

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN
HỆ ĐIỀU HÀNH - 21CLC04

Đồ án

SYSTEM CALLS & FILE NETWORK OPERATIONS



Giáo viên hướng dẫn

Lê Giang Thanh
Nguyễn Thanh Quân
Lê Hà Minh

Thành viên

Lý Nhật Hào - 21127041
Nguyễn Hữu Khánh - 21127072
Lê Văn Dương - 21127500

NIÊN KHOÁ 2022 - 2023



LỜI CẢM ƠN

Để hoàn thành được bài báo cáo này, chúng em đã nhận được sự giúp đỡ rất nhiều từ phía thầy cô giảng viên, trợ giảng và bạn bè. Nay em xin gửi lời cảm ơn chân thành và sâu sắc nhất đến giảng viên môn Hệ Điều Hành lớp 21CLC4, Khoa Công nghệ thông tin :

- Thầy **Lê Hà Minh**
- Thầy **Lê Giang Thanh**
- Thầy **Nguyễn Thanh Quân**

Các thầy cô đã đồng hành cùng nhóm, đã luôn quan tâm, hướng dẫn và truyền đạt, cung cấp kiến thức, tài liệu và các thủ thuật cần thiết để nhóm có thể hoàn thành bài báo cáo!

Trong quá trình thực hiện đồ án không thể tránh khỏi những thiếu sót. Chúng em rất mong nhận được nhiều ý kiến đóng góp từ các thầy, cô giáo và bạn bè để đồ án ngày càng hoàn thiện hơn!

Xin chân thành cảm ơn!

LỜI CAM ĐOAN



Chúng tôi xin cam đoan rằng “**System calls & File - Network Operations**” trong môn “**HỆ ĐIỀU HÀNH**” bao gồm nội dung bài báo cáo, đáp án các câu hỏi, các kết quả, số liệu trình bày trong nghiên cứu là trung thực, do chính tác giả thực hiện và nghiên cứu trong quá trình làm việc cùng nhau.



THÀNH VIÊN NHÓM BÁO CÁO

X

X

X

Lý Nhật Hào

Lê Văn Dương

Nguyễn Hữu Khánh



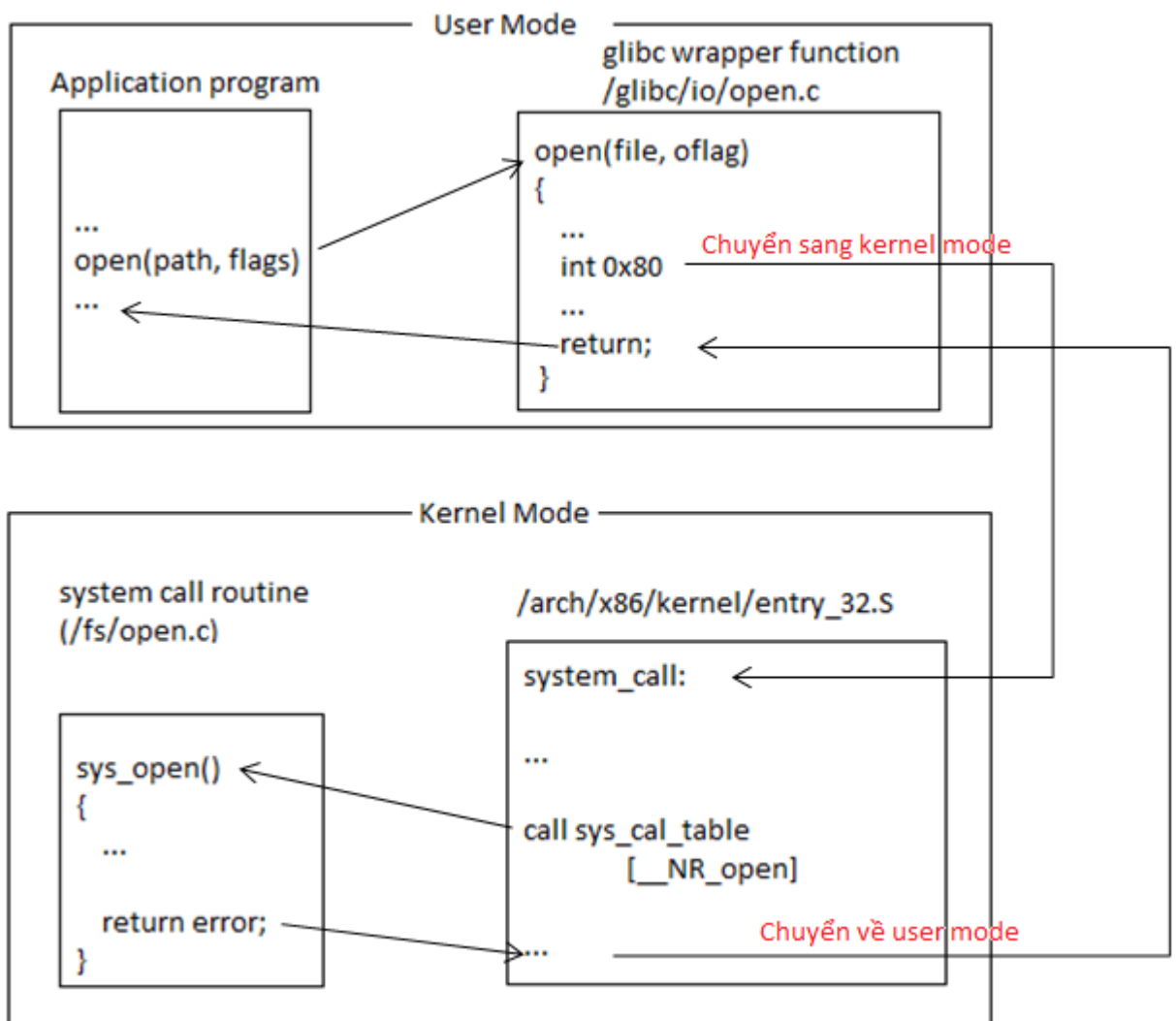
MỤC LỤC	
LỜI CẢM ƠN	2
LỜI CAM ĐOAN	3
MỤC LỤC	4
Chương I: Giới Thiệu	5
1.1. Lời giới thiệu:	5
1.2. Đôi nét về Thông tin thành viên:	6
1.3. Tổng quát yêu cầu đề án:	6
CHƯƠNG II: MỨC ĐỘ HOÀN THIỆN VÀ ĐÓNG GÓP CỦA CÁC THÀNH VIÊN	7
2.1. Mức Độ Hoàn Thiện:	7
Bảng ghi chú mức độ hoàn thiện đề án	8
2.2. Đóng Góp Của Thành Viên:	9
LÊ VĂN DƯƠNG:	9
LÝ NHẬT HÀO:	9
NGUYỄN HỮU KHÁNH:	10
CHƯƠNG III: KỊCH BẢN GIAO TIẾP	11
3.1. Giao thức trao đổi giữa client và server:	11
3.2. Môi trường lập trình và các Framework hỗ trợ:	13
3.3. Cấu trúc và Kiểu dữ liệu:	13
3.4. Hướng dẫn sử dụng và các tính năng của chương trình:	13
CHƯƠNG IV: KẾT LUẬN	15
CHƯƠNG V: PHỤ LỤC VÀ TÀI LIỆU THAM KHẢO	16
5.1. Phụ Lục:	16
5.1.1. Quá Trình Làm Việc và Biên Bản Họp Nhóm:	16
BIÊN BẢN HỌP NHÓM LẦN 1 NGÀY 20/11/2022	16
BIÊN BẢN HỌP NHÓM LẦN 2 NGÀY 27/11/2022	18
BIÊN BẢN HỌP NHÓM LẦN 3 NGÀY 14/12/2022	21
5.1.2. Hình ảnh làm việc nhóm thông qua các công cụ có sẵn:	24
5.2. Tài Liệu Tham Khảo:	25

