opentext™

ArcSight SmartConnectors

Software Version: CE 24.2

Configuration Guide for FlexConnector for Kafka

Document Release Date: April 2024
Software Release Date: April 2024

Legal Notices

Open Text Corporation

275 Frank Tompa Drive, Waterloo, Ontario, Canada, N2L 0A1

Copyright Notice

Trademark Notices

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number
- Document Release Date, which changes each time the document is updated
- Software Release Date, which indicates the release date of this version of the software

To check for recent updates or to verify that you are using the most recent edition of a document, go to:

https://www.microfocus.com/support-and-services/documentation

Contents

Configuration Guide for ArcSight Kafka FlexConnector

This guide provides information about installing the ArcSight Kafka FlexConnector and configuring the device for event collection.

The Arcsight Kafka FlexConnector helps you subscribe and collect events from a topic of a Kafka server or Azure Event Hubs.

Intended Audience

This guide provides information for IT administrators who are responsible for managing the ArcSight software and its environment.

Additional Documentation

The ArcSight SmartConnector documentation library includes the following resources:

- Technical Requirements Guide for SmartConnector, which provides information about operating system, appliance, browser, and other support details for SmartConnector.
- Installation and User Guide for SmartConnectors, which provides detailed information about installing SmartConnectors.
- Configuration Guides for ArcSight SmartConnectors, which provides information about configuring SmartConnectors to collect events from different sources.
- Configuration Guide for SmartConnector Load Balancer, which provides detailed information about installing Load Balancer.

For the most recent version of this guide and other ArcSight SmartConnector documentation resources, visit the documentation site for ArcSight SmartConnectors 8.4.

Contact Information

We want to hear your comments and suggestions about this book and the other documentation included with this product. You can use the comment on this topic link at the bottom of each page of the online documentation, or send an email to MFI-Documentation-Feedback@opentext.com.

For specific product issues, contact Open Text Support for Micro Focus products.

Product Overview

Kafka is a distributed event streaming technology that stores messages in a Kafka Topic and provides them to Subscribers. Events and other security relevant data are then typically streamed to ArcSight or to 3rd-party subscribers. The Arcsight Kafka FlexConnector enables you to subscribe to and read this data from a Topic or from an Azure Event Hub. The connector requires a parser to isolate relevant fields within the message data. The parser analyzes the message data and places it into field names. You must define and build parsers before configuring and installing the connector.

In a publish-subscribe system, publishers send messages to more than one consumers by using a single destination, called as topics. Topics only contain a specific event type. Consumers subscribe to messages published to topics, by using an instantaneous pull-based mechanism, which allows for handling of high volume of data in real-time. A topic can have more than one subscribers or consumers. For more information about Apache Kafka, refer to the Kafka documentation.

The Kafka FlexConnector provides the customers the ability to collect data from Kafka and write custom parsers to map, normalize, and categorize that data. Flex connectors are have all features and capabilities that SmartConnectors provide. But, the FlexConnectors do not come with out-of-the-box parsers and therefore, you must develop parsers based on your log format.

The Kafka FlexConnector can read and parse events from the following sources:

- **Apache Kafka:** Apache Kafka is a publish-subscribe based messaging queue system and a robust queue that handles a high volume of data. It enables you to pass messages from one end-point to another.
- **Microsoft Azure Event Hub:** Microsoft Azure Event Hubs provides a Kafka endpoint that can be used by your Kafka based applications as an alternative to run Kafka clusters.

  Kafka FlexConnector supports Apache Kafka protocol 1.0 and later to communicate with Azure Event Hubs. However, Microsoft Azure Event Hub Kafka endpoint does not support the Advanced Message Queuing Protocol (AMQP) and HTTPS protocols.

The Kafka FlexConnector supports events in the following formats:

- JSON
- CEF
- REGEX
- SYSLOG
- KEY-VALUE
- AVRO

Managing Parsers

Before you install and configure the FlexConnector, you must create parsers based on the log format. This section has the following infomation:

Creating Parsers

The Kafka FlexConnector cannot directly parse messages from Kafka, because there is no generic Kafka parser that can parse every message received from different devices that can send data to a given Kafka topic. You must create your individual parsers before setting up the connector.

To understand the parser file structure, see Parser File Structure in the ArcSight FlexConnector Developer's Guide.

To develop your connector, you should be familiar with FlexConnector development See the *FlexConnector Developer's Guide* for details.

Note: Consult the vendor documentation to verify the supported browsers and browser versions. If a supported browser is not used, the vendor login page might not display properly, preventing login, and you will be unable to configure the connector.

Example Parser Files

You can use the example configuration files for specific type of log format to create the parser files:

Log Format	Example Configuration files
JSON	Example Configuration File for JSON Log Format.
CEF	Configuration file is not required for CEF Log Format.
REGEX	Example Configuration File for Regex Log Format.
SYSLOG	Example Configuration File for Syslog Log Format.
KEY-VALUE	Example Configuration File for Key-Value Log Format.
AVRO	Example Configuration File for JSON Log Format. If the content type is **AVRO**, then you must create a JSON parser.

Parser File Location

The following table describes the location and filename of the configuration file used for each type of FlexConnector. The vendor or database is usually named for the device vendor (such as “superSecure”).

Type	Location	Filename
JSON	ARCSIGHT_HOME\user\agent\flexagent	vendor.jsonparser.properties
CEF	ARCSIGHT_HOME\user\agent\flexagent	
Regex	ARCSIGHT_HOME\user\agent\ flexagent	vendor.sdkrfilereader.properties
Syslog	ARCSIGHT_HOME\user\agent\flexagent\syslog	vendor.subagent.sdkrfilereader.properties
Key-Value	ARCSIGHT_HOME\user\agent\flexagent	
Avro	ARCSIGHT_HOME\user\agent\flexagent	

Serializing and Deserializing Data

The connector consumes only data that is serialized using the correct serializers, because the corresponding deserializers are not configurable. The Serializers are responsible for converting objects into data types and also deserializing parsed data to be converted back into complex types, after first validating the incoming data.

