



ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

VNUHCM - UNIVERSITY OF SCIENCE

Hệ Điều Hành

Đồ án 1 – Syscall for File System

BÁO CÁO

Giảng viên

Phạm Tuấn Sơn

Lê Viết Long

Mục Lục

Contents

1	THÔNG TIN THÀNH VIÊN	3
2	CÀI ĐẶT SYSTEM CALL VÀ EXCEPTION	4
2.1	Cài đặt lại các exception	4
2.2	Cài đặt System Call Create	4
2.3	Cài đặt System Call Open và System Call Close	4
2.4	Cài đặt System Call Read và System Call Write	6
2.5	Cài đặt System Call Seek	7
2.6	Cài đặt System Call Delete	8
2.7	Chương trình createfile để kiểm tra System Call CreateFile	8
2.8	Chương trình echo	8
2.9	Chương trình cat	9
2.10	Chương trình copy	9
2.11	Chương trình delete	10
3	DEMO HÌNH ẢNH MINH HỌA	11
3.1	Biên dịch chương trình (cat.c)	11
3.2	Chương trình createfile	12
3.3	Chương trình echo	12
3.4	Chương trình cat	13
3.5	Chương trình copy	13
3.6	Chương trình delete	14
	TÀI LIỆU THAM KHẢO	14

1 THÔNG TIN THÀNH VIÊN

MSSV	Họ tên	Đóng góp
20127577	Phan Nguyễn Phước Nguyên	100%
20127584	Trần Hữu Minh Nhật	100%
20127349	Trần Quốc Thuận	100%

2 CÀI ĐẶT SYSTEM CALL VÀ EXCEPTION

2.1 Cài đặt lại các exception

Vào thư mục ./code/machine vào file “machine.h” ta có danh sách các exception được liệt kê, ta qua file exception.cc viết lại các case này theo các ExceptionType mà tắt bắt được. Mỗi exception ta thêm lệnh interrupt->Halt() để tắt hệ điều hành.

2.2 Cài đặt System Call Create

CreateFile system call sẽ sử dụng **Nachos FileSystem Object** để tạo một file rỗng. Ban đầu chuỗi name đang ở trong user space, do đó buffer phải lưu lại giá trị chuỗi name này trong system space. **System call CreateFile** trả về 0 nếu thành công và -1 nếu có lỗi.

int Create (char* name)

Mô tả cài đặt SC Create:

- Input: Địa chỉ chứa tên file ở User Space
- Output: -1 lỗi, 0 thành công.
- Mục đích: tạo ra file rỗng với tham số là tên file.

Ta đọc địa chỉ của tham số **name** từ thanh ghi **r4**, sau đó thực hiện chép giá trị ở **r4** từ vùng nhớ User sang System bằng hàm **User2System()**. Giá trị chép được thực sự chính là tên file. Ta tiếp tục kiểm tra tên file có NULL không và file có được tạo ra với tên file đó không. Nếu thành công thì trả về 0, ngược lại thì trả về -1 vào thanh ghi **r2**

2.3 Cài đặt System Call Open và System Call Close

Đồ án 1 – Syscall for File System

Người dùng chương trình có thể mở 2 loại file đó là chỉ đọc và đọc&ghi. Mỗi tiến trình sẽ được cấp 1 bảng mô tả file với kích thước cố định.

System call gọi file cần được thực thi bằng cách chuyển đổi địa chỉ buffer trong user space khi cần thiết và viết hàm xử lý phù hợp trong kernel. Dùng class **FileSystem** trong thư mục **filesys**, System call Open trả về ID của file hoặc là -1 nếu xảy ra lỗi

*Quy ước giá trị của type:

-Type = 0 : đọc và ghi

-Type = 1 : chỉ đọc

-Type = 2 : stdin

-Type = 3: stdout

int Open (char* name, int type)

Mô tả cài đặt SC Open:

- Input: Địa chỉ của tên file trên user space, chế độ mở (type)
- Output: ID file nếu thành công, -1 nếu lỗi
- Mục đích: Mở 1 file với tham số truyền vào gồm tên file và cách mở

