**How to add a user to a group on RHEL 8 / CentOS 8**

Egidio Docile

**Contents**

* + [1. Software Requirements and Conventions Used](https://linuxconfig.org/redhat-8-add-user-to-group#h1-software-requirements-and-conventions-used)
  + [2. What is a group?](https://linuxconfig.org/redhat-8-add-user-to-group#h2-what-is-a-group)
  + [3. Primary and secondary groups](https://linuxconfig.org/redhat-8-add-user-to-group#h3-primary-and-secondary-groups)
  + [4. Add a user to a group by using usermod](https://linuxconfig.org/redhat-8-add-user-to-group#h4-add-a-user-to-a-group-by-using-usermod)
  + [5. Add user to a group directly](https://linuxconfig.org/redhat-8-add-user-to-group#h5-add-user-to-a-group-directly)
  + [6. Conclusions](https://linuxconfig.org/redhat-8-add-user-to-group#h6-conclusions)

In the context of a discretionary access control (DAC) mechanism, the access to system resources, files and directories, is based on the identity of the users and on the groups they are member of. This type of access control is called "discretionary" because a user can perform its own policy decisions (limited by its own permissions, of course). In this tutorial we will see how to add a user to a group and what is the difference between a primary and a secondary group on a [RHEL 8](https://linuxconfig.org/how-to-install-rhel-8) / CentOS 8 Linux system.

**In this tutorial you will learn:**

* What is the difference between a primary and a secondary group
* How to add a user to a group by using the usermod command
* How to add a user to a group directly with vigr

[](https://linuxconfig.org/images/rhel-add-user-to-group.png)

How to add a user to a group on Rhel8

**Software Requirements and Conventions Used**

| *Software Requirements and Linux Command Line Conventions* | |
| --- | --- |
| **Category** | **Requirements, Conventions or Software Version Used** |
| **System** | RHEL 8 / CentOS 8 |
| **Software** | No special software is needed to follow this tutorial |
| **Other** | Permission to run command with root privileges. |
| **Conventions** | **#** - requires given [linux commands](https://linuxconfig.org/linux-commands) to be executed with root privileges either directly as a root user or by use of sudo command **$** - requires given [linux commands](https://linuxconfig.org/linux-commands) to be executed as a regular non-privileged user |

**What is a group?**

Linux, being based on Unix, is a multi-user OS: multiple user does exist and share resources in the system at the same time. At the most simple level, access to this resources is managed by the use of a DAC (discretionary access control) model. The access to files and directories, for example, is based on a user's identity and on the groups he is is a member of. In this tutorial we will see how to add an user to an existing group on a Red Hat Enterprise Linux 8 machine.

***SUBSCRIBE TO NEWSLETTER*** *Subscribe to Linux Career*[*NEWSLETTER*](https://bit.ly/2X5D30q)*and receive latest Linux news, jobs, career advice and tutorials.*

**Primary and secondary groups**

Nowadays, Red Hat, like almost all the other major linux distributions uses a scheme which is called UPG, or User Private Group: each time a new user is created, automatically a new group with the same name of the user is created too, and the user becomes its sole member. This is what is called a primary or private group.

Each user has its own primary group, named after himself, with no other members. This setup makes possible to change the default umask value: traditionally it was 022 (this means 644 permissions for files and 755 for directories), now it is usually set to to 002 (664 permissions for files and 775 for directories).

Since, by default, each file or directory created by a user is created with that user's primary group, this setup, while preserving security (a user can still modify only its own files), simplifies resources sharing and collaboration between users which are members of the same group when the [setgid](https://linuxconfig.org/how-to-use-special-permissions-the-setuid-setgid-and-sticky-bits) bit is used, by allowing write permissions for the group.

We can obtain a list of the groups a user is member of, by using the groups command:

$ groups

egdoc wheel

As we can observe from the output of the command, the current user, egdoc, belongs to the egdoc group, which is its own primary group, and to the wheel group, which makes him able to run commands with sudo, and is what is called a secondary group: an optional group that is not associated with the user by default.

