DFU V2.2

DFU Disk and File Utilities for OpenVMS VAX and AXP

Revision Information:	This manual supersedes V2.1
Date:	October 1995
Operating System:	OpenVMS VAX V5.5 thru 7.0 ,AXP V6.1 thru 7.0
Note:	Part of the Freeware CD-ROM

OpenVMS Freeware for VAX and AXP Systems

The OpenVMS Freeware CD-ROM contains free software tools and utilities to aid software developers in creating applications and managing and using OpenVMS systems.

Many of these tools are popular packages already widely known and in use, while others are internally developed Digital tools our engineers are making available to our OpenVMS customers. For example, the OpenVMS Freeware CD-ROM includes Bliss-32 for OpenVMS VAX, Bliss-32 for OpenVMS AXP, and Bliss-64 for OpenVMS AXP.

Disclaimer on Testing, Quality, and Licensing

The OpenVMS Freeware for VAX and AXP Systems CD-ROM is provided "AS IS" without warranty. Digital imposes no restrictions on its distribution, nor on the redistribution of anything on it. Be aware, however, that some of the packages on the CD-ROM may carry restrictions on their use imposed by the original authors. Therefore, you should carefully read the documentation accompanying the products.

Digital Equipment Corporation is providing this software free of charge and without warranty. Included in the [FREEWARE] directory on this CD are the GNU software licenses. Unless otherwise stated in sources or source packages, the GNU licenses cover all software supplied on this CD.

Digital makes no claims about the quality of this software. It is provided to the customer as a free service.

Copyright ©1995

Contents

CHAPTER 1 INTRODUCTION 1–1

1.1 SUPPORT ON DFU 1–1

1.2 GENERAL DESCRIPTION 1–1

1.3 NEW FEATURES OF V2.2 1–2

1.4 RESTRICTIONS AND PARAMETERS 1–3

CHAPTER 2 USING DFU 2–1

2.1 INSTALLATION PROCEDURE 2–1

2.2 SECURITY AND ACCESS 2–2

2.3 RUNNING DFU 2–3

2.4 USING AN INDIRECT FILE LIST 2–3

CHAPTER 3 THE SMG OUTPUT INTERFACE 3–1

3.1 ENABLING/DISABLING SMG 3–1

3.2 AUTOMATIC SWITCHING BETWEEN SMG AND LINE MODE 3–1

3.3 LAYOUT OF THE SMG SCREEN 3–1

3.4 SPECIAL SCREEN COMMANDS 3–2

3.5 SIDE EFFECTS 3–3

CHAPTER 4 DFU BASICS 4–1

CHAPTER 5 DEFRAG 5–1

5.1 DESCRIPTION 5–1

5.2 ERROR MESSAGES 5–1
DEFRAG 5–1

CHAPTER 6 DELETE 6–1

6.1 INTRODUCTION 6–1
DELETE 6–1

CHAPTER 7 DIRECTORY 7–1

7.1 INTRODUCTION 7–1

7.2 DISCUSSION OF DIRECTORY COMPRESSION 7–2
DIRECTORY 7–3

CHAPTER 8 REPORT 8–1

8.1 DESCRIPTION 8–1

8.2 SYNTAX AND OUTPUT 8–1
REPORT 8–3

CHAPTER 9 SEARCH 9–1

9.1 INTRODUCTION 9–1

9.2 SYNTAX AND OUTPUT 9–1

9.3 VOLUME SET PROCESSING 9–1
SEARCH 9–1

CHAPTER 10 SET 10–1

10.1 INTRODUCTION 10–1
SET 10–1

CHAPTER 11 UNDELETE 11–1

11.1 INTRODUCTION 11–1
UNDELETE 11–3

CHAPTER 12 SPAWN 12–1

CHAPTER 13 VERIFY 13–1

13.1 INTRODUCTION 13–1

13.2 ERROR REPORTING 13–2
VERIFY 13–6

1 Introduction

The Disk and File Utilities (DFU) is a high perfomance tool meant as an aid to the system manager who is responsible for monitoring and maintaining disks, files and directories. It makes use of low level I/O features of VMS. As a result a high level of performance is reached with minimal overhead on the system. DFU supports all types of disk sets which comply to the VMS ODS-2 standard such as volume sets, stripe sets , shadow sets and RAID sets, and combinations of these.

1.1 Support on DFU

Although there is no official support on DFU you may send questions/remarks/bug-reports to dorland@utrtsc.enet.dec.com.

1.2 General description

DFU provides the following functions (in alphabetical order) :

- DEFRAG : This function allows simple defragmentation of a file or a list of files.
- DELETE : This function allows either a delete by file-id, or a fast delete of a complete directory or directory tree with all its subdirectories.
- DIRECTORY : This function has 7 options :
  1. COMPRESS a directory
  2. DUMP a directory block by block
  3. CREATE a directory with a preallocated size
  4. Search all directories on the disk for files which have multiple versions
  5. Search all directories on the disk for alias file entries
  6. Search all directories on the disk for empty directories
  7. Search all directories on the disk for problem directories.
- REPORT : Generates a file and free space report for a disk. Also a disk space usage report, based on UIC/Identifier, can be generated, even on a disk which has disk quota disabled.
- SEARCH : Look up files on a disk by specific file attributes (eg. files sizes, dates, ownership, and so on)
- SET : This functions allows setting of file attributes which are not available through DCL commands.
- UNDELETE : A safe file recovery function.

- VERIFY : This function performs a fast disk structure verification and can optionally repair certain disk structure errors (comparable with ANALYZE/DISK/REPAIR).

Each function is described in detail in the corresponding chapter.

DFU is supported on OpenVMS VAX V5.5 through V7.0, and OpenVMS AXP V6.1 thru V7.0. DFU has been tested on the Field test versions of VMS 7.0; so far no problems have been found. The official freeware release of DFU will distributed with the official release of VMS V7.0.

1.3 New features of V2.2

This version of DFU offers several improved and new features, as well as a couple of bugfixes. The most important new feature is the implementaion of Screen Management (SMG) routines to perform all output to the user. Secondly , the DIRECTORY/COMPRESS command has been enhanced such that the user has more control over the compress process. The table below lists all new features :

- SMG oriented output : Optionally DFU uses SMG routines to perform a window-style output. See chapter 3 for a description of the SMG capabilities.
- DIRECTORY/CHECK has been improved. Also, the new /REPAIR qualifier allows automatic removal of all invalid directory entries. Such entries are usually seen with a 'nosuchfile' error.
- DIRECTORY/VERSION now also shows the number of blocks of the found files.
- DEFRAG file : added /LBN to force moving a single file to a specific LBN (logical block number).
- SEARCH/CHARACTERISTICS : two new characteristics , /NOSHELVABLE and /ISSHELVED are added. These can be used on a system where file shelving is used.
- SEARCH/VERSION_NUMBER : this qualifier allows a search for files with a version number within a specific range ,eg. SEARCH/VERSION=MIN=30000 produces all files with a version number of at least 30,000.
- SEARCH/NOOWNER, /NOIDENT : this allows searching for files NOT owned by a certain UIC or Identifier.
- SET : added /OWNER_UIC and /IDENT qualifiers. This allows setting of file ownership for open files such as INDEXF.SYS.
- SET : like DEFRAG and DIRECTORY SET now also allows the usage of a indirect file as input for the SET command. Such a file can be produced with SEARCH/OUTPUT.
- SPAWN : this new command creates a subprocess and thus allows a quick escape to DCL from an interactive DFU session.
- Different /OUTPUT behaviour : when used /OUTPUT=file the output will be send to BOTH the file and the screen.

The following bugs are fixed :

- Under rare circumstances (usually seen during a minimum boot) DFU did not recognize that the DFU_ALLPRIV identifier was granted.
- The filename reported from a DIRECTORY/CREATE command could contain some bogus characters at the end.
- The calculation for the File Fragmentation Index in the REPORT command sometimes gives incorrect results. This was especially seen when a a disk contained a small number of small but fragmented files such as batchjob log files. This would result in a much too high value.
- Some commands such as UNDELETE and DEFRAG now check if the disk is write-locked or if erase-on-delete is set BEFORE performing any action. This results in better error reporting.
- The output filename from the SEARCH command on a volume set always gave the device-name of the volume set member currently being processed. Although this is not incorrect, RMS can't handle such a file name. Therefore using SEARCH/OUTPUT on a volume set and using the results in a DIRECTORY or DEFRAG command produced RMS-E-FNF errors. The behaviour is changed such that filenames are now given with the device name of the primary (RVN 1) member of the set, thus allowing RMS to find these files.
- Fixed a bug which could lead to an Access Violation error on all commands which do INDEXF.SYS processing. This error was rarely seen but showed up during testing on AXP F7.0.

1.4 Restrictions and Parameters

The DFU program has a few restrictions :

- DFU can not handle volume sets with more than 32 members.
- Due to changes in VMS run time libraries on VAX DFU.EXE files linked on VMS 6.2 cannot run on VMS 6.1 and lower without relinking the image.
- The SMG interface may not work at all on VMS 5.5. In that case you need to install CSCPAT_0320 and re-install DFU. Warning: CSCPAT_0320 replaces SMGSHR.EXE and may have side effects on other applications using SMG routines.

The following minimum process quotas are recommended for DFU usage:

- WSQUOTA : 1500
- WSEXTENT : 3000
- DIOLM : 40
- ASTLM : 40
- FILLM : 40
- BYTLM : 30000

- PGFLQUOTA : 30000

If you use DFU on volume sets with a large number of members you may need to increase the SYSGEN parameter CHANNELCNT.

