Don’t Miss the Bus – Splunking Kafka at Scale

Donald Tregonning – Senior Software Engineer
Ken Chen – Principal Software Engineer
Scott Haskell – Principal SE Architect

October 2018

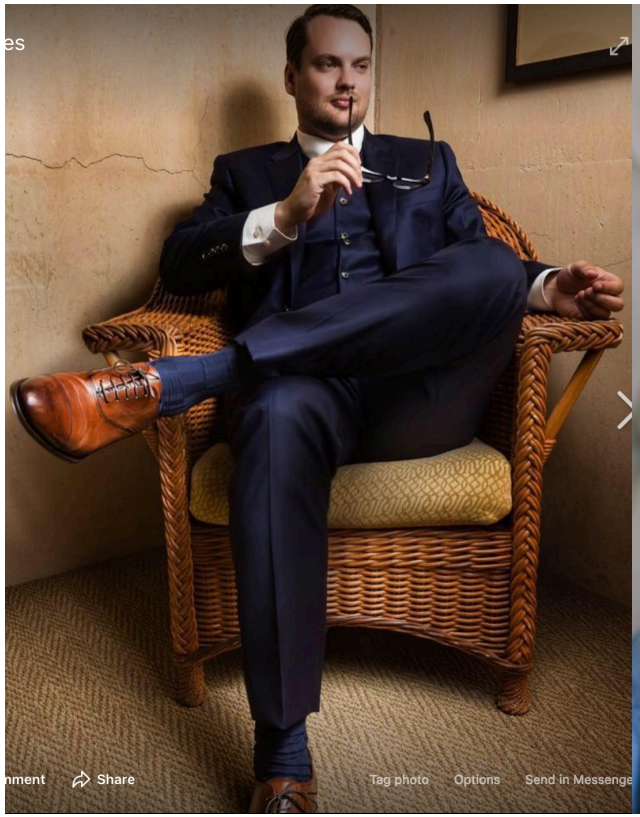
Forward-Looking Statements

During the course of this presentation, we may make forward-looking statements regarding future events or the expected performance of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results could differ materially. For important factors that may cause actual results to differ from those contained in our forward-looking statements, please review our filings with the SEC.

The forward-looking statements made in this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, this presentation may not contain current or accurate information. We do not assume any obligation to update any forward-looking statements we may make. In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionality described or to include any such feature or functionality in a future release.

splunk> .conf18

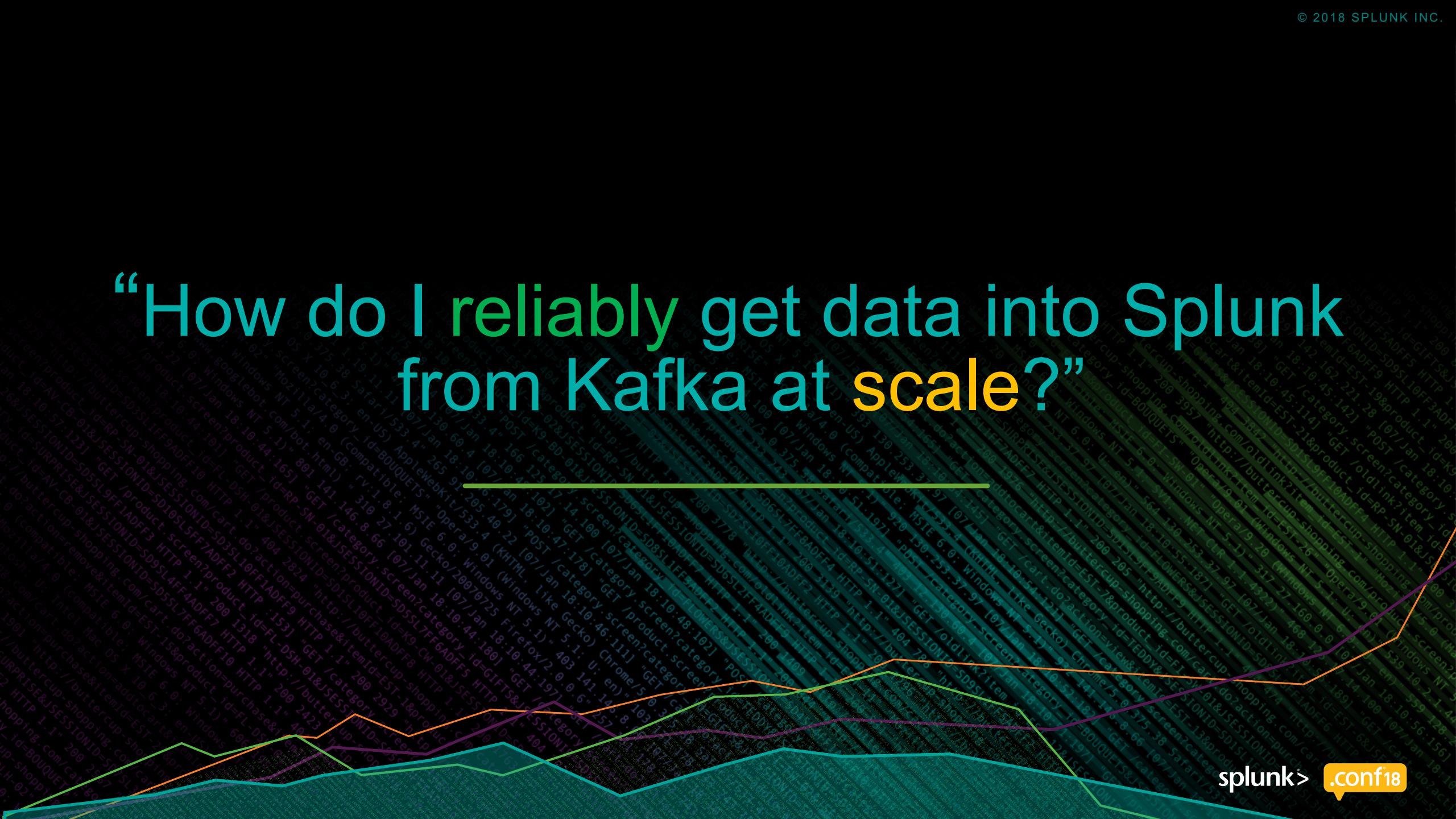
About Us



Agenda

1. Problem Statement / History Lesson
2. Kafka Connect Framework
3. Connector Deployment
4. Configuration & Tuning
5. Data Onboarding
6. Lessons Learned (Gotchas)
7. What's New!

“How do I reliably get data into Splunk from Kafka at scale?”



