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

import com.twitter.conversions.DurationOps.\_

import com.twitter.finagle.ThriftMux

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

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

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.finagle.thrift.ClientId

import com.twitter.finagle.thrift.service.Filterable

import com.twitter.finagle.thrift.service.MethodPerEndpointBuilder

import com.twitter.finagle.thrift.service.ServicePerEndpointBuilder

import com.twitter.finagle.thriftmux.MethodBuilder

import com.twitter.util.Duration

import org.apache.thrift.protocol.TProtocolFactory

sealed trait Idempotency

case object NonIdempotent extends Idempotency

case class Idempotent(maxExtraLoadPercent: Double) extends Idempotency

object FinagleThriftClientBuilder {

/\*\*

\* Library to build a Finagle Thrift method per endpoint client is a less error-prone way when

\* compared to the builders in Finagle. This is achieved by requiring values for fields that should

\* always be set in practice. For example, request timeouts in Finagle are unbounded when not

\* explicitly set, and this method requires that timeout durations are passed into the method and

\* set on the Finagle builder.

\*

\* Usage of

\* [[com.twitter.inject.thrift.modules.ThriftMethodBuilderClientModule]] is almost always preferred,

\* and the Product Mixer component library [[com.twitter.product\_mixer.component\_library.module]]

\* package contains numerous examples. However, if multiple versions of a client are needed e.g.

\* for different timeout settings, this method is useful to easily provide multiple variants.

\*

\* @example

\* {{{

\* final val SampleServiceClientName = "SampleServiceClient"

\* @Provides

\* @Singleton

\* @Named(SampleServiceClientName)

\* def provideSampleServiceClient(

\* serviceIdentifier: ServiceIdentifier,

\* clientId: ClientId,

\* statsReceiver: StatsReceiver,

\* ): SampleService.MethodPerEndpoint =

\* buildFinagleMethodPerEndpoint[SampleService.ServicePerEndpoint, SampleService.MethodPerEndpoint](

\* serviceIdentifier = serviceIdentifier,

\* clientId = clientId,

\* dest = "/s/sample/sample",

\* label = "sample",

\* statsReceiver = statsReceiver,

\* idempotency = Idempotent(5.percent),

\* timeoutPerRequest = 200.milliseconds,

\* timeoutTotal = 400.milliseconds

\* )

\* }}}

\* @param serviceIdentifier Service ID used to S2S Auth

\* @param clientId Client ID

\* @param dest Destination as a Wily path e.g. "/s/sample/sample"

\* @param label Label of the client

\* @param statsReceiver Stats

\* @param idempotency Idempotency semantics of the client

\* @param timeoutPerRequest Thrift client timeout per request. The Finagle default is

\* unbounded which is almost never optimal.

\* @param timeoutTotal Thrift client total timeout. The Finagle default is

\* unbounded which is almost never optimal.

\* If the client is set as idempotent, which adds a

\* [[com.twitter.finagle.client.BackupRequestFilter]],

\* be sure to leave enough room for the backup request. A

\* reasonable (albeit usually too large) starting point is to

\* make the total timeout 2x relative to the per request timeout.

\* If the client is set as non-idempotent, the total timeout and

\* the per request timeout should be the same, as there will be

\* no backup requests.

\* @param connectTimeout Thrift client transport connect timeout. The Finagle default

\* of one second is reasonable but we lower this to match

\* acquisitionTimeout for consistency.

\* @param acquisitionTimeout Thrift client session acquisition timeout. The Finagle default

\* is unbounded which is almost never optimal.

\* @param protocolFactoryOverride Override the default protocol factory

\* e.g. [[org.apache.thrift.protocol.TCompactProtocol.Factory]]

\* @param servicePerEndpointBuilder implicit service per endpoint builder

\* @param methodPerEndpointBuilder implicit method per endpoint builder

\*

\* @see [[https://twitter.github.io/finagle/guide/MethodBuilder.html user guide]]

\* @see [[https://twitter.github.io/finagle/guide/MethodBuilder.html#idempotency user guide]]