If you use the directory command on very large directory files (> 1000 blocks) and run into errors you may need to increase the following parameters:

- WSQUOTA at least 1000 + size-of-largest directory file
- WSEXTENT : WSQUOTA + at least 1000
- CTLPAGES (SYSGEN parameter) : at least 100
- WSMAX (SYSGEN parameter) : at least WSEXTENT

Do not change parameters unless you run into errors such as EXCEEDED QUOTA or INSUFFICIENT WORKING SET LIMITS.

2 Using DFU

This chapter describes the installation procedure and other actions which must be taken to run DFU.

2.1 Installation procedure

DFU can be installed with the VMSINSTAL utility. During the installation VMSINSTAL does the following:

- Moves the documentation to SYS$HELP (Optional)
- Relinks DFU
- Creates the DFU_ALLPRIV identifier needed to perform the write functions. (See 2.2)
- Asks if you want a systemwide or private installation

The DFU utility in fact contains just 2 files :

1 DFU.EXE : the DFU program

2 DFUHLP.HLB : the on-line help file

The location of these files depend on wether a systemwide or private installation is performed.

To install DFU log in into a privileged account and enter :

```
$ @sys$update:vmsinstal
  VAX/VMS Software Product Installation Procedure V6.2
  It is 02-Oct-1995 at 13:46.
  Enter a question mark (?) at any time for help.
  * Are you satisfied with the backup of your system disk [YES]?
```

VMSINSTAL now asks for the distribution media and product. You can also specify the directory where you have the DFU_FREEWARE_022.A file instead of a device name:

```
  * Where will the distribution volumes be mounted: [DDCU:]
  Enter the products to be processed from the first distribution volume set.
  * Products: DFU_FREEWARE_022
  * Enter installation options you wish to use (none):
  The following products will be processed:
         DFU_FREEWARE_ 2.2
  Beginning installation of DFU_FREEWARE_ V2.2  at 13:46
  %VMSINSTAL-I-RESTORE, Restoring product save set A ...
  Installing DFU version 2.2
```

```
The next question will prompt  you for the documentation files.
The DFU documentation is available in Line printer, PostScript and
Bookreader format. The documentation files will be copied to SYS$HELP.

* Do you want the documentation in Lineprinter format ? [Y]?
%VMSINSTAL-I-DOCMOVED, DFU022.LINE moved to SYS$HELP

* Do you want the documentation in PostScript(TM) format ?   [N]?
* Do you want the documentation in Bookreader format ? [N]?
```

VMSINSTAL now links the DFU image and adds the identifier to the Rightslist (see 2.2 for the meaning of the identifier).

```
Linking DFU image ...
%DFU-I-LINK, Linking DFU 2.2  image...
%DFU-S-DONE, DFU.EXE created
```

Next VMSINSTAL will create the DFU_ALLPRIV identifier if necessary.

```
%VMSINSTAL-I-ACCOUNT, This installation adds an identifier named
  DFU_ALLPRIV.
%UAF-I-RDBADDMSG, identifier DFU_ALLPRIV value: %X80010008 added to
  rights data base

%VMSINSTAL-I-MOVEFILES, Files will now be moved to their
 target directories...

Installation of DFU V2.2  completed at 13:47
```

You must grant any DFU user including yourself the necessary DFU_ ALLPRIV identifier if a user be allowed to use all DFU functions. See 2.2 for the meaning of this identifier.

2.2 Security and access

DFU has some functions which must not be made available for general usage. Therefore DFU has 2 groups of functions :

1 Read-only functions

   1 DIRECTORY (except DIRECTORY/COMPRESS)

   2 REPORT

   3 SEARCH

   4 SPAWN

   5 VERIFY (except VERIFY/FIX and VERIFY/REBUILD)

2 Write functions

   1 DEFRAG

   2 DELETE

   3 DIRECTORY/COMPRESS

   4 SET

   5 UNDELETE

   6 VERIFY/FIX and VERIFY/REBUILD

The Read-only functions need SYSPRV or BYPASS privilege. The Write functions require BYPASS privilege as well as the DFU_ALLPRIV identifier. The DFU_ALLPRIV identifier is created during the installation procedure. To enable a user to perform Write functions the system manager must grant this identifier as follows :

```
$ mc authorize
UAF> grant/id DFU_ALLPRIV username
```

It is highly recommended that Write functions are NOT made available to the world.

2.3 Running DFU

To run DFU you can enter :

```
$ RUN disk:[mydir]DFU
```

or (if DFU is installed systemwide):

```
$ MC DFU
```

You can also define and use a symbol for DFU as follows:

```
$ DFU == "$DFU"
$ DFU
```

DFU now prompts with :

```
    Disk and File Utilities for OpenVMS DFU V2.2
    Freeware version
    Copyright (C) 1995 Digital Equipment Corporation

DFU>
```

You can now enter DFU commands. HELP gives you online help. After each command DFU returns to the DFU> prompt. You can recall and edit commands, just as with DCL (20 commands will be stored in the recall buffer). Use EXIT or CTRL/Z to leave DFU. If you use a symbol for DFU , you also process a single command and return to DCL immediately:

```
$DFU SEARCH disk/qualifiers
```

During a DFU run , you can interrupt DFU with CTRL/C or CTRL/Y. These interrupts are catched by DFU to allow a correct rundown of the current command. It may therefore take some time before DFU responds to a CTRL/C or Y interrupt.

2.4 Using an indirect file list

There are 3 commands, DIRECTORY, DEFRAG and SET which accept an indirect file as input parameter. This is achieved by using the @ sign. An indirect file is a file containing a list of valid filenames (each line in such a file must contain exactly one file). Such indirect files can be the output file of the DFU Search command eg.:

```
$ DFU
DFU> SEARCH mydisk/CHAR=DIRECTORY/OUTPUT=x.x
DFU> DIRECTORY/COMPRESS @x.x
```

3 The SMG output interface

As of version V2.2 DFU uses SMG (Screen management routines) to perform output for an interactive DFU session. This chapter describes the layout of the DFU screen, the special commands used only in the SMG interface and other items with respect to the SMG interface.

3.1 Enabling/Disabling SMG

DFU automatically selects SMG output if you run an interactive DFU session on a terminal or workstation which supports SMG. In a batch job or a non-DEC terminal DFU will use normal line oriented output. You can explicitly disable SMG by defining the logical DFU$NOSMG to any value.

3.2 Automatic switching between SMG and Line mode

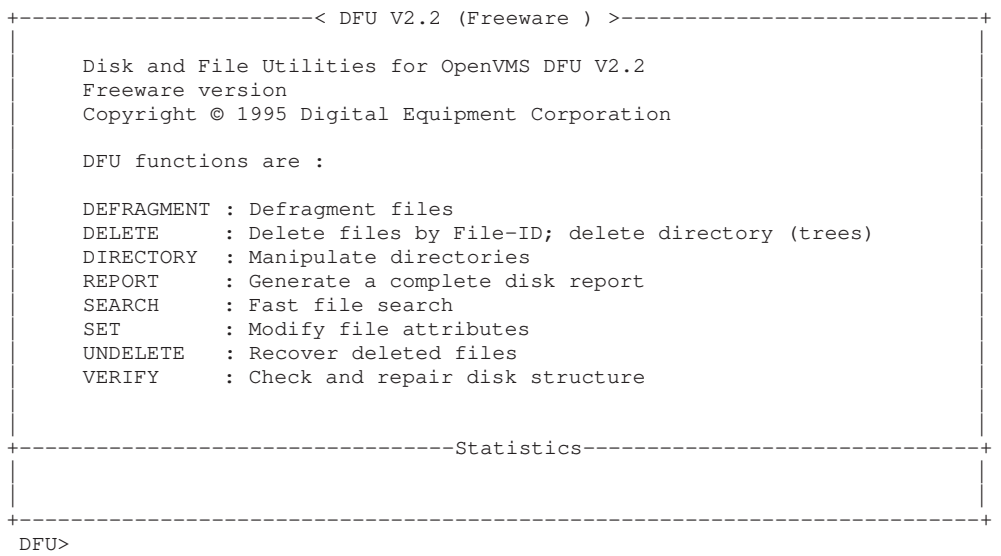
You may prefer to perform interactive DFU sessions in SMG mode, and single DFU commands in normal line mode. This can be done by using the command procedure listed below:

```
$! Procedure to switch beteen SMG and line mode
$ DFU = "$DFU"
$ if p1 .eqs. ""
$ then
$!Interactive seesion
$ def/user sys$input sys$command
$ dfu
$ else
$!Single command so disable SMG
$ def/user dfu$nosmg x
$ dfu 'p1' 'p2' 'p3' 'p4' 'p5' 'p6' 'p7' 'p8'
$ endif
$ exit
```

3.3 Layout of the SMG screen

After starting DFU the screen looks as follows :

The SMG output interface



This screen is divided into 3 sections:

- Main Screen : the top part of the screen is used for the output from DFU commands. Also broadcast messages are send to this window.
- Statistics : the lower part contains statistics from the current command. During most of the commands this screen will contain a progress indicator and a status for the current command.
- DFU prompt : the bottom line is used as the input command line.

3.4 Special Screen commands

In SMG mode the following keys perform special functions :

- DO : invokes a subprocess (same as the SPAWN command)
- HELP or PF2 : invokes a special SMG window for the help command.
- CTRL/W : redraws the screen
- PF4 : creates a screen dump into the file DFU_SCREEN.TXT
- PREV SCREEN : Scrolls back the output screen.
- NEXT SCREEN : Scrolls forward the output screen
- SELECT : Toggles between 80 and 132 columns.

Although only some 20 lines are visible in the output (main) screen DFU in fact remembers 500 output lines. You can scroll back and forward through these lines with the NEXT and PREV SCREEN keys. At the beginning of each new DFU command these lines are cleared.

