Initial AIX Java Data Collection Procedures

Table 2 (Data Collection Instructions) below identifies the minimum data collection procedures for the most common AIX Java problem areas. The commands shown will collect the minimum data required by support specialists to help understand the environment and more quickly identify the root cause of the issue. Once the data has been collected, please package the data, then send the packaged data to the AIX upload site (see AIX Data Upload section at the end of this document). In some situations, support specialists may request and require more information to further diagnose the reported issue.

The instructions in Table 2 make references to generic terms that will need to be replaced with information specific to support call and the environment. It is very important that consistent and accurate references to be used when collecting the data to ensure prompt and correct delivery of the data when uploaded.

Table 1. Terms and Definitions

Generic Term / Reference	Actions / Details
/PATH	Replace: The full path to a directory that has adequate free space to save and package the requested data.
JAVA_PID	Replace: The process id (e.g., from "ps" command) for the Java process being diagnosed.
PMR#	Replace: The PMR # assigned to your case in the format AAAAAA.BBB.CCC (e.g., 123456.789.012)
MM-DD-HH	Replace: Is the current month (MM), day (DD), and hour (HH) (e.g., 12-18-10).
JAVA_EXE	Replace: The full path to the Java executable (e.g., /usr/java6_64/jre/bin/java)
JAVA_HOME	Replace: The parent directory of Java executable (e.g., /usr/java6_64)
CORE_PATH	Replace: The full path [**including file name**] to the AIX core file (e.g., /tmp/core or /tmp/core.###.###.###.dmp).
Java core or javacore file	A text containing thread dumps generated by the JVM using "kill -3" command or automatically during exceptions
AIX core or system core file	A binary representation of the process stored to a file.
Heap dump or PHD file	A Java generated file that contains Java memory information about objects and classes
pdump.sh	A shell script provided by IBM that runs the AIX kernel debug (kdb) to obtain kernel level information
perfpmr.sh	A collection of scripts provided by IBM that runs most of AIX performance collection commands
GC or garbage collection	A component of the Java processes that manages or release memory of unused objects and classes

Table 2. Data Collection Instructions

Problem Area	Actions	Comments
Java application hang **High CPU utilization due to Java application** **Slow performance while running Java application** Java filename format are: javacore.*.txt The default file location is the directory that the Java process was started from. The '*' in the filename will be a timestamp and process id	**Download and install 'pdump.sh' tool:** Download from: ftp://ftp.software.ibm.com/aix/tools/debug/pdump.sh Install as: /opt/pdump/pdump.sh # chmod 755 /opt/pdump/pdump.sh **Download and install 'perfpmr' tool:** Download perfpmr for the release of AIX being used from: ftp://ftp.software.ibm.com/aix/tools/perftools/perfpmr/ Install using: # mkdir -p /opt/perfpmr # cd /opt/perfpmr # gzip -c < perf##.tar.Z | tar -xvf - # ./Install **Enable Java verbose garbage collection (GC):** { add this JVM option to your java command line or startup profile/script } -verbose:gc This will require the process to be restarted. **Save stderr (standard error or 2>file) for the process:** Save the standard error (stderr) for the Java process to a file if not already captured by the application. This will require the process to be restarted. This is required to capture the verbose GC data. **Collect data during hang, high CPU, slow performance:** { as "root" user while issue occurs } **# mkdir -p /PATH/PMR#/MM-DD-HH/data1** **# mkdir -p /PATH/PMR#/MM-DD-HH/perfpmr** **# cd /PATH/PMR#/MM-DD-HH/data1**	1. The "pdump.sh" script runs the AIX kernel debugger (kdb) to obtain kernel level data about the process. For a process when several hundreds threads, it may take a few minutes for the script to complete the inspection of all the threads. Because pdump.sh scans active threads, do not run this script for high CPU situations unless instructed to do so by the support specialist. 2. The "perfpmr" data to help us understand and identify the hot spots with AIX, networking, locking, JVM, application, etc. It will collect configuration data as well as run the majority of the AIX performance tools for the duration specified. 3. The AIX core (gencore and snapcore) is used to obtain application information about the active process. In order to inspect the AIX core, it can not be truncated. If the AIX core file is truncated, the following commands will have to be executed as the root user: # chuser fsize=-1 data=-1 core=-1 *user_id* # chdev -l sys0 -a fullcore=true Then Relogin as user and restart all processes, then recollect and upload the data. For J2EE Application Servers, the node agent / manager will also need to be restarted prior to restarting the application server instance. 4. If users are unable or unsure how to redirect standard error for the verbose GC data, enable the JVM option: -Xverbosegclog:/PATH/PMR#/MM-DD-HH/gc/gc.log Setting this option will require the process to be restarted. 5. The javacore files are used to obtain JVM and Java application level details. Multiple files are required to understand how/if the Java process is changing over a short period of time.

