NAME ufsdump – incremental file system dump

SYNOPSIS **/usr/sbin/ufsdump** [*options*] [*arguments*] *files_to_dump*

AVAILABILITY SUNWcsu

DESCRIPTION

ufsdump backs up all files specified by *files_to_dump* (normally either a whole file system or files within a file system changed after a certain date) to magnetic tape, diskette, or disk file. When running **ufsdump**, the file system must be inactive; otherwise, the output of **ufsdump** may be inconsistent and restoring files correctly may be impossible. A file system is inactive when it is unmouned or the system is in single user mode. A file system is not considered inactive if one tree of the file system is quiescent while another tree has files or directories being modified.

options is a single string of one-letter **ufsdump** options.

arguments may be multiple strings whose association with the options is determined by order. That is, the first argument goes with the first option that takes an argument; the second argument goes with the second option that takes an argument, and so on.

files_to_dump is required and must be the last argument on the command line. See **OPERANDS** for more information.

With most devices **ufsdump** can automatically detect the end-of-media. Consequently, the **d**, **s**, and **t** options are not necessary for multi-volume dumps, unless **ufsdump** does not understand the way the device detects the end-of-media, or the files are to be restored on a system with an older version of the **restore** command.

OPTIONS

The following options are supported:

0–**9** The “dump level.” All files specified by *files_to_dump* that have been modified since the last **ufsdump** at a lower dump level are copied to the *dump_file* destination (normally a magnetic tape device). For instance, if a “level **2**” dump was done on Monday, followed by a “level **4**” dump on Tuesday, a subsequent “level **3**” dump on Wednesday would contain all files modified or added since the “level **2**” (Monday) backup. A “level **0**” dump copies the entire file system to the *dump_file*.

a *archive_file* Archive file. Archive a dump table-of-contents in the specified *archive_file* to be used by **ufsrestore**(1M) to determine whether a file is in the dump file that is being restored.

b *factor* Blocking factor. Specify the blocking factor for tape writes. The default is 20 blocks per write for tapes of density less than 6250BPI (bytes-per-inch). The default blocking factor for tapes of density 6250BPI and greater is 64. The default blocking factor for cartridge tapes (**c** option) is 126. The highest blocking factor available with most tape drives is 126. Note: the blocking factor is specified in terms of 512-byte blocks, for compatibility with **tar**(1).

c Cartridge. Set the defaults for cartridge instead of the standard half-inch reel. This sets the density to 1000BPI and the blocking factor to 126. Since **ufsdump** can automatically detect the end-of-media, only the blocking parameter normally has an effect. When cartridge tapes are used, and this option is *not* specified, **ufsdump** will slightly miscompute the size of the tape. If the **b**, **d**, **s** or **t** options are specified with this option, their values will override the defaults set by this option.

d *bpi* Tape density. Not normally required, as **ufsdump** can detect end-of-media. This parameter can be used to keep a running tab on the amount of tape used per reel. The default density is 6250BPI except when the **c** option is used for cartridge tape, in which case it is assumed to be 1000BPI per track. Typical values for tape devices are:

1/2" tape	6250 BPI
1/4" cartridge	1000 BPI The tape densities and other options are documented in the **st**(7D) man page.

D Diskette. Dump to diskette.

f *dump_file* Dump file. Use *dump_file* as the file to dump to, instead of **/dev/rmt/0**. If *dump_file* is specified as −, dump to standard output.

