



Linux Ubuntu 26.04 Manual Pages on command 'setpriv.1'

\$ man setpriv.1

SETPRIV(1) User Commands SETPRIV(1)

NAME

setpriv - run a program with different Linux privilege settings

SYNOPSIS

setpriv [options] program [arguments]

DESCRIPTION

Sets or queries various Linux privilege settings that are inherited across `execve(2)`.

In comparison to `su(1)` and `runuser(1)`, `setpriv` neither uses PAM, nor does it prompt for a password. It is a simple, non-set-user-ID wrapper around `execve(2)`, and can be used to drop privileges in the same way as `setuidgid(8)` from `daemontools`, `chpst(8)` from `runit`, or similar tools shipped by other service managers.

OPTIONS

`--clear-groups`

Clear supplementary groups.

`-d, --dump`

Dump the current privilege state. This option can be specified more than once to show extra, mostly useless, information. Incompatible with all other options.

`--groups group...`

Set supplementary groups. The argument is a comma-separated list of GIDs or names.

`--inh-caps (+|-)cap..., --ambient-caps (+|-)cap..., --bounding-set (+|-)cap...`

Set the inheritable capabilities, ambient capabilities or the capability bounding set.

See `capabilities(7)`. The argument is a comma-separated list of `+cap` and `-cap` entries,

which add or remove an entry respectively. `cap` can either be a human-readable name as

seen in capabilities(7) without the cap_ prefix or of the format cap_N, where N is the internal capability index used by Linux. +all and -all can be used to add or remove all caps.

The set of capabilities starts out as the current inheritable set for --inh-caps, the current ambient set for --ambient-caps and the current bounding set for --bounding-set.

Note the following restrictions (detailed in capabilities(7)) regarding modifications to these capability sets:

- ? A capability can be added to the inheritable set only if it is currently present in the bounding set.
- ? A capability can be added to the ambient set only if it is currently present in both the permitted and inheritable sets.
- ? Notwithstanding the syntax offered by setpriv, the kernel does not permit capabilities to be added to the bounding set.

If you drop a capability from the bounding set without also dropping it from the inheritable set, you are likely to become confused. Do not do that.

--keep-groups

Preserve supplementary groups. Only useful in conjunction with --rgid, --egid, or --regid.

--init-groups

Initialize supplementary groups using initgroups3. Only useful in conjunction with --ruid or --reuid.

--list-caps

List all known capabilities. This option must be specified alone.

--no-new-privs

Set the no_new_privs bit. With this bit set, execve(2) will not grant new privileges. For example, the set-user-ID and set-group-ID bits as well as file capabilities will be disabled. (Executing binaries with these bits set will still work, but they will not gain privileges. Certain LSMs, especially AppArmor, may result in failures to execute certain programs.) This bit is inherited by child processes and cannot be unset. See prctl(2) and Documentation/prctl/no_new_privs.txt in the Linux kernel source.

The no_new_privs bit is supported since Linux 3.5.

--rgid gid, --egid gid, --regid gid

Set the real, effective, or both GIDs. The gid argument can be given as a textual group

name.

For safety, you must specify one of `--clear-groups`, `--groups`, `--keep-groups`, or `--init-groups` if you set any primary gid.

`--ruid uid, --euid uid, --reuid uid`

Set the real, effective, or both UIDs. The uid argument can be given as a textual login name.

Setting a uid or gid does not change capabilities, although the `exec` call at the end might change capabilities. This means that, if you are root, you probably want to do something like:

```
setpriv --reuid=1000 --regid=1000 --inh-caps=-all
```

`--securebits (+|-)securebit...`

Set or clear securebits. The argument is a comma-separated list. The valid securebits are `noroot`, `noroot_locked`, `no_setuid_fixup`, `no_setuid_fixup_locked`, and `keep_caps_locked`. `keep_caps` is cleared by `execve(2)` and is therefore not allowed.