Đọc tham số ID của file từ thanh ghi r4, sau đó kiểm tra xem file cần đóng có tồn tại không bằng open[id] đã cài đặt trong lớp FileSystem và kiểm tra ID của file có nằm ngoài mô tả file không. Nếu có 1 trong 2 lỗi trên thì xuất ra thông báo lỗi và gọi syscall Halt() để tắt hệ thống, nếu thành công thì xóa đi dữ liệu open[id] và gán lại b

int Close (Open id)

Mô tả cài đặt SC Close:

- Input: ID file
- Output: Không có (null)
- Mục đích: Đóng file với tham số truyền vào là ID file cần đóng

2.4 Cài đặt System Call Read và System Call Write

Các System call đọc và ghi vào file với ID cho trước. Phải chuyển vùng nhớ giữa user space và system space, cần phân biệt giữa **Console IO** (OpenFileID 0, 1) và File. Lệnh Read và Write sẽ làm việc như sau: Phần console read và write sẽ sử dụng lớp **SynchConsole**. Được khởi tạo qua biến toàn cục **gSynchConsole**.

int Read (char* buffer, int charcount, Open id)

Mô tả cài đặt SC Read:

- Input: Buffer, số ký tự cho phép, id của file.
- Output: -1 nếu lỗi, -2 nếu thành công với số byte được đọc.
- Mục đích: Đọc file với tham số là buffer, số ký tự cho phép (charcount) và id của file.

Sử dụng các hàm mặc định của **SynchConsole** để đọc và ghi. Đọc và ghi với **Console** sẽ trả về số bytes đọc và ghi thật sự, chứ không phải số bytes được yêu cầu. Trong trường hợp đọc hay ghi vào console bị lỗi thì trả về -1. Nếu đang đọc từ **console** và chạm tới cuối file thì trả về -2. Phần đọc, ghi vào file sẽ sử dụng các lớp được cung cấp trong file system. Sử dụng các hàm mặc định có sẵn của FileSystem và thông số trả về cũng phải giống như việc trả về trong **synchconsole**. Cả read và write trả số ký tự đọc, ghi thật sự. Cả Read và

Write trả về -1 nếu bị lỗi và -2 nếu cuối file. Cả Read và Write sử dụng dữ liệu binary.

int Write (char* buffer, int charcount, Open id)

Mô tả cài đặt SC Write:

- Input: Buffer, số ký tự cho phép, id của file.
- Output: -1 nếu lỗi, Số byte thực sự ghi được nếu thành công.
- Mục đích: Ghi file với tham số là buffer, số ký tự cho phép (charcount) và id của file.

Ta đọc địa chỉ của tham số buffer từ thanh ghi **r4**, tham số **charcount** từ thanh ghi **r5** và **id** của file từ thanh ghi **r6**, sau đó ta tiến hành kiểm tra id của file truyền vào có nằm ngoài bảng mô tả file không, file cần ghi có tồn tại không và file cần ghi có phải là stdin với type = 2 hay là file chỉ đọc với type = 1. Nếu vi phạm các điều kiện trên thì trả về -1 cho thanh ghi **r2** ngược lại là hợp lệ thì lấy vị trí con trỏ ban đầu trong file bằng phương thức **GetCurrentPos()** của lớp **FileSystem** gọi là **OldPos** và thực hiện chép giá trị ở **r4** từ phía User sang System bằng hàm **User2System()**. Giá trị chép được là **buffer** chứa chuỗi ký tự. Xét trường hợp ghi file đọc và ghi với type = 0, thì ta lấy vị trí con trỏ hiện tại trong file bằng phương thức **GetCurrentPos()** của lớp **FileSystem** gọi là

NewPos, trả về số byte thực sự ghi được cho thanh ghi **r2** bằng công thức: **NewPos – OldPos**. Xét trường hợp ghi file stdout với type = 3, ta gọi phương thức Write của lớp **SynchConsole** để ghi từng ký tự trong buffer và kết thúc là ký tự xuống dòng '\n', trả về số byte thực sự ghi được cho thanh ghi **r2**.

2.5 Cài đặt System Call Seek

Seek sẽ phải chuyển con trỏ tới vị trí thích hợp. **Pos** lưu vị trí cần chuyển tới, nếu $pos = -1$ thì di chuyển đến cuối file. Trả về vị trí thực sự trong file nếu thành công và -1 nếu bị lỗi. Gọi Seek trên console phải báo lỗi.