On a workstation with DecWindows/Motif you can change the DecTerm size to be more than the standard 24 lines. DFU's SMG screen will automatically adjust to the screen size of such a DecTerm.

3.5 Side effects

There are a few side effects when using SMG mode. CTRL/C and CTRL/Y are catched but not echoed. Also all broadcast messages are trapped and will be displayed in the main screen at the end of the current DFU command. Sometimes this may slightly disrupt the main screen, which can be repaired with the CTRL/W key.

Further, during the execution of a DFU command the cursor will disappear. After the command finishes the cursor will reappear at the DFU> prompt.

4 DFU basics

Note : this chapter is meant for first time users of DFU. If you already used DFU before you may probably skip this chapter.

This chapter gives a brief overview of the commands which can be used with DFU, and how they can provide the system manager with the necessary information.

One of most common actions by a system manager is trying to locate specific files eg. files with a certain file size. Although a lot can be done with the VMS DIRECTORY command this can be a time consuming process , especially if the complete disk much be scanned. The SEARCH command of DFU is exactly meant for this situation. It gives a very quick list of specific files, with their size, and if needed the number of file fragments. Some usefull SEARCH commands are :

```
DFU> SEARCH disk/SIZE=MINIMUM=1000 (files > 1000 blocks)
DFU> SEARCH disk/IDENT=SYSTEM (files owned by SYSTEM)
DFU> SEARCH disk/FRAGMENT=MINIMUM=10 (files with at least 10 fragments)
DFU> SEARCH disk/CHAR=MARKED (all files marked for delete)
DFU> SEARCH disk/CREATED=SINCE=YESTERDAY (all files created since YESTERDAY)
```

Of course these qualifiers can be combined to narrow the search. As the SEARCH command scans INDEXF.SYS , not directories, the output generated by SEARCH may look a bit random (but it is just the order in which the file headers are in the INDEXF.SYS file). To smooth the output use the /SORT qualifier.

Another important action for a system manager is to gather some overall disk information , such as number of files , free space statistics and fragmentation information. The REPORT command will generate such a disk report. To get individual disk space usage information the /USAGE qualifier can be used. This will work even if there is no disk quota enabled on the disk.

At regular intervals a system manager may need to check a disk with ANALYZE/DISK. The DFU VERIFY command provides more or less the same functionality but many times faster. The /FIX qualifier provides some REPAIR options which (unlike ANALYZE/DISK/REPAIR) do NOT lock the disk, and thus do not interfere with other disk operations. However, the disk can be rebuild completely by using the /REBUILD qualifier.

Occasionaly a user will delete the wrong file, leaving the system manager or operator with the tedious task of restoring the file from a backup set. The UNDELETE command may be able to recover the file if it has not yet been overwritten with a new one. Much care has been taken to make this command as safe as possible; therefore no disk corruptions will be the result of an UNDELETE. If DFU encounters a problem with a deleted file it will refuse to UNDELETE it.

Another type of typical system manager problems deals with directories. The DFU command DIRECTORY can handle such problems. DIRECTORY can generate reports about empty directories (/EMPTY qualifier) or directories which contains files with a certain number of versions (/VERSION=n qualifier) or even check for corrupted directories (/CHECK). Also , directories may become internally fragmented. This can be quickly solved with the /COMPRESS qualifier.

A second problem with directories is in deleting a directory with many files. This may take several hours using a DCL DELETE *.* command. This problem is solved by using DFU's DELETE/DIRECTORY command which can delete large directories some 10 times faster than the normal DELETE command does. Also, using DELETE/DIR/TREE will delete a complete directory tree with just one command.

The last 2 commands deal with individual files. SET allows settings of some special file attributes. Since VMS V6.0 this functionality is also provided by the DCL SET FILE/ATTRIBUTE command, but DFU's SET offers some more options. The DEFRAG command is an interface to the MOVEFILE function offered by VMS 5.5 and higher. This allows a simple defragmentation of individual files.

The next chapters gives a detailed description of each individual DFU command.

5 DEFRAG

This chapter describes the DEFRAG command with the related parameters and qualifiers.

5.1 Description

The DEFRAG command can be used to defragment a single file or a group of files. You can also enter an indirect command file containing a list of files. This feature allows you to create an output file with SEARCH and use that as an input to DEFRAG.

DEFRAG makes use of the file-primitive MOVEFILE function. Therefore files which are open, or which are marked NOMOVE will not be DEFRAGmented. Still you should be cautious about using DEFRAG. You should not make this function available for general usage.

5.2 Error messages

The most common errors returned by DEFRAG are :

- ACCONFLICT: File is open by another user
- DEVICEFULL: Device has not enough contiguous free space to move the entire file
- FILNOTACC: File has been set to NOMOVE.
- FILENUMCHK: You tried to move a reserved file (eg INDEXF.SYS).
- RMS Errors: Almost always caused by an invalid or wrong file specifcation.

DEFRAG

Defragment a file or a list of files on the disk.

FORMAT **DEFRAG** *file1,file2,@file...*

DEFRAG command
DEFRAG

PARAMETERS *file1,file2,@file...*

The file(s) to be defragmented. You can also use an indirect command file (@file) which contains a list of files to be defragmented. This may be an output file generated with the SEARCH/OUTPUT command.

QUALIFIERS */BESTTRY*

Normally DEFRAG tries to create a contiguous file. If there is not enough contiguous free space the operation will fail. With the /BESTTRY qualifier DEFRAG tries to move the file with the contiguous-best-try method. This means a maximum of 3 fragments. You should only use this qualifier when a normal DEFRAG fails. If a DEFRAG/BESTTRY also fails this is an indication that the disks free space is too fragmented. Run a REPORT to see what the largest contiguous free space is, and check the free space fragmentation index.

/DEVICE

The device on which the files reside. The device name is added to the file name. If you generate a file list with the DFU SEARCH command you don't need this qualifier anymore. However you may have a list of files produced from another source and thus need to append a device name to it. Example:

```
DFU> SEARCH mydisk/...../frag=min=50/output=x.x
DFU> DEFRAG @x.x
```

/LBN=logical-block-number

Forces a single file to be moved to the specified logical block number. Do not use this qualifier when processing a list of files.

/STATISTICS

This qualifier displays the performance statistics: CPU time, Elapsed Time, I/O and PageFaults.

/WRITECHECK

Perform a writecheck on the resultant file. The default is /NOWRITECHECK. Note that a writecheck doubles the number of involved disk I/O's.

6 DELETE

This chapter describes the DELETE command with the related parameters and qualifiers.

6.1 Introduction

The DELETE command is designed for 2 purposes:

1 Delete a file by file id (with the /FILE qualifier). This functionality is needed to be able to delete files which no longer have a parent directory. This way 'lost' files and (sometimes) files marked for delete can be removed from a disk.

2 A fast method to delete a complete directory or even a directory tree (with the /DIRECTORY qualifier). First all entries of a directory are deleted without removing the directory entry ; this saves a lot of unnecessary directory updates. Finally the directory file itself is deleted. This way a directory is cleaned up 2 to 3 times faster than with the DCL DELETE *.* command. On large directories DFU may be even 10 times (or more) faster. To delete a complete directory tree use the /TREE qualifier. The DCL command DELETE [.SUBDIR...]*.*;* will almost always fail to delete the intermediate subdirectory files forcing you to issue this command several times. The DFU DELETE command is smart enough to delete the files and the subdirectory files in the proper order; therefore only one command is sufficient to delete a complete directory tree.

DELETE

Deletes files or directories.

FORMAT

DELETE *device[:] or directory-file(s)*

PARAMETERS

device

The device on which to perform the DELETE/FILE=file-id command.

directory-file(s)

The filespecification of the directory on which to perform the DELETE/DIRECTORY command. Wildcard file specifications may be used. DFU will automatically add .DIR to the filespecification if a filetype is not provided.

QUALIFIERS

/DIRECTORY

Specify the directory file which must be deleted. Wildcards may be used but must be used very carefully. Make sure that the directory does not contain non-empty subdirectories. Files within such a subdirectory will become lost after the DELETE/DIRECTORY command (but can be recovered with ANALYZE/DISK/REPAIR). Note that the directory file itself will also be deleted. Example:

```
$ DIR EXAMPLE

Directory MYDISK:[RUBBISH]

EXAMPLE.DIR;1

$ DFU
DFU> DELETE/DIRECTORY EXAMPLE.DIR

%DFU-I-CLEANUP, Deleting MYDISK:[RUBBISH]EXAMPLE.DIR;1...
%DFU-S-DELETED, File DBGINI.COM;3 deleted
%DFU-S-DELETED, File ICON.COM;1 deleted
%DFU-S-DELETED, File I_FOOL.COM;1 deleted
%DFU-S-DELETED, File LOGICALS.COM;51 deleted
%DFU-S-DELETED, File LOGIN.COM;79 deleted
%DFU-S-DELETED, File LOGOUT.COM;4 deleted
%DFU-S-DELETED, File NOTE_COMPRESS.COM;3 deleted
%DFU-S-DELETED, File SHOWCL.COM;3 deleted
%DFU-S-DELETED, File SYMBOLS.COM;89 deleted
%DFU-S-DELETED, File TPUBUILD.COM;5 deleted
%DFU-S-TOTAL, 11 file(s) deleted
%DFU-S-DELETED, File EXAMPLE.DIR;1 deleted
```

/FILE=file-id

Specify a complete file-id (num,seq,rvn) of the file which must be deleted. This qualifier allows the deletion of lost files or files marked for delete. Example :

```
$ DFU
DFU> VERIFY MYDISK:

%DFU-S-CHKHOME, Home block info verified OK
%DFU-I-IFSCAN, Scanning INDEXF.SYS...
%DFU-E-INVBAKFID, file (86,17915,1) 0004CF04$BFS.;1 has invalid backlink
%DFU-W-DELETED, file (537,2878,1) APPL_UTRYIT_BOOK.TMP;1 marked for delete
 .
 .
 .

DFU> DELETE MYDISK/FILE=(86,17915,1)
%DFU-S-DELETED, File 0004CF04$BFS.;1 deleted

DFU> DELETE MYDISK/FILE=(537,2878,1)
%DFU-S-DELETED, File APPL_UTRYIT_BOOK.TMP;1 deleted
```

/NOLOG

Do not log succesfull deletes to SYS$OUTPUT. /LOG is the default.

