System call on tails OS

A **system call** is a mechanism that enables user-level programs to request services from the operating system's kernel. Because user applications are not permitted to directly access hardware or perform privileged operations, they rely on system calls to interact with system resources such as memory, files, devices, and processes. When a system call is made, the CPU switches from **user mode** to **kernel mode**, allowing the kernel to securely execute the requested function. After completing the operation, control is returned to the user program.

Implementing the mprotect() System Call

Now, I am going to demonstrate how to implement and use the mprotect() system call on the **Tails OS**. The mprotect() system call is used in Unix-like systems to change the access permissions (such as read, write, and execute) of a specific region of memory that was previously allocated, typically via the mmap() system call.
To implement and test this system call on Tails OS, we need to follow a few steps:

Install Required Development Tools

First, open the Terminal.Run the following commands to set up a C/C++ development environment on Tails OS (which is based on Debian):

```
sudo apt update
sudo apt install build-essential
```

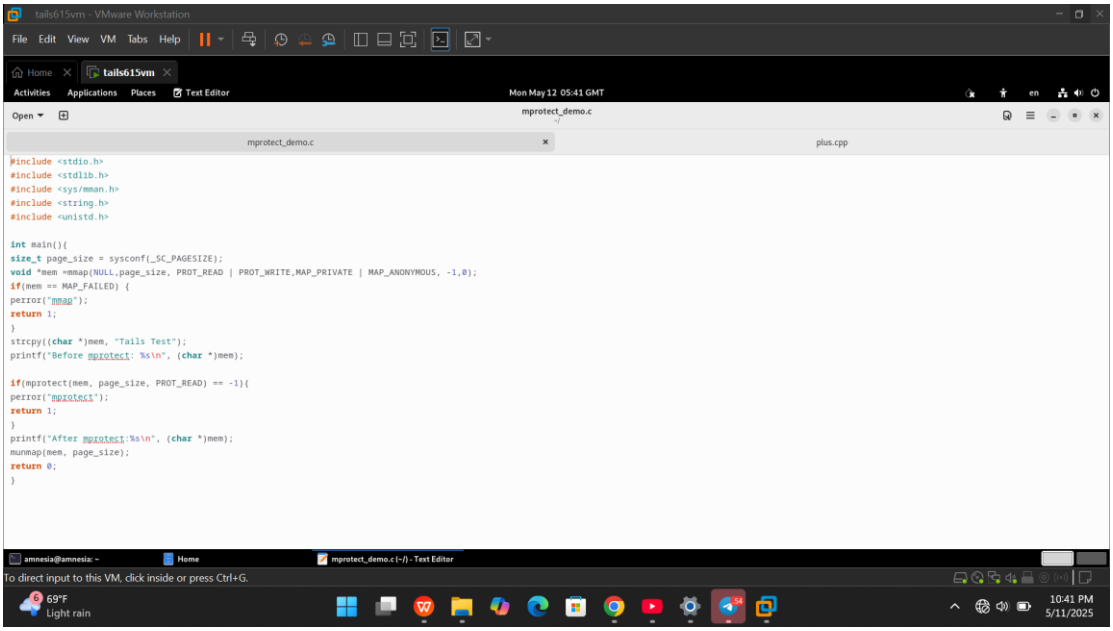
These commands ensure that the system has the necessary tools like gcc, make, and other essential libraries for compiling and building programs

