Making Backups of Mirrored Filesystems on AIX 3.2

AIX

By Jaime Vazquez

The AIX Version 3.2 Logical Volume Manager's capabilities were enhanced with the addition of filesystem mirroring. This article shows how to make backups of these mirrored filesystems—making use of one of the mirrored copies.

Although mirroring greatly increases the availability and robustness of filesystems on AIX, it is not meant to be a mechanism for offline backups. Any filesystem that has update activity occurring cannot be safely backed up and have complete assurances about the state of the backup. This is, of course, not the case for any filesystem mounted read-only. Still, a system administrator should take adequate measures to fully safeguard critical data. This may not be possible on systems that have applications or data that must be online continuously or offline for only a very short time.

LVM Background

The Logical Volume Manager (LVM) allows logical data partitions to be mapped to any physical partition available to the volume group. LVM carries within it a mapping structure that shows the relationship between a logical file partition and the physical disk partition that holds the data. The user can set the partition size when the volume group is created, with values of 1 MB to 128 MB (in powers-of-2 increments). The default value is 4 MB. A physical partition is the smallest unit by which a filesystem can be expanded.

Each LVM listing command has a suboption to display the mapping data at all levels. For example, the `lslv` command has the `-m` option for describing the logical-to-physical mapping for a specified logical volume. Figure 1 shows an example of the output. It is also possible to create a logical volume in which the user specifies exactly where logical volumes are located in relation to physical partitions and physical volumes. The allocation map file can be used as input to `mklv` for this purpose. The allocation map is an ASCII file that describes the disk and partition number for the `lv`, and the order in which the `lv` will be created. Figure 2 shows an example of such a file.

When a filesystem or logical volume is deleted, its physical partitions are released to the free list, but the data still physically resides on the disk; it is a logical deletion, not a true physical deletion. By extracting the mapping information of the logical volume or its copy and using this information as input to the `mklv` command, it is possible to re-create a filesystem using the same physical partitions in the same order as the original. The filesystem structure remains intact. From the time

```
$ lslv -m hd1
hd1:/home
LP    PP1  PV1               PP2  PV2
PP3  PV3
0001  0038 hdisk2            0019 hdisk3
0002  0042 hdisk2            0020 hdisk3
0003  0043 hdisk2            0021 hdisk3
0004  0005 hdisk2            0022 hdisk3
```

Figure 1. Sample output from the lslv command

```
hdisk3:0019
hdisk3:0020
hdisk3:0021
hdisk3:0022
```

Figure 2. Example of allocation map file

May '93
p.64

Jaime Vazquez

```
#!/bin/ksh
# User must have root authority.
# Input to the script is the logical volume name device to
be backed up.
#
if [ $# -ne 1 ]
then
   echo “Logical Volume name not specified on command line.”
   echo “Syntax: $0 <lv-name>  “
   exit 1
fi
# Make sure this is a valid logical volume
lslv $1 > /dev/null 2>&1
if [ $? -eq 1 ]
then echo “Logical volume $1 not found.”
     exit 1
fi
# Check if the filesystem is mounted.
# It must be unmounted!
FSNAME= getlvcb -L $1
mount | grep  $FSNAME > /dev/null 2>&1
if [ $? -eq 0 ]
then echo “Filesystem is still mounted. “
     echo “Unmount filesystem and try again. “
     echo “Process terminated.”
     exit 1
fi
echo “Filesystem “ $FSNAME “ is being processed.”
LVID= ~getlvodm -l $1~
VGNAME= ~getlvodm -b $LVID~
NUM= ~getlvcb -c $1~
if [ NUM -eq 1 ]
then echo “Filesystem is not mirrored. Process aborted.”
     exit 1
elif [ NUM -eq 2 ]
then lslv -m $1 | grep hdisk | awk ‘{ print $5”:”$4 }’ >
allocmap.file
     NEWNUM=1
else lslv -m $1 | grep hdisk | awk ‘{ print $7”:”$6 }’ >
allocmap.file
     NEWNUM=2
fi
MAPSIZE=~wc -l allocmap.file | awk ‘ {print $1} ‘ ~
echo “Allocation map file created of size of “ $MAPSIZE “
partitions.”
sync;sync
rmlvcopy $1 $NEWNUM
if [ $? -ne 0 ]
then echo “rmlvcopy command failed with rc = “ $?
     echo “Process terminated.”
     exit 2
fi
echo “Logical volume “ $1 “ has been reduced to “ $NEWNUM “
```

Note: This sample code is provided as-is without any warranty or guarantee from IBM.

Figure 3. Shell script to automate backup steps 1 through 4

the mirror copy is deleted until the image logical volume is created, there must be no changes to the global state of filesystems on that volume group. Deleting the mirror copy releases physical partitions to the free pool. These partitions can then be used for other filesystem actions, such as expanding the size of a filesystem.

Making the Backup

The steps to make the backup are as follows:

1. Create an allocation file of the physical partitions used in the most secondary mirror copy.
2. Unmount the filesystem.
3. “Break” a mirror by deleting the secondary mirror copy.
4. Re-create a logical volume using the allocation file from step 1.
5. Mount the new logical volume read-only.
6. Perform the backup.
7. Unmount and delete the created logical volume.
8. Add a mirror copy to the filesystem, reusing the allocation file.
9. Resynchronize the logical volume.

In the current implementation of LVM, breaking the mirror can be accomplished only by deleting one of the copies. The resulting free space still contains the original data. By reusing the same physical partitions in the exact order as the mirror copy, the filesystem still contains the original data.

The shell script in Figure 3 shows how to automate steps 1 through 4. It can be modified as required. Using the name `backuplv` for the created logical volume is not required. The created logical volume can be mounted on any mount point, not just `/mnt`.

After the shell script has been run, the mirror copy has been mounted and is available to be backed up by any means desired, such as `tar`, `cpio`, `pax`, and `backup`. It is important to note that there is one less mirror copy available to the user. If the logical volume had only itself and one mirror copy, the system would use only the original copy for the backup process. Normal user processing can begin on the original data.

The shell script in Figure 4 resets the logical volume structure to its original state. At this point, the system has returned to its pre-backup condition.

AIX Book Corner

The following new books may be of interest:

Power RISC System/6000: Concepts, Facilities, and Architecture, by Dipto Chakravarty. New York: McGraw-Hill, Inc., 1994 (ISBN 0-07-011047-6).

Part of the J. Ranade Workstation Series, this book describes RISC System/6000 hardware and architecture, as well as AIX architecture and administration. The book is filled with helpful information and tips. Mr. Chakravarty has been a contributing author to *AIXpert*.

The Whole Internet User's Guide & Catalog, by Ed Krol. O'Reilly & Associates, Inc., 1994. Second Edition (ISBN 1-56592-063-5).

This edition is an update to the popular first edition that was published in September 1992. It is divided into three sections: history and ethics of the Internet, Internet tools, and the catalog of information available on the Internet. The new edition includes looks at Mosaic, mail programs, ftp servers, xarchie, Gopher, and much more.

IBM First in U.S. Patents Awarded

IBM ranked first in the number of patents awarded in 1993 by the U.S. Government, marking the first time since 1985 that an American company has headed the list.

IBM received 1,088 patents, followed by Toshiba® Corporation, Canon® KK, and Eastman Kodak® Company, according to IFI/Plenum Data Corporation. In 1992, the first four companies were all Japanese.

The IBM patents were almost exclusively in the field of information processing, with an increase in software-related inventions. ■

Caveats to the Backup Process

This backup procedure has some areas that must be understood before it is implemented.