package com.twitter.cr\_mixer.module.core

import com.google.inject.Provides

import com.twitter.cr\_mixer.thriftscala.GetTweetsRecommendationsScribe

import com.twitter.finagle.mtls.authentication.ServiceIdentifier

import com.twitter.finatra.kafka.producers.FinagleKafkaProducerBuilder

import com.twitter.finatra.kafka.producers.KafkaProducerBase

import com.twitter.finatra.kafka.producers.NullKafkaProducer

import com.twitter.finatra.kafka.serde.ScalaSerdes

import com.twitter.inject.TwitterModule

import javax.inject.Singleton

import org.apache.kafka.clients.CommonClientConfigs

import org.apache.kafka.common.config.SaslConfigs

import org.apache.kafka.common.config.SslConfigs

import org.apache.kafka.common.record.CompressionType

import org.apache.kafka.common.security.auth.SecurityProtocol

import org.apache.kafka.common.serialization.Serdes

object KafkaProducerModule extends TwitterModule {

@Provides

@Singleton

def provideTweetRecsLoggerFactory(

serviceIdentifier: ServiceIdentifier,

): KafkaProducerBase[String, GetTweetsRecommendationsScribe] = {

KafkaProducerFactory.getKafkaProducer(serviceIdentifier.environment)

}

}

object KafkaProducerFactory {

private val jaasConfig =

"""com.sun.security.auth.module.Krb5LoginModule

|required

|principal="cr-mixer@TWITTER.BIZ"

|debug=true

|useKeyTab=true

|storeKey=true

|keyTab="/var/lib/tss/keys/fluffy/keytabs/client/cr-mixer.keytab"

|doNotPrompt=true;

""".stripMargin.replaceAll("\n", " ")

private val trustStoreLocation = "/etc/tw\_truststore/messaging/kafka/client.truststore.jks"

def getKafkaProducer(

environment: String

): KafkaProducerBase[String, GetTweetsRecommendationsScribe] = {

if (environment == "prod") {

FinagleKafkaProducerBuilder()

.dest("/s/kafka/recommendations:kafka-tls")

// kerberos params

.withConfig(SaslConfigs.SASL\_JAAS\_CONFIG, jaasConfig)

.withConfig(

CommonClientConfigs.SECURITY\_PROTOCOL\_CONFIG,

SecurityProtocol.SASL\_SSL.toString)

.withConfig(SslConfigs.SSL\_TRUSTSTORE\_LOCATION\_CONFIG, trustStoreLocation)

.withConfig(SaslConfigs.SASL\_MECHANISM, SaslConfigs.GSSAPI\_MECHANISM)

.withConfig(SaslConfigs.SASL\_KERBEROS\_SERVICE\_NAME, "kafka")

.withConfig(SaslConfigs.SASL\_KERBEROS\_SERVER\_NAME, "kafka")

// Kafka params

.keySerializer(Serdes.String.serializer)

.valueSerializer(ScalaSerdes.CompactThrift[GetTweetsRecommendationsScribe].serializer())

.clientId("cr-mixer")

.enableIdempotence(true)

.compressionType(CompressionType.LZ4)

.build()

} else {

new NullKafkaProducer[String, GetTweetsRecommendationsScribe]

}

}

}