- **To serialize data, use the following serializers:**
  - Key serializer: `org.apache.kafka.common.serialization.StringSerializer`
  - Value serializer: `org.apache.kafka.common.serialization.BytesSerializer`
- **To deserialize data, the Kafka FlexConnector uses the following deserializers:**
  - Key deserializer: `org.apache.kafka.common.serialization.StringDeserializer`
  - Value deserializer: `org.apache.kafka.common.serialization.BytesDeserializer`

Next Step:

- Stream Logs

Overriding Parser Files

To override parser files:

1. Stop the connector and navigate to the path `<connector_home>/current/users/agent/fcp/connectorname_log>`, for example

`<connector_home>/current/users/agent/fcp/cisco_syslog>`

The following files should be found under the location:

`cisco_syslog.subagent.sdkrfilereader.properties`

`cisco_sdsyslog.subagent.sdkrfilereader.properties`

As well as an "extra processor" parser required for main-level REGEX type agents:

`cisco_sdsyslog.sdkkeyvaluefilereader.properties`

2. In order to override these files, create the sub-folder structure and the required file(s) under

   `<connector_home>/current/users/agent/fcp/cisco_syslog`
3. Make sure the override only includes the changes or additions to the base /shipped parser.
4. Start the connector.
5. To confirm the override was successful, go to the `agent.out.wrapper.log` file look for the **"An over-ride file was found and loaded"** note.

> **Note**: The Override file should be created with the same file name and under the same folder location and replaced without affecting or making changes in the `agent.properties` file.

Streaming Logs

You can configure the flex connectors to stream logs from Apache Kafka and Azure Event Hubs. Select one of the following topics based on your event source.

Collecting Events From Apache Kafka

To enable the FlexConnector to read data from Kafka topics, you must configure the connector to read data from Kafka topics, after you have installed it. You can also configure advanced authentication and enable inter broker SSL communication.

When using Apache Kafka protocol with your clients, set the configuration for authentication and encryption using the SASL mechanisms.

To install and configure the FlexConnectors to collect event data from Apache Kafka, complete the following steps:

1. Prerequisites
2. Installing the FlexConnector
3. Copy the Parser file
4. Enable SSL Encryption and Authentication (Optional)

5. Enable SSL for Inter Broker Communication (Optional)
6. Configure the FlexConnector

Prerequisite

Before installing the FlexConnector, make sure that the following are available:

- Create the parser file based on the log format.
- Local access to the machine where the FlexConnector is to be installed.
- Vendor login credentials (user name and password). During the configuration, you are redirected to the vendor's login page, where you will log into the vendor's application using your vendor credentials. After you log into the vendor application, the connector can access and collect vendor log data.

Installing the FlexConnector

1. Download the latest executable for your operating system.
2. Start the FlexConnector Installer by running the executable.
3. Follow the installation wizard to install the core software
4. Exit the Installation wizard.

Copying the Parser File

You must copy the parser configuration file to the `ARCSIGHT_HOME\user\agent\flexagent` folder. For more information about the specific parser file locations, see the Parser File Locations and Names section in Developer's Guide to FlexConnectors.

Enabling SSL Encryption and Authentication (Optional)

If you want to enable advanced authentication, then you must configure the truststore, keystore, and password in the `server.properties` file of every broker.

Note: `ssl.truststore.password` is optional but highly recommended. If a password is not set, access to the truststore is still available, but integrity checking is disabled.

As Passwords are directly stored in the broker configuration file, restrict access to these by using file system permissions.

To configure trust store, keystore, and passwords, add the following lines in the server.properties file of every broker:

```
ssl.truststore.location=/var/private/ssl/kafka.server.truststore.jks
```

```
ssl.truststore.password=test1234
ssl.keystore.location=/var/private/ssl/kafka.server.keystore.jks
ssl.keystore.password=test1234
ssl.key.password=test1234
```

Enabling SSL for Inter Broker Communication (Optional)

You can add a configuration to the `config/server.properties` file to set up a secure communication between brokers, so that the Kafka brokers are available only through SSL. You must restart brokers after making changes to configuration.

- To enable SSL for inter broker communication, add the following line:

  `security.inter.broker.protocol=SSL`
- Configure the Apache Kafka broker ports which listen to client and inter-broker SSL connections. Configure the `listeners` and the `advertised.listeners`, in case the value is different.

  ```
  listeners=SSL://kafka1:9093
  advertised.listeners=SSL://0.0.0.0:9093
  ```
- Configure the `PLAINTEXT` ports if:
  - SSL is not enabled for inter-broker communication.
  - Some clients connecting to the cluster do not use SSL.

  ```
  listeners=PLAINTEXT://kafka1:9092,SSL://kafka1:9093
  advertised.listeners=PLAINTEXT://0.0.0.0:9092,SSL://0.0.0.0:9093
  ```

  Note: `advertised.host.name` and `advertised.port` configure a single `PLAINTEXT` port are incompatible with secure protocols. Use `advertised.listeners` instead.
- To enable the broker to authenticate clients (2-way authentication), configure all the brokers for client authentication. It is recommended to set this value to `required`.

