9.9. Quotas

The quota system allows limiting disk space allocated to a user or group of users. To set it up, you must have a kernel that supports it (compiled with the `CONFIG_QUOTA` option) — as is the case with Debian kernels. The quota management software is found in the *quota* Debian package.

To activate quota in a filesystem, you have to indicate the usrquota and grpquota options in `/etc/fstab` for the user and group quotas, respectively. Rebooting the computer will then update the quotas in the absence of disk activity (a necessary condition for proper accounting of already used disk space).

The `edquota` *`user`* (or `edquota -g` *`group`*) command allows you to change the limits while examining current disk space usage.

GOING FURTHER

Defining quotas with a script

The `setquota` program can be used in a script to automatically change many quotas. Its `setquota(8)` manual page details the syntax to use.

The quota system allows you to set four limits:

- two limits (called "soft" and "hard") refer to the number of blocks consumed. If the filesystem was created with a block-size of 1 kibibyte, a block contains 1024 bytes from the same file. Unsaturated blocks thus induce losses of disk space. A quota of 100 blocks, which theoretically allows storage of 102,400 bytes, will, however, be saturated with just 100 files of 500 bytes each, only representing 50,000 bytes in total.
- two limits (soft and hard) refer to the number of inodes used. Each file occupies at least one inode to store information about it (permissions, owner, timestamp of last access, etc.). It is thus a limit on the number of user files.

A "soft" limit can be temporarily exceeded; the user will simply be warned that they are exceeding the quota by the `warnquota` command, which is usually invoked by `cron`. A "hard" limit can never be exceeded: the system will refuse any operation that will cause a hard quota to be exceeded.

VOCABULARY

Blocks and inodes

The filesystem divides the hard drive into blocks — small contiguous areas. The size of these blocks is defined during creation of the filesystem, and generally varies between 1 and 8 kibibytes.

A block can be used either to store the real data of a file, or for meta-data used by the filesystem. Among this meta-data, you will especially find the inodes. An inode uses a block on the hard drive (but this block is not taken into consideration in the block quota, only in the inode quota), and contains both the information on the file to which it corresponds (name, owner, permissions, etc.) and the pointers to the data blocks that are actually used. For very large files that occupy more blocks than it is possible to reference in a single inode, there is an indirect block system; the inode references a list of blocks that do not directly contain data, but another list of blocks.

With the `edquota -t` command, you can define a maximum authorized "grace period" within which a soft limit may be exceeded. After this period, the soft limit will be treated like a hard limit, and the user will have to reduce their disk space usage to within this limit in order to be able to write anything to the hard drive.

GOING FURTHER

Setting up a default quota for new users

To automatically setup a quota for new users, you have to configure a template user (with `edquota` or `setquota`) and indicate their user name in the `QUOTAUSER` variable in the `/etc/adduser.conf` file. This quota configuration will then be automatically applied to each new user created with the `adduser` command.

9.10. Backup

Making backups is one of the main responsibilities of any administrator, but it is a complex subject, involving powerful tools which are often difficult to master.

Many programs exist, such as `amanda`, `bacula`, `BackupPC`. Those are client/server system featuring many options, whose configuration is rather difficult. Some of them provide user-friendly web interfaces to mitigate this. But Debian contains dozens of other backup software covering all possible use cases, as you can easily confirm with `apt-cache search backup`.

Rather than detailing some of them, this section will present the thoughts of the Falcot Corp administrators when they defined their backup strategy.

At Falcot Corp, backups have two goals: recovering erroneously deleted files, and quickly restoring any computer (server or desktop) whose hard drive has failed.

9.10.1. Backing Up with `rsync`

Backups on tape having been deemed too slow and costly, data will be backed up on hard drives on a dedicated server, on which the use of software RAID (see section 12.1.1, "Software RAID" page 328) will protect the data from hard drive failure. Desktop computers are not backed up individually, but users are advised that their personal account on their department's file server will be backed up. The `rsync` command (from the package of the same name) is used daily to back up these different servers.

BACK TO BASICS

The hard link, a second name for the file

A hard link, as opposed to a symbolic link, cannot be differentiated from the linked file. Creating a hard link is essentially the same as giving an existing file a second name. This is why the deletion of a hard link only removes one of the names associated with the file. As long as another name is still assigned to the file, the data therein remain present on the filesystem. It is interesting to note that, unlike a copy, the hard link does not take up additional space on the hard drive.

A hard link is created with the `ln` *`target link`* command. The *link* file is then a new name for the *target* file. Hard links can only be created on the same filesystem, while symbolic links are not subject to this limitation.