\* @return method per endpoint Finagle Thrift Client

\*/

def buildFinagleMethodPerEndpoint[

ServicePerEndpoint <: Filterable[ServicePerEndpoint],

MethodPerEndpoint

](

serviceIdentifier: ServiceIdentifier,

clientId: ClientId,

dest: String,

label: String,

statsReceiver: StatsReceiver,

idempotency: Idempotency,

timeoutPerRequest: Duration,

timeoutTotal: Duration,

connectTimeout: Duration = 500.milliseconds,

acquisitionTimeout: Duration = 500.milliseconds,

protocolFactoryOverride: Option[TProtocolFactory] = None,

)(

implicit servicePerEndpointBuilder: ServicePerEndpointBuilder[ServicePerEndpoint],

methodPerEndpointBuilder: MethodPerEndpointBuilder[ServicePerEndpoint, MethodPerEndpoint]

): MethodPerEndpoint = {

val service: ServicePerEndpoint = buildFinagleServicePerEndpoint(

serviceIdentifier = serviceIdentifier,

clientId = clientId,

dest = dest,

label = label,

statsReceiver = statsReceiver,

idempotency = idempotency,

timeoutPerRequest = timeoutPerRequest,

timeoutTotal = timeoutTotal,

connectTimeout = connectTimeout,

acquisitionTimeout = acquisitionTimeout,

protocolFactoryOverride = protocolFactoryOverride

)

ThriftMux.Client.methodPerEndpoint(service)

}

/\*\*

\* Build a Finagle Thrift service per endpoint client.

\*

\* @note [[buildFinagleMethodPerEndpoint]] should be preferred over the service per endpoint variant

\*

\* @param serviceIdentifier Service ID used to S2S Auth

\* @param clientId Client ID

\* @param dest Destination as a Wily path e.g. "/s/sample/sample"

\* @param label Label of the client

\* @param statsReceiver Stats

\* @param idempotency Idempotency semantics of the client

\* @param timeoutPerRequest Thrift client timeout per request. The Finagle default is

\* unbounded which is almost never optimal.

\* @param timeoutTotal Thrift client total timeout. The Finagle default is

\* unbounded which is almost never optimal.

\* If the client is set as idempotent, which adds a

\* [[com.twitter.finagle.client.BackupRequestFilter]],

\* be sure to leave enough room for the backup request. A

\* reasonable (albeit usually too large) starting point is to

\* make the total timeout 2x relative to the per request timeout.

\* If the client is set as non-idempotent, the total timeout and

\* the per request timeout should be the same, as there will be

\* no backup requests.

\* @param connectTimeout Thrift client transport connect timeout. The Finagle default

\* of one second is reasonable but we lower this to match

\* acquisitionTimeout for consistency.

\* @param acquisitionTimeout Thrift client session acquisition timeout. The Finagle default

\* is unbounded which is almost never optimal.

\* @param protocolFactoryOverride Override the default protocol factory

\* e.g. [[org.apache.thrift.protocol.TCompactProtocol.Factory]]

\*

\* @return service per endpoint Finagle Thrift Client

\*/

def buildFinagleServicePerEndpoint[ServicePerEndpoint <: Filterable[ServicePerEndpoint]](

serviceIdentifier: ServiceIdentifier,

clientId: ClientId,

dest: String,

label: String,

statsReceiver: StatsReceiver,

idempotency: Idempotency,

timeoutPerRequest: Duration,

timeoutTotal: Duration,

connectTimeout: Duration = 500.milliseconds,

acquisitionTimeout: Duration = 500.milliseconds,

protocolFactoryOverride: Option[TProtocolFactory] = None,

)(

implicit servicePerEndpointBuilder: ServicePerEndpointBuilder[ServicePerEndpoint]

): ServicePerEndpoint = {

val thriftMux: ThriftMux.Client = ThriftMux.client

.withMutualTls(serviceIdentifier)

.withClientId(clientId)

.withLabel(label)

.withStatsReceiver(statsReceiver)

.withTransport.connectTimeout(connectTimeout)

.withSession.acquisitionTimeout(acquisitionTimeout)

val protocolThriftMux: ThriftMux.Client = protocolFactoryOverride

.map { protocolFactory =>

thriftMux.withProtocolFactory(protocolFactory)

}.getOrElse(thriftMux)

val methodBuilder: MethodBuilder = protocolThriftMux

.methodBuilder(dest)

.withTimeoutPerRequest(timeoutPerRequest)

.withTimeoutTotal(timeoutTotal)

val idempotencyMethodBuilder: MethodBuilder = idempotency match {

case NonIdempotent => methodBuilder.nonIdempotent

case Idempotent(maxExtraLoad) => methodBuilder.idempotent(maxExtraLoad = maxExtraLoad)

}

idempotencyMethodBuilder.servicePerEndpoint[ServicePerEndpoint]

}

}