Create the Source File

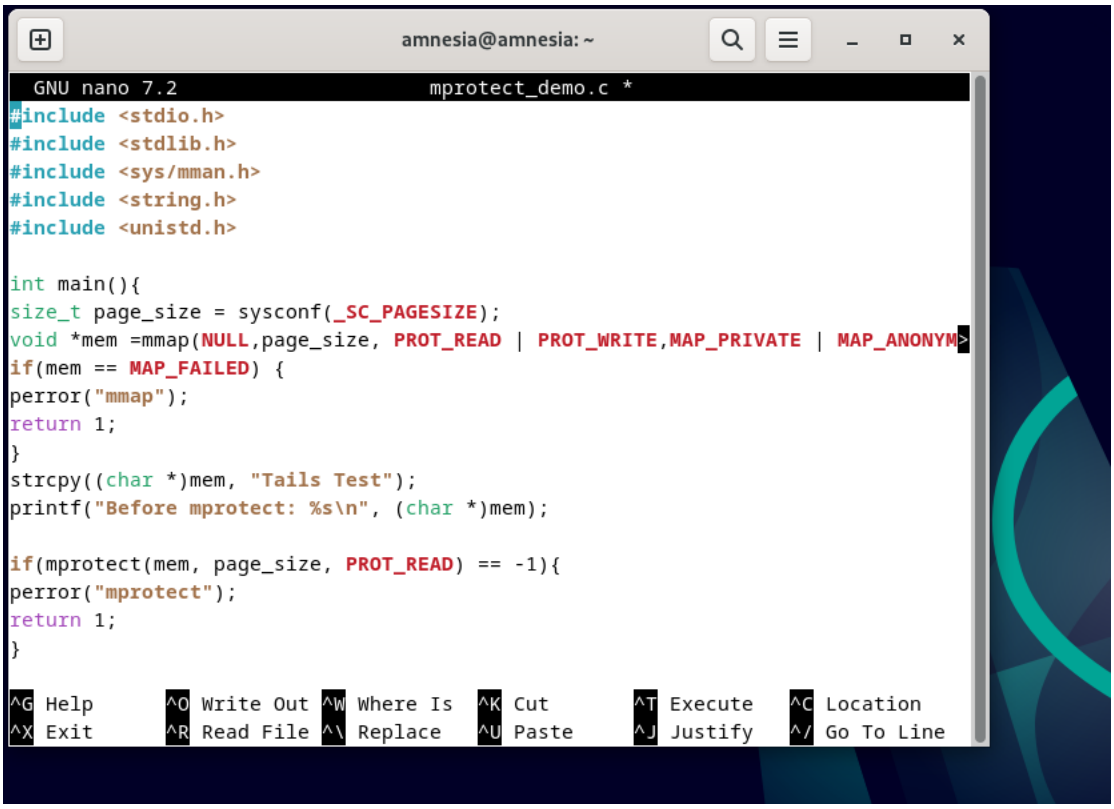
In the Terminal, create a new source file using the nano text editor:

```
nano mem_protect.c
```

This will open a text editor where you can write the C code to demonstrate or test the mprotect() system call.



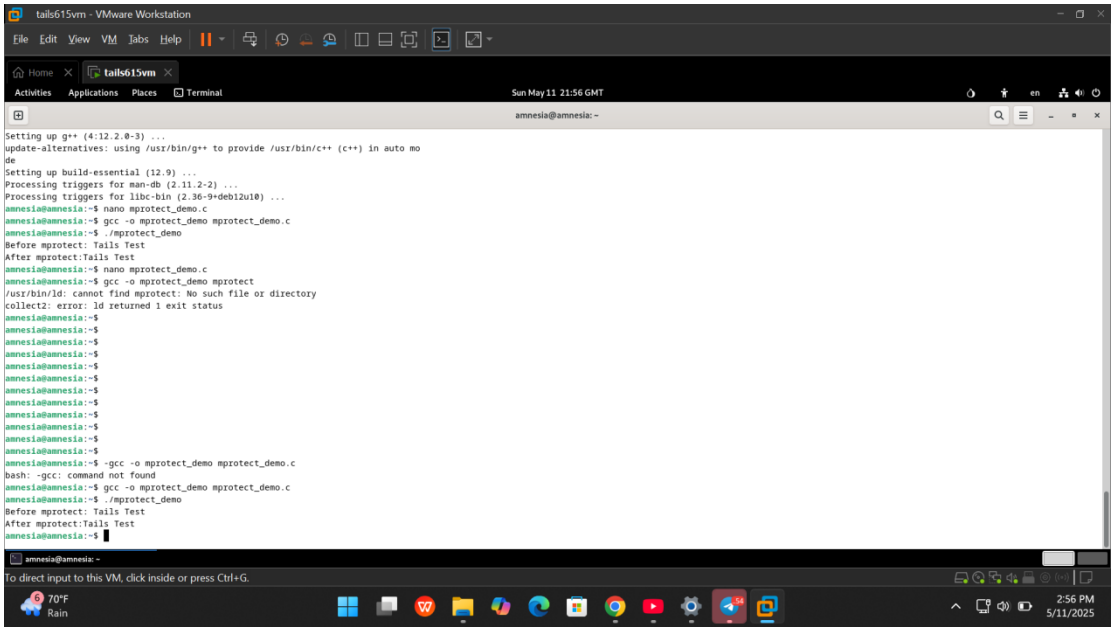
After I write the c code on the terminal it looks like

