**Suid and Sgid**

**Suid :-**

* File will Execute as Owner of the File irespective of who is Running it.
* **SUID** (**S**et owner **U**ser **ID** up on execution) is a special type of file permissions given to a file. Normally in Linux/Unix when a program runs, it inherit’s access permissions from the logged in user. SUID is defined as giving temporary permissions to a user to run a program/file with the permissions of the file owner rather that the user who runs it**. In simple words users will get file owner’s permissions as well as owner UID and GID when executing a file/program/command.**

**Learn SUID with examples :-**

* **Example1:  passwd command**

When we try to change our password we will use **passwd command,** which is owned by root. This **passwd command** file will try to edit some system config files such as /etc/passwd, /etc/shadow etc when we try to change our password. Some of these files cannot be opened or viewed by normal user only root user will have permissions. So if we try to remove SUID and give full permissions to this passwd command file it cannot open other files such as /etc/shadow file to update the changes and we will get permission denied error or some other error when tried to execute passwd command. So passwd command is set with SUID to give root user permissions to normal user so that it can update /etc/shadow and other files.

* **Example2: ping command**

Similarly if we take ping command, when we have to execute this command internally it should open socket files and open ports in order to send IP packets and receive IP packets to remote server. Normal users don’t have permissions to open socket files and open ports. So SUID bit is set on this file/command so that whoever executes this will get owner (Root user’s) permissions to them when executing this command. So when this command start executing it will inherit root user permissions to this normal user and opens require socket files and ports.

* **Example3: crontab and at command**

When scheduling the jobs by using crontab or at command it is obvious to edit some of the crontab related configuration files located in /etc which are not writable for normal users. So crontab/at commands are set with SUID in-order to write some data.

# ****How can I setup SUID for a file?****

Symbolic way:

[chmod](http://www.linuxnix.com/2011/10/chmod-command-explained-linuxunix.html) u+s file1.txt

Here owner permission execute bit is set to SUID with +s

Numerical way:

chmod 4750 file1.txt

**Removing SUID permission**

Symbolic method:

# chmod u-s /myscript.sh  
# ls -l  
-rwxrw-rw-. 1 root root 0 Oct 16 11:36 /myscript.sh

Numerical method:

# chmod 0755 /myscript.sh  
# ls -l  
-rwxrw-rw-. 1 root root 0 Oct 16 11:35 /myscript.sh

## ****Where is SUID used?****

1) Where root login is required to execute some commands/programs/scripts.

2) Where you don’t want to give credentials of a particular user, but want to run some programs as the owner.

3) Where you don’t want to use SUDO command, but want to give execute permission for a file/script etc.

**SGID ( setgid) :-**

Same as SUID, The process will have the same group rights of the file being executed. If SGID bit is set on any directory, all sub directories and files created inside will get same group ownership as main directory, it doesn’t matter who is creating.

**How to set SGID on a directory**

# chmod g+s [path\_to\_directory]

**How to Remove SGID on a directory**

# chmod g-s [path\_to\_directory]