



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

Linux Ubuntu 26.04 Manual Pages on command 'getgrnam.3'

\$ man getgrnam.3

getgrnam(3) Library Functions Manual getgrnam(3)

NAME

getgrnam, getgrnam_r, getgrgid, getgrgid_r - get group file entry

LIBRARY

Standard C library (libc, -lc)

SYNOPSIS

```
#include <sys/types.h>
```

```
#include <grp.h>
```

```
struct group *getgrnam(const char *name);
```

```
struct group *getgrgid(gid_t gid);
```

```
int getgrnam_r(size_t size;
```

```
    const char *restrict name, struct group *restrict grp,
```

```
    char buf[restrict size], size_t size,
```

```
    struct group **restrict result);
```

```
int getgrgid_r(size_t size;
```

```
    gid_t gid, struct group *restrict grp,
```

```
    char buf[restrict size], size_t size,
```

```
    struct group **restrict result);
```

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

```
getgrnam_r(), getgrgid_r():
```

```
    _POSIX_C_SOURCE
```

```
    || /* glibc <= 2.19: */ _BSD_SOURCE || _SVID_SOURCE
```

DESCRIPTION

The `getgrnam()` function returns a pointer to a structure containing the broken-out fields of the record in the group database (e.g., the local group file `/etc/group`, NIS, and LDAP) that matches the group name `name`.

The `getgrgid()` function returns a pointer to a structure containing the broken-out fields of the record in the group database that matches the group ID `gid`.

The group structure is defined in `<grp.h>` as follows:

```
struct group {
    char *gr_name;    /* group name */
    char *gr_passwd;  /* group password */
    gid_t gr_gid;     /* group ID */
    char **gr_mem;    /* NULL-terminated array of pointers
                        to names of group members */
};
```

For more information about the fields of this structure, see `group(5)`.

The `getgrnam_r()` and `getgrgid_r()` functions obtain the same information as `getgrnam()` and `getgrgid()`, but store the retrieved group structure in the space pointed to by `grp`. The string fields pointed to by the members of the group structure are stored in the buffer `buf` of size `size`. A pointer to the result (in case of success) or `NULL` (in case no entry was found or an error occurred) is stored in `*result`.

The call

```
sysconf(_SC_GETGR_R_SIZE_MAX)
```

returns either `-1`, without changing `errno`, or an initial suggested size for `buf`. (If this size is too small, the call fails with `ERANGE`, in which case the caller can retry with a larger buffer.)

RETURN VALUE

The `getgrnam()` and `getgrgid()` functions return a pointer to a group structure, or `NULL` if the matching entry is not found or an error occurs. If an error occurs, `errno` is set to indicate the error. If one wants to check `errno` after the call, it should be set to zero before the call.

The return value may point to a static area, and may be overwritten by subsequent calls to `getgrent(3)`, `getgrgid()`, or `getgrnam()`. (Do not pass the returned pointer to `free(3)`.)

On success, `getgrnam_r()` and `getgrgid_r()` return zero, and set `*result` to `grp`. If no matching group record was found, these functions return 0 and store `NULL` in `*result`. In case of

ERRORS

The given name or gid was not found.

EIO I/O error.

EMFILE The per-process limit on the number of open file descriptors has been reached.

ENFILE The system-wide limit on the total number of open files has been reached.

ENOMEM Insufficient memory to allocate group structure.

ERANGE Insufficient buffer space supplied.

FILES

/etc/group

local group database file

ATTRIBUTES

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

[illegible]

?	Interface	?	Attribute	?	Value	?

[illegible]

? getgrnam() ? Thread safety ? MT-Unsafe race:grnam locale ?

[illegible]

? getgrgid() ? Thread safety ? MT-Unsafe race:grgid locale ?

[illegible]

? getgrnam r(), ? Thread safety ? MT-Safe locale ?

? getrgid r() ? ? ?

[illegible]

VERSIONS

The formulation given above under "RETURN VALUE" is from POSIX.1. It does not call "not found" an error, hence does not specify what value `errno` might have in this situation. But that makes it impossible to recognize errors. One might argue that according to POSIX `errno` should be left unchanged if an entry is not found. Experiments on various UNIX-like systems show that lots of different values occur in this situation: 0, `ENOENT`, `EBADF`, `ESRCH`, `EWOULD?` `BLOCK`, `EPERM`, and probably others.

STANDARDS

POSIX.1-2008.

HISTORY

POSIX.1-2001, SVr4, 4.3BSD.

SEE ALSO

[endgrent\(3\)](#), [fgetgrent\(3\)](#), [getgrent\(3\)](#), [getpwnam\(3\)](#), [setgrent\(3\)](#), [group\(5\)](#)

Linux man-pages 6.17

2026-02-08

[getgrnam\(3\)](#)