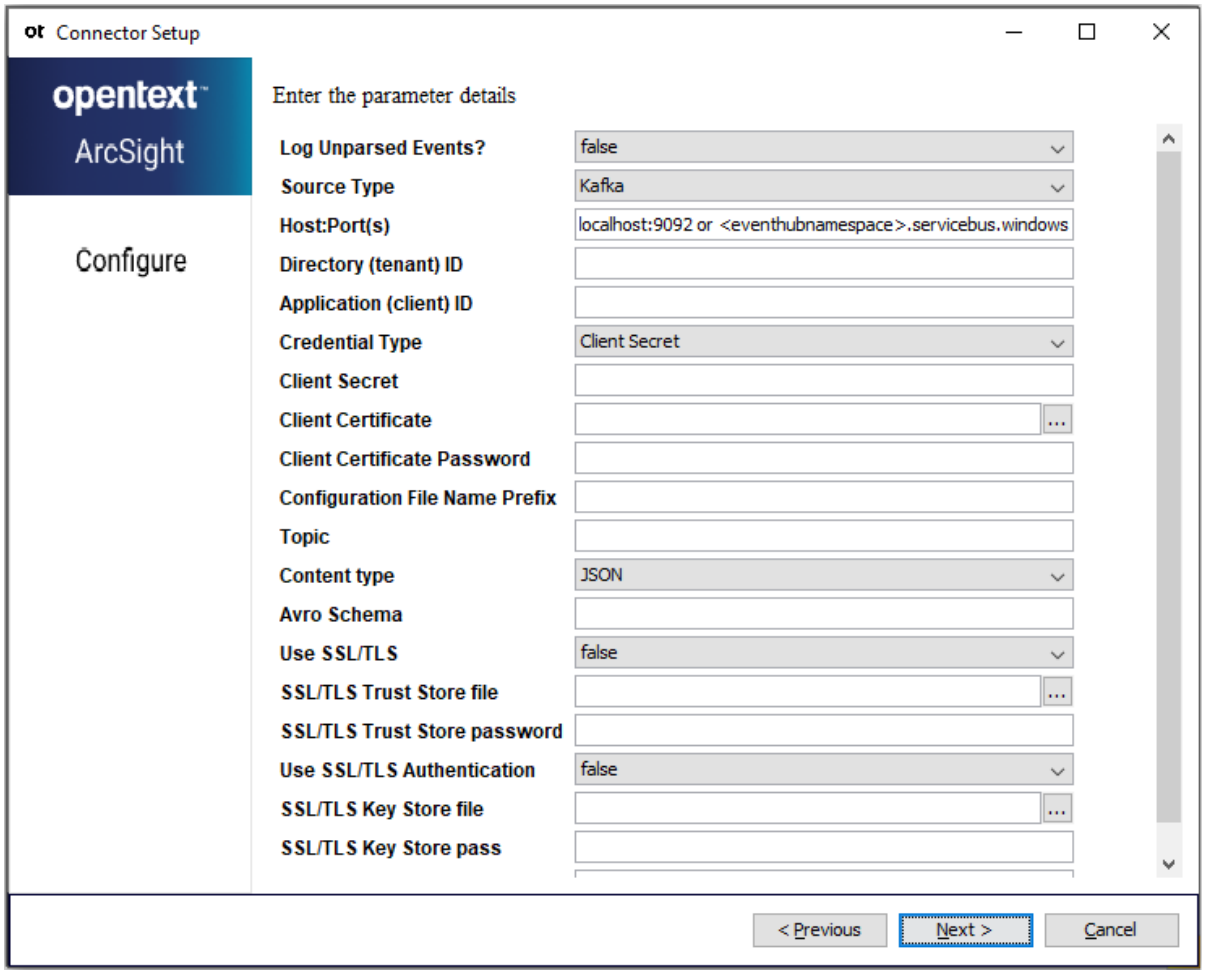
  `ssl.client.auth=required`

  Note: Do not use `requested` as it creates a false sense of security.

Important: If any of the SASL authentication mechanisms are enabled on a given listener, the SSL client authentication is disabled, even if `ssl.client.auth=required` is previously configured. The broker will only authenticate clients via SASL on that listener.

Configuring the FlexConnector

1. Browse to `$ARCSIGHT_HOME/current/bin`, then double-click **runagentsetup.bat** file to start the SmartConnector Configuration Wizard.
2. Specify the relevant Global Parameters, when prompted.
3. From the **Type** drop-down menu, select **ArcSight FlexConnector Kafka** and click **Next**.



4. Enter the parameter details and click **Next**.

Parameter	Setting
Log Unparsed Events?	Log unparsed events on the log folder and only use it for regex and syslog content types.
Source Type	Select the required source type as either **Azure Event Hub** or **Kakfa** to read the data from.

Parameter	Setting
Host:Port(s)	Enter the host and port of the Kafka server or the Event Hubs Namespace. For the host value, refer to the Overview section of the Event Hub Namespace. The recommended port value is 9093.
Directory (tenant) ID	Enter the Directory (tenant) ID of your registered application. For this value, refer to the Overview section of the registered application.
Application (Client) ID	Enter the Client ID generated for your registered application. For this value, refer to the Overview section of the registered application.
Credential Type	Enter the credential type as either **Client Secret** or **Client Certificate**.
Client Secret	Enter the client secret value generated while registering the application. This value is obfuscated. This field is mandatory if the Credential Type is Client secret.
Client Certificate	Specify the client certificate path. This field is mandatory if the Credential Type is Client Certificate.
Client Certificate Password	Enter the password of client certificate.
Configuration File Name Prefix	Enter prefix for the name of the parser file. For example: for `$ARCSIGHT_HOME\current\user\agent\flexagent\google.jsonparser.properties`. You can enter the prefix google, and the connector assumes the file name is `google.jsonparser.properties` and resides in `$ARCSIGHT_ HOME\current\user\agent\flexagent`. For more information, see Developer's Guide to FlexConnectors.
Topic	Enter Event Hub(s) namespace name.
Content type	Select content type from the drop-down list. The supported content types are: **JSON**, **CEF**, **SYSLOG**, **REGEX**, **KEY-VALUE**, and **AVRO**.
Avro Schema	(Applicable only if the content type is Avro) Enter a schema file name with full path and file extension (For example: `/opt/TestSchema.avsc`). Note that this must be the same schema that was used while writing the security events to Kafka topic.
Use SSL/TLS	Select **true**, if you have configured advanced authentication or if Kafka server requires it for encrypted data.
SSL/TLS Trust Store file	(Applicable only if you have selected true for the previous option) Enter file path of the SSL/TLS Trust Store file. To enable SASL plain authentication, do not specify any value here.
SSL/TLS Trust Store password	(Applicable only if you have selected true for the previous option) Enter the **SSL/TLS Trust Store** password of the store file above. To enable SASL plain authentication, do not specify any value here.

Parameter	Setting
Use SSL/TLS Authentication	(Applicable only if you have selected true for the previous option) Select **true** from the drop-down list if the Kafka server requires it for authentication. You also need to enable the **Use SSL/TLS** parameter. To enable SASL plain authentication, select **false** from the drop-down list.
SSL/TLS Key Store file	(Applicable only if you have selected true for the previous option) Enter the file path of the SSL/TLS Key Store file. To enable SASL plain authentication, do not specify any value here.
SSL/TLS Key Store pass	(Applicable only if you have selected true for the previous option) Enter the SSL/TLS Key Store password. To enable SASL plain authentication, do not specify any value here.
SSL/TLS Key password	(Applicable only if you have selected true for the previous option) Enter the SSL/TLS Key password. To enable SASL plain authentication, do not specify any value here.

5. Select a destination and configure parameters.
6. Specify a name for the connector.
7. (Conditional) If you have selected **ArcSight Manager** as the destination, the certificate import window for the ArcSight Manager is displayed. Select **Import the certificate to the connector from destination** and click **Next**.

