110

AIX update

December 2004

In this issue

A tc PUBLICATION

AIX Update

Published by

Xephon Inc
PO Box 550547
Dallas, Texas 75355
USA

Phone: 214-340-5690
Fax: 214-341-7081

Editor

Trevor Eddolls
E-mail: trevore@xephon.com

Publisher

Bob Thomas
E-mail: info@xephon.com

Subscriptions and back-issues

A year's subscription to *AIX Update*, comprising twelve monthly issues, costs $275.00 in the USA and Canada; £180.00 in the UK; £186.00 in Europe; £192.00 in Australasia and Japan; and £190.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the November 2000 issue, are available separately to subscribers for $24.00 (£16.00) each including postage.

AIX Update on-line

Code from *AIX Update*, and complete issues in Acrobat PDF format, can be downloaded from our Web site at http://www.xephon. com/aix; you will need to supply a word from the printed issue.

Disclaimer

Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, scripts, and other contents of this journal before making any use of it.

Contributions

When Xephon is given copyright, articles published in *AIX Update* are paid for at the rate of $160 (£100 outside North America) per 1000 words and $80 (£50) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of $32 (£20) per 100 lines. To find out more about contributing an article, without any obligation, please download a copy of our *Notes for Contributors* from www.xephon.com/nfc.

Printed in England.

Finding the biggest file in directories on the same filesystem

Sometimes I need to find the largest file in a directory and all its subdirectories. Finding the biggest file is easy:

```
ls -alR|sort -nu +4n
```

The biggest file is the last one in the list.

Sometimes, however, the biggest file may reside in that directory, but not in the same volume group.

Somewhere below the */usr/* directory there could be a mounted filesystem, like */usr/sap/trans*.

The directory */usr* may reside on the filesystem */dev/hd2*, but */usr/sap/trans* resides on */dev/lvtrans*.

If I am now looking for the biggest file under */usr*, I might find the biggest file, but it's not on the same filesystem as */usr*.

I need to exclude certain directories below */usr* to find genuinely the biggest file on the same filesystem:

```
ls|while read x; do print $x|grep -v "sap"; done|xargs ls -alR| sort -nu
+4n|tail
```

This script excludes any filenames containing the string 'sap'. If more than just the 'sap' subdirectory needs to be excluded, this line can be expanded:

```
ls|while read x; do print $x|grep -Ev "sap|trans"; done|xargs ls -alR|
sort -nu +4n|tail
```

This script runs perfectly on:

- AIX 5.1.0 ML 03
- AIX 5.2.0 ML 03.

Robert Kaiser
Systems Analyst
Bayerischer Rundfunk (Germany)

Creating a physical partition map of an hdisk by logical volume

The following code can be used to create a physical partition map of an hdisk by logical volume.

MKDISKMAP.KSH

```
#!/bin/ksh
#
# Create a disk map of PPs and display on screen.
# Simulates the manual labour performed to create
# the same map in Excel.
#
# Original created 1/7/2004 by Bill Verzal
#
# Check command line args
#
if [ "$#" -eq "0" ] ; then
   echo "Please specify one or more hdisks"
   exit 1
fi
#
i01="         1         2         3         4         5         6"
i02="123456789012345678901234567890123456789012345678901234567890"
i03="ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz"
#
# row 1 = 1-60
# row 2 = 61-120
# and so on
#
# Create an array of single characters for display purposes
#
charsInArray='echo "$i03"|wc -c|awk {'print $1'}'
charsInArray='expr $charsInArray - 1'
arrayCNT=1

while : ; do
   if [ "$arrayCNT" -gt "$charsInArray" ] ; then
      break
   fi
   charArray[${arrayCNT}]='echo "$i03"|cut -c ${arrayCNT}'
   arrayCNT='expr $arrayCNT + 1'
done
```

```
hdisks="$@"

for hdisk in $hdisks ; do

   if [ ! -r /dev/${hdisk} ] ; then
      echo "Invalid disk ($hdisk) specified - skipping"
      continue
   fi

   echo "Partition map for $hdisk:"
   totalPPs='lspv $hdisk|grep TOTAL|awk {'print $3'}'
   echo "$totalPPs PPS total on disk"
   x=0

   while [ "$x" -le "$totalPPs" ] ; do
      ppArray[${x}]="."
      x='expr $x + 1'
   done

   LVs1='lspv -l $hdisk|awk {'print $1'}'
   LVsCount='echo "$LVs1"|wc|awk {'print $1'}'
   LVsCount2='expr $LVsCount - 2'
   LVs2='echo "$LVs1"|tail -$LVsCount2'
   charcounter=0
   key="Unused: .\n"

   for LV in $LVs2 ; do

      charcounter='expr $charcounter + 1'
      LVChar="${charArray[${charcounter}]}"
      key="${key}${LV}: $LVChar\n"

      for pp in 'lslv -m $LV|grep -v LP|awk {'print $2'}' ; do
         ppArray[${pp}]="$LVChar"
      done

   done

   echo "$i01"
   echo "$i02"

   a=1
   b=1
   while [ "$a" -le "$totalPPs" ] ; do
      echo "${ppArray[${a}]}\c"
      a='expr $a + 1'
      b='expr $b + 1'
      if [ "$b" -gt "60" ] ; then
         echo " "
         b=1
      fi
   done

   echo "\n"

   echo "$key"

done

exit 0
```

MKHTMLDISKMAP.KSH

Below is a modification to the script. It creates HTML formatted output.

```
#!/bin/ksh
#
# Create an HTML disk map of PPs
# Simulates the manual labour performed to create
# the same map in Excel.
#
# Original created 1/7/2004 by Bill Verzal
#
# For best results, redirect output to a .htm/.html file.
#
# This version modified from original by
# Frank Chiavoni <Frank.Chiavoni@OIT.STATE.NJ.US>
#
#
# Check command line args
#
if [ "$#" -eq "0" ] ; then
   echo "Please specify one or more hdisks"
   exit 1
fi
#
i01="         1         2         3         4         5         6"
i02="123456789012345678901234567890123456789012345678901234567890"
# ### colors="#f2c8d0 #2d970b #2e0c7f #6b8a76 #45c53a #37dfb1 #b49618 \
# ###  #0507d6 #32525f #b42363 #84ed57 #c359b7 #103660 #45ec3d #324565 \
# ###  #940964 #51b538"
# colors=#xxyyzz where xxyyzz are corresponding values for RGB shades
# ### colors="#ff0000 #00ff00 #0000ff #ff9393 #93ff93 #9393ff \
# ###         #930000 #009300 #00f6ff #ffff10 #ff10ff #10ffff \
# ###         #808080 #cacaca #5078b4 #c37350 #96ffaf #009024"
colors="#500000 #505000 #005000 #005050 #000050 #500050 \
        #a70000 #a7a700 #00a700 #00a7a7 #0000a7 #a700a7 \
        #ff0000 #ffff00 #00ff00 #00ffff #0000ff #ff00ff \
        #508080 #505080 #a78080 #a7a780 #ff8080 #ff0080 \
        #500080 #a70080 #ff8000"

#
# row 1 = 1-60
# row 2 = 61-120
# and so on
#
# Create an array of single characters for display purposes
#

arrayCNT=1
for colour in $colors
do
        colorArray[${arrayCNT}]=$colour
        arrayCNT='expr $arrayCNT + 1'
done

hdisks="$@"

echo  "<html><head><title>mdiskmap</title></head><body>"

for hdisk in $hdisks ; do

##Didn't run as root so removed...
   if [ ! -r /dev/${hdisk} ] ; then
        :
      ##echo "Invalid disk ($hdisk) specified - skipping"
      ##continue
   fi

   echo "<font size=3 color=red><b>Partition map for
$hdisk:</b></font><br>"
   totalPPs='lspv $hdisk|grep TOTAL|awk {'print $3'}'
   echo "<font size=3 color=red>$totalPPs PPS total on disk</font><p>"
   x=0

   defaultColor="#000000"
   while [ "$x" -le "$totalPPs" ] ; do
      ppArray[${x}]=$defaultColor
      x='expr $x + 1'
   done

   LVs1='lspv -l $hdisk|awk {'print $1'}'
   LVsCount='echo "$LVs1"|wc|awk {'print $1'}'
   LVsCount2='expr $LVsCount - 2'
   LVs2='echo "$LVs1"|tail -$LVsCount2'
   charcounter=0
    key="<tr><td><font><b><i>Unused</i></b></font></td><td
```

```
bgcolor=\"${defaultColor}\"><font
color=\"${defaultColor}\">X....X</font></td></tr>"

   for LV in $LVs2 ; do

      charcounter='expr $charcounter + 1'
      LVColor="${colorArray[${charcounter}]}"
      key="${key}<tr><td><font><b><i>${LV}</i></b></font></td><td
bgcolor=\"${LVColor}\"><font  color=\"${LVColor}\">X</font></td></tr>"

      for pp in 'lslv -m $LV|grep -v LP|awk {'print $2'}' ; do
         ppArray[${pp}]="$LVColor"
      done

   done

   echo "<table><tr>"

   a=1
   b=1
   while [ "$a" -le "$totalPPs" ] ; do
      echo "<td bgcolor=\"${ppArray[${a}]}\"><font
color=\"${ppArray[${a}]}\">X</font></td>"
      a='expr $a + 1'
      b='expr $b + 1'
      if [ "$b" -gt "60" ] ; then
         echo "</tr>"
         b=1
      fi
   done

   echo "</tr></table>"

   echo "<p><p><table><tr><td colspan=2><font color=red><b>Color
Codes</b></td></tr>"
   echo "$key</table>"

done

echo "</body></html>"

exit 0
```

