umask

On Linux and Unix operating systems, all new files are created with a default set of permissions. The `umask` utility allows you to view or to set the file mode creation mask, which determines the permissions bits for newly created files or directories.

It is used by mkdir, touch, tee , and other commands that create new files and directories.

Linux Permissions

In Linux, each file is associated with an owner and a group and assigned with permission access rights for three different classes of users:

- The file owner.
- The group members.
- Everyone else.

There are three permissions types that apply to each class:

- The read permission.
- The write permission.
- The execute permission.

This concept allows you to specify which users are allowed to read the file, write to the file, or execute the file.

To view the file permissions, use the `ls` command:

```
ls -l dirname
drwxr-xr-x 12 linuxize users 4.0K Apr  8 20:51 dirname
|[-][-][-]    [------] [---]
| |  |  |        |       |
| |  |  |        |       +-----------> Group
| |  |  |        +-------------------> Owner
| |  |  +----------------------------> Others Permissions
| |  +-------------------------------> Group Permissions
| +----------------------------------> Owner Permissions
+------------------------------------> File Type
```

The first character represents the file type which can be a regular file (`-`), a directory (`d`), a symbolic link (`l`), or any other special type of file.

The next nine characters represent the permissions, three sets of three characters each. The first sets show the owner permissions, the second one group permissions, and the last set shows everybody else permissions.

Character `r` with an octal value of `4` stands for read, `w` with an octal value of `2` for write, `x` with an octal value of `1` for execute permission, and (`-`) with an octal value of `0` for no permissions.

In the example above (`rwxr-xr-x`) means that the owner has read, write and execute permissions (`rwx`), the group and others have read and execute permissions.

If we represent the file permissions using a numeric notation, we will come up to the number `755`:

- Owner: `rwx` = `4+2+1 = 7`
- Group: `r-x` = `4+0+1 = 5`
- Other: `r-x` = `4+0+1 = 5`

When represented in numeric notation, permissions can have three or four octal digits (0-7). The first digit represents the special permissions, and if it is omitted, it means that no special permissions are set on the file. In our example `755` is the same as `0755`. The first digit can be a combination of `4` for `setuid`, `2` for `setgid`, and `1` for `Sticky Bit`.

File permissions can be changed using the `chmod` command and ownership using the `chown` command.

Understanding umask

By default, on Linux systems, the default creation permissions are `666` for files, which gives read and write permission to user, group, and others, and to `777` for directories, which means read, write and execute permission to user, group, and others. Linux does not allow a file to be created with execute permissions.

The default creation permissions can be modified using the `umask` utility.

`umask` affects only the current shell environment. On most Linux distributions, the default system-wide umask value is set in `/etc/profile` file.

If you want to specify a different value on a per-user basis, edit the user's shell configuration files such as `~/.bashrc` or `~/.zshrc`. You can also change the current session `umask` value by running `umask` followed by the desired value.

To view the current mask value, simply type `umask` without any arguments:

```
umask
```

The output will include the

```
022
```

The `umask` value contains the permission bits that will **NOT** be set on the newly created files and directories.

As we have already mentioned, the default creation permissions for files are `666` and for directories `777`. To calculate the permission bits of the new files, subtract the umask value from the default value.

For example, to calculate how `umask 022` will affect newly created files and directories, use:

- Files: `666 - 022 = 644`. The owner can read and modify the files. Group and others can only read the files.
- Directories: `777 - 022 = 755`.The owner can `cd` into the directory, and list, read, modify, create or delete the files in the directory. Group and others can `cd` into the directory and list and read the files.

You can also display the mask value in symbolic notation using the `-S` option:

```
umask -S
u=rwx,g=rx,o=rx
```

Unlike the numeric notation, the symbolic notation value contains the permission bits that will be set on the newly created files and directories.

Setting the Mask Value

The file creation mask can be set using octal or symbolic notation. To make the changes permanent, set the new `umask` value in a global configuration file like `/etc/profile` file which will affect all users or in a user's shell configuration files such as `~/.profile`, `~/.bashrc` or `~/.zshrc`, which will affect only the user. The user files have precedence over the global files.

Before making changes to the `umask` value, make sure the new value doesn't pose a potential security risk. Values less restrictive than `022` should be used with great caution. For example, `umask 000` means anyone has read, write, and execute permissions on all newly created files.

Let's say we want to set more restrictive permissions for the newly created files and directories so others will not be able to `cd` to the directories and read files. The permissions we want are `750` for directories and `640` for files.

To calculate the `umask` value, simply subtract the desired permissions from the default one:

Umask value: `777-750 = 027`

The desired `umask` value represented in numeric notation is `027`.

To permanently set the new value system-wide, open the `/etc/profile` file with your text editor:

```
sudo nano /etc/profile
```

and change or add the following line at the beginning of the file:

/etc/profile
```
umask 027
```

For changes to take effect, log out and log in:

To verify the new settings, we will create one new file and directory using `mkdir` and `touch` :

```
mkdir newdirtouch newfile
```

If you check the permissions using the `ls` command, you will notice that the new file has `640` and the new directory `750` permissions, as we wanted:

```
drwxr-x--- 2 linuxize users 4096 Jul  4 18:14  newdir
-rw-r----- 1 linuxize users    0 Jul  4 18:14  newfile
```

Another way to set the file creation mask is by using symbolic notation. For example `umask u=rwx,g=rx,o=` is same as `umask 027`.

Conclusion

Here we have discusssed how to use the `umask` command to set the permissions bits for newly created files or directories.