splunk> .conf18

Splunk Add-On for Kafka

https://www.splunk.com/blog/2016/10/31/splunking-kafka-at-scale.html

Lacks Scale

Painful Config

Poll Based

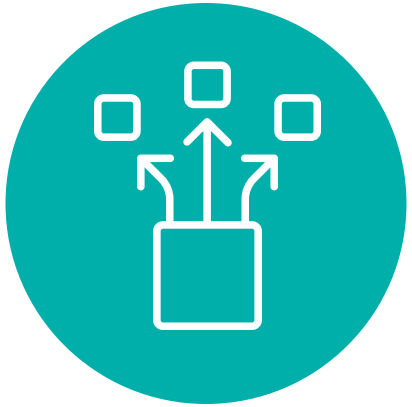
No Fault Tolerance

Pre-Bulit Panels

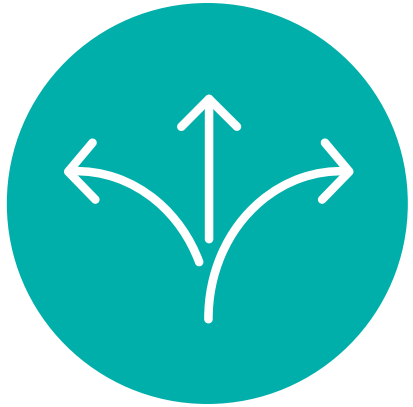
Splunk Connect for Kafka

Scalable

Configuration REST API



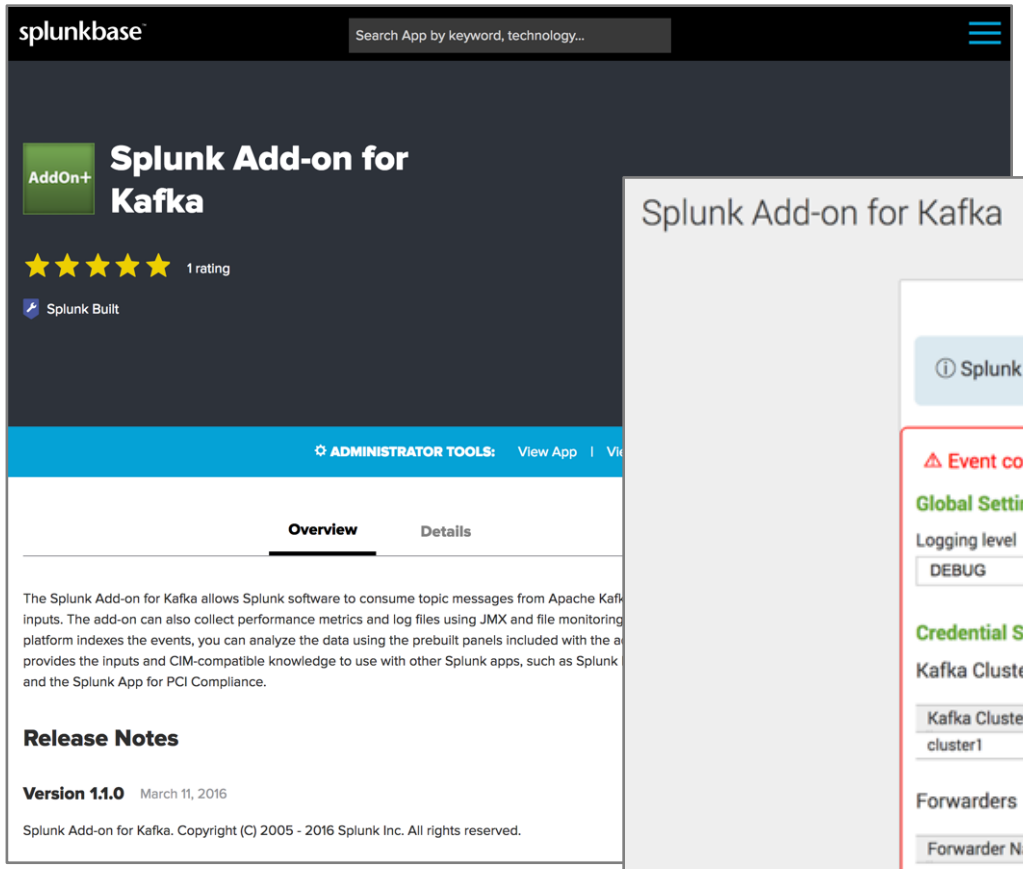
Push To HTTP Event Collector

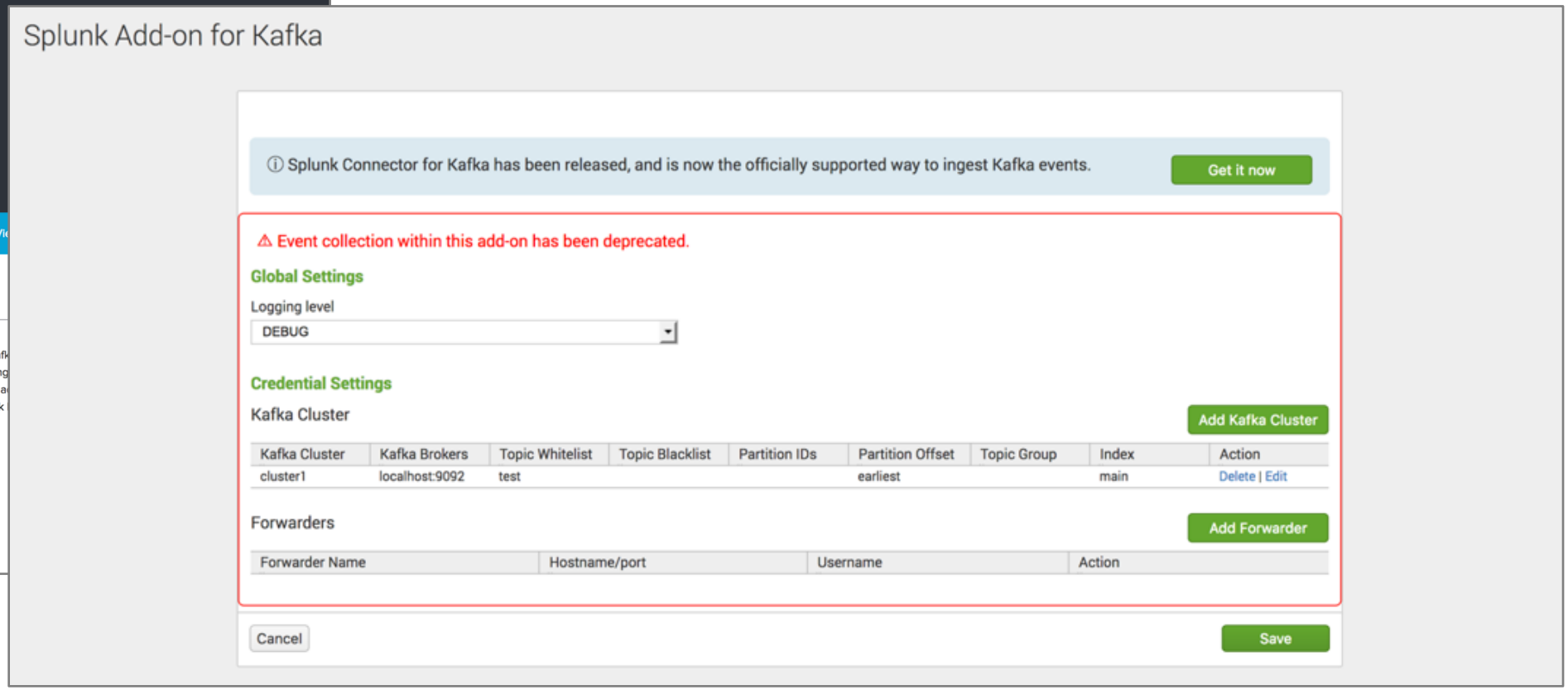


