Introduction to KafkaLogParser4j Project

The KafkaLogParser4j project is a solution designed to parse, enrich, and consume log data from various sources using Kafka messaging system and .NET technologies. The project comprises 4 individual components, each serving a specific role in the log processing pipeline:

- **KafkaLogParsar4j**: Initiates the Zookeeper server and Kafka server, serving as the backbone of the entire processing pipeline.
- **KafkaLogProducer**: Monitors a designated directory for new log files, extracts log entries, and publishes them to the First-Topic in Kafka.
- **KafkaLogEnricher**: Consumes log entries from the First-Topic, enriches them by extracting data relevant to specific services, and publishes the enriched entries to the Second-Topic in Kafka.
- **KafkaLogConsumer**: Consumes log entries from the Second-Topic, processes them further (e.g., filtering, formatting), and stores the extracted data into a database.

Order of Execution:

- **KafkaLogParsar4j**: Initializes Zookeeper and Kafka servers.
- **KafkaLogProducer**: Monitors and publishes log entries to the First-Log-Parser-Topic.
- **KafkaLogEnricher**: Consumes and enriches log entries from the First-Log-Parser-Topic, publishing them to the Second-Log-Parser-Topic.
- **KafkaLogConsumer**: Consumes enriched log entries from the Second-Log-Parser-Topic and stores them in a database.

Required Software and Packages

To set up and run the KafkaLogParser4j project, you'll need the following:

- **Visual Studio 2022**: The project is developed using .NET technologies, so Visual Studio 2022 provides an integrated development environment for building, debugging, and deploying .NET applications.
- **Kafka 2.12-3.5.1**: Apache Kafka is used as the messaging system for handling log data. You'll need to download and configure Kafka 2.12-3.5.1, including Zookeeper and Kafka server components, to manage the messaging infrastructure.
- **OpenSSL 3.2.1**: OpenSSL is used for secure communication between Kafka clients and brokers. You'll need to install OpenSSL 3.2.1 to generate SSL certificates and configure secure communication settings in Kafka.

NuGet Packages Involved -

KafkaLogParsar4j Project

- **NuGet Packages**:
    - Confluent.Kafka (Version 2.3.0): Provides .NET client for Apache Kafka.
    - Microsoft.Extensions.Hosting.WindowsServices (Version 8.0.0): Enables hosting .NET worker services as Windows services.

KafkaLogProducer Project

- **NuGet Packages**:
    - Confluent.Kafka (Version 2.3.0): Provides .NET client for Apache Kafka.
    - Microsoft.Extensions.Hosting.WindowsServices (Version 8.0.0): Enables hosting .NET worker services as Windows services.

KafkaLogEnricher Project

- **NuGet Packages**:
    - Confluent.Kafka (Version 2.3.0): Provides .NET client for Apache Kafka.
    - Microsoft.Extensions.Hosting.WindowsServices (Version 8.0.0): Enables hosting .NET worker services as Windows services.

KafkaLogConsumer Project

- **NuGet Packages**:
    - Confluent.Kafka (Version 2.3.0): Provides .NET client for Apache Kafka.
    - Microsoft.Extensions.Hosting.WindowsServices (Version 8.0.0): Enables hosting .NET worker services as Windows services.

KafkaClassLibrary Shared Project

- **NuGet Packages**:
    - Microsoft.Data.SqlClient (Version 5.2.0): Provides SQL Server data access.
    - Microsoft.Extensions.Configuration.FileExtensions (Version 8.0.0): Extends configuration file support for .NET applications.
    - Microsoft.Extensions.Hosting (Version 8.0.0): Enables hosting .NET worker services.

Configuration

The **appsettings.json** file contains configuration settings for various components of the KafkaLogParser4j project:

- **Logging Configuration**: Specifies logging levels and settings.
- **Connection Strings**: Defines database connection details.
- **Log Directory Path**: Specifies the directory path where log files are located.
- **Kafka Configuration**: Configures Zookeeper and Kafka server settings, including paths to server scripts, topic names, producer/consumer configurations, and SSL settings.

Ensure that the configuration settings are correctly specified according to your environment and requirements before running the project.

By setting up Visual Studio 2022, Kafka 2.12-3.5.1, and OpenSSL 3.2.1, and configuring the project with the provided **appsettings.json**, you'll be ready to run the KafkaLogParser4j project and process log data efficiently.

Steps to do Apache Kafka SSL Configuration

Steps to set up the Kafka 2.12-3.5.1 installation directory with the SSL folder using OpenSSL:

- **Download Kafka**:
    - Download Apache Kafka version 2.12-3.5.1 from the official website or repository.
- **Extract Kafka Archive**:
    - Extract the downloaded Kafka archive to a desired location, such as **`C:\kafka_2.12-3.5.1`**.
- **Create SSL Folder**:
    - Navigate to the Kafka installation directory (**`C:\kafka_2.12-3.5.1`**) using File Explorer or Command Prompt.
    - Create a new folder named **`ssl`** within the Kafka installation directory.
- **Download and Install OpenSSL**:
    - Download OpenSSL from the official website or repository if you haven't already installed it.
    - Run the OpenSSL installer and follow the installation instructions.
    - Open OpenSSL from Start Menu
- **Navigate to OpenSSL Installation Directory**:
    - In Win64 OpenSSL Command Prompt, Use the **`cd`** command to navigate to the directory where OpenSSL is installed. For example: cd C:\Program Files\OpenSSL-Win64

Now you have successfully extracted Kafka, created the SSL folder, installed OpenSSL, and navigated to the SSL folder within the Kafka installation directory. Next, we need execute commands over Win64 OpenSSL Command Prompt for implementing CA, truststore and keystore for Kafka SSL Configuration

Steps to implement CA, truststore and keystore for Kafka SSL Configuration

Password to be used (Any): BXygHIlBap8RDt9AmxeZBaakF4yASz0C
Execute following commands over Win64 OpenSSL Command Prompt over specified location

Secure SSL/TLS Setup for Kafka Zookeeper

1. Generate CA
```
openssl req -new -x509 -keyout ca-key -out ca-cert -days 365
```
2. Create Truststore
```
keytool -keystore kafka.zookeeper.truststore.jks -alias ca-cert -importcert -file ca-cert
```
3. Create Keystore
```
keytool -keystore kafka.zookeeper.keystore.jks -alias zookeeper -validity 365 -genkey -keyalg RSA -ext SAN=dns:localhost
```
4. Create certificate signing request (CSR)
```
keytool -keystore kafka.zookeeper.keystore.jks -alias zookeeper -certreq -file ca-request-zookeeper
```
5. Sign the CSR
```
openssl x509 -req -CA ca-cert -CAkey ca-key -in ca-request-zookeeper -out ca-signed-zookeeper -days 365 -CAcreateserial
```
6. Import the CA into Keystore
```
keytool -keystore kafka.zookeeper.keystore.jks -alias ca-cert -importcert -file ca-cert
```
7. Import the signed certificate into Keystore
```
keytool -keystore kafka.zookeeper.keystore.jks -alias zookeeper -importcert -file ca-signed-zookeeper
```