/NOREMOVE

Can only be used with /FILE. Default DELETE/FILE will try to remove the file from the parent directory. If the file's backlink no longer points to a valid directory this will generate an error. The /NOREMOVE qualifier overrules this behaviour allowing such files to be deleted.

/STATISTICS

This qualifier displays the performance statistics: CPU time, Elapsed Time, I/O and PageFaults.

/TREE

Can only be used with /DIRECTORY. The delete command will delete all subdirectories within the directory file specified. DFU will first sort out the subdirectory tree and than delete the files in the proper order. Thus a complete directory tree is deleted with one command.

DELETE

7 DIRECTORY

This chapter describes the DIRECTORY command with the related parameters and qualifiers.

7.1 Introduction

The DIRECTORY command offers 7 major functions. 4 functions deal with all directories on a disk, the other 3 functions are used for manipulating individual directories. These functions are activated by special qualifiers:

- Individual directory functions :

  1 /COMPRESS : Rebuild one or more directory files. Directories are alphabetically sorted lists of filenames and file id's. Because file names are randomly inserted and deleted, holes will occur in directory files. Such holes will only be reused if a file can be alphabetically fitted into it. As a result directory files will grow. Especially MAIL subdirectories suffer from this behaviour.

  To make matters worse directory files above 127 blocks will no longer be cached by the file system. This will degrade performance on such directories dramatically. The COMPRESS function will rebuild the complete directory file. The /TEST qualifier can be used to preview the results of a possible COMPRESS command.

  2 /CREATE/ALLOCATE=n : Create a directory with a preallocated size of 'n' blocks. Note that 'n' will be rounded up to a multiple of the disks cluster size.

  3 /DUMP : Make a dump of the directory blocks and formats the output. This option is usefull for debugging directory files which are suspected or corrupted. This function is equivalent to the previously undocumented VMS command DUMP/DIRECTORY, although the layout of the output is different.

- Disk-wide directory functions :

  1 /ALIAS : Scan all directories on the disk for alias file entries.

  2 /CHECK : Scan the disk for problem directories.

  3 /EMPTY : Scan the disk for empty directories

  4 /VERSION=n : Scan all directories on the disk for files which have at least 'n' versions.

The diskwide function qualifiers can be used in one command; other combinations of qualifiers are not possible. Note that the diskwide directory functions can take several minutes to complete because a complete directory scan uses a lot of disk I/O (approximately 2 I/O's per directory).

An indirect command file containing a list of files can be used as a parameter to /COMPRESS or /DUMP. Such a list can be generated with the SEARCH command. Example:

```
DFU> SEARCH disk/FILE=*.DIR/SIZE=MIN=50/OUT=x.x
DFU> DIRECTORY/COMPRESS @x.x
```

Caution: Care should be taken when compressing critical system directories. To be safe you should first use the /TEST qualifier before compressing the directory. Another problem may be caused by a systemcrash just when DFU is rewriting the new compressed directory (although the chance that this happens is very small). When this happens you can recover the files by executing the following procedure:

1 Do a $SET FILE/NODIR of the corrupted directory

2 Delete the directory and create a new one

3 Do a $ANALYZE/DISK/REPAIR ; this will move all the lost files to [SYSLOST].

4 Move the files from [SYSLOST] to the newly created directory.

7.2 Discussion of directory compression

This version of DFU has some major changes in the directory compression process (issued by the DIRECTORY/COMPRESS command). Previously DFU always performed a maximum compression , and then truncated the resultant directory. This may be usefull on a dormant directory , but on an active directory this method introduced new problems:

- When new files were added to the directory it must very quickly be extended. Not only is this a slow operation, but on a fragmented disk this can lead to DIRALLOC errors.
- Active directies where files are inserted in a random alphabetical order very quickly lead to expansion of the just compressed directory.

The first problem is now solved by NOT doing an automatic TRUNCATION of the directory. The /TRUNCATE qualifier has been added to allow truncation on the users request. Also /TRUNCATE=blocksize is possible to perform partial truncation; DFU will not truncate below the blocksize issued in /TRUNCATE. (The actual blocksize will be rounded up to a multiple of the disks cluster size.) The second problem is solved by the new /FILL_FACTOR qualifier. Here a user can direct DFU to fill each directory block up to a certain percentage (between 50 and 100). Not specifying /FILL_FACTOR is the same as doing /FILL_FACTOR=100 (the 'old' DFU behaviour). Note however that the directory's allocated size must be large enough to allow a lower fill factor. If that is not possible

DFU will issue a DFU-E-EXTERR error, and you must use a higher fill factor. So, what strategy should be used when comnpressing directories?

1 Inactive directories : use DIR/COMPRESS/TRUNCATE.

2 Not very active diredctories : use DIR/COMPRESS (/TRUNCATE=n).

3 Active directories, files added at the end (such as MAIL directories) : use DIR/COMPRESS

4 Active directories, files added in random alphabetical order : use DIR/COMPRESS/FILL_FACTOR=n

DIRECTORY

Performs directory functions.

FORMAT

DIRECTORY *device[:] or directory-files(s)*

PARAMETERS

device

The device on which to perform one of the diskwide directory functions.

directory-files(s)

The file specification of the directory on which to perform the /CREATE , /COMPRESS or /DUMP function. Wildcard file specifications may be used. DFU will automatically add .DIR to the filespecification if a filetype is not provided.

QUALIFIERS

/ALIAS

The /ALIAS qualifier directs DFU to scan all directories for alias file entries. Only alias files in DIFFERENT directories will be found (this is the most common situation). Normally only the system disk should contain alias files. This qualifier can be combined with /VERSION=n and /EMPTY.

/ALLOCATE=n

Used in combination with /CREATE. Creates the new directory with an allocated size of 'n' blocks.

/CHECK

Perform a diskwide scan for corrupted directories. Directories with a first byte = 0, and directories which contain files out of alphabetical order are reported as being corrupt. Also directories with invalid file entries are reported. These last files can be removed by using the /REPAIR qualifier.

/COMPRESS

Performs the directory compression unless /TEST is also specified. The output will show the results in terms of file sizes. Example:

```
DFU> DIR/COMP OWN$:[000000]MAIL

%DFU-S-DONE, OWN$:[000000]MAIL.DIR;1: 31 files; was : 4/9, now : 3/3 blocks

DFU>
```

/CREATE

Create a new , empty directory. The allocated size will be 1 cluster unless /ALLOCATE is specified.

/DEVICE

The device on which the files reside. The device name is added to the file name. If you generate a file list with the DFU SEARCH command you don't need this qualifier. However you may have a list of files produced from another source and thus need to append a device name to it. This qualifier can only be used in combination with /COMPRESS or /DUMP.

/DUMP

Produce a block level dump of a directory. /DUMP will interpret the directory entries found in each block. A directory entry contains a size, version-limit, type and name field. Next the directory entry contains a list of versions and file ID's for this entry. /DUMP will produce a rather low level output of this information. Example :

```
DFU> DIR/DUMP C.DIR

DUMP of directory block 1
Size: 22, Version limit: 3, Type: 0, Name(10): CHKDSK.EXE
   Version: 3, FID : (1027,1456,0)
Size: 26, Version limit: 3, Type: 0, Name(13): CHKDSK.README
   Version: 2, FID : (33892,1171,0)
   Version: 3, FID : (1256,11234,0)
Size: 22, Version limit: 3, Type: 0, Name(10): CHKDSK.SAV
   Version: 2, FID : (13947,100,0)
Size: 24, Version limit: 3, Type: 0, Name(11): CHKDSK.SAVE
   Version: 1, FID : (12531,2114,0)
Size: 26, Version limit: 3, Type: 0, Name(14): CHKDSK_LNK.COM
   Version: 2, FID : (12314,72,0)
Size: 18, Version limit: 3, Type: 0, Name(5): C_D.C
   Version: 3, FID : (32650,32,0)
Size: 20, Version limit: 3, Type: 0, Name(7): C_D.EXE
   Version: 9, FID : (2675,64,0)
Size: 20, Version limit: 3, Type: 0, Name(7): C_D.OBJ
   Version: 4, FID : (9835,117,0)
Size: 20, Version limit: 3, Type: 0, Name(7): C_D_2.C
   Version: 15, FID : (12428,84,0)
Size: 22, Version limit: 3, Type: 0, Name(9): C_D_2.OBJ
   Version: 9, FID : (8439,119,0)
Size: 22, Version limit: 3, Type: 0, Name(9): C_D_3.CLD
   Version: 3, FID : (9777,178,0)
Size: 22, Version limit: 3, Type: 0, Name(9): C_D_3.OBJ
   Version: 2, FID : (9737,161,0)
Size: 20, Version limit: 3, Type: 0, Name(7): C_D_4.C
   Version: 3, FID : (10594,1580,0)
Size: 22, Version limit: 3, Type: 0, Name(9): C_D_4.OBJ
   Version: 2, FID : (10177,109,0)

DUMP of directory block 2
Size: 18, Version limit: 3, Type: 0, Name(5): DFU.C
   Version: 169, FID : (31123,52,0)
```

```
.
.
.
%DFU-I-TOTAL, OWN$:[SOURCES.FORTRAN.DFUSRC]C.DIR;1: 42 files
```

/EMPTY

The /EMPTY qualifier will produce a list of empty directories. This qualifier can be combined with /ALIAS and /VERSION=n.