Flexible Data Onboarding

Splunk Supported

Legacy Splunk Add-on for Kafka

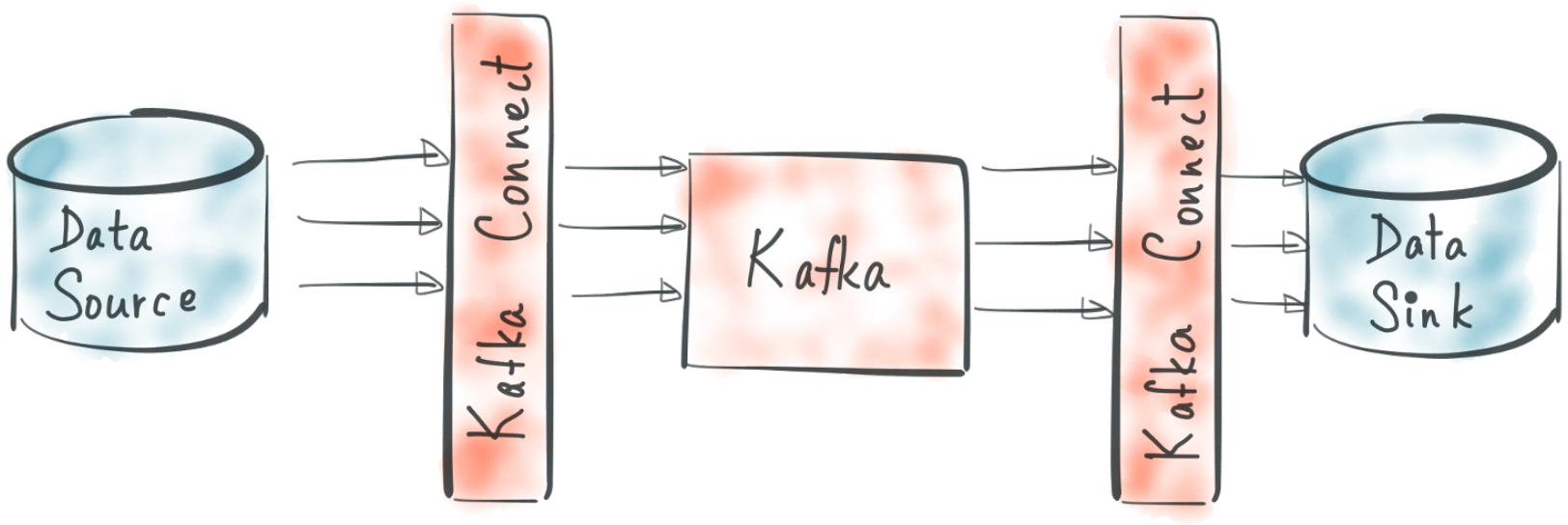




splunk> .conf18

Kafka Connect Framework

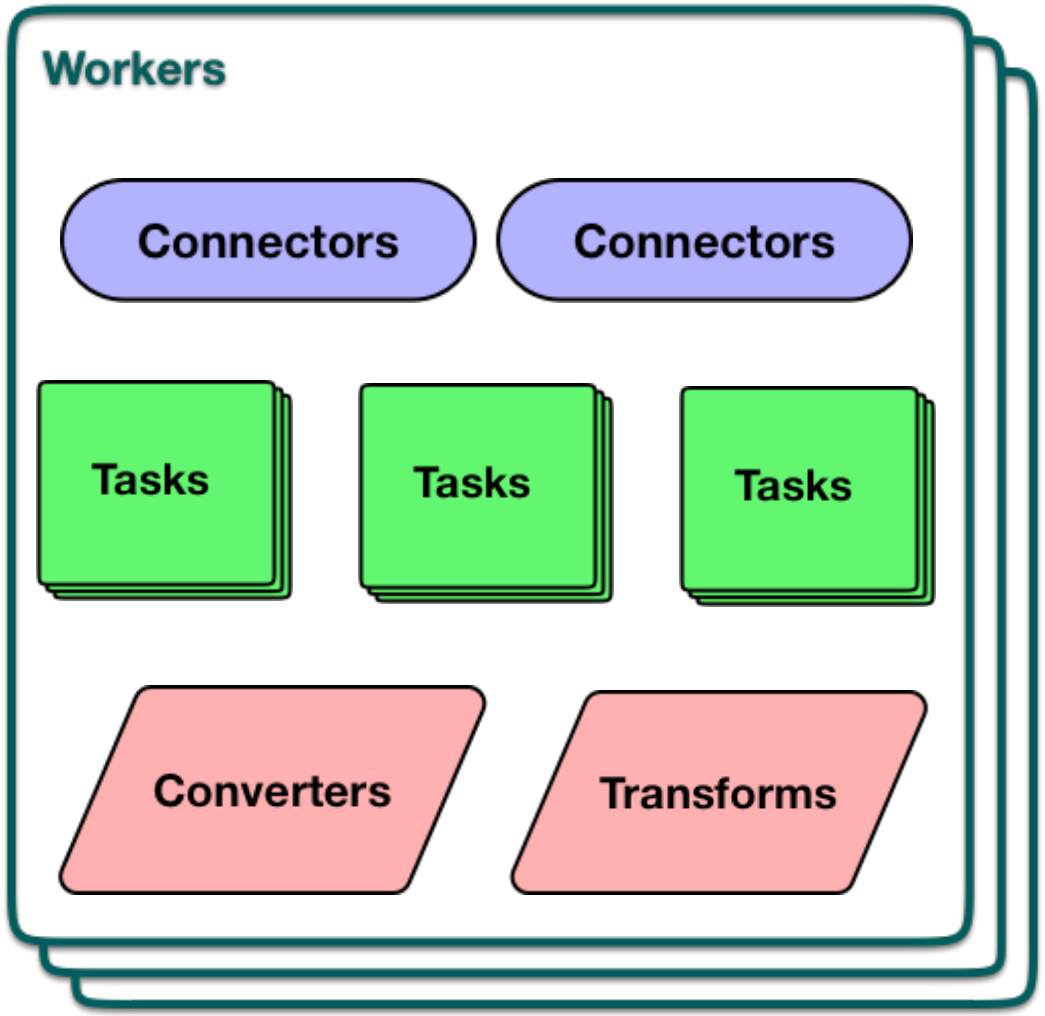
Kafka Connect



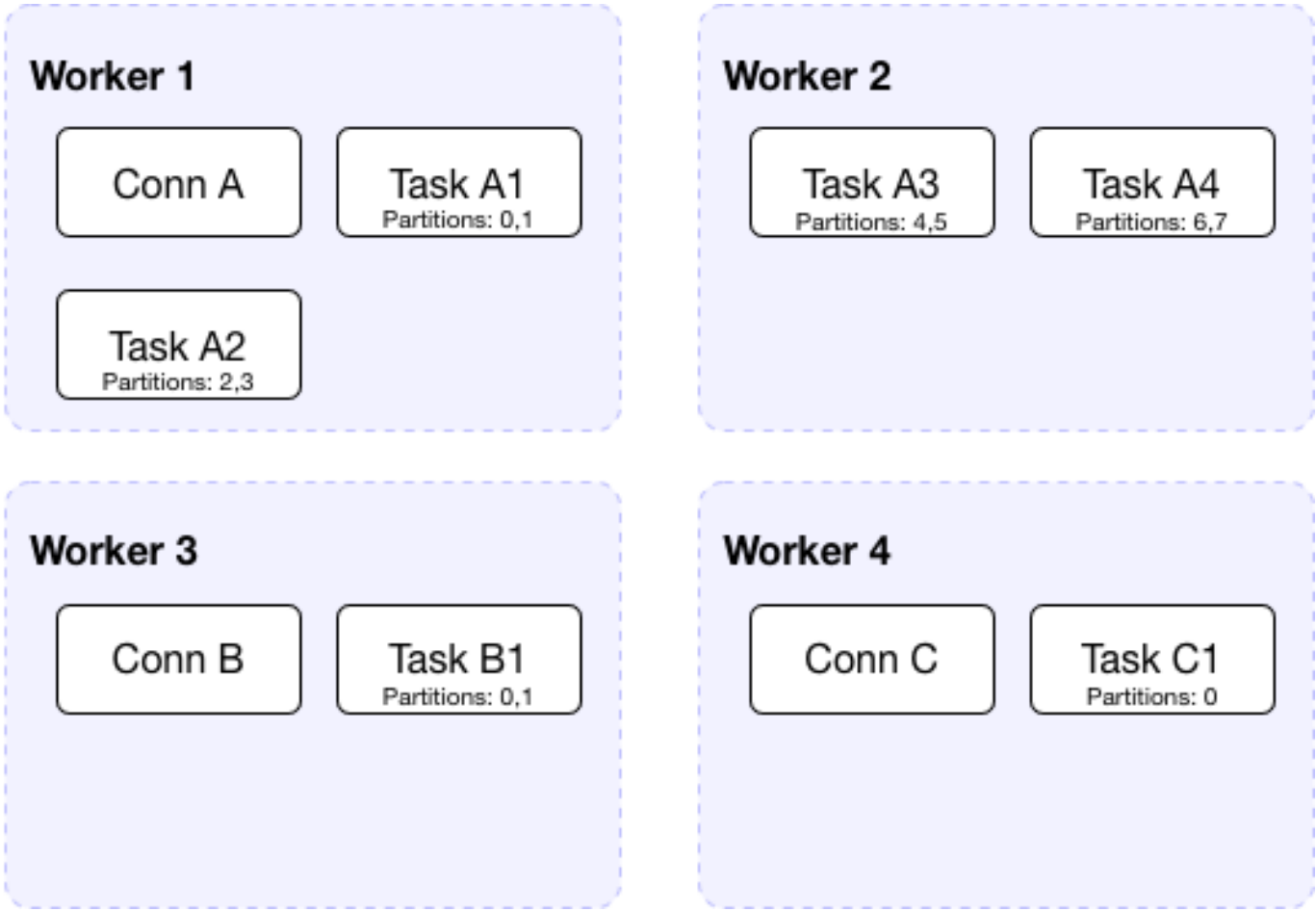
- Ships with Kafka
- Scalable & Reliable Data Streaming
- Auto offset management
- Flexible Deployment
- Source & Sink Connectors
- Streaming & Batch
- REST Interface

