EDIT - bug #10075

Fix deserialization exception for bytebuddy base hibernate SerializableProxy

06/16/2022 02:51 PM - Andreas Müller

Status:	Closed	**Start date:**	
Priority:	Highest	**Due date:**	
Assignee:	Andreas Müller	**% Done:**	100%
Category:	cdmlib	**Estimated time:**	0:00 hour
Target version:	Release 5.32		
Severity:	normal	**Found in Version:**	

Description

... this is related to hibernate upgrade to v5.4 (#10067)

This happens when deserializing a org.hibernate.proxy.pojo.bytebuddy.SerializableProxy during ehCache read from disc.

Generally hibernate proxies (org.hibernate.proxy.HibernateProxy) are serialized by using the writeReplace() method which transforms the HibernateProxy into a SerializableProxy. For deserializing the readResolve() method from SerializableProxy is used.

This works for serializing/deserializing during httpInvoker transportation. It also seems to work with the javassist SerializableProxy (org.hibernate.proxy.pojo.javassist.SerializableProxy). But does not work for the bytebuddy version.

The problem occurs during deserialization of bytebuddy.SerializableProxy itself (so still before calling readResolve()). The exact moment when it happens is deserialization of the parameter "Class[] SerializableProxy.identifierSetterMethodParams" which has as content "int" and the net.sf.ehcache.util.PreferredLoaderObjectInputStream used by ehCache uses Class.forName(String name, boolean initialize, ClassLoader loader) for class loading which does not handle primitive types ("this method cannot be used to obtain any of the Class objects representing primitive types").
httpInvoker uses org.springframework.remoting.rmi.CodebaseAwareObjectInputStream.
TODO: check what method spring uses for class loading.

javassist:

It is unclear why we did not observe this problem with javassist under hibernate 5.1. as the javassist.SerializableProxy looks equal in terms of datastructure to be loaded (
https://docs.jboss.org/hibernate/orm/5.1/javadocs/org/hibernate/proxy/pojo/javassist/SerializableProxy.html).

Related issues:

Related to EDIT - task #10077: upgrade ehcache to 3.x	**New**

Associated revisions

Revision 9ee1ca44 - 06/16/2022 09:47 PM - Andreas Müller

fix #10075 fix deserialization by overriding PreferredLoaderObjectInputStream from ehcache

Revision e16caaca - 07/22/2022 11:52 PM - Andreas Müller

ref #10075, ref #9908 adapt test sql for OriginalSourceBase.accessed and Credits.timePeriod

History

#1 - 06/16/2022 03:24 PM - Andreas Müller

- Description updated

#2 - 06/16/2022 04:50 PM - Andreas Müller

it happens in
readObject0.readOrdinaryObject.readSerialData.defaultReadFields.readObject0.(TCArray)checkResolve(readArray().readObject0.readClass.readClassDesc.readNonProxyDesc.resolveClass[classDesc.name=int] => catches ClassNotFoundException and handles primitive types in ordinary ObjectInputStream.resolveClass() used e.g. by CodbaseAwareOIS.

The ehCache PreferredLoaderObjectInputStream overrides this method and *avoids the handling the primitive types*. (The classloader used here is in current context EhCacheDefaultClassLoader by calling Class.forName(desc.getName(), false, loader).

So maybe it is possible to override resolveClass() with the same behaviour as in default ObjectInputStream or to avoid usage of PreferredLoaderObjectInputStream.

Note: default OIS uses latestUserDefinedLoader() which resolves to an EquinoxClassLoader in httpInvoker, but this is not really relevant here.

#3 - 06/16/2022 09:43 PM - Andreas Müller

- Related to task #10077: upgrade ehcache to 3.x added

#4 - 06/16/2022 09:47 PM - Andreas Müller

- Status changed from New to Resolved

- % Done changed from 0 to 70

Applied in changeset taxeditor|9ee1ca440a5bbe003ed1c34e8abcc9f76ec97533.

#5 - 06/16/2022 09:50 PM - Andreas Müller

Fixed with a workaround by overriding the class net.sf.ehcache.util.PreferredLoaderObjectInputStream which does not correctly load primitive types.

This fix can probably be removed once we move to ehcache 3.x (#10077).

#6 - 06/16/2022 09:50 PM - Andreas Müller

- Assignee changed from Andreas Müller to Katja Luther

Please review.

#7 - 06/16/2022 09:51 PM - Andreas Müller

- Related to feature request #7648: Create taxonrelation to genus or species when subordinate names are created added

#8 - 06/16/2022 09:51 PM - Andreas Müller

- Related to deleted (feature request #7648: Create taxonrelation to genus or species when subordinate names are created)

#9 - 07/08/2022 10:54 PM - Andreas Müller

- Status changed from Resolved to Closed

- Assignee changed from Katja Luther to Andreas Müller

- % Done changed from 70 to 100

I think we can close this ticket as the problem did not show up anymore.