Scenario-9

Problem statement

The client just upgraded their SSL certificates used for the inter broker communication. The cluster was healthy before the certificate updates. After the certificate updates, the client sees the following error in the broker logs -

```
Caused by: java.util.concurrent.CompletionException:
org.apache.kafka.common.errors.TopicAuthorizationException: Not authorized to access topics:
[_confluent-metadata-auth]
        at java.base/java.util.concurrent.CompletableFuture.encodeRelay(CompletableFuture.java:367)
        at java.base/java.util.concurrent.CompletableFuture.completeRelay(CompletableFuture.java:376)
        at java.base/java.util.concurrent.CompletableFuture$AnyOf.tryFire(CompletableFuture.java:1663)
        at java.base/java.util.concurrent.CompletableFuture.postComplete(CompletableFuture.java:506)
        at java.base/java.util.concurrent.CompletableFuture.completeExceptionally(CompletableFuture.java:2088)
        at io.confluent.security.auth.provider.ConfluentProvider.lambda$null$10(ConfluentProvider.java:543)
        at java.base/java.util.concurrent.CompletableFuture.uniExceptionally(CompletableFuture.java:986)
        at
java.base/java.util.concurrent.CompletableFuture$UniExceptionally.tryFire(CompletableFuture.java:970)
        at java.base/java.util.concurrent.CompletableFuture.postComplete(CompletableFuture.java:506)
        at java.base/java.util.concurrent.CompletableFuture.completeExceptionally(CompletableFuture.java:2088)
        at io.confluent.security.store.kafka.clients.KafkaReader.lambda$start$1(KafkaReader.java:102)
        at java.base/java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:515)
        at java.base/java.util.concurrent.FutureTask.run(FutureTask.java:264)
        at java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1128)
        at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:628)
        at java.base/java.lang.Thread.run(Thread.java:829)
```

Observation

There is an issue with the newly created SSL certificate.

Root cause analysis

The issue persists only after creating new SSL certificates, hence lets create new SSL certificates and configure the brokers to use them.

Solution to debug the issue

1. Create a new Certificate Authority (CA) using OpenSSL request

→openssl req -new -x509 -keyout ca-key.pem -out ca-cert.pem -days 365 -subj "/C=IN/ST=Karnataka/L=Uttarahalli/O=Platformatory/OU=Kafka/CN=Platformatory-CA"

2. Creating new certificates for kafka broker 1

→keytool -keystore kafka1.server.keystore.jks -alias kafka1 -validity 365 -genkey -keyalg RSA -dname "CN=kafka1, OU=Kafka, O=Platformatory, L=Uttarahalli, ST=Karnataka, C=IN"

→keytool -keystore kafka1.server.keystore.jks -alias kafka1 -certreq -file kafka1-cert-file

→openssl x509 -req -CA ca-cert.pem -CAkey ca-key.pem -in kafka1-cert-file -out kafka1-cert-signed.pem -days 365 -CAcreateserial

→keytool -keystore kafka1.server.keystore.jks -alias CARoot -import -file ca-cert.pem

→keytool -keystore kafka1.server.keystore.jks -alias kafka1 -import -file kafka1-cert-signed.pem

→openssl verify -CAfile ca-cert.pem kafka1-cert-signed.pem
#kafka1-cert-signed.pem: OK

→keytool -importcert -file ca-cert.pem -alias CARoot -keystore kafka1.server.truststore.jks

3. Creating new certificates for kafka broker 3

→keytool -keystore kafka2.server.keystore.jks -alias kafka2 -validity 365 -genkey -keyalg RSA -dname "CN=kafka2, OU=Kafka, O=Platformatory, L=Uttarahalli, ST=Karnataka, C=IN"

→keytool -keystore kafka2.server.keystore.jks -alias kafka2 -certreq -file kafka2-cert-file

→openssl x509 -req -CA ca-cert.pem -CAkey ca-key.pem -in kafka2-cert-file -out kafka2-cert-signed.pem -days 365 -CAcreateserial

→keytool -keystore kafka2.server.keystore.jks -alias CARoot -import -file ca-cert.pem

