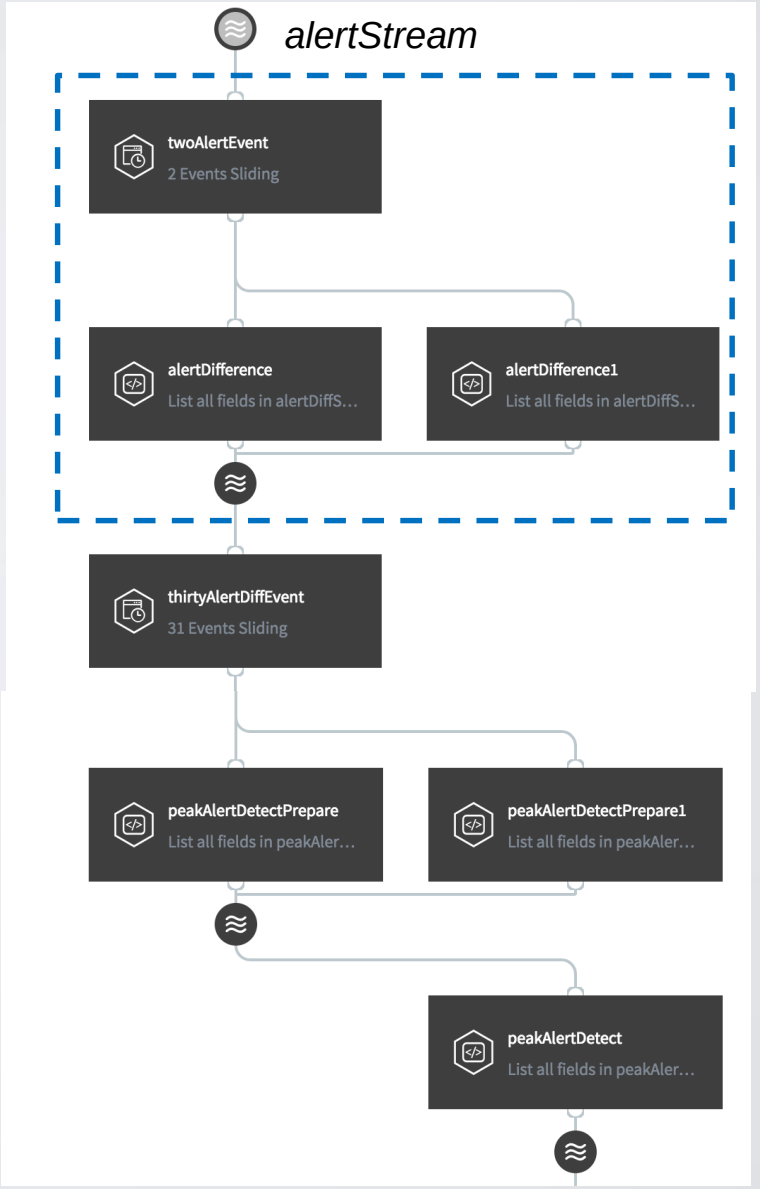
QUERY

```
SELECT last(datetime) as datetime, name, to_double(last(total)-
first(total)) as totalDiff, to_double(last(anomalyNum)-
first(anomalyNum)) as anomalyDiff FROM twoAlertEvent having
count(*) = 2
```

alertDifference1

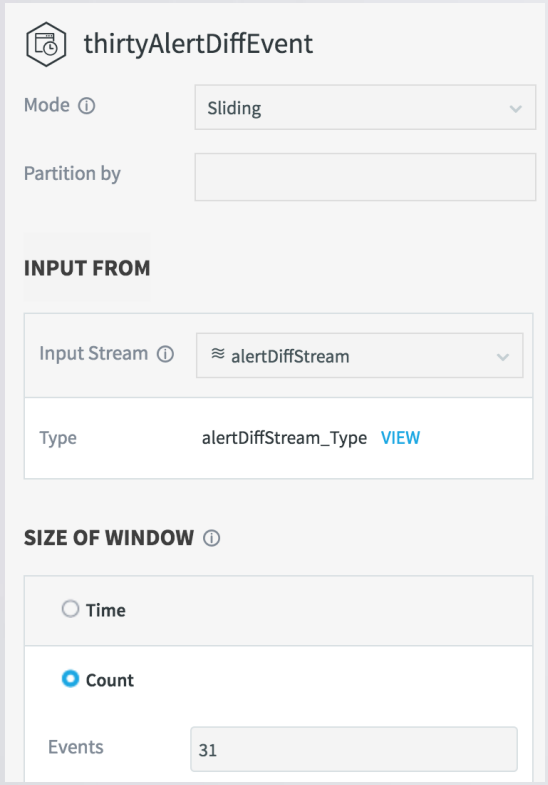
QUERY

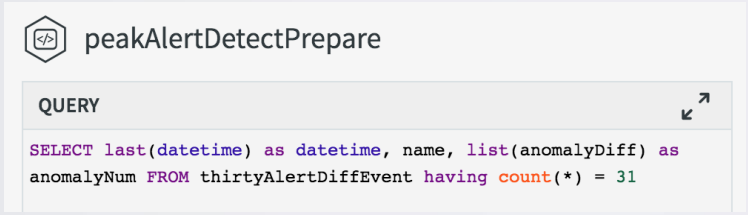
```
SELECT last(datetime) as datetime, name, 0.0 as totalDiff, 0.0 as
anomalyDiff FROM twoAlertEvent having count(*) < 2
```

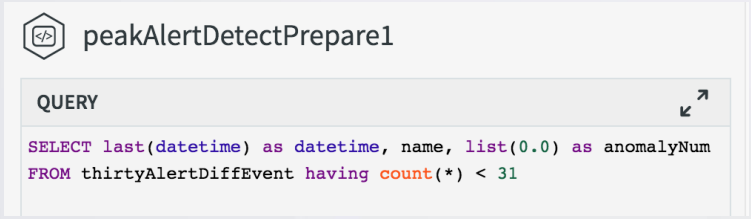


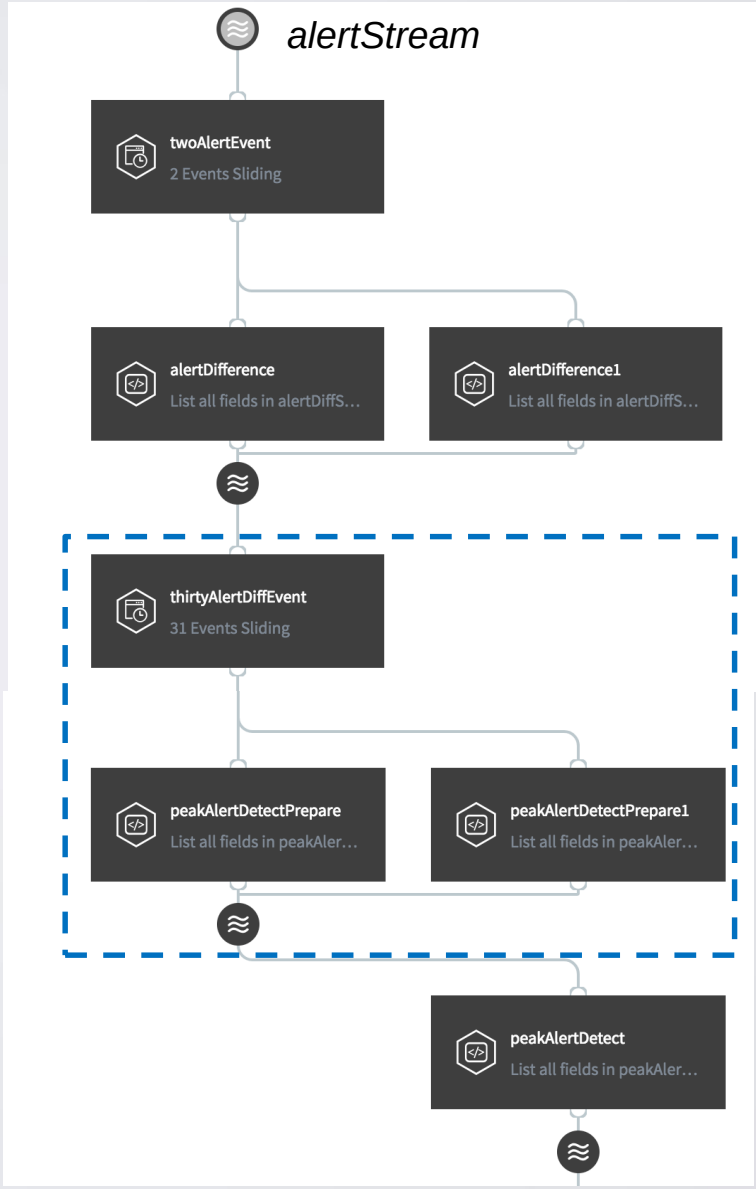
NIDS Task 2 (1): Detect Data Pattern Change

- Aggregate recent time events
  - Constrain peak detection on









striim

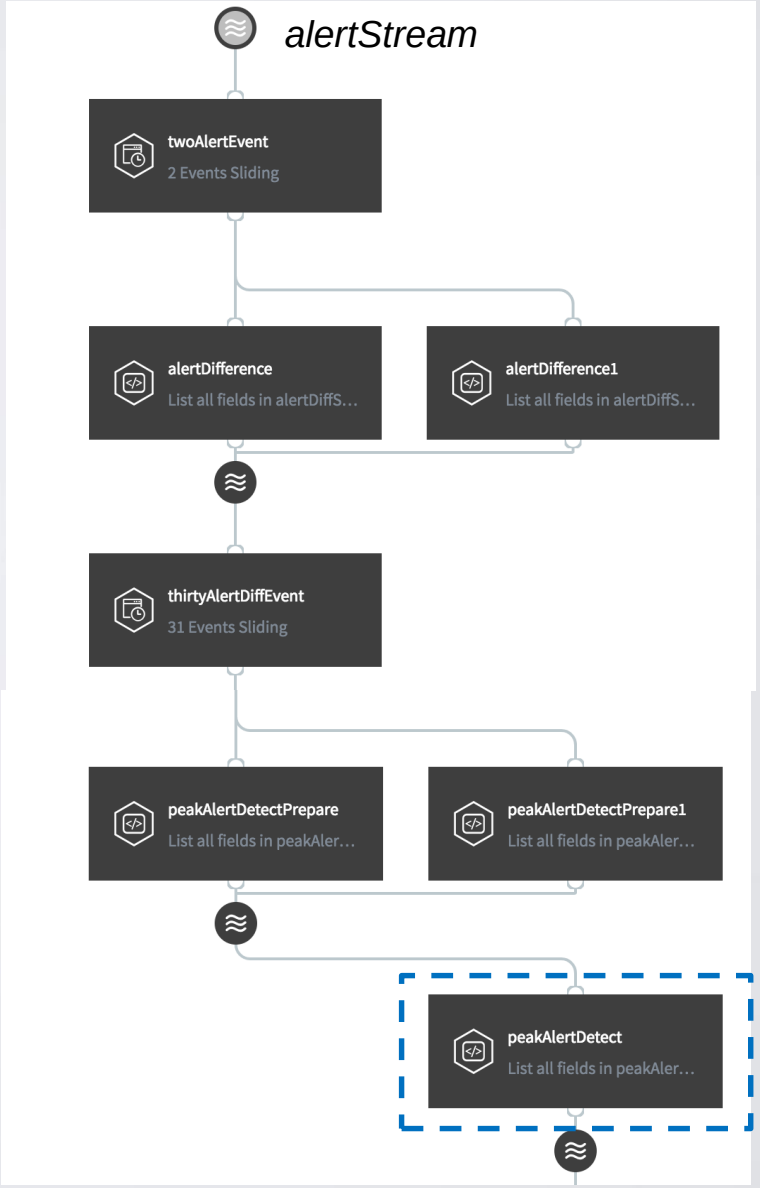
NIDS Task 2 (1): Detect Data Pattern Change