Configure Kafka Zookeeper Client SSL/TLS

```
keytool -keystore kafka.zookeeper-client.truststore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.zookeeper-client.keystore.jks -alias zookeeper-client -validity 365 -genkey -keyalg RSA -ext SAN=dns:localhost
keytool -keystore kafka.zookeeper-client.keystore.jks -alias zookeeper-client -certreq -file ca-request-zookeeper-client
openssl x509 -req -CA ca-cert -CAkey ca-key -in ca-request-zookeeper-client -out ca-signed-zookeeper-client -days 365 -CAcreateserial
keytool -keystore kafka.zookeeper-client.keystore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.zookeeper-client.keystore.jks -alias zookeeper-client -importcert -file ca-signed-zookeeper-client
```

Configure Kafka Broker 0 SSL/TLS

```
keytool -keystore kafka.broker0.truststore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.broker0.keystore.jks -alias broker0 -validity 365 -genkey -keyalg RSA -ext SAN=dns:localhost
keytool -keystore kafka.broker0.keystore.jks -alias broker0 -certreq -file ca-request-broker0
openssl x509 -req -CA ca-cert -CAkey ca-key -in ca-request-broker0 -out ca-signed-broker0 -days 365 -CAcreateserial
keytool -keystore kafka.broker0.keystore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.broker0.keystore.jks -alias broker0 -importcert -file ca-signed-broker0
```

Configure Kafka Broker 1 SSL/TLS

```
keytool -keystore kafka.broker1.truststore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.broker1.keystore.jks -alias broker1 -validity 365 -genkey -keyalg RSA -
ext SAN=dns:localhost
keytool -keystore kafka.broker1.keystore.jks -alias broker1 -certreq -file ca-request-broker1
openssl x509 -req -CA ca-cert -CAkey ca-key -in ca-request-broker1 -out ca-signed-broker1 -days
365 -CAcreateserial
keytool -keystore kafka.broker1.keystore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.broker1.keystore.jks -alias broker1 -importcert -file ca-signed-broker1
```

Configure Kafka Broker 2 SSL/TLS

```
keytool -keystore kafka.broker2.truststore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.broker2.keystore.jks -alias broker2 -validity 365 -genkey -keyalg RSA -
ext SAN=dns:localhost
keytool -keystore kafka.broker2.keystore.jks -alias broker2 -certreq -file ca-request-broker2
openssl x509 -req -CA ca-cert -CAkey ca-key -in ca-request-broker2 -out ca-signed-broker2 -days
365 -CAcreateserial
keytool -keystore kafka.broker2.keystore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.broker2.keystore.jks -alias broker2 -importcert -file ca-signed-broker2
```

Configure Kafka Admin SSL/TLS

```
keytool -keystore kafka.admin.truststore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.admin.keystore.jks -alias admin -validity 365 -genkey -keyalg RSA -ext
SAN=dns:localhost
keytool -keystore kafka.admin.keystore.jks -alias admin -certreq -file ca-request-admin
openssl x509 -req -CA ca-cert -CAkey ca-key -in ca-request-admin -out ca-signed-admin -days 365
-CAcreateserial
keytool -keystore kafka.admin.keystore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.admin.keystore.jks -alias admin -importcert -file ca-signed-admin
```

Configure Kafka Producer SSL/TLS

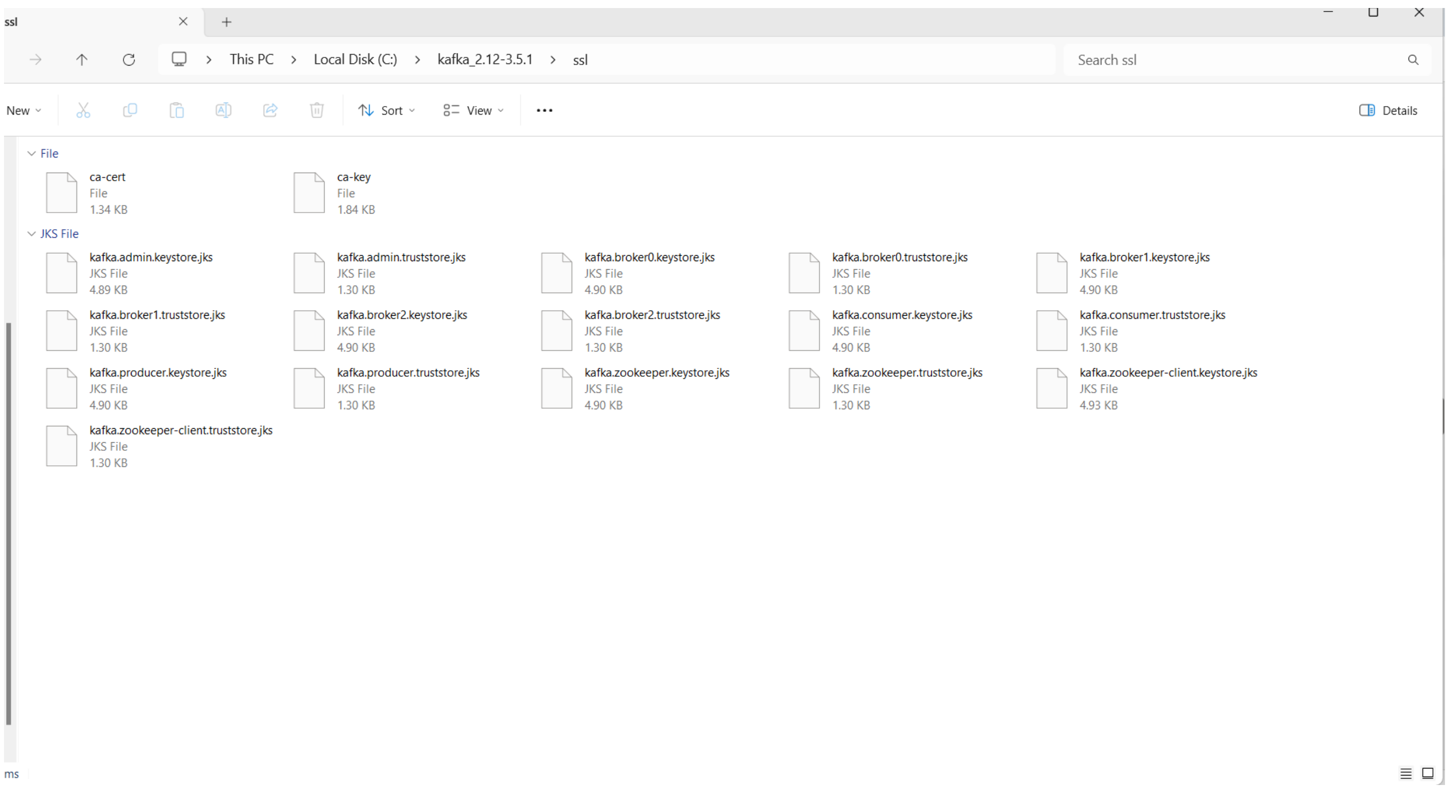
```
keytool -keystore kafka.producer.truststore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.producer.keystore.jks -alias producer -validity 365 -genkey -keyalg RSA
-ext SAN=dns:localhost
keytool -keystore kafka.producer.keystore.jks -alias producer -certreq -file ca-request-
producer
openssl x509 -req -CA ca-cert -CAkey ca-key -in ca-request-producer -out ca-signed-producer -
days 365 -CAcreateserial
keytool -keystore kafka.producer.keystore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.producer.keystore.jks -alias producer -importcert -file ca-signed-
producer
```

Configure Kafka Consumer SSL/TLS

```
keytool -keystore kafka.consumer.truststore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.consumer.keystore.jks -alias consumer -validity 365 -genkey -keyalg RSA
-ext SAN=dns:localhost
keytool -keystore kafka.consumer.keystore.jks -alias consumer -certreq -file ca-request-
consumer
openssl x509 -req -CA ca-cert -CAkey ca-key -in ca-request-consumer -out ca-signed-consumer -
days 365 -CAcreateserial
keytool -keystore kafka.consumer.keystore.jks -alias ca-cert -importcert -file ca-cert
keytool -keystore kafka.consumer.keystore.jks -alias consumer -importcert -file ca-signed-
consumer
```

After Execution of above commands, following would be files present over SSL Path.

