Utilities

Linux ulimit Command

1 year ago • by Sidratul Muntaha

"ulimit" is an interesting Linux shell command that can set or report the resource limit of the current user.

Of course, because of its nature, working with "ulimit" requires admin access (when changing value). Moreover, it'll only work on systems that allow control through the shell. Without further ado, let's get started with "ulimit"!

Location

Fire up the terminal and run the following command.
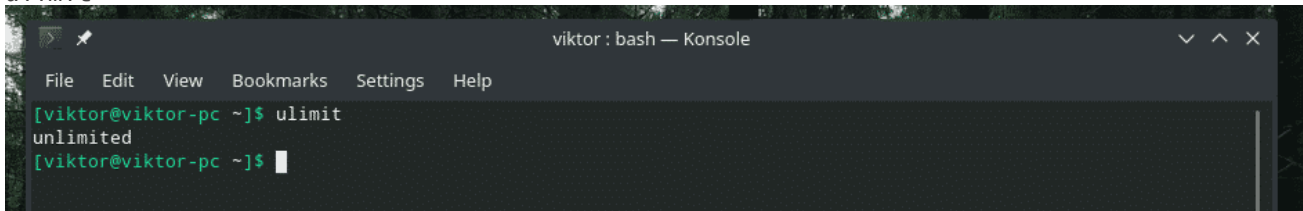
```
which ulimit
```

```
[viktor@viktor-pc ~]$ which limit
which: no limit in (/usr/local/sbin:/usr/local/bin:/usr/bin:/usr/bin:site_perl:/usr/bin/vendor_perl:/usr/bin/
core_perl:/var/lib/snapd/snap/bin)
[viktor@viktor-pc ~]$
```

This shows up an error, right? Why? It's because "ulimit" isn't a separate binary. It's embedded into the shell itself.
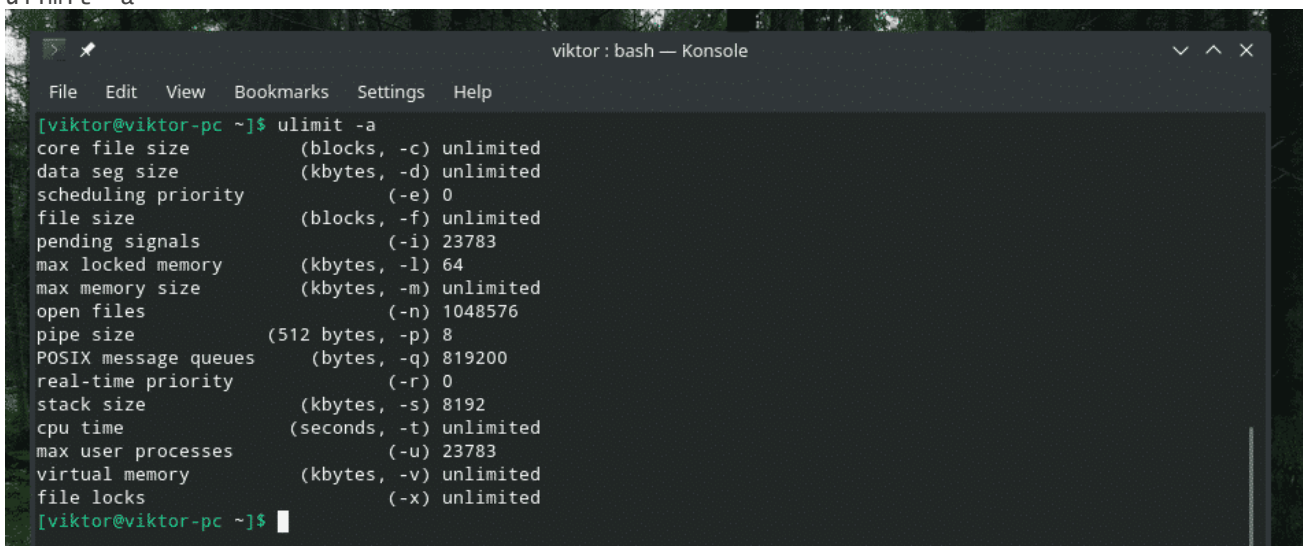
Basic usage

Just run the command by itself.

```
ulimit
```

```
                                    viktor : bash — Konsole               ∨ ∧ ✕
 File   Edit   View   Bookmarks   Settings   Help
 [viktor@viktor-pc ~]$ ulimit
 unlimited
 [viktor@viktor-pc ~]$
```

As the output suggests, the current user has "unlimited" amount of resources to be accessed. Essentially, this means the current user can consume all the resources the current system supports.

