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

import com.google.inject.Inject;

import com.google.inject.Singleton;

import com.twitter.finagle.Service;

import com.twitter.finatra.thrift.AbstractThriftFilter;

import com.twitter.finatra.thrift.ThriftRequest;

import com.twitter.inject.annotations.Flag;

import com.twitter.search.common.metrics.SearchRateCounter;

import com.twitter.search.feature\_update\_service.thriftjava.FeatureUpdateResponse;

import com.twitter.search.feature\_update\_service.thriftjava.FeatureUpdateResponseCode;

import com.twitter.search.feature\_update\_service.whitelist.ClientIdWhitelist;

import com.twitter.util.Future;

@Singleton

public class ClientIdWhitelistFilter extends AbstractThriftFilter {

private final boolean enabled;

private final ClientIdWhitelist whitelist;

private final SearchRateCounter unknownClientIdStat =

SearchRateCounter.export("unknown\_client\_id");

private final SearchRateCounter noClientIdStat =

SearchRateCounter.export("no\_client\_id");

@Inject

public ClientIdWhitelistFilter(

ClientIdWhitelist whitelist,

@Flag("client.whitelist.enable") Boolean enabled

) {

this.whitelist = whitelist;

this.enabled = enabled;

}

@Override

@SuppressWarnings("unchecked")

public <T, R> Future<R> apply(ThriftRequest<T> request, Service<ThriftRequest<T>, R> svc) {

if (!enabled) {

return svc.apply(request);

}

if (request.clientId().isEmpty()) {

noClientIdStat.increment();

return (Future<R>) Future.value(

new FeatureUpdateResponse(FeatureUpdateResponseCode.MISSING\_CLIENT\_ERROR)

.setDetailMessage("finagle clientId is required in request"));

} else if (!whitelist.isClientAllowed(request.clientId().get())) {

// It's safe to use get() in the above condition because

// clientId was already checked for emptiness

unknownClientIdStat.increment();

return (Future<R>) Future.value(

new FeatureUpdateResponse(FeatureUpdateResponseCode.UNKNOWN\_CLIENT\_ERROR)

.setDetailMessage(String.format(

"request contains unknown finagle clientId: %s", request.clientId().toString())));

} else {

return svc.apply(request);

}

}

}