History Command Examples In Linux

The ‘history’ command available in Bash can be used to simply display your shell history, however there’s also a whole lot more that you can do with it, which we’ll demonstrate here.

Bash history allows us to quickly see what has been executed previously on a system, allowing you to hold users at least somewhat accountable for their actions (more on this later). It’s also useful if you’ve run something before and forgot the command, I can’t begin to tell you the number of times that I’ve done this!

How To Use History – Command Examples

- ### 1. Print History

  In its most simple form, you can run the ‘history’ command by itself and it will simply print out the bash history of the current user to the screen. Commands are numbered, with older commands at the top and newer commands at the bottom.

```
[user@centos7 ~]$ history
    1  ip a
    2  exit
    3  ls -la
    4  pwd
[user@centos7 ~]#
```

  The history is stored in the ~/.bash_history file by default. You could also run ‘cat ~/.bash_history’ which is similar but does not include the line numbers or formatting.

- ### 2. Print ‘n’ Lines

  While the default is to print all history lines, you can specify a number after the history command to output this amount of the most recent lines.

```
[user@centos7 ~]$ history 3
   16  passwd
   17  getenforce
   18  history 3
```

- ### 3. Repeat Most Recent Command

  The most recent command can be executed simply by entering ‘!!’.

```
[root@centos7 ~]# date
Sun Aug 28 03:14:55 PDT 2016
[root@centos7 ~]# !!
date
Sun Aug 28 03:14:57 PDT 2016
```

Alternatively you can simply press the ‘up’ arrow key to display the last command and then press enter to execute it.

- ## 4. Repeat Specific Command

As shown above, the bash history command displays line numbers. It is possible to repeat a command by specifying its line number.

```
[root@centos7 ~]# history 2
  101  date
  102  history 2
[root@centos7 ~]# !101
date
Sun Aug 28 03:18:55 PDT 2016
```

In this example, the ‘date’ command was the 101st line in the history file, and we can run it again with ‘!101’. Note that the line numbers can change, especially if your history file fills up, so don’t rely on the same number always pointing to the same command.

- ## 5. Repeat Command Starting With A String

We can repeat the last command starting with a specified string. This is done with !string, where string is the start of a previously executed command.

```
[root@centos7 ~]# systemctl start httpd
[root@centos7 ~]# systemctl stop chronyd
[root@centos7 ~]# systemctl restart chronyd
[root@centos7 ~]# !systemctl
systemctl restart chronyd
```

As shown the most recent command that started with ‘systemctl’ has been run again.

While useful, this can obviously be dangerous if the last command is actually different from what you expect. You can run this with ‘:p’ on the end to instead print the command rather than execute it straight away.

```
[root@centos7 ~]# !systemctl:p
systemctl stop chronyd
```

This has not actually performed the restart, it merely displays the command.

- ## 6. Piping History

We can of course pipe the output of the history command into many other useful commands, such as less or grep. When piping into less we can scroll through the output of the history file rather than having it all output to the terminal. By outputting to grep we can search for commands that have been run previously.

```
[root@centos7 ~]# history | grep httpd
   65  yum install httpd -y
  106  systemctl stop httpd
  107  systemctl start httpd
  117  history | grep httpd
```

- ## 7. Write To History File

  Usually the history file is written to upon logout, therefore if you have an SSH session that has timed out you will not have your history from that session when you log back in. We can force the current history to write to the users ~/.bash_history file with the -w option.

```
[root@centos7 ~]# history -w
```

- ## 8. Clear History File

  We can clear all contents of the history file with the -c command.

```
[root@centos7 ~]# history -c
```

  Note that this will only clear the history in memory, the changes will be written when the user logs out however we can save the changes to the .bash_history file immediately by running 'history -w' afterwards.

  We could also delete or otherwise remove the contents of the ~/.bash_history file, however keep in mind that the current history is written to the file at log out, so if you delete the file then log out the history of your current session will still be saved.

- ## 9. Delete Specific Line

  Clearing the whole history file may be overkill, we can instead delete a specific line number from the history file with the -d option.

```
[root@centos7 ~]# history | grep password
  121  Sun 28 Aug 2016 03:33:11 AM PDT mysql -u root -p
oops_this_is_my_password
  122  Sun 28 Aug 2016 03:33:19 AM PDT history | grep password
[root@centos7 ~]# history -d 121
[root@centos7 ~]# history | grep password
  121  Sun 28 Aug 2016 03:33:19 AM PDT history | grep password
  123  Sun 28 Aug 2016 03:33:29 AM PDT history | grep password
```

  In this example the user accidentally left their MySQL password in the bash history at line 121, which we then remove with the -d option and specify the line number to remove. We can see that line 121 is now our history command, as mentioned previously be aware that the line numbers can change so they should not be relied on to remain static.

- ## 10. Run Single Command Without Logging

  We can run a single command without it being logged to the bash history.

```
[root@centos7 ~]# echo "secret command";history -d $(history 1)
secret command
```

  This deletes the most recently run command straight after execution.

