LD_LIBRARY_PATH in v20: xsltproc causes segmentation fault

- **Article Type:** General
- **Product:** Aleph

Description:
We have a custom procedure (p_custom_12) that runs fine under V18, but under V20 it is causing a Segmentation Fault.
The problem statement is (for example)
(in /home/aleph/r3) /usr/bin/xsltproc -o gmdtest flat_file.xsl normal_xml.20091207.1010

This is an XSL file that converts an Aleph printout file into a flat file for further manipulation.

Our sysadmin figured out that the problem is that the variable LD_LIBRARY_PATH references /exlibris/aleph/a20_2/product/local/libxml/lib/ before /usr/bin/. Here's what he reported:

I think see the problem ....

it's with the LD_LIBRARY_PATH environment variable .... as part of the
aleph login's.

>>echo $LD_LIBRARY_PATH
/exlibris/aleph/a20_2/product/local/cobol/coblib:/exlibris/aleph/a20_2/product/lib:/exlibris/aleph/a20_2/product/local/perl/lib
:/exlibris/aleph/a20_2/product/local/libxml/lib:/exlibris/aleph/a20_2/product/local/openssl/lib:/usr/sfw/lib
:/exlibris/aleph/a20_2/product/local/gcc/lib:/lib:/usr/lib:/usr/ucblib:/exlibris/app/oracle/product/11/lib32
:/exlibris/app/oracle/product/11/ctx/lib:/exlibris/aleph/a20_2/product/local/java/jre/lib/sparc:/exlibris/aleph/a20_2/product/local/java/jre/lib/sparc/server

with the above environment set xsltproc drops core...

if I clear out LD_LIBRARY_PATH it works fine...

the problem seems to be in the lib's are being found inside the aleph
install.

with normal alpeh login & LD_LIBRARY_PATH
>>ldd /usr/bin/xsltproc
libxslt.so.1 => /usr/lib/libxslt.so.1
libexslt.so.0 => /usr/lib/libexslt.so.0
libxml2.so.2 =>
/exlibris/aleph/a20_2/product/local/libxml/lib/libxml2.so.2
libxml2.so.2 (SUNW_1.4) => (version not found)
libz.so.1 => /usr/lib/libz.so.1
libpthread.so.1 => /lib/libpthread.so.1
libsocket.so.1 => /lib/libsocket.so.1
libnsl.so.1 => /lib/libnsl.so.1
libm.so.2 => /lib/libm.so.2

libc.so.1 => /lib/libc.so.1
libxml2.so.2 (SUNW_1.5) => (version not found)
libxml2.so.2 (SUNW_1.4) => (version not found)
libdl.so.1 => /lib/libdl.so.1
libm.so.1 => /lib/libm.so.1
libgcc_s.so.1 => /usr/sfw/lib/libgcc_s.so.1
libmp.so.2 => /lib/libmp.so.2
libmd5.so.1 => /lib/libmd5.so.1
libscf.so.1 => /lib/libscf.so.1
libdoor.so.1 => /lib/libdoor.so.1
libuutil.so.1 => /lib/libuutil.so.1
/platform/SUNW,Sun-Fire-880/lib/libc_psr.so.1
/platform/SUNW,Sun-Fire-880/lib/libmd5_psr.so.1

no LD_LIBRARY_PATH
>>ldd /usr/bin/xsltproc
libxslt.so.1 => /usr/lib/libxslt.so.1
libexslt.so.0 => /usr/lib/libexslt.so.0
libxml2.so.2 => /usr/lib/libxml2.so.2
libz.so.1 => /usr/lib/libz.so.1
libpthread.so.1 => /usr/lib/libpthread.so.1
libsocket.so.1 => /usr/lib/libsocket.so.1
libnsl.so.1 => /usr/lib/libnsl.so.1
libm.so.2 => /usr/lib/libm.so.2
libc.so.1 => /usr/lib/libc.so.1
libmp.so.2 => /lib/libmp.so.2
libmd5.so.1 => /lib/libmd5.so.1
libscf.so.1 => /lib/libscf.so.1
libdoor.so.1 => /lib/libdoor.so.1
libuutil.so.1 => /lib/libuutil.so.1
/platform/SUNW,Sun-Fire-880/lib/libc_psr.so.1
/platform/SUNW,Sun-Fire-880/lib/libmd5_psr.so.1

i'm not sure if anything would break if you just simply re-ordered the
library paths, having /lib and /usr/lib before
/exlibris/aleph/a20_2/product/local/libxml/lib/

========== end of sysadmin quote ==========

For the moment, I've added
setenv LD_LIBRARY_PATH ""
to p_custom_12, to limit its effect to just that procedure. Is there a better fix? Is there an underlying problem that needs to be fixed?

Resolution:
Ex Libris writes as follows:

1. The Aleph libs always come before any system libs and therefore /lib and /usr/lib should be *after* /exlibris/aleph/a20_2/product/local/libxml/lib/ , as they were.

2. Is xsltproc running from an Aleph procedure? - if so it should be using $aleph_product/local/libxslt/bin/xsltproc and not the system /usr/bin/xsltproc command

3. If this is a standard Aleph procedure and our application do use the /usr/bin/xsltproc then this should be reported to the Aleph dev to correct.

<end Ex Libris>

I see that you have the following line in the p_custom_12:

/usr/bin/xsltproc -o flatfile.${TODAY}.${TIME} flat_file.xsl normal_xml.${TODAY}.${TIME}

Please specify the following instead:

$aleph_product/local/libxslt/bin/xsltproc -o flatfile.${TODAY}.${TIME} flat_file.xsl normal_xml.${TODAY}.${TIME}

[From site:]

Actually, you have our problem backwards. This is a locally written procedure that, under V18, worked fine with the default xsltproc, but now causes a core dump under V20. I think this answers my question, though. For our locally written procedures that need the standard xsltproc instead of the Aleph xsltproc, we'll refine the variables only for the duration of that execution.

- **Article last edited:** 10/8/2013