- Detect time series peaks
  - Calculate z-score of the 31th event in the sliding window

peakAlertDetect

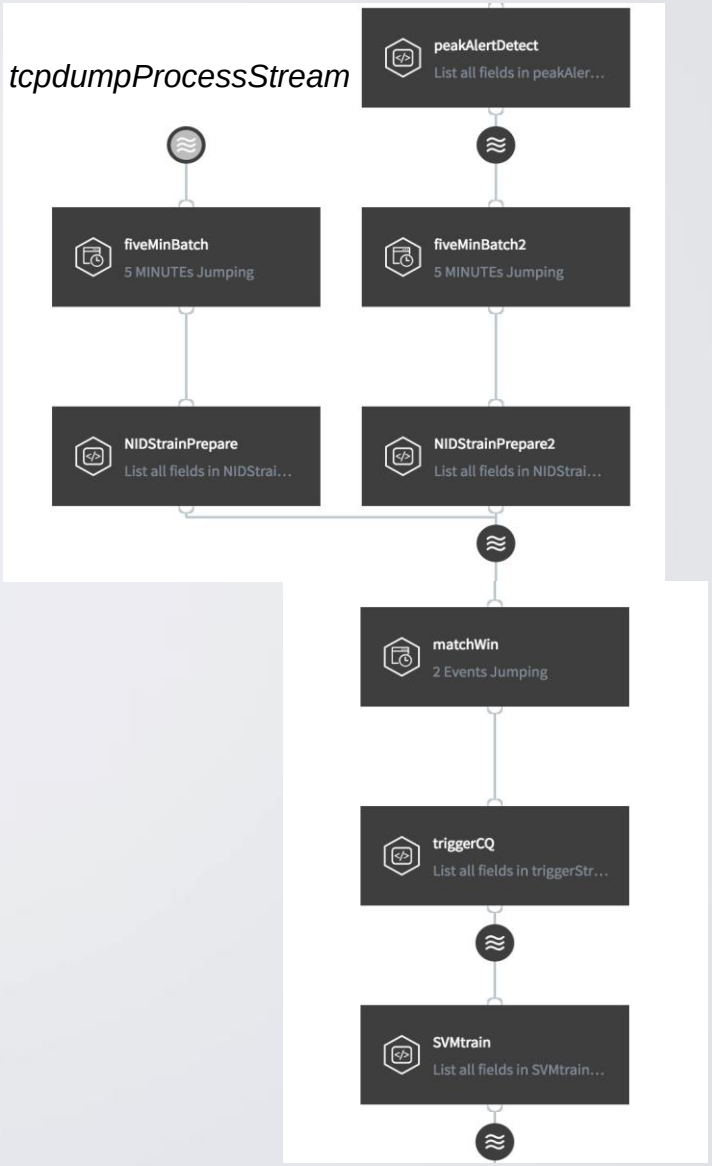
QUERY

```
SELECT datetime as datetime, name as name, peakDetect(anomalyNum,
5) as anomalyPeak
FROM peakAlertDetectStream
```



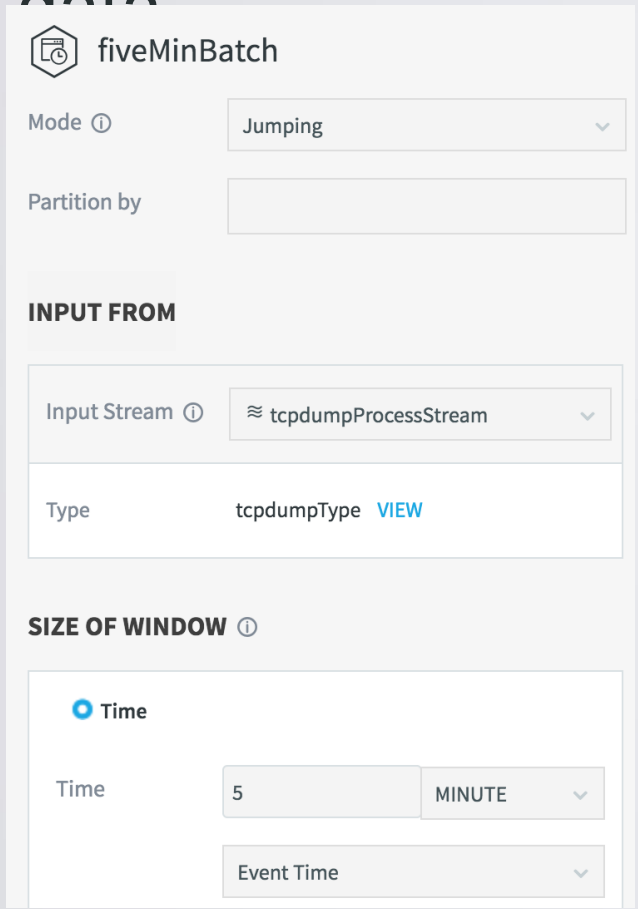
NIDS Task 2 (2): Retrain Model

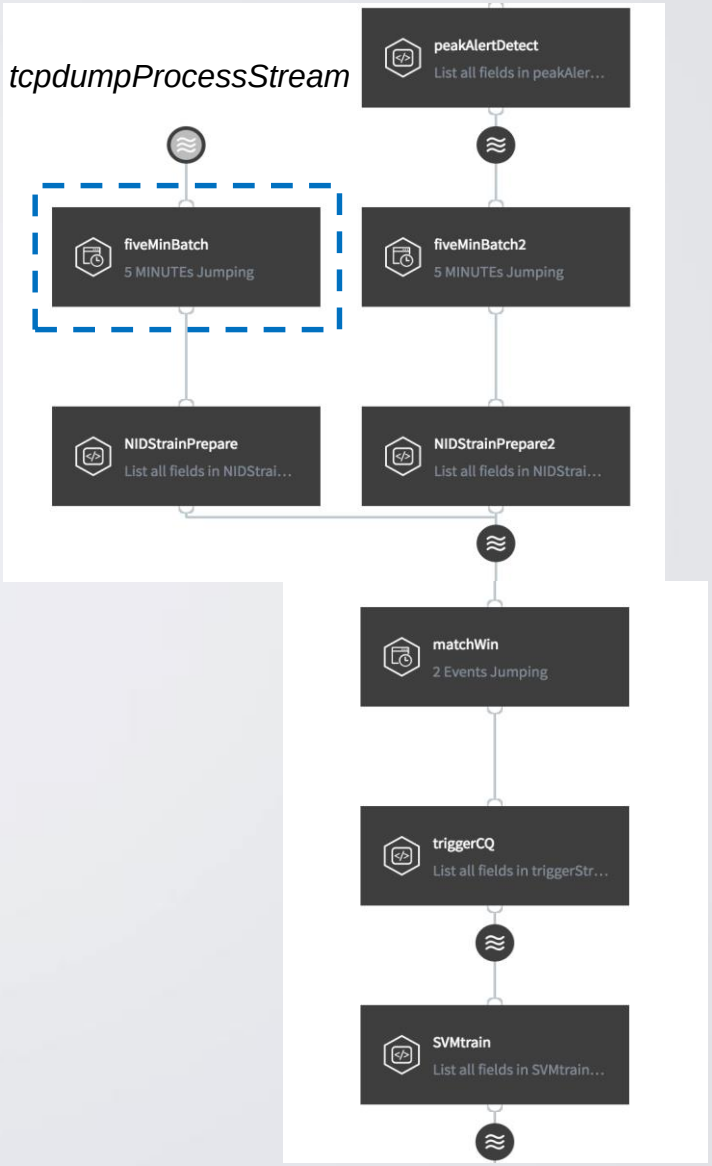
- Aggregate new training data
- Aggregate peak signals
- Join two parallel flows
- Cancel/trigger retraining



