AOS Assignment - 2

Adding system calls to Linux kernel

Overview

- Download the 4.19.210 Linux kernel from the official source.
- Extract the compressed kernel.
- Add the directory with the new system call and its makefile.
- Add the directory to the kernel makefile.
- Add the system call to syscalls_64.tbl file.
- Add the system-call signature to syscalls.h file.
- Install necessary dependencies.
- Compile the kernel using sudo make command.
- Install the kernel using sudo make modules_install install command.
- Reboot into 4.19.210 kernel.
- Test the newly added system call.

1. Downloading and Extract Kernel

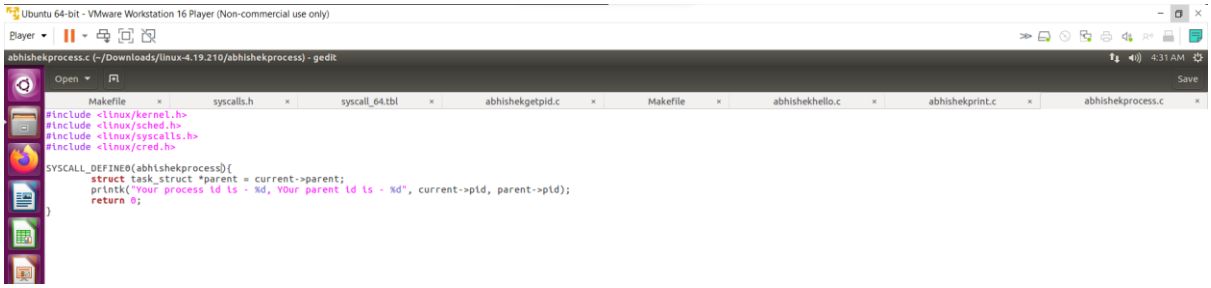
- Use the **wget** command to download the kernel.
- Use the **tar** command to extract it.

2. Writing System Calls

- After extracting the kernel, change the working directory to it.
- Now create a directory for each system call.
- Create **.c file** and **Makefile** for each system call in its respective directory.
- For Example -
    - Directory - abhishekhello
        - File - abhishekhello.c (contains the system call function definition)
        - File - Makefile (contains the recipe for compilation)
- Sample content of Makefile for a system-call -
    - obj-y := abhishekhello.o

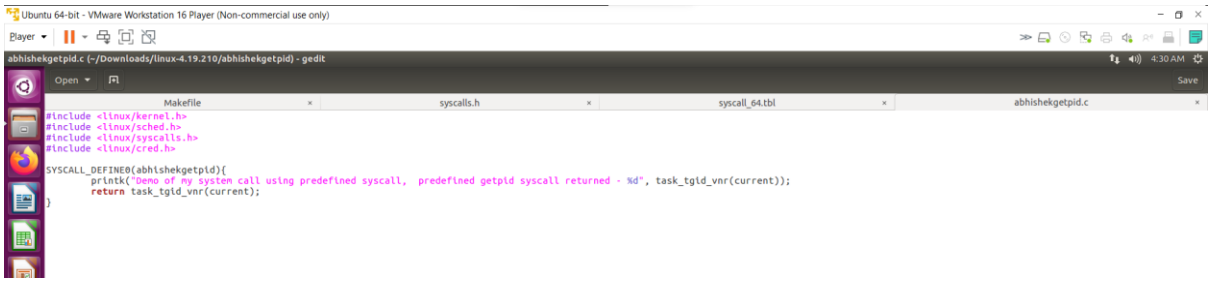
2.2 Print the Argument Passed From Userspace in Kernelspace

2.3 Print the Current Process and Parent Process ID



- The parent and child process id are different because parent process uses fork to create child process, which assigns a new pid.
- In our case **parent process is bash shell and child process is ./a.out**

2.4 Use Any Pre-defined System-call



3. Adding System Call Table Entry

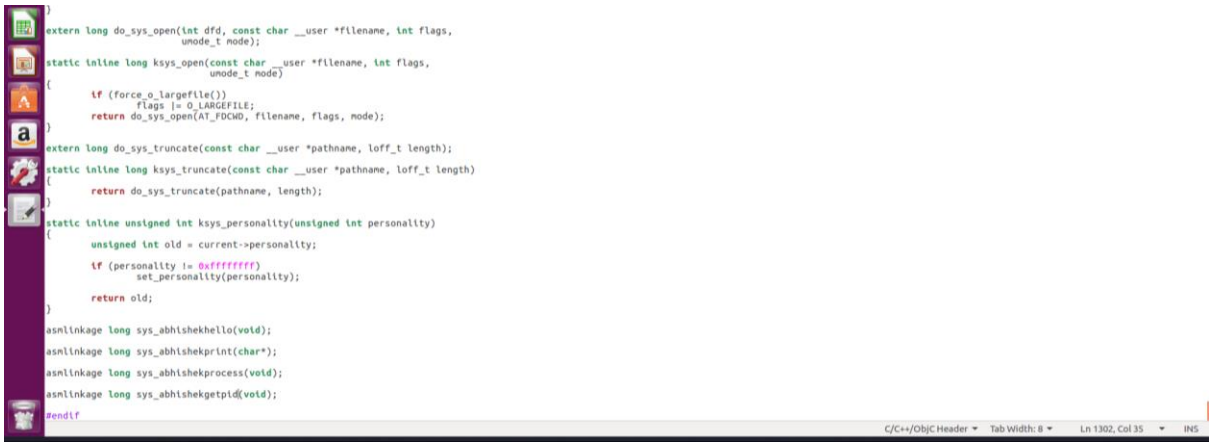
- Navigate to /linux-4.19.210/arch/x86/entry/syscalls.
- Open **syscall_64.tbl file**.
- Make entries for newly created system-calls at the latest available position.
- My entries are at **335, 336, 337, 338**.

/linux-4.19.210/arch/x86/entry/syscalls/syscall_64.tbl

4. Adding System Call Signature to Header File

- Navigate to /linux-4.19.210/include/linux.
- Open **syscalls.h**.
- Make entries for newly added system-calls at the end of the file with the function declaration.

/linux-4.19.210/include/linux/syscalls.h



5. Adding Directories to Kernel Makefile

- Now we want to let the Makefile know the directories in which our system-calls are present, so that while compiling the kernel, our system calls will also be included.
- For this, Navigate to /linux-4.19.210.
- Open **Makefile** and **all directories** which contains newly added system-calls to it.

/linux-4.19.210/Makefile

6. Compiling and Installing the Kernel

- Run the following commands.
  - sudo make
  - sudo make modules_install install
- After successful execution,the /boot directory is now populated with newly installed kernel with version 4.19.210. Verify the same.
- Now reboot into kernel 4.19.210

7. Testing the System Calls

- Verify the kernel version using **uname -r** command.
- The test code contains the function calls with syscall(335), syscall(336), syscall(337), syscall(338).
- The output of these syscalls is printed in **/var/log/syslog** file.
- This can be viewed with dmesg command.
- Following is the testing code and resultant output.

test.c

Output of test.exe

Output of dmesg