To get the report in details, add the "-a" flag at the end. This will print all the resource limits for the current user.

```
ulimit -a
```

```
                                    viktor : bash — Konsole               ∨ ∧ ✕
 File   Edit   View   Bookmarks   Settings   Help
 [viktor@viktor-pc ~]$ ulimit -a
 core file size          (blocks, -c) unlimited
 data seg size           (kbytes, -d) unlimited
 scheduling priority             (-e) 0
 file size               (blocks, -f) unlimited
 pending signals                 (-i) 23783
 max locked memory       (kbytes, -l) 64
 max memory size         (kbytes, -m) unlimited
 open files                      (-n) 1048576
 pipe size            (512 bytes, -p) 8
 POSIX message queues     (bytes, -q) 819200
 real-time priority              (-r) 0
 stack size              (kbytes, -s) 8192
 cpu time               (seconds, -t) unlimited
 max user processes              (-u) 23783
 virtual memory          (kbytes, -v) unlimited
 file locks                      (-x) unlimited
 [viktor@viktor-pc ~]$
```

Note: There are 2 types of resource limitation: "hard" and "soft". Hard resource limit defines the physical limit that the user can reach. The "soft" resource limit is manageable by the user. Its value can go up to the "hard" limit.

The system resources are defined in a configuration file located at "/etc/security/limits.conf". "ulimit", when called, will report these values.

```
cat /etc/security/limits.conf
```

```
viktor : bash — Konsole

File   Edit   View   Bookmarks   Settings   Help

[viktor@viktor-pc ~]$ cat /etc/security/limits.conf
# /etc/security/limits.conf
#
#Each line describes a limit for a user in the form:
#
#<domain>        <type>  <item>  <value>
#
#Where:
#<domain> can be:
#        - a user name
#        - a group name, with @group syntax
#        - the wildcard *, for default entry
#        - the wildcard %, can be also used with %group syntax,
#                for maxlogin limit
#
#<type> can have the two values:
#        - "soft" for enforcing the soft limits
#        - "hard" for enforcing hard limits
#
#<item> can be one of the following:
#        - core - limits the core file size (KB)
#        - data - max data size (KB)
#        - fsize - maximum filesize (KB)
#        - memlock - max locked-in-memory address space (KB)
#        - nofile - max number of open file descriptors
#        - rss - max resident set size (KB)
#        - stack - max stack size (KB)
#        - cpu - max CPU time (MIN)
#        - nproc - max number of processes
#        - as - address space limit (KB)
#        - maxlogins - max number of logins for this user
#        - maxsyslogins - max number of logins on the system
#        - priority - the priority to run user process with
```
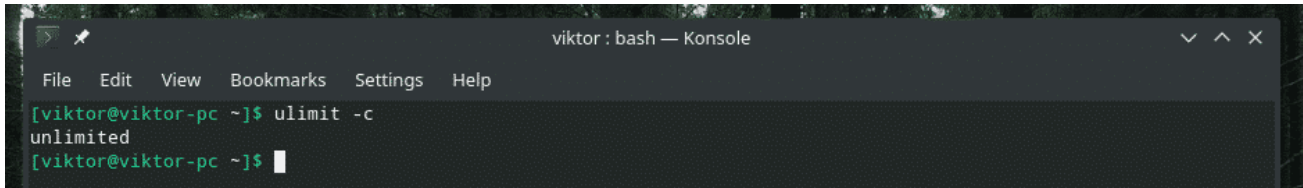
Checking different limits of a user

How about displaying the individual parameters? Then call "ulimit" with the respective parameters.