Chương I: Giới Thiệu

1.1. Lời giới thiệu:

Hệ Điều Hành (Operating Systems) là một phần không thể thiếu trong một hệ thống máy tính. Một máy tính mặc dù đắt tiền, cấu hình cao nhưng nếu thiếu đi hệ điều hành thì hầu như không thể hoạt động được. Hệ điều hành điều khiển mọi hoạt động của máy tính, giúp việc sử dụng máy tính trở nên đơn giản, dễ dàng và hiệu quả hơn rất nhiều.

Lời gọi hệ thống (system call) là một yêu cầu do một chương trình thực hiện đối với hệ điều hành (operating system). Nó cho phép một ứng dụng truy cập các hàm (function) và lệnh từ API của hệ điều hành.



1.2. Đôi nét về Thông tin thành viên:

Nhóm thực hiện đồ án bao gồm 3 thành viên:

- **Lý Nhật Hào**
 - MSSV: 21127041.
 - Email: lnhao21@clc.fitus.edu.vn
- **Lê Văn Dương**
 - MSSV: 21127500.
 - Email: lvduong21@clc.fitus.edu.vn
- **Nguyễn Hữu Khánh**
 - MSSV: 21127072.
 - Email: nhkhanh21@clc.fitus.edu.vn

1.3. Tổng quát yêu cầu đồ án:

Part1. Implement system call for File operations

1. Implement the **int Create(char *name)** system call. The createfile system call will use the Nachos Filesystem Object Instance to create a zero-length file. Remember, the filename exists in user space. This means the buffer that the user space pointer points to must be translated from user memory space to system memory space. The createfile system call returns 0 for successful completion, -1 for an error.
2. Implement the **OpenFileID Open(char *name, int type)** and **int Close(OpenFileID id)** system calls. The user program can open two types of “files”, files that can be read only and files that can be read and write. Each process will allocate a fixed size file descriptor table. For now, *set this size to be 20 file descriptors*. The first two file descriptors, 0 and 1, will be reserved for console input and console output respectively. The open file system call will be responsible for translating the user space buffers when necessary and allocating the appropriate kernel constructs. You will use the filesystem objects provided to you in the filesystem directory. (NOTE: We are using the FILESYSTEM_STUB code) The calls will use the Nachos Filesystem Object Instance to open and close files.