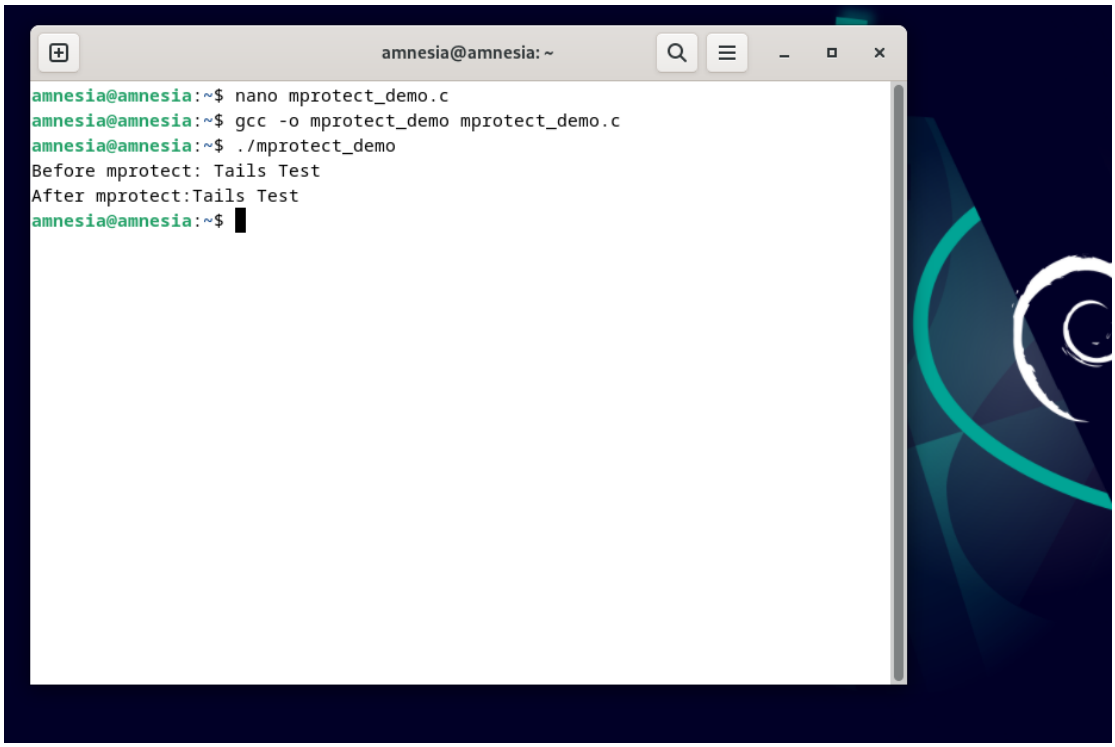


After this I we will use gcc -o mem_protect mem_protect.c to compile the code and ./mem_protect To run the code.

Before mprotect: Tails Test

After mprotect: Tails Test this is my expected out put.






```
Before mprotect: Tails Test
After mprotect: Tails .
```

What This Output Means:

Before mprotect():
The program successfully writes the string "Tails Test" into the memory. This means the memory has read and write permissions at this point. The full string is printed as expected.

After mprotect():
The program changes the memory region’s permissions to read-only using the mprotect() system call. After this, it tries to overwrite part of the string (e.g., replacing "Test" with "."). However, because the memory is now read-only, this write operation is blocked.