NAME

setnetgrent, endnetgrent, getnetgrent, getnetgrent_r, innetgr − handle network group entries

SYNOPSIS

#include <netdb.h>

int setnetgrent(const char ***_netgroup_);**

void endnetgrent(void);

int getnetgrent(char ****_host_, char ****_user_**, char ****_domain_**);**

int getnetgrent_r(char ****_host_, char ****_user_**,**
char ****_domain_, char ***_buf_**, size_t** _buflen_**);**

int innetgr(const char ***_netgroup_, const char ***_host_**,**
const char ***_user_, const char ***_domain_**);**

Feature Test Macro Requirements for glibc (see **feature_test_macros**(7)):

setnetgrent(), **endnetgrent**(), **getnetgrent**(), **getnetgrent_r**(), **innetgr**():
Since glibc 2.19:
_DEFAULT_SOURCE
Glibc 2.19 and earlier:
_BSD_SOURCE || _SVID_SOURCE

DESCRIPTION

The *netgroup* is a SunOS invention. A netgroup database is a list of string triples (*hostname*, *username*, *domainname*) or other netgroup names. Any of the elements in a triple can be empty, which means that anything matches. The functions described here allow access to the netgroup databases. The file */etc/nsswitch.conf* defines what database is searched.

The **setnetgrent**() call defines the netgroup that will be searched by subsequent **getnetgrent**() calls. The **getnetgrent**() function retrieves the next netgroup entry, and returns pointers in *host*, *user*, *domain*. A null pointer means that the corresponding entry matches any string. The pointers are valid only as long as there is no call to other netgroup-related functions. To avoid this problem you can use the GNU function **getnetgrent_r**() that stores the strings in the supplied buffer. To free all allocated buffers use **endnetgrent**().

In most cases you want to check only if the triplet (*hostname*, *username*, *domainname*) is a member of a netgroup. The function **innetgr**() can be used for this without calling the above three functions. Again, a null pointer is a wildcard and matches any string. The function is thread-safe.

RETURN VALUE

These functions return 1 on success and 0 for failure.

FILES

/etc/netgroup
/etc/nsswitch.conf

ATTRIBUTES

For an explanation of the terms used in this section, see **attributes**(7).

Interface	Attribute	Value
setnetgrent(), **getnetgrent_r**(), **innetgr**()	Thread safety	MT-Unsafe race:netgrent locale
endnetgrent()	Thread safety	MT-Unsafe race:netgrent
getnetgrent()	Thread safety	MT-Unsafe race:netgrent race:netgrentbuf locale

In the above table, *netgrent* in *race:netgrent* signifies that if any of the functions **setnetgrent**(), **getnetgrent_r**(), **innetgr**(), **getnetgrent**(), or **endnetgrent**() are used in parallel in different threads of a program, then data races could occur.

CONFORMING TO

These functions are not in POSIX.1, but **setnetgrent**(), **endnetgrent**(), **getnetgrent**(), and **innetgr**() are available on most UNIX systems. **getnetgrent_r**() is not widely available on other systems.

NOTES

In the BSD implementation, **setnetgrent**() returns void.

SEE ALSO

sethostent(3), **setprotoent**(3), **setservent**(3)

COLOPHON

This page is part of release 5.02 of the Linux *man-pages* project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man−pages/.