

Groups

Linux provides a facility for separating sets of related users into groups. Each user is associated with a primary group ID (GID), which is associated with a name. The group name and GID can be used interchangeably. In addition, users can also be associated with one or more secondary groups. This flexibility means that while a user might have a primary group membership based on their employment or organizational status (for example, staff or managers), they can actively share data and system privileges with other groups based on their workgroup needs (for example, sales, engineer).

Group Characteristics

Information about groups in Linux is stored in the groups database (*/etc/group*), but the local groups database may also be supplemented by the NIS/NIS+ or LDAP databases. Let's examine a typical set of groups:

```
# cat /etc/group
root::0:root
other::1:
bin::2:root,bin,daemon
sys::3:root,bin,sys,adm
adm::4:root,adm,daemon
uucp::5:root,uucp
mail::6:root
tty::7:root,tty,adm
lp::8:root,lp,adm
nuucp::9:root,nuucp
staff::10:paul,maya,brad,natashia
postgres:a.mBzQnr1ei2D.:100:postgres, paul
daemon::12:root,daemon
sysadmin::14:
nobody::60001:
noaccess::60002:
nogroup::65534:
```

We can see that the lower group numbers are associated with all of the system functions and accounts, such as the bin group, which has the members root, bin, and daemon, and the sys group, which has the members root, bin, sys, and adm. Higher-numbered groups, such as staff, contain several different users, such as paul, maya, brad, and natashia. Notice also that paul has a secondary group membership in the postgres group, giving him database access privileges. A group password can also be set for each group, although most groups don't use this facility. In this group database, we can see that the postgres group is the only group that has an encrypted password (a.mBzQnr1ei2D.).

You can obtain a list of all groups that a user belongs to by using the groups command. For example, to view all of the groups that the root users belongs to, we use the command

```
# groups root
other root bin sys adm uucp mail tty lp nuucp daemon
```

You can also see the converse, who belongs to a particular group using the command:

```
# getent group groupname
```

For example:

```
# getent group root
root::0:root
```

Adding Groups

To add a new group to the system, you may either manually edit the */etc/group* file or use the `groupadd` command, which has the following syntax:

```
/usr/sbin/groupadd -g gid group_name
```

Thus, to add a group called `managers` to the system, with a GID of 500, we would use the command

```
# groupadd -g 500 managers
```

We would then be able to verify the new group's existence by searching the groups database:

```
# grep management /etc/group
managers::500:
```

Caution The `groupadd` command will fail if the GID that you specify has already been allocated to an existing group, unless you use the `-o` option, or if the `group_name` is greater than eight characters.

Managing Groups

If you want to change your group from the primary to the secondary during an interactive session, to ensure that all of the files that you create are associated with the correct GID, you need to use the `newgrp` command. For example, the root user has the following primary group membership:

```
# id
uid=0(root) gid=0(root)
```

However, if the root user wishes to act as a member of another group, such as `sys`, the following command would have to be used:

```
# newgrp sys
```

The effective GID would then change to `sys`:

```
# id
uid=0(root) gid=3(sys)
```

Any operations that the root user performs after using newgrp, such as creating files, will be associated with the GID of 3 (sys) rather than 0 (root). For example, if we created a new file with the primary group, the group associated with the new file would be GID 0:

```
# touch root.txt
# ls -l root.txt
-rw-r--r--  1 root  root  0 Oct 12 11:17 root.txt
```

However, if the root user then changes groups to sys and creates a new file, then the group associated with the file will be sys rather than root:

```
# newgrp sys
# touch sys.txt
# ls -l sys.txt
-rw-r--r--  1 root  sys  0 Oct 12 11:18 sys.txt
```