If the name of the file is of the form *machine***:***device*, the dump is done from the specified machine over the network using **rmt**(1M). Since **ufsdump** is normally run by root, the name of the local machine must appear in the **/.rhosts** file of the remote machine. If the file is specified as *user@machine***:***device*, **ufsdump** will attempt to execute as the specified user on the remote machine. The specified user must have a **.rhosts** file on the remote machine that allows the user invoking the command from the local machine to access the remote machine.

l Autoload. When the end-of-tape is reached before the dump is complete, take the drive offline and wait up to two minutes for the tape drive to be ready again. This gives autoloading (stackloader) tape drives a chance to load a new tape. If the drive is ready within two minutes, continue. If it is not, prompt for another tape and wait.

n Notify all operators in the **sys** group that **ufsdump** requires attention by sending messages to their terminals, in a manner similar to that used by the **wall**(1M) command. Otherwise, such messages are sent only to the terminals (such as the console) on which the user running **ufsdump** is logged in.

o Offline. Take the drive offline when the dump is complete or the end-of-media is reached and rewind the tape, or eject the diskette. In the case of some autoloading 8mm drives, the tape is removed from the drive automatically. This prevents another process which rushes in to use the drive, from inadvertently overwriting the media.

s *size* Specify the *size* of the volume being dumped to. Not normally required, as **ufsdump** can detect end-of-media. When the specified size is reached, **ufsdump** waits for you to change the volume. **ufsdump** interprets the specified size as the length in feet for tapes and cartridges, and as the number of 1024-byte blocks for diskettes. The values should be a little smaller than the actual physical size of the media (for example, 425 for a 450-foot cartridge). Typical values for tape devices depend on the **c** option, for cartridge devices, and the **D** option for diskettes:

1/2" tape	2300 feet
60-Mbyte 1/4" cartridge	425 feet
150-Mbyte 1/4" cartridge	700 feet
diskette	1422 blocks (Corresponds to a 1.44-Mbyte diskette, with one cylinder reserved for bad block information.)

S Size estimate. Determine the amount of space that is needed to perform the dump without actually doing it, and display the estimated number of bytes it will take. This is useful with incremental dumps to determine how many volumes of media will be needed.

t *tracks* Specify the number of tracks for a cartridge tape. Not normally required, as **ufsdump** can detect end-of-media. The default is 9 tracks. The **t** option is not compatible with the **D** option. Values for Sun-supported tape devices are:

60-Mbyte 1/4" cartridge	9 tracks
150-Mbyte 1/4" cartridge	18 tracks

u Update the dump record. Add an entry to the file **/etc/dumpdates,** for each file system successfully dumped that includes the file system name, date, and dump level.

v Verify. After each tape or diskette is written, verify the contents of the media against the source file system. If any discrepancies occur, prompt for new media, then repeat the dump/verification process. The file system *must* be unmounted. This option cannot be used to verify a dump to standard output.

w Warning. List the file systems that have not been backed up within a day. This information is gleaned from the files **/etc/dumpdates** and **/etc/vfstab**. When the **w** option is used, all other options are ignored. After reporting, **ufsdump** exits immediately.

W Warning with highlight. Similar to the **w** option, except that the **W** option includes all file systems that appear in **/etc/dumpdates**, along with information about their most recent dump dates and levels. File systems that have not been backed up within a day are highlighted.

OPERANDS

The following operand is supported:

files_to_dump Specifies the files to dump. Usually it identifies a whole file system by its raw device name (for example, **/dev/rdsk/c0t3d0s6**). Incremental dumps (levels **1** to **9**) of files changed after a certain date only apply to a whole file system. Alternatively, *files_to_dump* can identify individual files or directories. All files or directories are dumped, which is equivalent to a level **0** dump; however, **/etc/dumpdates** is not updated, even with the **u** option specified. In all cases, the files must be contained in the same file system, and the file system must be local to the system where **ufsdump** is being run.

files_to_dump is required and must be the last argument on the command line.

If no *options* are given, the default is **9uf /dev/rmt/0** *files_to_dump*.