Mô tả cài đặt SC Seek:

- Input: Vị trí cần chuyển tới, id của file.
- Output: -1: Lỗi, Vị trí thực sự trong file: Thành công.
- Mục đích: Di chuyển con trỏ đến vị trí thích hợp trong file với tham số là vị trí cần dịch chuyển và id của file.

2.6 Cài đặt System Call Delete

2.7 Chương trình createfile để kiểm tra System Call CreateFile

Dùng tên file cố định hoặc cho người dùng nhập vào từ console.

Ta gọi lại system call Open để mở file stdin với type quy ước bằng 2. Nếu mở file thành công thì gọi system call Read đọc tên file vừa nhập từ stdin và gọi system call CreateFile để tạo file với tham số truyền vào là tên file đọc được. Cuối cùng là đóng file stdin với system call Close.

2.8 Chương trình echo

Echo cho phép khi nhập một dòng từ console thì console xuất lại dòng đó.

Ta gọi lại system call Open để mở file stdin với type quy ước bằng 2 và stdout với type quy ước bằng 3. Nếu mở thành công thì gọi system call Read đọc nội dung nhập từ stdin vào buffer đồng thời trả về số byte thực sự đọc được, sau đó gọi system call Write để ghi nội dung vừa đọc từ buffer ra stdout với charcount bằng số byte thực sự đọc được ở trên. Cuối cùng là đóng file stdin và file stdout với system call Close.

2.9 Chương trình **cat**

Chương trình cat yêu cầu nhập vào filename rồi hiển thị nội dung của file đó.

Ta gọi lại system call ReadString để nhập vào filename. Sau đó gọi system

call Open để mở file đó với type bằng 1, nếu mở file thành công thì gọi system call Seek để dịch chuyển con trỏ về cuối file lấy kích thước thực sự của file và di chuyển lại con trỏ ra đầu file. Tiến hành đọc từng kí tự trong file bằng system call Read và in từng kí tự đó ra màn hình bằng system call PrintChar cho đến hết kích thước của file. Cuối cùng là đóng file với system call Close.

2.10 Chương trình **copy**

Copy yêu cầu nhập tên file nguồn và file đích và thực hiện copy nội dung từ file nguồn sang file đích.

Ta gọi lại system call ReadString để nhập vào tên file nguồn và tên file đích. Sau đó gọi system call Open để mở file nguồn với type bằng 1 và file đích với type bằng 0, id của file nguồn và file đích phải khác nhau để tránh trường hợp người dùng mở cùng một file. Nếu mở file thành công thì gọi system call Seek để trỏ đến cuối file nguồn và lấy kích thước thực sự của file nguồn, di chuyển con trỏ trở lại đầu file nguồn và đầu file đích. Tiến hành đọc từng kí tự từ đầu file nguồn bằng system call Read và ghi từng kí tự đó ra file đích bằng system call Write cho đến hết kích thước của file nguồn. Cuối cùng là đóng file nguồn và file đích với system call Close.