The Open system call returns the file descriptor id (OpenFileID == an integer number), or -1 if the call fails. Open can fail for several reasons, such as trying to open a file or mailbox that does not exist or if there is not enough room in the file descriptor table. The type parameter will be set to 0 for a standard file



and 1 for a read only file. If the type parameter is set to any other value, the system call should fail. The close system call will take a file descriptor as the parameter. The system call will return -1 on failure and 0 on success

3. Implement the int **Read(char *buffer, int charcount, OpenFileID id)** and int **Write(char *buffer, int charcount, OpenFileID id)** system calls. These system calls respectively read and write to a file descriptor ID. Remember, you must translate the character buffers appropriately and you must differentiate between console IO (OpenFileID 0 and 1) and File (any other valid OpenFileID).

In case the Console is a read and write console, using the *SynchConsoleInput* class and the *SynchConsoleOutput* class for reading and writing, you must ensure that the correct data is returned to the user. Reading and writing to the console will return the correct number of characters read or written, not the charcount. In case of read or write failure, return -1

4. Implement the int **Seek(int pos, OpenFileID id)** system call. Seek will move the file cursor to a specified location. The parameter pos will be the absolute character position within a file. If pos is a -1, the position will be moved to the end of file. The system call will return the actual file position upon success, -1 if the call fails. Seeks on console I/O will fail
5. Install system call int **Remove(char *name)**. The Remove system call will use the Nachos FileSystem Object to delete the file. Note: need to check if the file is open or not before deleting.

Part2. Implement system call for Network operations

1. Implement system call int **SocketTCP()**. You build an array of file descriptor tables with the size of 20 file descriptors.

The system function call "**SocketTCP**" will return the file descriptor id (int is an integer), or -1 if there is an error.

2. Implement system call int **Connect(int socketid, char *ip, int port)**. Connect to server according to IP and port information. Returns 0 if the connection is successful, and -1 if it fails
3. Implement system call int **Send(int socketid, char *buffer, int len), int Receive(int socketid, char *buffer, int len)**. Send and receive data from socket.



- If successful, returns the number of bytes sent or received
 - If connection is closed, return 0
 - If it fails, return -1
4. Implement system call **int Close(int socketid)**. Close socket with 0 for success, -1 for error

Part3. Advanced

As part 1 and part 2, there are 2 file descriptor table arrays. In this requirement,

- You are only allowed to use **a single array** for file and network management
- Only use **Read/Write** system call for file system and network (it means that you don't use **Send/Receive** system call at all).

Part4. Test program

CHƯƠNG II: MỨC ĐỘ HOÀN THIỆN VÀ ĐÓNG GÓP CỦA CÁC THÀNH VIÊN

2.1. Mức Độ Hoàn Thiện:

- Nhóm nghiên cứu đã **hoàn thành trả lời toàn bộ 100% yêu cầu** của đề án.
- Kết quả sau khi thực hiện đã được trình bày đầy đủ trong bản báo cáo đề án, kèm hình ảnh minh họa, minh chứng và giải thích câu trả lời.
- Bài báo cáo bao gồm đầy đủ các yêu cầu đề bài bao gồm:
 - Giới thiệu.
 - Thành viên, đóng góp, mức độ hoàn thành công việc.
 - Mô tả kết quả bài tập của đề án.
 - Kịch bản giao tiếp của chương trình.
 - Hướng dẫn sử dụng các tính năng chương trình
 - Kết luận .

- Phụ lục(hình ảnh, tài liệu tham khảo và biên bản họp nhóm)
- Các thành viên có sử dụng công cụ quản lý và làm việc nhóm như :
Discord, Google Meet và Messenger để trao đổi, nghiên cứu, lên kế hoạch và thực hiện bài báo cáo.

Bảng ghi chú mức độ hoàn thiện đề án

PART		Ý nghĩa	Test-cases
1	1	syscall Create	<input checked="" type="checkbox"/>
	2	syscall OpenFileID, Close	<input checked="" type="checkbox"/>
	3	syscall Read,Write	<input checked="" type="checkbox"/>
	4	syscall Seek	<input checked="" type="checkbox"/>
	5	syscall Remove	<input checked="" type="checkbox"/>
2	1	syscall socketTCP	<input checked="" type="checkbox"/>
	2	syscall Connect	<input checked="" type="checkbox"/>
	3	syscall Send	<input checked="" type="checkbox"/>
		syscall Receive	<input checked="" type="checkbox"/>
3		Advanced	<input checked="" type="checkbox"/>
4		Test program	<input checked="" type="checkbox"/>
	1,2,3,4,5	create, copy, cat, delete, concatenate (0,25 for each of parts)	<input checked="" type="checkbox"/>

	6	echo	<input checked="" type="checkbox"/>
	7	file transfer	<input checked="" type="checkbox"/>
		Report (your project will not be graded without the report)	<input checked="" type="checkbox"/>

2.2. Đóng Góp Của Thành Viên:

Tất cả các thành viên trong nhóm đều có sự nỗ lực cũng như đóng góp một cách công bằng, có sự thống nhất và đồng đều nhau cũng như đã đóng góp một cách tốt nhất có thể để cho đồ án trên được hoàn thành tốt đẹp.

