grp — Unix Group Database

Purpose: Read group data from Unix group database.

The `grp` module can be used to read information about Unix groups from the group database (usually `/etc/group`). The read-only interface returns tuple-like objects with named attributes for the standard fields of a group record.

Index	Attribute	Meaning
0	gr_name	Name
1	gr_passwd	Password, if any (encrypted)
2	gr_gid	Numerical id (integer)
3	gr_mem	Names of group members

The name and password values are both strings, the GID is an integer, and the members are reported as a list of strings.

Querying All Groups

This example prints a report of all of the “real” groups on a system, including their members (where “real” is defined as having a name not starting with “_”). To load the entire password database, use `getgrall()`.

```
# grp_getgrall.py

import grp
import textwrap

# Load all of the user data, sorted by username
all_groups = grp.getgrall()
interesting_groups = {
    g.gr_name: g
    for g in all_groups
    if not g.gr_name.startswith('_')
}
print(len(interesting_groups.keys()))

# Find the longest length for a few fields
name_length = max(len(k) for k in interesting_groups) + 1
gid_length = max(len(str(u.gr_gid))
                 for u in interesting_groups.values()) + 1

# Set the members field width to avoid table columns
# wrapping
members_width = 19

# Print report headers
fmt = ' '.join(['{:<{name_length}}',
                '{:{gid_length}}',
                '{:<{members_width}}',
                ])
print(fmt.format('Name',
                 'GID',
                 'Members',
                 name_length=name_length,
                 gid_length=gid_length,
                 members_width=members_width))
print('-' * name_length,
      '-' * gid_length,
      '-' * members_width)

# Print the data
prefix = ' ' * (name_length + gid_length + 2)
```

```
for name, g in sorted(interesting_groups.items()):
    # Format members to start in the column on the same line but
    # wrap as needed with an indent sufficient to put the
    # subsequent lines in the members column. The two indent
    # prefixes need to be the same to compute the wrap properly,
    # but the first should not be printed so strip it.
    members = textwrap.fill(
        ', '.join(g.gr_mem),
        initial_indent=prefix,
        subsequent_indent=prefix,
        width=members_width + len(prefix),
    ).strip()
    print(fmt.format(g.gr_name,
                     g.gr_gid,
                     members,
                     name_length=name_length,
                     gid_length=gid_length,
                     members_width=members_width))
```

The return value is a list with an undefined order, so it needs to be sorted before printing the report.

```
$ python3 grp_getgrall.py

34
Name                            GID         Members
------------------------------ ----------- ------------------
accessibility                           90
admin                                   80 root
authedusers                             50
bin                                      7
certusers                               29 root, _jabber,
                                            _postfix, _cyrus,
                                            _calendar, _dovecot
com.apple.access_disabled              396
com.apple.access_ftp                   395
com.apple.access_screensharing         398
com.apple.access_sessionkey            397
com.apple.access_ssh                   399
com.apple.sharepoint.group.1           701 dhellmann
consoleusers                            53
daemon                                   1 root
dialer                                  68
everyone                                12
group                                   16
interactusers                           51
kmem                                     2 root
localaccounts                           61
mail                                     6 _teamsserver
netaccounts                             62
netusers                                52
network                                 69
nobody                          4294967294
nogroup                                 -1
operator                                 5 root
owner                                   10
procmod                                  9 root
procview                                 8 root
staff                                   20 root
sys                                      3 root
tty                                      4 root
utmp                                    45
wheel                                    0 root
```

Group Memberships for a User

Another common task might be to print a list of all the groups for a given user:

```
# grp_groups_for_user.py

import grp
```

```
username = 'dhellmann'
group_names = set(
    g.gr_name
    for g in grp.getgrall()
    if username in g.gr_mem
)
print(username, 'belongs to:', ', '.join(sorted(group_names)))
```

The set of unique group names is sorted before they are printed.

```
$ python3 grp_groups_for_user.py

dhellmann belongs to: _appserveradm, _appserverusr, _lpadmin, ad
min, com.apple.sharepoint.group.1
```

Finding a Group By Name

As with pwd, it is also possible to query for information about a specific group, either by name or numeric id.

```
# grp_getgrnam.py
```

```
import grp

name = 'admin'
info = grp.getgrnam(name)
print('Name    :', info.gr_name)
print('GID     :', info.gr_gid)
print('Password:', info.gr_passwd)
print('Members :', ', '.join(info.gr_mem))
```

The `admin` group has two members:

```
$ python3 grp_getgrnam.py

Name    : admin
GID     : 80
Password: *
Members : root, dhellmann
```

Finding a Group by ID

To identify the group running the current process, combine `getgrgid()` with `os.getgid()`.

```
# grp_getgrgid_process.py

import grp
import os

gid = os.getgid()
group_info = grp.getgrgid(gid)
print('Currently running with GID={} name={}'.format(
    gid, group_info.gr_name))
```

```
$ python3 grp_getgrgid_process.py

Currently running with GID=20 name=staff
```

And to get the group name based on the permissions on a file, look up the group returned by `os.stat()`.

```
# grp_getgrgid_fileowner.py

import grp
import os

filename = 'grp_getgrgid_fileowner.py'
```

```
stat_info = os.stat(filename)
owner = grp.getgrgid(stat_info.st_gid).gr_name

print('{} is owned by {} ({})'.format(
    filename, owner, stat_info.st_gid))
```

The file status record includes ownership and permission data for the file or directory.

```
$ python3 grp_getgrgid_fileowner.py

grp_getgrgid_fileowner.py is owned by staff (20)
```

See also

- Standard library documentation for grp
- pwd – Read user data from the password database.
- spwd – Read user data from the shadow password database.
- os – Operating system interfaces.

pwd — Unix Password Database

Porting Notes

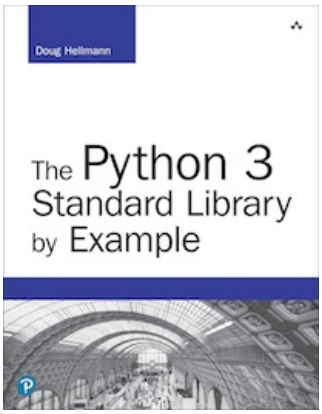
Quick Links

Querying All Groups
Group Memberships for a User
Finding a Group By Name
Finding a Group by ID

This page was last updated 2016-12-18.

Navigation

pwd — Unix Password Database
Porting Notes



Get the book

The output from all the example programs from PyMOTW-3 has been generated with Python 3.7.1, unless otherwise noted. Some of the features described here may not be available in earlier versions of Python.

Looking for examples for Python 2?

This Site

Module Index
Index

Other Writing

- Blog
- The Python Standard Library By Example