`--pdeathsig keep|clear|<signal>`

Keep, clear or set the parent death signal. Some LSMs, most notably SELinux and AppArmor, clear the signal when the process' credentials change. Using `--pdeathsig keep` will restore the parent death signal after changing credentials to remedy that situation.

`--ptracer pid|any|none`

When Yama's restricted ptrace mode is in effect (that is, when `/proc/sys/kernel/yama/ptrace_scope` is set to 1), allow being traced via `ptrace(2)` by the process with the specified PID, or any process, or no process. See `PR_SET_PTRACER(2const)`. (Note that this is not inherited by child processes, though it is preserved across `execve(2)`.) This option has no effect when Yama is not enabled or is in a mode other than restricted ptrace.

`--selinux-label label`

Request a particular SELinux transition (using a transition on `exec`, not `dyntrans`). This will fail and cause `setpriv` to abort if SELinux is not in use, and the transition may be ignored or cause `execve(2)` to fail at SELinux's whim. (In particular, this is unlikely to work in conjunction with `no_new_privs`.) This is similar to `runcon(1)`.

`--apparmor-profile profile`

Request a particular AppArmor profile (using a transition on `exec`). This will fail and

cause setpriv to abort if AppArmor is not in use, and the transition may be ignored or cause execve(2) to fail at AppArmor's whim.

`--landlock-access access`

Enable landlock restrictions for a specific set of system accesses. To allow specific subgroups of accesses use `--landlock-rule`.

Block all filesystem access:

```
setpriv --landlock-access fs
```

Block all file deletions and directory creations:

```
setpriv --landlock-access fs:remove-file,mkdir
```

For a complete set of supported access categories use `setpriv --help`.

`--landlock-rule rule`

Allow one specific access from the categories blocked by `--landlock-access`.

The syntax is as follows:

```
--landlock-rule $ruletype:$access:$rulearg
```

For example grant file read access to everything under `/boot`:

```
--landlock-rule path-beneath:read-file:/boot
```

`--seccomp-filter file`

Load raw BPF seccomp filter code from a file.

Filters can for example be created with `enosys`.

`--reset-env`

Clears all the environment variables except `TERM`; initializes the environment variables

`HOME`, `SHELL`, `USER`, `LOGNAME` according to the user's `passwd` entry; sets `PATH` to

`/usr/local/bin:/bin:/usr/bin` for a regular user and to

`/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin` for root.

The environment variable `PATH` may be different on systems where `/bin` and `/sbin` are

merged into `/usr`. The environment variable `SHELL` defaults to `/bin/sh` if none is given in

the user's `passwd` entry.

`-h, --help`

Display help text and exit.

`-V, --version`

Display version and exit.

NOTES

If applying any specified option fails, program will not be run and `setpriv` will return with

exit status 127.

Be careful with this tool ? it may have unexpected security consequences. For example, setting `no_new_privs` and then execing a program that is SELinux-confined (as this tool would do) may prevent the SELinux restrictions from taking effect.

EXAMPLES

If you're looking for behavior similar to `su(1)`/`runuser(1)`, or `sudo(8)` (without the `-g` option), try something like:

```
setpriv --reuid=1000 --regid=1000 --init-groups
```

If you want to mimic `daemontools`' `setuid(8)`, try:

```
setpriv --reuid=1000 --regid=1000 --clear-groups
```

AUTHORS

Andy Lutomirski <luto@amacapital.net>

SEE ALSO

`runuser(1)`, `su(1)`, `prctl(2)`, `capabilities(7)` `landlock(7)`

REPORTING BUGS

For bug reports, use the issue tracker <<https://github.com/util-linux/util-linux/issues>>.

AVAILABILITY

The `setpriv` command is part of the `util-linux` package which can be downloaded from Linux Kernel Archive <<https://www.kernel.org/pub/linux/utils/util-linux/>>.

util-linux 2.41.3

2025-12-15

SETPRIV(1)