   The certificate is imported and the Add connector Summary window is displayed.

   (If you select **Do not import the certificate to connector from destination**, the connector installation will end.)
8. Select whether you want to run the connector as a service or in the standalone mode.
9. Complete the installation.

Next Step:

- Configure Advanced Parameters (Optional)
- Run the Connector

Collecting Events From Azure Event Hub Kafka

The Azure Event Hub Kafka endpoint can be used from applications with just a minimal configuration change:

- Update the connection string in the configurations to point to the Kafka endpoint exposed by your event hub instead of pointing to a Kafka cluster and start streaming events from the applications that use the Kafka protocol into Event Hubs.
- This integration also supports frameworks like Kafka Connect, which is currently in preview.

The Arcsight Kafka FlexConnector uses Shared Access Signature (SAS) to authorize access to secure resources.

For more information, see the Azure Documentation.

To install and configure the FlexConnectors to collect event data from Apache Kafka, complete the following steps:

1. Configuration
2. Prerequisites
3. Installing the FlexConnector
4. Copy the Parser file
5. Enable SASL_SSL Authentication
6. Enable Shared Access Signature (Optional)
7. Configure the FlexConnector

Prerequisite

Before installing the FlexConnector, make sure that the following are available:

- Create the parser file based on the log format.
- Local access to the machine where the FlexConnector is to be installed.
- Vendor login credentials (user name and password). During the configuration, you are redirected to the vendor's login page, where you will log into the vendor's application using your vendor credentials. After you log into the vendor application, the connector can access and collect vendor log data.

Installing the FlexConnector

1. Download the latest executable for your operating system.
2. Start the FlexConnector Installer by running the executable.
3. Follow the installation wizard to install the core software
4. Exit the Installation wizard.

Copying the Parser File

You must copy the parser configuration file to the `ARCSIGHT_HOME\user\agent\flexagent` folder. For more information about the specific parser file locations, see the Parser File Locations and Names section in Developer's Guide to FlexConnectors.

Enabling SASL_SSL Authentication

Every time events are published or consumed from Event Hubs for Kafka, your clients are trying to access the Event Hubs resources. Ensure that the resources are accessed with an authorized entity. Event Hubs for Kafka require the TLS-encryption (as all data in transit with Event Hubs is TLS encrypted). This can be done by specifying the SASL_SSL option in the configuration file.

Enabling Shared Access Signatures(Optional)

Azure Event Hubs also provide the Shared Access Signatures (SAS) for delegated access to Event Hubs for Kafka resources. Authorizing access with an OAuth 2.0 token-based mechanism provides superior security and ease of use over SAS. The built-in roles can also eliminate the need for ACL-based authorization, which has to be maintained and managed by the user.

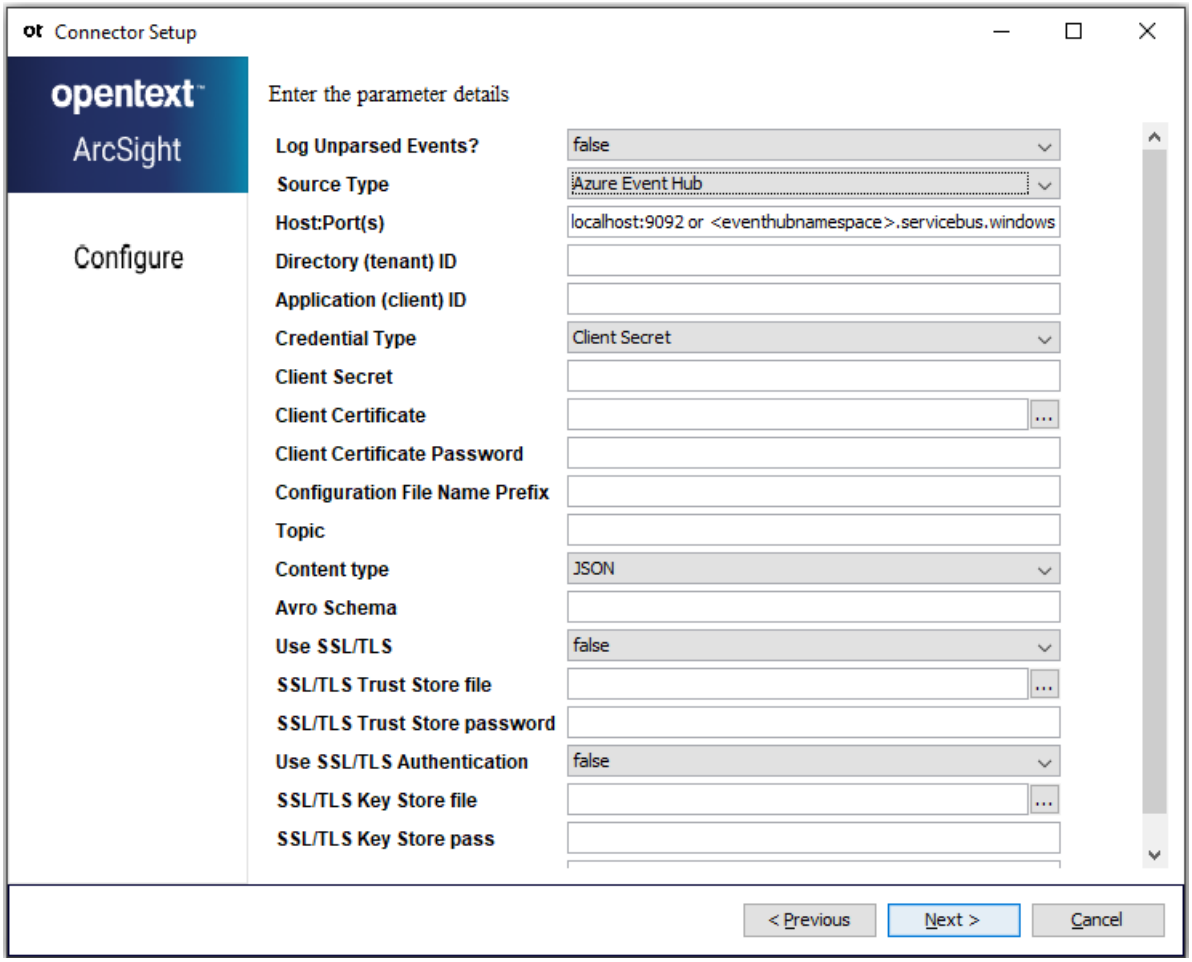
This feature can be used with your Kafka clients by specifying the SASL_SSL for the protocol and PLAIN for the mechanism, as shown in the following example:

```
bootstrap.servers=NAMESPACENAME.servicebus.windows.net:9093

security.protocol=SASL_SSL

sasl.mechanism=PLAIN

sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required username="$ConnectionString" password="
{YOUR.EVENTHUBS.CONNECTION.STRING}";
```

Note: When using SAS authentication with Kafka clients, established connections are not disconnected after the SAS key is regenerated.

