3.Implement system calls(getgid()).

Implementing the getgid() system call in a Linux-like operating system kernel is an advanced task, usually done in kernel development (like in Linux kernel modules or writing system-level code in C).

How to Implement getgid() (System CallLlevel)

Here's how you might implement it inside a kernel, simplified for educational purposes (like adding a syscall in Linux kernel source):

Step 1: Define the System Call

In the kernel source (e.g., kernel/sys.c), add:

```
#include <linux/syscalls.h>
#include <linux/sched.h>

SYSCALL_DEFINE0(getgid)
{
return current->cred->gid.val;
}
```

- current is the current process.
- cred is the credentials structure.
- gid.val accesses the real group ID.

Step 2: Declare It in Headers

In include/linux/syscalls.h, declare:

```
asmlinkage long sys_getgid(void);
```

Or for newer kernels using SYSCALL_DEFINE0, no need — it generates automatically.

Step 3: Add to System Call Table

In arch/x86/entry/syscalls/syscall_64.tbl (or for your architecture):

```
__NR_getgid 50  common   getgid
```

- 50 is the syscall number (can vary).
- common means available to all users.

Step 4: Compile and Test

Recompile the kernel and reboot into the new version. Then from user space:

```
#include <unistd.h>

#include <stdio.h>

int main() {
```

```
printf("GID: %d\n", getgid());
return 0;
}
```

Compile and run it:

```
gcc getgid_test.c -o test
./test
```