	{ ** DO NOT RUN pdump.sh for high CPU utilization situations } **# /opt/pdump/pdump.sh JAVA_PID** **# cd /PATH/PMR#/MM-DD-HH/perfpmr** **# perfpmr.sh 240** **# cd /PATH/PMR#/MM-DD-HH/data1** **# gencore JAVA_PID ../core.dmp** **# kill -3 JAVA_PID** **# sleep 5** **# kill -3 JAVA_PID** **# sleep 5** **# kill -3 JAVA_PID** { copy the javacore.*.txt files to data1 directory } { copy application log files to data1 directory } { copy stderr/stdout files to data1 directory } { identify full path to java executable } **# snapcore -d . ../core.dmp JAVA_EXE** { capture the following minimum configuration information } **# prtconf > prtconf.out 2>&1** **# lslpp -hac > lslpp-hac.out 2>&1** **# instfix -i > instfix-i.out 2>&1** **# emgr -lv3 > emgr-lv3.out 2>&1** **# JAVA_EXE -version > java-version.out 2>&1** **Package data:** **# cd /PATH/PMR#/MM-DD-HH** **# tar -cf - data1 perfpmr | gzip -c > PMR#.MM-DD-HH.tgz** **Check List (data to be captured and uploaded):** 1. pdump.sh output (except for high CPU situations) 2. perfpmr data 3. snapcore data (which contains the AIX core and libraries) 4. 3 javacore files 5. Minimum configuration information 6. Application log files which contain verbose:gc entries	
Java core dump Java filename format are: javacore.*.txt Snap.*.trc heapdump.*.phd The default file location is the directory that the Java process was started from. The '*' in the filename will be a timestamp and process id.	**Collect data:** **# mkdir -p /PATH/PMR#/MM-DD-HH/data1** { capture the following minimum configuration information } **# cd /PATH/PMR#/MM-DD-HH/data1** **# prtconf > prtconf.out 2>&1** **# lslpp -hac > lslpp-hac.out 2>&1** **# instfix -i > instfix-i.out 2>&1** **# emgr -lv3 > emgr-lv3.out 2>&1** **# svmon -O segment=category -P > svmon-p.out 2>&1** **# ipcs -saPrX > ipcs.out 2>&1** **# errpt -a > errpt-a.out 2>&1** **# JAVA_EXE -version > java-version.out 2>&1** { copy the javacore.*.txt files to data1 directory } { copy any heapdump.*.phd files to data1 directory } { copy any Snap.*.trc files to the data1 directory } { copy application log files to data1 directory } **Package data:** **# cd /PATH/PMR#/MM-DD-HH** **# tar -cf - data1 | gzip -c > PMR#.MM-DD-HH.tgz** **Check List (data to be captured and uploaded):** 1. All javacore files 2. Any heapdump.*.phd or Snap.*.trc files 3. Minimum configuration information 4. Application log files	1. Once the data was generated/collected, look for the line at the top with 1TISIGINFO. It will be similar to: 1TISIGINFO Dump Event "user" (00004000) received 2. Identify the source of the exception as being "user generated" (e.g., from kill -3 command), "gpf" (AIX core / process dump), or "Out Of Memory", then collect the additional data for that scenario. 3. If segmentation violation/abort/gpf, then collect data for AIX core file, validate AIX core is not truncated, and collect snapcore. 4. If OutOfMemory, follow out of memory instructions 5. Look for CORE_DUMP labels in the error log (errpt -a output) to identify if an AIX (process/system) core has been generated for the process, the time of the process core, and the location of the process core. If an AIX (process/system) core has been generated, also follow the data collection procedures for the section "**AIX core dump**" below.