/FILL_FACTOR=n

Specifies ,in percentage, the filling of directory blocks during a /COMPRESS operation. Fill factor may be between 50 and 100. Omitting this qualifier is the same as /FILL_FACTOR=100 (maximum compression). If the directory has not enough allocated space a low fill factor may result in failure of the Compress operation. DFU will issue a DFU-E-EXTERR error, and the directory file will not be modified.

/OUTPUT=filename

This qualifier redirects the output to a file. The output will also go to SYS$OUTPUT.

/STATISTICS

This qualifier displays the performance statistics: CPU time, Elapsed Time, I/O and PageFaults.

/TEST

The /TEST can only be used in conjunction with /COMPRESS. The qualifier will show the results of the /COMPRESS but will not compress the directory file.

/TRUNCATE=blcoksize

During a compress operation truncate the directory back to the blocksize specified. If blocksize is not specified, the file will be truncated back to the end-of-file block number. Truncation will always be rounded up to the next highest multiple of the disks cluster size.

/VERSION=n

The /VERSION qualifier directs DFU to scan all directories on the device to produce a list of files which have at least 'n' versions. This qualifier can be combined with /ALIAS and /EMPTY.

8 REPORT

This chapter describes the REPORT command with the related parameters and qualifiers.

8.1 Description

The REPORT command function generates a report of the file and free space fragmentation of the disk. Also a graph may be generated by the /GRAPH qualifier; this visualises the free space distribution on your disk. The /USAGE qualifier will generate a diskspace usage report. This is especially usefull if DISKQUOTA is not enabled on the disk.

8.2 Syntax and Output

The syntax of the report command is:

```
DFU> REPORT device/QUALIFIERS
```

Below follows an example and output of the report command:

```
DFU> REPORT USER3:/GRAPH/USAGE

%DFU-I-REPORT, Reporting on USER3: ($1$DUA102:)

  ***** Volume info for USER3: (from HOME block) *****
 Volume name                      :  USER3
 Volume owner                     :  SYSTEM
 Structure name                   :
 Clustersize                      :  3
 Maximum # files                  :  367618
 First header VBN                 :  103
 Header count                     :  51547 (1)
 Free headers                     :  10951 (2)

  ***** File statistics (from INDEXF.SYS) *****
 INDEXF.SYS fragments /map_in_use :  21 / 61 (39% used)   (3)
 Total files                      :  40595
 Files with extension headers     :  0
 Files marked for delete          :  19 , size : 25/63 blocks
 Directory files                  :  1957
 Empty files                      :  177
 Files with allocation            :  40418
 Contiguous files                 :  40326 (4)
 Total size used /allocated       :  2492431 /2533209
 Total fragments                  :  40770
 Average fragments per file       :  1.009  (5)
 File fragmentation index         :  0.692 (excellent) (6)
 Average size per fragment        :  62
 Most fragmented file             :
    $1$DUA102:[LARGEUSER]A.DAT;1 ( 180/630 blocks; 30  fragments)
```

```
 ***** Free space statistics (from BITMAP.SYS) *****
Total blocks on disk             :  2940951
Total free blocks                :  407742
Percentage free (rounded)        :  13
Total free extents               :  260
Largest free extent              :  42912 blocks at LBN: 2463597 (7)
Average extent size (rounded)    :  1568
Free space fragmentation index   :  0.191 (excellent) (6)

%free----------------------------------------------------------------
100 |                                                               |
 95 |                                                               |
 90 |                                                               |
 85 |                           *                                   |
 80 |                           *                                   |
 75 |                           *                                   |
 70 |                           *                          *        |
 65 |                           *                          *        |
 60 |                           *                          *        |
 55 |                           *                          *        |
 50 |                      *    *                        * *        |
 45 |                      *    *                        * *    *   |
 40 |                      *    **                       * *    *   |
 35 |                      *    **                       * *    **  |
 30 |                      *   * *** *                 *  * *    **  |
 25 |              *       **   * *** *            *   **  ***    ** * |
 20 |              *       **   * *** *            * * **  ***    ** * |
 15 |      *  *  * *   * * ********** *            * * **  ***    ** * |
 10 |      *  *  * *   *** ********** **           * * ** ****    ** * |
  5 |      *  *  * * ***** ***************        * * ** ****    ** * |
    ----------------------------------------------------------------
LBN: 0                                                        2940951
Free space distribution, each * =  2100  free blocks

 ***** Disk Usage Table (from INDEXF.SYS and QUOTA.SYS) *****
Identifier/UIC                Used/Allocated   Headers   Quota Used/Perm
------------------------------------------------------------------------
[SYSTEM]                     499554/505626     4954     510580/2000000
[SSG,MRX]                     42079/46356      2259      48615/60000
[SSG,TESTJE]                 189306/254742     4935     259677/300000
[SSG,LARGEUSER]               47961/50238      1648      51886/65000
.
.
.
```

The various items have the following meaning :

1 The header count is calculated based on the size of INDEXF.SYS. If more headers are needed INDEXF.SYS has to be extended.

2 The free headers gives the number of free entries in INDEXF.SYS before it has to extend.

3 The INDEXF.SYS number of fragments and mapwords in use are very important figures to determine if the INDEXF.SYS file can be extended. The theoretical maximum number of mapwords is 155. This value may be less if there are ACL's on INDEXF.SYS. Each fragment, and therefore each retrieval pointer must be in the mapword area. As the size of a retrieval pointer depends on the physical size of a disk, it is not easy to say how many fragments INDEXF.SYS can store in the mapwords area. If the mapword area is full the INDEXF file can no longer extend, and creating new files may result in a SYSTEM-W-HEADERFULL error.

4 Files which have exactly 1 retrieval pointer are considered to be contiguous. This doesn't mean that the CONTIGUOUS bit is set in the file header.

5 The average fragments per file give some indication about the total file fragmentation.

6 The file and free space fragmentation index classifies your disk as follows :

- 0-1 : Excellent
- 1-2 : Good
- 2-3 : Fair
- 3-4 : Poor
- >4 : Bad

If your index is greater than 3 you should consider defragmenting the disk using Digitals DFO software, or by performing an Image BACKUP / Restore.

7 The largest free extent is a usefull figure for defragmentation purposes. You cannot defragment a file when its size is greater than this value.

The disks free space distribution as a bar graph. On the horizontal axis the Logical Block Number (LBN) is shown, running from 0 to the maximum block number. Each * represents a certain number of contiguous free blocks.

If you use the /USAGE qualifier a usage table will be added. Per Identifier / UIC the blocks used/allocated , and the number of file headers are shown. If Diskquota is enabled a third column will be included which shows the blocks used/permitted allowing to QUOTA.SYS. Normally quota should satisfy the following rule :

```
Quota used = blocks allocated + # of file headers
```

REPORT

Create a file and free space report of a disk device.

FORMAT

REPORT *device[:]*

PARAMETERS

device[:]

Device to be reported. You may also specify a logical devicename.

QUALIFIERS

/GRAPH

This qualifier generates a graph table which visualizes the free space distribution on the disk. /NOGRAPH is the default.

/NOFILE

The /NOFILE qualifier directs DFU to skip the "File Statistics" reporting phase. When /NOFILE is used the /USAGE qualifier can not be used.

/OUTPUT=filename

This qualifier redirects the output to a file. The output will also go to SYS$OUTPUT.

/STATISTICS

This qualifier displays the performance statistics: CPU time, Elapsed Time, I/O and PageFaults.

/USAGE

The /USAGE qualifier will generate a diskspace usage report based on Identifiers/UIC. If diskquota is enabled on the disk the blocks used/permitted allowing to QUOTA.SYS will also be shown. /USAGE is very usefull when diskquota is not enabled on the disk. /USAGE can not be used together with /NOFILE.

9 SEARCH

This chapter describes the SEARCH command with the related parameters and qualifiers.

9.1 Introduction

The search function allows very quick disk-wide searches for specific files. You can use almost any possible file attribute as a search qualifier.

9.2 Syntax and Output

The command syntax for SEARCH is:

```
DFU> SEARCH device/qualifiers
```

The output is displayed on your terminal. You can also sort the output with the /SORT qualifier. The output is shown in 2 or 3 columns eg:

```
DFU> SEARCH MYDISK/FILE=X.X/FRAGMENT
 .
 .
$1$DUA102:[USER.COMMAND]X.X;1             1/3               1/1
 .
 .
```

The first column shows the full file name . As of this version of DFU the device name is included in the file name. The second columns shows the file size as actual/allocated size. The 3rd column is optional and will only be shown when the /FRAGMENT qualifier is used. It shows the number of fileheaders / number of file fragments.

9.3 Volume Set processing

SEARCH checks if the device to-be-searched is member of a volume set. If so, the complete volume set will be processed, starting with Relative Volume Number 1 up to the last member in set. You can override this behaviour with the /NOVOLSET qualifier.

SEARCH

Fast file search through INDEXF.SYS.

FORMAT **SEARCH** *device[:]*

SEARCH command
SEARCH

PARAMETERS *device[:]*

Device to be searched. You may also specify a logical devicename.

