Infrastructure - Bug #3369

new HttpClient version causing SSLSocketException: "Socket is closed"

2012-10-25 21:03 - Rob Nahf

Status:	Closed	**Start date:**	2014-09-17
Priority:	Normal	**Due date:**	2014-12-15
Assignee:	Rob Nahf	**% Done:**	100%
Category:	d1_libclient_java	**Estimated time:**	0.00 hour
Target version:	CCI-1.5.0		
Milestone:	None	**Story Points:**	
Product Version:	1.3.1		

Description

The upgrade of the httpcomponents dependency from v4.1.3 to v4.2.2 caused some integration tests to fail with a low-level connection error. SSLSocketException Socket is closed.

I tested locally with the trust-all TrustManager that we used to have, but that did not change the situation, so is orthogonal to the problem. Reverting httpComponents to v4.1.3 fixed the problem both locally and on Hudson. see http://dev-testing.dataone.org:8080/hudson/job/d1_integration_DEV_CN_search/

I updated the pom files in trunk/d1_common_java and d1_libclient_java to limit the version to <= 4.1.3 using the maven notation "(,4.1.3]" to complain if dependency management tries to go beyond what is known to work.

It is still unclear whether v4.2.2 fixes a security hole, or is faulty. Guess is the latter, because of the exception thrown - it seems like different subcomponents aren't communicating, because usually a peer not authenticated exception is thrown.

Related issues:

Related to Infrastructure - Task #6510: libclient_java 1.3.1 Release	**Closed**	**2014-10-06**

History

#1 - 2014-04-30 20:16 - Rob Nahf

I recreated the errors that were no longer in Hudson, using the same integration test and using HttpClient 4.2.6. The exception does not affect all http calls in the test case, but once it occurs, the rest of the calls return the same exception (Socket is Closed). It seems to happen in the first of the many-object AbstractAuthorizationITDataone tests (with about 8 other tests passing first).

When running one unit test in isolation, I get the same result, so it is not a case of a build up of connections in the pool, or some timeout. It may be something with the subjects themselves, or how these types of tests are put together.

The stack-trace:

20140430-14:09:38: [INFO]: ====>>>>> pid of procured test Object: TierTesting:cn-stage-orc-1:RightsHolder_testPerson.14 [org.dataone.integration.it.ContextAwareTestCaseDataone]
20140430-14:09:38: [INFO]: client setup as Subject: CN=testPerson,DC=dataone,DC=org [org.dataone.integration.it.ContextAwareTestCaseDataone]
checkServerTrusted - RSA
20140430-14:09:40: [INFO]: GET https://cn-stage-orc-1.test.dataone.org/cn/v1/query/solr/?q=id:%22TierTesting:cn-stage-orc-1:RightsHolder_testPerson.14%22 [org.dataone.integration.it.ContextAwareTestCaseDataone]
java.net.SocketException: Socket is closed
at com.sun.net.ssl.internal.ssl.SSLSocketImpl.checkEOF(SSLSocketImpl.java:1334)
at com.sun.net.ssl.internal.ssl.AppInputStream.read(AppInputStream.java:65)
at org.apache.http.impl.io.AbstractSessionInputBuffer.fillBuffer(AbstractSessionInputBuffer.java:166)
at org.apache.http.impl.io.SocketInputBuffer.fillBuffer(SocketInputBuffer.java:90)
at org.apache.http.impl.io.AbstractSessionInputBuffer.readLine(AbstractSessionInputBuffer.java:281)
at org.apache.http.impl.io.ChunkedInputStream.getChunkSize(ChunkedInputStream.java:251)
at org.apache.http.impl.io.ChunkedInputStream.nextChunk(ChunkedInputStream.java:209)
at org.apache.http.impl.io.ChunkedInputStream.read(ChunkedInputStream.java:171)
at org.apache.http.conn.EofSensorInputStream.read(EofSensorInputStream.java:138)
at sun.nio.cs.StreamDecoder.readBytes(StreamDecoder.java:264)
at sun.nio.cs.StreamDecoder.implRead(StreamDecoder.java:306)
at sun.nio.cs.StreamDecoder.read(StreamDecoder.java:158)
at java.io.InputStreamReader.read(InputStreamReader.java:167)
at java.io.Reader.read(Reader.java:123)
at org.apache.commons.io.IOUtils.copyLarge(IOUtils.java:1128)

