**Ctime-mtime-atime**

http://www.unix.com/tips-and-tutorials/20526-mtime-ctime-atime.html

------------------------------------------------------------------------

Unix keeps 3 timestamps for each file: **mtime, ctime, and atime.** Most people seem to understand atime (access time), it is when the file was last read. There does seem to be some confusion between mtime and ctime though. ctime is the inode change time while mtime is the file modification time. "Change" and "modification" are pretty much synonymous. There is no clue to be had by pondering those words. Instead you need to focus on what is being changed. **mtime changes when you write to the file.** **It is the age of the data in the file. Whenever mtime changes, so does ctime. But ctime changes a few extra times. For example, it will change if you change the owner or the permissions on the file**.

Let's look at a concrete example. We run a package called Samba that lets PC's access files. To change the Samba configuration, I just edit a file called smb.conf. (This changes mtime and ctime.) I don't need to take any other action to tell Samba that I changed that file. Every now and then Samba looks at the mtime on the file. If the mtime has changed, Samba rereads the file. Later that night our backup system runs. It uses ctime, which also changed so it backs up the file. But let's say that a couple of days later I notice that the permissions on smb.conf are 666. That's not good..anyone can edit the file. So I do a "chmod 644 smb.conf". This changes only ctime. Samba will not reread the file. But later that night, our backup program notices that ctime has changes, so it backs up the file. That way, if we lose the system and need to reload our backups, we get the new improved permission setting.

Here is a second example. Let's say that you have a data file called employees.txt which is a list of employees. And you have a program to print it out. The program not only prints the data, but it obtains the mtime and prints that too. Now someone has requested an employee list from the end of the year 2000 and you found a backup tape that has that file. Many restore programs will restore the mtime as well. When you run that program it will print an mtime from the end of the year 2000. But the ctime is today. So again, our backup program will see the file as needing to be backed up.

Suppose your restore program did not restore the mtime. You don't want your program to print today's date. Well no problem. mtime is under your control. You can set it to what ever you want. So just do:

Code:

**$ touch -t 200012311800 employees.txt**

This will set mtime back to the date you want and it sets ctime to now. You have complete control over mtime, but the system stays in control of ctime. So mtime is a little bit like the date on a letter while ctime is like the postmark on the envelope.