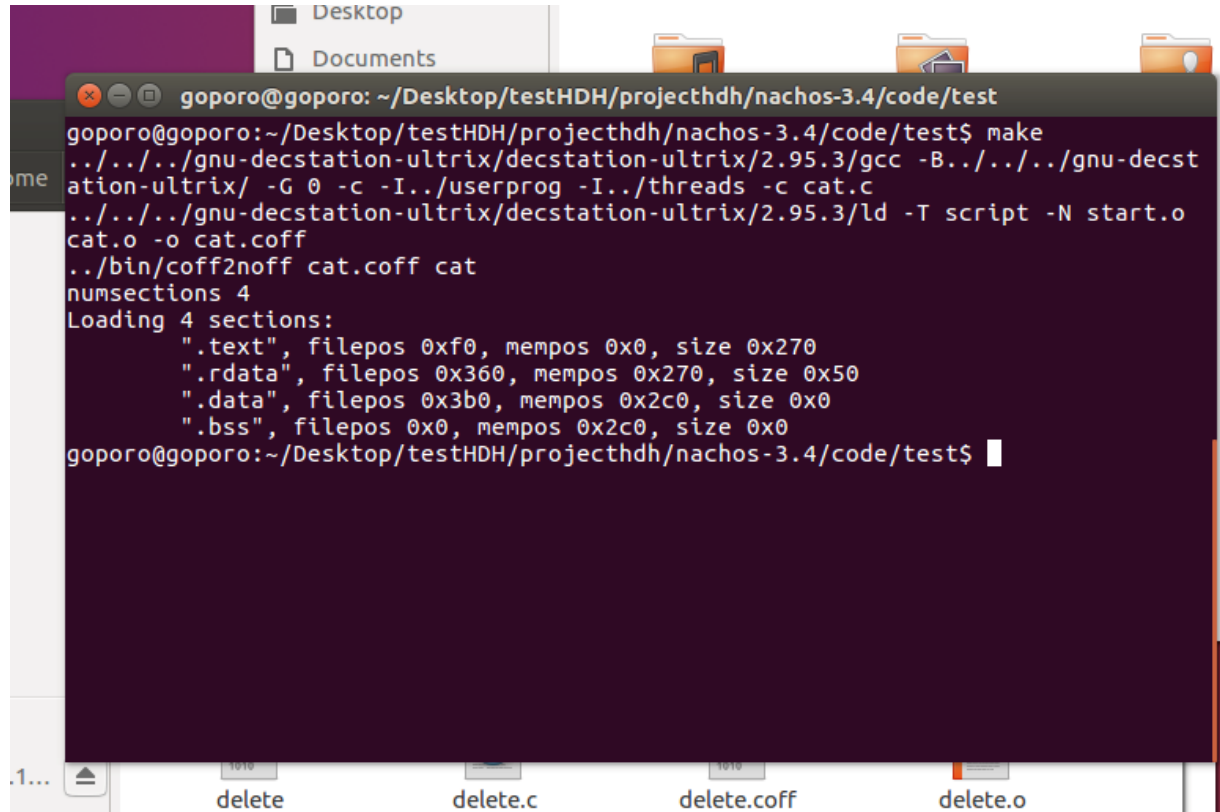
2.11 Chương trình `delete`

Delete yêu cầu người dùng nhập filename để xóa file

Ta gọi lại system call `ReadString` để nhập tên filename. Sau đó `SysCal_Delete` dựa theo buffer đến thư mục chứa file để unlink ra khỏi thanh ghi. Cuối cùng là đóng file `stdin` và file `stdout` với system call `Close`.