Sau đây là những công việc chi tiết, cụ thể của từng thành viên trong nhóm từ lúc được phân công sau cuộc họp đầu tiên vào ngày 20/11/2022 cho đến khi hoàn tất công việc.

LÊ VĂN DƯƠNG:

- Nghiên cứu, lập trình và biên dịch được chương trình với Ubuntu.
- Tìm hiểu về lập trình phân socket trong nachos, các kiến thức để hoàn thành phần 2 đồ án.
- Nghiên cứu phân nâng cao cho đồ án
- Tham gia đầy đủ các cuộc họp để trao đổi với cả nhóm về những kiến thức mới, phù hợp yêu cầu đề bài để đẩy nhanh quá trình hoàn thiện bài tập lớn.
- Cùng với nhóm thực hiện demo bài tập thông qua họp nhóm trực tiếp hoặc ứng dụng Messenger video call..
- Đóng góp ý kiến, chỉnh sửa và bổ sung hoàn thiện các thiếu sót tồn đọng trong bài báo cáo.

LÝ NHẬT HÀO:

- Tìm hiểu về lập trình phần Syscall trong nachos, các kiến thức để hoàn thành phần 1 và phần 4(Test) của đồ án.
- Kiểm tra kết quả chương trình, kiểm duyệt hình ảnh demo, số liệu, hình ảnh minh họa của nhóm để đưa ra kết luận phù hợp nhất giải quyết yêu cầu bài tập, được phân công nghiên cứu và chịu trách nhiệm chính trong việc viết bài báo cáo chi tiết.
- Thành lập các văn bản họp nhóm, thu thập nguồn tham khảo tài liệu cho các bài tập của nhóm.
- Tham gia đầy đủ các cuộc họp để trao đổi với cả nhóm về những kiến thức mới, phù hợp yêu cầu đề bài để đẩy nhanh quá trình hoàn thiện bài tập lớn.
- Cùng với nhóm thực hiện demo bài tập thông qua họp nhóm trực tiếp hoặc ứng dụng Messenger video call..
- Tiếp thu ý kiến của nhóm để chỉnh sửa và bổ sung hoàn thiện các thiếu sót tồn đọng trong bài báo cáo.

NGUYỄN HỮU KHÁNH:

- Nghiên cứu, lập trình và biên dịch được chương trình với Ubuntu.
- Tìm hiểu về lập trình phần Syscall trong nachos, các kiến thức để hoàn thành phần 1 của đồ án.
- Nghiên cứu và thực hiện chính cho phần nâng cao của đồ án.
- Tham gia đầy đủ các cuộc họp để trao đổi với cả nhóm về những kiến thức mới, phù hợp yêu cầu đề bài để đẩy nhanh quá trình hoàn thiện bài tập lớn.
- Cùng với nhóm thực hiện demo bài tập thông qua họp nhóm trực tiếp hoặc ứng dụng Messenger video call..
- Đóng góp ý kiến, chỉnh sửa và bổ sung hoàn thiện các thiếu sót tồn đọng trong bài báo cáo.

CHƯƠNG III: KỊCH BẢN GIAO TIẾP

3.1. Nội dung khoa học của đề án:

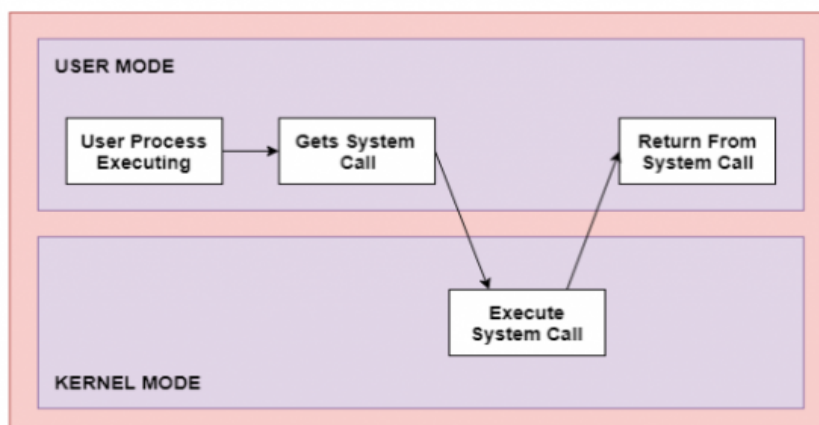
3.1.1 Tổng quan

a) Sơ lược về cơ chế hoạt động

Người dùng NachOS sẽ giao tiếp với nhân hệ điều hành thông qua các System Call. Các hàm System Call sẽ giao tiếp trực tiếp với nhân hệ điều hành và trả về kết quả cho người dùng. Thông qua system call, các lỗi chương trình không đáng có phát sinh từ phía người dùng sẽ được ngăn chặn. Tránh được các sự cố ảnh hưởng đến nhân hệ điều hành, thậm chí là phần cứng máy tính.

