package com.twitter.search.feature\_update\_service.modules;

import javax.inject.Singleton;

import com.google.inject.Provides;

import com.twitter.finagle.Service;

import com.twitter.finagle.ThriftMux;

import com.twitter.finagle.builder.ClientBuilder;

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

import com.twitter.finagle.mtls.client.MtlsThriftMuxClient;

import com.twitter.finagle.stats.StatsReceiver;

import com.twitter.finagle.thrift.ClientId;

import com.twitter.finagle.thrift.ThriftClientRequest;

import com.twitter.finagle.zipkin.thrift.ZipkinTracer;

import com.twitter.inject.TwitterModule;

import com.twitter.spam.finagle.FinagleUtil;

import com.twitter.tweetypie.thriftjava.TweetService;

import com.twitter.util.Duration;

public class TweetypieModule extends TwitterModule {

@Provides

@Singleton

private ThriftMux.Client providesThriftMuxClient(ServiceIdentifier serviceIdentifier) {

return new MtlsThriftMuxClient(ThriftMux.client())

.withMutualTls(serviceIdentifier)

.withClientId(new ClientId("feature\_update\_service.prod"));

}

private static final Duration DEFAULT\_CONN\_TIMEOUT = Duration.fromSeconds(2);

private static final Duration TWEET\_SERVICE\_REQUEST\_TIMEOUT = Duration.fromMilliseconds(500);

private static final int TWEET\_SERVICE\_RETRIES = 5;

@Provides @Singleton

private TweetService.ServiceIface provideTweetServiceClient(

ThriftMux.Client thriftMux,

StatsReceiver statsReceiver) throws InterruptedException {

// TweetService is TweetService (tweetypie) with different api

// Since TweetService will be primarly used for interacting with

// tweetypie's flexible schema (MH), we will increase request

// timeout and retries but share other settings from TweetService.

@SuppressWarnings("unchecked")

ClientBuilder clientBuilder = FinagleUtil.getClientBuilder()

.name("tweet\_service")

.stack(thriftMux)

.tcpConnectTimeout(DEFAULT\_CONN\_TIMEOUT)

.requestTimeout(TWEET\_SERVICE\_REQUEST\_TIMEOUT)

.retries(TWEET\_SERVICE\_RETRIES)

.reportTo(statsReceiver)

.tracer(ZipkinTracer.mk(statsReceiver));

@SuppressWarnings("unchecked")

final Service<ThriftClientRequest, byte[]> finagleClient =

FinagleUtil.createResolvedFinagleClient(

"tweetypie",

"prod",

"tweetypie",

clientBuilder);

return new TweetService.ServiceToClient(finagleClient);

}

}