Bill Verzal
Project Leader
Komatsu America (USA)

AIX console mirroring

How many times during a support call have you wished you had some sort of software on the AIX server that would allow you to see what the person on the other end of the phone is unsuccessfully trying to describe, but you don’t have any shadowing software installed on the server. Well, AIX comes with a native utility called **portmir** that will allow you to perform console mirroring.

The command **portmir** is available on all AIX levels starting with AIX 4.2.1 and it is part of the *bos.sysmgt.serv_aid* fileset:

```
# whence portmir
/usr/sbin/portmir
# lslpp -w /usr/sbin/portmir
 File                                     Fileset               Type
  ----------------------------------------------------------------------
  /usr/sbin/portmir                       bos.sysmgt.serv_aid   File
```

Before using the **portmir** utility, you have to note the following:

- Make sure that the terminal type (set by the environment variable TERM) on the monitor or source console is the same as the one on the target console that you are trying to connect to. For example, if connecting to an IBM3151 or VT100, make sure you are using or emulating an IBM3151 or VT100. If the TERM types are not the same, you will get garbage and strange characters on your screen when you connect.
- The **portmir** command is supported for use with tty, pts, and lft devices only.
- Only a single mirror session may be running at any one time.
- A file, *$HOME/.mir*, contains a list of users who are authorized to monitor the port. This is not required for the root user. When the mirror daemon starts, it checks who is

on that port and then checks whether the user is authorized to monitor that port.

- If a user **su**s to the root user during a mirroring session, then both users will have root authority.

The syntax for the **portmir** command is shown below (the **-t** flag is used to specify the target terminal):

```
portmir  -t  {target_terminal}
```

For example, if connecting to pts/8 from your terminal:

```
# portmir  -t  pts/8
```

To start mirroring to target pts/8 from monitor pts/3 while you are on a different terminal:

```
# portmir -t pts/8 -m pts/3
```

Once connected, the following message will appear on both terminals:

```
portmir: Remote user connected, mirroring active.
```

If the user is in an application screen, they may need to refresh the screen by pressing CTRL-L to make the message disappear. At this point, the terminal mirroring is established and you will be able to see what the user is doing and you will be able to take over their session at any time. Both terminals will be able to use their keyboards.

There are two ways to stop the mirroring. If you are mirroring to a network connection (pts), the user will have to exit their session to drop the mirroring. Alternatively, from command line, type:

```
portmir -o
```

For example to disconnect from pts/8, type on the command line of the target or the monitoring terminal:

```
# portmir -o
```

The following message will appear on both terminals:

```
portmir: Mirroring is stopped.
```

This command can be used to connect to a tty that is at its login prompt, but in order to stop the mirroring, you will have to login and type:

```
> 'portmir -o ; exit'
```

from the command line in order to disconnect the mirror and get the terminal back to its login prompt.

For more information about the **portmir** utility please refer to the AIX command reference guide or the man pages.

Basim Chafik
Senior Systems Analyst
IBM Certified Advanced Technical Expert (CATE)
Plexus (Division of BancTec) (Canada)

Using ClamAV

IBM has recently removed support for its AIX security services. Anti-virus protection is becoming more and more necessary, especially with increasing numbers of Windows boxes appearing in the workplace.

There are, of course, plenty of mature anti-virus products available for the AIX environment; including F-Prot for AIX, and the usual suspects such as Norton and Symantec. However, we have started to use an Open Source product called ClamAV. We were specifically interested in rapid high-volume mail scanning.

ClamAV is a GPL virus scanner toolkit developed specifically for the Unix environment. It includes a command-line scanner (Clamscan), a quick multi-threaded daemon, an interface for sendmail, and a virus scanner C library. ClamAV comes with freshclam, a tool that can periodically check for new database releases, or alternatively the database can be updated manually. It currently has a virus detection database, which includes about 25,000 definitions. It also has built-in support for RAR

(2.0), Zip, Gzip, Bzip2, Mbox, Maildir, and raw mail files. Crucially, the virus database is kept up to date.

ClamAV is released under the GPL V2 licence. What this means for the end user is that, unlike most anti-virus products, ClamAV is completely free and has no yearly subscription fee, and it comes with source code for anyone who wishes to use it.

Freeware is based on the following principles; the authors maintain the ownership and copyright of the freeware program and grant the user a licence. Usually a licence is granted to use and distribute the software in its unmodified form. This allows the authors to maintain legal and intellectual control over their work.

Freeware should not be confused with free software or shareware. Free software is software that is free to copy, redistribute, and modify, which implies that it is available as source code. Although it does not necessarily mean that it is free of charge (anyone can sell free software as long as they do not impose any new restrictions on its redistribution or use). Shareware is software distributed without charge, but for which the author requests some payment, which may buy additional support, documentation, or functionality.

SUPPORT

Because the freeware applications are available free of charge, there is no warranty for the programs. What this usually means is that the site owners accept no responsibility for the condition of the programs in the archives or for the results of copying, retrieving, compiling, linking, or executing any program. Nor are they responsible for the results if you follow any of the advice in any message, news item, or note contained in the archives. Usually the entire risk as to the quality and performance of the program is with the user; and, should the program prove defective, the user would assume the cost of all necessary servicing, repair, or correction. ClamAV has a number of mailing lists you can join that are devoted to user questions, the virus database, and developer news.

There is a degree of reticence within some organizations about installing Open Source software on their systems. However, in the AIX world, the HTTP server is based on the Open Source Apache Server.

THE AIX PORT

An AIX port for ClamAV can be found at the AIX Public Domain Software Library at http://aixpdslib.seas.ucla.edu/aixpdslib.html.

The AIX PDSLIB is run by the UCLA (University of California at Los Angeles) School of Engineering and Applied Science. The Web site is mainly an interface to the AIXPDSLIB ftp site.

The binary packages for AIX are available in the AIX PDSLIB at http://aixpdslib.seas.ucla.edu/packages/clamav.html.

Daily built precompiled snapshots are available at http://clamav.or.id/snapshot.

In addition to AIX, ClamAV has been ported to Debian, FreeBSD, RedHat - Fedora, Interix, PLD Linux Distribution, MacOS, Mandrake, OpenBSD, OSF, Slackware, Solaris 8(Sparc),SuSE, and MS Windows.

Crucially, the functionality of ClamAV is available for the MS Windows environment using the Cygwin compatibility layer for Win32. You can download a copy of the Cygwin source code from http://www.sosdg.org/cygwin. This means that you can try it out at home to see whether it justifies being brought in for testing in your enterprise. You can download it from http://www.sosdg.org/clamav-win32/index.php.

A GUI for ClamAV is also available at http://www.clamwin.net. It provides a Graphical User Interface to the Clam anti-virus scanning engine. ClamWin provides most of the functionality of ClamAV, such as scanning folders or files, configuring settings, and updating the virus databases. Remember that the GUI and Cygwin compatibility layer will reduce performance on a Windows system compared with your AIX box at work.

