Defunct Process Problem Determination

Special Notices

Information in this document is correct to the best of our knowledge at the time of this writing. Please send feedback by fax to "AIXServ Information" at (512) 823-4009.

Please use this information with care. IBM will not be responsible for damages of any kind resulting from its use. The use of this information is the sole responsibility of the customer and depends on the customer's ability to evaluate and integrate this information into the customer's operational environment.

About this Document

This document explains how defunct processes may occur, how to determine their cause, and what information to provide to software vendors if further problem determination is necessary.

This document applies to all levels of AIX 3.1 through 4.1.

Introduction

The AIX Operating System implements a hierarchy of processes in which each process has a parent process. A process which terminates may notify the parent process and the parent process may collect status information for that child. When the parent process fails to collect the status information in a timely manner a "defunct" process may remain.

Because a "defunct" process is already dead, killing the process will not have any effect. The defunct process does not use any system resources, such as CPU or disk, and uses a minimum amount of memory for storing the exit status and resource usage information.

Problem Determination

The parent of a child process is responsible for reaping (removing) the child process when it terminates. You can find the parent process by running the "ps -ef" command. The "PPID" column gives the process ID of the parent.

If the PPID is "1"

If the PPID for the defunct process is "1", the parent is the "init" process. The "init" process is the ancestor of all processes on the system. Under normal conditions the "init" process reaps all defunct process with a PPID of 1.

If the parent process ID is "1", you must determine if the defunct process has been dead for more than a few minutes. (To do this, wait a few minutes and then see if the process is still there.) It is acceptable for defunct processes owned by init to exist for one or two minutes, particularly on heavily loaded systems. Defunct processes are frequently created by complex shell scripts as a result of the way the command shells are designed. This continual stream of defunct processes is normal and does not indicate a problem.

A problem may occur when init has not completed processing /etc/inittab and is waiting on a specific entry (most commonly, an /etc/rc script) to terminate. During this time, init will ignore all other dead child processes and wait for the specific child process to die. The most common indication that a hung /etc/inittab task is causing the problem is that the number of defunct processes owned by init grows without bound. In AIX 4.1.3 and greater, init has been enhanced to effectively handle defunct processes, even while processing inittab wait entries. This function is available in AIX 3.2.5 as well via the installation of APAR IX50480.

To determine if init has finished processing /etc/inittab, find the entries in /etc/inittab which have a third field of "wait" or "once". For example:

```
qdaemon:2:wait:/bin/startsrc -sqdaemon
```

For 3.1-3.2:

For those entries, match the first field ("qdaemon" in the above example) with the "id=" field in the "who -a" output to see if those processes have exited. The "who -a" output will have entries of the form

```
Name   ST Line   Time      Activity  PID Hostname/Exit
.         .   May 21 18:19   old   9354 id=qdaemon term=0 exit=0
```

If the process has exited, "exit=" will be shown in the "who -a" output. If it has not exited, there will be no "exit=" field for that entry in "who -a". In the above example, the qdaemon process (which has "wait" for the third field in /etc/inittab) has exited with a status of "0".

If an entry with "wait" as the action does not have an "exit=" value, that entry has become hung and must exit or be killed before init will begin reaping dead children, unless APAR ix50480 has been installed.

For 4.1:

For those entries, match the first field ("qdaemon" in the above example) with the 6th field of the who -d output. Check each "wait" entry in the inittab file, they should have a entry in the who -d output. If there is no entry in the output, this means the process did not finish. In this example qdaemon has finished so it is not the problem process

```
.    .    Sep 20 10:01    old       6536 id=qdaemon term=0 exit=0
```

Look for a process that should start from inittab but never made it to the who -d output. If an entry with "wait" as the action does not have a entry, the process has become hung and must exit or be killed before init will begin reaping dead children. If on level 4.1.3 or greater, however, the dead children will continue to be reaped.

Note: if the /etc/inittab file has a entry like:

```
install_assist:2:wait:/usr/lib/lpd/pio/etc/pioinit
> /dev/null 2>&1
```

It should be removed. This should have been removed by the Installation Assistant after the initial installation. This process will hang and cause the defunct processes.

If the PPID is Not "1"

If the parent process ID is other than "1", the process referenced by that PID is responsible for reaping the dead child process.

One cause of unreaped child processes is shell pipelines. The shell is implemented such that a pipeline causes commands to have a parent-child relationship. Many of the commands on the system do not expect to have children because they produce no children of their own. These processes are then unprepared to handle the responsibilities of a dead child process. This is particularly visible when a shell pipeline includes one or more short-lived processes and one or more long-lived processes. The short-lived process will die and be inherited by its potentially long-lived parent. This defunct process will be present as long as the parent process is still running.

Any program which creates child processes of its own should accept responsibility for the child processes when they die. When init must process "orphaned" defunct processes, the load on the system is increased, because init must perform additional work to determine if the dead process was a child of init. These programs should be coded to handle dead child processes. If programs do not do so, this should be reported to their respective software vendors as a potential defect.

If Further Problem Determination is Needed

Should you determine that a process is defunct and that the process should reasonably have been reaped by the parent, the following information will probably be required by the software vendor of the program in order to analyze the cause of the problem.

- The output of the "ps -ef" command.
- The output of the "who -a" command.
- The /etc/inittab file.
- The output of the following, where "<PPID>" is the parent process ID of the defunct process. Run the following:

```
SLOT= expr <PPID> / 256
(echo u $SLOT ; echo trace -k $SLOT) | crash  >  <filename>
```

  where "<filename>" is the file to which the output should be written.

02/01/96, 4FAX# 1771

Reader's Comments

Please fax this form to (512) 823-4009, attention "AIXServ Information". You may also e-mail comments to: elizabet@austin.ibm.com. These comments should include the same customer information requested below.

Use this form to tell us what you think about this document. If you have found errors in it, or if you want to express your opinion about it (such as organization, subject matter, appearance) or make suggestions for improvement, this is the form to use.

If you need technical assistance, contact your local branch office, point of sale, or 1-800-CALL-AIX (for information about support offerings). These services may be billable. Faxes on a variety of subjects may be ordered free of charge from 1-800-IBM-4FAX. Outside the U.S. call 415-855-4329 using a fax machine phone.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Note: If you have a problem report or item number, supplying that number may help us determine why a procedure did or did not work in your specific situation.

Problem Report or Item #: Branch Office or Customer #:

Be sure to print your name and fax number below if you would like a reply:

Name: Fax Number:

END OF DOCUMENT (defunct.procs.32-41.cmd, 4FAX# 1771)