DESCRIPTION

The SEARCH command is used for quick disk-wide searches for specific files. The qualifiers describe the file attributes used as the search criteria. You can combine almost all of the possible qualifiers, unless otherwise specified. Eg.:

```
DFU> SEARCH DISK1/SIZE=MIN=10/OWN=[1,4]/CREATED=SINCE=YESTERDAY
```

is a valid command. Note that qualifiers will be used in a logical AND manner, that is, SEARCH will only display files which obey to all qualifiers and options specified.

QUALIFIERS

/ALLOCATED

Default, SEARCH will use the actual file size. The /ALLOCATED qualifier forces SEARCH to use the allocated filesize. This qualifier is only meaningfull if you combine it with /SIZE.

/BACKUP=option(,option)

The /BACKUP qualifier is used to search files depending on their backup date. The possible options are :

- /BACKUP=BEFORE=date
- /BACKUP=SINCE=date
- /BACKUP=NONE

The NONE option gives files which don't have a backup date recorded.The other 2 options can be used to get files which have a backup date before or after a specific date. The BEFORE and SINCE option can be combined in one command.

/BAKFID=backlink-file-id

You can use this qualifier to look for a file with a specific backlink. Only the first part of the file id must be specified, that is, if the file id is (x,y,z) you must specify 'x' as the file id. You can use this qualifier to get files from a specific directory. If you use /BAKFID=0 you will get files not belonging to a directory (often temporary files).

/CHARACTERISTIC=(char1,char2...)

This qualifier is used to get files with specific file characteristics.The characteristics can have the following values:

- Directory : directory files
- Nobackup : files marked nobackup
- Contiguous : files marked as contiguous
- Erase : erase file before deletion
- Spool : spool files
- Badblock : files which contain suspected bad block(s)

- Badacl : files with a corrupted ACL
- Besttry : files marked contiguous-best-try
- Scratch : files marked as scratch files
- Nocharge : files not charged against quota
- Nomove : files marked NoMove (VMS 5.5+)
- Locked : files with the deaccess lock bit set
- Marked : files marked for deletion
- Noshelvable : files which can not be shelved
- Isshelved : files which are shelved

The several characteristics can be combined in one command.If you want a logical OR behaviour in stead of the default AND behaviour, you can use the special option MATCH=OR eg:

```
SEARCH disk/CHAR=(NOMOVE,DIR,MATCH=OR)
```

/CREATED=option(,option)

The /CREATED qualifier is used to search files depending on their creation date. The options are :

- /CREATED=BEFORE=date
- /CREATED=SINCE=date

These 2 options can be used to get files which have a creation date before or after a specific date. The BEFORE and SINCE option can be combined in one command.

/EXCLUDE=(file1,file2...)

This qualifier is used to exclude files from the search. You can use wildcard filenames ; you can also specifiy a list of files.

/EXPIRED=option(,option)

The /EXPIRED qualifier is used to search files depending on their expiration date. The options are :

- /EXPIRED=BEFORE=date
- /EXPIRED=SINCE=date
- /EXPIRED=NONE

The NONE option gives files which don’t have a expiration date recorded.The other 2 options can be used to get files which have a expiration date before or after a specific date. The BEFORE and SINCE option can be combined in one command.

/FID=file-id

The /FID qualifier is a special function of SEARCH. It directs SEARCH to go directly to the specified file without processing the rest of the Index file. Therefore, you cannot combine this qualifierwith other search qualifiers such as /FILE. Normally a file-id consists is in the form (x,y,z) where x

is the number of the file header within INDEXF.SYS y is the sequence number, and z is the Relative Volume Number. You only need to specifiy the x value in the /FID qualifier.

/FILE=(file1,file2...)

This qualifier is used to search for files with a specific file name.You can use wildcard file names; also you can specify a list of files.

/FRAGMENT(=(MINIMUM=nr,MAXIMUM=nr))

This qualifier displays the number of fragments and file headers of each file found. You can further limit the files found by using the MINIMUM and MAXIMUM= options. For example:

```
DFU> disk/FRAG
DFU> disk/FRAG=min=10
DFU> disk/FRAG=(min=10,max=100)
```

/FULL

If you use this qualifier , SEARCH will give a full output for each file found. This output is a look-alike of the output from DIRECTORY/FULL. You cannot combine this qualifier with /SORT or /SUMMARY.

/HOME

This qualifier directs search to give output from the disk's home block. SEARCH uses the home block for calculating the free and used file headers within INDEXF.SYS.

/(NO)IDENT=identifier

You can search for files owned by a specific identifier. Another way is the /OWNER_UIC qualifier; you cannot combine these 2 qualifiers in one SEARCH command. Issueing /NOIDENT directs DFU to search for files NOT owned by this IDENTIFIER, eg: DFU> SEARCH sys$sysdevice/NOIDENT=SYSTEM.

/LBN=logical-block-number

The /LBN qualifier is a special function of SEARCH. This allows you to find a file which contains a specific LBN. This may be usefull if you have bad blocks logged in the error log. You cannot combine /LBN with search qualifiers such as /FILE.

/(NO)OWNER_UIC=[x,y]

This qualifier is used to get files owned by uic [x,y]. You cannot combine this qualifier with /IDENT. Specifying /NOOWNER directs DFU to search for files NOT owned by this UIC.

/MODIFIED=option(,option)

The /MODIFIED qualifier is used to search files depending on their modification date. The options are :

- /MODIFIED=BEFORE=date
- /MODIFIED=SINCE=date

These 2 options can be used to get files which have a modification date before or after a specific date. The BEFORE and SINCE option can be combined in one command.

/MULTIPLE

This qualifier searches for files which have more than 1 file header.

/NOSEARCH

This qualifier can only be used in combination with /HOME. It is used to get the Homeblock info, without searching the entire disk.

/NOVOLSET

The default behaviour of SEARCH is to process an entire volume set. You can override this behaviour with /NOVOLSET. This may be usefull when doing a /LBN search.

/OUTPUT=file

This qualifier redirects the output from SEARCH to an output file.This file can later be used as input to the DEFRAG , DIRECTORY or SET command.

/PLACED

Checks for files which have a placement control retrieval pointer.This is sometimes set by non-DEC disk defragmentation programs.

/SIZE=(minimum=size1,maximum=size2)

The /SIZE qualifier is used to limit files found by their file size.You can either use minimum= or maximum= or use both options together. SEARCH will use the actual file size for selecting files, except when the /ALLOCATED qualifier is used.

/SORT

This qualifier will sort the files found before being output. You can combine this with the /OUTPUT qualifier (the default output is SYS$OUTPUT).

/STATISTICS

This qualifier displays the performance statistics: CPU time, Elapsed Time, I/O and PageFaults.

/SUMMARY

This qualifier will limit the output to the total number of files found, with their size (and optional the fragmentation when /FRAG is used). You cannot combine /SUMMARY with /FULL.

/VERSION_ NUMBER=(minimum=version1,maximum=version2)

The /VERSION_NUMBER qualifier is used search for files within a range of specific file version numbers. Logfiles which are produced with the same name may pose problems when they reach the maximum version number (32767). Using /VERSION=MIN=32000 is a quick way to find such files.

10 SET

This chapter describes the SET command with the related parameters and qualifiers.

10.1 Introduction

The SET command allows you to modify file attributes which can't be modified through DCL commands. Note that you should be carefull about using this function; you may easily corrupt files. NOTE: This much wanted functionality is finally introduced in VMS 6.0 with a new DCL command $SET FILE/ATTRIBUTES. Still DFU's SET has some options which are not in SET FILE/ATTRIBUTES (such as setting a BACKUP date).

SET

Modify file attributes.

FORMAT

SET *file1,file2,...,@file*

PARAMETERS

file1,file2,...,@file

The files to be modified. The attributes to be modified are specified with qualifiers. Wildcards area allowed in the filename. An indirect file can be used by using the @ sign. This allows processing of a file list produced by a DFU SEARCH command.

QUALIFIERS

/BACKUP_DATE=date
/NOBACKUP_DATE

Controls whether a new backup date is assigned to the specified files. Specify the date according to the rules described in Chapter 1 of the VMS DCL Concepts Manual. Absolute date keywords are allowed. If you specify 0 as the date, today's date is used. If you specify /NOBACKUP_DATE the Backup date field will be cleared.

/BADACL
/NOBADACL

Sets or resets the 'BADACL' flag in the file header. This enables deletion of a file with a corrupted ACL.

/BUCKETSIZE=size

Sets a new value for the bucket size in the file header.

/CONFIRM
/NOCONFIRM (default)

Controls whether a request is issued before each individual SET operation to confirm that the operation should be performed on that file. When the system issues the prompt, you can issue any of the following responses:

```
YES       NO      QUIT
TRUE      FALSE   CTRL/Z
1         0       ALL
          <RET>
```

You can use any combination of upper- and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE). Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and <RET>. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, the prompt will be reissued.

/CONTIGUOUS_BEST_TRY
/NOCONTIGUOUS_BEST_TRY

Sets or resets the 'CONTIGUOUS_BEST_TRY' bit in the file header.

/CREATION_DATE=date
/NOCREATION_DATE

Controls whether a new creation date is assigned to the specified files. Specify the date according to the rules described in Chapter 1 of the VMS DCL Concepts Manual. Absolute date keywords are allowed. If you specify 0 as the date, today's date is used.

/DIRECTORY
/NODIRECTORY

Sets or resets the directory attribute of a file. This qualifier allows you to set the directory bit of a file which was mistakingly reset by the 'SET FILE/NODIRECTORY' command. If it is done on a non-directory file, then access to that directory will give a 'BADIRECTORY' error.