→keytool -keystore kafka2.server.keystore.jks -alias kafka2 -import -file kafka2-cert-signed.pem

→openssl verify -CAfile ca-cert.pem kafka2-cert-signed.pem
#kafka2-cert-signed.pem: OK

→keytool -importcert -file ca-cert.pem -alias CARoot -keystore kafka2.server.truststore.jks

4. Creating new certificates for kafka broker 3

→keytool -keystore kafka3.server.keystore.jks -alias kafka3 -validity 365 -genkey -keyalg RSA -dname "CN=kafka3, OU=Kafka, O=Platformatory, L=Uttarahalli, ST=Karnataka, C=IN"

→keytool -keystore kafka3.server.keystore.jks -alias kafka3 -certreq -file kafka3-cert-file

→openssl x509 -req -CA ca-cert.pem -CAkey ca-key.pem -in kafka3-cert-file -out kafka3-cert-signed.pem -days 365 -CAcreateserial

→keytool -keystore kafka3.server.keystore.jks -alias CARoot -import -file ca-cert.pem

→keytool -keystore kafka3.server.keystore.jks -alias kafka3 -import -file kafka3-cert-signed.pem

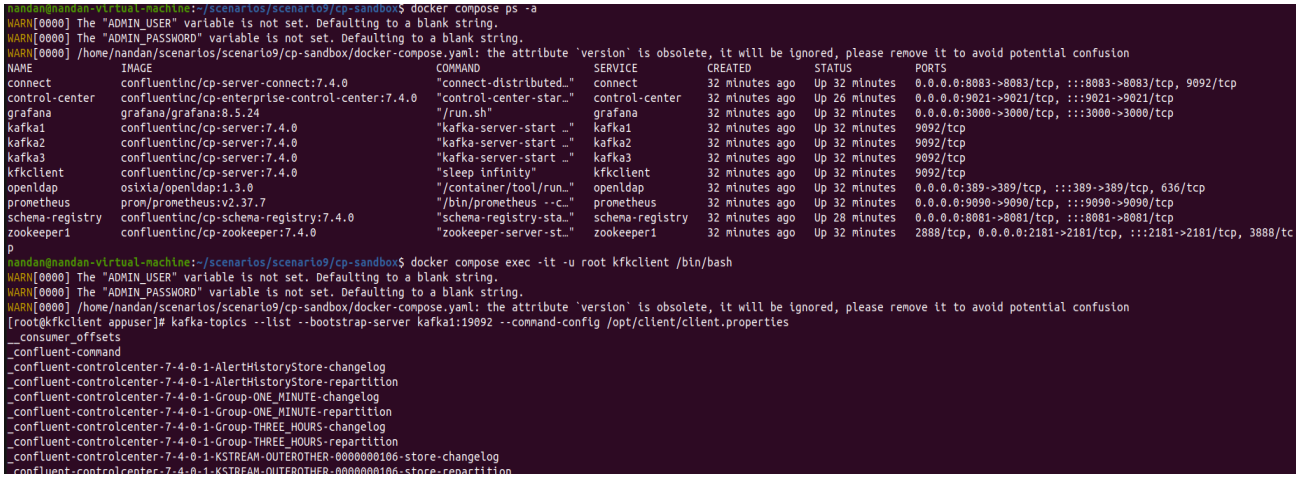
→openssl verify -CAfile ca-cert.pem kafka3-cert-signed.pem
#kafka3-cert-signed.pem: OK

→keytool -importcert -file ca-cert.pem -alias CARoot -keystore kafka3.server.truststore.jks

Note: Move the CA certificate and its key and also the keystores and truststore files to their respective location on their respective brokers (Change name if necessary).

5. Test the newly generated certificates by running the following command

→ kafka-topics --list --bootstrap-server kafka1:19092 --command-config /opt/client/client.properties



Observation

All the containers are configured to use the newly generated SSL certificates and all the containers are up and running, thus it can list all the topics

Conclusion

Since the previously created certificates had some errors while creating them, we just delete the old certificates and replace them with the newly created ones.