TA-kafka-streaming-platform Documentation

Release 1

Guilhem Marchand

May 04, 2019

Contents

The Technology addon for Kafka streaming platform is a simple addon that provides indexing and searching time configuration to monitor and index the events from Apache Kafka components, as well as Confluent stack components:

- Zookeeper
- Apache Kafka Brokers
- Apache Kafka Connect
- Confluent schema-registry
- Confluent ksql-server
- Confluent kafka-rest

The addon replaces advantageously the deprecated Splunk addon for Kafka, which manages only Zookeeper and brokers components.

The addon is as well used by the Kafka monitoring application and ITSI module to provide integration between metrics and logs:

- https://da-itsi-telegraf-kafka.readthedocs.io
- https://telegraf-kafka.readthedocs.io

It is recommended to read the unified guide for Kafka and Confluent monitoring first:

https://splunk-guide-for-kafka-monitoring.readthedocs.io

CHAPTER 1

Overview:

1.1 About

- Author: Guilhem Marchand
- First release published in October 2018
- Purposes:

The Technology addon for Kafka streaming platform is a simple addon that provides indexing and searching time configuration to monitor and index the events from Apache Kafka components, as well as Confluent stack components.

1.2 Compatibility

1.2.1 Splunk compatibility

All Splunk versions are supported.

1.3 Known Issues

There are no known issues at the moment.

1.4 Support

This Splunk application is community supported.

To get support, use of one the following options:

1.4.1 Splunk Answers

Open a question in Splunk answers for the application:

- https://answers.splunk.com/app/questions/4302.html

1.4.2 Splunk community slack

Contact me on Splunk community slack, or even better, ask the community !

- https://splunk-usergroups.slack.com

1.4.3 Open a issue in Git

To report an issue, request a feature change or improvement, please open an issue in Github:

- https://github.com/guilhemmarchand/TA-kafka-streaming-platform

1.4.4 Email support

- guilhem.marchand@gmail.com

However, previous options are far betters, and will give you all the chances to get a quick support from the community of fellow Splunkers.

1.5 Download

The Splunk application can be downloaded from:

1.5.1 Splunk base

- https://splunkbase.splunk.com/app/4302

1.5.2 GitHub

- https://github.com/guilhemmarchand/TA-kafka-streaming-platform

CHAPTER 2

Deployment and configuration:

2.1 Deployment & Upgrades

2.1.1 Deployment matrix

Splunk roles	required
Search head	yes
Indexer tiers	yes
Heavy Forwarders	conditional

If Heavy Forwarders are used as intermediate forwarders, the TA must be deployed

If Splunk search heads are running in Search Head Cluster (SHC), the Splunk application must be deployed by the SHC deployer.

The deployment and configuration requires the creation of a dedicated event index (by default called **kafka**), see the implementation section.

2.1.2 Initial deployment

- The Splunk application is a tar compressed archive, which content must be placed in the apps directory of the Splunk instance
- If deployed on the Kafka component to be monitored (example: Kafka broker), the default/inputs.conf must be tuned and adapted to your context, and its inputs activated
- For SHC configurations (Search Head Cluster), extract the tgz content in the SHC deployer and publish the SHC bundle

2.1.3 Upgrades

Upgrading the Splunk application is pretty much the same operation, use one of the techniques that matches your conditions / requirements.

2.2 Installation

2.2.1 Event time stamp format

The Technology addon assumes that you rely on the default logging format provided by both Apache Kafka and Confluent stacks.

Example:

```
[2018-11-20 22:02:15,435] INFO Registered kafka:type=kafka.Log4jController MBean␣
↪(kafka.utils.Log4jControllerRegistration$)
```

This uses the following log4j configuration:

```
DatePattern='.'yyyy-MM-dd-HH
layout.ConversionPattern=[%d] %p %m (%c)%n
```

If you are relying on a different log format, copy the default/props.conf to local/ and achieve the relevant customizations.

The JVM garbage collector has its own format that is unlikely to be customized in most installation.

2.2.2 Indexers deployment

Index creation

By default, the Technology Addon assumes the usage of a specific index name:

- kafka

This is optional and you can decide to create and use a different index name.