Note: Not all of these commands will work as not every system incorporates the features.
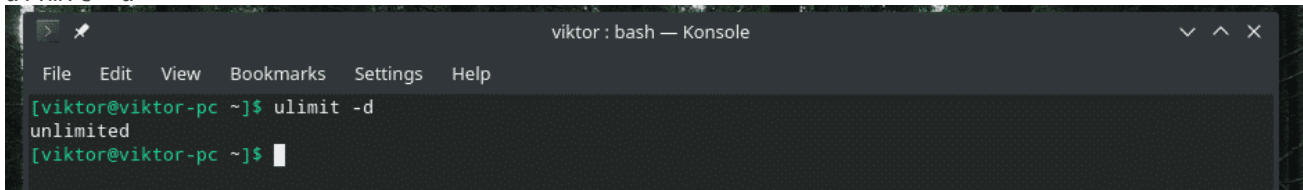
To check the value of max "core file size", use the flag "-c". The result will be shown in blocks.

```
ulimit -c
```

```
                                    viktor : bash — Konsole                    ⌄ ⌃ ✕

  File   Edit   View   Bookmarks   Settings   Help
[viktor@viktor-pc ~]$ ulimit -c
unlimited
[viktor@viktor-pc ~]$ ▮
```
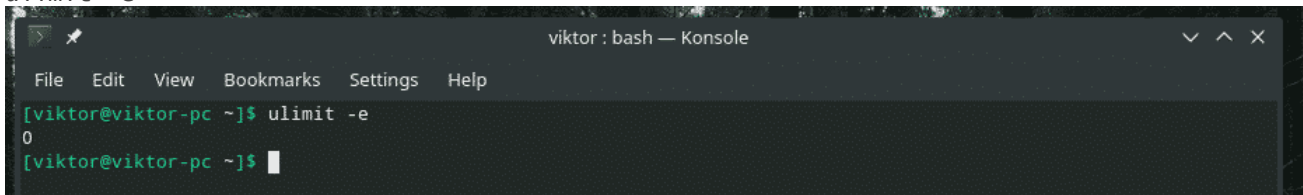
For max "data seg size" (in kilobytes), use the flag "-d".

```
ulimit -d
```

```
                                    viktor : bash — Konsole                    ⌄ ⌃ ✕

  File   Edit   View   Bookmarks   Settings   Help
[viktor@viktor-pc ~]$ ulimit -d
unlimited
[viktor@viktor-pc ~]$ ▮
```
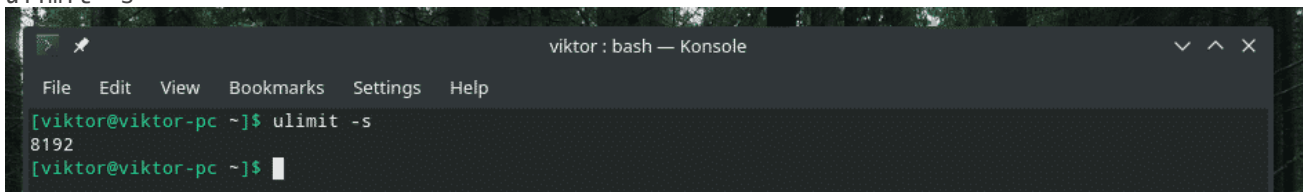
In scheduling, priority matters. Check out the max scheduling priority of the current user with the help of the "-e" flag.
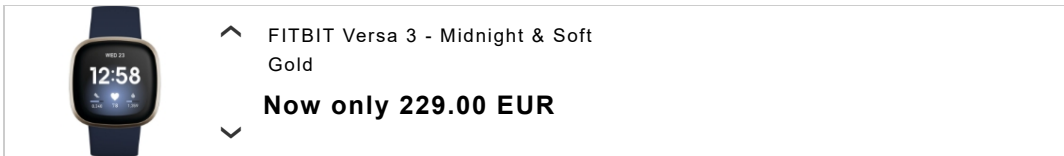
```
ulimit -e
```

```
                                    viktor : bash — Konsole                    ⌄ ⌃ ✕

  File   Edit   View   Bookmarks   Settings   Help
[viktor@viktor-pc ~]$ ulimit -e
0
[viktor@viktor-pc ~]$ ▮
```

To check the maximum stack size of the current user, use the flag "-s".

```
ulimit -s
```

```
                                    viktor : bash — Konsole                    ⌄ ⌃ ✕

  File   Edit   View   Bookmarks   Settings   Help
[viktor@viktor-pc ~]$ ulimit -s
8192
[viktor@viktor-pc ~]$ ▮
```

For the maximum number of user processes, use the flag "-u".

```
ulimit -u
```

For the maximum number of threads, use the flag "-T".

```
ulimit -T
```
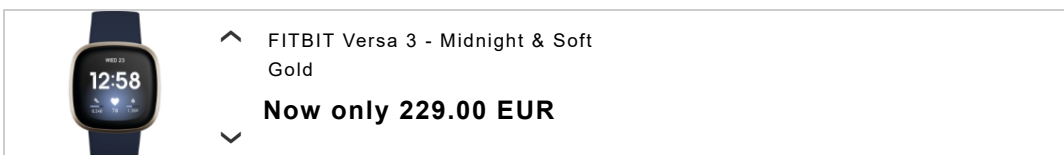For getting the size of virtual memory, use the following one.

```
ulimit -v
```

Let's check out the socket buffer size!

```
ulimit -b
```
This one is an interesting one. The following command will report the time each process is allowed to run for.
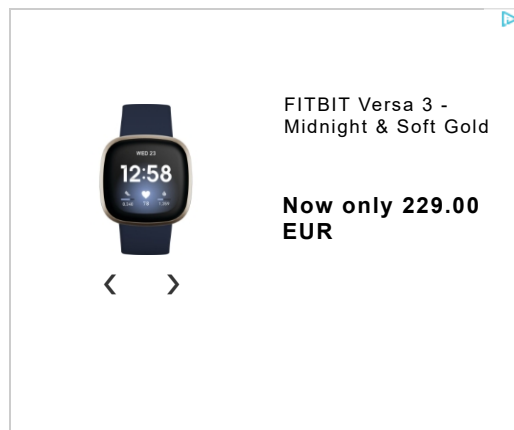
```
=ulimit -t
```

File descriptors are another interesting part of the Linux ecosystem. To check out how many file descriptors a process can have, run the following one.

```
ulimit -n
```

For the complete list of "ulimit" flags, check out the help page of "ulimit".

```
ulimit --help
```

Setting values

Up until now, we've seen how to check the values of the resource limits of the current system. Now, it's time to see how to manually modify these values.