AIX core dump (a.k.a. process dump / system dump) Java filename format are: core.*.dmp javacore.*.txt heapdump.*.phd Snap.*.trc The default file location is the directory that the Java process was started from. The '*' in the filename will be a timestamp and process id. Look for CORE_DUMP labels in the error log (errpt -a output) to identify if an AIX (process) core has been generated for the process, the time of the process core, and the location of the process core.	**Collect data:** **# mkdir -p /PATH/PMR#/MM-DD-HH/data1** { capture the following minimum configuration information } **# cd /PATH/PMR#/MM-DD-HH/data1** **# prtconf > prtconf.out 2>&1** **# lslpp -hac > lslpp-hac.out 2>&1** **# instfix -i > instfix-i.out 2>&1** **# emgr -lv3 > emgr-lv3.out 2>&1** **# svmon -O segment=category -P > svmon-p.out 2>&1** **# ipcs -saPrX > ipcs.out 2>&1** **# errpt -a > errpt-a.out 2>&1** **# JAVA_EXE -version > java-version.out 2>&1** { copy the javacore.*.txt files to data1 directory } { copy any Snap.*.trc files to data1 directory } { copy any heapdump.*.phd files to data1 directory } { copy application log files to data1 directory } **# snapcore -d . CORE_PATH JAVA_EXE** **Package data:** **# cd /PATH/PMR#/MM-DD-HH** **# tar -cf - data1 | gzip -c > PMR#.MM-DD-HH.tgz** **Check List (data to be captured and uploaded):** 1. snapcore data (which contains the AIX core and libraries) 2. All javacore files 3. Any heapdump.*.phd or Snap.*.trc files 4. Minimum configuration information 5. Application log files	1. The AIX core (gencore and snapcore) is used to obtain application information about the active process. In order to inspect the AIX core, it can not be truncated. If the AIX core file is truncated, the following commands will have to be executed as the root user: # chuser fsize=-1 data=-1 core=-1 *user_id* # chdev -l sys0 -a fullcore=true Then Relogin as user and restart all processes, then recollect and upload the data. For J2EE Application Servers, the node agent / manager will also need to be restarted prior to restarting the application server instance. 2. For POWER7 systems have minimum required Java versions depending on if POWER6 or POWER7 mode is used. If an unsupported (old) version of Java is used, the JVM will crash, usually on startup. To confirm, visit the web page: http://www.ibm.com/developerworks/java/jdk/power7/index.html to view the minimum Java versions for each POWER mode. Then compare that information with the information provided in the prtconf.out (POWER mode) and java-version.out (Java version). If the level of Java is not at the minimum level, the JVM must be upgraded to a more recent version.
Java download **Java installation** **Java security alerts** **Java version support**	**Primary download page:** http://www.ibm.com/developerworks/java/jdk/aix/service.html **Power7 compatibility page:** http://www.ibm.com/developerworks/java/jdk/power7/index.html **Power8 feature support page:** http://www.ibm.com/developerworks/java/jdk/power8/index.html **Jave security alerts:** http://www.ibm.com/developerworks/java/jdk/alerts/ http://www-01.ibm.com/support/docview.wss?uid=swg21687173 **Jave 1.4.x end of service/support statement:** http://www.ibm.com/developerworks/java/jdk/aix/outofservice.html	All of the java downloads are available from the main download page. Customers can download the latest version or install the base version + a specific SR to install a specific level. If customer is running Power7 and/or Power8, the JVM may have to be at minimum levels otherwise the processes may crash. Java 1.4.x is out of support and there is no extended support. If the customer is entitled another product that includes 1.4.x, please forward PMR to that team.
Security Issue	**Enable Java Security Debug Parameters** { add these JVM options to your application startup profile/script } -Djavax.net.debug=all -Dcom.ibm.security.jgss.debug=all -Dcom.ibm.security.krb5.Krb5Debug=all This will require the process to be restarted. **Save stderr (standard error or 2>file) for the process:** Save the standard error (stderr) for the Java process to a file if not already captured by the application. This will require the process to be restarted. This is required to capture the debug Java security information. **Collect data:** **# mkdir -p /PATH/PMR#/MM-DD-HH/data1** { capture the following minimum configuration information } **# cd /PATH/PMR#/MM-DD-HH/data1** **# prtconf > prtconf.out 2>&1** **# lslpp -hac > lslpp-hac.out 2>&1**	Java security issues can be complex, difficult, and time consuming. To expedite the resolution of your Java security issue, it is recommended that customers provide a complete and standalone sample program (including execution instructions) that demonstrates the issue. In some situations, customers may be provided an IBM JVM for another platform (such as Linux or Microsoft Windows) to confirm if the issue applies across all IBM JVM implementations or is specific to the IBM AIX JVM. Additional data collection procedures are available on the web page: http://www.ibm.com/support/docview.wss?uid=swg21162961