**Add a user to a group by using usermod**

While a user is the only member of its primary group, we may want to add a user to a secondary group, perhaps to grant him access to some kind of resources. Say for example we have a test user, and we want to add it to the existing group linuxconfig: the easiest and recommended way to accomplish this task is by using the usermod command:

$ sudo usermod -a -G linuxconfig test

Let's examine the options we used. The usermod utility, let us modify a user account; by using it we can perform a vast range of operations, like changing a user's home directory, setting an expiration date for its account or lock it immediately. The command let us also add the user to an existing group. The options we used in this case are -G (short for --groups) and -a, (which is the short form of --append).

The -G or --groups option let us provide a list of comma-separated supplementary groups the user should be a member of. As we said before, each provided group must already exist on the system. One very important thing to remember is that the list of the provided groups is interpreted differently whether the -a option is also provided or not: in the first case, the list is interpreted as the supplementary groups the user should be added to in addition to the ones he is already a member of; when the -a option is not provided, instead, the list is interpreted as the absolute list of groups the user should be a member of. As stated in the command manpage, in the latter case, if the user is currently member of a group which is not part of the list provided to the command, it will be removed from that group!

The user "test" is now a member of the "linuxconfig" group. Let's verify it:

$ sudo groups test

test : test linuxconfig

**Add user to a group directly**

Using usermod is the easiest way to add a user to a group. For the sake of completeness, we now will examine another way of performing the same task by using the vigr [linux command](https://linuxconfig.org/linux-commands). This command let us edit the /etc/group and /etc/gshadow files directly, also locking them while they are open, to prevent their corruption and ensure consistency.  
  
The "shadow" version of the file (/etc/gshadow) is modified only when the -s option is used. To add our "test" user to the "linuxconfig" group with this method, we should run the vigr command as superuser: the /etc/group file will be opened in the default editor (usually vi):

[...]

chrony:x:993:

egdoc:x:1000:

cgred:x:992:

docker:x:991:

apache:x:48:

test:x:1001:test

linuxconfig:x:1002:

[...]

The syntax used to represent each group is the following:

group-name:group-password:group-id:users

The fields are separated by a colon: the first one is the group name, the second one is the "password" of the group (which is usually not set) and the third field is the GID or group-id. The last field is the comma separated list of the members of the group. To add our "test" user to the "linuxconfig" group, we should modify this field, so that the line becomes:

linuxconfig:x:1002:test

Once the change is performed, we can save and close the file. A message will appear on the terminal:

You have modified /etc/group.

You may need to modify /etc/gshadow for consistency.

Please use the command 'vigr -s' to do so.

Since we changed the /etc/group file, the message suggests us to change also the related shadow file, which is /etc/gshadow. For those of you who don't know, a shadow file, is used to store the encrypted version of information that would not be safe to store in plaintext form. For example, as we saw before, an x is reported in the /etc/group file, in place of the optional group password; the hashed version of the password, if existent, would be stored in the shadow-file.

Now, lets make the same change we did before, to the /etc/gshadow file, so that it gets in sync with /etc/group. All we have to do, is to provide the -s flag to the vigr command:

$ sudo vigr -s

Once the file is opened, we make the needed change:

linuxconfig:!::test

After that, we must force the writing of this file, since it is read-only: when using vi, we can do this by running the w! command.

***SUBSCRIBE TO NEWSLETTER*** *Subscribe to Linux Career*[*NEWSLETTER*](https://bit.ly/2X5D30q)*and receive latest Linux news, jobs, career advice and tutorials.*

An alternative way to keep the two files in sync, is to use the grpconv command, which creates the /etc/gshadow file from /etc/group, and optionally from an already existing /etc/gshadow file:

$ sudo grpconv

At this point, we can verify the consistency between the two files by running:

$ sudo grpck

No output should be displayed at this point.

**Conclusions**

In this tutorial we saw the difference between a primary and a secondary group and what are their roles in a DAC model. We saw how can we add a user to a group either by using the usermod command, which is the recommended way, or directly by using the vigr command securely editing the /etc/group and /etc/gshadow files. Whatever procedure you decide to use to perform this administrative task, you should always pay the maximum attention.