ADVANTAGES

We have found the following advantages in using ClamAV:

- Speed – the multi-threaded daemon makes the scanning of mail files extremely rapid. We regularly scan tens of thousands of mail messages per day.
- Updating – the virus definitions are updated almost every day. Certainly, whenever there are new viruses out in the wild, definitions are posted rapidly.
- Cost effective – the open source model makes the use of ClamAV the most cost-effective solution on the market.

A brief look at the ClamAV Web site reveals that a large number of ISPs, network security companies, and educational establishments are using the product in a production environment. This is instructive. A careful review of the data shows, for example, that ClamAV is able to scan large numbers of e-mails per day, more than is adequate for most enterprises.

CONCLUSIONS

We have found that ClamAV is a highly effective virus scanner toolkit, especially if you require rapid mail scanning. Obviously, we would recommend that you look at all the anti-virus options available on the marketplace, but would suggest that a review of ClamAV is warranted.

Paul Jones
Systems Programmer (UK)

Happy 18th birthday AIX

AIX was first released in 1986, which means that IBM has quietly celebrated its 18th birthday this year.

AIX was IBM’s first major bite at the Unix cherry, and for a long time ran on proprietary RS/6000 hardware. It has sometimes been overshadowed by IBM’s second big attempt at selling a Unix variant, Linux, which runs on every platform from mainframes to laptops (and on some PDAs).

AIX is an acronym for Advanced Interactive eXecutive, although the only time you’ll ever need to know that is if you’re doing a quiz. AIX is always referred to as AIX and almost never by its full name.

Like all 18-year olds, it has had growing pains. Many people have highlighted its inconsistencies with other Unix systems. However, this argument has always been countered by aficionados who list all its interesting new ideas. You can’t be innovative without being different – they say. So what’s special about AIX? Well, there are the commands, like chuser, mkuser, rmuser, and similar, which make user administration as easy as administering files. Interestingly, AIX’s level of support for volume management can now be found being included in other Unix variants.

Much like the mainframe operating system MVS was jokingly meant to stand for Man Versus System; AIX, because of its inconsistencies with other Unix systems, was said to be an acronym for Ain’t unIX. Another common name was ‘AIX and pains’.

AIX started life as the user interface to the operating system for the RT/PC CAD/CAM workstation in 1986. It was later ported to the PS/2 and S/370. At that time, it gained POSIX conformance, 52-bit addressability, enhancements in file reliability, TCP/IP, fault tolerance, NFS support, FDDI support, etc.

There was an AIX/370, which was designed to run on mainframes. It was announced in 1989 and was a replacement for IX/370. Basically it ran under VM as a guest operating system.

In 1991 there was AIX/ESA. This again ran on mainframes, either under VM or in its own PR/SM partition. Its presence on the mainframe was a little superfluous because IBM had started its Linux movement with the introduction of MVS/ESA OpenEdition.

Although the IBM Personal Computer RT (RISC Technology) was launched in January 1986 running AIX, it was after the engineers (in Austin) developed AIX for the second generation RISC System 6000 (the RS/6000) that AIX really took off. This was AIX Version 3, and was launched in February 1990.

Version 4 came out in 1994, Version 5L in 2000, and 5.3 is currently available.

AIX is an implementation of Unix derived from both AT&T's Unix System V and 4.3 BSD. AIX offers the Korn (ksh) shell, Bourne (sh), and C (csh) shells; however, it defaults to the Korn shell.

In early 1993, AT&T sold it Unix System Laboratories (where it had originally produced Unix System V) to Novell. In 1995 SCO bought the Unix Systems business from Novell, and Unix system source code and technology continued to be developed by SCO. And that is the reason behind the recent court case between SCO and IBM over AIX. SCO tried to terminate IBM's right to use AIX in its business, development, distribution, and sales. SCO wanted to derive more revenue from the company's Unix intellectual property.

So that just about brings you up to date. Happy 18th birthday AIX.

Nick Nourse
Independent Consultant (UK)

Deploying multiple compiler versions on AIX

DEFINITION OF THE PROBLEM

Until recently IBM C/C++ and Fortran compilers had to be installed in a fixed directory, */usr/vac*, */usr/vacpp*, and */usr/xlf* respectively. The system administrator performing the installation had absolutely no control over the target filesystem location to place the installed products. Effectively this IBM policy has limited any single computer to installing only one version of each compiler.

In many environments such a policy is not acceptable. Existing production systems cannot adopt new versions of the compilers until the completion of extensive testing activities. In such cases, customers have no other choice but to purchase a dedicated testing server – an expense that is not always bearable by a particular project.

IBM SOLUTION

Enter the NDI tool that ships with VisualAge C++ V6.0 for AIX and XL Fortran V8.1 for AIX, and is supported only by those versions (or later) of their respective products.

These versions of the NDI tool also install the corresponding version of the run-time environment when an additional version of the compiler is installed.

Newly provided NDI utilities are contained in the following file sets:

# lslpp -l | grep ndi:

- vac.ndi – 6.0.0.2 APPLIED – C for AIX non-default.
- vacpp.ndi – 6.0.0.2 APPLIED – VisualAge C++ non-default.
- xlf.ndi – 8.1.0.3 APPLIED – XL Fortran compiler.

For brief usage options, you can execute the appropriate tool (**xlfndi**, **vacndi**, or **vacppndi)** with the **-h** argument:

```
# /usr/vac/bin/vacndi -h
```

Usage:

```
perl ./vacppndi [-h] [-u <file>] [-m] [-b <path>] -d <path> [-e log] [-
z] [-tryandbuy] [-i]
```

Option definitions:

- -h – display help; ie usage information.
- -u – user-created PTF information file (list of PTF packages).
- -m – minimal compiler install; ie no documents, no samples.
- -b – install target path (directory to create base install).
- -d – install source path (directory where the filesets are).
- -e – log file name; default is used if the option is not specified.
- -z – omit back-up prior to PTF update.
- -tryandbuy – installing the try-and-buy version of the compiler.
- -i – use *trans.tbl* file mapping.

As you can see, in addition to the installation of the compiler products' filesets, the tool supports the installation of PTFs for products installed in non-default locations.

Prerequisite filesets for the usage of the tool are:

- bos.adt.include – base application development INCLUDE files.
- bos.adt.lib – base application development libraries.
- bos.adt.libm – base application development maths library.
- ifor_ls.base.cli – LUM run-time – required for evaluation versions.

The space required is approximately 150MB for the base installation. For PTF updates, the tool requires approximately another 150MB to back-up the base installation (to allow for easy recovery in the event of an error) plus the additional space requirements for the PTFs you wish to apply.

For complete details of the features and usage of the NDI tool, you can also refer to the documentation included at the beginning of the NDI script itself.

```
# more /usr/vac/bin/vacndi
```

I wanted to use the tool to install the newly-announced IBM XL C/C++ compiler Version 7.0 on a server that already has VisualAge Version 6.0 installed.

The first step was to install new versions of the vacppndi tool:

```
# installp -aXTYgd . -e   ndi.log vacpp.ndi
+-----------------------------------------------------------------------------+
                    Pre-installation Verification...
+-----------------------------------------------------------------------------+
Verifying selections...done
Verifying requisites...done
Results...

SUCCESSES
---------
  Filesets listed in this section passed pre-installation verification
  and will be installed.

  Selected Filesets
  -----------------
  vacpp.ndi 7.0.0.0                   # IBM XL C/C++ Non-Default Ins...

  Requisites
  ----------
  (being installed automatically;  required by filesets listed above)
  vacpp.licAgreement 7.0.0.0          # IBM XL C++ Electronic Licens...

  << End of Success Section >>

FILESET STATISTICS
------------------
    1  Selected to be installed, of which:
        1  Passed pre-installation verification
    1  Additional requisites to be automatically installed
  ----
    2  Total to be installed

… (Omitted Lines)

Installation Summary
--------------------
Name                    Level           Part        Event       Result
-----------------------------------------------------------------------------
vacpp.ndi               7.0.0.0         USR         APPLY       SUCCESS
vacpp.licAgreement      7.0.0.0         USR         APPLY       SUCCESS
```

TOOL WORKINGS OVERVIEW

The NDI tool installs the VisualAge C++ V6.0 for AIX or XL Fortran V8.1 (or later versions) for AIX compiler packages to a non-default location. The operations performed by the tool are:

- Processing of AIX **installp** fileset to extract and copy files from that fileset to the non-default target installation location.
- Modification of the compiler configuration file to replace references to the default installation location (eg */usr/vacpp*) with the non-default installation location.
- Wrapping of compiler executables with a shell script that does the call to the actual compiler executable, reinitializing the LIBPATH (NLSPATH) shell variable to point to libraries, message catalogs, etc from the non-default location. If the executable being wrapped is a compiler invocation, the **-F** compiler option is appended to the compiler invocation call to direct the compiler to pick up the translated configuration file (instead of the default installation location in */etc* directory).