Note: Please remove files generated other than these.



File

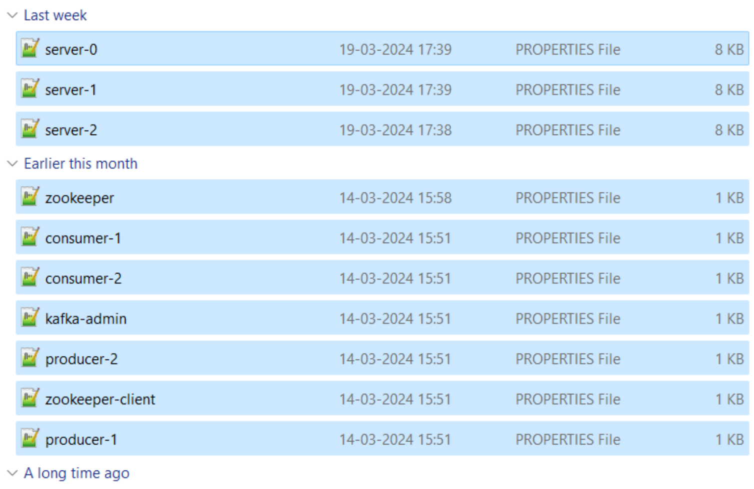
- ca-cert — File — 1.34 KB
- ca-key — File — 1.84 KB

JKS File

- kafka.admin.keystore.jks — JKS File — 4.89 KB
- kafka.admin.truststore.jks — JKS File — 1.30 KB
- kafka.broker0.keystore.jks — JKS File — 4.90 KB
- kafka.broker0.truststore.jks — JKS File — 1.30 KB
- kafka.broker1.keystore.jks — JKS File — 4.90 KB
- kafka.broker1.truststore.jks — JKS File — 1.30 KB
- kafka.broker2.keystore.jks — JKS File — 4.90 KB
- kafka.broker2.truststore.jks — JKS File — 1.30 KB
- kafka.consumer.keystore.jks — JKS File — 4.90 KB
- kafka.consumer.truststore.jks — JKS File — 1.30 KB
- kafka.producer.keystore.jks — JKS File — 4.90 KB
- kafka.producer.truststore.jks — JKS File — 1.30 KB
- kafka.zookeeper.keystore.jks — JKS File — 4.90 KB
- kafka.zookeeper.truststore.jks — JKS File — 1.30 KB
- kafka.zookeeper-client.keystore.jks — JKS File — 4.93 KB
- kafka.zookeeper-client.truststore.jks — JKS File — 1.30 KB

After this please update/add files over path: **C:\kafka_2.12-3.5.1\config** for using above SSL/TLS Configurations

Upload all the files here :: Web Version Does not Supporting



Last week

server-0	19-03-2024 17:39	PROPERTIES File	8 KB
server-1	19-03-2024 17:39	PROPERTIES File	8 KB
server-2	19-03-2024 17:38	PROPERTIES File	8 KB

Earlier this month

zookeeper	14-03-2024 15:58	PROPERTIES File	1 KB
consumer-1	14-03-2024 15:51	PROPERTIES File	1 KB
consumer-2	14-03-2024 15:51	PROPERTIES File	1 KB
kafka-admin	14-03-2024 15:51	PROPERTIES File	1 KB
producer-2	14-03-2024 15:51	PROPERTIES File	1 KB
zookeeper-client	14-03-2024 15:51	PROPERTIES File	1 KB
producer-1	14-03-2024 15:51	PROPERTIES File	1 KB

A long time ago

CMD command for running over windows machine (Anywhere on CMD)

For starting Zookeeper Server and Kafka Clients as 3 brokers and One Zookeeper Shell, execute following commands

Start Zookeeper (Run on new CMD)-

```
C:\kafka_2.12-3.5.1\bin\windows\zookeeper-server-start.bat C:\kafka_2.12-
3.5.1\config\zookeeper.properties
```

To create a SASL/SCRAM user (Run on new CMD) -

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-configs.bat --zookeeper localhost:2182 --zk-tls-config-
file C:/kafka_2.12-3.5.1/config/zookeeper-client.properties --entity-type users --entity-name
broker-admin --alter --add-config 'SCRAM-SHA-512=[password=BXygHIlBap8RDt9AmxeZBaakF4yASz0C]'
```

To describe/list all available SASL/SCRAM users -

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-configs.bat --zookeeper localhost:2182 --zk-tls-config-
file C:/kafka_2.12-3.5.1/config/zookeeper-client.properties --entity-type users --describe
```

Start 3 Brokers (Run on new 3 CMDs) -

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-server-start.bat C:\kafka_2.12-3.5.1\config\server-
0.properties
C:\kafka_2.12-3.5.1\bin\windows\kafka-server-start.bat C:\kafka_2.12-3.5.1\config\server-
1.properties
C:\kafka_2.12-3.5.1\bin\windows\kafka-server-start.bat C:\kafka_2.12-3.5.1\config\server-
2.properties
```

For Starting Zookeeper Shell

```
C:\kafka_2.12-3.5.1\bin\windows\zookeeper-shell.bat localhost:2182 -zk-tls-config-file
C:\kafka_2.12-3.5.1\config\zookeeper-client.properties
```

--> ls /brokers/ids

For Creating kafka-admin user (SASL credential) (Run on new CMD and all the remaining Commands)

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-configs.bat --zookeeper localhost:2182 --zk-tls-config-
file C:\kafka_2.12-3.5.1\config\zookeeper-client.properties --entity-type users --entity-name
kafka-admin --alter --add-config 'SCRAM-SHA-512=[password=BXygHIlBap8RDt9AmxeZBaakF4yASz0C]'
```

<u>For granting the super-user access (kafka-admin user), execute following commands</u>

FULL ACCESS for Topics

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-acls.bat --authorizer-properties
zookeeper.connect=localhost:2182 --zk-tls-config-file C:\kafka_2.12-3.5.1\config\zookeeper-
client.properties --add --allow-principal User:kafka-admin --operation READ --operation WRITE -
-operation DESCRIBE --operation DESCRIBECONFIGS --operation ALTER --operation ALTERCONFIGS --
operation CREATE --operation DELETE --topic '*'
```

FULL ACCESS for Groups

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-acls.bat --authorizer-properties
zookeeper.connect=localhost:2182 --zk-tls-config-file C:\kafka_2.12-3.5.1\config\zookeeper-
client.properties --add --allow-principal User:kafka-admin --operation READ --operation
DESCRIBE --operation DELETE --group '*'
```