Cơ chế hoạt động:

- Các system call sẽ được khai báo bằng một hàm với tên gọi và các tham số/tham chiếu, tên gọi này sẽ được các chương trình người dùng sử dụng như một hàm thông thường. Đồng thời, hệ thống cũng sẽ phân biệt các system call bằng những *số nguyên dương* khác nhau. Tên hàm và các số nguyên dương sẽ được khai báo cũng như định nghĩa trong tập tin **syscall.h**.
- Khi các system call được gọi bởi chương trình người dùng. Một chương trình MIPS tên là **Start.S** sẽ nhận dạng và tiến hành ghi số nguyên tương ứng vào thanh ghi S2 để hệ thống ở tầng kernel nhận dạng được. Song song đó, các tham biến, tham chiếu cũng được ghi lần lượt vào các thanh ghi S4, S5, S6,...
- Kế đến, chương trình **exception.cc** sẽ tiến hành đọc giá trị từ các thanh ghi S2, S4, S5, S6..., sàng lọc các trường hợp lỗi và gọi các hàm xử lý tương ứng.
- Các hàm xử lý được **exception.cc** gọi sẽ được thiết kế để đáp ứng được yêu cầu đề án. Thiết kế chương trình được giới thiệu ở mục tiếp theo.



b) Khái quát về thiết kế chương trình

Phần code của đồ án được xây dựng hướng đối tượng và được thiết kế để có thể đáp ứng được 100% yêu cầu đồ án (kể cả phần “Advanced”).

Các phần sẽ được định nghĩa bởi các đối tượng riêng biệt:

- **NormalFile** là lớp để quản lý các file đọc ghi thông thường (phần 1)
- **Socket** là lớp dùng để quản lý các luồng socket client) (phần 2)
- **File** là lớp hợp nhất giữa 2 lớp **NormalFile** và **Socket** nhằm hoàn thành phần 3.
- **FileTable** là lớp ngoài cùng với các hàm quản lý và hỗ trợ truy xuất các đối tượng **File**.

3.1.2 Phần 1: Implement system call for File operations

- Phần này sẽ bao gồm các system calls: *Create, Open, Close, Read, Write, Seek, Remove*.
- Các system call hoạt động theo thiết kế của các đối tượng đã được định nghĩa bên trong tập tin **FileManager.h**, gồm: *NormalFile, File, Table*. Các đối tượng này lại hoạt động dựa trên các lớp sẵn có của NachOS, được định nghĩa bên trong các files : *filesys.h, kernel.h, openfile.h, sysdep.h*.
- Lớp **File** là đối tượng chứa các thuộc tính và các hàm xử lý của cả tập tin thông thường lẫn socket(sẽ được giới thiệu sau). Nói riêng về phần quản lý file, lớp này sẽ gồm các hàm hỗ trợ đóng, mở, đọc, ghi và dịch chuyển con trỏ văn bản. Ngoài ra, lớp **File** cũng hỗ trợ xác định các mode: *read-and-write* và *read-only* qua thuộc tính *type*.
- Các **File** sẽ mang một **File Descriptor**(FD) riêng biệt là một số nguyên. Ở đây, đối tượng **File** sẽ lưu giữ 2 FD khác nhau: 1 là **FileID**, 2 là **SocketID**. Ban đầu, **FileID** được tạo ra một cách tự động bởi NachOS. Tuy nhiên để tránh việc bị trùng lặp với **SocketID** và nhằm giúp cho các chương trình phía end-user hoạt động chính xác, chúng em đã tạo ra 1 ID mới để trả về phía người dùng. ID này sẽ là một số nguyên có 3 chữ số với công thức: $ID = MODE * 100 + (FileID \parallel SocketID)$. Từ đó, các file và socket sẽ tránh được trường hợp bị trùng File descriptor(FD) ở phía người dùng. Ví dụ: một file được mở với chế độ *read-only* có FD được NachOS tạo ra là 2. Vậy ID khi trả về phía người dùng sẽ là: 102 với 1 thể hiện chế độ *read-only* và 02 là FD của file này. Tương tự, một file mở với chế độ *read-and-write* với FD là 2 sẽ có ID là 2 (002).
- Lớp **Table** sẽ gồm một mảng 20 phần tử chứa các đối tượng **File**. Khi một tập tin được mở ra bởi system call **Open**, tập tin đó sẽ được quản lý dưới dạng con trỏ **OpenFile*** (lớp có sẵn của NachOS) và được bao hàm bởi lớp **File**.

Theo đó, một phần tử **File** sẽ được thêm vào một chỗ trống trong mảng của **Table**. Lớp Table sẽ không cho phép mở quá 20 file cùng một lúc.

3.1.3 Phần 2: Implement system call for Network operations

- Phần này bao gồm các system calls: *SocketTCP*, *Connect*, *Send*, *Receive* và *Close*.
- Các system call này sẽ hoạt động dựa trên các thư viện của C thay vì các lớp sẵn có trong NachOS (vì một số lỗi không xác định, các hàm này không thể hoạt động nhờ vào **sysdep.h**). Các thư viện C được sử dụng: *arpa/inet.h*, *stdio.h*, *string.h*, *sys/socket.h*, *unistd.h*.
- Hàm **SocketTCP** sẽ tiến hành tạo ra một socket với giao thức *TCP/IP* và *IPv4*. Socket này sẽ được thêm vào một chỗ trống trong Table để quản lý.
- Hàm **Connect** sẽ kết nối vào một server với địa chỉ IP và Port cho trước. Trong phạm vi đồ án, chúng em mặc định dùng một server socket chạy độc lập trên cùng một máy tính với NachOS. Do đó, địa chỉ IP mặc định sẽ là IP localhost "*127.0.0.1*" và port mặc định là *8081*.
- Các hàm *Send*, *Receive* và *Close* hoạt động như bình thường theo như yêu cầu.

