package com.twitter.search.ingester.pipeline.wire;

import java.util.concurrent.TimeoutException;

import javax.naming.Context;

import javax.naming.InitialContext;

import javax.naming.NamingException;

import org.apache.thrift.protocol.TBinaryProtocol;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import com.twitter.common\_internal.zookeeper.TwitterServerSet;

import com.twitter.finagle.Name;

import com.twitter.finagle.Resolvers;

import com.twitter.finagle.Service;

import com.twitter.finagle.ThriftMux;

import com.twitter.finagle.builder.ClientBuilder;

import com.twitter.finagle.builder.ClientConfig;

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

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

import com.twitter.finagle.mux.transport.OpportunisticTls;

import com.twitter.finagle.service.RetryPolicy;

import com.twitter.finagle.stats.DefaultStatsReceiver;

import com.twitter.finagle.thrift.ClientId;

import com.twitter.finagle.thrift.ThriftClientRequest;

import com.twitter.servo.util.WaitForServerSets;

import com.twitter.tweetypie.thriftjava.TweetService;

import com.twitter.util.Await;

import com.twitter.util.Duration;

final class TweetyPieWireModule {

private static final Logger LOG = LoggerFactory.getLogger(ProductionWireModule.class);

private static final int TWEETYPIE\_CONNECT\_TIMEOUT\_MS = 100;

private static final int TWEETYPIE\_REQUEST\_TIMEOUT\_MS = 500;

// This is actually the total tries count, so one initial try, and one more retry (if needed).

private static final int TWEETYPIE\_REQUEST\_NUM\_TRIES = 3;

private static final int TWEETYPIE\_TOTAL\_TIMEOUT\_MS =

TWEETYPIE\_REQUEST\_TIMEOUT\_MS \* TWEETYPIE\_REQUEST\_NUM\_TRIES;

private static final String TWEETYPIE\_SD\_ZK\_ROLE =

WireModule.JNDI\_PIPELINE\_ROOT + "tweetypieSDZKRole";

private static final String TWEETYPIE\_SD\_ZK\_ENV =

WireModule.JNDI\_PIPELINE\_ROOT + "tweetypieSDZKEnv";

private static final String TWEETYPIE\_SD\_ZK\_NAME =

WireModule.JNDI\_PIPELINE\_ROOT + "tweetypieSDZKName";

private TweetyPieWireModule() {

}

private static TwitterServerSet.Service getTweetyPieZkServerSetService()

throws NamingException {

Context jndiContext = new InitialContext();

TwitterServerSet.Service service = new TwitterServerSet.Service(

(String) jndiContext.lookup(TWEETYPIE\_SD\_ZK\_ROLE),

(String) jndiContext.lookup(TWEETYPIE\_SD\_ZK\_ENV),

(String) jndiContext.lookup(TWEETYPIE\_SD\_ZK\_NAME));

LOG.info("TweetyPie ZK path: {}", TwitterServerSet.getPath(service));

return service;

}

static TweetService.ServiceToClient getTweetyPieClient(

String clientIdString, ServiceIdentifier serviceIdentifier) throws NamingException {

TwitterServerSet.Service service = getTweetyPieZkServerSetService();

// Use explicit Name types so we can force a wait on resolution (COORD-479)

String destString = String.format("/cluster/local/%s/%s/%s",

service.getRole(), service.getEnv(), service.getName());

Name destination = Resolvers.eval(destString);

try {

Await.ready(WaitForServerSets.ready(destination, Duration.fromMilliseconds(10000)));

} catch (TimeoutException e) {

LOG.warn("Timed out while resolving Zookeeper ServerSet", e);

} catch (InterruptedException e) {

LOG.warn("Interrupted while resolving Zookeeper ServerSet", e);

Thread.currentThread().interrupt();

}

LOG.info("Creating Tweetypie client with ID: {}", clientIdString);

ClientId clientId = new ClientId(clientIdString);

MtlsThriftMuxClient mtlsThriftMuxClient = new MtlsThriftMuxClient(

ThriftMux.client().withClientId(clientId));

ThriftMux.Client tmuxClient = mtlsThriftMuxClient

.withMutualTls(serviceIdentifier)

.withOpportunisticTls(OpportunisticTls.Required());

ClientBuilder<

ThriftClientRequest,

byte[],

ClientConfig.Yes,

ClientConfig.Yes,

ClientConfig.Yes> builder = ClientBuilder.get()

.stack(tmuxClient)

.name("retrieve\_cards\_tweetypie\_client")

.dest(destination)

.reportTo(DefaultStatsReceiver.get())

.connectTimeout(Duration.fromMilliseconds(TWEETYPIE\_CONNECT\_TIMEOUT\_MS))

.requestTimeout(Duration.fromMilliseconds(TWEETYPIE\_REQUEST\_TIMEOUT\_MS))

.timeout(Duration.fromMilliseconds(TWEETYPIE\_TOTAL\_TIMEOUT\_MS))

.retryPolicy(RetryPolicy.tries(

TWEETYPIE\_REQUEST\_NUM\_TRIES,

RetryPolicy.TimeoutAndWriteExceptionsOnly()));

Service<ThriftClientRequest, byte[]> clientBuilder = ClientBuilder.safeBuild(builder);

return new TweetService.ServiceToClient(clientBuilder, new TBinaryProtocol.Factory());

}

}