package com.twitter.search.earlybird.common;

import org.apache.commons.codec.binary.Base64;

import org.apache.thrift.TException;

import org.apache.thrift.TSerializer;

import org.apache.thrift.protocol.TBinaryProtocol;

import org.slf4j.Logger;

import com.twitter.search.common.util.FinagleUtil;

import com.twitter.search.earlybird.thrift.EarlybirdRequest;

/\*\*

\* This class logs all requests that misses either the finagle Id or the client Id.

\*/

public final class UnknownClientRequestForLogging {

private static final Logger GENERAL\_LOG = org.slf4j.LoggerFactory.getLogger(

UnknownClientRequestForLogging.class);

private static final Logger LOG = org.slf4j.LoggerFactory.getLogger(

UnknownClientRequestForLogging.class.getName() + ".unknownClientRequests");

private final String logLine;

private final EarlybirdRequest request;

private final String clientId;

private final String finagleId;

private final Base64 base64 = new Base64();

private final TSerializer serializer = new TSerializer(new TBinaryProtocol.Factory());

private UnknownClientRequestForLogging(

String logLine,

EarlybirdRequest request,

String clientId,

String finagleId) {

this.logLine = logLine;

this.request = request;

this.clientId = clientId;

this.finagleId = finagleId;

}

/\*\*

\* Returns an UnknownClientRequestForLogging instance if a client ID is not set on the given

\* earlybird request. If the request has a client ID set, {@code null} is returned.

\*

\* @param logLine Additional information to propagate to the log file, when logging this request.

\* @param request The earlybird request.

\*/

public static UnknownClientRequestForLogging unknownClientRequest(

String logLine, EarlybirdRequest request) {

String clientId = ClientIdUtil.getClientIdFromRequest(request);

String finagleId = FinagleUtil.getFinagleClientName();

if (clientId.equals(ClientIdUtil.UNSET\_CLIENT\_ID)) {

return new UnknownClientRequestForLogging(logLine, request, clientId, finagleId);

} else {

return null;

}

}

private String asBase64() {

try {

// Need to make a deepCopy() here, because the request may still be in use (e.g. if we are

// doing this in the pre-logger), and we should not be modifying crucial fields on the

// EarlybirdRequest in place.

EarlybirdRequest clearedRequest = request.deepCopy();

clearedRequest.unsetClientRequestTimeMs();

return base64.encodeToString(serializer.serialize(clearedRequest));

} catch (TException e) {

GENERAL\_LOG.error("Failed to serialize request for logging.", e);

return "failed\_to\_serialize";

}

}

public void log() {

LOG.info("{},{},{},{}", clientId, finagleId, logLine, asBase64());

}

}