INSTALLATION ASSISTED BY THE TOOL

To install the compiler using the NDI tool, you need access to the **installp** compiler filesets and write permission to a directory with sufficient disk space. If there is insufficient disk space, the NDI tool will exit and inform you how much is required. An additional requirement is to make sure that the directory into which you are going to install the tool does not already exist.

In my case, I have chosen directory */opt_new/vacpp* for the installation of the new compiler version:

```
# perl /usr/vacpp/bin/vacppndi -d . -e myvacppndi.log -b /opt _new/vacpp
-- Start of install for "IBM XL C/C++ V.7.0" at Tue Sep 28 17:09:31 IST
2004
-- Command line is "/usr/vacpp/bin/vacppndi -d . -e myvacppndi.log -b /
opt_new/vacpp"
-- Installation target path is "/opt_new/vacpp"
---- Executing component "inst-prep" at Tue Sep 28 17:09:31 IST 2004
---- Running custom install code for "inst-prep"..
```

```
------ Checking for package "/system_dir/vacpp7/./memdbg.adt"..

… (Omitted Lines)

------ Checking for package "/system_dir/vacpp7/./vacpp.msg.ZH_CN"..
------ Checking for package "/system_dir/vacpp7/./vacpp.pdf.en_US"..
------ Checking for package "/system_dir/vacpp7/./vacpp.samples"..
---- "inst-prep" successful at Tue Sep 28 17:09:31 IST 2004
---- Executing component "setup" at Tue Sep 28 17:09:31 IST 2004
---- Running custom install code for "setup"..
---- "setup" successful at Tue Sep 28 17:09:31 IST 2004
---- Executing component "extract-vrmf" at Tue Sep 28 17:09:31 IST 2004
---- Running custom install code for "extract-vrmf"..
---- "extract-vrmf" successful at Tue Sep 28 17:09:35 IST 2004
---- Executing component "gen-list" at Tue Sep 28 17:09:35 IST 2004
---- Running custom install code for "gen-list"..
---- "gen-list" successful at Tue Sep 28 17:09:35 IST 2004
---- Executing component "vrmf-check" at Tue Sep 28 17:09:35 IST 2004
---- Running custom install code for "vrmf-check"..
---- "vrmf-check" successful at Tue Sep 28 17:09:35 IST 2004
---- Executing component "extract-lpp" at Tue Sep 28 17:09:35 IST 2004
---- Running custom install code for "extract-lpp"..
---- "extract-lpp" successful at Tue Sep 28 17:09:49 IST 2004
---- Executing component "sys-check" at Tue Sep 28 17:09:49 IST 2004
---- Running custom install code for "sys-check"..
---- "sys-check" successful at Tue Sep 28 17:09:49 IST 2004
---- Executing component "unwind" at Tue Sep 28 17:09:49 IST 2004
---- Running custom install code for "unwind"..
------ Restoring fileset "vacpp.pdf.en_US" from "vacpp.pdf.en_US"..

… (Omitted Lines)

------ Restoring fileset "vac.pdf.en_US.C" from "vac.pdf.en_US.C"..
------ Restoring fileset "vacpp.msg.JA_JP.cmp.tools" from
"vacpp.msg.JA_JP"..
---- "unwind" successful at Tue Sep 28 17:11:33 IST 2004
---- Executing component "pre-inst" at Tue Sep 28 17:11:33 IST 2004
---- Running custom install code for "pre-inst"..
---- "pre-inst" successful at Tue Sep 28 17:11:34 IST 2004
---- Executing component "integrate" at Tue Sep 28 17:11:34 IST 2004
---- Running custom install code for "integrate"..
---- "integrate" successful at Tue Sep 28 17:11:34 IST 2004
---- Executing component "lpp-magic" at Tue Sep 28 17:11:34 IST 2004
---- Running custom install code for "lpp-magic"..
---- "lpp-magic" successful at Tue Sep 28 17:11:49 IST 2004
---- Executing component "wrappers" at Tue Sep 28 17:11:49 IST 2004
---- Running custom install code for "wrappers"..
---- "wrappers" successful at Tue Sep 28 17:12:03 IST 2004
---- Executing component "cfg-files" at Tue Sep 28 17:12:03 IST 2004
---- Running custom install code for "cfg-files"..
```

```
---- "cfg-files" successful at Tue Sep 28 17:12:04 IST 2004
---- Executing component "post-inst" at Tue Sep 28 17:12:04 IST 2004
---- Running custom install code for "post-inst"..
---- "post-inst" successful at Tue Sep 28 17:12:04 IST 2004
---- Executing component "summary" at Tue Sep 28 17:12:04 IST 2004
---- Running custom install code for "summary"..
------ Installation Summary:
 INSTALLED: vacpp.pdf.en_US 7.00.0000.0

 … (Omitted Lines)

 INSTALLED : vacpp.msg.JA_JP.cmp.tools 7.00.0000.0
------ End Installation Summary.
---- "summary" successful at Tue Sep 28 17:12:04 IST 2004
---- Executing component "cleanup" at Tue Sep 28 17:12:04 IST 2004
---- Running custom install code for "cleanup"..
---- "cleanup" successful at Tue Sep 28 17:12:13 IST 2004
-- End of install for "IBM XL C/C++ V.7.0" at Tue Sep 28 17:12:13 IST
2004
-- Install was successful.
-- Log file is "/system_dir/vacpp7/myvacppndi.log"
```

The file */system_dir/vacpp7/myvacppndi.log* contains much more extensive logging information.

To see what was installed:

```
# cat /opt_new/vacpp/.vrmf_history

### BEGIN: Tue Sep 28 17:12:04 IST 2004 ###
vacpp.pdf.en_US : 7.00.0000.0
vac.msg.JA_JP.C : 7.00.0000.0
memdbg.msg.zh_CN : 5.1.0.0
vacpp.cmp.core : 7.00.0000.0
xlC.msg.ja_JP.rte : 7.00.0000.0000
… (Omitted Lines)
vacpp.cmp.aix50.tools : 7.00.0000.0
vac.pdf.en_US.C : 7.00.0000.0
vacpp.msg.JA_JP.cmp.tools : 7.00.0000.0
### END: Tue Sep 28 17:12:04 IST 2004 ###
```

As can be seen from the results of the following command, the AIX ODM repository has not been updated with the fact that an additional version of the compiler has been installed.

```
# lslpp -l|grep -i vacpp
  vacpp.cmp.aix50.lib    6.0.0.7  COMMITTED  VisualAge C++ Libraries for
  vacpp.cmp.aix50.tools  6.0.0.1  COMMITTED  VisualAge C++ Tools for AIX
  vacpp.cmp.core         6.0.0.8  COMMITTED  VisualAge C++ Compiler
  vacpp.cmp.include   6.0.0.7  COMMITTED  VisualAge C++ Compiler Include
  vacpp.cmp.lib          6.0.0.0  COMMITTED  VisualAge C++ Libraries
  vacpp.cmp.rte          6.0.0.0  COMMITTED  VisualAge C++ Compiler
  vacpp.cmp.tools        6.0.0.7  COMMITTED  VisualAge C++ Tools
  vacpp.lic              6.0.0.0  COMMITTED  VisualAge C++ Licence Files
  vacpp.licAgreement   7.0.0.0  COMMITTED  IBM XL C++ Electronic License
  vacpp.memdbg.aix50.lib 6.0.0.4  COMMITTED  VA C++ User Heap/Memory
Debug
  vacpp.memdbg.aix50.rte 6.0.0.7  COMMITTED  VA C++ User Heap/Memory
Debug
  vacpp.memdbg.lib      6.0.0.0  COMMITTED  VisualAge C++ User Heap and
  vacpp.memdbg.rte       6.0.0.0  COMMITTED  VisualAge C++ User Heap and
  vacpp.ndi              7.0.0.0  COMMITTED  IBM XL C/C++ Non-Default
  vacpp.cmp.core         6.0.0.0  COMMITTED  VisualAge C++ Compiler
```

ADDITIONAL CONSIDERATIONS

The XL Fortran and C++ run-time environments support backward compatibility. Executables generated by an earlier release of a compiler will work with a later version of the run-time environment. For example, an executable generated by Version 6.1 of the XL Fortran compiler will work with Version 8.1 of the XL Fortran run-time environment. An executable generated by VisualAge C++ Version 5 will work with Version 6 of the C++ run-time environment. Forward compatibility, however, is not supported. Consider the following example.