FULL ACCESS for delegation-tokens

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-acls.bat --authorizer-properties
zookeeper.connect=localhost:2182 --zk-tls-config-file C:\kafka_2.12-3.5.1\config\zookeeper-
client.properties --add --allow-principal User:kafka-admin --operation DESCRIBE --delegation-
token '*'
```

FULL ACCESS for transactional clients

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-acls.bat --authorizer-properties
zookeeper.connect=localhost:2182 --zk-tls-config-file C:\kafka_2.12-3.5.1\config\zookeeper-
client.properties --add --allow-principal User:kafka-admin --operation DESCRIBE --operation
WRITE --transactional-id '*'
```

FULL ACCESS to the cluster

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-acls.bat --authorizer-properties
zookeeper.connect=localhost:2182 --zk-tls-config-file C:\kafka_2.12-3.5.1\config\zookeeper-
client.properties --add --allow-principal User:kafka-admin --operation ALTER --operation
ALTERCONFIGS --operation CLUSTERACTION --operation CREATE --operation DESCRIBE --operation
DESCRIBECONFIGS --operation IDEMPOTENTWRITE --cluster m0oxOd58QzKfBA-1MvnvRw
```

Note - If you want to know how to find the ID of your cluster, connect to zookeeper shell and execute *get /cluster/id*.

To list all the ACLs associated with the user

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-acls.bat --authorizer-properties
zookeeper.connect=localhost:2182 --zk-tls-config-file C:\kafka_2.12-3.5.1\config\zookeeper-
client.properties --list --principal User:kafka-admin
```

For creating First Topic and Second Topic

Create First topic

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-topics.bat --bootstrap-server
localhost:9092,localhost:9093,localhost:9094 --command-config C:/kafka_2.12-3.5.1/config/kafka-
admin.properties --create --topic First-Log-Parser-Topic --partitions 1 --replication-factor 2
--config min.insync.replicas=2
```

Create Second topic

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-topics.bat --bootstrap-server
localhost:9092,localhost:9093,localhost:9094 --command-config C:/kafka_2.12-3.5.1/config/kafka-
admin.properties --create --topic Second-Log-Parser-Topic --partitions 1 --replication-factor 2
--config min.insync.replicas=2
```

Listing the available topics inside the Kafka cluster

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-topics.bat --bootstrap-server
localhost:9092,localhost:9093,localhost:9094 --command-config C:/kafka_2.12-3.5.1/config/kafka-
admin.properties --list
```

Describe a specific topic

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-topics.bat --bootstrap-server
localhost:9092,localhost:9093,localhost:9094 --command-config C:/kafka_2.12-3.5.1/config/kafka-
admin.properties --describe --topic First-Log-Parser-Topic
```

For creating Producer Consumer Set for First and Second Topic Respectively

Create First Producer

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-configs.bat --zookeeper localhost:2182 --zk-tls-config-
file C:/kafka_2.12-3.5.1/config/zookeeper-client.properties --entity-type users --entity-name
First-Topic-Sasl-Producer --alter --add-config 'SCRAM-SHA-
512=[password=BXygHIlBap8RDt9AmxeZBaakF4yASz0C]'
```

Create First Consumer

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-configs.bat --zookeeper localhost:2182 --zk-tls-config-
file C:/kafka_2.12-3.5.1/config/zookeeper-client.properties --entity-type users --entity-name
First-Topic-Sasl-Consumer --alter --add-config 'SCRAM-SHA-
512=[password=BXygHIlBap8RDt9AmxeZBaakF4yASz0C]'
```

Create Second Producer

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-configs.bat --zookeeper localhost:2182 --zk-tls-config-
file C:/kafka_2.12-3.5.1/config/zookeeper-client.properties --entity-type users --entity-name
Second-Topic-Sasl-Producer --alter --add-config 'SCRAM-SHA-
512=[password=BXygHIlBap8RDt9AmxeZBaakF4yASz0C]'
```

Create Second Consumer

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-configs.bat --zookeeper localhost:2182 --zk-tls-config-
file C:/kafka_2.12-3.5.1/config/zookeeper-client.properties --entity-type users --entity-name
Second-Topic-Sasl-Consumer --alter --add-config 'SCRAM-SHA-
512=[password=BXygHIlBap8RDt9AmxeZBaakF4yASz0C]'
```

ACL commands for operations on specific resources/groups of resources

To grant First Producer access for the user to the topic *First-Log-Parser-Topic*

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-acls.bat --authorizer-properties
zookeeper.connect=localhost:2182 --zk-tls-config-file C:\kafka_2.12-3.5.1\config\zookeeper-
client.properties --add --allow-principal User:First-Topic-Sasl-Producer --topic First-Log-
Parser-Topic --operation DESCRIBE --operation DESCRIBECONFIGS --operation WRITE
```

To grant Second Producer access for the user to the topic *Second-Log-Parser-Topic*

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-acls.bat --authorizer-properties
zookeeper.connect=localhost:2182 --zk-tls-config-file C:\kafka_2.12-3.5.1\config\zookeeper-
client.properties --add --allow-principal User:Second-Topic-Sasl-Producer --topic Second-Log-
Parser-Topic --operation DESCRIBE --operation DESCRIBECONFIGS --operation WRITE
```

To grant First Consumer access for the user to the topic *First-Log-Parser-Topic* and with consumer group *First-Group-Log-Parser*

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-acls.bat --authorizer-properties
zookeeper.connect=localhost:2182 --zk-tls-config-file C:\kafka_2.12-3.5.1\config\zookeeper-
client.properties --add --allow-principal User:First-Topic-Sasl-Consumer --operation READ --
operation DESCRIBE --topic First-Log-Parser-Topic

C:\kafka_2.12-3.5.1\bin\windows\kafka-acls.bat --authorizer-properties
zookeeper.connect=localhost:2182 --zk-tls-config-file C:\kafka_2.12-3.5.1\config\zookeeper-
client.properties --add --allow-principal User:First-Topic-Sasl-Consumer --group First-Group-
Log-Parser --operation READ
```

To grant First Consumer access for the user to the topic *Second-Log-Parser-Topic* and with consumer group *Second-Group-Log-Parser*

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-acls.bat --authorizer-properties
zookeeper.connect=localhost:2182 --zk-tls-config-file C:\kafka_2.12-3.5.1\config\zookeeper-
client.properties --add --allow-principal User:Second-Topic-Sasl-Consumer --operation READ --
operation DESCRIBE --topic Second-Log-Parser-Topic

C:\kafka_2.12-3.5.1\bin\windows\kafka-acls.bat --authorizer-properties
zookeeper.connect=localhost:2182 --zk-tls-config-file C:\kafka_2.12-3.5.1\config\zookeeper-
client.properties --add --allow-principal User:Second-Topic-Sasl-Consumer --group Second-Group-
Log-Parser --operation READ
```

Commands to produce and consume to/from a topic (If required to check)

To Produce messages over First Producer

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-console-producer.bat --topic First-Log-Parser-Topic --
producer.config C:\kafka_2.12-3.5.1\config\producer-1.properties --broker-list
localhost:9092,localhost:9093
```

To Consume messages from First Consumer

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-console-consumer.bat --topic First-Log-Parser-Topic --
from-beginning --consumer.config C:\kafka_2.12-3.5.1\config\consumer-1.properties --bootstrap-
server localhost:9092,localhost:9093
```

