About Sparse Files

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 describes sparse files, exposure due to sparse files, and the effects of certain commands on sparse files. This document applies to AIX 3.2.

Overview

Many applications, particularly databases, maintain data in sparse files. A sparse file is a file with empty space, or gaps, left open for future addition of data. If the empty spaces are filled with the ASCII null character and the spaces are large enough, the file will be sparse, and disk blocks will not be allocated to it.

This creates an exposure: a large file will be created, but the disk blocks will not be allocated. Then, as data is added to the file, the disk blocks will be allocated but there may not be enough free disk blocks in the file system. Then the file system will be full and writes to any file in the file system will fail.

You can prevent these problems by either assuring that you have no sparse files on your system or by planning to have enough free space in the file system for the future allocation of the blocks.

You also need to be aware of how you manipulate sparse or potentially sparse files because you can easily change them from sparse to not sparse or vice-versa.

Creating a Sparse File

An example sparse file can be created fairly easily. To do this, open the file, seek to a large address, and write some data. This can be demonstrated with the dd command, as follows:

1. First, create a regular file:

```
date > notsparse
ls -l
```

The output of the ls command will be similar to:

```
total 8
-rw-r--r--   1 root     sys           29 Dec 21 08:12 notsparse
```

2. Use the fileplace command to see how many allocated and unallocated blocks are included in the file "notsparse".

   (Note: BOS Extensions 1/Extended Commands [bosext1.extcmds.obj] must be installed to enable the fileplace commands or perfagent.tools for 4.1.)

   ```
   fileplace notsparse
   ```

   The output for 3.2 will look similar to:

   ```
   File: notsparse  Size: 29 bytes  Vol: /dev/lv03 (4096 byte blks)

     Logical blocks
     --------------
     00016                          1 blk,      4 KB,  100.0%
   ```

   (Note: BOS Extensions 1 [bosext1.extcmds.obj] must be installed to enable the fileplace command.)

   The output for 4.1 will look similiar to:

   ```
    File: notsparse  Size: 29 bytes  Vol: /dev/lv03
    Blk Size: 4096  Frag size: 4096 Nfrags: 1 Compress: no

     Logical Fragment
     ----------------
     00716                   1 frags         4096 bytes,  100.0%
   ```

   (Note: Performance Analysis and Control Commands [perfagent.tools] must be installed to enable the fileplace command for 4.1.)

3. The du command will also reflect how many 512-byte blocks a file occupies.

   ```
   du -rs *
   ```

   Example output:

   ```
   8 notsparse
   ```

4. Now create a sparse file using the regular file "notsparse" as input:

   ```
   touch sparse.1
   dd if=notsparse of=sparse.1 seek=100
   ```

   Example output:

   ```
   dd: 0+1 records in.
   dd: 0+1 records out.
   ```

   The dd command takes the data from the regular file and places it 100 512-byte blocks into the "sparse.1" file. Note that nothing is written to the

initial 99 512-byte blocks. The following steps show the characteristics of the resulting file.

5. The ls command reports the distance from block zero to the last block in the file:

   ```
   ls -l
   ```

   Example output:

   ```
   total 16
   -rw-r--r--   1 root     sys           29 Dec 21 08:12 notsparse
   -rw-r--r--   1 root     sys        51229 Dec 21 08:13 sparse.1
   ```

6. The fileplace command tells the story accurately — there are 12 unallocated 4K blocks and one allocated 4K block in the file:

   ```
   fileplace sparse.1
   ```

   Example output for 3.2:

   ```
   File: sparse.1  Size: 51229 bytes  Vol: /dev/lv03 (4096 byte blks)

     Logical blocks
     --------------
     unallocated                  12 blks,    48 KB,  92.3%
     00018                         1 blk,      4 KB,   7.7%
   ```

   Example output for 4.1

   ```
   File: sparse.1  Size: 51229 bytes  Vol: /dev/lv03
   Blk Size: 4096  Frag Size: 4096  Nfrags:  1   Compress: no

   Logical Fragment
   ----------------
   unallocated                   12 frags   49152 Bytes,  0.0%
   0000769                        1 frags    4096 Bytes, 100.0%
   ```

7. The du command reports the number of allocated blocks the file takes:

   ```
   du -rs *
   ```

   Example output:

   ```
   8 notsparse
   8 sparse.1
   ```

The Effect of Certain Commands on Sparse Files

backup/restore (by name and inode)

The restore command aggressively preserves sparseness. In fact, the restore command will unallocate any blocks filled with zeroes, thus making a file sparse.

Currently there is a defect in the restbynode command where inode backups do not preserve sparseness on the restore. The fix is ix50200, while the

workaround calls for the use of a version of restbyinode with the sccs ID for tape.c at 1.25.1.1.

| Note: Users with 4.1.3 or later will not need this fix, however, users with earlier
| versions of 4.1 will need ix50815.

To order the fix via voice, call 1-512-823-9468 or 1-817-961-6077. To request a fix via fax send a fax message to 1-800-225-5249 referencing the ix number above. Finally, Internet users can obtained electronic download instructions from IBM 4FAX (1-800-426-4329). Request Index number 1228
1228 "FixDist - Electronic Fixes for AIX on the Internet".

cp

The cp command does not preserve the sparseness of a file.

cpio

If you create a backup using the cpio command on sparse files, you will need to use the pax command to restore that data. Using the cpio command to restore the data will not preserve sparseness.

Note: The pax command can only read cpio backups if the cpio "c" flag was used when the backup was created. If the "c" flag was not used, the pax command will not be able to restore the data.

| Note: This pax limitation is no longer valid for 4.1.

Workaround If "c" Flag Not Used

1. Restore the file(s) to disk.
2. Back up the file(s) with one of the following:
   - backup (by name or inode)
   - cpio with the "c" flag
   - tar
3. Verify that you have a good backup by listing a table of contents from your backup:
   - If tar or cpio was used:

     ```
     pax -vf /dev/rmt#
     ```

   - If backup by name was used:

     ```
     restore -Tvf /dev/rmt0
     ```

   - If backup by inode was used:

     ```
     restore -tvf /dev/rmt0
     ```

4. Remove the file(s) you restored.
5. Now, restore the file(s) from the newer backup (that you just made).

   Note: If you used cpio or tar, restore with pax.

dd

Using the dd command on the file itself does not preserve sparseness. However, using dd on the file system device does preserve the state of the individual files.

Example: Backing up a logical volume:

```
dd if=/dev/datalv of=/dev/rmt0 ibs=4096 obs=1024 conv=sync
```

mksysb

The mksysb command in AIX 3.2 uses tar to back up the mounted JFS file systems in the rootvg. See the section on tar. Mksysb uses the pax command to restore the system. See the section on pax. The mksysb command in AIX 4.1 uses backup/restore. See the section on backup/restore.

pax

Note: The pax command can read tar archives and can read cpio archives if the "c" flag was used.

The pax command aggressively preserves sparseness. In fact, the pax command will unallocate any blocks filled with zeroes, thus making a file sparse.

sysback

Sysback will use either backup by name or inode to backup the data on the system. See the section on backup.

tar

If you create a backup using the tar command on sparse files, you will have to use the pax command to restore that data. Using the tar command to restore the data will not preserve sparseness.

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:

11/27/95, 4FAX 3753

END OF DOCUMENT (sparse.files.32.bak, 4FAX# 3753)