package com.twitter.graph\_feature\_service.server.modules

import com.google.inject.Provides

import com.twitter.conversions.DurationOps.\_

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

import com.twitter.finagle.mtls.client.MtlsStackClient.\_

import com.twitter.finagle.ThriftMux

import com.twitter.finagle.service.RetryBudget

import com.twitter.graph\_feature\_service.thriftscala

import com.twitter.inject.TwitterModule

import com.twitter.inject.annotations.Flag

import com.twitter.util.{Await, Duration}

import javax.inject.Singleton

case class GraphFeatureServiceWorkerClients(

workers: Seq[thriftscala.Worker.MethodPerEndpoint])

object GraphFeatureServiceWorkerClientsModule extends TwitterModule {

private[this] val closeableGracePeriod: Duration = 1.second

private[this] val requestTimeout: Duration = 25.millis

@Provides

@Singleton

def provideGraphFeatureServiceWorkerClient(

@Flag(ServerFlagNames.NumWorkers) numWorkers: Int,

@Flag(ServerFlagNames.ServiceRole) serviceRole: String,

@Flag(ServerFlagNames.ServiceEnv) serviceEnv: String,

serviceIdentifier: ServiceIdentifier

): GraphFeatureServiceWorkerClients = {

val workers: Seq[thriftscala.Worker.MethodPerEndpoint] =

(0 until numWorkers).map { id =>

val dest = s"/srv#/$serviceEnv/local/$serviceRole/graph\_feature\_service-worker-$id"

val client = ThriftMux.client

.withRequestTimeout(requestTimeout)

.withRetryBudget(RetryBudget.Empty)

.withMutualTls(serviceIdentifier)

.build[thriftscala.Worker.MethodPerEndpoint](dest, s"worker-$id")

onExit {

val closeable = client.asClosable

Await.result(closeable.close(closeableGracePeriod), closeableGracePeriod)

}

client

}

GraphFeatureServiceWorkerClients(workers)

}

}