To Produce messages over Second Producer

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-console-producer.bat --topic Second-Log-Parser-Topic --
producer.config C:\kafka_2.12-3.5.1\config\producer-2.properties --broker-list
localhost:9092,localhost:9093
```

To Consume messages from Second Consumer

```
C:\kafka_2.12-3.5.1\bin\windows\kafka-console-consumer.bat --topic Second-Log-Parser-Topic --
from-beginning --consumer.config C:\kafka_2.12-3.5.1\config\consumer-2.properties --bootstrap-
server localhost:9092,localhost:9093
```

Creation of Windows Background Service

To create the .NET Worker Service app as a Windows Service, it's recommended that you publish the app as a single file executable, so after having published Application, we are supposed to use those paths for the .exe file for the projects.

We will create the Windows Service, using the native Windows Service Control Manager's (sc.exe) create command. Run CMD/PowerShell as an Administrator, execute following commands.

Create the Windows Service

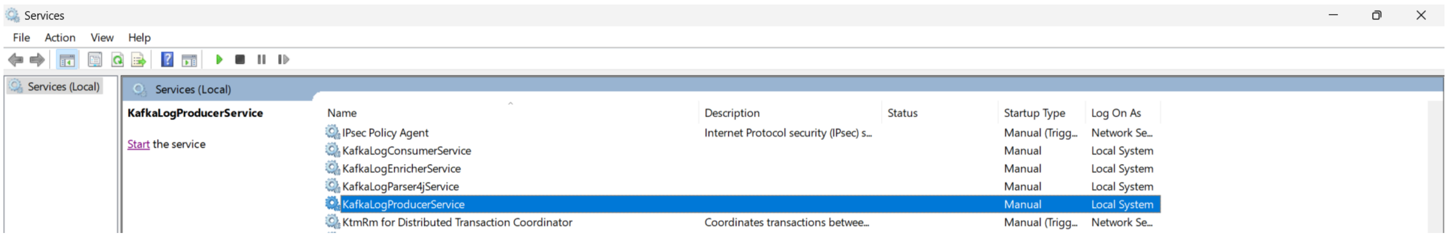
```
sc.exe create KafkaLogParser4jService
binPath="c:\users\mayankcoinstation\source\repos\kafkalogparser4j\KafkaLogParser4j\bin\release\
net8.0\publish\KafkaLogParser4j.exe"
```

```
sc.exe create KafkaLogProducerService
binPath="c:\users\mayankcoinstation\source\repos\kafkalogparser4j\kafkalogproducer\bin\release\
net8.0\publish\KafkaLogProducer.exe"
```

```
sc.exe create KafkaLogEnricherService
binPath="c:\users\mayankcoinstation\source\repos\kafkalogparser4j\kafkalogenricher\bin\release\
net8.0\publish\KafkaLogEnricher.exe"
```