After performing the installation of the new compiler I was eager to test it so I built one of the samples included with it:

```
#cd /opt_new/vacpp/usr/vacpp/samples/stl/map
# export PATH=/opt_new/vacpp/usr/vacpp/bin/:$PATH
# make
      xlC -qrtti=all -blibpath:"/usr/vacpp/lib:/usr/lib:/lib" -c map.cpp
xlC -qrtti=all -blibpath:"/usr/vacpp/lib:/usr/lib:/lib" map.o -o map_aix
Target "all" is up to date.
# ./map_aix
exec(): 0509-036 Cannot load program ./map_aix because of the following
errors:
   0509-130 Symbol resolution failed for map_aix because:
   0509-136   Symbol _Getctype__FPCc (number 52) is not exported from
                  dependent module /usr/lib/libC.a(ansi_32.o).
   0509-192 Examine .loader section symbols with the 'dump -Tv' command.
```

What has happened here? The reason for the failure of the sample program is that it has been linked with **libpath** pointing to the old version of the C++ run-time libraries.

To fix the problem you should execute:

```
xlC -qrtti=all -blibpath:"/opt_new/vacpp/lib:/usr/lib:/lib" map.o -o
map_aix
```

When multiple versions of a compiler/run-time environment are installed on the same computer, it is the developers' responsibility to make sure that a run-time environment for their application is at the same level as or higher than the compiler used to generate the executables used. The LIBPATH environment variable can be set to point to the directory containing the appropriate version of the run-time libraries.

In any one process, there can be at most one C++ run-time environment and at most one Fortran run-time environment. For example, it is not valid to have a program compiled with VisualAge C++ Version 6 link to a shared library that explicitly uses a renamed VisualAge C++ Version 5 run-time library.

Consider the following guidelines in order to prevent erroneous situations:

- Do not rename the C++ or Fortran run-time libraries. Renaming a C++ or Fortran run-time library will introduce the potential for having multiple run-time environments in one process, and therefore is not valid.
- Do not statically link to the run-time libraries. A program or shared library that statically links in a C++ or Fortran run-time library is considered to be a version of the library. For example, it is not valid to statically link a C++ or Fortran run-time library to a main program, and also dynamically link to a shared library that references the run-time library at the same or a different level.
- Do not link explicitly with an instance of the C++ or Fortran run-time library (for example, **xlC /myNonDefaultInstall/lib/libC.a file.cpp** ...).
- Beware of the usage of **dlopen** or **load** that can potentially load a shared library, which can introduce multiple run-time environments. This is not valid.

PMRs regarding problems that occur at run-time will not be accepted as valid problem reports by the IBM compiler service team if the application does not follow the above guidelines.

DETECTING MULTIPLE RUN-TIME USAGE

Shared objects may have different library search paths in their loader sections from modules that are dependent on them, or modules that they are dependent on. This may cause the loading of different versions of run-time libraries at start-up.

To investigate the problem you can use the **dump -H** command:

```
# cd /opt_new/vacpp/usr/samples/vacpp/stl/map
# dump -H map_aix

map_aix:

                         ***Loader Section***
                       Loader Header Information
VERSION#         #SYMtableENT      #RELOCent         LENidSTR
0x00000001       0x000000da        0x00000242        0x0000006b

#IMPfilID        OFFidSTR          LENstrTBL         OFFstrTBL
0x00000006       0x00002fa8        0x00002c59        0x00003013


                         ***Import File Strings***
INDEX  PATH                          BASE                MEMBER
0      /usr/vacpp/lib:/usr/lib:/lib
1                                    libc.a              shr.o
2                                    libC.a              shr2.o
3                                    libC.a              shr3.o
4                                    libC.a              ansi_32.o
5                                    libC.a              shr.o
```

You can also use the **ldd** command:

```
#  ldd map_aix
map_aix needs:
         /usr/lib/libc.a(shr.o)
         /usr/vacpp/lib/libC.a(shr2.o)
         /usr/vacpp/lib/libC.a(shr3.o)
         /usr/vacpp/lib/libC.a(ansi_32.o)
         /usr/vacpp/lib/libC.a(shr.o)
         /unix
         /usr/lib/libcrypt.a(shr.o)
         /usr/vacpp/lib/libC.a(shrcore.o)
```

```
        /usr/vacpp/lib/libC.a(ansicore_32.o)
        /usr/lib/libc_r.a(shr.o)
```

SUMMARY

The ability to install multiple compiler versions facilitates reliability and performance across software upgrades, by decreasing the chance of a compiler failure impacting user work and by providing more optimization choices.

The ability to use shared (GPFS, NFS, and AFS) file space for the compiler directory provides access to different compiler versions to all nodes that mount the filespace after a single installation.

The ability to back up and restore the compiler installation directory facilitates easy retrieval of previous compiler installations for regression testing.

There is no mechanism to check the integrity of the NDI install (such as **lppchk** does for the default installation), so careful review of the installation log is required. However, the script does catch, report, and abort when major errors are encountered.

Compiling and linking multi-language applications requires special attention. Special attention should also be given to the proper use of run-time libraries during link and execution of programs built with compilers installed in non-default locations.

REFERENCES

1 IBM AIX compiler information centre: http://publib.boulder.ibm.com/infocenter/comphelp/index.jsp.

2 *Developing and Porting C and C++ Applications on AIX*, SG24-5674 (HTML, PDF)

3 *AIX 5L Porting Guide*, SG24-6034 (HTML, PDF)

4 XLF Support FAQ: http://www.ibm.com/software/awdtools/fortran/xlfortran/support/

5 VACPP Support FAQ: http://www.ibm.com/software/awdtools/vacpp/support/

Alex Polak
System Engineer
APS (Israel)

Mirroring tips and techniques in AIX

This article is aimed at system administrators who are new to AIX or those coming from other flavours of Unix. This article will summarize the benefits of using mirroring, the steps taken to mirror your logical volumes, some troubleshooting scenarios, and some tips that should be considered when planning for mirroring. It will also provide you with the answers to some of the most commonly-asked questions:

- Mirroring is the easiest, cheapest, and the first-level method of protecting your data. It involves only extra storage space and in some cases an additional adapter. The price of storage devices these days makes mirroring very convenient to implement.
- Mirroring reduces, and in most cases eliminates, system down time in cases of hard disk failure.
- Implementing mirroring can be done with users on.
- Maintaining and troubleshooting mirroring (in most cases) can also be done with users on.
- The increase in system overhead is very low.
- Mirroring can be used on any type of disk (SCSI, SSA, etc), whether it is internal or external.

TIPS AND ROLES FOR MIRRORING

In order to take full advantage when setting up mirroring, here are some tips and roles that should be taken into consideration during the implementation:

- You can have up to three mirrored copies of each logical volume on each system.
- Original copies and mirrored copies should reside on separate physical volumes (ie hdisks). You can split the copies into groups of disks using any appropriate method you find suitable for your situation. For example you can use even/odd disk numbers, eg hdisk2, hdisk4, hdisk6, etc for the originals and hdisk3, hdisk5, hdisk7, etc for the mirrored copies, or a continuous set of disk numbers, for instance, hdisk2, hdisk3, and hdisk4 for the originals and hdisk5, hdisk6, and hdisk7 for the mirrored copies.
- It is possible to mirror all types of logical volume, eg jfs, jfslog, jfs2, jfs2log and paging spaces (with some restrictions when mirroring dumps).
- It is recommended that you split the copies into two equal sets of disks (in terms of size and numbers).
- For performance and high-availability efficiency, it is recommended that you separate the two sets of physical volumes onto separate adapters.
- In order to prevent copies interfering on the same physical volume, it is recommended that your response to:

  ```
  "EACH LP COPY ON A SEPARATE PV ?"
  ```

  is to set it to **superstrict** for each mirrored logical volume by using the command (**chlv -s's' LV_name**). The interference will usually occur after increasing the logical volume size.
- Mirroring dump devices is supported in AIX 4.3.3 and higher, but some dump analyser utilities, like the **crash** command, are not capable of reading mirrored raw devices.