Configuring the FlexConnector

1. Browse to `$ARCSIGHT_HOME/current/bin`, then double-click **runagentsetup.bat** file to start the SmartConnector Configuration Wizard.
2. Specify the relevant Global Parameters, when prompted.
3. From the **Type** drop-down menu, select **ArcSight FlexConnector Kafka** and click **Next**.



4. Enter the parameter details and click **Next**.

Parameter	Setting
Log Unparsed Events?	Log unparsed events on the log folder and only use it for regex and syslog content types.
Source Type	Select the required source type as either **Azure Event Hub** or **Kakfa** to read the data from.
Host:Port(s)	Enter the host and port of the Kafka server or the Event Hubs Namespace. For the host value, refer to the Overview section of the Event Hub Namespace. The recommended port value is 9093.
Directory (tenant) ID	Enter the Directory (tenant) ID of your registered application. For this value, refer to the Overview section of the registered application.
Application (Client) ID	Enter the Client ID generated for your registered application. For this value, refer to the Overview section of the registered application.
Credential Type	Enter the credential type as either **Client Secret** or **Client Certificate**.

Parameter	Setting
Client Secret	Enter the client secret value generated while registering the application. This value is obfuscated. This field is mandatory if the Credential Type is Client secret. For detailed information, see Registering the Application in Azure AD section of the Configuration Guide of Microsoft Azure Event Hub Connector.
Client Certificate	Specify the client certificate path. This field is mandatory if the Credential Type is Client Certificate. For detailed information, see Registering the Application in Azure AD section of the Configuration Guide of Microsoft Azure Event Hub Connector.
Client Certificate Password	Enter the password of client certificate.
Configuration File Name Prefix	Enter prefix for the name of the parser file. For example: for `$ARCSIGHT_HOME\current\user\agent\flexagent\google.jsonparser.properties`. You can enter the prefix google, and the connector assumes the file name is `google.jsonparser.properties` and resides in `$ARCSIGHT_ HOME\current\user\agent\flexagent`. For more information, see Developer's Guide to FlexConnectors.
Topic	Enter Event Hub(s) namespace name.
Content type	Select content type from the drop-down list. The supported content types are: **JSON**, **CEF**, **SYSLOG**, **REGEX**, **KEY-VALUE**, and **AVRO**.
Avro Schema	(Applicable only if the content type is Avro) Enter a schema file name with full path and file extension (For example: `/opt/TestSchema.avsc`). Note that this must be the same schema that was used while writing the security events to Kafka topic.
Use SSL/TLS	Select **true**, if you have configured advanced authentication or if Kafka server requires it for encrypted data.
SSL/TLS Trust Store file	(Applicable only if you have selected true for the previous option) Enter file path of the SSL/TLS Trust Store file. To enable SASL plain authentication, do not specify any value here.
SSL/TLS Trust Store password	(Applicable only if you have selected true for the previous option) Enter the **SSL/TLS Trust Store** password of the store file above. To enable SASL plain authentication, do not specify any value here.
Use SSL/TLS Authentication	(Applicable only if you have selected true for the previous option) Select **true** from the drop-down list if the Kafka server requires it for authentication. You also need to enable the **Use SSL/TLS** parameter. To enable SASL plain authentication, select **false** from the drop-down list.

Parameter	Setting
SSL/TLS Key Store file	(Applicable only if you have selected true for the previous option) Enter the file path of the SSL/TLS Key Store file. To enable SASL plain authentication, do not specify any value here.
SSL/TLS Key Store pass	(Applicable only if you have selected true for the previous option) Enter the SSL/TLS Key Store password. To enable SASL plain authentication, do not specify any value here.
SSL/TLS Key password	(Applicable only if you have selected true for the previous option) Enter the SSL/TLS Key password. To enable SASL plain authentication, do not specify any value here.

5. Select a destination and configure parameters.
6. Specify a name for the connector.
7. (Optional) If you have selected ArcSight Manager as the destination, the certificate import window for the ArcSight Manager is displayed. Select **Import the certificate to the connector from destination** and click **Next**.

   The certificate is imported and the Add connector Summary window is displayed.

   (If you select **Do not import the certificate to connector from destination**, the connector installation will end.)
8. Select whether you want to run the connector as a service or in the standalone mode.
9. Complete the installation.

Next Step:

- Configure Advanced Parameters (Optional)
- Run the Connector

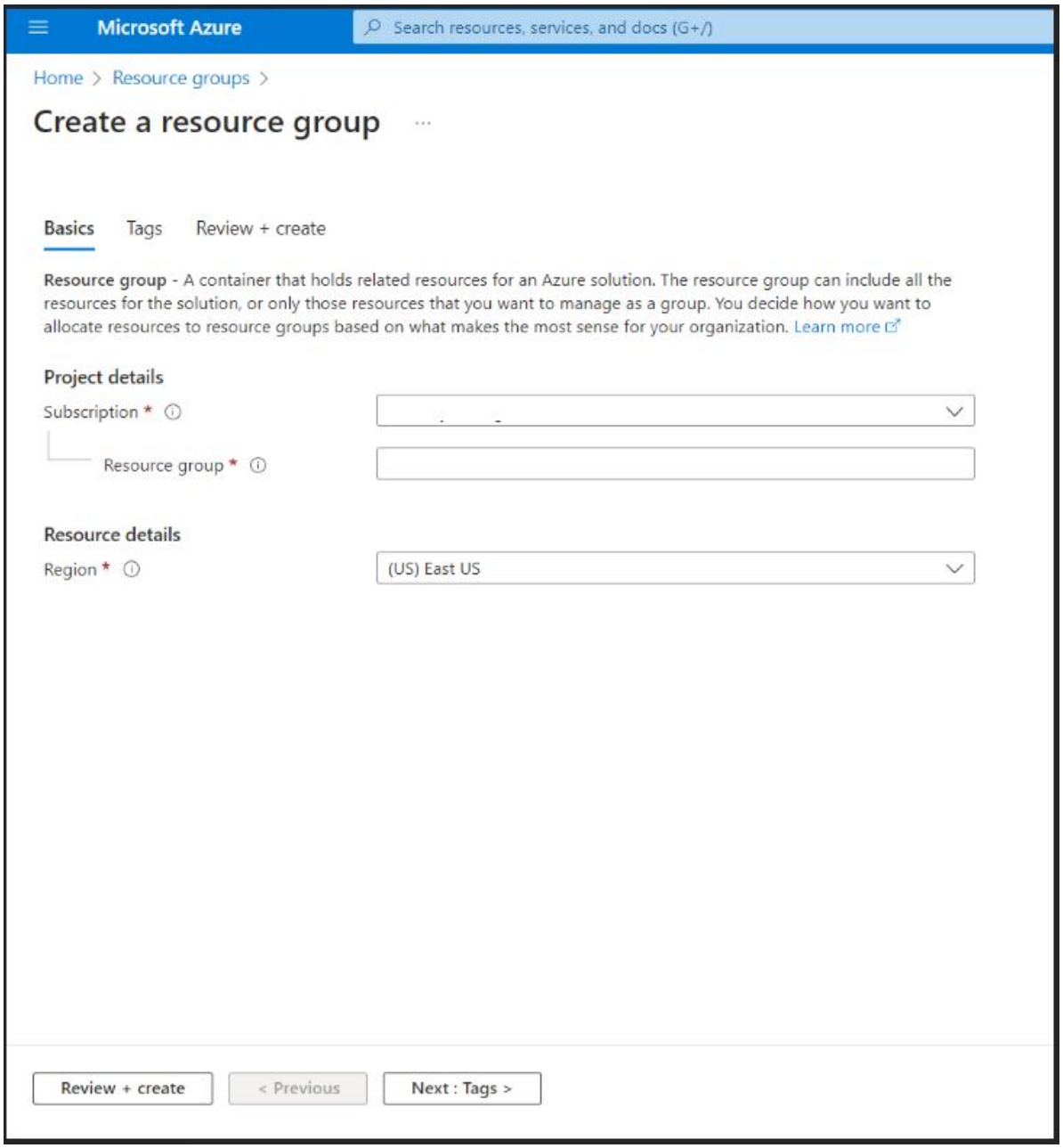
Configuration

The following configuration steps must be implemented before installing the connector:

Creating a Resource Group

1. Log in to the Azure portal and navigate to the **Resource Group** service.
2. Click **+Create**.
3. Select the **Subscription** and enter a name for the group in the **Resource group** field.
4. Navigate to **Resource details** and select the region.
5. (Optional) Click **Next:Tags** to create a tag for the group.

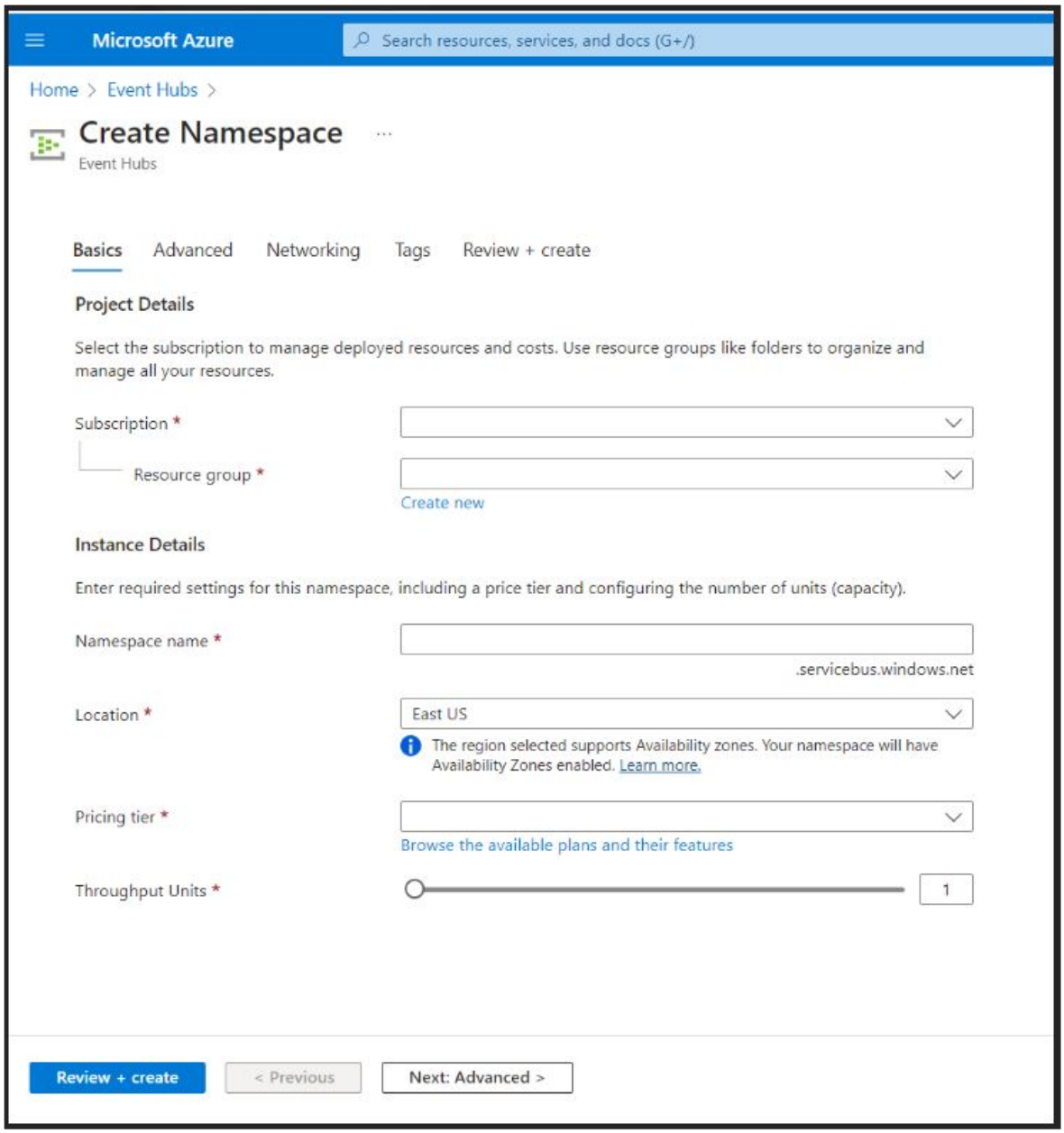
6. Click **Next:Review** > **+Create**.