NIDS Task 2 (2): Retrain Model

- Aggregate new training data

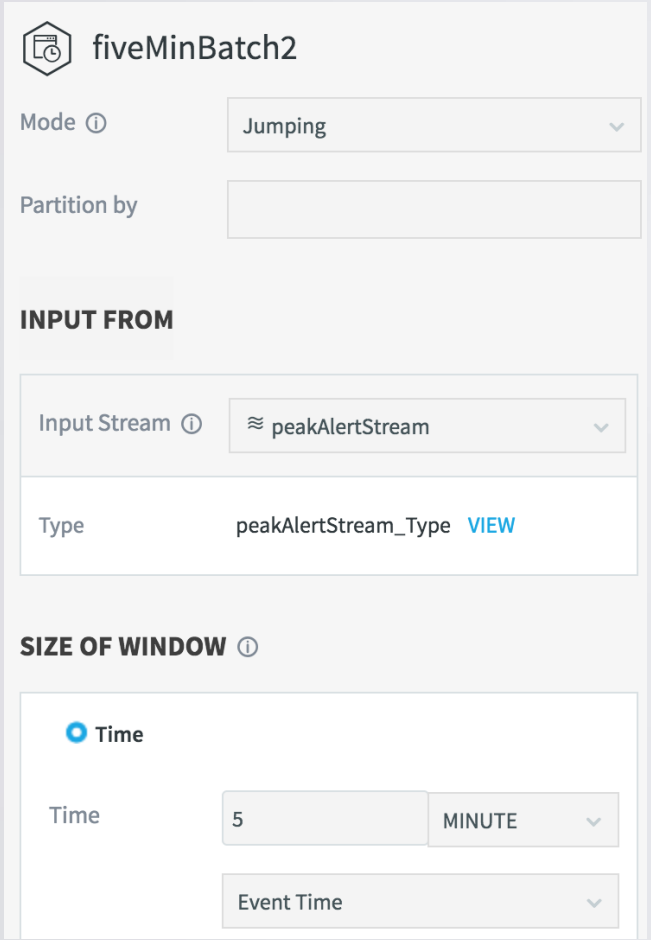


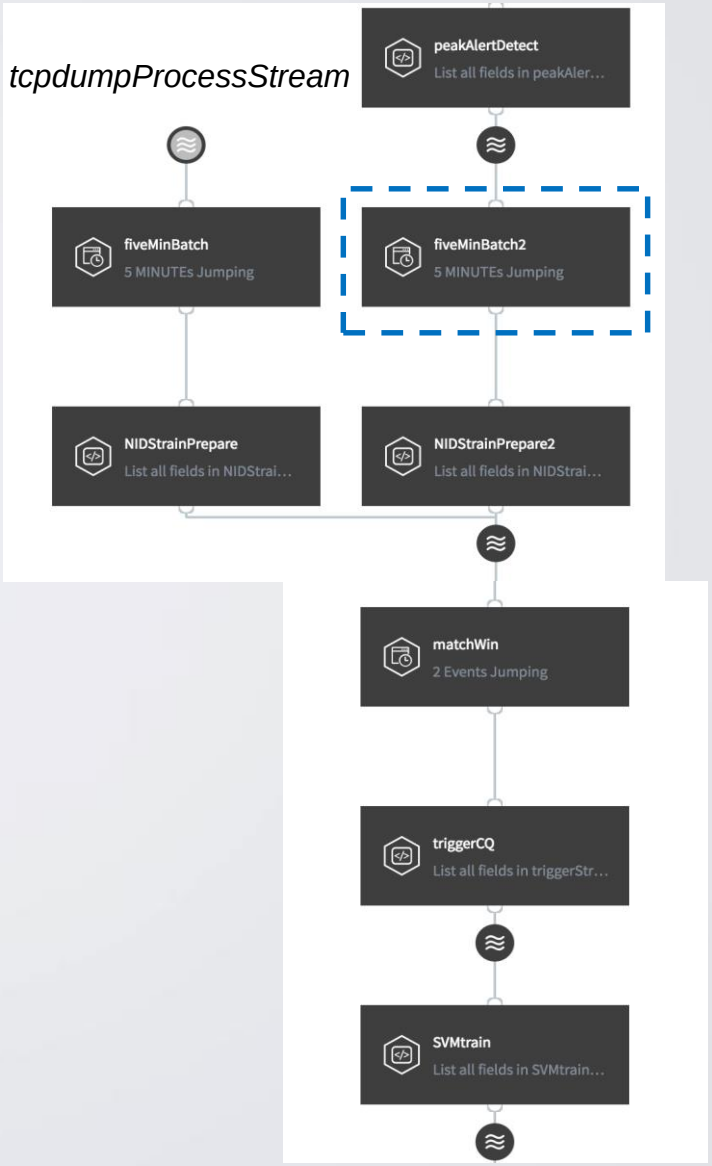


striim

NIDS Task 2 (2): Retrain Model

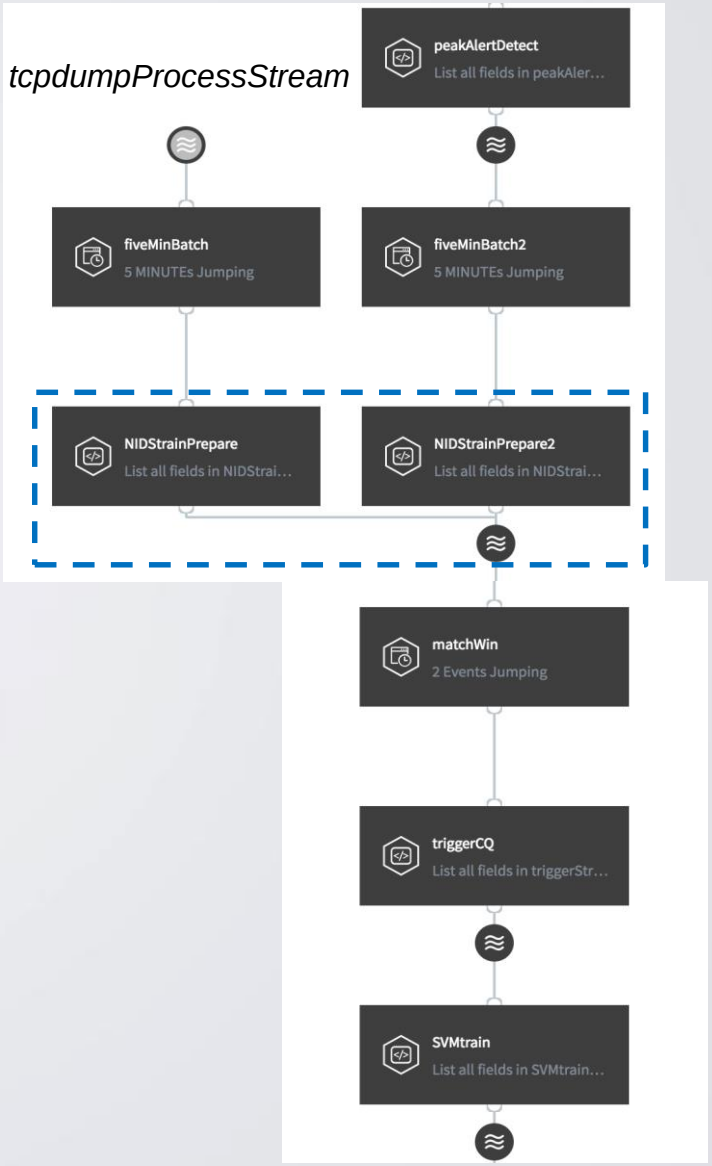
- Aggregate peak signals





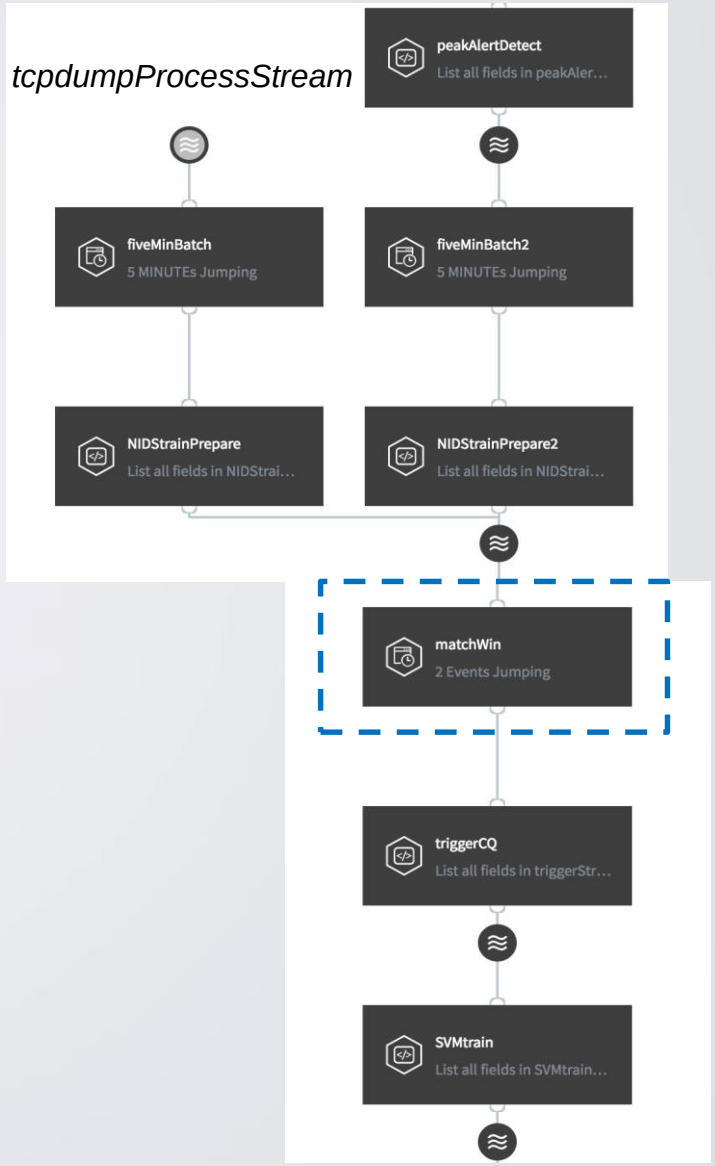
NIDS Task 2 (2): Retrain Model

- Join two parallel flows
  - Features from *NIDStrainPrepare*
  - Data change signal from *NIDStrainPrepare1*
  - Insert into the same stream



NIDS Task 2 (2): Retrain Model

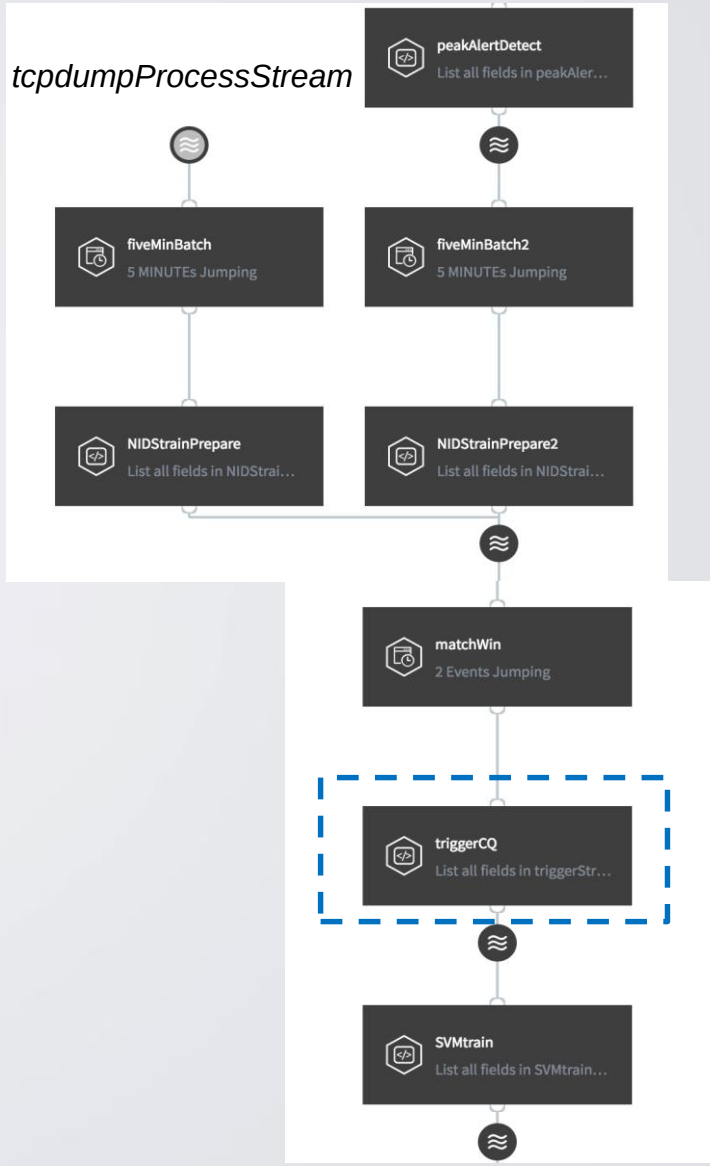
- Cancel/trigger retraining
  - *CREATE JUMPING WINDOW matchWin OVER NIDStrainDataStream KEEP 2 rows WITHIN 30 second;*