The addon will **NOT** create the default index for you. (Splunk good practice, indexes must be configured by administrators)

Installation

Follow standard Splunk deployment rules depending on your environment:

- Standalone indexers: deploy manually or using deployment solution
- Clustered indexers: deploy the TA to the cluster master, and publish the cluster bundle

If you are using Heavy Forwarders as intermediate forwarders before reaching the indexers, it is not required to deploy the TA on the indexers BUT it is required to install it on the heavy forwarders.

2.2.3 Search Heads deployment

Installation

Follow standard Splunk deployment rules depending on your environment:

- Standalone search heads: deploy manually or using deployment solution
- Search Head Cluster: extract the content of the TA to your deployer, and publish the SHC bundle

Eventtypes customization

If you are **NOT** using the default kafka index name, you need to customize the definition of the eventtypes.

The default eventtypes are defined in:

```
TA-kafka-streaming-platform/default/eventtypes.conf
```

You can either:

- create a local/eventtypes.conf, customize the searches definition and bundle this file in your deployment
- use Splunk Web UI and achieve the modifications (must be repeated for each search heads if standalone search heads

2.2.4 Universal Forwarder deployment (Kafka components)

Deploy the Technology Addon to your Splunk Universal Forwarders that are running on the Kafka components.

In most cases, you are using a Splunk Deployment Server (DS) to manage the configuration of Universal Forwarders.

A default/inputs.conf is provided as an example:

- Each input is disabled by default
- Create a local/inputs.conf that you use to activate the inputs depending on your needs and the components you are using
- Achieve any modification that would be required, if you are using different paths, different index(es) name(s), etc.

It is acceptable to push the same package to all of your Kafka components even if some inputs will only make sense on specific servers. (aka brokers inputs on zookeeper nodes)

If the inputs is activated but match files that do not exist, Splunk will simply ignore these as long as no files match the monitor definition.

Otherwise, you can as well create a full copy of the Technology Addon on a per category basis (one for Zookeeper, one for Kafla brokers, etc.) and push this package to the relevant servers.

2.2.5 Confluent Enterprise / OSS

Zookeeper

By default, Confluent may use the same logging location for both Zookeeper and Kafka brokers, suggested configuration to avoid this:

Configuring the systemd for Zookeeper:

- Edit: `/lib/systemd/system/confluent-zookeeper.service`
- Configure the logs location with the LOG_DIR environment variable

```
[Unit]
Description=Apache Kafka - ZooKeeper
Documentation=http://docs.confluent.io/
After=network.target

[Service]
Type=simple
User=cp-kafka
Group=confluent
ExecStart=/usr/bin/zookeeper-server-start /etc/kafka/zookeeper.properties
Environment="LOG_DIR=/var/log/zookeeper"
TimeoutStopSec=180
Restart=no

[Install]
WantedBy=multi-user.target
```

- Create the log directory:

```
sudo mkdir /var/log/zookeeper
sudo chown cp-kafka:confluent /var/log/zookeeper
```

- Restart Zookeeper and verify that logs are properly generated in the directory:

```
sudo systemctl status confluent-zookeeper
```

Kafka brokers

By default, the Confluent platform generates brokers logs in the following location:

```
/var/log/kafka
```

Kafka Connect

Kafka Connect does not log to a file by default, it only logs to the console.

To change this behaviour, you need to edit the log4j configuration:

Configuring the systemd service file for Connect:

- Edit: `/lib/systemd/system/confluent-kafka-connect.service`
- Configure the logs location with the LOG_DIR environment variable

```
[Unit]
Description=Apache Kafka Connect - distributed
Documentation=http://docs.confluent.io/
After=network.target confluent-kafka.target

[Service]
Type=simple
User=cp-kafka-connect
Group=confluent
```

(continues on next page)

(continued from previous page)

```
ExecStart=/usr/bin/connect-distributed /etc/kafka/connect-distributed.properties
Environment="LOG_DIR=/var/log/connect"
TimeoutStopSec=180
Restart=no

[Install]
WantedBy=multi-user.target
```

- Create the log directory:

```
sudo mkdir /var/log/connect
sudo chown cp-kafka-connect:confluent /var/log/connect
```

Configuring log4j:

- Edit: */etc/kafka/connect-log4j.properties*
- Add a file appender:

```
log4j.rootLogger=INFO, stdout, FILE

log4j.appender.FILE=org.apache.log4j.DailyRollingFileAppender
log4j.appender.FILE.DatePattern='.'yyyy-MM-dd-HH
log4j.appender.FILE.File=${kafka.logs.dir}/connect.log
log4j.appender.FILE.layout=org.apache.log4j.PatternLayout
log4j.appender.FILE.layout.ConversionPattern=[%d] %p %m (%c)%n

log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=[%d] %p %m (%c:%L)%n

log4j.logger.org.apache.zookeeper=ERROR
log4j.logger.org.I0Itec.zkclient=ERROR
log4j.logger.org.reflections=ERROR
```

- Restart Connect and verify that the log file is being created:

```
sudo systemctl restart confluent-kafka-connect
```

Schema registry

By default, the Confluent platform generates Schema registry log in the following location:

```
/var/log/confluent/schema-registry
```

ksql-server

ksql-server does not log to a file by default, it only logs to the console.

Notes: By default, the systemd already defines the log directory location, which should already be existing with the correct permissions.

Verifying the systemd service file for ksql:

- Edit: `/lib/systemd/system/confluent-ksql.service`

- Verify the logs location with the LOG_DIR environment variable

```
[Unit]
Description=Streaming SQL engine for Apache Kafka
Documentation=http://docs.confluent.io/
After=network.target confluent-kafka.target confluent-schema-registry.target

[Service]
Type=simple
User=cp-ksql
Group=confluent
Environment="LOG_DIR=/var/log/confluent/ksql"
ExecStart=/usr/bin/ksql-server-start /etc/ksql/ksql-server.properties
TimeoutStopSec=180
Restart=no

[Install]
WantedBy=multi-user.target
```

- Verify and create the log directory if required:

```
sudo mkdir -p /var/log/confluent/ksql
sudo chown cp-kafka-connect:confluent /var/log/confluent/ksql
```

Configuring log4j:

- Edit: */etc/ksql/log4j.properties*
- Add a file appender:

```
log4j.rootLogger=INFO, stdout, FILE

log4j.appender.FILE=org.apache.log4j.DailyRollingFileAppender
log4j.appender.FILE.DatePattern='.'yyyy-MM-dd-HH
log4j.appender.FILE.File=${ksql.log.dir}/ksql-server.log
log4j.appender.FILE.layout=org.apache.log4j.PatternLayout
log4j.appender.FILE.layout.ConversionPattern=[%d] %p %m (%c)%n

log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=[%d] %p %m (%c:%L)%n

log4j.appender.streams=org.apache.log4j.ConsoleAppender
log4j.appender.streams.layout=org.apache.log4j.PatternLayout
log4j.appender.streams.layout.ConversionPattern=[%d] %p %m (%c:%L)%n

log4j.logger.kafka=ERROR, stdout
log4j.logger.org.apache.kafka.streams=INFO, streams
log4j.additivity.org.apache.kafka.streams=false
log4j.logger.org.apache.zookeeper=ERROR, stdout
log4j.logger.org.apache.kafka=ERROR, stdout
log4j.logger.org.I0Itec.zkclient=ERROR, stdout
```

- Restart ksql-server and verify that the log file is being created:

```
sudo systemctl restart confluent-ksql
```

kafka-rest

By default, the Confluent platform generates kafka-rest logs in the following location:

```
/var/log/confluent/kafka-rest
```

2.2.6 Post-deployment verifications

Once you have started to ingest the Kafka components logs, you want to ensure that:

- log parsing is achieved successfully (line breaking etc) essentially if you have customized the timestamp recognition
- eventtypes are matching the actual data

CHAPTER 3

Troubleshoot:

3.1 Troubleshoot & FAQ

CHAPTER 4

Versioniong and build history:

4.1 Release notes

4.1.1 Version 1.0.4

- Rename inputs.conf to inputs.conf.sample to allow Splunk Cloud vetting

4.1.2 Version 1.0.3

- Remove useless inputs for Zookeeper inherited from shared log4j configuration with Kafka

4.1.3 Version 1.0.2

- feature: EVENT_BREAKER props improvements
- fix: Garbage Collector indexing time parsing issues
- fix: Instructions corrections for Kafka Connect

4.1.4 Version 1.0.1

- fix: Updated default logging location for Confluent

4.1.5 Version 1.0.0

- initial and first public release