Creating an Event Hub Namespace

1. Log in to the Azure portal and navigate to the **Event Hubs** service.
2. Click **+Create**.
3. Select the **Subscription** and the **Resource group** that is created above.
4. Navigate to **Instance Details** and enter a name in the **Namespace name** field.
5. Select the same **Location** that is selected while creating the **Resource Group**.

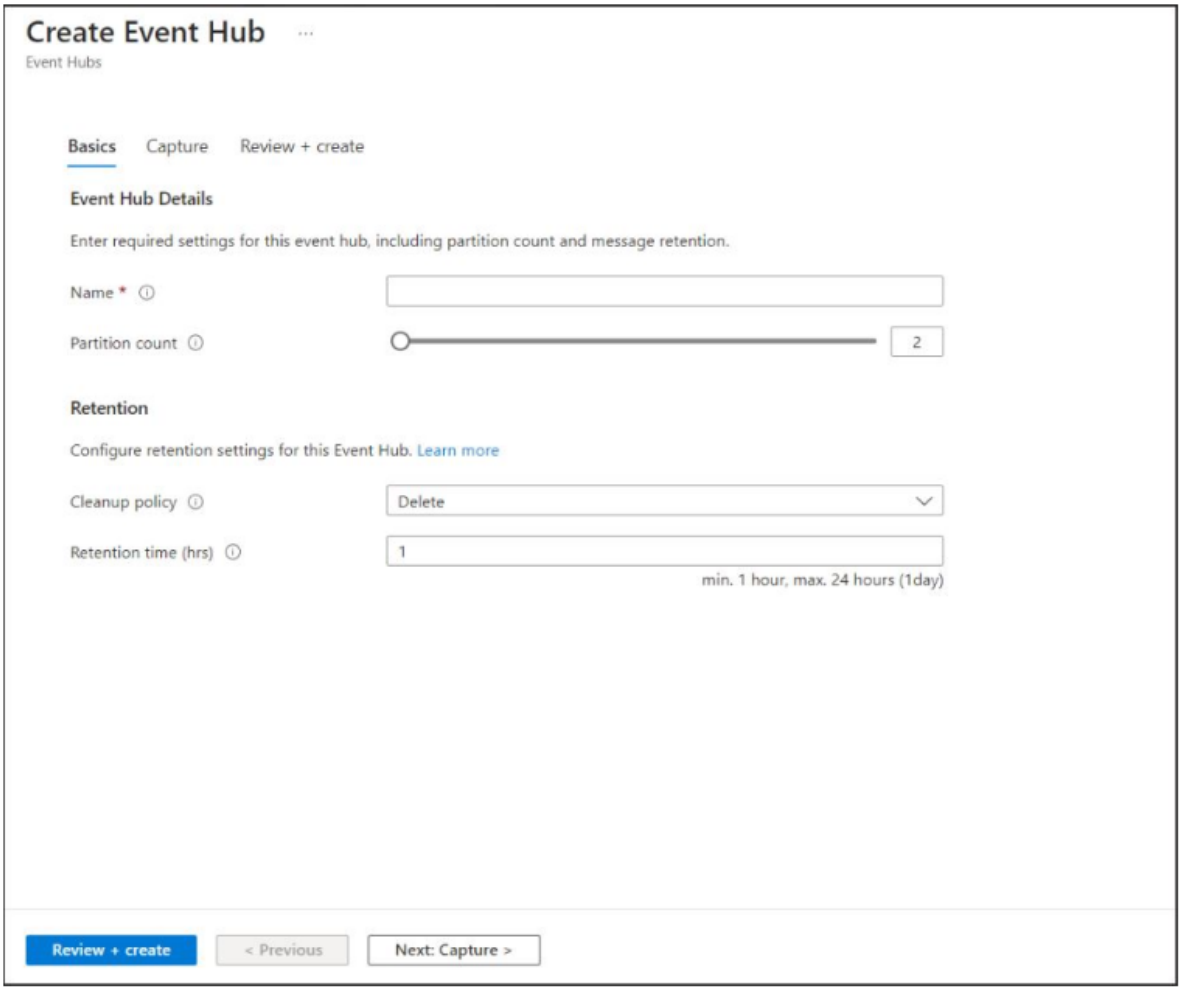
6. Select the **Pricing tier** as **Standard**. Refer to this Microsoft documentation for more plans.
7. Click **Review + Create**.



Creating an Event Hub

1. Click the **Namespace** that is created above.
2. Navigate to **Event Hubs** > **+Event Hub**.
3. Enter a **Name** for the hub and select the **Partition count**. For more information regarding Partition count, refer to this Microsoft Documentation.
4. Navigate to **Retention**. Select the **Cleanup policy** and choose the required **Retention time**.

5. Click **Review + create**.



Note: Ensure to select the same **Subscription** and **Region** while creating the **Resource Group** and **Event Hub Namespace**.

Registering the App

Registering an application in Azure portal is necessary when you want to use Azure services like Azure Active Directory, Azure Monitor, and more. By registering an application, you can create a unique identity for it, and then configure the necessary permissions and settings so that it can interact with the Azure services. To authenticate an application, you can use Azure Active Directory, which provides a secure and scalable way to manage user identities and access to resources.

For registration of the App, the following steps must be implemented:

1. Log in to Azure Portal.
2. Navigate to **Azure Active Directory** and select **App registrations**.
3. Click **+New registration** to create a new application registration.
4. Enter a name for the application, select the appropriate account type and click **Register**.

For authenticating the App, the following steps must be implemented:

To authenticate the application, you can either choose **Client Certificate** or **Client Secret**.

Note: From a security standpoint, **Client Certificates** are often considered to be more secure than **Client Secrets**.

1. If **Client Certificate** is opted to Client Certificate in Azure.

   To generate the self-signed certificate implement the following steps:

   a. Open the command prompt and run the below command by replace the certificate filename and password.