NIDS Task 2 (2): Retrain Model

- Cancel/trigger retraining
  - Select fields from upstreams

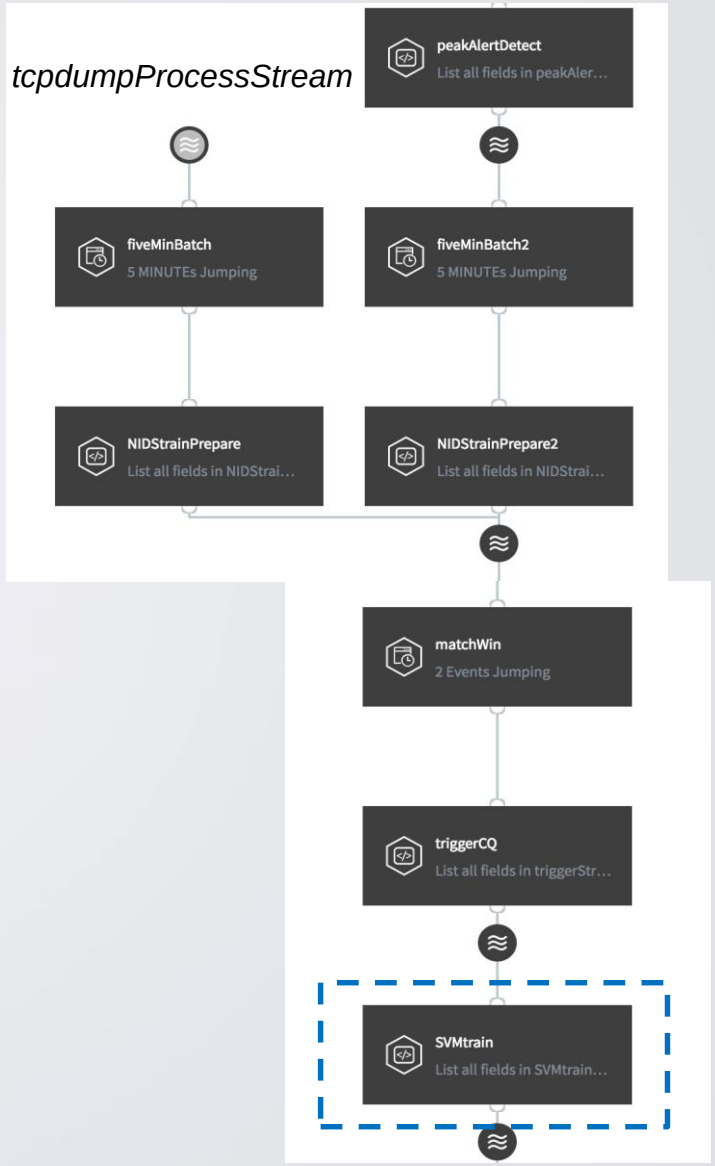
```
SELECT first(name) as name, first(datetime) as ts1, last(datetime) as ts2,
first(total) as total1, last(total) as total2, first(triggerFlag) as flag1,
last(triggerFlag) as flag2, first(f1) as f1, first(f2) as f2, first(f3) as f3, first(f4)
as f4, first(f5) as f5, first(f6) as f6, first(f7) as f7, first(f8) as f8, first(f9) as f9,
first(f10) as f10, first(f11) as f11, first(f12) as f12, first(f13) as f13, first(f14)
as f14, first(f15) as f15, first(f16) as f16, first(f17) as f17, first(f18) as f18,
first(f19) as f19, first(f20) as f20, first(f21) as f21, first(f22) as f22, first(f23)
as f23, first(f24) as f24, first(f25) as f25, first(f26) as f26, first(f27) as f27,
first(f28) as f28, first(f29) as f29, first(f30) as f30, first(f31) as f31, first(f32)
as f32, first(f33) as f33, first(f34) as f34, first(f35) as f35, first(f36) as f36,
first(f37) as f37, first(f38) as f38, first(f39) as f39, first(f40) as f40, first(f41)
as f41, first(f42) as f42 from matchWin having count(*) = 2
```



NIDS Task 2 (2): Retrain Model

- Cancel/trigger retraining
  - If there is data change signal, trigger retraining, otherwise cancel it.

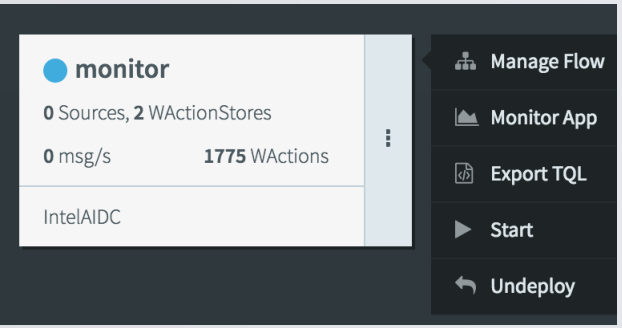
```
SELECT ts1, ts2, flag2 AS signal,
CASE WHEN signal > 0 THEN
AIDCSVMtrain(f1, f2, f3, f4, f5, f6, f7, f8, f9, f10, f11, f12, f13, f14, f15, f16,
f17, f18, f19, f20, f21, f22, f23, f24, f25, f26, f27, f28, f29, f30, f31, f32, f33,
f34, f35, f36, f37, f38, f39, f40, f41, f42,
'/home/admin/Striim/IntelAICon/appData/arffHeader.txt',
'/home/admin/Striim/IntelAICon/appData/trainData.arff', '-S 2 -K 2 -D 3 -G
0.1 -R 0.0 -N 0.1 -M 40.0 -C 1.0 -E 0.001 -P 0.1',
'/home/admin/Striim/IntelAICon/idsOneClass.model')
ELSE 1
END as status,
CASE WHEN signal > 0 THEN
'retrain'
ELSE 'model serving'
END as retrainMsg
FROM triggerStream;
```

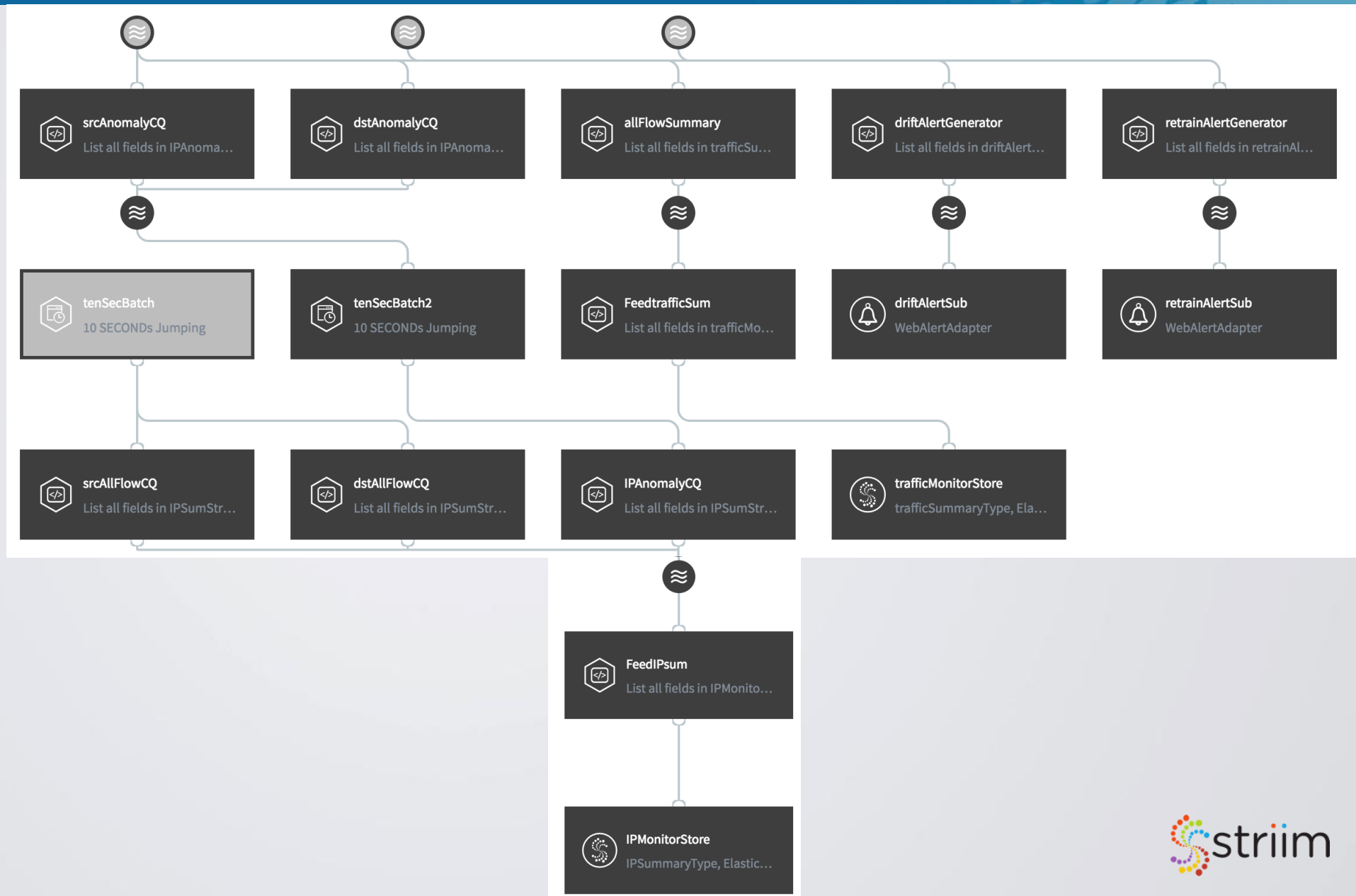


NIDS Task 3 (1): Create WactionStores and Alerts

Deploy *monitor* application (*anomalyDetection* and *retrain* have been deployed in Task 2)