/EBLOCK[=block]

This qualifier will reset the end-of-file mark to the highest block allocated if no block has been specified. Otherwise the end-of-file mark will be set to the specified block.

/EBYTE[=byte]

This qualifier will set the end-of-file byte mark to the highest byte if it has not been specified. Otherwise the end-of-file byte mark will be set to the specified byte.

/EXPIRATION_DATE=date
/NOEXPIRATION_DATE

Controls whether an expiration date is assigned to the specified files. Specify the date according to the rules described in Chapter 1 of the VMS

DCL Concepts Manual. Absolute date keywords are allowed. If you specify 0 as the date, today's date is used.

/IDENT=identifier

Modify the file ownership. See also /OWNER_UIC. This allows modification of the file-ownership even if the file is open, eg. INDEXF.SYS. You cannot use both /IDENT and /OWNER_UIC.

/LOCKED
/NOLOCKED

This qualifier will lock a file for future use. Nothing else can then be done with the file, until it is unlocked (which can also be done with the VMS 'UNLOCK' command.

/LOG (default)
/NOLOG

Controls whether the SET command displays the file specification of each file after the modification is made.

/MAXREC=record

Sets a new value for the maximum record number in the file header.

/ORGANIZATION=keyword

The following keywords are used as parameters for the ORGANIZATION qualifier: DIRECT, INDEXED, RELATIVE and SEQUENTIAL. This will allow you to modify the file organization type in the file header. Of course this won't change the real organization of the file.

/OWNER_UIC=[x,y]

Modify the file ownership to UIC [x,y]. See also /IDENT. This allows modification of the file-ownership even if the file is open, eg. INDEXF.SYS. You cannot use both /IDENT and /OWNER_UIC.

/RECATTRIBUTES=keyword

The following keywords are used as parameters for the RECATTRIBUTES qualifier: NONE, FORTRAN, IMPLIED, PRINT and NOSPAN. This will allow you to modify the file's record attributes in the file header. NONE, FORTRAN, IMPLIED and PRINT are mutually exclusive, but can be used in combination with NOSPAN. When NOSPAN is omitted SPAN is assumed (the default is to allow records to cross block boundaries).

/RECSIZE=size

Sets a new value for the record size in the file header.

/RECTYPE=keyword

The following keywords are used as parameters for the RECTYPE qualifier: FIXED, STREAM, STREAMCR, STREAMLF, UNDEFINED, VARIABLE and VFC. This will allow you to modify the file's record type in the file header.

/REVISION_DATE=date
/NOREVISION_DATE

Controls whether a revision date is assigned to the specified files. Specify the date according to the rules described in Chapter 1 of the VMS DCL

Concepts Manual. Absolute date keywords are allowed. If you specify 0 as the date, today's date is used.

/RVCOUNT=count

Sets a new value for the revision count in the file header.

/UPDATE
/NOUPDATE (default)

Normally the file's revision date will be updated after any modification to it. SET however disables this update (otherwise the REVISION date could not be set). Specify this qualifier if you want the revision date to be updated.

/VFCSIZE=size

Sets a new value for the VFC size in the file header. This value will only be used with the VFC record type.

11 UNDELETE

This chapter describes the UNDELETE command with the related parameters and qualifiers.

11.1 Introduction

The UNDELETE function is designed to recover deleted files, if possible. UNDELETE operates in a safe mode such that it first checks if the deleted file header and diskblocks are still available. Only in that case will the file be recovered. Otherwise UNDELETE will leave the disk unmodified. UNDELETE has some powerfull options:

- Generate a list of recoverable files with the /LIST qualifier. This qualifier will not undelete any file, and therefore it does not lock up the disk.
- Undelete multiple files in one pass.
- Make file selections based on the owner uic with the /OWNER or /IDENT qualifier.
- Undeleted files which cannot be entered back in their original directory will be moved to the [SYSLOST] directory. This saves an additional ANALYZE/DISK/REPAIR action.

The UNDELETE command operates as follows :

1 Lock the volume. In the same manner as ANALYZE/DISK/REPAIR and SET VOLUME /REBUILD Undelete locks the disk for other writers such that no modifications to the INDEXF, BITMAP or QUOTA file can be made.

2 Search the INDEXF.SYS file if the requested file is still there. A check is made if it really is a valid deleted file header.

3 Checks if the deleted file header and all possible extension headers are still valid.

4 Checks if the blocks previously owned by this file are still free.

5 If all these conditions are met DFU will display the file and asks for a confirmation.

6 Rebuilds all of the deleted file headers and bitmaps in memory. If no errors are found all the file headers and bitmaps are written back to disk. The files original backlink is saved in a table in memory.

7 If appropiate the blocks and headers are charged against diskquota.

8 When all of INDEXF.SYS has been scanned the volume is unlocked.

9 The file(s) is (are) entered in the original directory if possible, using the table build up in memory. If the enter command fails (probably because the original directory is gone) the file will be entered in [SYSLOST]. DFU will create this directory if needed.

If the /LIST qualifier is used, DFU will just list the recoverable files without performing any action on the disk. The disk will not be locked.

It is important to note that during the bitmap and file header processing any error will immediately terminate the recovery process, and unlock the disk. Files which have been recovered up to that point will still be entered in the appropiate directory. As no wrong information has been written back to the disk this should leave the disk in a proper state.

NOTE: The blocks recovered will NOT be subtracted from the Volume's free block count. To get the actual freeblock count a SET VOLUME/REBUILD=FORCE is necessary. Also a complete reMOUNT will reset the freeblock count.

IMPORTANT: DFU takes care to recover only files whose original blocks are free. However these blocks may have been modified in the meantime by another file which also has been deleted. Therefore each recovered file must be checked manually to check its integrity.

Example of a Undelete session :

```
DFU> undel $1$dua102:/list

Recoverable file [TEST]CHANGE_UIC.FOR;2 found
Recoverable file [TEST]CHECK_ID.FOR;1 found
Recoverable file [TEST]CHKPRDIMG.FOR;1 found
Recoverable file [TEST]CHRLEN.FOR;1 found

DFU> undel/file=*.for $1$DUA102:

%DFU-I-READBMAP, Reading BITMAP.SYS...
%DFU-W-LOCKED, Volume now LOCKED for write
%DFU-I-UNDEL, Start search on $1$DUA102:

Recoverable file [TEST]CHANGE_UIC.FOR;2 found
Recover this file? (Y/N) [N] : n
Recoverable file [TEST]CHECK_ID.FOR;1 found
Recover this file? (Y/N) [N] : y
%DFU-S-RECOVER, File succesfully recovered
%DFU-I-ADDQUOTA, updating diskquota...

Recoverable file [TEST]CHKPRDIMG.FOR;1 found
Recover this file? (Y/N) [N] : n
Recoverable file [TEST]CHRLEN.FOR;1 found
Recover this file? (Y/N) [N] : y

%DFU-S-RECOVER, File succesfully recovered
%DFU-I-ADDQUOTA, updating diskquota...
%DFU-I-UNLOCK, Volume unlocked
%DFU-I-ENTER, Entering file(s) in directory...
%DFU-S-ENTERED, File CHECK_ID.FOR entered in original directory
%DFU-S-ENTERED, File CHRLEN.FOR entered in original directory

DFU> EXIT
```

UNDELETE

Recover one or more deleted files on a device.

FORMAT

UNDELETE *device[:]*

PARAMETERS

device

The device on which to undelete one or more files. The device will be write-locked during the undelete process.

QUALIFIERS

/FILE=filename

The file to be recovered. Wildcards may be used. If a matching file is found Undelete will ask a confirmation. If a certain file is confirmed, that file will be recovered if possible, and control will be returned to the DFU> prompt. If the /FILE qualifier is omitted DFU will assume *.*;* .

/IDENT=identifier

You can search for files owned by a specific identifier. Another way is the /OWNER_UIC qualifier; you cannot combine these 2 qualifier in one UNDELETE command.

/LIST(=output-file-name)

Generate a list of recoverable files. No undelete will take place, and the disk will not be locked. The defaultoutput is SYS$OUTPUT.

/OWNER_UIC=[x,y]

This qualifier is used to select files by uic. The uic must be of the form [x,y]. You cannot combine this qualifier with /IDENT.

/STATISTICS

This qualifier displays the performance statistics: CPU time, Elapsed Time, I/O and PageFaults.

12 SPAWN

The new SPAWN command creates an interactive subprocess. This allows a quick escape to the DCL level from within an interactive DFU session. The subprocess has a default prompr DFU_sub$. In SMG mode the <DO> key also performs a SPAWN command. You must logout explicitly to return to the DFU session.

13 VERIFY

This chapter describes the VERIFY command with the related parameters and qualifiers.

13.1 Introduction

The VERIFY command makes an analysis of the disk, scanning for file and disk structure errors. VERIFY performs almost all of the checks normally done by a ANALYZE/DISK command. But VERIFY is several times faster than ANALYZE/DISK, and using less resources.

VERIFY checks and reports the following errors:

- Checks the logical information contained in the disks HOME block
- Reports files marked for deletion
- Reports mismatches in the index file's bitmap
- Checks all backlinks.
- Reports multiple allocated blocks
- Reports blocks incorrectly marked free and allocated
- Reports all lost files
- Checks for mismatches between blocks used and blocks charged in QUOTA.SYS. (Only done when diskquota is enabled)

The /FIX qualifier can be used to perform some basic repair actions. Unlike ANALYZE/DISK/REPAIR is does not lock the disk! /FIX repairs the following errors:

- Deletes file marked for delete (if possible).
- Recovers lost files and directories into [SYSLOST].
- Recovers a very rare problem which will get ANALYZE/DISK/REPAIR into a computable loop (This problem is fixed in VMS V6.2 or with the VERI02_061 patch kit).