```
sc.exe create KafkaLogConsumerService
binPath="c:\users\mayankcoinstation\source\repos\kafkalogparser4j\kafkalogconsumer\bin\release\
net8.0\publish\KafkaLogConsumer.exe"
```

After successful Services creation, we can see the services over Windows Services as below



KafkaLogProducerService

Start the service

Name	Description	Status	Startup Type	Log On As
IPsec Policy Agent	Internet Protocol security (IPsec) s...		Manual (Trigg...	Network Se...
KafkaLogConsumerService			Manual	Local System
KafkaLogEnricherService			Manual	Local System
KafkaLogParser4jService			Manual	Local System
KafkaLogProducerService			Manual	Local System
KtmRm for Distributed Transaction Coordinator	Coordinates transactions betwee...		Manual (Trigg...	Network Se...

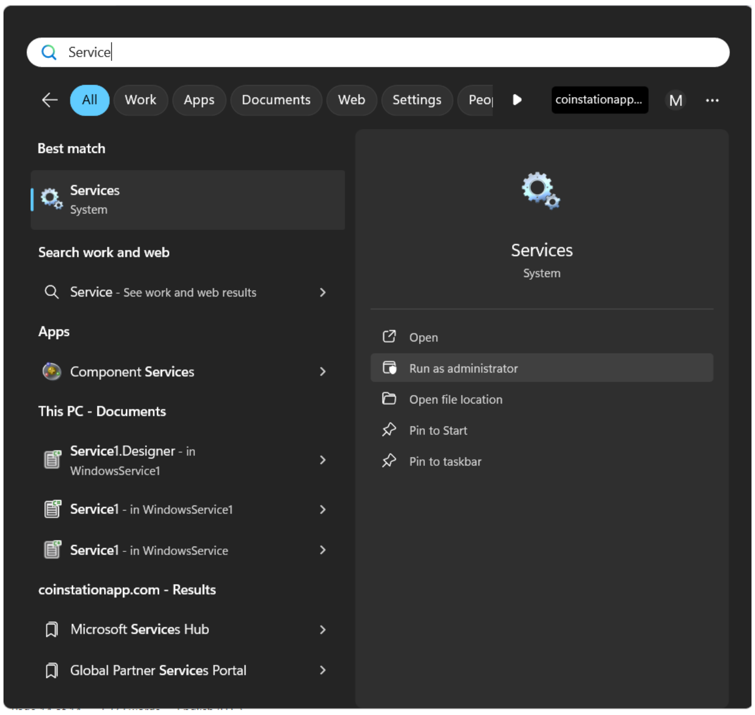
Configure the Windows Service

After the service is created, you can optionally configure it. Windows Services provide recovery configuration options. You can query the current configuration using the sc.exe qfailure "<Service Name>" (where <Service Name> is your services' name) command to read the current recovery configuration values :

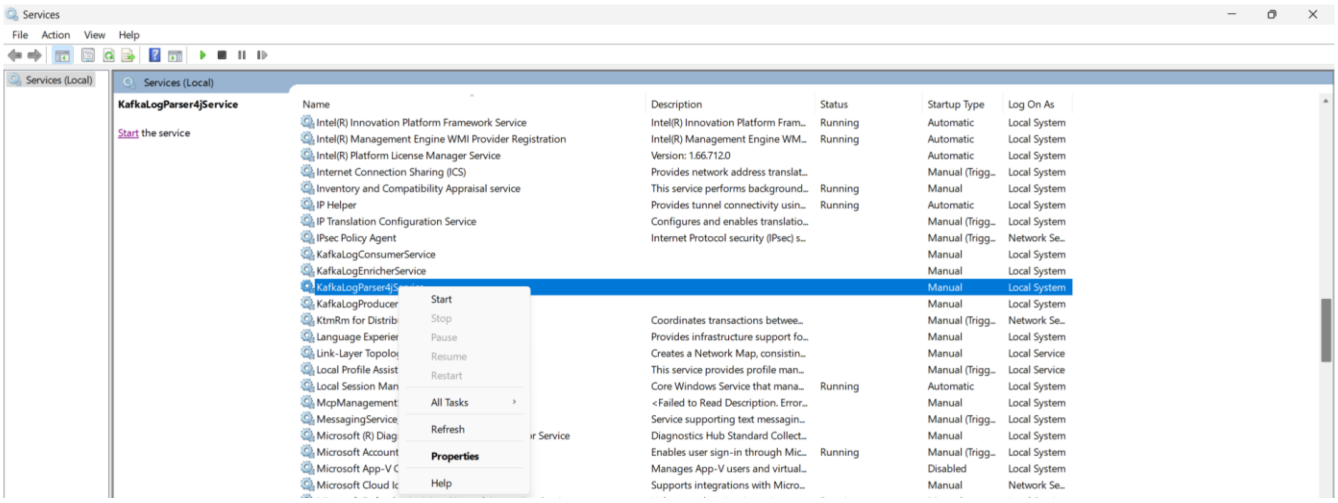
```
sc qfailure " KafkaLogParser4jService"
```

To configure the recovery options for all the 4 Services, proceed with following Steps for each of the 4 Services

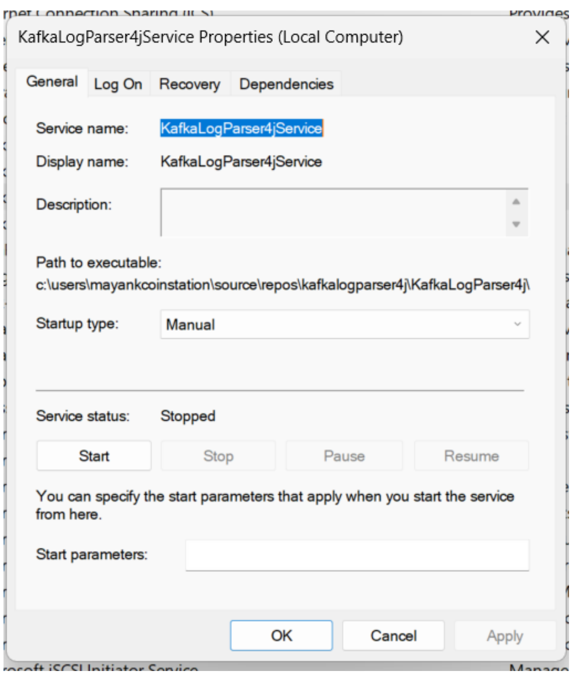
1. Open Service from Windows Startup in Administrator Mode



2. Navigate to the Service and Right Click, Open Properties window



Name	Description	Status	Startup Type	Log On As
Intel(R) Innovation Platform Framework Service	Intel(R) Innovation Platform Fram...	Running	Automatic	Local System
Intel(R) Management Engine WMI Provider Registration	Intel(R) Management Engine WM...	Running	Automatic	Local System
Intel(R) Platform License Manager Service	Version: 1.66.712.0		Automatic	Local System
Internet Connection Sharing (ICS)	Provides network address translat...		Manual (Trigg...	Local System
Inventory and Compatibility Appraisal service	This service performs background...	Running	Manual	Local System
IP Helper	Provides tunnel connectivity usin...	Running	Automatic	Local System
IP Translation Configuration Service	Configures and enables translatio...		Manual (Trigg...	Local System
IPsec Policy Agent	Internet Protocol security (IPsec) s...		Manual (Trigg...	Network Se...
KafkaLogConsumerService			Manual	Local System
KafkaLogEnricherService			Manual	Local System
KafkaLogParser4jS			Manual	Local System
KafkaLogProducer			Manual	Local System
KtmRm for Distrib	Coordinates transactions betwee...		Manual (Trigg...	Network Se...
Language Experien	Provides infrastructure support fo...		Manual	Local System
Link-Layer Topolo	Creates a Network Map, consistin...		Manual	Local Service
Local Profile Assist	This service provides profile man...		Manual (Trigg...	Local Service
Local Session Man	Core Windows Service that mana...	Running	Automatic	Local System
McpManagement	<Failed to Read Description. Error...		Manual	Local System
MessagingService	Service supporting text messagin...		Manual (Trigg...	Local System
Microsoft (R) Diag ... r Service	Diagnostics Hub Standard Collect...		Manual	Local System
Microsoft Account	Enables user sign-in through Mic...	Running	Manual (Trigg...	Local System
Microsoft App-V C	Manages App-V users and virtual...		Disabled	Local System
Microsoft Cloud Ic	Supports integrations with Micro...		Manual	Network Se...


KafkaLogParser4jService Properties (Local Computer)

Service name: KafkaLogParser4jService

Display name: KafkaLogParser4jService

Description:

Path to executable:
c:\users\mayankcoinstation\source\repos\kafkalogparser4j\KafkaLogParser4j\

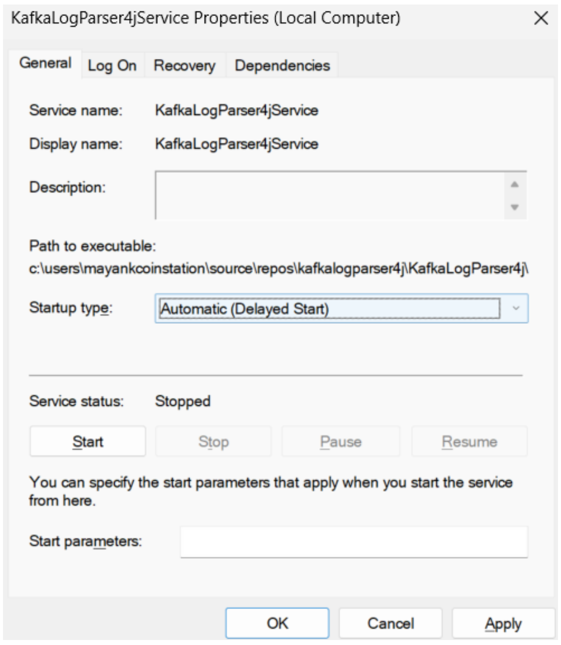
Startup type: Manual

Service status: Stopped

You can specify the start parameters that apply when you start the service from here.

Start parameters:

3. On **General** Tab, Change **Startup type** from **Manual** to **Automatic (Delayed)** which makes service to start a short while after the system has finished starting up


KafkaLogParser4jService Properties (Local Computer)

Service name: KafkaLogParser4jService

Display name: KafkaLogParser4jService

Description:

Path to executable:
c:\users\mayankcoinstation\source\repos\kafkalogparser4j\KafkaLogParser4j\

Startup type: Automatic (Delayed Start)

Service status: Stopped

You can specify the start parameters that apply when you start the service from here.

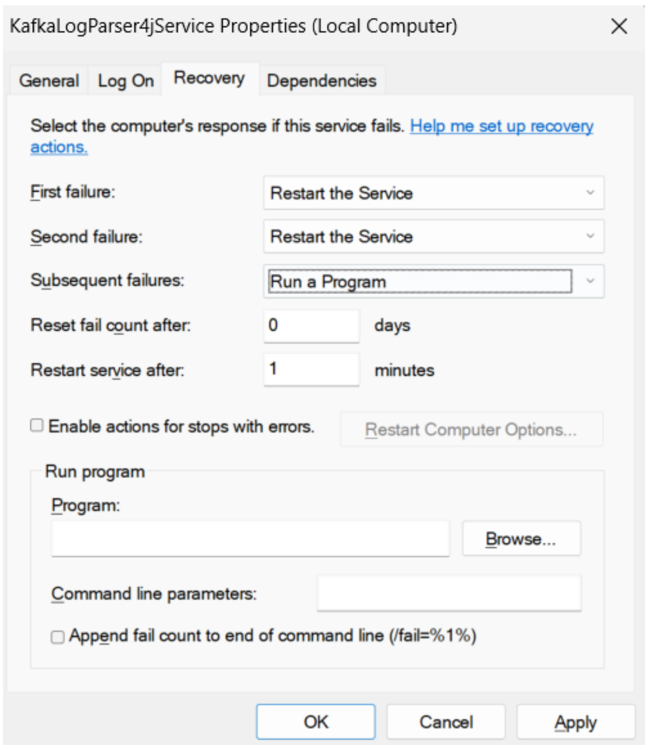
Start parameters:

4. Navigate to **Recovery** Tab under this Service Properties window and Change following and further click on **Apply** Button and **OK** Button to exit from Properties window

   First failure : Restart the Service

   Second failure: Restart the Service

Subsequent failure: Run a program


KafkaLogParser4jService Properties (Local Computer)

Select the computer's response if this service fails. Help me set up recovery actions.

First failure: Restart the Service

Second failure: Restart the Service

Subsequent failures: Run a Program

Reset fail count after: 0 days

Restart service after: 1 minutes

☐ Enable actions for stops with errors.

Run program

Program:

Command line parameters:

☐ Append fail count to end of command line (/fail=%1%)

5. Open the Properties window again and verify the changed values

Commands for Start the Services:

```
sc.exe start KafkaLogParser4jService
sc.exe start KafkaLogProducerService
sc.exe start KafkaLogEnricherService
sc.exe start KafkaLogConsumerService
```

Commands for Stop the Services (if required to execute else ignore to execute):

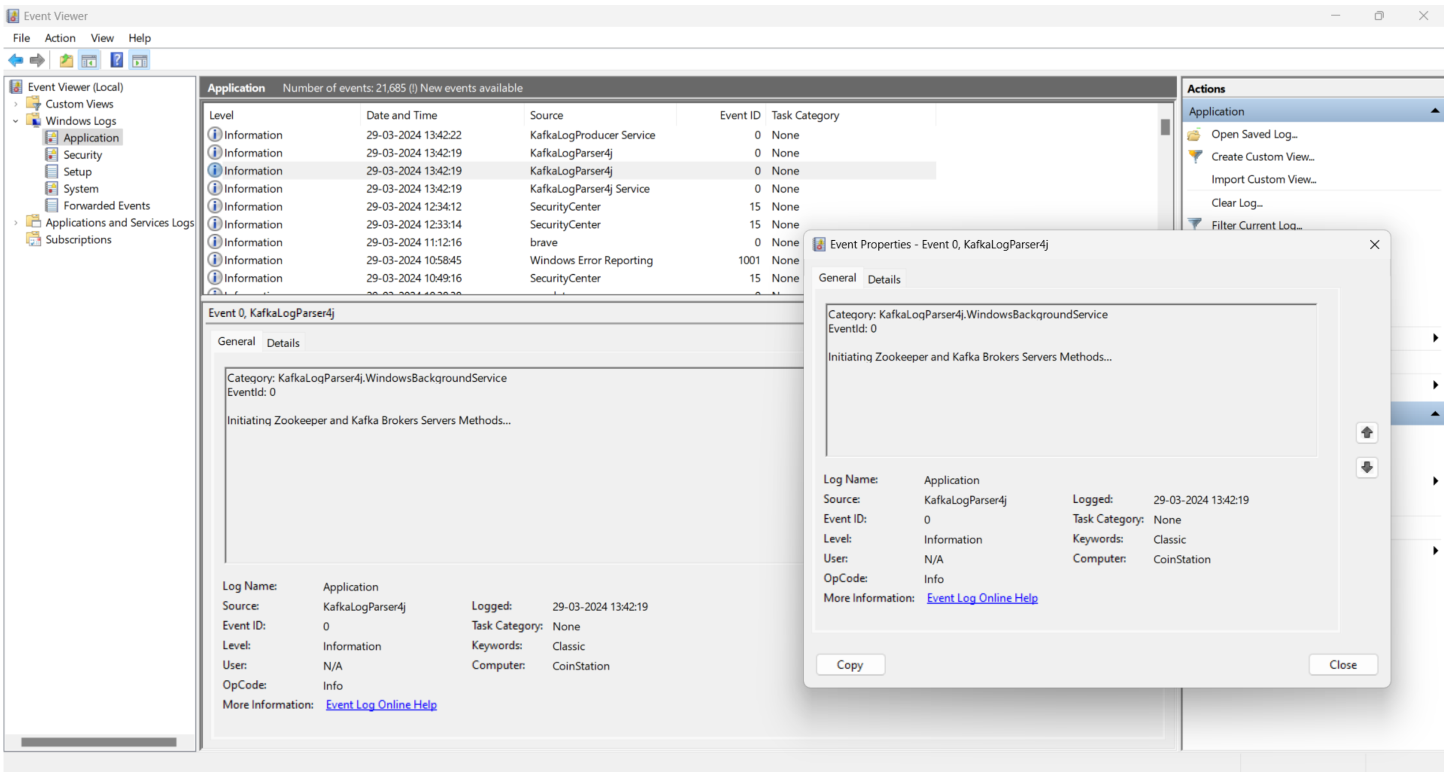
```
sc.exe stop KafkaLogParser4jService
sc.exe stop KafkaLogProducerService
sc.exe stop KafkaLogEnricherService
sc.exe stop KafkaLogConsumerService
```

Commands for Delete the Services (if required to execute else ignore to execute):

```
sc.exe delete KafkaLogParser4jService
sc.exe delete KafkaLogProducerService
sc.exe delete KafkaLogEnricherService
sc.exe delete KafkaLogConsumerService
```

View logs

To view logs, open the **Event Viewer**. Select the Windows key (or Ctrl + Esc), and search for "Event Viewer". Select the **Event Viewer (Local)** > **Windows Logs** > **Application** node. You should see a **Information** level entry with a **Source** matching the apps namespace. Double-click the entry, or right-click and select **Event Properties** to view the details.



Application Number of events: 21,685 (!) New events available

Level	Date and Time	Source	Event ID	Task Category
Information	29-03-2024 13:42:22	KafkaLogProducer Service	0	None
Information	29-03-2024 13:42:19	KafkaLogParser4j	0	None
Information	29-03-2024 13:42:19	KafkaLogParser4j	0	None
Information	29-03-2024 13:42:19	KafkaLogParser4j Service	0	None
Information	29-03-2024 12:34:12	SecurityCenter	15	None
Information	29-03-2024 12:33:14	SecurityCenter	15	None
Information	29-03-2024 11:12:16	brave	0	None
Information	29-03-2024 10:58:45	Windows Error Reporting	1001	None
Information	29-03-2024 10:49:16	SecurityCenter	15	None

Event 0, KafkaLogParser4j

General Details