View Flow of *monitor*

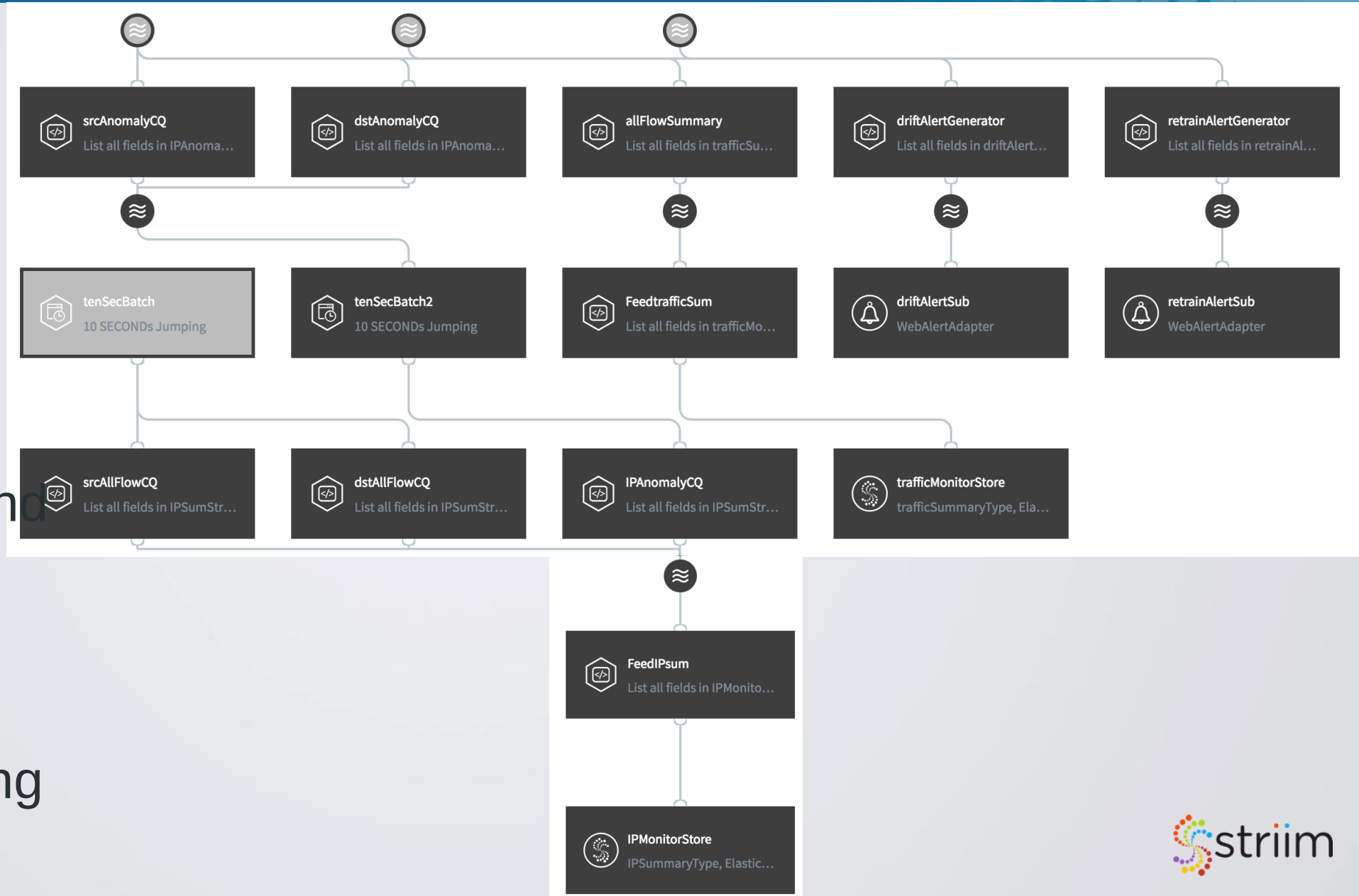




striim

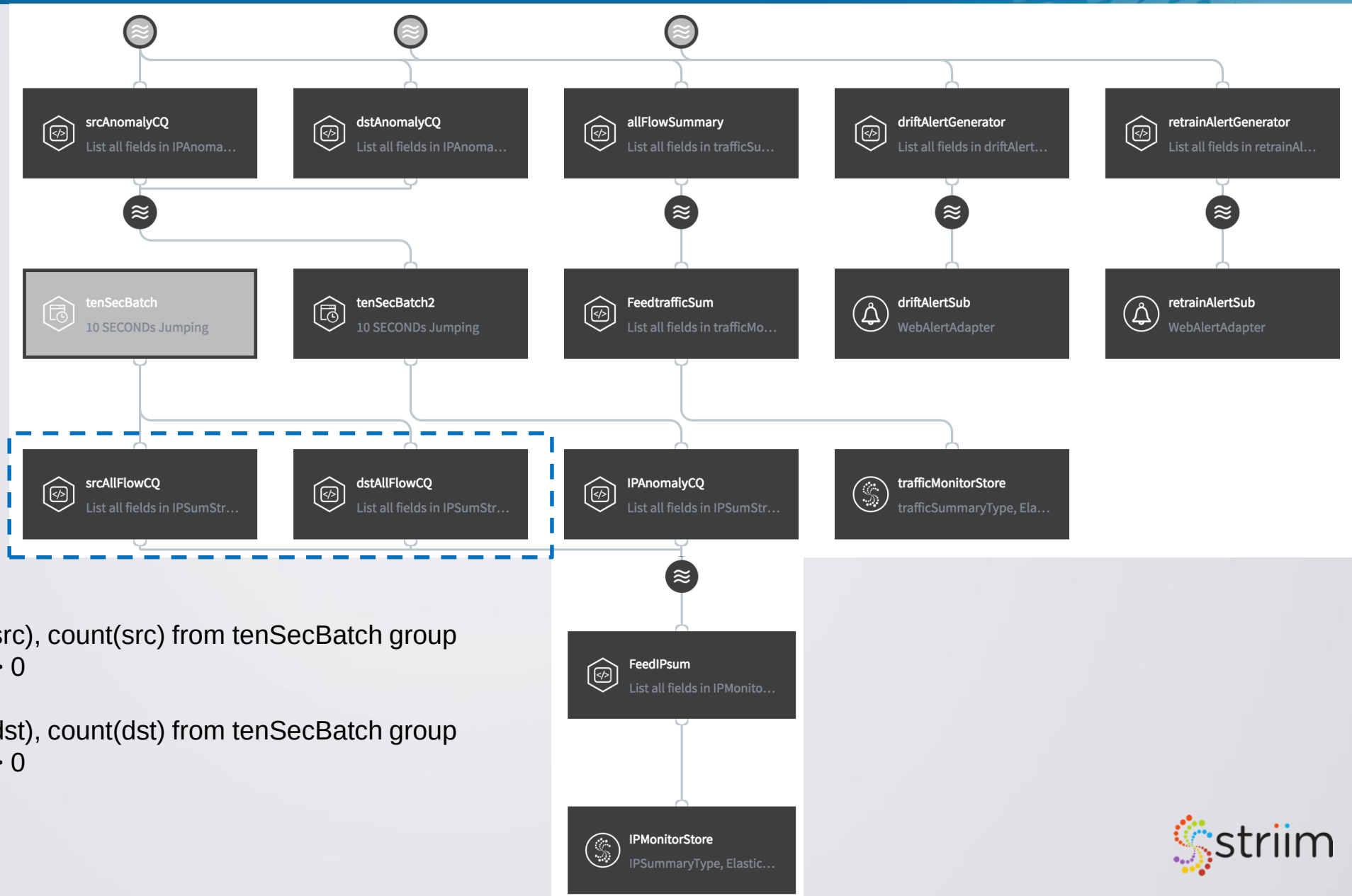
NIDS Task 3 (1): Create WactionStores and Alerts

- Monitor traffic (normal and abnormal) sources and destinations
- Monitor No. of network flows and anomalies
- Alert on data pattern change
- Alert on retraining



NIDS Task 3 (1): Create WactionStores and Alerts

- Monitor traffic sources and destinations

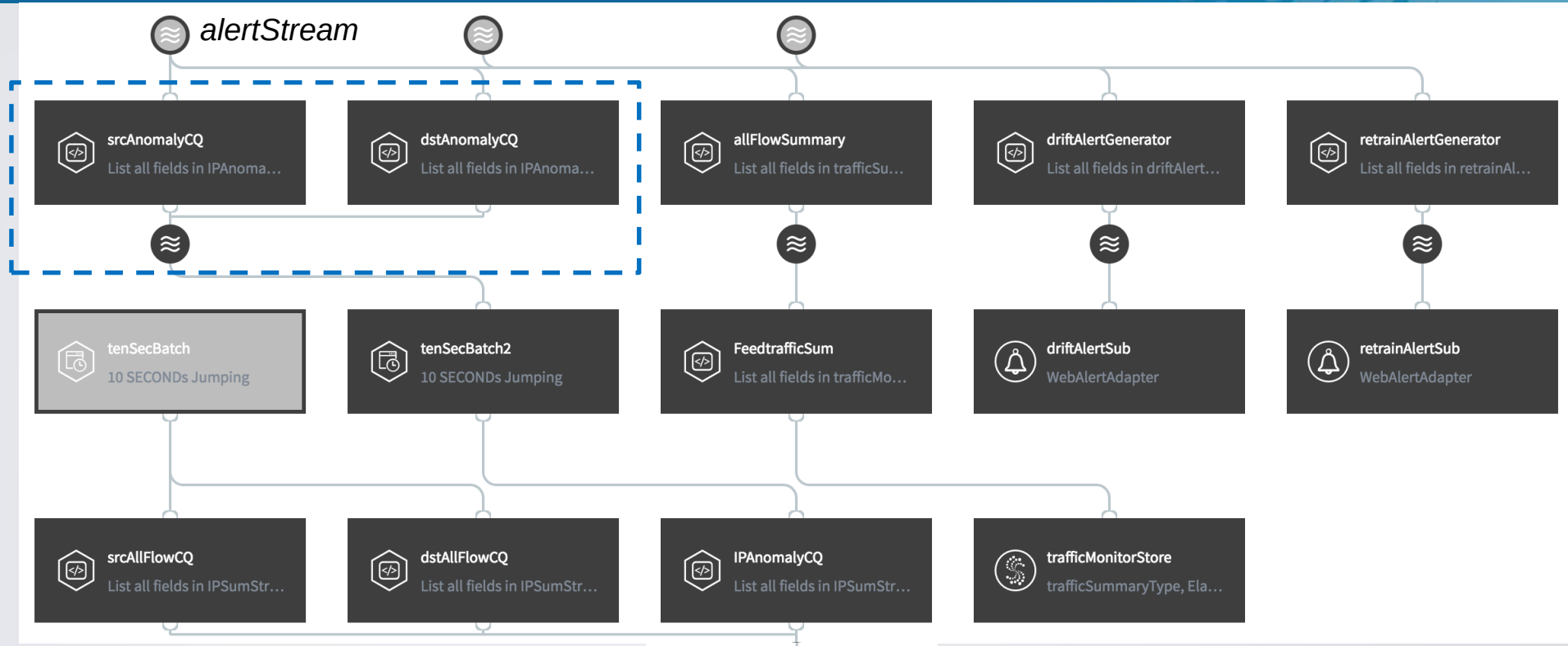


select datetime, 'srcIP', 'all', first(src), count(src) from tenSecBatch group by src, datetime having count(*) > 0

select datetime, 'dstIP', 'all', first(dst), count(dst) from tenSecBatch group by dst, datetime having count(*) > 0

NIDS Task 3 (1): Create WactionStores and Alerts

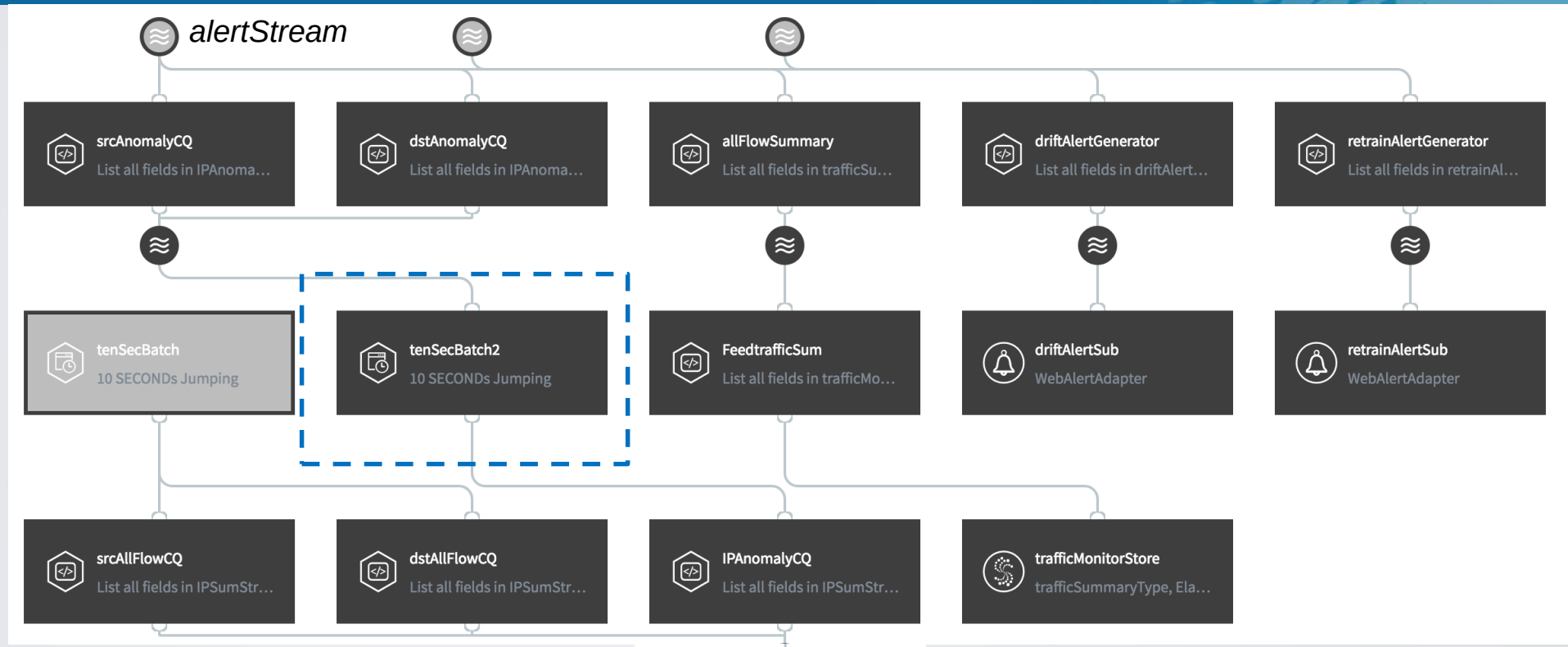
- Monitor abnormal traffic sources and destinations