EXAMPLES

To make a full dump of a root file system on **c0t3d0**, on a 150-MByte cartridge tape unit **0**, use:

example# ufsdump 0cfu /dev/rmt/0 /dev/rdsk/c0t3d0s0

To make and verify an incremental dump at level **5** of the **usr** partition of **c0t3d0**, on a 1/2" reel tape unit **1,** use:

example# ufsdump 5fuv /dev/rmt/1 /dev/rdsk/c0t3d0s6

EXIT STATUS

While running, **ufsdump** emits many verbose messages. **ufsdump** returns the following exit values:

0 Normal exit.
1 Startup errors encountered.
3 Abort – no checkpoint attempted.

FILES

/dev/rmt/0 default unit to dump to
/etc/dumpdates dump date record
/etc/group to find group **sys**
/etc/hosts to gain access to remote system with drive
/etc/vfstab list of file systems

SEE ALSO

cpio(1), **tar**(1), **dd**(1M), **devnm**(1M), **prtvtoc**(1M), **rmt**(1M), **shutdown**(1M), **ufsrestore**(1M), **volcopy**(1M), **wall**(1M), **wall**(1M) **st**(7D)

NOTES

Read Errors

Fewer than 32 read errors on the file system are ignored.

Process Per Reel

Because each reel requires a new process, parent processes for reels that are already written hang around until the entire tape is written.

Operator Intervention

ufsdump requires operator intervention on these conditions: end of volume, end of dump, volume write error, volume open error or disk read error (if there are more than a threshold of 32). In addition to alerting all operators implied by the **n** option, **ufsdump** interacts with the operator on **ufsdump**'s control terminal at times when **ufsdump** can no longer proceed, or if something is grossly wrong. All questions **ufsdump** poses *must* be answered by typing **yes** or **no**, as appropriate.

Since backing up a disk can involve a lot of time and effort, **ufsdump** checkpoints at the start of each volume. If writing that volume fails for some reason, **ufsdump** will, with operator permission, restart itself from the checkpoint after a defective volume has been replaced.

Suggested Dump Schedule

It is vital to perform full, "level **0**", dumps at regular intervals. When performing a full dump, bring the machine down to single-user mode using **shutdown**(1M). While preparing for a full dump, it is a good idea to clean the tape drive and heads. Incremental dumps should be performed with the system running in single-user mode.

Incremental dumps allow for convenient backup and recovery of active files on a more frequent basis, with a minimum of media and time. However, there are some tradeoffs. First, the interval between backups should be kept to a minimum (once a day at least). To guard against data loss as a result of a media failure (a rare, but possible occurrence), capture active files on (at least) two sets of dump volumes. Another consideration is the desire to keep unnecessary duplication of files to a minimum to save both operator time and media storage. A third consideration is the ease with which a particular backed-up version of a file can be located and restored. The following four-week schedule offers a reasonable tradeoff between these goals.

	Sun	*Mon*	*Tue*	*Wed*	*Thu*	*Fri*
Week 1:	**Full**	**5**	**5**	**5**	**5**	**3**
Week 2:		**5**	**5**	**5**	**5**	**3**
Week 3:		**5**	**5**	**5**	**5**	**3**
Week 4:		**5**	**5**	**5**	**5**	**3**

Although the Tuesday through Friday incrementals contain "extra copies" of files from Monday, this scheme assures that any file modified during the week can be recovered from the previous day's incremental dump.

Process Priority of ufsdump

ufsdump uses multiple processes to allow it to read from the disk and write to the media concurrently. Due to the way it synchronizes between these processes, any attempt to run dump with a **nice** (process priority) of '–5' or better will likely make **ufsdump** run *slower* instead of faster.

Overlapping Partitions

Most disks contain one or more overlapping slices because slice 2 covers the entire disk. The other slices are of various sizes and usually do not overlap. For example, a common configuration places **root** on slice 0, **swap** on slice 1, **/opt** on slice 5 and **/usr** on slice 6.

It should be emphasized that **ufsdump** dumps one **ufs** file system at a time. Given the above scenario where slice 0 and slice 2 have the same starting offset, executing **ufsdump** on slice 2 with the intent of dumping the entire disk would instead dump only the **root** file system on slice 0. To dump the entire disk, the user must dump the file systems on each slice separately.

BUGS

The **/etc/vfstab** file does not allow the desired frequency of backup for file systems to be specified (as **/etc/fstab** did). Consequently, the **w** and **W** options assume file systems should be backed up daily, which limits the usefulness of these options.