

# Re: File Management

---

*Source:* <http://coding.derkeiler.com/Archive/Perl/perl.beginners/2005-07/msg00541.html>

---

- *From:* [fxn@xxxxxxxxxxxxx](mailto:fxn@xxxxxxxxxxxxx) (Xavier Noria)
  - *Date:* Sat, 23 Jul 2005 09:46:05 +0200
- 

On Jul 23, 2005, at 7:56, Joel Divekar wrote:

We have a windoz based file server with thousand of user accounts. Each user is having thousand of files in his home directory. Most of these files are duplicate / modified or updated version of the existing files. These files are either .doc or .xls or .ppt files which are shared by groups or departments.

Due to this my server is having terabyte of data, most of which are redundant and our sysadmin has tough time maintaining storage space.

For this I though of writing a small program to locate similar or duplicate files stored on my file server and delete them with the help of the user. The program should work very fast and I don't know from where to start.

Well, to come with the right solution one would need to play around a bit in the server. I propose an approach based on the description above, just in case it helps.

Since there is big number of files, we need to walk the tree at least once, and store some data for each file to compare, I would choose a quick test first that speeds up the tree traversal as much as possible, purges the tree, and then do heavier operations on the remaining candidates.

For instance:

1. Walk the tree and build a map using `-s`

## Re: File Management

size -> filenames

2. Purge the entries that have just one filename associated, since they have no duplicate for sure
3. Work on the rest of the entries.

If the map in (1) gets too big to fit in a hash in memory you could use some sort of database table, maybe something simple to setup as SQLite. For (3), if the number of candidates is still not small you could make an additional refinement constructing a map with MD5s, until you get a small number of files and can compare their contents.

Trace as less as possible the tree traversal, printing to the console a debug line for each file, for instance, would slow down the script by orders of magnitude.

Then, to maintain that tree, I don't know, maybe the time to do this is assumable? Running that procedure periodically might be a simple but good enough solution.

-- fxn  
.