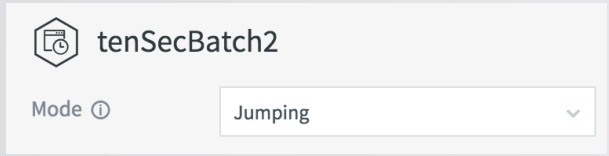


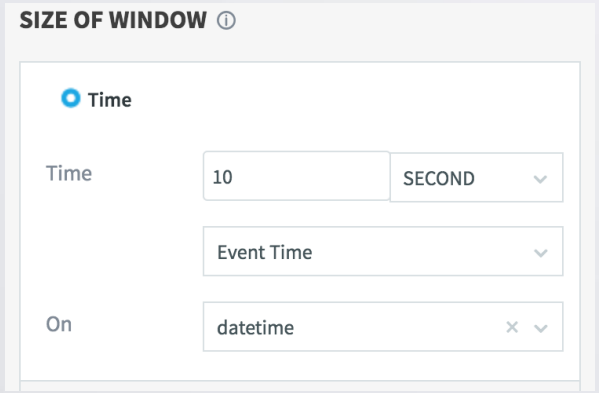

```
select datetime, 'srcIP' as name, 'anomaly' as flowType, to_string(lst) as IP
from alertStream s, iterator(s.anomalySrc) lst;

select datetime, 'dstIP' as name, 'anomaly' as flowType, to_string(lst) as IP
from alertStream s, iterator(s.anomalyDst) lst;
```

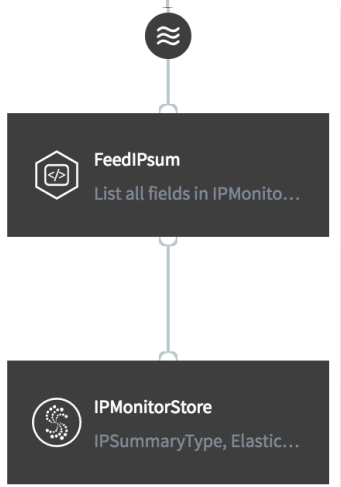
NIDS Task 3 (1): Create WactionStores and Alerts

- Monitor abnormal traffic sources and destinations





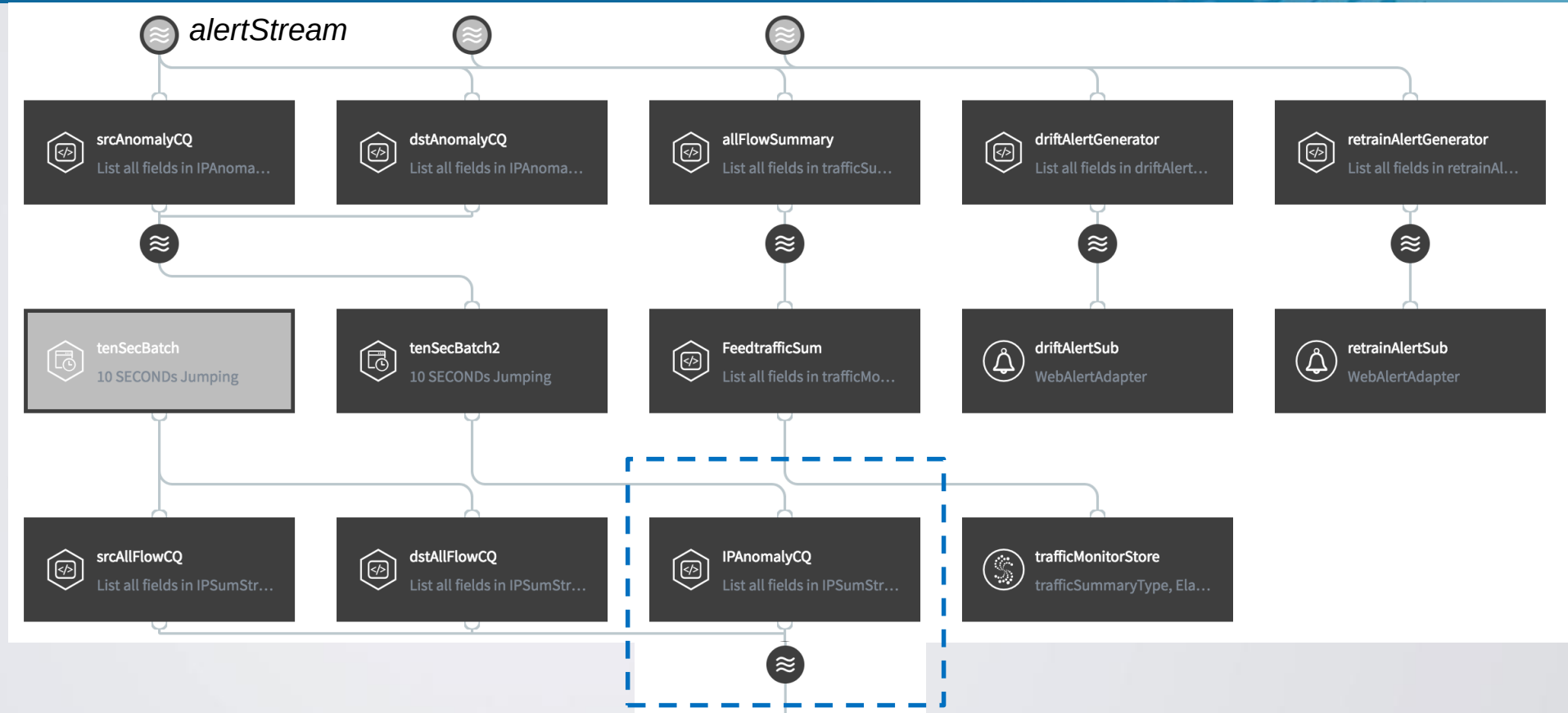




striim

NIDS Task 3 (1): Create WactionStores and Alerts

- Monitor abnormal traffic sources and destinations

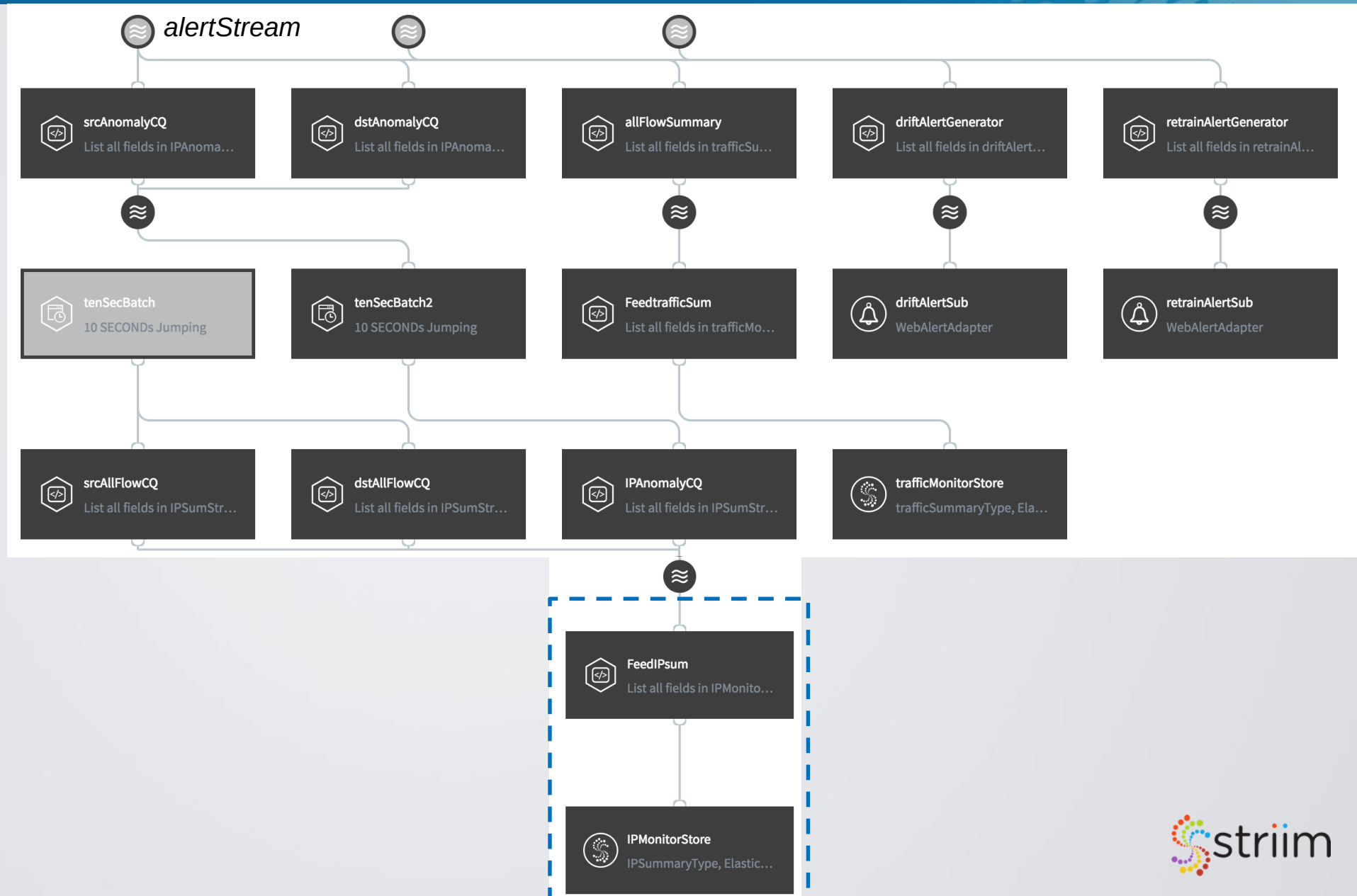


select datetime, name, flowType, first(IP), count(IP) from tenSecBatch2 group by IP, datetime, name having count(*) > 0;

