M.I.T. Laboratory for Computer Science

Request for Comments No. 239
January 25, 1983

File System Backup on the VAX 11-750s using Remote Disks

by Michael Greenwald

1. Introduction

Our group now has several VAX 11-750s running UNIX. These machines have a single RK07 disk drive, and no tape drive. If the disk crashes, or the file-system gets corrupted, a substantial amount of work will be lost.

Backing up a file system to tape involves shutting the machine down, removing its disk pack, and transporting the disk to AJAX to be copied to tape. Since tape backup is a time-consuming and painful task, it isn't done as often as it should be.

A more convenient method of backup is to dump file systems to remote disks. We can copy them to tape at our leisure. This note describes the procedure we currently plan to use for file system backup.

2. Advantages of backup to remote disks

~ We don't have to shut a machine down to back it up.

~ We don't have to carry a disk pack from our machine to AJAX.

~ We can do many copies to tape at once, so we don't have to make many trips.

~ Responsibility for backup is distributed to people on the individual machines.

~ Although backing up to a remote disk is slower than directly backing up to tape, (Since we have to copy the files twice; once to the disk, and then from the disk to the tape) the time is spaced more conveniently. Many incremental dumps can be copied to tape in one operation.

~ UNIX assumes that a dump, even an incremental one, will take up at least one tape. There is no provision made to combine more than one dump on a single tape. Our file systems are small

- even complete dumps can fit on a single tape. Backing up to a remote disk allows us to combine many dumps on a single tape, since we *tar* the files from the remote disk to the tape.

3. Backing up a file system

To backup a file system you need to spin up the *backup* remote disk in exclusive mode.

Once the disk is spun up you can run the *dump* command. (The appropriate page from the UNIX programmers manual is included). Since the dump is going to a file, instead of to tape, we need to use the *f* key. This should be followed by the name of the file we are copying to.

The backup disk is usually mounted in the /mnt directory. The convention for naming files on the backup disk is:

```
<mounted directory>/<machine id>/<filesystem>.<level>
```

For example, if we were doing a level 0 dump of /usr on borax, the file would be /mnt/borax/usr.0. The entire dump command would be:

```
dump 0uf /mnt/borax/usr.0 /usr
```

The *filesystem* component of the file name must be either *usr* or *root*.

The frequency of filesystem dumps will be decided on a per-machine basis.

3.1. Backing up remote disks

To have a remote disk backed up, copy its name into the directory /remote on the backup disk. When the regular filesystem dumps get copied to tape, the remote disks specified in /mnt/remote will be backed up.

If you want a remote disk backed up *immediately*, then you must do this yourself.

4. Copying a backed up file system to tape

Periodically, someone will spin *backup* up on AJAX, and copy all the dumps on it to their appropriate tapes.

For the time being we will only use two tapes per machine. Whenever we do a level 0 dump of /usr[1] we will switch tapes. This way the last 2 complete dumps will always be available.

Every other time we do a level 0 dump of /usr, we will also do a level 0 dump of /. There is probably no need for incremental dumps of / on most machines, but that is up to the people on each machine.

When a file is actually dumped to tape it will be deleted from the backup disk. If you want to check whether your last dump is safely on tape, you can check to see whether it has been deleted from the remote disk.

5. Restoring files from tape

UNIX has no good way to restore files. If you want any files restored, we copy all the files from the last tape onto the remote disk. We can then spinup the backup disk and run restor(8). Good luck!

---

[1] Probably about once a month

NAME

dump — incremental file system dump

SYNOPSIS

/etc/dump [key [*argument* ...] filesystem]

DESCRIPTION

Dump copies to magnetic tape all files changed after a certain date in the *filesystem*. The *key* specifies the date and other options about the dump. *Key* consists of characters from the set **0123456789fuJsdWn.**

0–9 This number is the 'dump level'. All files modified since the last date stored in the file */etc/dumpdates* for the same filesystem at lesser levels will be dumped. If no date is determined by the level, the beginning of time is assumed; thus the option **0** causes the entire filesystem to be dumped.

f Place the dump on the next *argument* file instead of the tape.

u If the dump completes successfully, write the date of the beginning of the dump on file */etc/dumpdates*. This file records a separate date for each filesystem and each dump level. The format of */etc/dumpdates* is readable by people, consisting of one free format record per line: filesystem name, increment level and *ctime(3)* format dump date. */etc/dumpdates* may be edited to change any of the fields, if necessary. Note that */etc/dumpdates* is in a format different from that which previous versions of *dump* maintained in */etc/ddate*, although the information content is identical.

J This option is intended to be invoked only when the old format */etc/ddate* files are updated to the new format */etc/dumpdates* format. The effect of this option is to convert between the old, obsolete format and to the new format. If the **J** option is invoked, all other options are ignored, and *dump* terminates immediately.

s The size of the dump tape is specified in feet. The number of feet is taken from the next *argument*. When the specified size is reached, *dump* will wait for reels to be changed. The default tape size is 2300 feet.

d The density of the tape, expressed in BPI, is taken from the next *argument*. This is used in calculating the amount of tape used per reel. The default is 1600.

W *Dump* tells the operator what file systems need to be dumped. This information is gleaned from the files */etc/dumpdates* and */etc/fstab*. The **W** option causes *dump* to print out, for each file system in */etc/dumpdates* the most recent dump date and level, and highlights those file systems that should be dumped. If the **W** option is set, all other options are ignored, and *dump* exits immediately.

w Is like W, but prints only those filesystems which need to be dumped.

n Whenever *dump* requires operator attention, notify by means similar to a *wall(1)* all of the operators in the group "operator".