3.1.4 Phần 3: Advanced

- Như yêu cầu của phần này, chúng em đã thiết kế đối tượng File và Table để gộp chung các file thông thường và các socket và một mảng duy nhất để quản lý. Theo đó, Table sẽ lưu giữ và quản lý 20 đối tượng File. Các đối tượng File này có thể là socket hoặc file thông thường.
- Song song đó, các hàm *Send*, *Receive* và *Close* cũng sẽ được lần lượt sáp nhập vào *Write*, *Read*, và *Close* của đối tượng File. Ở phía end-user sẽ chỉ cần gọi *Write* hoặc *Read* khi muốn gửi và nhận message với socket. Chỉ cần truyền vào ID của socket tương ứng là được.

3.1.5 Phần 4: Program Test

- Phần này, hầu hết chúng em chỉ "hardcode" chứ không thiết kế UI do không đủ thời gian. Các chương trình sẽ được chạy sẽ được chạy bằng lệnh: *../build.linux/nachos -x <program_name>*.
- Tất cả các chương trình đều hoạt động tốt. Trong thư mục Source cũng có sẵn một chương trình server đã được tinh chỉnh nhằm hoàn thành yêu cầu của các chương trình như: *concatenate*, *echoclient* và *fileclient*.

3.2. Môi trường lập trình và các Framework hỗ trợ:

Đồ án System calls & File - Network Operations được thực thi bằng ngôn ngữ lập trình C++, MIPS,... và được thực hiện trong file NachOS trên hệ điều hành Ubuntu 20.04 LTS (cài đặt trực tiếp, không thông qua máy ảo).

Đồ án có sử dụng trình biên dịch đúng theo yêu cầu:

- C/C++ Compiler: GCC 4.8/G++ 4.8
- Cross compiler: Decstation Ultrix

3.3. Hướng dẫn sử dụng và các tính năng của chương trình:

Để chạy được chương trình, trước tiên ta cần cài đặt một máy ảo chạy hệ điều hành Ubuntu. Sau đó vào file Nachos đã được lập trình sẵn yêu cầu đồ án đặt ra để thực hiện các yêu cầu.

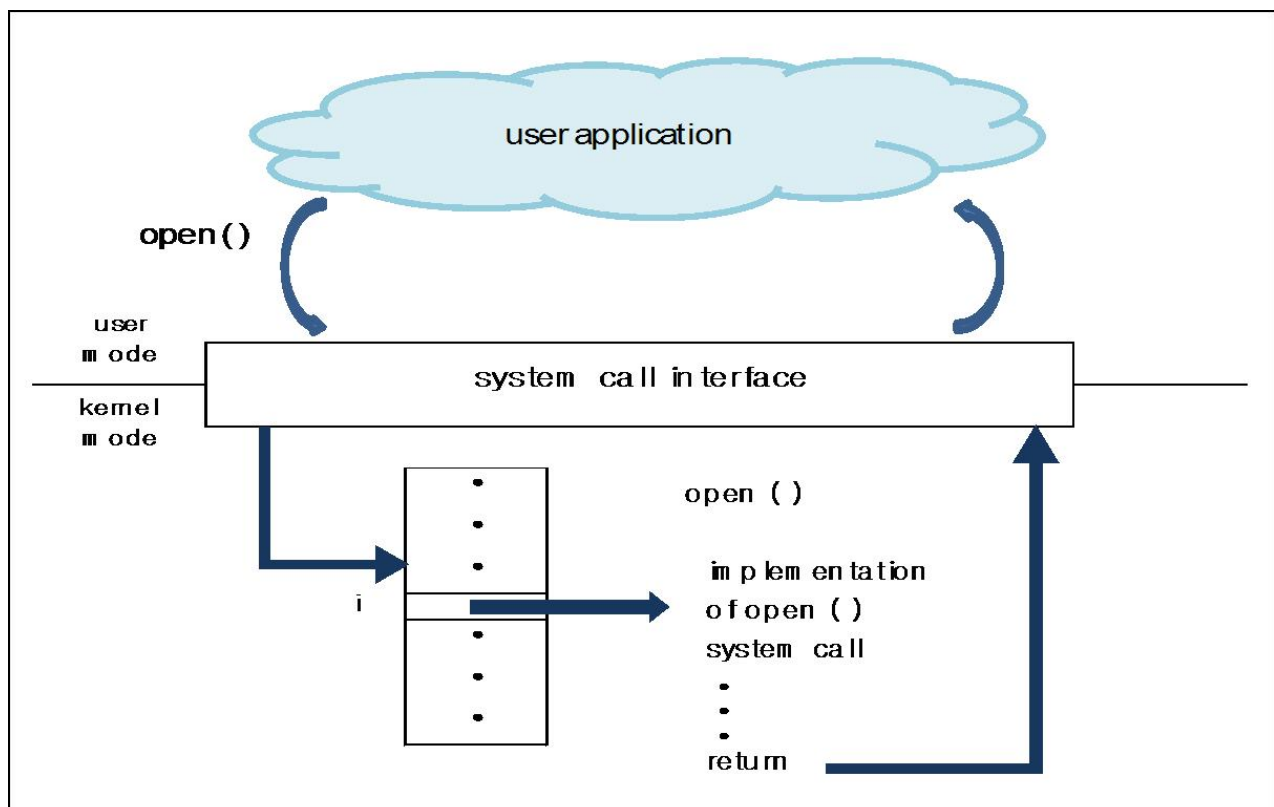
Chương trình chạy bằng lệnh command line.

