getuid, geteuid - get user identity

SYNOPSIS

```
#include <unistd.h>
#include <sys/types.h>

uid_t getuid(void);
uid_t geteuid(void);
```

DESCRIPTION

getuid() returns the real user ID of the calling process.

geteuid() returns the effective user ID of the calling process.

ERRORS

These functions are always successful.

setuid - set user identity

SYNOPSIS

```
#include <sys/types.h>
#include <unistd.h>

int setuid(uid_t uid);
```

DESCRIPTION

If the user is root or the program is set-user-ID-root, special care must be taken: setuid() checks the effective user ID of the caller and if it is the superuser, all process-related user ID's are set to uid. After this has occurred, it is impossible for the program to regain root privileges.

Thus, a set-user-ID-root program wishing to temporarily drop root privileges, assume the identity of an unprivileged user, and then regain root privileges afterward cannot use setuid(). You can accomplish this with seteuid(2).

RETURN VALUE

On success, zero is returned. On error, -1 is returned, and errno is set appropriately.

getgid, getegid - get group identity

SYNOPSIS

```
#include <unistd.h>
#include <sys/types.h>

gid_t getgid(void);
gid_t getegid(void);
```

DESCRIPTION

getgid() returns the real group ID of the calling process.

getegid() returns the effective group ID of the calling process.

ERRORS

These functions are always successful.

Program

```
#include<stdio.h>
#include<stdlib.h>

int main(void)
{
   printf("user id : %d \t effective uid %d\n",getuid(),geteuid());
   printf("group id : %d \t effective gid %d\n",getgid(),getegid());
   setuid(6000);
   printf("After setuid user id : %d \t effective uid %d\n",getuid(),geteuid());
}
```

Output:

```
student@513-4:~/temp$ gcc getuid.c
student@513-4:~/temp$ sudo ./a.out
user id : 0      effective uid 0
group id : 0      effective gid 0
After setuid user id : 6000      effective uid 6000
```