striim

NIDS Task 3 (1): Create WactionStores and Alerts

- Feed monitor stream into WactionStore



FeedIPsum

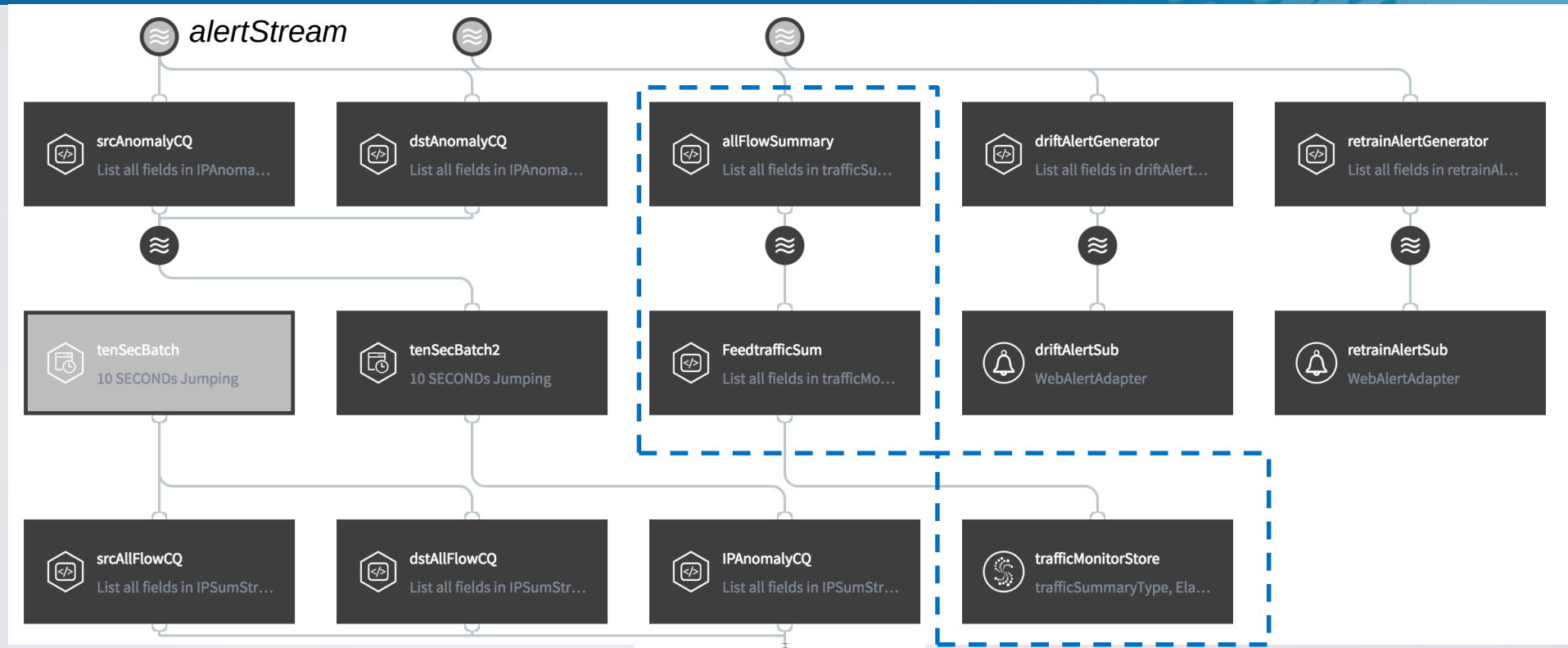
QUERY

```
select * from IPSumStream
```


striim

NIDS Task 3 (1): Create WactionStores and Alerts

- Monitor No. of network flows and anomalies
- Feed stream into WactionStore



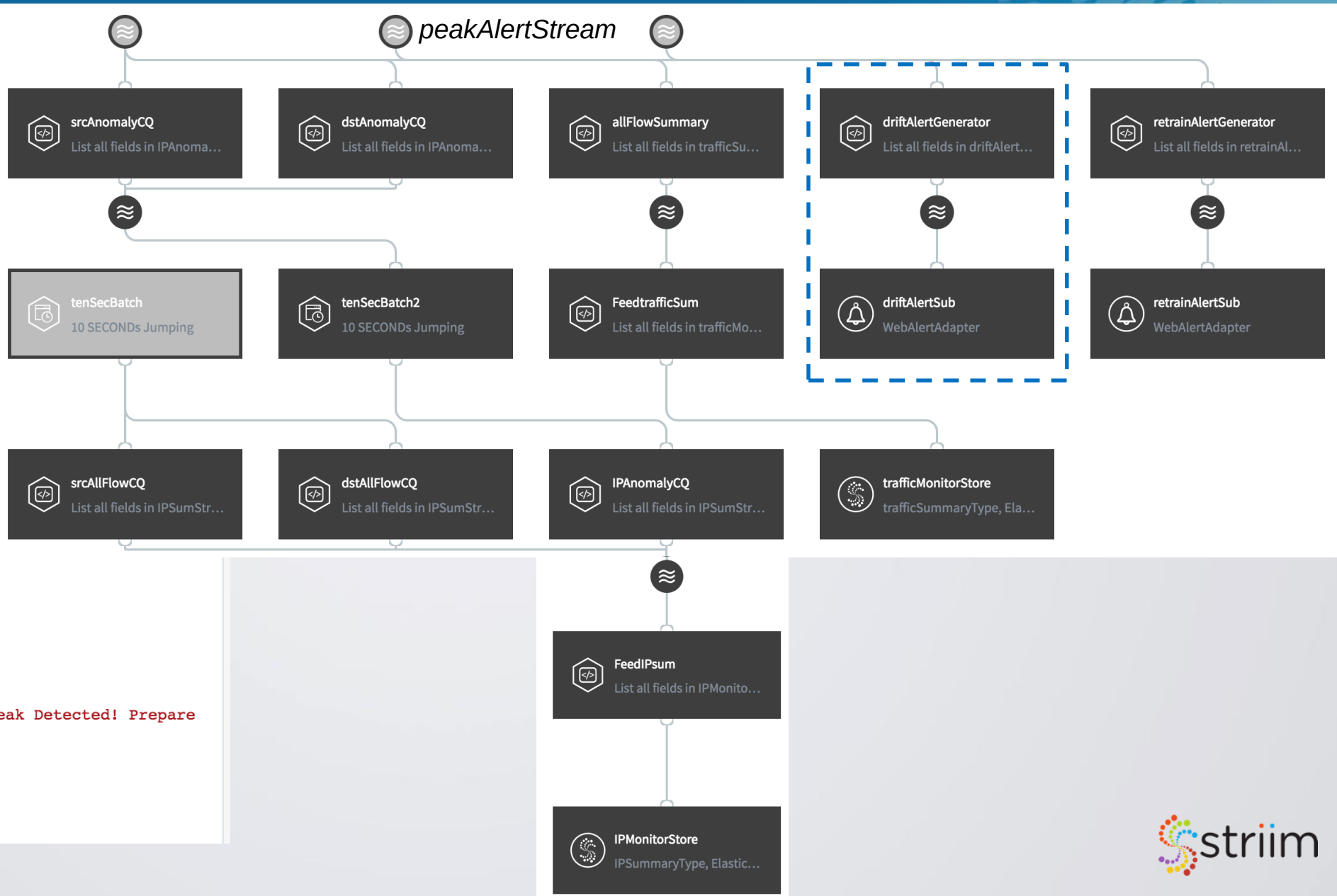
allFlowSummary

QUERY

```
select datetime, total, anomalyNum from alertStream
```

NIDS Task 3 (1): Create WactionStores and Alerts

- Alert on data pattern change



driftAlertGenerator

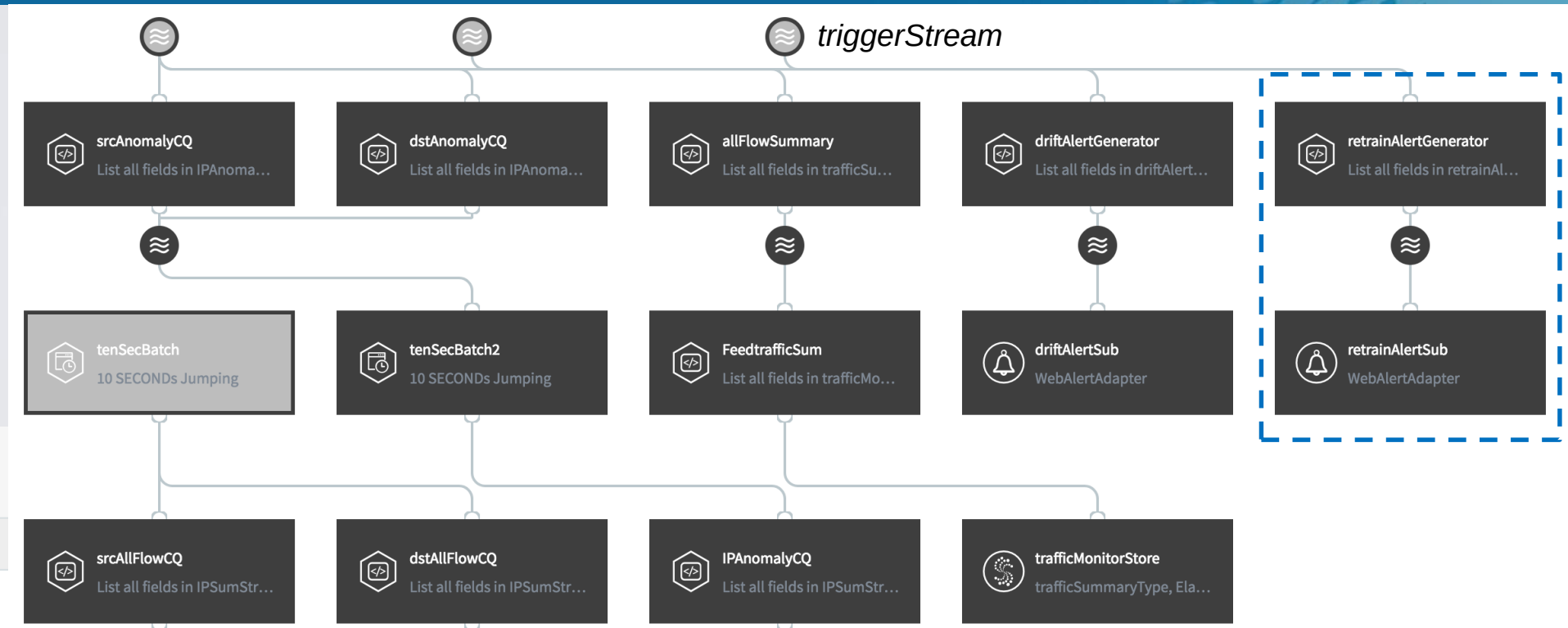
QUERY

```
SELECT 'Data Drift', s.name,
  CASE
    WHEN s.anomalyPeak < 1 THEN 'info'
    ELSE 'warning' END,
  CASE
    WHEN s.anomalyPeak < 1 THEN 'cancel'
    ELSE 'raise' END,
  CASE
    WHEN s.anomalyPeak > 0 THEN 'Anomaly Peak Detected! Prepare
to retrain the model!'
    ELSE ''
    END
FROM peakAlertStream s
```