Note: For changing the "hard" limit, it'll require admin access, i.e. "root" permission. Make sure

We mentioned the "limits.conf" file, right? It's the core that DEFINES all the limits that'll be applicable to the users.

```
sudo vim /etc/security/limits.conf
```
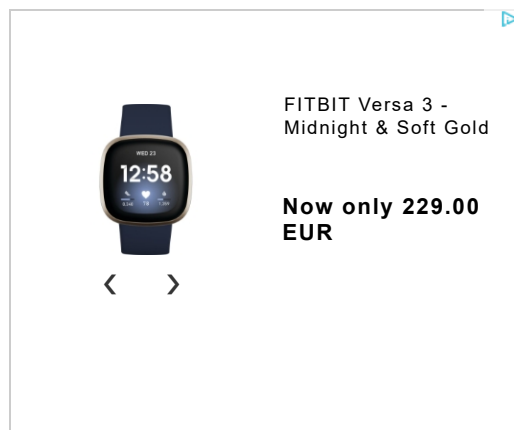
As shown in the file, each entry has to follow the following structure.

```
<domain> <type> <item> <value>
```

The "domain" part can be any of these values.

- A specific user
- A group
- Wildcard (* and %)

The "type" part allows the following values.

- "soft" (for implementing soft limits)
- "hard" (for implementing hard limits)

Next up, the "item" part. The list of available options is quite long! Here are some of the interesting ones.

- core: Core file size (in KB).
- data: Max data size (in KB)
- fsize: Max file size (in KB).
- memlock: Max locked-in-memory address space (in KB).
- nofile: Max number of file descriptors
- stack: Max size of the stack (in KB).
- cpu: Maximum CPU time (in MIN).
- maxlogins: Maximum number of logins for the current user/group
- priority: Set the priority of the user's processes
- rtprio: Max priority in real-time.

As of the "value" field, put an integer value. Note that the unit of each field is associated with
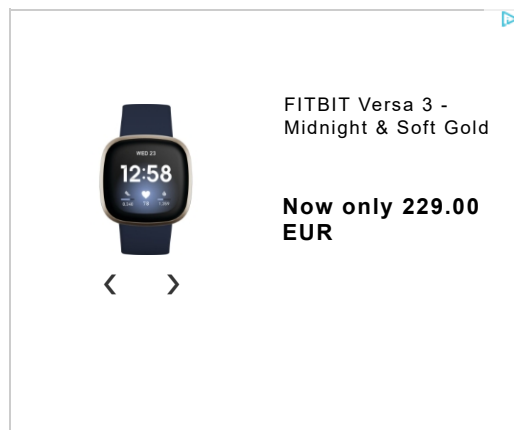
The changes will take into effect after exiting and re-logging into the associated user account.

For the in-depth info on the "limits.conf" configuration file, check out its man page.
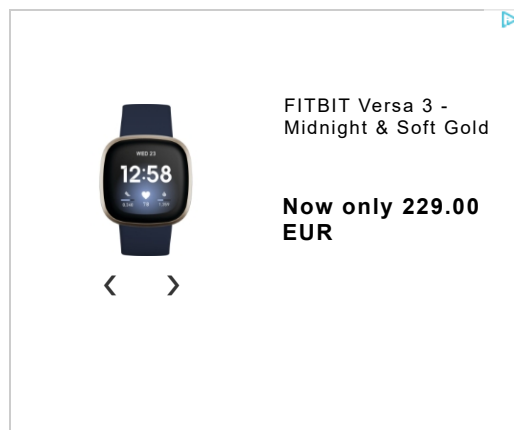
```
man limits.conf
```

Final thoughts

While the usage of the "ulimit" command isn't so complex, it plays an important role in determining the available system resources and ultimately, determining the performance. To know more about this command, check out its man and info pages.

```
ulimit --help
```

```
man ulimit
```

```
info ulimit
```

Enjoy!

ABOUT THE AUTHOR

Sidratul Muntaha

Student of CSE. I love Linux and playing with tech and gadgets. I use both
Ubuntu and Linux Mint.

View all posts

RELATED LINUX HINT POSTS

How to Use hexdump Command in Linux?

Zip and Unzip Commands in Linux

Unzip command on Linux

Echo Command in Linux

How to Check Your Linux Version

Downloading Files on Linux using the Command Line

What is /dev/null and How to Use It