Ta chỉ cần gõ các dòng lệnh Nachos khác nhau tùy vào yêu cầu mong muốn như: **tạo file, mở và đọc file, đọc và ghi file, sao chép hay xóa file**,.... vào chương trình thông qua màn hình command line, sau đó chương trình sẽ tự động nhận diện dòng lệnh và thực hiện yêu cầu và trả nội dung về máy.

CHƯƠNG IV: KẾT LUẬN

System call cho người dùng cách tiếp cận những tiện ích/ dịch vụ của hệ điều hành. Mọi phương thức của người dùng qua giao diện (GUI), tập lệnh (Batch) hay câu lệnh (command line, như cmd trong window) đều có cùng mục đích chung là gọi system call, để yêu cầu hệ điều hành thực hiện tác vụ cho mình. Mục tiêu của đề án này tập trung chủ yếu vào 2 vấn đề: lập trình **SYSTEMS CALL**, xây dựng giao thức trao đổi **SOCKET**. Ngoài ra, Đề án giúp sinh viên rèn luyện và nâng cao các khái niệm và cách thức hoạt động trong lĩnh vực Hệ Điều Hành.

Nhóm đã cơ bản hoàn thành yêu cầu và mục tiêu đề ra của đề án. Cơ bản hiểu được cách thức và các nguyên lý hoạt động của lập trình NachOs cũng như cách thức gọi một System Call, lập trình Socket,... Đồng thời cũng nắm được cách sử dụng NachOs chạy trên một máy ảo có hệ điều hành Ubuntu. Tạo nền tảng để học tập về các lĩnh vực liên quan khác trong thế giới Công nghệ thông tin.



CHƯƠNG V: PHỤ LỤC VÀ TÀI LIỆU THAM KHẢO

5.1. Phụ Lục:

5.1.1. Quá Trình Làm Việc và Biên Bản Họp Nhóm:

BIÊN BẢN HỌP NHÓM LẦN 1 NGÀY 4/3/2023

1. Thông tin chung

Các thành viên có tham dự:

STT	MSSV	Họ và tên	Email
1	21127041	Lý Nhật Hào	lnhao21@clc.fitus.edu.vn
2	21127500	Lê Văn Dương	lvduong21@clc.fitus.edu.vn
3	21127072	Nguyễn Hữu Khánh	nhkhanh21@clc.fitus.edu.vn

Các thành viên vắng mặt: **Không**

Mục tiêu cuộc họp nhằm:

- Nghiên cứu, cài đặt và lập trình, biên dịch được chương trình trong Nachos với Ubuntu thông qua máy ảo.
- Lên kế hoạch tìm kiếm tài liệu phân tích những ý chính của yêu cầu lập trình Nachos.

Địa điểm:

- Messenger (Video call)

Thời gian bắt đầu: 08:30 AM 4/3/2023.

Thời gian kết thúc: 05:00 PM 4/3/2023.



Nghỉ trưa 60' lúc: 11:30 AM 4/3/2023.

2. Kết quả buổi họp

- Hoàn thành tốt các yêu cầu và mục tiêu đặt ra từ ban đầu.
- Cơ bản phân chia được lộ trình làm việc cho đồ án và lên thiết kế cụ thể của bài báo cáo đồ án.
- Nhóm cùng nhau làm việc để cơ bản hoàn thành một số kiến thức đơn giản và đi đến những thống nhất chung.

3. Bảng phân công công việc

STT	Họ và tên	Phân công	Đánh giá	Ngày kết thúc
-----	-----------	-----------	----------	---------------

1	Lý Nhật Hào	<ul style="list-style-type: none"> - Thành lập biên bản họp nhóm. - Thiết kế bìa bài báo cáo đồ án. - Soạn thảo bản báo cáo đồ án với đầy đủ các đề mục theo yêu cầu. - Tải về máy ảo, cài đặt Ubuntu và Nachos. - Nghiên cứu, cài đặt cùng nhóm về lập trình Nachos để nắm bắt được tình hình hoạt động của nhóm đồ án. - Gửi kết luận lên group Messenger của nhóm để dễ dàng làm việc cùng nhau. 		11:00 PM 4/3/2023
2	<p>Nguyễn Hữu Khánh</p> <p>Lê Văn Dương</p>	<ul style="list-style-type: none"> - Tải về máy ảo, cài đặt Ubuntu và Nachos. - Nghiên cứu, lập trình và biên dịch thử các yêu cầu đơn giản trong lập trình Nachos. - Tìm hiểu về lập trình một Syscall Call và Socket, các kiến thức để hoàn thành đồ án từ đó 		11:00 PM 4/3/2023

