



*Full credit is given to the above companies including the OS that this PDF file was generated!*

### ***Rocky Enterprise Linux 9.2 Manual Pages on command 'nsenter.1'***

***\$ man nsenter.1***

NSENTER(1) User Commands NSENTER(1)

NAME

nsenter - run program in different namespaces

SYNOPSIS

nsenter [options] [program [arguments]]

DESCRIPTION

The nsenter command executes program in the namespace(s) that are specified in the command-line options (described below). If program is not given, then "\${SHELL}" is run (default: /bin/sh).

Enterable namespaces are:

mount namespace

Mounting and unmounting filesystems will not affect the rest of the system, except for filesystems which are explicitly marked as shared (with mount --make-shared; see /proc/self/mountinfo for the shared flag). For further details, see mount\_namespaces(7) and the discussion of the CLONE\_NEWNS flag in clone(2).

UTS namespace

Setting hostname or domainname will not affect the rest of the

system. For further details, see `uts_namespaces(7)`.

#### IPC namespace

The process will have an independent namespace for POSIX message queues as well as System V message queues, semaphore sets and shared memory segments. For further details, see `ipc_namespaces(7)`.

#### network namespace

The process will have independent IPv4 and IPv6 stacks, IP routing tables, firewall rules, the `/proc/net` and `/sys/class/net` directory trees, sockets, etc. For further details, see `network_namespaces(7)`.

#### PID namespace

Children will have a set of PID to process mappings separate from the `nsenter` process. `nsenter` will fork by default if changing the PID namespace, so that the new program and its children share the same PID namespace and are visible to each other. If `--no-fork` is used, the new program will be executed without forking. For further details, see `pid_namespaces(7)`.

#### user namespace

The process will have a distinct set of UIDs, GIDs and capabilities. For further details, see `user_namespaces(7)`.

#### cgroup namespace

The process will have a virtualized view of `/proc/self/cgroup`, and new cgroup mounts will be rooted at the namespace cgroup root. For further details, see `cgroup_namespaces(7)`.

#### time namespace

The process can have a distinct view of `CLOCK_MONOTONIC` and/or `CLOCK_BOOTTIME` which can be changed using `/proc/self/timens_offsets`. For further details, see `time_namespaces(7)`.

## OPTIONS

Various of the options below that relate to namespaces take an optional file argument. This should be one of the `/proc/[pid]/ns/*` files described in `namespaces(7)`, or the pathname of a bind mount that was

created on one of those files.

`-a, --all`

Enter all namespaces of the target process by the default `/proc/[pid]/ns/*` namespace paths. The default paths to the target process namespaces may be overwritten by namespace specific options (e.g., `--all --mount=[path]`).

The user namespace will be ignored if the same as the caller's current user namespace. It prevents a caller that has dropped capabilities from regaining those capabilities via a call to `setns()`. See `setns(2)` for more details.

`-t, --target PID`

Specify a target process to get contexts from. The paths to the contexts specified by pid are:

`/proc/pid/ns/mnt`

the mount namespace

`/proc/pid/ns/uts`

the UTS namespace

`/proc/pid/ns/ipc`

the IPC namespace

`/proc/pid/ns/net`

the network namespace

`/proc/pid/ns/pid`

the PID namespace

`/proc/pid/ns/user`

the user namespace

`/proc/pid/ns/cgroup`

the cgroup namespace

`/proc/pid/ns/time`

the time namespace

`/proc/pid/root`

the root directory

`/proc/pid/cwd`

the working directory respectively

`-m, --mount[=file]`

Enter the mount namespace. If no file is specified, enter the mount namespace of the target process. If file is specified, enter the mount namespace specified by file.

`-u, --uts[=file]`

Enter the UTS namespace. If no file is specified, enter the UTS namespace of the target process. If file is specified, enter the UTS namespace specified by file.

`-i, --ipc[=file]`

Enter the IPC namespace. If no file is specified, enter the IPC namespace of the target process. If file is specified, enter the IPC namespace specified by file.

`-n, --net[=file]`

Enter the network namespace. If no file is specified, enter the network namespace of the target process. If file is specified, enter the network namespace specified by file.

`-p, --pid[=file]`

Enter the PID namespace. If no file is specified, enter the PID namespace of the target process. If file is specified, enter the PID namespace specified by file.

`-U, --user[=file]`

Enter the user namespace. If no file is specified, enter the user namespace of the target process. If file is specified, enter the user namespace specified by file. See also the `--setuid` and `--setgid` options.

`-C, --cgroup[=file]`

Enter the cgroup namespace. If no file is specified, enter the cgroup namespace of the target process. If file is specified, enter the cgroup namespace specified by file.

`-T, --time[=file]`

Enter the time namespace. If no file is specified, enter the time namespace of the target process. If file is specified, enter the time namespace specified by file.

`-G, --setgid gid`

Set the group ID which will be used in the entered namespace and drop supplementary groups. nsender always sets GID for user namespaces, the default is 0.

`-S, --setuid uid`

Set the user ID which will be used in the entered namespace. nsender always sets UID for user namespaces, the default is 0.

`--preserve-credentials`

Don't modify UID and GID when enter user namespace. The default is to drops supplementary groups and sets GID and UID to 0.

`-r, --root[=directory]`

Set the root directory. If no directory is specified, set the root directory to the root directory of the target process. If directory is specified, set the root directory to the specified directory.

`-w, --wd[=directory]`

Set the working directory. If no directory is specified, set the working directory to the working directory of the target process. If directory is specified, set the working directory to the specified directory.

`-F, --no-fork`

Do not fork before exec'ing the specified program. By default, when entering a PID namespace, nsender calls fork before calling exec so that any children will also be in the newly entered PID namespace.

`-Z, --follow-context`

Set the SELinux security context used for executing a new process according to already running process specified by `--target PID`. (The util-linux has to be compiled with SELinux support otherwise the option is unavailable.)

`-V, --version`

Display version information and exit.

`-h, --help`

Display help text and exit.

Eric Biederman <biederm@xmission.com>, Karel Zak <kzak@redhat.com>

## SEE ALSO

clone(2), setns(2), namespaces(7)

## REPORTING BUGS

For bug reports, use the issue tracker at

<https://github.com/karelzak/util-linux/issues>.

## AVAILABILITY

The nsenter 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 2.37.4

2022-02-14

NSENTER(1)