striim

NIDS Task 3 (1): Create WactionStores and Alerts

- Alert on retraining



retrainAlertGenerator

QUERY

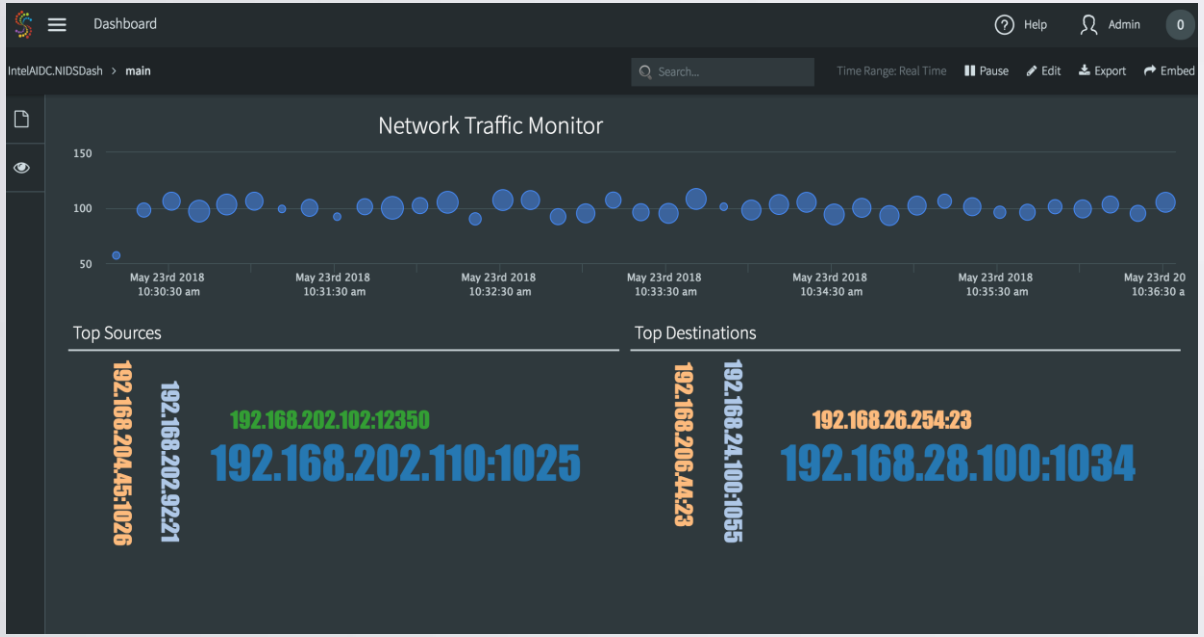
```
SELECT 'Retrain', s.name,
  CASE
    WHEN s.flag1+s.flag2 < 1 THEN 'info'
    ELSE 'warning' END,
  CASE
    WHEN s.flag1+s.flag2 < 1 THEN 'cancel'
    ELSE 'raise' END,
  CASE
    WHEN s.flag1+s.flag2 > 0 THEN 'Model retraining!'
    ELSE ''
    END
FROM triggerStream s
```

striim

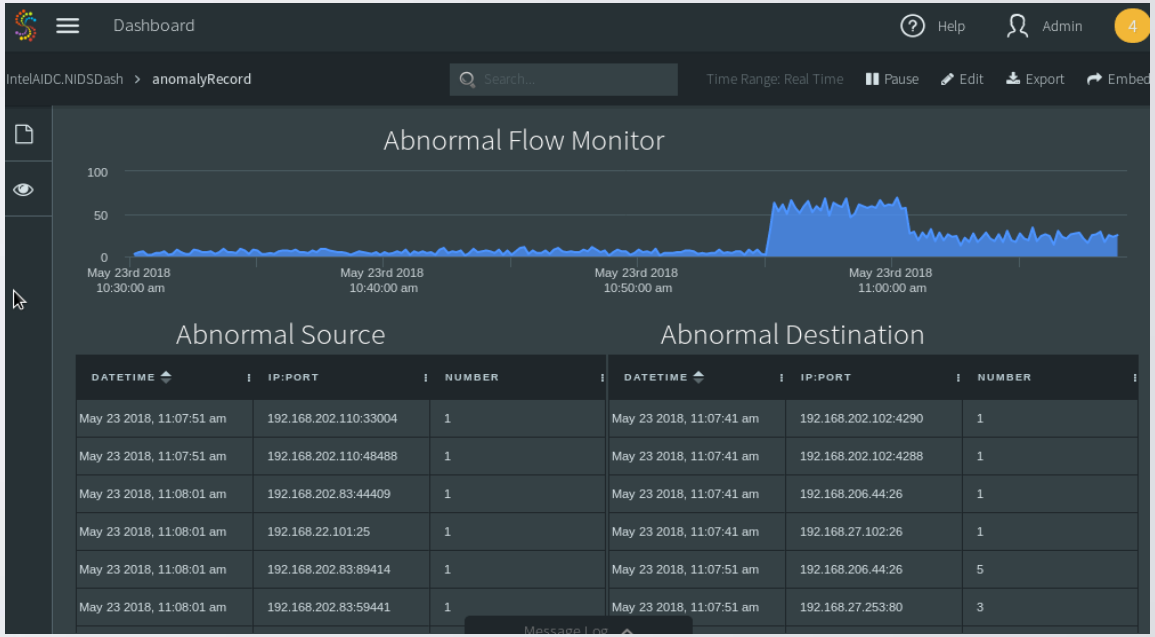
NIDS Task 3 (2): Build Dashboard

1. Start applications *monitor*, then *retrain*, then *anomalyDetection*

2. Navigate to Dashboard and view NIDSDash to see real-time visualization

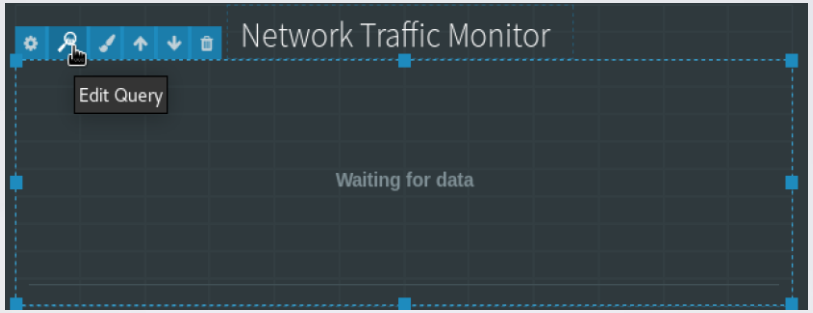


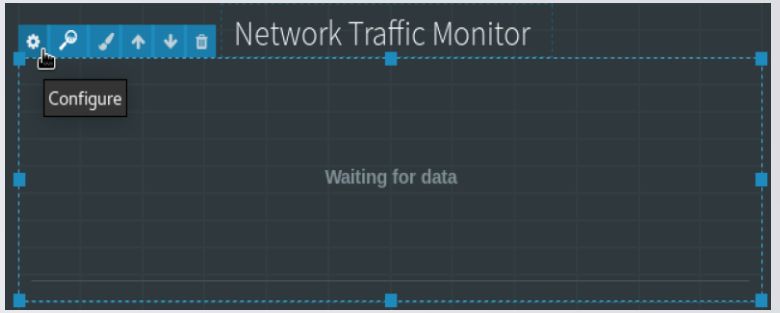
3. Click “Network Traffic Monitor” on the main page to drill down and to see anomaly details



Learn More About Striim Product

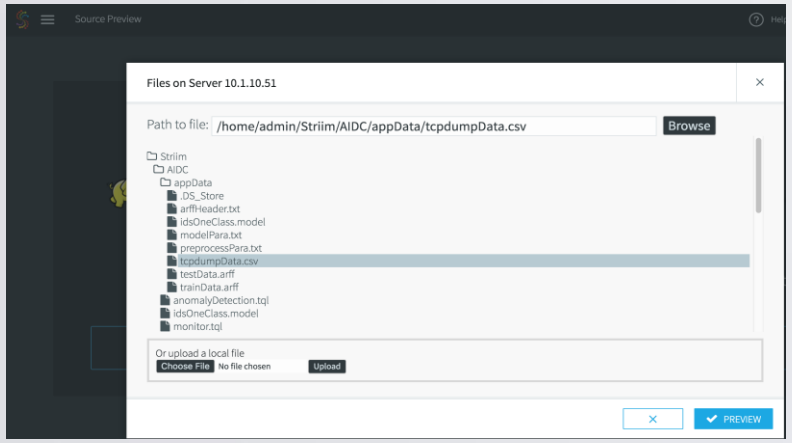
Click *Edit* on the top right corner of dashboard to see how to configure a visualization chart.



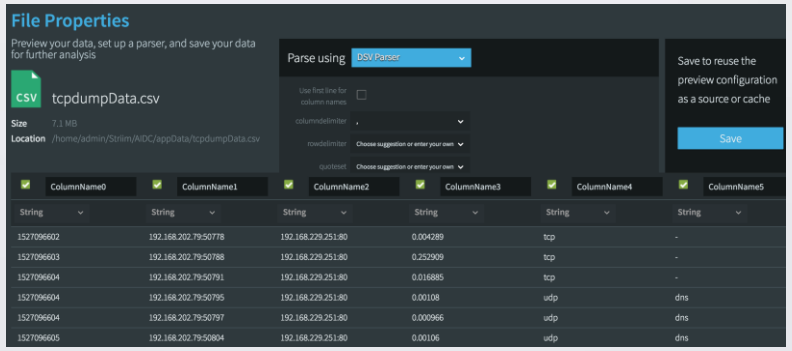


Open a new tab

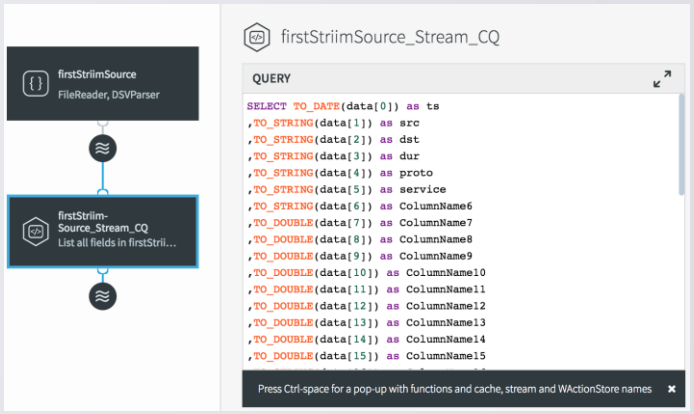
1. Go to Source Preview -> Browse -> choose tcpdumpData.csv to preview



2. Set columndelimiter as “,”, configure column name and type, save as source



3. Set application name as “firstStriimApp”, and source name as “firstStriimSource”



4. Edit a continuous query with SQL-like language, deploy your application and run to see output.

striim

Key Takeaways

- Streaming integration paves the way to ML operationalization
- Striim provides reliable and efficient streaming integration solution
- Support fast-track ML operationalization
- Striim filters, enriches and prepares streaming data
- Striim lands data continuously for model training
- Striim supports continuous model serving on data streams
- Striim handles model lifecycles with high automation
- Striim visualizes the real-time data and predictions, and alerts on issues

striim