		<p>nhận định thể mạnh để chia công việc có thể tạo ra hiệu suất cao.</p> <ul style="list-style-type: none"> - Tìm kiếm thêm các thông tin liên quan đến đề án thông qua internet. - Gửi kết luận lên group Messenger của nhóm để nhóm dễ dàng làm việc cùng nhau. - Gửi kết luận lên group Messenger của nhóm để dễ dàng làm việc cùng nhau. 		
--	--	---	--	--

BIÊN BẢN HỌP NHÓM LẦN 2 NGÀY 20/3/2022

1. Thông tin chung

Các thành viên có tham dự:

STT	MSSV	Họ và tên	Email
1	21127041	Lý Nhật Hào	lnhao21@clc.fitus.edu.vn
2	21127500	Lê Văn Dương	lvduong21@clc.fitus.edu.vn
3	21127072	Nguyễn Hữu Khánh	nhkhanh21@clc.fitus.edu.vn

Các thành viên vắng mặt: Không

Mục tiêu cuộc họp nhằm:

- Tổng hợp, thống nhất tất cả kết quả, source code, tài liệu và hình ảnh làm được để chuẩn bị hoàn thành đồ án.
- Hoàn thành những phần nâng cao của đồ án để nhận được điểm cao.
- Chỉnh sửa, hoàn thiện bài báo cáo lần cuối từ những kết quả tìm được.

Địa điểm:

- Tại Nhà riêng của một thành viên trong nhóm, chung cư Topaz City quận 8.

Thời gian bắt đầu: 08:30 AM **20/3/2022**.

Thời gian kết thúc: 03:30 PM **20/3/2022**.

Nghỉ trưa 60' lúc: 12:30 AM **20/3/2022**.

2. Kết quả buổi họp

- Hoàn thành nhiệm vụ được đề ra trong đồ án lập trình Nachos về việc viết một System call đơn giản và Socket, có làm được các phần nâng cao và bài báo cáo được hoàn thành tốt đẹp.
- Cả nhóm cùng nhau chạy demo tất cả các phần công việc được giao lần cuối, kiểm tra các phần nâng cao, kiểm tra lỗi để kết thúc đồ án.

3. Bảng phân công công việc

STT	Họ và tên	Phân công	Đánh giá	Ngày kết thúc
-----	-----------	-----------	----------	---------------

1	Lý Nhật Hào	<ul style="list-style-type: none"> - Thành lập biên bản họp nhóm. - Cùng với nhóm nghiên cứu những phần nâng cao trong đề án với mong muốn đạt được điểm cao hơn. - Chỉnh lý, bổ sung lần cuối cùng cho bài báo cáo dựa trên những kết quả cả nhóm tìm được trong suốt cả 2 cuộc họp lớn của nhóm. - Trình bày bài báo cáo chi tiết cho nhóm, thuyết trình sơ bộ về những ý chính và quan trọng để nhóm nắm cơ bản những gì được trình bày đến giảng viên và đóng góp ý kiến. - Gửi kết luận lên group Messenger của nhóm để dễ dàng lưu trữ. - Trích xuất nguồn tài liệu tham khảo cho bài tập của mình 		11:00 PM 20/3/2022
2	Nguyễn Hữu Khánh Lê Văn Dương	<ul style="list-style-type: none"> - Trình bày lại tất cả những gì đã tìm được để hoàn thiện chi tiết đề án và thực hiện các phần nâng cao giúp nhóm đi đến thống nhất cuối cùng. - Code hoàn chỉnh hoàn toàn 100% đề án và biên dịch thành công. 		11:00 PM 20/3/2022

		<ul style="list-style-type: none">- Thu thập kết quả, số liệu và hình ảnh trong quá trình biên dịch chương trình để làm tư liệu cho bài báo cáo được trực quan dễ hiểu..-Trích xuất nguồn tài liệu tham khảo cho bài tập của mình.- Gửi tất cả file lên group để tổng hợp, thống nhất lần cuối trước khi nén file và kết thúc đồ án.		
--	--	--	--	--

5.2. Tài Liệu Tham Khảo:

- Giao thức Udp, cấu trúc header, truyền dữ liệu | Tự học ICT (tuhocict.com)

- [Tìm hiểu giao thức TCP và UDP \(viblo.asia\)](http://viblo.asia)
- [Sử dụng lệnh Nslookup \(blogcafeit.com\)](http://blogcafeit.com)
- <https://viblo.asia/p/doc-hieu-va-phan-tich-cac-thong-so-cua-report-trong-jmeter-gGJ59NprKX2>
- <https://vietnix.vn/tracert-la-gi/>
- <https://viblo.asia/p/phan-biet-router-switch-va-hub-mang-may-tinh-bJzKm0j659N>
- <https://www.totolink.vn/article/632-arp-la-gi-muc-dich-va-cach-thuc-hoat-dong-cua-arp.html>