```
Category: KafkaLogParser4j.WindowsBackgroundService
EventId: 0

Initiating Zookeeper and Kafka Brokers Servers Methods...
```

Log Name:	Application		
Source:	KafkaLogParser4j	Logged:	29-03-2024 13:42:19
Event ID:	0	Task Category:	None
Level:	Information	Keywords:	Classic
User:	N/A	Computer:	CoinStation
OpCode:	Info		
More Information:	Event Log Online Help		

Event Properties - Event 0, KafkaLogParser4j

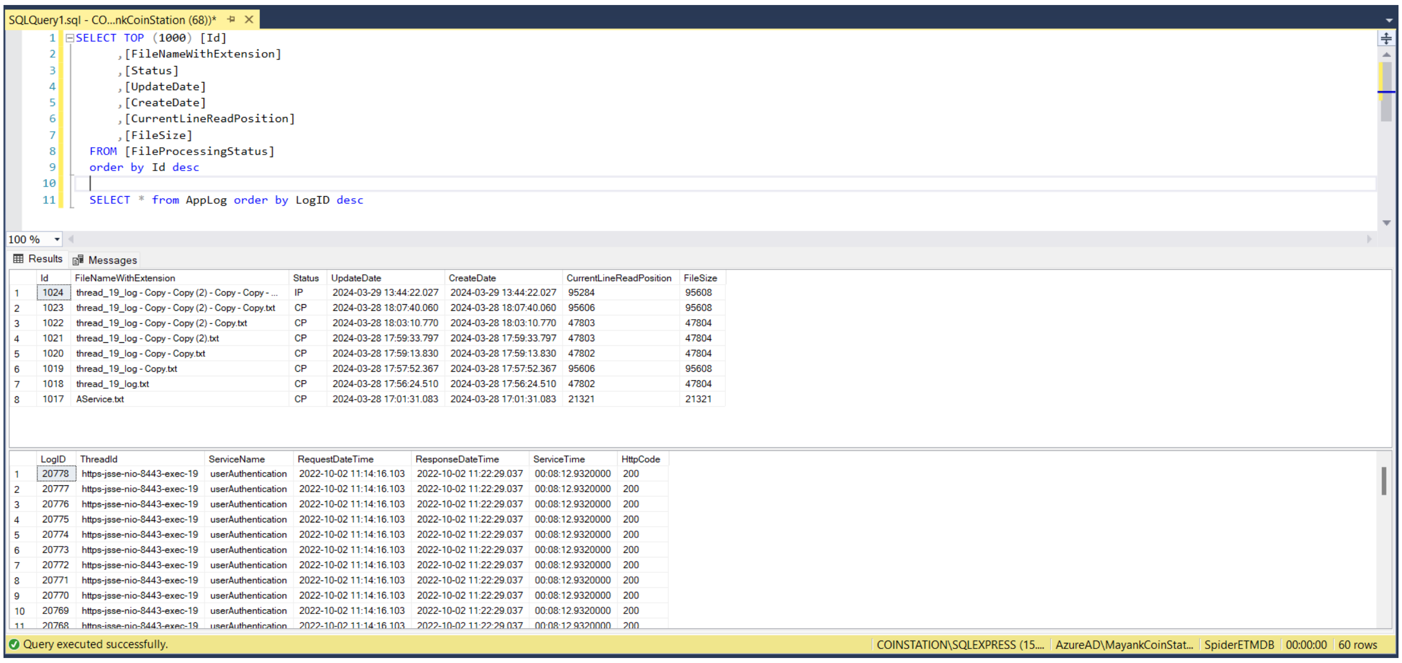
General Details

```
Category: KafkaLogParser4j.WindowsBackgroundService
EventId: 0

Initiating Zookeeper and Kafka Brokers Servers Methods...
```

Log Name:	Application		
Source:	KafkaLogParser4j	Logged:	29-03-2024 13:42:19
Event ID:	0	Task Category:	None
Level:	Information	Keywords:	Classic
User:	N/A	Computer:	CoinStation
OpCode:	Info		
More Information:	Event Log Online Help		

Database: If have database access, you can check tables - [FileProcessingStatus] and AppLog



```sql
SELECT TOP (1000) [Id]
      ,[FileNameWithExtension]
      ,[Status]
      ,[UpdateDate]
      ,[CreateDate]
      ,[CurrentLineReadPosition]
      ,[FileSize]
  FROM [FileProcessingStatus]
  order by Id desc

  SELECT * from AppLog order by LogID desc
```

	Id	FileNameWithExtension	Status	UpdateDate	CreateDate	CurrentLineReadPosition	FileSize
1	1024	thread_19_log - Copy - Copy (2) - Copy - Copy - ...	IP	2024-03-29 13:44:22.027	2024-03-29 13:44:22.027	95284	95608
2	1023	thread_19_log - Copy - Copy (2) - Copy - Copy.txt	CP	2024-03-28 18:07:40.060	2024-03-28 18:07:40.060	95606	95608
3	1022	thread_19_log - Copy - Copy (2) - Copy.txt	CP	2024-03-28 18:03:10.770	2024-03-28 18:03:10.770	47803	47804
4	1021	thread_19_log - Copy - Copy (2).txt	CP	2024-03-28 17:59:33.797	2024-03-28 17:59:33.797	47803	47804
5	1020	thread_19_log - Copy - Copy.txt	CP	2024-03-28 17:59:13.830	2024-03-28 17:59:13.830	47802	47804
6	1019	thread_19_log - Copy.txt	CP	2024-03-28 17:57:52.367	2024-03-28 17:57:52.367	95606	95608
7	1018	thread_19_log.txt	CP	2024-03-28 17:56:24.510	2024-03-28 17:56:24.510	47802	47804
8	1017	AService.txt	CP	2024-03-28 17:01:31.083	2024-03-28 17:01:31.083	21321	21321

	LogID	ThreadId	ServiceName	RequestDateTime	ResponseDateTime	ServiceTime	HttpCode
1	20778	https-jsse-nio-8443-exec-19	userAuthentication	2022-10-02 11:14:16.103	2022-10-02 11:22:29.037	00:08:12.9320000	200
2	20777	https-jsse-nio-8443-exec-19	userAuthentication	2022-10-02 11:14:16.103	2022-10-02 11:22:29.037	00:08:12.9320000	200
3	20776	https-jsse-nio-8443-exec-19	userAuthentication	2022-10-02 11:14:16.103	2022-10-02 11:22:29.037	00:08:12.9320000	200
4	20775	https-jsse-nio-8443-exec-19	userAuthentication	2022-10-02 11:14:16.103	2022-10-02 11:22:29.037	00:08:12.9320000	200
5	20774	https-jsse-nio-8443-exec-19	userAuthentication	2022-10-02 11:14:16.103	2022-10-02 11:22:29.037	00:08:12.9320000	200
6	20773	https-jsse-nio-8443-exec-19	userAuthentication	2022-10-02 11:14:16.103	2022-10-02 11:22:29.037	00:08:12.9320000	200
7	20772	https-jsse-nio-8443-exec-19	userAuthentication	2022-10-02 11:14:16.103	2022-10-02 11:22:29.037	00:08:12.9320000	200
8	20771	https-jsse-nio-8443-exec-19	userAuthentication	2022-10-02 11:14:16.103	2022-10-02 11:22:29.037	00:08:12.9320000	200
9	20770	https-jsse-nio-8443-exec-19	userAuthentication	2022-10-02 11:14:16.103	2022-10-02 11:22:29.037	00:08:12.9320000	200
10	20769	https-jsse-nio-8443-exec-19	userAuthentication	2022-10-02 11:14:16.103	2022-10-02 11:22:29.037	00:08:12.9320000	200
11	20768	https-jsse-nio-8443-exec-19	userAuthentication	2022-10-02 11:14:16.103	2022-10-02 11:22:29.037	00:08:12.9320000	200

Query executed successfully. | COINSTATION\SQLEXPRESS (15.... | AzureAD\MayankCoinStat... | SpiderETMDB | 00:00:00 | 60 rows