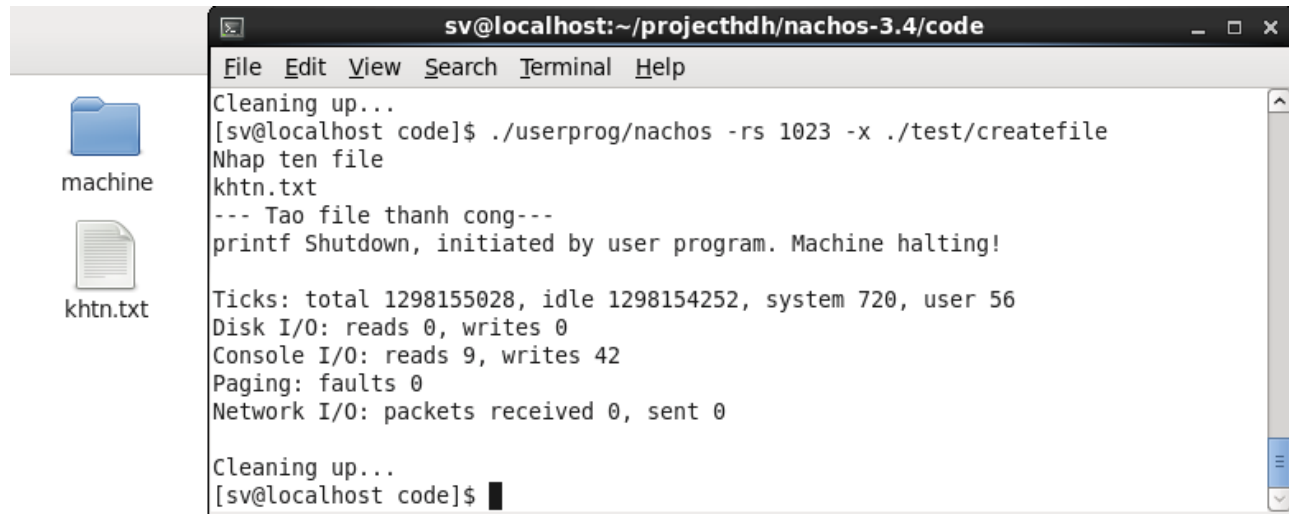
3 DEMO HÌNH ẢNH MINH HỌA

3.1 Biên dịch chương trình (cat.c)



```
goporo@goporo: ~/Desktop/testHDH/projectthdh/nachos-3.4/code/test
goporo@goporo:~/Desktop/testHDH/projectthdh/nachos-3.4/code/test$ make
../../../../gnu-decstation-ultrix/decstation-ultrix/2.95.3/gcc -B../../../../gnu-decstation-ultrix/ -G 0 -c -I../userprog -I../threads -c cat.c
../../../../gnu-decstation-ultrix/decstation-ultrix/2.95.3/ld -T script -N start.o cat.o -o cat.coff
../bin/coff2noff cat.coff cat
numsections 4
Loading 4 sections:
  ".text", filepos 0xf0, mempos 0x0, size 0x270
  ".rdata", filepos 0x360, mempos 0x270, size 0x50
  ".data", filepos 0x3b0, mempos 0x2c0, size 0x0
  ".bss", filepos 0x0, mempos 0x2c0, size 0x0
goporo@goporo:~/Desktop/testHDH/projectthdh/nachos-3.4/code/test$
```

3.2 Chương trình createfile



The screenshot shows a terminal window titled "sv@localhost:~/projectthdh/nachos-3.4/code". The terminal output is as follows:

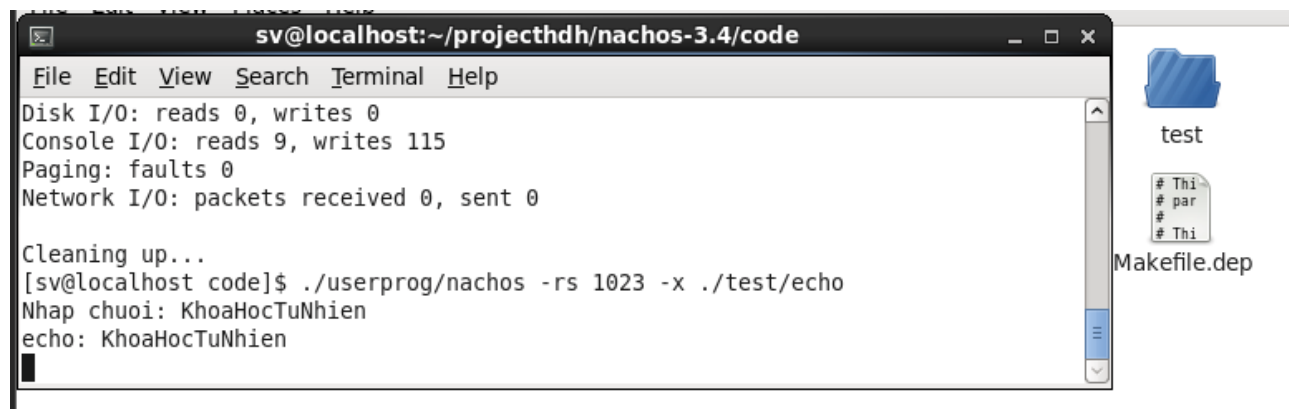
```
File Edit View Search Terminal Help
Cleaning up...
[sv@localhost code]$ ./userprog/nachos -rs 1023 -x ./test/createfile
Nhap ten file
khtn.txt
--- Tao file thanh cong---
printf Shutdown, initiated by user program. Machine halting!

Ticks: total 1298155028, idle 1298154252, system 720, user 56
Disk I/O: reads 0, writes 0
Console I/O: reads 9, writes 42
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
[sv@localhost code]$
```

On the left side of the terminal window, there is a file explorer showing a folder named "machine" and a file named "khtn.txt".

