How to keep track of All User accounts executed commands, highest CPU consumers and user times on Linux

Author : admin

For people interested into statistics of how Linux existing users are spending, there log in times and what kind of commands each of users is executing, take a look at **acct**
acct is existing on all mainstream Linux distributions is a great sysadmin tool. acct is a great tool whether you have a system where a multitude of users you don't trust has to be monitored. It is an absolutely must have for anyone willing to run, lets say **experimental honeypot** or **free shell host**. acct is useful for paranoid sysadmins who like to always knows what there users are running as well as in situation where some of users is suspected to be a potential cracker trying to root the host.

Below is description of acct package on Debian:

 # apt-cache show acct| grep -i description -A 8
Description: The GNU Accounting utilities for process and login accounting
GNU Accounting Utilities is a set of utilities which reports and summarizes
data about user connect times and process execution statistics.
.
"Login accounting" provides summaries of system resource usage based on connect
time, and "process accounting" provides summaries based on the commands
executed on the system.
.
The 'last' command is provided by the sysvinit package and not included here.

To start using acct, just install it with usual:

 # apt-get install --yes acct

(Whether on Debian / Ubuntu Linux)**;**

On Fedora, CentOS and RHEL and other RPM based Linuxes issue;

yum --y install psacct

On deb based Linux distributions, whether acct collects statistics is controlled via:

/etc/default/acct

 # cat /etc/default/acct

```
 # Defaults for acct

# If you want to keep acct installed, but not started automatically, set this
# variable to 0. Because /etc/cron.daily/acct calls the initscript daily, it is
# not sufficient to stop acct once after booting if your machine remains up.
ACCT_ENABLE="1"

# Amount of days that the logs are kept.
ACCT_LOGGING="30"
```

After installed to start collecting user "process accounting" data run acct via init script;

 # /etc/init.d/acct start

```
Turning on process accounting, file set to '/var/log/account/pacct'.
Done..
```

The file gathering info on system usage, CPU load, user ran commands */var/log/account/psacct* is a binary and unreadable tailing it with **tail -f** .

On *CentOS / Fedora Linux* to Enable acct account statistics gathering in future boot and from present moment on do;

```
# chkconfig psacct on
# /etc/init.d/psacct start
```

1. Find out all commands executed by Linux user account (lastcomm)

Once user accounting is running to get information of every command ever executed on user shell use **lastcomm** cmd. For example:

 # lastcomm hipo

```
bash           F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
bash           F    hipo     pts/1      0.03 secs Tue Feb  5 00:20
sed                 hipo     pts/1      0.00 secs Tue Feb  5 00:20
bash           F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
uname               hipo     pts/1      0.00 secs Tue Feb  5 00:20
```

```
bash         F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
dircolors         hipo     pts/1      0.00 secs Tue Feb  5 00:20
bash         F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
bash         F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
bash         F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
uname             hipo     pts/1      0.00 secs Tue Feb  5 00:20
bash         F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
bash         F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
ls             hipo     pts/1      0.00 secs Tue Feb  5 00:20
bash         F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
bash         F    hipo     pts/1      0.03 secs Tue Feb  5 00:20
sed              hipo     pts/1      0.00 secs Tue Feb  5 00:20
bash         F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
uname             hipo     pts/1      0.00 secs Tue Feb  5 00:20
bash         F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
id             hipo     pts/1      0.00 secs Tue Feb  5 00:20
mesg             hipo     pts/1      0.00 secs Tue Feb  5 00:20
verse            hipo     pts/1      0.00 secs Tue Feb  5 00:20
cowrand            hipo     pts/1      0.00 secs Tue Feb  5 00:20
cowsay           hipo     pts/1      0.03 secs Tue Feb  5 00:20
cowrand      F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
head             hipo     pts/1      0.00 secs Tue Feb  5 00:20
tail           hipo     pts/1      0.00 secs Tue Feb  5 00:20
head             hipo     pts/1      0.00 secs Tue Feb  5 00:20
ls             hipo     pts/1      0.00 secs Tue Feb  5 00:20
cowrand      F    hipo     pts/1      0.00 secs Tue Feb  5 00:20
awk              hipo     pts/1      0.00 secs Tue Feb  5 00:20
wc               hipo     pts/1      0.00 secs Tue Feb  5 00:20
ls             hipo     pts/1      0.00 secs Tue Feb  5 00:20
```

A lot of the initial commands shown to run on pts/1 is not actual commands, by the user but are just stuff run on user login time via **/etc/bash.bashrc, /etc/profile, ~/.bashrc. ~/.bash_profile**.

lastcomm displayed output from 2nd column is a special flag giving more information on how and for what purpose command was executed. In above output
F - indicates the command run after a fork.
X - is returned if a command exit with SIGTERM (kill signal)
D - in case of generated command core dump (D is good one to look for whether checking a suspicious user profile, as it is so common exploits use core dumping to get root superuser access)
S - means the command is run with superuser privileges (this one you will see usually whether inspecting user profile of a cracker who run exploit using core dump - a lot of Ds followed by some shell code to run as superuser)

2. Get statistics on CPU use time of services (daemons) and user accounts

psacct is very handy, whether you have CPU server overloads and you have difficulty finding out what

are the "CPU hungry processes". To get those use *summarized accounting information* tool;

 # sa -m

```
                2619     31.06re      0.54cp        0avio      2907k
root            2448     30.19re      0.52cp        0avio      2817k
www-data          33      0.06re      0.02cp        0avio      3687k
hipo              72      0.15re      0.01cp        0avio      6217k
qscand            11      0.36re      0.00cp        0avio      5326k
vpopmail          48      0.25re      0.00cp        0avio      1486k
qmails             6      0.00re      0.00cp        0avio       968k
sshd               1      0.04re      0.00cp        0avio     12632k
```

-m (prints user summary).

3. Find all system users running certain commands

Another good use of **lastcomm** command is to grep over all users executed command for precise commands of interest. One very good use case is if you catch a system abuser running certain exploit or DoS tool on the host and you want to make sure no-one else on the system doesn't try running it.

 # lastcomm ls

```
ls              www-data __       0.00 secs Tue Feb  5 00:40
ls              www-data __       0.00 secs Tue Feb  5 00:30
ls              hipo     pts/7    0.00 secs Tue Feb  5 00:20
ls              hipo     pts/1    0.00 secs Tue Feb  5 00:20
ls              hipo     pts/1    0.00 secs Tue Feb  5 00:20
ls              hipo     pts/1    0.00 secs Tue Feb  5 00:20
ls              hipo     pts/1    0.00 secs Tue Feb  5 00:20
ls              hipo     pts/1    0.00 secs Tue Feb  5 00:20
ls              www-data __       0.00 secs Tue Feb  5 00:20
ls              root     pts/0    0.00 secs Tue Feb  5 00:10
ls              root     pts/0    0.00 secs Tue Feb  5 00:10
ls              www-data __       0.00 secs Tue Feb  5 00:10
```

4. Get statistics of most active system users in hours

There is one tool called ac, which is similar in what it does to **last** command, just like last it uses /var/log/wtmp binary log file to get its user login times stats . The difference is ac provides more and better structured user login time length info.

Its very useful if you want to have idea, which user spends most time connected to host.

$ **ac -p**

sic	4.86
hipo	4.80
root	25.80
play	0.02

To get general info on *how much overall hours all existing users spend doing stuff on node*;

$ **ac** total 35.61

To know which days from the month users were most active:

$ **ac -d**
Feb 1 total 14.54
Feb 2 total 0.97
Feb 3 total 12.47
Feb 4 total 5.96
Today total 1.73