	# instfix -i > instfix-i.out 2>&1 **# emgr -lv3 > emgr-lv3.out 2>&1** **# errpt -a > errpt-a.out 2>&1** **# JAVA_EXE -version > java-version.out 2>&1** **# cd JAVA_HOME/jre/lib/security** **# tar –cvf /PATH/PMR#/MM-DD-HH/data1/secfiles.tar ./** **Package data:** **# cd /PATH/PMR#/MM-DD-HH** **# tar -cf - data1 | gzip -c > PMR#.MM-DD-HH.tgz** **Check List (data to be captured and uploaded):** 1. Minimum configuration information 2. Application log files which contain debug Java entries	
Memory Issue **Out Of Memory** **System paging** **High system memory usage** Java filename format are: heapdump.*.phd javacore.*.txt The default file location is the directory that the Java process was started from. The '*' in the filename will be a timestamp and process id.	**Enable Java verbose garbage collection:** { add these JVM options to your application startup profile/script } -verbose:gc -Xdump:heap:events=systhrow,filter=java/lang/OutOfMemoryError This will require the process to be restarted. **Save stderr (standard error or 2>file) for the process:** Save the standard error (stderr) for the Java process to a file if not already captured by the application. This will require the process to be restarted. This is required to capture the verbose GC data. **Collect data:** **# mkdir -p /PATH/PMR#/MM-DD-HH/data1** **# cd /PATH/PMR#/MM-DD-HH/data1** { if process is still active or run prior to out of memory } **# vmstat -lt 1 20 > vmstat.out 2>&1 &** **# svmon -O segment=category -P > svmon-p.out 2>&1** **# ipcs -saPrX > ipcs.out 2>&1** **# ps avwwwg > ps-avwwwg.out 2>&1** **# gencore JAVA_PID ../core.dmp** { copy the javacore.*.txt files to data1 directory } { copy the java headdump.*.phd files to data1 directory } { copy the Snap.*.trc files to data1 directory } { copy application log files to data1 directory } { identify full path to java executable } **# snapcore -d . ../core.dmp JAVA_EXE** { capture the following minimum configuration information } **# prtconf > prtconf.out 2>&1** **# lslpp -hac > lslpp-hac.out 2>&1** **# instfix -i > instfix-i.out 2>&1** **# emgr -lv3 > emgr-lv3.out 2>&1** **# JAVA_EXE -version > java-version.out 2>&1** **Package data:** **# cd /PATH/PMR#/MM-DD-HH** **# tar -cf - data1 | gzip -c > PMR#.MM-DD-HH.tgz** **Check List (data to be captured and uploaded):** 1. snapcore data (which contains the AIX core and libraries) 2. performance data (e.g., vmstat, svmon, etc) 3. All javacore files 4. Any heapdump.*.phd or Snap.*.trc files 5. Minimum configuration information 6. Application log files which contain verbose:gc entries	The primary types of memory issues are: 1. Running out of Java heap 2. Running out of Native heap 3. Overcommitted on system memory 1. The AIX core (gencore and snapcore) is used to obtain application information about the active process. In order to inspect the AIX core, it can not be truncated. If the AIX core file is truncated, the following commands will have to be executed as the root user: # chuser fsize=-1 data=-1 core=-1 *user_id* # chdev -l sys0 -a fullcore=true Then Relogin as user and restart all processes, then recollect and upload the data. For J2EE Application Servers, the node agent / manager will also need to be restarted prior to restarting the application server instance. 2. The Xdump option enables the JVM to generate a heapdump file that can be analyzed using the IBM Health Center tool. 3. In the javacore file (a text file), look for: The amount of used Java heap, JIT code cache, JIT data cache, and Shared Class cache. 4. If its a 32bit process and there is svmon data, check the "working storage" segments. There is a high probably that LDR_CNTRL may need to be changed to something similar to: LDR_CTRNL=MAXDATA=0xB0000000@DSA 5. If there is potential native memory issue, do: a. Add JVM option: {the following is one option, no spaces} -Xdump:system:events=systhrow, filter=java/lang/OutOfMemoryError, request=exclusive+prepwalk b. AIX environment: MALLOCDEBUG=log:extended,stack_depth:12 c. When OutOfMemory (OOM) occurs, collect AIX core, snapcore, etc. 5. If customer is unable or unsure how to redirect standard error for the verbose GC data, enable the JVM option: -Xverbosegclog:/PATH/PMR#/MM-DD-HH/gc/gc.log Setting this option will require the process to be restarted.

More detailed information on AIX procedures is available by accessing these web pages:

IBM Support Handbook
ftp://ftp.software.ibm.com/software/server/handbook/webhndbk.pdf

AIX MustGather
http://www-01.ibm.com/support/docview.wss?uid=aixtools-5041a981

AIX Support Center Tools
http://www.ibm.com/support/aixtools

AIX Support and Service Page
http://www-03.ibm.com/systems/power/software/aix/service.html

AIX Data Upload Instructions and Sites
HTTP (Secure) Sites
http://www.ecurep.ibm.com/app/upload
http://testcase.boulder.ibm.com
FTP Sites
http://www-05.ibm.com/de/support/ecurep/send_ftp.html
ftp://testcase.boulder.ibm.com

Testcase files can manually be uploaded using the FTP command by following these instructions. Replace FILE_TO_UPLOAD with the name of the file to be uploaded. The full PMR# must be included in the filename (e.g., 12345.567.000.data.tar).

```
ftp testcase.boulder.ibm.com
login: ftp
password: ftp
cd /toibm/aix
bin
put FILE_TO_UPLOAD
quit
```

If you experience issues uploading data using one method, please try one of the other available methods. Also, here is a reference document for troubleshooting connection issues to ftp://testcase.boulder.ibm.com (/toibm/aix):

http://techsupport.services.ibm.com/390/ftp_pdpsi_ext.html

Last Change: 03-31-2015