Source: Confluent.io

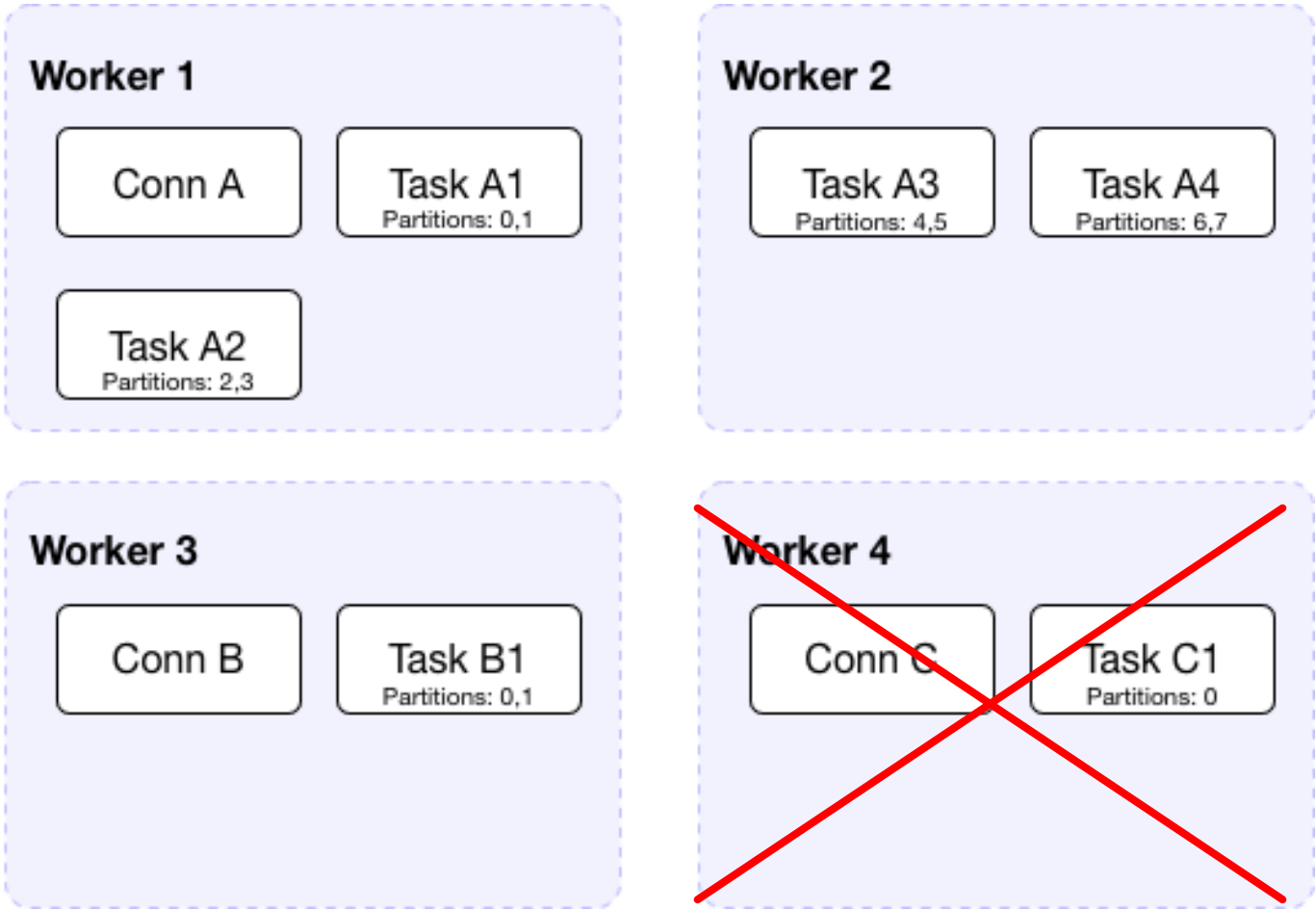
Kafka Connect Concepts



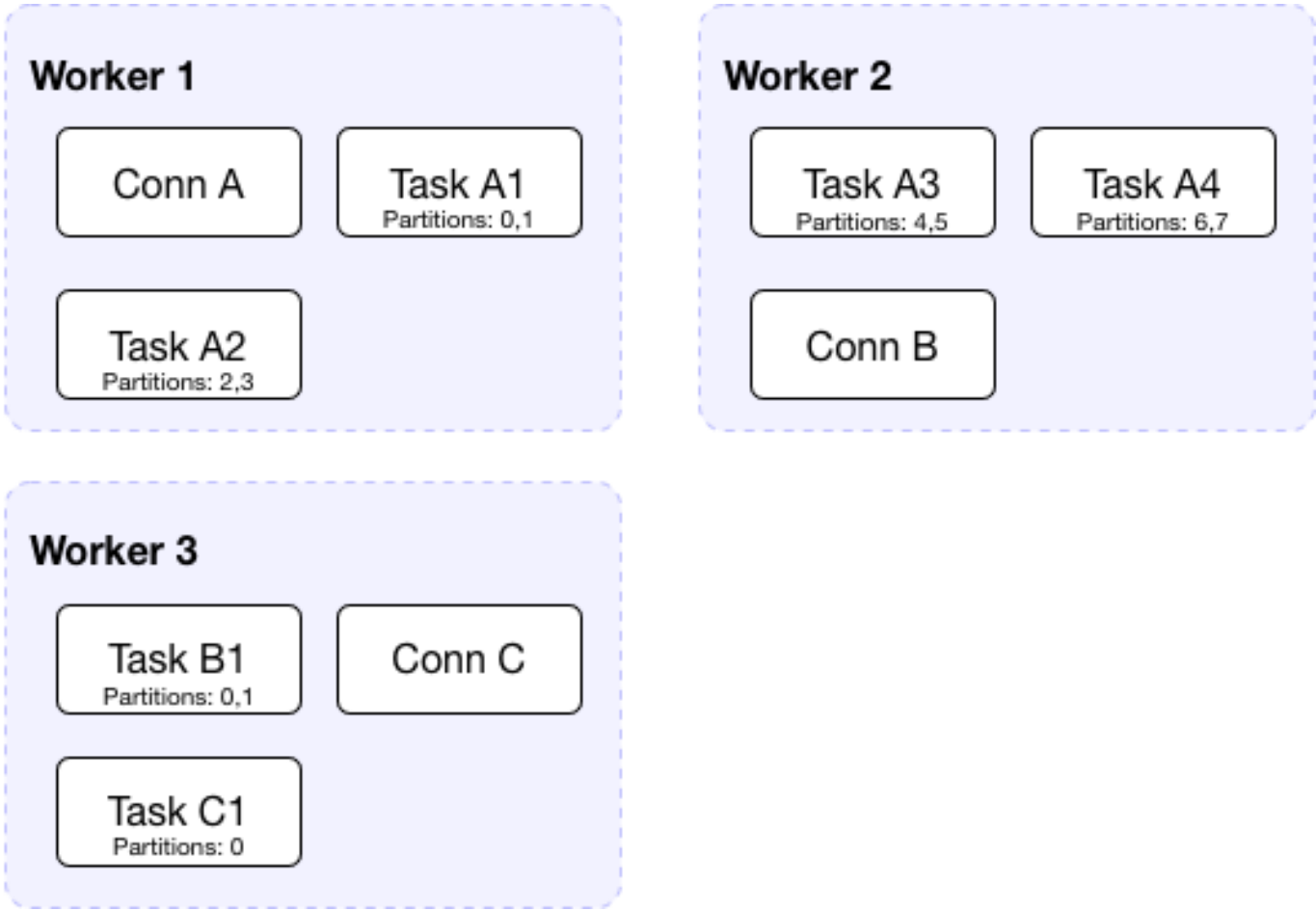
Distributed Execution Model



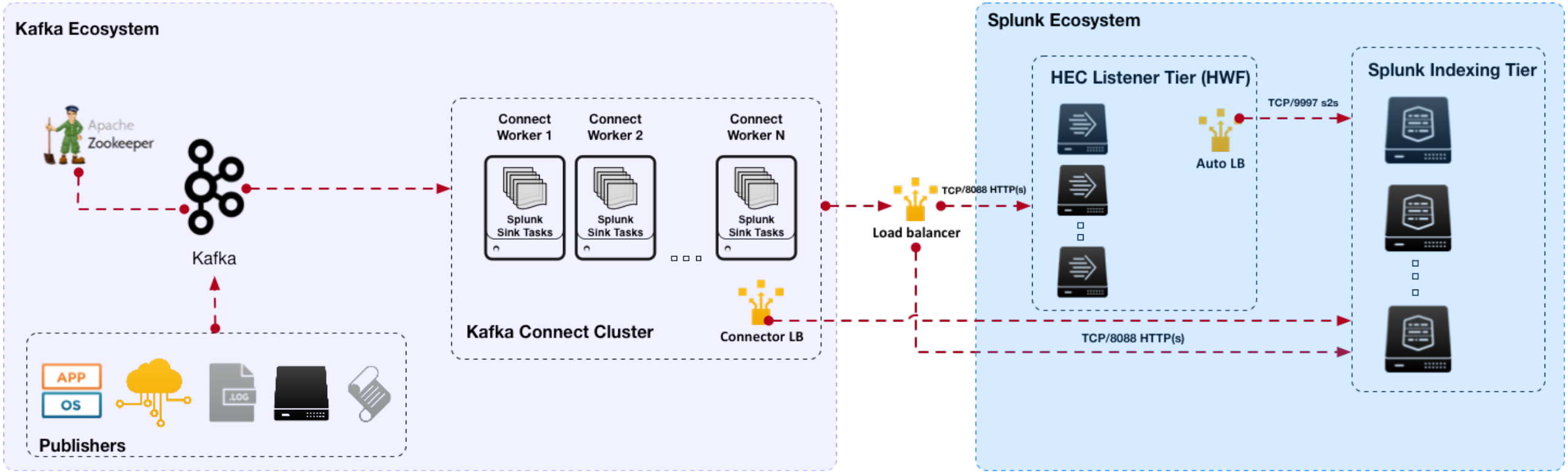
Distributed Execution Model



Distributed Execution Model



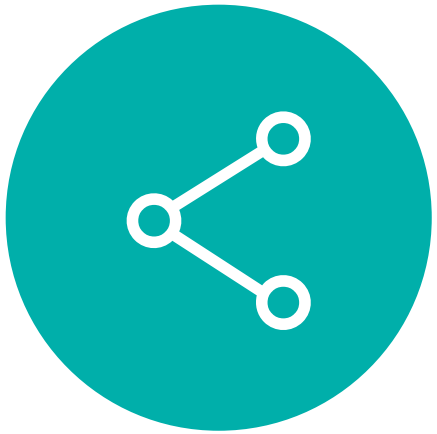
Architecture



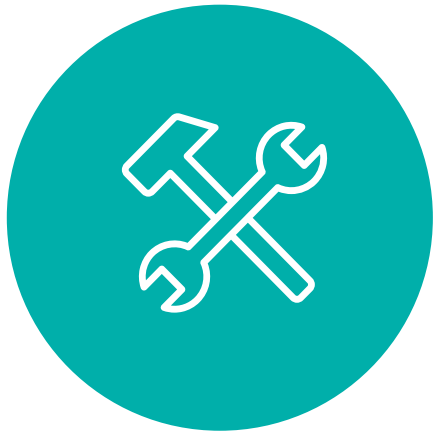
Deployment

Deployment Considerations

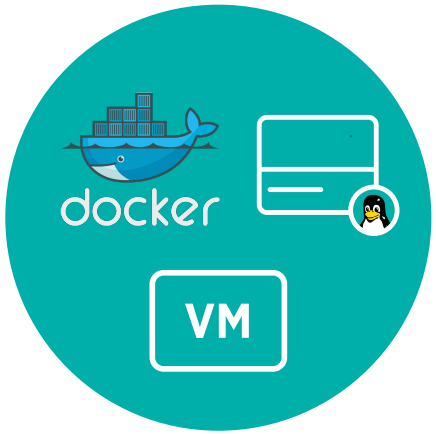
Don’t Co-Mingle With Splunk



Direct Access to Kafka Brokers

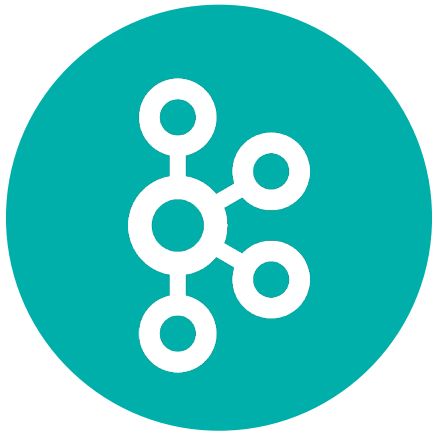


Deploy Centrally or Manage Yourself



Physical, Virtual or Containers

Pre-Requisites



Kafka 0.11.0+

Java 1.8+

Splunk 6.5+

HEC Enabled with Valid token(s)

Quick Start

Apache Kafka with KAFKA_HOME=/opt/kafka

1. Download from Splunkbase - https://splunkbase.splunk.com/app/3862/

2. On **all** Connect instances create plugin directory and copy plugin

```
# mkdir $KAFKA_HOME/plugins/splunk-connect
# cp splunk-kafka-connect-<version>.jar $KAFKA_HOME/plugins/splunk-connect
```

Quick Start (Cont..)