- ## 11. Run All Commands Without Logging

  Additionally we can unset the history file variable for the current bash session which will prevent all history for the current session from being stored.

```
[root@centos7 ~]# echo $HISTFILE
/root/.bash_history
[root@centos7 ~]# unset HISTFILE
[root@centos7 ~]# echo $HISTFILE
```

  Note that this is not permanent, when you log out and log back in HISTFILE will be reset back to the default. This example will allow you to have an unlogged session, though you could specify the unset in ~/.bashrc to never log history.

- ## 12. Ignore Specific Commands

  We can specify a list of commands that should never be logged in the history file with the $HISTIGNORE variable, which is not set by default.

```
[root@centos7 ~]# echo 'export HISTIGNORE="ls:cd"' >> ~/.bashrc
```

  As before when this file is written to you need to log out and log back in for it to execute.

```
[root@centos7 ~]# ls
anaconda-ks.cfg  new_history
[root@centos7 ~]# pwd
/root
[root@centos7 ~]# cd
[root@centos7 ~]# echo hi
hi
[root@centos7 ~]# history 5
  123  history
  124  du
  125  pwd
  126  echo hi
  127  history 5
```

  As shown the ‘ls’ and ‘cd’ commands that we have run were not stored in the logs.

- ## 13. Increase History Size

By default 1000 lines of history will be stored, as per the values stored in the $HISTSIZE and $HISTFILESIZE variables.

```
[root@centos7 ~]# echo $HISTFILESIZE
1000
[root@centos7 ~]# echo $HISTSIZE
1000
```

The default for all users is stored in the /etc/profile file, this can be modified or you can otherwise append the following lines to the bottom of ~/.bashrc which will apply to that user at next login.

```
HISTSIZE=2000
HISTFILESIZE=2000
```

Note that if your history file fills up, the oldest commands will be rotated out first and removed as new lines are added in.

- ## 14. Add Timestamps To History

As you may have noticed by default we are not able to see the date and time that commands were executed, merely their order. We can set the $HISTTIMEFORMAT variable with a specific date and time format, the easiest option is to use %c as shown below.

```
echo 'export HISTTIMEFORMAT="%c "' >> ~/.bashrc
```

Once this user logs out and back in for the export to execute, the existing history file will show all contents as executing at the exact same time as the time information was not previously recorded. From here onward however, the date and time will be stored with each command in the bash history file.

```
[root@centos7 ~]# history 5
   39  Sun 28 Aug 2016 02:37:54 AM PDT firewall-cmd --add-service=http
--permanent
   40  Sun 28 Aug 2016 02:37:54 AM PDT firewall-cmd --reload
   41  Sun 28 Aug 2016 02:37:54 AM PDT tailf /var/log/messages
   42  Sun 28 Aug 2016 02:37:54 AM PDT restorecon -v
/var/www/html/index.html
   43  Sun 28 Aug 2016 02:49:27 AM PDT history 5
```

- ## 15. Change History File Location

By default the bash history is written to ~/.bash_history, this is set in the $HISTFILE variable as shown below.

```
[root@centos7 ~]# echo $HISTFILE
/root/.bash_history
[root@centos7 ~]# su - user
```

```
[user@centos7 ~]$ echo $HISTFILE
/home/user/.bash_history
```

We can set a custom file in ~/.bashrc as shown below.

```
[root@centos7 ~]# echo 'export HISTFILE="/root/new_history"' >>
~/.bashrc
```

After logging out and back in all history will be stored in /root/new_history instead.

- ## 16. Do Not Store Duplicate Commands

By default /etc/profile sets the $HISTCONTROL variable to ‘ignoredups’ which will ignore duplicate commands that are run one after the other.

For example if we execute the ‘pwd’ command multiple times, it will only show once in the history.

```
[root@centos7 ~]# pwd
/root
[root@centos7 ~]# pwd
/root
[root@centos7 ~]# pwd
/root
[root@centos7 ~]# pwd
/root
[root@centos7 ~]# history | grep pwd
    1  Sun 28 Aug 2016 04:01:07 AM PDT pwd
    2  Sun 28 Aug 2016 04:01:15 AM PDT history | grep pwd
```

As this is in the /etc/profile file, it is set for all users on the system by default.

- ## 17. Reverse Search

While we can browse previous commands with the techniques previously listed, my favourite is reverse search which is executed with ‘ctrl+r’.

After pressing ‘ctrl+r’ you will see the the (reverse-i-search)`’: prompt, at this point you can start typing a command that has previously been executed and it will display the most recent command. You can cycle back further through previous commands that also contain this string by pressing ‘ctrl+r’ again and again until you find what you’re after.

```
(reverse-i-search)`httpd': systemctl start httpd
```

In this example I started typing httpd and it showed that my most recent command was starting Apache. Once you’ve found what you’re after, press enter to execute it.

All history should also be taken with a grain of salt, as it is very easy to modify as by default a user has write permissions on their own ~/.bash_history file so they can modify it however they want, including deleting the contents to cover their tracks.

You could instead look at sending bash history to an external syslog server so that it cannot be modified, but that’s a story for another time.