Therefore, it is recommended that you use the secondary dump device method where a secondary dump device is created equal in size to the primary and will act as a failover device in case the dump cannot be done to the primary dump device.

MIRRORING A VOLUME GROUP

This section will provide the procedure to mirror a volume group that is supported on all AIX versions. The following example assumes that the base physical volume is *hdisk0* and the mirrored disk is a new disk *hdisk1*. All steps must be performed as user root.

1 Extend the volume group to at least two physical volumes:

```
# extendvg  your_vg  hdisk1
```

2 It is a good idea to turn off quorum (if you have it turned on), especially if the volume group you are mirroring is rootvg, which usually has two disks. If quorum is on and you lost the first disk, which contains over 50% of the VGDA, the volume group will be varied off and no-one will be able to access this volume group.

```
# chvg  -Qn  rootvg
```

3 Make two copies of each logical volume:

```
# mklvcopy  your_lv  2  hdisk1
```

The *2* in the above syntax is the number of copies. You can prepare a small script to make the copies for more convenience.

```
# for  lv  in  my_lv_1 my_lv_2 my_lv_3
>do
> mklvcopy ${lv}  2  hdisk1
>done
```

4 Synchronize the copies. This also can be done directly in the previous step with the use of the **-k** switch with the **mklvcopy** command:

```
# syncvg  -v  your_vg
```

5 If the volume group being mirrored is rootvg, update the boot logical volume and the IPL bootlist:

```
# bosboot  -a  -l  /dev/hd5  -d  /dev/hdisk1
# bootlist  -m  normal  hdiskØ  hdisk1
```

6 Also if the volume group being mirrored is rootvg, create and set the secondary dump device (unless you decide to mirror the dump device, then you can do it in the same way as in Step 3):

```
# mklv  -y  hd7x  -t  sysdump  -a  e  rootvg  16  hdisk1
# sysdumpdev  -P  -s  /dev/hd7x
```

The *16* in the **mklv** command syntax is the number of PPs in the new dump device.

7 Check the state of the volume group. All LVs in the volume group should have a PP size double the LP size, and the LV state should be synchronized:

```
# lsvg -l your_vg
```

8 Alternatively, to mirror a complete volume group you can use the **mirrorvg** command to mirror the whole volume group in one shot. This command is very convenient when mirroring non-rootvg:

```
# mirrorvg -S -c 2 -m your_vg hdiskØ hdisk1
```

The **-m** switch will match the exact maps between the original and the mirrored disks.

BREAKING A MIRRORED VOLUME GROUP

The following procedure can be used to break a mirrored volume group, eg for disk replacement:

1 Invalidate the previous bootlist in case hdisk1 was used for the last boot (for rootvg only). Then set the bootlist to use hdisk0:

```
# bootlist  -m  prevboot  -i
# bootlist  -m  normal  hdiskØ
# mkboot  -d  /dev/hdisk1  -c
```

2. Change the secondary dump device to *sysdumpnull* and remove the secondary dump device from the hdisk1:

   ```
   # sysdumpdev  -P  -s  /dev/sysdumpnull
   # rmlv  hd7x
   ```

3. Remove the logical volume copies:

   ```
   # rmlvcopy  your_lv  1  hdisk1
   ```

   The *1* in the above syntax is the number of copies to remove. You can prepare a small script to remove the copies for more convenience:

   ```
   # for  lv  in  my_lv_1 my_lv_2 my_lv_3
   >do
   > rmlvcopy ${lv}  1  hdisk1
   >done
   ```

4. Remove hdisk1 from the volume group:

   ```
   # reducevg  your_vg  hdisk1
   ```

5. Recreate the boot image if the volume group is rootvg:

   ```
   # bosboot  -d  /dev/hdisk0  -a -l /dev/hd5
   ```

6. If hdisk1 is to be replaced, it is best to remove it from the system:

   ```
   # chdev  -l  hdisk1  -a  pv=clear
   # rmdev  -l  hdisk1  -d
   ```

7. Alternatively, to break the mirror on a complete volume group you can use the **unmirrorvg** command to remove the mirror on the whole volume group in one shot. This command is very convenient when breaking mirrored non-rootvg:

   ```
   # unmirrorvg -c 1 datavg hdisk1
   ```

Notes:

- There is an ODM object called vg-lock. Whenever an LVM modification command runs (like **mklvcopy**, **mklv**, **mifratepv**, **rmlv**, **reorgvg**, etc), the command will use vg-lock to lock the volume group being modified. If for some reason this lock is left behind because of an unsuccessful termination of the command, the volume group can be unlocked by running the command **varyonvg -b**, which can be run on a volume group that is already varied on.

- It is not recommended to kill or terminate any of the mirroring-related (or the LVM commands in general) processes because that can corrupt the ODM.
- Because some mirroring-related commands may lock the volume group, you cannot run more than one command on the same volume group at the same time, even in background mode.

MIRRORING RECOVERY AND TROUBLESHOOTING TECHNIQUES AND SCENARIOS

The following scenarios are the most common ones you will face when recovering and troubleshooting mirrored logical volumes. Most of the problems you might face can be narrowed down to one of these scenarios.

Scenario # 1

A mirrored rootvg disk hdisk0 is failing. The disk is still available and accessible, but the error log is giving hardware errors and the analysis indicates that the disk needs to be replaced.

Recovery procedure

The procedure to recover from this situation is to repeat the steps in *Breaking a mirrored volume group* on the failing disk hdisk0. Physically replace hdisk0, then mirror back rootvg by repeating the steps in *Mirroring a volume group*. Pay special attention in both procedures to those steps dedicated to rootvg.

Scenario # 2

A mirrored hdisk0 is failing. The disk is not available and not accessible and the physical volume is declared missing. The error log is giving hardware permanent errors and the disk needs to be replaced.

Recovery procedure

The procedure to recover from this situation is to repeat the steps in *Breaking a mirrored volume group* on the failing disk hdisk0. You might face problems using some commands like **rmfs**, **reducevg**, and/or **migratepv** because the physical volume is missing and more tricky and low-level commands are needed to clean out the presence of the unavailable disk from the ODM. After the configuration of the missing disk is removed from the system, physically replace hdisk0, then mirror back rootvg by repeating the steps in *Mirroring a volume group*.

Scenario # 3

A mirrored datavg disk hdisk2 is failing. The disk is still available and accessible, but the error log is giving hardware errors and the analysis indicates the disk needs to be replaced. Datavg has two sets of disks – one set is mirrored on the other.

Recovery procedure

Investigate whether the other disk members in the same hdisk2 set have sufficient spare space to host hdisk2's contents. If they do have, migrate (move) all logical partitions out from the failing hdisk2 to other disk members in the same set using the **migratepv** command, and then remove the hdisk2 configuration from the system using the **reducevg** and **rmdev** commands. Physically replace hdisk2, then move back the logical partition to the new hdisk2 using the **migratepv** or **reorgvg** commands.

If the space is not sufficient, or hdisk2 is the only member in the set (datavg has only two disks, one is mirrored on the other), repeat the steps in *Breaking a mirrored volume group* on the failing disk hdisk2 (or break the mirror for just those logical volumes located on hdisk2). Physically replace hdisk2 then mirror back datavg by repeating the steps in *Mirroring a volume group* (or mirror back only those logical volumes that have been broken on hdisk2).

Scenario # 4

When checking a mirrored volume group using the **lsvg -l** command, the state of some or all of the logical volumes is showing as STALE.

Recovery procedure

Investigate what caused the logical volume copies to be STALE by checking the error log (errpt) for hard disk errors. If the cause is a hard disk failure, compare and match to one of the previous scenarios. If the cause is not a hard disk failure, it might be a corruption in one of the copies because of the termination of a synchronization process. To recover from this situation run the **syncvg** command on the STALE logical volumes, physical volumes, or volume groups.

Scenario # 5

You are being asked to separate or split the two copies of the */myfs* filesystem temporarily, in order to perform a back-up of the mirrored copy while the users are still accessing the filesystem. After the back-up is completed, you need to re-join the copies.

