AMQ8568 RC=2495 mqjbnd was not found, or AMQ8351 Java environment has not been configured correctly

https://www.ibm.com/support/pages/node/133265

Date last updated: 16-Feb-2023

IBM MQ Support
https://www.ibm.com/products/mq/support
Find all the support you need for IBM MQ

+++ Problem +++

This technote addresses scenarios such as:

a) You are trying to use dspmqver to display the version of additional components, either by using the option -a or the option -p:

```
dspmqver -a
dspmqver -p 7
```

You get the error:

```
AMQ8351 Java environment has not been configured correctly
```

b) You run an MQ Java or MQ JMS client application.

You get the error:

```
... mqjbnd=CC=2;RC=2495;AMQ8568: The native JNI library 'mqjbnd' was not found.
[3=mqjbnd]::/opt/mqm/java/lib/libmqjbnd.so: /opt/mqm/java/lib/libmqjbnd.so: cannot
open shared object file: No such file or directory
```

You use the listing/directory command on the file that is mentioned in the error, and the file exists and has the proper ownership and permissions:

```
$ ls -l /opt/mqm/java/lib/libmqjbnd.so
-r-xr-xr-x 1 mqm mqm 132872 Dec 14 09:41 /opt/mqm/java/lib/libmqjbnd.so
```

++ Cause ++

The following web page in the MQ documentation explains the environment variables needed to run programs that use MQ classes for Java™ and for JMS (such as dspmqver):

https://www.ibm.com/docs/en/ibm-mq/9.3?topic=java-environment-variables-relevant-mq-classes
IBM MQ / 9.3
Environment variables relevant to IBM MQ classes for Java

In the case of the mentioned AMQ8568 error, the problem is that the 64-bit library directory was NOT specified when dealing with 64-bit operating systems and instead, the following environment variable pointed to the 32-bit library file:

$ echo $MQ_JAVA_LIB_PATH
/opt/mqm/java/lib

++ Resolving The Problem ++

You need to run the command setmqenv to setup the proper MQ environment variables, and the proper setup for PATH and CLASSPATH.

Step 1: You must use "setmqenv" to setup the MQ environment variables.

In Linux and AIX you must "source" it! (That is, type a dot, then a space, then the setmqenv command and parameters).
The following use the default Installation directories.

Linux:
```
. /opt/mqm/bin/setmqenv -n Installation
```
AIX:
```
. /usr/mqm/bin/setmqenv -n Installation
```

Windows:
```
"C:\Program Files\IBM\MQ\bin\setmqenv" -n Installation1
```

Step 2: You need to specify Java 64-bit for runtime.

A common cause for 2495 is that the Java 32-bit code is executed at runtime on a 64-bit machine.
Thus, it is important to set the PATH to point to the 64-bit code, for example:
export JAVA_HOME=/usr/java5_64
export JAVA_BINDIR=$JAVA_HOME/bin
export PATH=$JAVA_BINDIR:$PATH

You can use the following command to check which "java" are you executing at runtime. The following shows that it is a 64-bit version:
% which java
/usr/java5_64/bin/java

% java -version
java version "1.5.0"
Java(TM) 2 Runtime Environment, Standard Edition (build pap64dev-20070511(SR5))
IBM J9 VM (build 2.3, J2RE 1.5.0 IBM J9 2.3 AIX ppc64-64 j9vmap6423-20070426 (JIT enabled)

Step 3: You need to specify -Djava.library.path with the proper 64-bit java libraries

The 2495 error also occurs when using the 32-bit libraries to run MQ Java programs in 64-bit. This is incorrect:
AIX: java -Djava.library.path=/usr/mqm/java/lib applicationName
Non-AIX: java -Djava.library.path=/opt/mqm/java/lib applicationName

This is correct:
AIX: java -Djava.library.path=/usr/mqm/java/lib64 applicationName
Non-AIX: java -Djava.library.path=/opt/mqm/java/lib64 applicationName

+ Note for z/OS

Ensure that you are not using trailing blanks.
For example, notice that there is a trailing blank in the 4th line in the following set of statements.

```
LIBPATH="/lib:/usr/lib:${JAVA_HOME}/bin:${JAVA_HOME}/bin/classic"
JVMJZBL2999T #LIBPATH="${LIBPATH}:/mqm/V7R0M1/java/lib"
JVMJZBL2999T #LIBPATH="${LIBPATH}:/usr/lpp/java/J6.0.1/lib"
JVMJZBL2999T LIBPATH="${LIBPATH}:/usr/lpp/mqm/V7R0M1/java/lib/ "
JVMJZBL2999T export LIBPATH
```

It was necessary to change the LIBPATH setting in the JCL to be:
LIBPATH="${LIBPATH}:/usr/lpp/mqm/V7R0M1/java/lib"

+ WebSphere Application Server when using MQ as the JMS provider

If you are seeing the exception "mqjbnd (Not found in java.library.path)" in WebSphere Application Server when using MQ as the JMS provider, see the following technotes:

https://www.ibm.com/support/pages/node/6489407
WAS error com.ibm.mq.jmqi.JmqiException RC=2495;AMQ8568: The native JNI library 'mqjbnd' was not found.

https://www.ibm.com/support/pages/node/476243
UnsatisfiedLinkError: mqjbnd (Not found in java.library.path), CC=2;RC=2495 AMQ8568: The native JNI library 'mqjbnd' was not found. [3=mqjbnd]

+ Related Information

https://www.ibm.com/support/pages/node/709281
Display the IBM MQ Version

+++ end +++