Hadoop Common / HADOOP-217

IllegalAcessException when creating a Block object via WritableFactories

Details

Type:	Bug	Status:	CLOSED
Priority:	Major	Resolution:	Fixed
Affects Version/s:	0.3.0	Fix Version/s:	0.3.0
Component/s:	None		
Labels:	None		

Description

When I ran the dfs namenode, I received an error message listed below. Changing Block class to be public will be able to fix the problem.

java.lang.RuntimeException: java.lang.IllegalAccessException: Class org.apache.hadoop.io.WritableFactories can not access a member of class org.apache.hadoop.dfs.Block with modifiers "public"
java.lang.RuntimeException: java.lang.IllegalAccessException: Class org.apache.hadoop.io.WritableFactories can not access a member of class org.apache.hadoop.dfs.Block with modifiers "public"
at org.apache.hadoop.io.WritableFactories.newInstance(WritableFactories.java:49)
at org.apache.hadoop.io.ObjectWritable.readObject(ObjectWritable.java:226)
at org.apache.hadoop.io.ObjectWritable.readObject(ObjectWritable.java:163)
at org.apache.hadoop.io.ObjectWritable.readObject(ObjectWritable.java:211)
at org.apache.hadoop.ipc.RPC$Invocation.readFields(RPC.java:88)
at org.apache.hadoop.ipc.Server$Connection.run(Server.java:154)
Caused by: java.lang.IllegalAccessException: Class org.apache.hadoop.io.WritableFactories can not access a member of class org.apache.hadoop.dfs.Block with modifiers "public"
at sun.reflect.Reflection.ensureMemberAccess(Reflection.java:65)
at java.lang.Class.newInstance0(Class.java:344)
at java.lang.Class.newInstance(Class.java:303)
at org.apache.hadoop.io.WritableFactories.newInstance(WritableFactories.java:45)
... 5 more

Attachments

access.patch	0.5 kB	13/May/06 07:58
block.patch	3 kB	02/Jun/06 06:09

Issue Links

relates to

HADOOP-264 WritableFactory has no permissions to create DatanodeRegistration — CLOSED

Activity

Hairong Kuang created issue - 13/May/06 07:56

Hairong Kuang made changes - 13/May/06 07:58

Field	Original Value	New Value
Attachment		access.patch [12327464]

Sameer Paranjpye added a comment - 31/May/06 03:59

This does not appear to hamper normal DFS operation in any way. Looks like the Block constructor is registered as a writable factory but is not public. Does it need to be? Should Block objects ever be constructed this way?

This needs further investigation, making the constructor public just addresses the symptom, not the problem IMO. Need to figure out what part of the code is trying to create blocks through a writable factory, and should it be doing that or something different?

Sameer Paranjpye made changes - 31/May/06 04:00

Fix Version/s		0.4 [12311021]
Fix Version/s	0.3 [12310930]	

Doug Cutting added a comment - 31/May/06 04:10

I think this has been fixed, no?

The problem is that, when we modified Hadoop to use a classloader the writable factory stuff partially broke. In particular, one can now obtain a reference to a Class instance when the static initializers of that class have not yet been run. In particular, the RPC code calls something like 'WritableFactories.newInstance(ClassLoader.findClass("foo.bar.Baz")), and foo.bar.Baz's static initializer, which registers the writable factory, has not yet run, and the call to newInstance fails as above. The work-around we've been using thus far is to, in the RPC client code, add a 'static
{ new Baz(); }
' to force the static initializers of Baz to run. Long-term we should find a better solution.

We'd like to be able to pass objects in RPCs that are not public classes. This means that the RPC code needs to be able to construct an instance. The WritableFactory mechanism was created to solve this, but, as described above, it mysteriously broke when we stopped using the bootstrap classloader.

Owen O'Malley added a comment - 02/Jun/06 00:44

This is a general problem with new tools breaking because they get passed back objects back from RPC that haven't been loaded by the application.

Owen O'Malley made changes - 02/Jun/06 00:44

Link		This issue blocks ~~HADOOP-264~~ [~~HADOOP-264~~]

Owen O'Malley made changes - 02/Jun/06 00:45

Link	This issue blocks ~~HADOOP-264~~ [~~HADOOP-264~~]	

Owen O'Malley made changes - 02/Jun/06 00:45

Link		This issue relates to ~~HADOOP-264~~ [~~HADOOP-264~~]

Hairong Kuang added a comment - 02/Jun/06 01:41

The exception occurs when we restart dfs after the namenode is reformated. The namenode starts as an empty file system so the Block class is loaded. When the name node receives block reports containing blocks from the previous file system, it throws the exception.

To avoid this, we can either make the Block class public or explicitly load the class in NameNode.

Hairong Kuang made changes - 02/Jun/06 06:09

Attachment		block.patch [12334916]

Doug Cutting added a comment - 03/Jun/06 02:59

I just committed this. Thanks!

Doug Cutting made changes - 03/Jun/06 02:59

Fix Version/s		0.3 [12310930]
Fix Version/s	0.4 [12311021]	
Resolution		Fixed [1]
Status	Open [1]	Resolved [5]

Doug Cutting made changes - 06/Jun/06 06:06

Status	Resolved [5]	Closed [6]

Doug Cutting made changes - 06/Jun/06 06:17

Workflow	jira [12363100]	no reopen closed [12373125]

Doug Cutting made changes - 07/Jun/06 04:39

Workflow	no reopen closed [12373125]	no-reopen-closed [12373461]

Doug Cutting made changes - 03/Aug/06 17:46

Workflow	no-reopen-closed [12373461]	no-reopen-closed, patch-avail [12377775]

Owen O'Malley made changes - 08/Jul/09 16:41

Component/s	dfs [12310710]	

Transition	Time In Source Status	Execution Times
Doug Cutting made transition - 03/Jun/06 02:59 OPEN → RESOLVED	20d 19h 2m	1
Doug Cutting made transition - 06/Jun/06 06:06 RESOLVED → CLOSED	3d 3h 7m	1

People

Assignee:

Hairong Kuang

Reporter:

Hairong Kuang

Votes:

0 Vote for this issue

Watchers:

0 Start watching this issue

Dates

Created:

13/May/06 07:56

Updated:

08/Jul/09 16:41

Resolved:

03/Jun/06 02:59