3.3 Chương trình echo



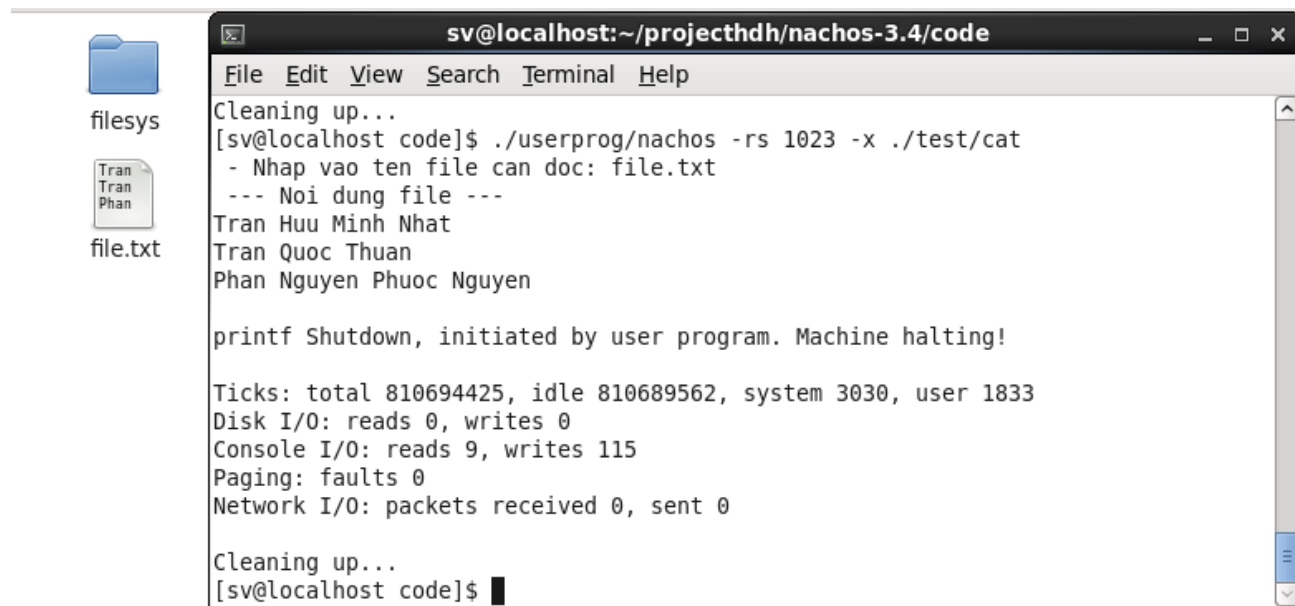
The screenshot shows a terminal window titled "sv@localhost:~/projectthdh/nachos-3.4/code". The terminal output is as follows:

```
File Edit View Search Terminal Help
Disk I/O: reads 0, writes 0
Console I/O: reads 9, writes 115
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
[sv@localhost code]$ ./userprog/nachos -rs 1023 -x ./test/echo
Nhap chuoi: KhoaHocTuNhiem
echo: KhoaHocTuNhiem
```

On the right side of the terminal window, there is a file explorer showing a folder named "test" and a file named "Makefile.dep".

3.4 Chương trình cat



The screenshot shows a file manager on the left with a folder named 'filesys' containing a file 'file.txt' with the text 'Tran Tran Phan'. To the right is a terminal window titled 'sv@localhost:~/projectthdh/nachos-3.4/code'. The terminal output shows the execution of './userprog/nachos -rs 1023 -x ./test/cat', which prompts for a filename ('file.txt') and displays its contents: 'Tran Huu Minh Nhat', 'Tran Quoc Thuan', and 'Phan Nguyen Phuoc Nguyen'. It then prints a shutdown message and system statistics: 'Ticks: total 810694425, idle 810689562, system 3030, user 1833', 'Disk I/O: reads 0, writes 0', 'Console I/O: reads 9, writes 115', 'Paging: faults 0', and 'Network I/O: packets received 0, sent 0'. The terminal ends with 'Cleaning up...' and a prompt '[sv@localhost code]\$'.

