Bahir Dar University
BIT Faculty Of Computing
Department Of Software Engineering

Course Title: Operating Systems and System Programming

Individual Assignment

SYSTEM CALL IMPLEMENTATION

UBUNTU 25.04 SERVER OS

Prepared by:
Degnet Habtamu
Section A
ID Number: 1601270

Submitted to: Lec. Wendimu B.

Submittion date: 04/09/2017

3. System call

A system call is a mechanism that allows a program to request services from the operating system's kernel. It serves as a link between user apps and the OS-managed hardware resources.

I've been given the task of working with the system call mmap().

What is mmap()?

Unix-like operating systems offer a robust way to map files or anonymous memory into a process's virtual address space through the mmap() system call.

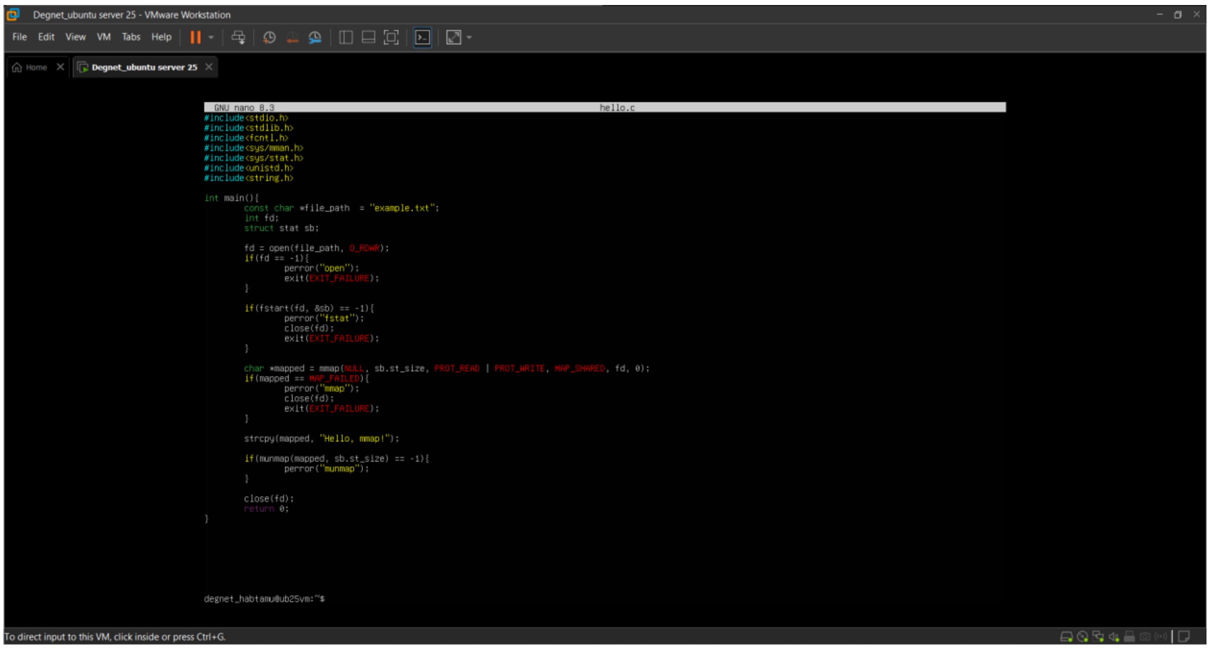
The mmap system call in Ubuntu Server is a function used to map files or devices into memory. It allows a process to access files as if they were part of its memory space, enabling efficient file I/O operations.

Key Features of mmap()

1. **Memory Mapping**: mmap maps a file or device into the virtual memory space of a process, allowing direct access to the file's contents.
2. **Performance:** It can improve performance by reducing the number of read and write system calls needed, as the file can be accessed directly in memory.
3. **Shared Memory:** mmap can be used to share memory between processes, facilitating inter-process communication.
4. **Lazy Loading:** It supports lazy loading of file contents, meaning that data is loaded into memory only when accessed.

To implement the mmap system call on Ubuntu Server 25.04, write a C program that demonstrates how to use mmap for memory-mapped file I/O.

```c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sys/mman.h>
#include <sys/stat.h>
#include <unistd.h>
#include <string.h>

int main() {
    const char *file_path = "example.txt";
    int fd;
    struct stat sb;

    // Open the file
    fd = open(file_path, O_RDWR);
    if (fd == -1) {
        perror("open");
        exit(EXIT_FAILURE);
    }

    // Get file size
    if (fstat(fd, &sb) == -1) {
        perror("fstat");
        close(fd);
        exit(EXIT_FAILURE);
    }

    // Memory-map the file
    char *mapped = mmap(NULL, sb.st_size, PROT_READ | PROT_WRITE,
MAP_SHARED, fd, 0);
    if (mapped == MAP_FAILED) {
        perror("mmap");
        close(fd);
        exit(EXIT_FAILURE);
    }

    // Modify the mapped memory
    strcpy(mapped, "Hello, mmap!");

    // Clean up
    if (munmap(mapped, sb.st_size) == -1) {
        perror("munmap");
    }

    close(fd);
    return 0;
}
```


Degnet_ubuntu server 25 - VMware Workstation
File Edit View VM Tabs Help
Home
Degnet_ubuntu server 25
To direct input to this VM, click inside or press Ctrl+G.

```
  GNU nano 8.3                                                   hello.c
#include<stdio.h>
#include<stdlib.h>
#include<fcntl.h>
#include<sys/mman.h>
#include<sys/stat.h>
#include<unistd.h>
#include<string.h>

int main(){
        const char *file_path  = "example.txt";
        int fd;
        struct stat sb;

        fd = open(file_path, O_RDWR);
        if(fd == -1){
                perror("open");
                exit(EXIT_FAILURE);
        }

        if(fstart(fd, &sb) == -1){
                perror("fstat");
                close(fd);
                exit(EXIT_FAILURE);
        }

        char *mapped = mmap(NULL, sb.st_size, PROT_READ | PROT_WRITE, MAP_SHARED, fd, 0);
        if(mapped == MAP_FAILED){
                perror("mmap");
                close(fd);
                exit(EXIT_FAILURE);
        }

        strcpy(mapped, "Hello, mmap!");

        if(munmap(mapped, sb.st_size) == -1){
                perror("munmap");
        }

        close(fd);
        return 0;
}




degnet_habtamu@ub25vm:~$
```

Implementation of mmap () in Ubuntu 25.04 server OS