Recovery procedure

Using the **chfs -a splitcopy=/tempfs /myfs** command splits the two copies of the */myfs* filesystem and mount a read-only copy in */tempfs*. It is recommended, for data consistency reasons, that this step be done while users are off the filesystem. The back-up can be performed now in the */tempfs* filesystem while users are accessing the original filesystem. After the back-up is complete, you simply have to un-mount the temporary filesystem */tempfs* for the copies to join and sync with the original. The **splitcopy** attribute of the **chfs** command is supported only in AIX 4.3.3 and higher.

Scenario # 6

You are being asked to separate or split the two copies of a logical volume onto two different logical volumes, because some developers need to conduct a few tests on the temporary filesystem. After the tests are completed, you need to re-build the copies.

Recovery procedure

Using the **splitlvcopy** command splits the two copies of the logical volume into two separate logical volumes. It is recommended, for data consistency reasons, that this step be done while users are off. Add a new filesystem (eg */tempfs*) stanza in the */etc/filesystems* file. Don't create a new filesystem over the new logical volume using the **mkfs** command or you will lose whatever data is on that logical volume. Mount the new filesystem */tempfs*. Developers can now access the data in */tempfs* and conduct their own testing, while users are accessing the original filesystem. After the testing is complete and there is no need for the temporary filesystem */tempfs*, you can rebuild the mirrored copies of the logical volume. To do that you first have to un-mount and remove the temporary filesystem */tempfs* using the **rmfs** command. Then you can repeat the steps in *Mirroring a volume group* on that particular logical volume.

Basim Chafik
Senior Systems Analyst
IBM Certified Advanced Technical Expert (CATE)
Plexus (Division of BancTec) (Canada)

Please note that the correct contact address for Xephon Inc is PO Box 550547, Dallas, TX 75355, USA. The phone number is (214) 340 5690, the fax number is (214) 341 7081, and the e-mail address to use is info@xephon.com.

Generating a logo

INTRODUCTION

The script, genlog.lib, is a library of functions that can be used to generate logos on the screen.

LISTING

```
#####################################################################
#  Name     : genlogo.lib
#
#  Overview : This script is a collection of functions that can be
#              used to generate logos on the screen.
#
#  Notes    : 1. The script contains the following functions:
#                             o InitialiseVariables
#                             o MoveCursor
#                             o DrawVerticalLine
#                             o DrawHorizontalLine
#####################################################################
######################################################################
#   Name    : InitialiseVariables
#  Overview : The function initializes the required variables.
#  Notes    :
#####################################################################
InitialiseVariables ()
{
#
# terminal capabilities
#
BOLDON=`tput smso`  ; export BOLDON
BOLDOFF=`tput rmso` ; export BOLDOFF
#
ESC="\0033["
#
SINGLE_SPACE="  "
#
}
#####################################################################
#    Name     : MoveCursor
#    Input    : Y and X coordinates
#    Returns  : None
#    Overview : It moves the cursor to the required location (Y,X).
#    Notes    : 1. The  co-ordinate (y=0, x=0) is at the top left-hand
#                   corner and therefore, a positive y co-ordinate is
#                   downwards, and a negative y co-ordinate
#                   is upwards.
######################################################################
MoveCursor  ( )
{

YCOR=$1
XCOR=$2

print -n  "${ESC}${YCOR};${XCOR}H"

}
#######################################################################
#    Name     : DrawVerticalLine
#    Input    : X coordinate
#               Y coordinate start
#               Y coordinate end
#    Overview : It draws a vertical line from X,Y_STAR to X,Y_END
#    Notes    :
####################################################################
DrawVerticalLine()
{
#
# assign parameters
#
XCOR=$1
YCOR_START=$2
YCOR_END=$3
#
# move the cursor to starting position
#
while [ $YCOR_START -ne  $YCOR_END  ]
do
   MoveCursor $YCOR_START  $XCOR
   echo "${BOLDON}${SINGLE_SPACE}${BOLDOFF}"
   YCOR_START=`expr  $YCOR_START +  1`
done

}
#
#######################################################################
#    Name     : DrawHorizontalLine
#    Input    : X coordinate start
#               Y coordinat start
#               X coordinat end
#    Overview : It draws a horizontal line from X_START,Y to X_END, Y
#    Notes    :
#######################################################################
DrawHorizontalLine()
```

```
{
#
# assign parameters
#
XCOR_START=$1
YCOR=$2
XCOR_END=$3
#
while [ $XCOR_START -ne  $XCOR_END  ]
do
   MoveCursor $YCOR  $XCOR_START
   echo "${BOLDON}${SINGLE_SPACE}${BOLDOFF}"
   XCOR_START=`expr  $XCOR_START +  1`
done

}
#
######################################################################
#    Name     : DrawBackSlash
#    Input    : X coordinate start
#               X coordinat end
#               Y coordinat start
#               X coordinat end
#    Overview : It draws a line resembling a backslash(\)
#               from X_START,Y_START to X_END, Y_END
#    Notes    :
######################################################################
DrawBackSlash()
{
#
# assign parameters
#
XCOR_START=$1
XCOR_END=$2
YCOR_START=$3
YCOR_END=$4
#
while [ $XCOR_START -ne  $XCOR_END -a $YCOR_START -ne $YCOR_END  ]
#
do
   MoveCursor $YCOR_START  $XCOR_START
   echo "${BOLDON}${SINGLE_SPACE}${BOLDOFF}"
   XCOR_START=`expr  $XCOR_START +  1`
   YCOR_START=`expr  $YCOR_START +  1`
done

}
######################################################################
#    Name     : DrawForwardSlash
#    Input    : X coordinate start
#               X coordinate  end
#               Y coordinate start
#               Y coordinate end (less than Y co-ordinate start)
#    Overview : It draws a line resembling a forwardslash(/)
#               from X_START,Y_START to X_END, Y_END
#    Notes    :
######################################################################
DrawForwardSlash ()
{
#
# assign parameters
#
XCOR_START=$1
XCOR_END=$2
YCOR_START=$3
YCOR_END=$3
#
while [ $XCOR_START -ne  $XCOR_END && $YCOR_START -ne $YCOR_END  ]
#
do
   MoveCursor $YCOR_START  $XCOR_START
   echo "${BOLDON}${SINGLE_SPACE}${BOLDOFF}"
   XCOR_START=`expr  $XCOR_START +  1`
   YCOR_START=`expr  $YCOR_START -  1`
done

}
#
######################################################################
#    Name     : DrawA
#    Overview : It draws the letter A
#    Notes    :
######################################################################
DrawA ()
{
DrawVerticalLine    10  15  22
DrawHorizontalLine  10  15  20
DrawVerticalLine    20  15  22
DrawHorizontalLine  10  18  20

}
#
####################################################################
#    Name     : DrawB
#    Overview : It draws the letter B
#    Notes    :
######################################################################
DrawB ()
{
DrawVerticalLine   10  15  22
DrawHorizontalLine 10  15  21
DrawVerticalLine   20  16  22
DrawHorizontalLine 10  18  20
DrawHorizontalLine 10  22  21

}
#
######################################################################
#    Name     : DrawC
#    Overview : It draws the letter C
#    Notes    :
######################################################################
DrawC ()
{
DrawVerticalLine    10  15  22
DrawHorizontalLine  10  15  20
DrawHorizontalLine  10  22  20

}
#
######################################################################
#    Name     : DrawD
#    Overview : It draws the letter D
#    Notes    :
######################################################################
DrawD ()
{
DrawVerticalLine    10  15  22
DrawHorizontalLine  10  15  20
DrawVerticalLine    20  15  22
DrawHorizontalLine  10  22  21

}
#
#######################################################################
#    Name     : DrawE
#    Overview : It draws the letter E
#    Notes    :
######################################################################
DrawE ()
{
DrawVerticalLine    10  15  22
DrawHorizontalLine  10  15  20
DrawHorizontalLine  10  18  20
DrawHorizontalLine  10  22  20

}
#
######################################################################
#    Name     : DrawM
#    Overview : It draws the letter M
#    Notes    :
######################################################################
DrawE ()
{
DrawVerticalLine    10  15  22
DrawHorizontalLine  10  15  20
DrawVerticalLine    10  15  22
DrawHorizontalLine  10  18  20
DrawHorizontalLine  10  22  20

}
#
InitialiseVariables
#
# To test a letter remove the comment from any of the
#  following invocations
#
#DrawA
#DrawB
#DrawC
#DrawD
#DrawE
```

Arif Zaman
DBA/Developer (UK)

Default user password settings in AIX

INTRODUCTION

This article reviews some of the default AIX settings that affect user password security.

