MAKING HISTORY A RECORD

Call it a artifact from my last decade of work. Call it a wide streak of laziness. Call it being overly complicated. But regardless, I have found that I generally try to chose a solution that does not cost a lot of money. I like solutions that leverage existing capabilities, I find that elegant. And since I am writing this without unconsenting voices, I guess I get to have my say regardless. This specific issue is around the use of the linux shell. When you have just got a couple of users, you pretty much know who is going to do what, and if something goes wrong you go and flush that persons fish down the loo and explain why what they did was a bad thing. But in larger organisations or those with regulative requirements, you need a better idea of what your users did. Now all linux shells come with the history file idea, but that is generally disregarded as any user can change their own one and thus your record is useless. Now you could patch your shell, you could do all sorts of things, but I like doing something with what I already have. So lets make the history files read-only for the users.

Shells
I am aiming this paper at the use of the *bash* shell, so my steps relate to that. But the astute reader will note that it can easily be amended to cater for your shell of choice. With that in mind, lets make sure the only shells the users can use are bash and nothing. So comment out everything else, this will leave you with something like;

```
> cat /etc/shells
#/bin/sh
/bin/bash
/sbin/nologin
#/bin/tcsh
#/bin/csh
#/bin/ksh
#/bin/zsh
```

Now lets remove the ability for the users to even try to change their shell. Remove the suid bit from the *chsh* binary so you get this;

```
> ls -l `which chsh`
-rwx------ 1 root root 19128 May 24  2008 /usr/bin/chsh
```

Shell Setup
Now we want to setup some variables so that when the users login they get what we want and that is it. So add the following to your */etc/profile*;

```
HISTSIZE=1000
HISTTIMEFORMAT="%F %T "
readonly HISTTIMEFORMAT
readonly HISTFILE
readonly HISTSIZE
readonly HISTFILESIZE
```

Now the one line which does not really need to be there in the above settings is the *HISTTIMEFORMAT* setting. But what that is doing is adding a timestamp to your history lines, so you not only see what the user typed but also when they typed it.

User Environment
Now you need to setup the user environment somewhat. First you are going to make sure that each user has a history file setup, then you are going to make sure that only the user has read or write access and lastly you are going to set the attributes of the file so that it can only be appended to, nothing else. I use a simple little script to do this for myself;

```
> cat ./fix_history
for PERSON in bob sue john jimmy
 do
  echo "Doing $PERSON..."
  touch /home/$PERSON/.bash_history
  chown $PERSON:root /home/$PERSON/.bash_history
  chmod 600 /home/$PERSON/.bash_history
  chattr +a /home/$PERSON/.bash_history
```

```
done
```

Results

With all that done, let your users work for a bit and then take a look. You should see something like the following;

```
 115  2009-05-26 12:16:35 su -
 116  2009-05-26 12:16:35 ls -l
 117  2009-05-26 12:16:35 mkdir backup
 118  2009-05-26 12:16:35 ls -l
```

Final Words

Yes I realise that there are ways this can be circumvented, but it does raise the bar on trying to a user trying to hide something. And if you really are in a situation where your users are both knowledgable and malicous, then more drastic measures are probably needed anyway. But this is a simple easy step which can be used in most situations to add another layer of accountability in. As always, try it out, play around, have fun and learn.