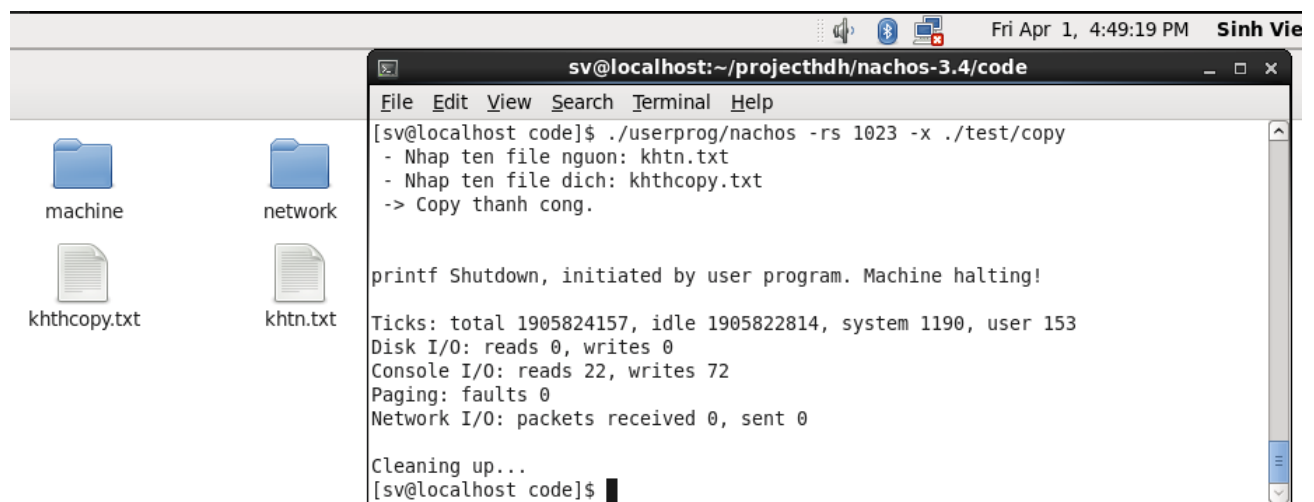
```
sv@localhost:~/projectthdh/nachos-3.4/code
File Edit View Search Terminal Help
Cleaning up...
[sv@localhost code]$ ./userprog/nachos -rs 1023 -x ./test/cat
- Nhap vao ten file can doc: file.txt
--- Noi dung file ---
Tran Huu Minh Nhat
Tran Quoc Thuan
Phan Nguyen Phuoc Nguyen

printf Shutdown, initiated by user program. Machine halting!

Ticks: total 810694425, idle 810689562, system 3030, user 1833
Disk I/O: reads 0, writes 0
Console I/O: reads 9, writes 115
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
[sv@localhost code]$
```

3.5 Chương trình copy



The screenshot shows a file manager on the left with folders 'machine' and 'network', and files 'khthcopy.txt' and 'khtn.txt'. To the right is a terminal window titled 'sv@localhost:~/projectthdh/nachos-3.4/code'. The terminal output shows the execution of './userprog/nachos -rs 1023 -x ./test/copy', which prompts for source ('khtn.txt') and destination ('khthcopy.txt') filenames and confirms the copy. It then prints a shutdown message and system statistics: 'Ticks: total 1905824157, idle 1905822814, system 1190, user 153', 'Disk I/O: reads 0, writes 0', 'Console I/O: reads 22, writes 72', 'Paging: faults 0', and 'Network I/O: packets received 0, sent 0'. The terminal ends with 'Cleaning up...' and a prompt '[sv@localhost code]\$'.


```
sv@localhost:~/projectthdh/nachos-3.4/code
File Edit View Search Terminal Help
[sv@localhost code]$ ./userprog/nachos -rs 1023 -x ./test/copy
- Nhap ten file nguon: khtn.txt
- Nhap ten file dich: khthcopy.txt
-> Copy thanh cong.

printf Shutdown, initiated by user program. Machine halting!

Ticks: total 1905824157, idle 1905822814, system 1190, user 153
Disk I/O: reads 0, writes 0
Console I/O: reads 22, writes 72
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
[sv@localhost code]$
```

3.6 Chương trình delete



The screenshot shows a terminal window titled "sv@localhost:~/projecthhdh/nachos-3.4/code". The terminal output is as follows:

```
File Edit View Search Terminal Help
Cleaning up...
[sv@localhost code]$ ./userprog/nachos -rs 1023 -x ./test/delete
Nhap ten file can xoa
khtn.txt
Open file aa for testing

--- Xoa file thanh cong ---
can be delete
printf Shutdown, initiated by user program. Machine halting!

Ticks: total 695002832, idle 695001643, system 1120, user 69
Disk I/O: reads 0, writes 0
Console I/O: reads 13, writes 78
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
[sv@localhost code]$
```

TÀI LIỆU THAM KHẢO

- File trong tài liệu hướng dẫn thực hành
- <https://github.com/nguyenthanhchungfit/Nachos-Programing-HCMUS>