As of V1.8 the /REBUILD qualifier can be used to performs the same function as a SET VOLUME/REBUILD=FORCE command. It will lock the disk for a short period of time (usually less than 30 seconds). /REBUILD repairs the following errors :

- Mismatches in the INDEXF.SYS bitmap (files reported with the NOBITSET and NOBITCLR errors)
- Blocks incorrectly marked allocated or free in the BITMAP.SYS.
- Mismatches in the diskquota file.

Note that /REBUILD does NOT change the disks free block count.

CAUTION: On a system (cluster) wide mounted disk there will probably be concurrent disk activity during the VERIFY command. Therefore VERIFY may report some errors which are not really errors. Try running VERIFY 2 or 3 times to see if the errors are reported again. To get a really consistent report from VERIFY you can use the /LOCK qualifier. This will write-lock the disk during the VERIFY run, so don't use this qualifier too often. (The /REPAIR qualifier of ANALYZE/DISK also write-locks the disk). Even if DFU ends ungracefully the DFU exit handler will always UNLOCK the disk.

13.2 Error reporting

An example of a typical VERIFY run follows:

```
DFU> VERIFY mydisk
%DFU-I-VERIFY, Verifying MYDISK:
%DFU-S-CHKHOME, Home block info verified OK
%DFU-I-IFSCAN, Scanning INDEXF.SYS ...
%DFU-I-CHKBITMAP, Checking BITMAP.SYS...
%DFU-I-CHKLOST, Checking for lost files...
%DFU-I-CHKQUOTA, Checking QUOTA.SYS...

DFU> VERIFY/REBUILD sys$sysdevice
%DFU-W-LOCKED, Volume now write locked
%DFU=I-VERIFY, Verifying SYS$SYSDEVICE:
%DFU-S-CHKHOME, Home block info verified OK
%DFU-W-DELETED, file (620,351,1) RSF_DI_RSF_SERVER0.TMP;3 marked for delete
%DFU-W-DELETED, file (6349,173,1) DCLTABLES.EXE;937 marked for delete
%DFU-W-DELETED, file (19745,35,1) RSF_DI_RSF_SERVER0.TMP;3 marked for delete
%DFU-I-CHKBITMAP, Checking BITMAP.SYS...
%DFU-E-ALLOCCLR, blocks LBN 2667141 through 2667143 incorrectly marked allocated
%DFU-E-ALLOCCLR, blocks LBN 2667153 through 2667161 incorrectly marked allocated
  .
  .
%DFU-S-RBDBITMAP, BITMAP.SYS succesfully rebuild
%DFU-I-CHKLOST, Checking for lost files...
%DFU-I-UNLOCK, Volume unlocked

DFU>
```

VERIFY can report the following errors:

ALLOCCLR, blocks incorrectly marked allocated,

Severity: ERROR

Explanation: Blocks were found which are not allocated by a file but set in the BITMAP file. This error is often seen when VERIFY is run and there is concurrent file activity on the disk. Also, if after a system crash the disk is mounted with /NOREBUILD such blocks may exist. ReRun VERIFY with /LOCK to see if the error is still there.

User Action: Run VERIFY/REBUILD to rebuild the BITMAP file.

ALLOCSET, blocks incorrectly marked free,

Severity: ERROR

Explanation: Blocks were found which are allocated by a file but not set in the BITMAP file. This error is often seen when VERIFY is run and there is concurrent file activity on the disk. Also, if after a system crash the disk is mounted with /NOREBUILD such blocks may exist. ReRun VERIFY with /LOCK to see if the error is still there.

User Action: Run VERIFY/REBUILD to rebuild the BITMAP file.

BADBLOCK, file has suspected bad blocks,

Severity: ERROR

Explanation: A file has suspected bad blocks. This is likely to be caused by hardware errors on the disk.

User Action: Try to copy the file to another location. Also check the Error Log.

BADEXTLNK, link to extension header broken,

Severity: ERROR

Explanation: VERIFY tries to follow the extension link of the file and it failed, or the extension header is invalid.

User Action: Try ANALYZE/DISK/REPAIR. However, this may be an unrepairable error, in which case the file will be corrupted and unreliable.

DELETED, file marked for delete,

Severity: WARNING

Explanation: A file was found marked for deletion. Such files usually exists as a result of file being deleted while it was still INSTALLed , or as a result of a system crash. It is not considered as an error. You can try to delete such files by using VERIFY/FIX.

ERRHOME, Home block info not OK,

Severity: ERROR

Explanation: The Info contained in the Home Block was not consistent or corrupted.

User Action: Run ANALYZE/DISK/REPAIR to try to fix the error.

INVBAKFID, file has invalid backlink,

Severity: ERROR

Explanation: The files backlink file id is not filled in. This is likely to be a lost file. (Note that once VERIFY reports this error, it will not be reported again during the LOSTFILE check). But is also possible that the file is in a valid directory, but that the Backlink file id is not filled in.

User Action: Run ANALYZE/DISK/REPAIR. If the file is in a valid directory the backlink will be repaired. Otherwise the file will be moved to

the [SYSLOST] directory. You can also use VERIFY/FIX but then the file will unconditionally be moved to [SYSLOST].

LOCKED, file is deaccess locked,

Severity: WARNING

Explanation: The file is deaccess locked. This may be a result of a system crash.

User Action: Try to unlock the file with the DFU SET command, or the DCL UNLOCK command.

LOSTHDR1, file found in nonexistent directory,

Severity: WARNING

Explanation: During the lost file check a file was discovered in a nonexistent directory. This error can be caused as follows : set a directory file to NODIRECTORY and delete it.

User Action: Run VERIFY/FIX to move the file to the [SYSLOST] directory.

LOSTHDR2, file found in directory with bad backlink,

Severity: WARNING

Explanation: During the lost file check a file was discovered in a valid directory. However, the directory has an invalid backlink. This error can be caused by doing a SET FILE/REMOVE of a directory file.

User Action: Run VERIFY/FIX to move the directory to the [SYSLOST] directory. Then move the directory back to the correct location. The files in the directory should be accesible again.

LOSTHDR3, file found in invalid directory,

Severity: WARNING

Explanation: During the lost file check a file was discovered in a directory which is not a valid directory. Either the file has a wrong backlink, or the directory file has the directory file attribute not set.

User Action: Look up the directory and use DFU SET to set the directory bit. If the directory file was OK then run VERIFY/FIX to recover the lost file.

MULTALLOC, blocks multiple allocated,

Severity: ERROR

Explanation: A file has blocks allocated which already belong to another file. Note that VERIFY will make a second pass through INDEXF.SYS to report all the files involved. This is a serious error, and may be a result of a disk being mounted on two seperated VAXClusters (or a partitioned VAXCluster).

User Action: Copy all the files found to another location. Next all the files must be deleted. Run VERIFY/REBUILD to repair the BITMAP. All

the afflicted files must be manually inspected to see which files are valid and which are corrupted.

NOBITCLR, Deleted file header marked BUSY,

Severity: WARNING

Explanation: A file was deleted. The corresponding bit in the INDEXF bitmap should be cleared but is still set. This is not a serious error.

User Action: Run VERIFY/REBUILD to fix the error.

NOBITSET, index file bitmap bit not set,

Severity: WARNING

Explanation: A valid file was found , but the corresponding index file bitmap bit is clear. This is not a serious error.

User Action: Run VERIFY/REBUILD to fix the error.

NOOWNER, file has no owner,

Severity: WARNING

Explanation: A file was found with UIC [0,0] as the owner. This is not a normal situation.

User Action: Look up the file and modify the file owner.

QUOTAERR, UIC [x,y] has n blocks used, QUOTA indicates m blocks,

Severity: WARNING

Explanation: If a disk has diskquota enabled, VERIFY calculates the blocks used and compares them with the value in the QUOTA.SYS file. Any mismatch is reported. This error is often seen when VERIFY is run, and there is concurrent file activity on the disk. Also, if after a system crash the disk is mounted with /NOREBUILD such Quota mismatches may exist. ReRun VERIFY with /LOCK to see if the error is still there.

User Action: Run VERIFY/REBUILD to rebuild the QUOTA file. Or run a DISKQUOTA> REBUILD command.

SLFBAKFID, backlink points to itself,

Severity: WARNING

Explanation: This error indicates that DFU has found a directory with a backlink pointing to itself. This feature is only allowed for the MFD 000000.DIR. This error will result in ANALYZE/DISK going into a COMputable loop.

User Action: Run VERIFY/FIX to move the directory to the [SYSLOST] directory. Then remove the directory entry from itself with a SET FILE/REMOVE command. The files in the directory should be accesible again.

VERIFY

VERIFY a device for disk structure errors.

FORMAT

VERIFY *device[:]*

PARAMETERS

device

The device to be verified.

QUALIFIERS

/FIX
/NOFIX (default)

Repair errors on the disk. File marked for delete will be deleted and lost files and directories will be moved to [SYSLOST].

/LOCK
/NOLOCK (default)

Locks the disk for file allocation/deletion. This gives a consistent report for the disk, but other users may experience a delay when accessing the disk. The device will be unlocked a the end of the VERIFY command.

/OUTPUT=filename

This qualifier redirects the output to a file. The output will also go to SYS$OUTPUT. CAUTION: if you use /LOCK in combination with /OUTPUT be carefull to specify an outputfile on ANOTHER disk, or DFU will finish with a severe error.

/REBUILD
/NOREBUILD (default)

Perform a volume rebuild. Using this qualifier will temporarily lock the disk for other users.

/STATISTICS

This qualifier displays the performance statistics: CPU time, Elapsed Time, I/O and PageFaults.