If no arguments are given, the *key* is assumed to be **9u** and a default file system is dumped to the default tape.

Dump requires operator intervention on these conditions: end of tape, end of dump, tape write error, tape 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** key, *dump* interacts with the operator on *dump's* control terminal at times when *dump* can no longer proceed, or if something is grossly wrong. All questions *dump* poses must be answered by typing "yes" or "no", appropriately.

Since making a dump involves a lot of time and effort for full dumps, *dump* checkpoints itself at the start of each tape volume. If writing that volume fails for some reason, *dump* will, with operator permission, restart itself from the checkpoint after the old tape has been rewound and

removed, and a new tape has been mounted.

Dump tells the operator what is going on at periodic intervals, including usually low estimates of the number of blocks to write, the number of tapes it will take, the time to completion, and the time to the tape change. The output is verbose, so that others know that the terminal controlling *dump* is busy, and will be for some time.

Now a short suggestion on how to perform dumps. Start with a full level 0 dump

dump 0un

Next, dumps of active file systems are taken on a daily basis, using a modified Tower of Hanoi algorithm, with this sequence of dump levels:

3 2 5 4 7 6 9 8 9 9 ...

For the daily dumps, a set of 10 tapes per dumped file system is used on a cyclical basis. Each week, a level 1 dump is taken, and the daily Hanoi sequence repeats with 3. For weekly dumps, a set of 5 tapes per dumped file system is used, also on a cyclical basis. Each month, a level 0 dump is taken on a set of fresh tapes that is saved forever.

FILES

/dev/rrp1g	default filesystem to dump from
/dev/rmt8	default tape unit to dump to
/etc/ddate	old format dump date record (obsolete after −J option)
/etc/dumpdates	new format dump date record
/etc/fstab	Dump table: file systems and frequency
/etc/group	to find group *operator*

SEE ALSO

restor(1), dump(5), dumpdir(1), fstab(5)

DIAGNOSTICS

Many, and verbose.

BUGS

Sizes are based on 1600 BPI blocked tape; the raw magtape device has to be used to approach these densities. Fewer than 32 read errors on the filesystem are ignored. Each reel requires a new process, so parent processes for reels already written just hang around until the entire tape is written.

It would be nice if *dump* knew about the dump sequence, kept track of the tapes scribbled on, told the operator which tape to mount when, and provided more assistance for the operator running *restor*.

NAME

dumpdir — print the names of files on a dump tape

SYNOPSIS

/etc/dumpdir [f filename]

DESCRIPTION

Dumpdir is used to read magtapes dumped with the *dump* command and list the names and inode numbers of all the files and directories on the tape.

The f option causes *filename* as the name of the tape instead of the default.

FILES

default tape unit varies with installation
rst*

SEE ALSO

dump(1), restor(1)

DIAGNOSTICS

If the dump extends over more than one tape, it may ask you to change tapes. Reply with a new-line when the next tape has been mounted.

BUGS

There is redundant information on the tape that could be used in case of tape reading problems. Unfortunately, *dumpdir* doesn't use it.

NAME

restor – incremental file system restore

SYNOPSIS

/etc/restor key [argument ...]

DESCRIPTION

Restor is used to read magtapes dumped with the *dump* command. The *key* specifies what is to be done. *Key* is one of the characters rRxt optionally combined with f.

f Use the first *argument* as the name of the tape instead of the default.

r or R The tape is read and loaded into the file system specified in *argument*. This should not be done lightly (see below). If the key is R *restor* asks which tape of a multi volume set to start on. This allows restor to be interrupted and then restarted (an *icheck* –s must be done before restart).

x Each file on the tape named by an *argument* is extracted. The file extracted is placed in a file with a numeric name supplied by *restor* (actually the inode number). In order to keep the amount of tape read to a minimum, the following procedure is recommended:

Mount volume 1 of the set of dump tapes.

Type the *restor* command.

Restor will announce whether or not it found the files, give the number it will name the file, and rewind the tape.

It then asks you to 'mount the desired tape volume'. Type the number of the volume you choose. On a multivolume dump the recommended procedure is to mount the last through the first volume in that order. *Restor* checks to see if any of the files requested are on the mounted tape (or a later tape, thus the reverse order) and doesn't read through the tape if no files are. If you are working with a single volume dump or the number of files being restored is large, respond to the query with '1' and *restor* will read the tapes in sequential order.

If you have a hierarchy to restore you can use *dumpdir*(8) to produce the list of names and a shell script to move the resulting files to their homes.

t Print the date the tape was written and the date the filesystem was dumped from.

The r option should only be used to restore a complete dump tape onto a clear file system or to restore an incremental dump tape onto this. Thus

```
/etc/mkfs /dev/rrp0g 145673
restor r /dev/rrp0g
```

is a typical sequence to restore a complete dump. Another *restor* can be done to get an incremental dump in on top of this.

A *dump* followed by a *mkfs* and a *restor* is used to change the size of a file system.

FILES

default tape unit varies with installation
rst*

SEE ALSO

dump(8), mkfs(8), dumpdir(8)

DIAGNOSTICS

There are various diagnostics involved with reading the tape and writing the disk. There are also diagnostics if the i-list or the free list of the file system is not large enough to hold the dump.

If the dump extends over more than one tape, it may ask you to change tapes. Reply with a new-line when the next tape has been mounted.

BUGS

There is redundant information on the tape that could be used in case of tape reading problems. Unfortunately, *restor* doesn't use it.