3. Edit $KAFKA_HOME/config/connect-distributed.properties

```
# These settings may already be configured if you have deployed
# a connector in your Kafka Connect Environment
bootstrap.servers=broker1:9092,broker2:9092
plugin.path=/opt/kafka/plugins

#Required configurations for Splunk Connect for Kafka
key.converter=org.apache.kafka.connect.storage.StringConverter
value.converter=org.apache.kafka.connect.storage.StringConverter
key.converter.schemas.enable=false
value.converter.schemas.enable=false
internal.key.converter=org.apache.kafka.connect.json.JsonConverter
internal.value.converter=org.apache.kafka.connect.json.JsonConverter
internal.key.converter.schemas.enable=false
internal.value.converter.schemas.enable=false
offset.flush.interval.ms=10000

#Recommended configurations for Splunk Connect for Kafka
group.id=kafka-connect-splunk-hec-sink
```

Quick Start (Cont..)

```
# Topic to use for storing offsets. This topic should have many partitions and be replicated and compacted.
offset.storage.topic=connect-offsets
offset.storage.replication.factor=1
#offset.storage.partitions=25

# Topic to use for storing connector and task configurations; note that this should be a single partition,
# highly replicated, and compacted topic.
config.storage.topic=connect-configs
config.storage.replication.factor=1

# Topic to use for storing statuses. This topic can have multiple partitions and should be replicated
# and compacted.
status.storage.topic=connect-status
status.storage.replication.factor=1
#status.storage.partitions=5
```

Quick Start (Cont..)

4. Start Kafka Connect

```
# $KAFKA_HOME/bin/connect-distributed.sh $KAFKA_HOME/config/connect-distributed.properties
```

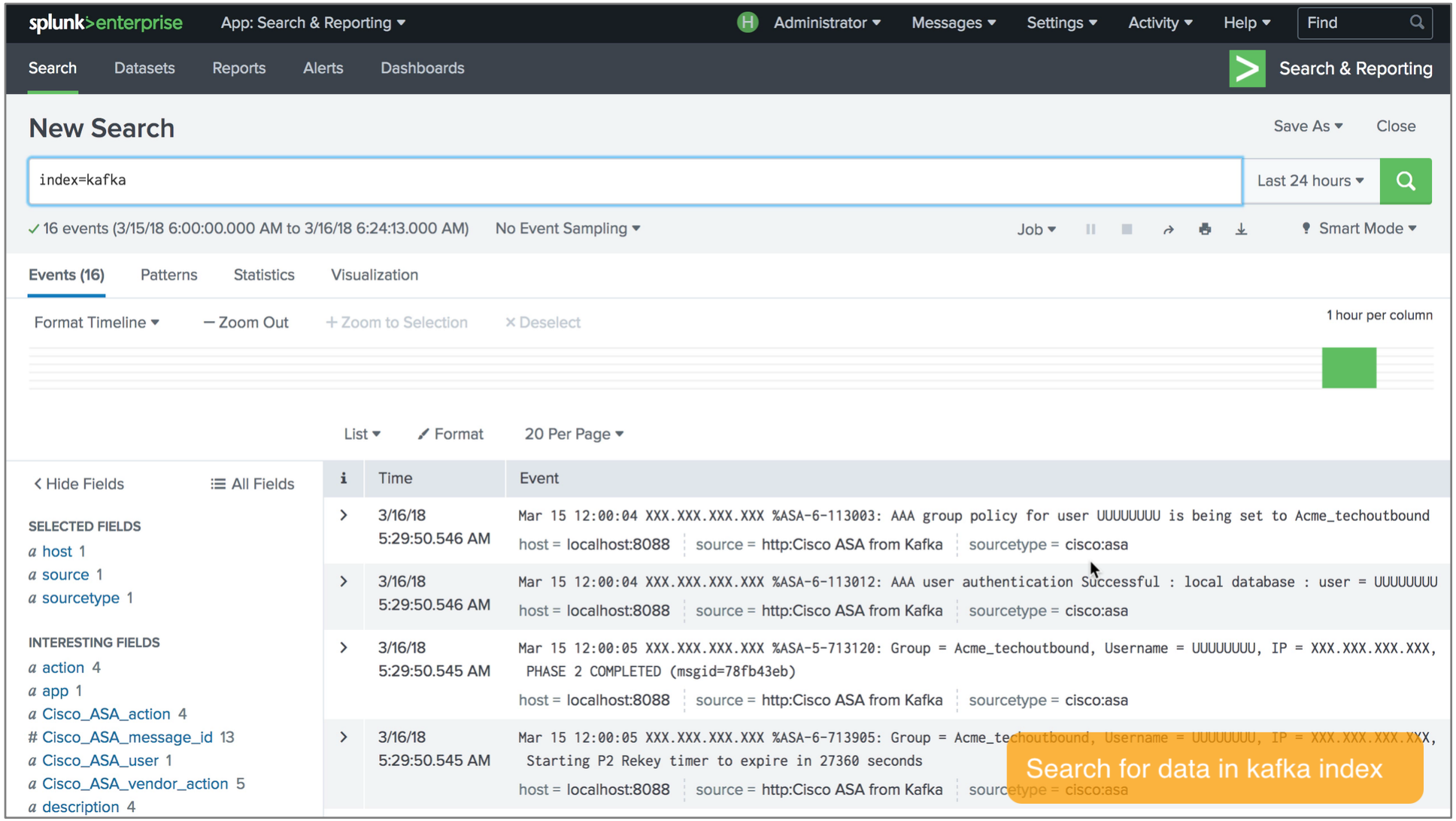
5. Verify connector

```
# curl http://<KAFKA_CONNECT_HOST>:8083/connector-plugins
{"class":"com.splunk.kafka.connect.SplunkSinkConnector","type":"sink","version":"v1.0.0”}
```

Create Connector

```
$ curl localhost:8083/connectors -X POST -H "Content-Type: application/json" -d '{
 "name": "CiscoToSplunk",
 "config": {
  "connector.class": "com.splunk.kafka.connect.SplunkSinkConnector",
  "tasks.max": "1",
  "topics":"cisco_asa",
  "splunk.hec.uri": "https://localhost:8088",
  "splunk.hec.token": "9948d956-b83e-48f8-811c-f332f0082d47",
  "splunk.hec.ack.enabled": "false",
  "splunk.hec.ssl.validate.certs": "false"
 }
}'
```

Search for Data in Splunk



splunk> .conf18

Kafka Connect REST API

Base URI – http://connect-worker:8083

`GET /connector-plugins/` - list installed connector plugins in cluster

`GET /connectors` - list active connectors

`POST /connectors` - create new connector

`GET /connectors/(string: name)` - get info about connector

`GET /connectors/(string: name)/config` - get connector config

`GET /connectors/(string: name)/status` - get connector status

`POST /connectors/(string: name)/restart` - restart connector

`PUT /connectors/(string: name)/pause` - pause connector and tasks

`PUT /connectors/(string: name)/resume` - resume paused connector

`DELETE /connectors/(string: name)/` - delete connector, halting all tasks and delete config