at org.apache.commons.io.IOUtils.copy(IOUtils.java:1104)
at org.apache.commons.io.IOUtils.copy(IOUtils.java:1078)
at org.apache.commons.io.IOUtils.toString(IOUtils.java:382)
at org.dataone.integration.it.CNodeTier1_search_IT.runAuthTest(CNodeTier1_search_IT.java:103)
at org.dataone.integration.it.AbstractAuthorizationITDataone.checkExpectedIsAuthorizedOutcome(AbstractAuthorizationITDataone.java:99)
at
org.dataone.integration.it.AbstractAuthorizationITDataone.testIsAuthorized_NullPolicy_testPerson_is_RightsHolder(AbstractAuthorizationITDataone.java:319)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
at java.lang.reflect.Method.invoke(Method.java:597)
at org.junit.runners.model.FrameworkMethod$1.runReflectiveCall(FrameworkMethod.java:44)
at org.junit.internal.runners.model.ReflectiveCallable.run(ReflectiveCallable.java:15)
at org.junit.runners.model.FrameworkMethod.invokeExplosively(FrameworkMethod.java:41)
at org.junit.internal.runners.statements.InvokeMethod.evaluate(InvokeMethod.java:20)
at org.junit.rules.Verifier$1.evaluate(Verifier.java:34)
at org.junit.internal.runners.statements.RunBefores.evaluate(RunBefores.java:28)
at org.junit.runners.BlockJUnit4ClassRunner.runChild(BlockJUnit4ClassRunner.java:76)
at org.junit.runners.BlockJUnit4ClassRunner.runChild(BlockJUnit4ClassRunner.java:50)
at org.junit.runners.ParentRunner$3.run(ParentRunner.java:193)
at org.junit.runners.ParentRunner$1.schedule(ParentRunner.java:52)
at org.junit.runners.ParentRunner.runChildren(ParentRunner.java:191)
at org.junit.runners.ParentRunner.access$000(ParentRunner.java:42)
at org.junit.runners.ParentRunner$2.evaluate(ParentRunner.java:184)
at org.junit.runners.ParentRunner.run(ParentRunner.java:236)
at org.eclipse.jdt.internal.junit4.runner.JUnit4TestReference.run(JUnit4TestReference.java:50)
at org.eclipse.jdt.internal.junit.runner.TestExecution.run(TestExecution.java:38)
at org.eclipse.jdt.internal.junit.runner.RemoteTestRunner.runTests(RemoteTestRunner.java:467)
at org.eclipse.jdt.internal.junit.runner.RemoteTestRunner.runTests(RemoteTestRunner.java:683)
at org.eclipse.jdt.internal.junit.runner.RemoteTestRunner.run(RemoteTestRunner.java:390)
at org.eclipse.jdt.internal.junit.runner.RemoteTestRunner.main(RemoteTestRunner.java:197)
20140430-14:09:40: [INFO]: client setup as Subject: CN=testPerson,DC=dataone,DC=org
[org.dataone.integration.it.ContextAwareTestCaseDataone]

#2 - 2014-04-30 21:52 - Rob Nahf

When debugging the first sub-test, the first call is to procure a test object (find or create), and begins with a getSystemMetadata call. This succeeds (and finds an object).

A create is not needed, so the actual permission test begins. The https call succeeds, returning a 200 OK status. The Socket is Closed exception comes when trying to read the input stream, which in this case is a query response, so not turned into a D1 datatype by the cn api method.

the behavior of closeIdleConnections seems to have changed between 4.13 and 4.2.1 - commenting out the D1ResClient.closeIdleConnections call in the d1node.query method caused the tests to pass.

public void closeIdleConnections()
{
getHttpClient().getConnectionManager().closeIdleConnections(0,TimeUnit.MILLISECONDS);
}

#3 - 2014-05-02 07:17 - Rob Nahf

refactored the d1node.query method to cache the input stream in a local ByteArrayInputStream, so we can close the connection. This is what we did with get() and getReplica(). It's not ideal, but gets around the change in behavior in connection management between 4.1.3 and 4.2.3. (In httpclient 4.1.3, closeIdleConnections did not cause a "socket is closed" (SocketException, which I believe is the correct behavior).

To really diagnose where and how the behavior change happened, it would probably be easiest to download the source code and run under the debugger in Eclipse.

#4 - 2014-06-18 23:00 - Ben Leinfelder

- Milestone changed from CCI-1.1.0 to None

- Due date deleted (2012-10-27)

- Status changed from Closed to In Progress

- Start date deleted (2012-10-25)

- Target version deleted (Sprint-2012.41-Block.6.1)

Can we revist this? Rather than put the entire response from query() or get() into memory, it would be good if the client streamed the result. There will be cases (hopefully many) where the search results and certainly any data files will be large and will not fit in memory.
Why not omit the call to closeIdleConnections if this makes the tests pass? Seems the connections are not actually idle when they are closed....

#5 - 2014-09-03 09:13 - Rob Nahf

wrt/ how the closeIdleConnections method works, since the method is called before the inputStream is returned from d1.query and consumed, it does not close the connection to that one. At least, that's how it worked with v4.1.3. The closeIdleConnections was put in to clean up other possible idle connections. The method is deprecated in httpclient v4.3.x so will be removed in the 2.0 libclient refactor. Will keep it in 1.3 branch as insurance against the vague problems with system resources being consumed.

#6 - 2014-10-01 22:53 - Rob Nahf

- Target version set to CCI-1.4.2

- Due date set to 2014-10-01

- Start date set to 2014-10-01

#7 - 2014-10-02 19:34 - Robert Waltz

- Target version changed from CCI-1.4.2 to CCI-1.4.1

- Due date changed from 2014-10-01 to 2014-10-02

#8 - 2014-10-02 22:02 - Rob Nahf

- Status changed from In Progress to Testing

#9 - 2014-10-06 22:57 - Rob Nahf

- Product Version set to 1.3.1

#10 - 2014-10-07 17:38 - Robert Waltz

- Due date changed from 2014-10-17 to 2014-10-07

- Target version changed from CCI-1.4.1 to CCI-1.5.0

#11 - 2014-12-15 18:32 - Rob Nahf

- Status changed from Testing to Closed

- % Done changed from 0 to 100

- Due date changed from 2014-10-07 to 2014-12-15

tested in stage for CN-1.5.0 release. Showing no problems.