How to make sure your Linux system users won't hide or delete their .bash_history / Securing .bash_history file - Protect Linux system users shell history

Author : admin

If you're running multi user login Linux system, you have probably realized that there are some clever users that prefer to prevent their command line executed commands to be logged in .bash_history.
To achieve that they use a number of generally known methodologist to prevent the Linux system from logging into their $HOME/.bash_history file (of course if running bash as a default user shell).
This though nice for the user is a real nightmare for the sysadmin, since he couldn't keep track of all system command events executed by users. For instance sometimes an unprivilegd user might be responsible for executing a malicious code which crashes or breaks your server.
This is especially unpleasent, because you will find your system crashed and if it's not some of the system services that causes the issue you wonâ€™t even be able to identify which of all the users is the malicious user account and respectively the code excecuted which fail the system to the ground.
In this post I will try to tell you a basic ways that some malevolent users might use to hide their bash history from the system administrator.
I will also discuss a few possible ways to assure your users .bash_history keeps intact and possibly the commands executed by your users gets logged in in their.
The most basic way that even an unexperienced shell user will apply if he wants to prevent his **.bash_history** from sys admins review would be of directly wiping out the .bash_history file from his login account or alternatively emptying it with commands like:

malicious-user@server:~$ rm -f. bash_history
ormalicious-user@server:~# cat /dev/null > ~/.bash_history

In order to prevent this type of attack against cleaning the .bash_history you can use the **chattr** command.
To counter attack this type of history tossing method you can set your **malicious-user .bash_historyâ€™s file the (append only flag) with chattr** like so:

root@server:~# cd /home/malicious-user/
root@server:~# chattr +a .bash_history

Itâ€™s also recommended that the immunable flag is placed to the file **~/.profile** in user home

root@server:~# chattr +i ~/.profile

It would be probably also nice to take a look at all **chattr command attributes** since the command is like swiss army knife for the Linux admin:
Here is all available flags that can be passed to chattr
append only (a)
compressed (c)
don~@~Yt update atime (A)
synchronous directory updates (D)
synchronous updates (S)
data journalling (j)
no dump (d)
top of directory hierarchy (T)
no tail-merging (t)
secure deletion (s)
undeletable (u)
immutable (i)

Itâ€™s also nice that setting the **â€œappend onlyâ€•** flag in to the user .bash_history file prevents the user to link the .bash_history file to /dev/null like so:

malicious-user@server:~$ ln -sf /dev/null ~/.bash_history
ln: cannot remove `.bash_history': Operation not permitted

malicious-user@server:~$ echo > .bash_history
bash: .bash_history: Operation not permitted

However this will just make your .bash_history append only, so the user trying to execute **cat /dev/null > .bash_history** wonâ€™t be able to truncate the content of **.bash_history**.

Unfortunately he will yet be able to delete the file with **rm** so this type of securing your .bash_history file from being overwritten is does not completely guarantee you that user commands will get logged.
Also in order to prevent user to play tricks and escape the .bash_history logging by changing the default bash shell variables for HISTFILE an d HISTFILESIZE, exporting them either to a different file location or a null file size.
You have to put the following bash variables to be loaded in **/etc/bash.bashrc** or in **/etc/profile**

```
# #Prevent unset of histfile, /etc/profile
HISTFILE=~/.bash_history
HISTSIZE=10000
HISTFILESIZE=999999
# Don't let the users enter commands that are ignored# in the history file
HISTIGNORE=""
HISTCONTROL=""
readonly HISTFILE
```

readonly HISTSIZE
readonly HISTFILESIZE
readonly HISTIGNORE
readonly HISTCONTROL
export HISTFILE HISTSIZE HISTFILESIZE HISTIGNORE HISTCONTROL

everytime a user logs in to your Linux system the bash commands above will be set.
The above tip is directly taken from Securing debian howto which by the way is quite an interesting and nice reading for system administrators :)

If you want to apply **an append only attribute to all user .bash_history to all your existing Linux server system users** assuming the default users directory is **/home** in bash you can execute the following 1 liner shell code:

#Set .bash_history as attr +a
2. find /home/ -maxdepth 3|grep -i bash_history|while read line; do chattr +a "$line"; done

Though the above steps will stop some of the users to voluntary clean their **.bash_history** history files it wonâ€™t a 100% guaranttee that a good cracker wonâ€™t be able to come up with a way to get around the imposed .bash_history security measures.

One possible way to get around the user command history prevention restrictions for a user is to simply using another shell from the ones available on the system:
Here is an example:

malicious-user:~$ /bin/csh
malicious-user:~>

csh shell logs by default to the file .history

Also as far as I know it should be possible for a user to simply delete the **.bash_history** file overwritting all the .bash_history keep up attempts up-shown.
If you need a complete statistics about accounting youâ€™d better take a look at The GNU Accounting Utilities

In Debian the **GNU Accounting Utilities** are available as a package called **acct**, so installation of acct on Debian is as simple as:

debian:~# apt-get install acct

I wonâ€™t get into much details about acct and would probably take a look at it in my future posts.
For complete .bash_history delete prevention maybe the best practice is to useg grsecurity (grsec)

Hopefully this article is gonna be a step further in tightening up your Server or Desktop Linux based system security and will also give you some insight on .bash_history files :) .