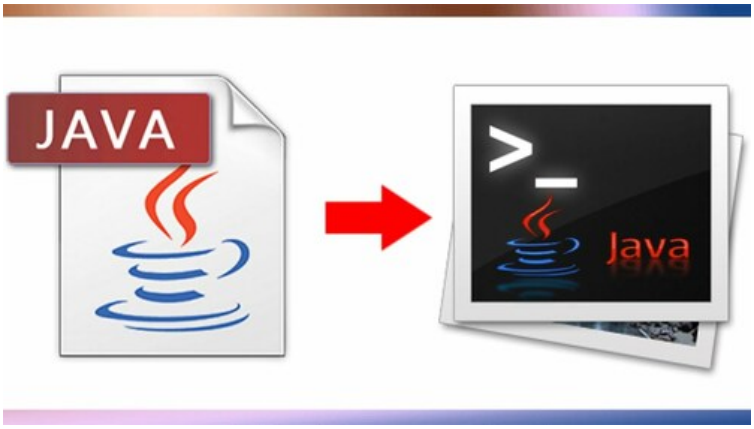
How to install jcmd on CentOS 7 to diagnose running Java Virtual Machine crashing applications

Author : admin



jcmd utility is well known in the Brane New wonderful world of Java but if you're like me a classical old school sysadmins and non-java developer you probably never heard it hence before going straight into how to install it on **CentOS 7 Linux servers**, I'll shortly say few words on what it is.
jcmd is used to send diagnostic requests to running Java **Virtual Machine (JVM)** it is available in both in ***Oracle Java*** as well as ***OpenJDK.***
The requests jcmd sends to VM are based on the running Java PID ID and are pretty useful for controlling Java Flight Recordings, troubleshoot, and diagnose JVM and Java Applications. It must be used on the same machine where the JVM is running.
Used without arguments or with the -l option, jcmd prints the list of running Java processes with their process id, their main class and their command line arguments.
When a main class is specified on the command line, **jcmd** sends the diagnostic command request to all Java processes for which the command line argument is a substring of the ***Java process***' main class.
jcmd could be useful if the **JConsole** / **JMX** (Java Management Extensions) can't be used for some reason on the server or together with **Java Visual VM (visual interface for viewing detailed info about Java App).**

In most Linux distributionsas as of year 2020 **jcmd** is found in **java-*-openjdk-headless.**

To have **jcmd** on lets say Debian GNU / Linux, you're up to something like:

apt-get install --yes openjdk-12-jdk-headless

or

apt-get install openjdk-11-jdk-headless

however in CentOS 7 jcmd is not found in **java*openjdk*headless** but instead to have it on server, thus it take me a while to look up where it is foundso after hearing
from some online post it is part of **package java*openjdk*devel*** to make sure this so true, I've used the **--download-only** option

```
yum install --downloadonly --downloaddir=/tmp java-1.8.0-openjdk-devel
```

So the next question was how to inspect the downloaded rpm package into /tmp usually, this is possible via Midnight Commander (mc) easily to view contents, however as this
server did not have installed mc due to security policies I had to do it differently after pondering a while on **how to to list the RPM package file content come up using following command**

```
rpm2cpio java-1.8.0-openjdk-devel-1.8.0.232.b09-0.el7_7.x86_64.rpm |cpio -idmv|grep -i jcmd|less
```

To then install **java-1.8.0-openjdk-devel-1.8.0.232.b09-0.el7_7.x86_64.rpm** to do so run:

yum -y install java-1.8.0-openjdk-devel-1.8.0*

Once the jcmd, I've created the following bash script that was set that was tracking for application errors and checking whether the JBoss application server **pool-available-count** is not filled up and hence jboss refuses to serve connections through **jboss-cli.sh** query automatically launching **jcmd** to **get various diagnostic data about Java Virtual Machine (e.g. a running snapshot) - think of it like the UNIX top for debugging or Windows System Monitor but run one time.**

```
# PID_OF_JAVA=$(pgrep -l java)
# jcmd $PID_OF_JAVA GC.heap_dump GC.heap_dump_file-$(date
'+%Y-%m-%d_%H-%M-%S').jfr
# jcmd $PID_OF_JAVA Thread.print > Thread.print-$(date
'+%Y-%m-%d_%H-%M-%S').jfr
```

The produced log files can then be used by the developer to visualize some Java specific stuff "Flight recordings" like in below screenshot:

If you're interested on some other interesting tools that can be used to Monitor and Debug a Running Java VM take a look at Java's official documentation Monitoring Tools.
So that's all **Mission Accomplished :)** Now the Java Application developer could observe the log and tell why exactly the application crashed after the multitude of thrown Exceptions in the ***JBoss* server.log**.