Kafka Connect REST API

`GET /connectors/(string: name)/tasks` - list running tasks for connector

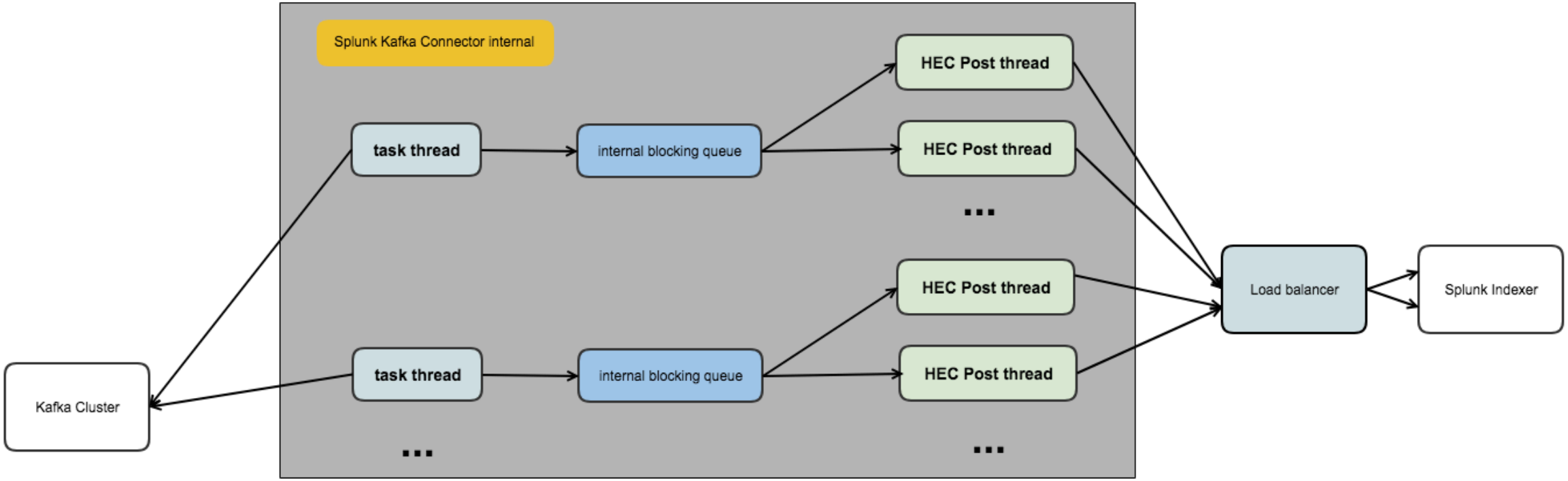
`GET /connectors/(string: name)/tasks/(int: taskid)/status` - get task's status

`POST /connectors/(string: name)/tasks/(int: taskid)/restart` - restart individual task

Configuration & Tuning

Key Components

Splunk Kafka Connect Internals



Kafka Consumer Options

- Kafka consumer options (connect-distributed.properties)
  - consumer.max.poll.records
  - consumer.max.poll.interval
  - consumer.fetch.min.bytes
  - consumer.fetch.max.bytes
  - consumer.max.partition.fetch.bytes
  - consumer.receive.buffer.bytes

```
# Set to a list of filesystem paths separated by commas (,) to enable class loading isolation for plu
# (connectors, converters, transformations). The list should consist of top level directories that in
# any combination of:
# a) directories immediately containing jars with plugins and their dependencies
# b) uber-jars with plugins and their dependencies
# c) directories immediately containing the package directory structure of classes of plugins and the
# Examples:
# plugin.path=/usr/local/share/java,/usr/local/share/kafka/plugins,/opt/connectors,
plugin.path=/Users/dtregonning/Development/kafka/kafka_2.0/connectors

consumer.max.poll.records=500
consumer.max.poll.interval=1000
consumer.fetch.min.bytes=500
consumer.fetch.max.bytes=500
consumer.max.partition.fetch.bytes=500
consumer.receive.buffer.bytes=500
```

Kafka Connect Options

Scale up

- Number of concurrent Tasks
  - Adjust the concurrency according to number of topic partitions and your hardware resources
  - Possible to overload a Splunk environment with too many tasks
- HEC batch size
  - Events are either flushed after a certain time or batched
- HTTP keepalive

Scale out

- Adding more “worker” nodes to Kafka Connect Cluster
  - When seeing consumer group “lag”

Http Event Collector(HEC) Tuning Options

Scale up

- dedicatedIoThreads = <number> (inputs.conf)
  - ~ number of cores
- parallelIngestionPipelines = <number> (servers.conf)
  - ~ 2 (can increase even larger if there are lots of CPU resources)

Scale out

- Add more HEC nodes
  - When seeing HTTP 503 server busy error
  - When seeing the processing pipeline full

