Rocky Enterprise Linux 9.2 Manual Pages on command 'libnss_systemd.so.2.8'

$ man libnss_systemd.so.2.8

NSS-SYSTEMD(8) nss-systemd NSS-SYSTEMD(8)

NAME

nss-systemd, libnss_systemd.so.2 - UNIX user and group name resolution for user/group lookup via Varlink

SYNOPSIS

libnss_systemd.so.2

DESCRIPTION

nss-systemd is a plug-in module for the GNU Name Service Switch (NSS) functionality of the GNU C Library (glibc), providing UNIX user and group name resolution for services implementing the User/Group Record Lookup API via Varlink[1], such as the system and service manager systemd(1) (for its DynamicUser= feature, see systemd.exec(5) for details), systemd-homed.service(8), or systemd-machined.service(8). This module also ensures that the root and nobody users and groups (i.e. the users/groups with the UIDs/GIDs 0 and 65534) remain resolvable at all times, even if they aren't listed in /etc/passwd or /etc/group, or if these files are missing.

This module preferably utilizes systemd-userdbd.service(8) for

resolving users and groups, but also works without the service running.

To activate the NSS module, add "systemd" to the lines starting with "passwd:", "group:", "shadow:" and "gshadow:" in /etc/nsswitch.conf. It is recommended to place "systemd" after the "files" or "compat" entry of the /etc/nsswitch.conf lines so that /etc/passwd, /etc/group, /etc/shadow and /etc/gshadow based mappings take precedence.

STATIC DROP-IN JSON USER/GROUP RECORDS

Besides user/group records acquired via the aforementioned Varlink IPC interfaces and the synthesized root and nobody accounts, this module also makes user and group accounts available to the system that are defined in static drop-in files in the /etc/userdb/, /run/userdb/, /run/host/userdb/ and /usr/lib/userdb/ directories.

This is a simple mechanism to provide static user and group records via JSON drop-in files. Such user records should be defined in the format described by the JSON User Records[2] specification and be placed in one of the aforementioned directories under a file name composed of the user name suffixed with .user, with a world-readable access mode. A symlink named after the user record's UID formatted in decimal and suffixed with .user pointing to the primary record file should be created as well, in order to allow both lookups by username and by UID.

Privileged user record data (e.g. hashed UNIX passwords) may optionally be provided as well, in a pair of separate companion files with the .user-privileged suffix. The data should be stored in a regular file named after the user name, suffixed with .user-privileged, and a symlink pointing to it, named after the used numeric UID formatted in decimal with the same suffix. These companion files should not be readable to anyone but root. Example:

```
-rw-r--r--. 1 root root  723 May 10 foobar.user
-rw-------. 1 root root  123 May 10 foobar.user-privileged
lrwxrwxrwx. 1 root root   19 May 10 4711.user -> foobar.user
lrwxrwxrwx. 1 root root   19 May 10 4711.user-privileged -> foobar.user-privileged
```

Similarly, group records following the format described in JSON Group Record[3] may be defined, using the file suffixes .group and

.group-privileged.

The primary user/group record files (i.e. those with the .user and .group suffixes) should not contain the "privileged" section as described in the specifications. The privileged user/group record files (i.e. those with the .user-privileged and .group-privileged suffixes) should contain this section, exclusively.

Note that static user/group records generally do not override conflicting records in /etc/passwd or /etc/group or other account databases. In fact, before dropping in these files a reasonable level of care should be taken to avoid user/group name and UID/GID conflicts.

CONFIGURATION IN /ETC/NSSWITCH.CONF

Here is an example /etc/nsswitch.conf file that enables nss-systemd correctly:

```
passwd:         compat systemd
group:          compat [SUCCESS=merge] systemd
shadow:         compat systemd
gshadow:        files systemd
hosts:          mymachines resolve [!UNAVAIL=return] files myhostname dns
networks:       files
protocols:      db files
services:       db files
ethers:         db files
rpc:            db files
netgroup:       nis
```

EXAMPLE: MAPPINGS PROVIDED BY SYSTEMD-MACHINED.SERVICE

The container "rawhide" is spawned using systemd-nspawn(1):

```
# systemd-nspawn -M rawhide --boot --network-veth --private-users=pick
Spawning container rawhide on /var/lib/machines/rawhide.
Selected user namespace base 20119552 and range 65536.
...
$ machinectl --max-addresses=3
MACHINE CLASS     SERVICE        OS     VERSION ADDRESSES
rawhide container systemd-nspawn fedora 30      169.254.40.164 fe80::94aa:3aff:fe7b:d4b9
```

```
$ getent passwd vu-rawhide-0 vu-rawhide-81
vu-rawhide-0:*:20119552:65534:vu-rawhide-0:/:/usr/sbin/nologin
vu-rawhide-81:*:20119633:65534:vu-rawhide-81:/:/usr/sbin/nologin
$ getent group vg-rawhide-0 vg-rawhide-81
vg-rawhide-0:*:20119552:
vg-rawhide-81:*:20119633:
$ ps -o user:15,pid,tty,command -e|grep '^vu-rawhide'
vu-rawhide-0      692 ?        /usr/lib/systemd/systemd
vu-rawhide-0      731 ?        /usr/lib/systemd/systemd-journald
vu-rawhide-192    734 ?        /usr/lib/systemd/systemd-networkd
vu-rawhide-193    738 ?        /usr/lib/systemd/systemd-resolved
vu-rawhide-0      742 ?        /usr/lib/systemd/systemd-logind
        vu-rawhide-81         744  ?              /usr/bin/dbus-daemon  --system  --address=systemd:  --nofork  --nopidfile
--systemd-activation --syslog-only
vu-rawhide-0      746 ?        /usr/sbin/sshd -D ...
vu-rawhide-0      752 ?        /usr/lib/systemd/systemd --user
vu-rawhide-0      753 ?        (sd-pam)
vu-rawhide-0     1628 ?        login -- zbyszek
vu-rawhide-1000  1630 ?        /usr/lib/systemd/systemd --user
vu-rawhide-1000  1631 ?        (sd-pam)
vu-rawhide-1000  1637 pts/8    -zsh
```

SEE ALSO

systemd(1), systemd.exec(5), nss-resolve(8), nss-myhostname(8), nss-mymachines(8), systemd-userdbd.service(8), systemd-homed.service(8), systemd-machined.service(8), nsswitch.conf(5), getent(1)

NOTES

1. User/Group Record Lookup API via Varlink

   https://systemd.io/USER_GROUP_API

2. JSON User Records

   https://systemd.io/USER_RECORD

3. JSON Group Record

   https://systemd.io/GROUP_RECORD