As with most software, the default options that are set during the installation procedure are of minimal security value. They are designed to get a system up and running without causing too many problems for users.

Unfortunately, this does not make for good security. Reducing problems for users also reduces problems for hackers. It is often the case that these default settings are forgotten after the installation is complete. As we shall see below, the proper management of the default settings can be highly beneficial for increasing system security.

The default user password settings can be found in:

```
Configure /etc/security/user
```

Most of the security features are set to **off** by default. Reviewing and updating these settings will ensure that user passwords are more difficult to crack.

PASSWORD LENGTH

One of the principal determinants of the security of passwords is password length. Quite simply it is much easier to attack a short user password than it is to attack a longer one. Today, a simple five character alphabetic password (ie 40 bits long) can be attacked and exposed using a brute force or dictionary technique, in less than a second. And it does not take long on the Internet to find plenty of software (such as LC4) that can be used to break passwords. It is therefore, essential that the **minlen** setting, which determines the minimum number of characters required for a password, be set to around eight characters (**minlen = 8**).

Settings such as **minalpha** and **minother** give form to the passwords. The **minalpha** value indicates the minimum number of alphabetic characters needed in a password. The default is off (**minalpha = 0**). Similarly, the value of **minother** indicates the minimum number of non-alphabetic characters that are required in a password. The default is off (**minother = 0**).

Providing a value such as 1 or 2 for these settings (**minother = 2**) forces users to actually create alphanumeric passwords and passwords with punctuation that are harder to crack.

Another setting in AIX that can be used to increase password security is **maxrepeats**. This is the maximum number or repeated characters that are allowed in a password. The default setting is 8 (**maxrepeats = 8**).

This, of course, could allow a user to create an eight-character password such as AAAAAAAA (and, believe me, this happens more than you might think).

This would be extremely vulnerable to brute force attack methodologies. It would be strongly advisable to decrease this default setting to 2 (**maxrepeats = 2**).

CHANGING USER PASSWORDS

The kinds of simple passwords that are created by users are, at the end of the day, comparatively easy to crack. The best way to deal with the security implications of this is to enforce the changing of passwords.

There are a number of settings that enforce the changing of user passwords in specific time periods and determine the similarity of new and old passwords.

The **minage** and **maxage** values indicate the minimum number of weeks before a password can be changed and the maximum number of weeks before a password must be changed. Both default settings are off (ie **minage = 0**).

Think about changing the **minage** setting to 1 (**minage = 1**) and the **maxage** value to somewhere between four weeks for high security installations and 26 weeks for the lowest security sites. However, we do not recommend keeping passwords active for this length of time in a production environment. You can allow passwords to expire at different intervals, depending on the user profile.

For example, we tend to allow regular users to keep their passwords for three months (**maxage = 12**), but users with extra privileges only one month (**maxage = 4**).

The **mindiff** value determines the minimum number of characters that must be changed from the previous password. The default setting is off (**mindiff = 0**). A good value would be 4 (**mindiff = 4**). This is useful because it encourages users not to create a password sequence such as mypassword1, mypassword2, and mypassword3, at each renewal. Obviously, these sequences are relatively easy to crack.

The **histsize** value determines the number of previous passwords that cannot be reused. The default setting is off (**histsize = 0**). A good value would be any number above 15 (eg **histsize = 25**). This prevents users recycling a couple of passwords every time a renewal is required. To complement this the **histexpire** setting determines the number of weeks before a password can be reused. The default is off (**histexpire = 0**); a good value would be 26 (ie **histexpire = 26**).

The **pwdwarntime** setting gives the user a warning a number of days before a change of password is required. The default **pwdwarntime** setting is off (**pwdwarntime = 0**). This should be changed to three to five days (ie **pwdwarntime = 5**).

LOGIN RETRIES

The **loginretries** setting determines the number of unsuccessful login attempts that are allowed before a user's account is locked. The default **loginretries** setting is off (**loginretries = 0**). A good value for this could be 3 (l**oginretries = 3**). Three is a standard value for this kind of operation.

CONCLUSIONS

The principal issue that has been raised by the above overview is that the default installation setting has much of its security functionality turned off. There is some logic in this, because having the security functionality turned on would impose a steep learning curve on top of the software itself.

However, the default settings need to be reviewed on a regular basis to make sure that they are still appropriate to the running of your systems. This will significantly increase security at your site.

We all know of users who do use simple, short, alphabetic passwords, who do not change their passwords for years at a time, and who write their passwords down. Changing the default settings mentioned above will go some way to automating the creation of more secure passwords than is otherwise likely.

Obviously there is a compromise to be made between increasing the security and complexity of a password and making it too complex for users to remember. This obviously leads to users writing down passwords, which is not good for security.

At our shop, we also like to use password hacking tools (eg Crack) to check whether any users have easily-guessed passwords. Changing the default settings mentioned above has significantly reduced the number of passwords that are vulnerable to these tools.

The other implication of changing the default values is that you will need to inform the users of the changes. It may be appropriate to create a revision to the security policy, explaining why stronger user passwords are necessary and what changes have been made (eg new minimum character lengths for passwords). Users should be given advice on choosing passwords such as: at least eight characters long, which can include combinations of letters, numbers, punctuation, and changes in case. Passwords should not be dictionary words, names or initials, or telephone numbers, or street addresses.

It is obviously a good idea to inform users about password security. Explain why it is necessary to use stronger passwords and what sorts of change have been made (such as time expiry, minimum number of characters, types of character, etc). The above-mentioned settings will help enforce these rules. The security policy will also need to be updated in the light of any new changes.

We are fortunate that AIX is a highly-secure and resilient operating system, but it would still be beneficial for organizations to review their default AIX security settings on a regular basis.

The use of well thought out passwords is one element of system security.

John Edwards
Systems Administrator (UK)

AIX news

Macromedia has announced Version 1.5 of Flex, its presentation server and application framework, which is designed to enable enterprise development teams to put more effective interfaces on critical business applications.

The product has improved data display and visualization, more versatile skinning and styling, additional new deployment platforms, and enhanced performance.

Flex 1.5 now includes support for AIX, Oracle Application Server 10g, and Fujitsu Interstage 6 on the server side, while on the client side developers can deploy their applications to the browser or the Macromedia Central desktop deployment environment.

For further information contact:
URL: www.macromedia.com/go/flex.

* * *

NEC Solutions (America) has announced the NEC Tricryption Data Security System, a three-layered data encryption solution.

NEC's Tricryption System integrates with databases, operating platforms, and enterprise applications. The File Protection client is designed to operate in all Microsoft Windows environments; and the Database Protection element works with leading databases, such as Microsoft SQL, Oracle, and DB2, as well as major operating platforms, including AIX, Linux, Solaris, HP-UX, and Windows.

For further information contact:
URL: www.necsam.com/Tricryption.

* * *

Willow Technology has announced its Ectropyx integration platform for sensors. Ectropyx is designed to provide online, secure, and reliable integration between sensors and enterprise applications and databases, allowing centralized administration and control of sensors and controllers, as well as the processing of collected sensor data. For sensors such as RFID readers, this permits true online transaction processing.

Supported environments include WebSphere MQ, WebSphere Integration Broker, and JMS providers such as WebSphere, BEA WebLogic, and the open source JBoss Application Server. Supported Integration Server platforms are AIX, HP-UX, Linux, Solaris, and Windows.

For further information contact:
URL: www.willowtech.com.

* * *

There's a security warning about the ctstrtcasd command. Local exploitation of an input validation vulnerability in the command could allow for the corruption or creation of arbitrary files anywhere on the system.

The ctstrtcasd program is a setuid root application, installed by default under newer versions of AIX. It is part of the Reliable Scalable Cluster Technology (RSCT) system.

If a user specifies a file with the -f option, the contents of that file will be overwritten with 65,535 bytes of application trace data. If the file doesn't exist, it will be created. The file creation/overwrite is done with root privileges, thus allowing an attacker to cause a denial of service condition by damaging the file system or by filling the drive with 65,535 byte files.

For further information contact:
URL: http://www.idefense.com/application/poi/display?id=144&type=vulnerabilities.