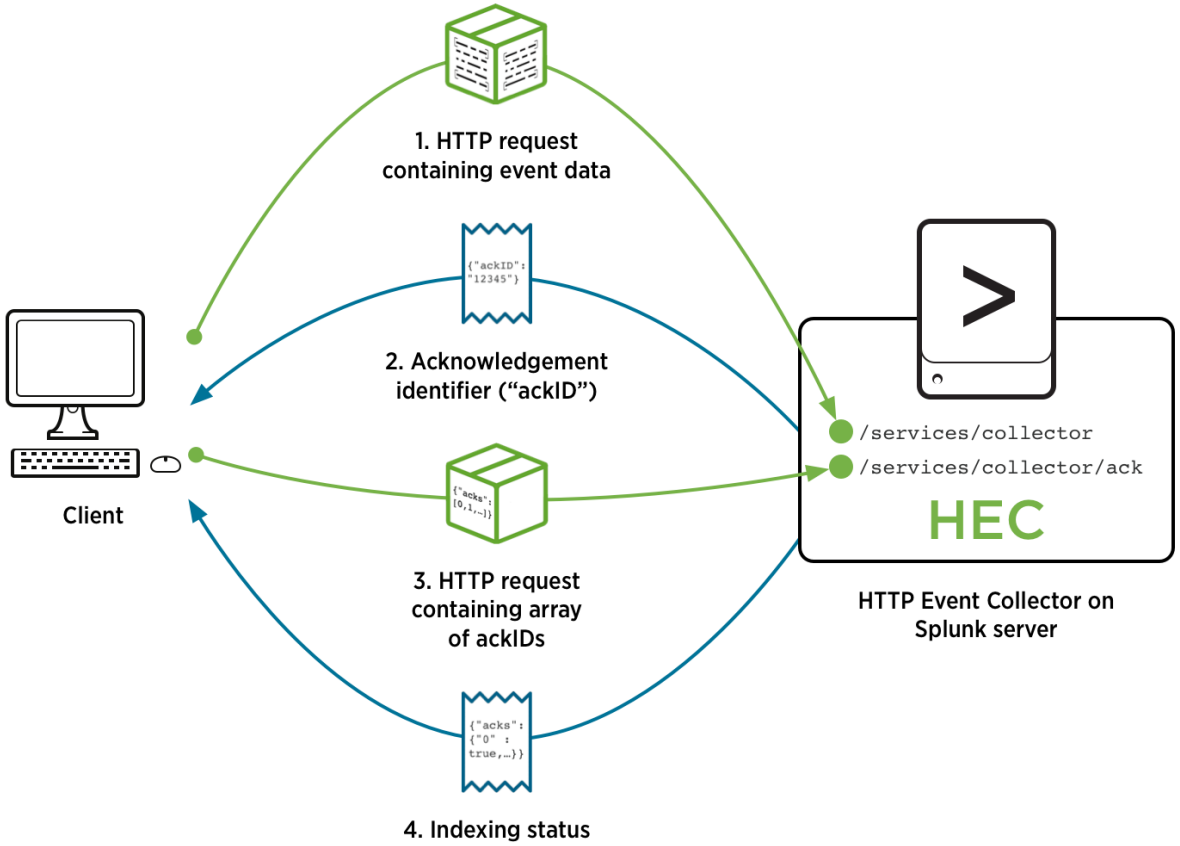
JVM Tuning Options

- KAFKA_HEAP_OPTS="-Xmx16G –Xms8G”

```
[2018-09-07 13:50:52,041] ERROR WorkerSinkTask{id=kafka-connect-splunk-0} Task threw an uncaught and unrecoverable exception (org.apache.kafka.connect.runtime.WorkerTask:177)
java.lang.OutOfMemoryError: Java heap space
        at java.util.Arrays.copyOfRange(Arrays.java:3664)
        at java.lang.String.<init>(String.java:207)
        at java.io.BufferedReader.readLine(BufferedReader.java:356)
        at java.io.BufferedReader.readLine(BufferedReader.java:389)
        at org.apache.http.conn.util.PublicSuffixListParser.parseByType(PublicSuffixListParser.java:111)
        at org.apache.http.conn.util.PublicSuffixMatcherLoader.load(PublicSuffixMatcherLoader.java:54)
        at org.apache.http.conn.util.PublicSuffixMatcherLoader.load(PublicSuffixMatcherLoader.java:63)
        at org.apache.http.conn.util.PublicSuffixMatcherLoader.getDefault(PublicSuffixMatcherLoader.java:89)
        at org.apache.http.impl.client.HttpClientBuilder.build(HttpClientBuilder.java:936)
        at com.splunk.hecclient.HttpClientBuilder.build(HttpClientBuilder.java:87)
        at com.splunk.hecclient.Hec.createHttpClient(Hec.java:275)
        at com.splunk.hecclient.Hec.newHecWithAck(Hec.java:75)
        at com.splunk.kafka.connect.SplunkSinkTask.createHec(SplunkSinkTask.java:425)
        at com.splunk.kafka.connect.SplunkSinkTask.start(SplunkSinkTask.java:47)
        at org.apache.kafka.connect.runtime.WorkerSinkTask.initializeAndStart(WorkerSinkTask.java:301)
        at org.apache.kafka.connect.runtime.WorkerSinkTask.execute(WorkerSinkTask.java:190)
        at org.apache.kafka.connect.runtime.WorkerTask.doRun(WorkerTask.java:175)
        at org.apache.kafka.connect.runtime.WorkerTask.run(WorkerTask.java:219)
        at java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:511)
        at java.util.concurrent.FutureTask.run(FutureTask.java:266)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
        at java.lang.Thread.run(Thread.java:748)
```

Data Onboarding

Indexer Acknowledgement



- Splunk guarantees "at least once" delivery semantics
- Kafka ACK Options
  - splunk.hec.ack.enabled
  - splunk.hec.ack.poll.interval
  - splunk.hec.ack.poll.threads
  - splunk.hec.event.timeout
- Using Acknowledgement may increase Splunk Connector memory usage

Load Balancing

- Connector supports hardware and software Load Balancing
- Sticky Sessions must be enabled for Indexer ACK
- Set Sticky Session timeout value to highest value available
- Duplicates may be a side-effect on session expiry
- Load Balancing Options
  - splunk.hec.total.channels
  - Set HEC channels (**splunk.hec.total.channels**) to multiple HEC endpoints (= indexers or 2 * indexers behind the load balancer)

Onboarding Data with HEC Raw and Event Endpoint

- Version 1.0 behavior (global)
- Version 1.1 behavior (event based)
- Raw endpoint *(services/collector/raw)*
  - props.conf and transforms.conf
  - Use **splunk.hec.raw.line.breaker** to inject string to be used by Splunk to break events

```
[s1] # sourcetype name
LINE_BREAKER = (####)
SHOULD_LINEMERGE = false
```

- Event endpoint *(services/collector/event)*
  - **splunk.hec.json.event.enrichment** is used to add extra fields to All Kafka Records(Global)
  - Enabling **splunk.hec.track.data** will send through some Kafka specific data (topics, partitions, offsets)
  - Good for investigating “data loss” and “data duplication”
  - Records already in the Splunk HEC JSON format… use **splunk.hec.json.event.formatted**

splunk> .conf18

What are Kafka Headers?

- Disabled by default
  - splunk.header.support = true
- Facilitates per-event routing
- Ability to map Kafka metadata to be used as Splunk metadata
  - (host, source, sourcetype, index, custom fields)
- Support added for both HEC /event and /raw

Headers (Cont..)

- Kafka Headers contain extra metadata associated with Kafka Records.
- Version 1.1 of connector includes new parameters
  - "**splunk.header.support**" – Used to enable Header support
  - "**splunk.header.custom**" – Used to add custom made headers to a Splunk event
  - "**splunk.header.index**" – Used to set a Kafka Headers value as the destination index in Spunk
  - "**splunk.header.source**" –Used to set a Kafka Headers value as the source in Spunk
  - "**splunk.header.sourcetype**" – Used to set a Kafka Headers value as the sourcetype in Spunk
  - "**splunk.header.host**" – Used to set a Kafka Headers value as the host in Spunk

Headers Example (Cont..)

REST Example to deploy connector with Header support

```
curl localhost:8083/connectors -X POST -H "Content-Type: application/json" -d'{
  "name": "splunk-kafka-connect",
    "config": {
      "connector.class": "com.splunk.kafka.connect.SplunkSinkConnector",
      "tasks.max": "1",
      "topics": "kafka-demo-header-raw-3",
      "splunk.hec.uri": "http://localhost:8222",
      "splunk.hec.ack.enabled": "false",
      "splunk.hec.raw" : "false",
      "splunk.hec.token": "b87c2ba3-84b2-4de9-8777-453ab6ccd864",
      "splunk.header.support": "true",
      "splunk.header.index" : "destination_storage",
      "splunk.header.source" : "Financial_Processing_Application",
      "splunk.header.sourcetype" : "ledger_format_1",
      "splunk.header.host": "finance.company.host"
    }
}'
```

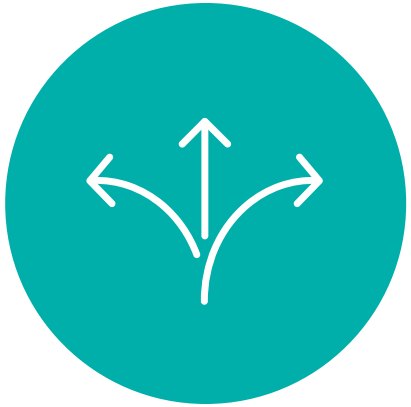
splunk> .conf18

What's new for Splunk Connect for Kafka

splunk> .conf18

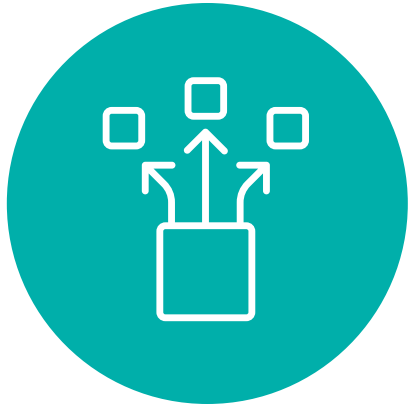
Splunk Connect for Kafka Version 1.1

Structed Data (Avro, JSON)



Per-Event Routing / Metadata Override with Kafka Headers

HEC Event Format



Sticky Session Improvements

Custom Java Keystore Location

Lessons Learned (Gotchas)

Some gotchas we've seen on our adventures

- Match Kafka Connect version to Broker Version. Use Compatibility Matrix!
- Auto-topic creation is default behavior, you may have to create them manually
- Too many partitions and tasks can overwhelm HEC
  - Start small and scale out
- Heap Size for high throughput systems
- Double check connector jar md5sum against GitHub

Don't forget to rate this session
in the .conf18 mobile app