package com.twitter.product\_mixer.shared\_library.http\_client

import com.twitter.finagle.Http

import com.twitter.finagle.Service

import com.twitter.finagle.client.Transporter

import com.twitter.finagle.http.ProxyCredentials

import com.twitter.finagle.http.Request

import com.twitter.finagle.http.Response

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.product\_mixer.shared\_library.http\_client.FinagleHttpClientBuilder.buildFinagleHttpClient

import com.twitter.util.Duration

object FinagleHttpClientWithProxyBuilder {

/\*\*

\* Build a Finagle HTTP client with Egress Proxy support using Credentials

\*

\* @param twitterProxyHostPort Twitter egress proxy host port

\* @param remoteProxyHostPort Remote proxy host port

\* @param requestTimeout HTTP client request timeout

\* @param connectTimeout HTTP client transport connect timeout

\* @param acquisitionTimeout HTTP client session acquisition timeout

\* @param proxyCredentials Proxy credentials

\* @param statsReceiver Stats

\*

\* @return Finagle HTTP client with Egress Proxy support using Credentials

\*/

def buildFinagleHttpClientWithCredentialProxy(

twitterProxyHostPort: HttpHostPort,

remoteProxyHostPort: HttpHostPort,

requestTimeout: Duration,

connectTimeout: Duration,

acquisitionTimeout: Duration,

proxyCredentials: ProxyCredentials,

statsReceiver: StatsReceiver,

): Http.Client = {

val httpClient = buildFinagleHttpClient(

requestTimeout = requestTimeout,

connectTimeout = connectTimeout,

acquisitionTimeout = acquisitionTimeout,

statsReceiver = statsReceiver

)

httpClient.withTransport

.httpProxyTo(

host = remoteProxyHostPort.toString,

credentials = Transporter.Credentials(proxyCredentials.username, proxyCredentials.password))

.withTls(remoteProxyHostPort.host)

}

/\*\*

\* Build a Finagle HTTP client with Egress Proxy support

\*

\* @param twitterProxyHostPort Twitter egress proxy host port

\* @param remoteProxyHostPort Remote proxy host port

\* @param requestTimeout HTTP client request timeout

\* @param connectTimeout HTTP client transport connect timeout

\* @param acquisitionTimeout HTTP client session acquisition timeout

\* @param statsReceiver Stats

\*

\* @return Finagle HTTP client with Egress Proxy support

\*/

def buildFinagleHttpClientWithProxy(

twitterProxyHostPort: HttpHostPort,

remoteProxyHostPort: HttpHostPort,

requestTimeout: Duration,

connectTimeout: Duration,

acquisitionTimeout: Duration,

statsReceiver: StatsReceiver,

): Http.Client = {

val httpClient = buildFinagleHttpClient(

requestTimeout = requestTimeout,

connectTimeout = connectTimeout,

acquisitionTimeout = acquisitionTimeout,

statsReceiver = statsReceiver

)

httpClient.withTransport

.httpProxyTo(remoteProxyHostPort.toString)

.withTls(remoteProxyHostPort.host)

}

/\*\*

\* Build a Finagle HTTP service with Egress Proxy support

\*

\* @param finagleHttpClientWithProxy Finagle HTTP client from which to build the service

\* @param twitterProxyHostPort Twitter egress proxy host port

\*

\* @return Finagle HTTP service with Egress Proxy support

\*/

def buildFinagleHttpServiceWithProxy(

finagleHttpClientWithProxy: Http.Client,

twitterProxyHostPort: HttpHostPort

): Service[Request, Response] = {

finagleHttpClientWithProxy.newService(twitterProxyHostPort.toString)

}

}