   ```
   keytool -genkey -keystore <filename.pfx> -storetype PKCS12 -keyalg RSA -storepass <password> -validity 730 -keysize 2048
   ```

   This will generate the **.pfx** certificate file. This certificate will be uploaded while installing the connector when **Client Certificate** is used to authenticate the Azure AD Application.

   b. Run the below command to generate **.cer** certificate file by replacing the same filename and password as mentioned in the above step.

   ```
   keytool -export -keystore <filename.pfx> -file <client.cer> -storetype PKCS12 -storepass <password>
   ```

   This will generate the **.cer** certificate file. This must be uploaded in the Azure portal to authenticate the connector to access the Azure AD application. Implement the following steps to upload this certificate to the Azure portal:

   i. Navigate to the Application that is registered in step 3 and click **Certificates & secrets**.

   ii. Under **Certificates** > **Upload Certificate**.

   iii. Select the Public Client Certificate from the drop down and enter a **Description** to it. Then click **Add**.

   Certificate will be listed in **Certificates** section.

2. If **Client Secret** is opted then implement the following steps to configure the Client Secret in Azure:

a. Navigate to the application that is registered in step 3 and click **Certificates & secrets**.

b. Under **Client Secret** > **+New Client Secret**.

c. Enter a **Description** to the secret. Set the value for expiry and click **Add**. This will generate the **Client Secret**.

Important: Note the generated Secret Key Value to provide the same while installing and configuring the connector. If you do not note down the Secret Value, you will not be able to retrieve it later.

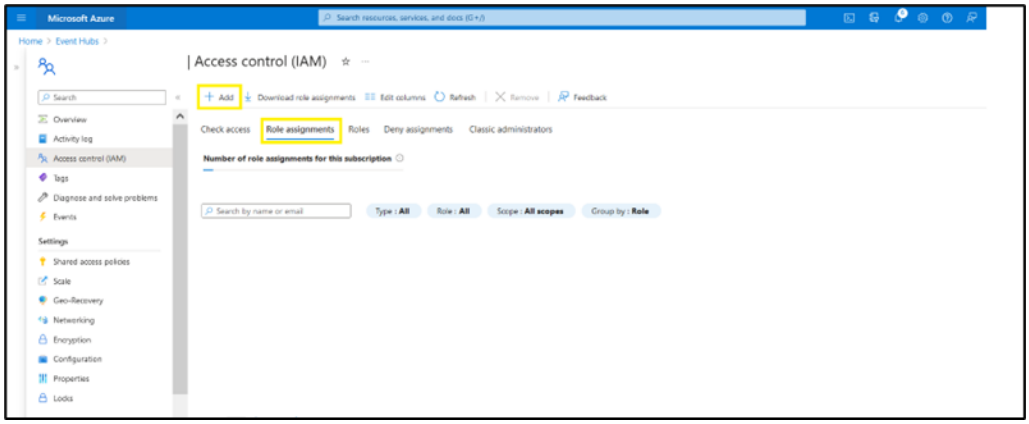
Note: Ensure to note the expiry date of the **Client Secret**and **Client Certificate**. After the Client Secret/ Certificate expires, the connector will fail to authenticate the application and it will stop working. To reconfigure the new Client Secret or Client Certificate, see the Troubleshooting section.

Assigning IAM Role

IAM role must be assigned to this application in Event Hubs Namespace to allow the application to read data from Azure Event Hub.

To assign IAM role:

1. Navigate to **Event Hubs Namespace** created here and select **Access Control (IAM)**.
2. Click **Role Assignments** > **+Add** and select **Add role assignment**.
3. Assign **Azure Event Hubs Data Receiver** role and click **Next**.
4. Click **+Select members** and select the registered application.
5. Click **Next** > **Review + Assign**.



Configuring Advanced Parameters

If you choose to perform any of the operations shown in the following table, do so before adding your connector. After installing core software, you can set the following parameters in the **agent.properties** file, as required:

Parameter	Setting
`bootstrap.servers`	Host-IP.
`group.id`	Use for multiple connectors in a Kafka topic.
`max.poll.records`	The maximum number of records returned in a single call to a poll( ). Default value is 500 (maximum).
`auto.commit.interval.ms`	The frequency in milliseconds in which the consumer offsets are auto-committed to Kafka if the `enable.auto.commit` value is set to **True**: 5000 miliseconds.
`reconnect.backoff.ms`	The base waiting time, before attempting to reconnect to a given host. It avoids repeatedly connecting to a host in a tight loop. This backoff applies to all client connection attempts to a broker: 50 times
`retry.backoff.ms`	The amount of waiting time before attempting to retry a failed request to a given topic partition. It avoids repeatedly sending requests in a tight loop under some failure scenarios: 100 times.
`request.timeout.ms`	It controls the maximum amount of waiting time for a request response. If the response is not received before the timeout elapses, the client resends or fails the request (if the connection attempts have reached the limit: 30000 milliseconds.
`client.id`	An id string to pass to the server when making requests. It tracks the request source beyond just ip/port, by allowing a logical application name to be included on the server-side login request. For tracking: arcsight
`heartbeat.interval.ms`	The expected time between heartbeats to the consumer coordinator when using Kafka's group management facilities. Heartbeats are used to ensure that the consumer's session stays active and facilitates rebalancing when new consumers join or leave the group. The value must be set lower than session.timeout.ms and higher than 1/3 of that value. It can be adjusted even lower to control the expected time for normal rebalances.

Parameter	Setting
`connections.max.idle.ms` (Idle connections timeout)	The server socket processor threads close the connections that appear idle for more than 600000 ms.
`auto.offset.reset`	It can be executed when there is not an initial offset in Kafka or if the current offset does not exist in the server anymore.

`disable.activemq`	The values can be:
	`True: Disable ActiveMQ`
	`False: Enable ActiveMQ`

Running the Connector

Connectors can be installed and run in stand-alone mode, on Windows platforms as a Windows service, or on UNIX platforms as a UNIX daemon, depending upon the platform supported. On Windows platforms, Connectors also can be run using shortcuts and optional **Start** menu entries.

For more information, see Running SmartConnectors.

Publication Status

Released: April 2024

Updated: April 2024

Send Documentation Feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this computer, click the link above and an email window opens with the following information in the subject line:

Feedback on Configuration Guide for FlexConnector for Kafka (SmartConnectors CE 24.2)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to MFI-Documentation-Feedback@opentext.com .

We appreciate your feedback!