CS342301: Operating System
MP1: System Call

Deadline: 2017/10/16 08:00

I. Goal

1. Understand how to work under Linux platform.
2. Understand how system calls are done by OS.
3. Understand the difference of user space and kernel space memory.

II. Assignment

1. Implement system call in NachOS
   (a). Part I: Implement a console I/O system call.

   ```
   void PrintInt(int number)
   ```

   Output the number and a line separator to the console.

   (b). Part II: Implement four file I/O system call.

   ```
   OpenFileId Open(char *name);
   ```

   Open a file with the name, and returns its corresponding OpenFileId.

   Return -1 if open fails.

   ```
   int Write(char *buffer, int size, OpenFileId id);
   ```

   Write “size” characters from buffer into the file.

   Returns number of characters actually written to the file.

   Return -1, if attempt writing to an invalid id.

   ```
   int Read(char *buffer, int size, OpenFileId id);
   ```

   Read “size” characters from the file and copy them into buffer.

   Returns number of characters actually read from the file.

   Return -1, if attempt reading from an invalid id.

   ```
   int Close(OpenFileId id);
   ```

   Close the file with id.

   Return 1 if successfully close the file. Otherwise, return 0.

＊ All your implements should not use any IO functions from standard libraries (e.g. printf(), cout, fopen(), fwrite(), write(), etc.).

2. Write homework report
   (a). Cover page

   Team members, Team member contribution

   (b). Trace System Call

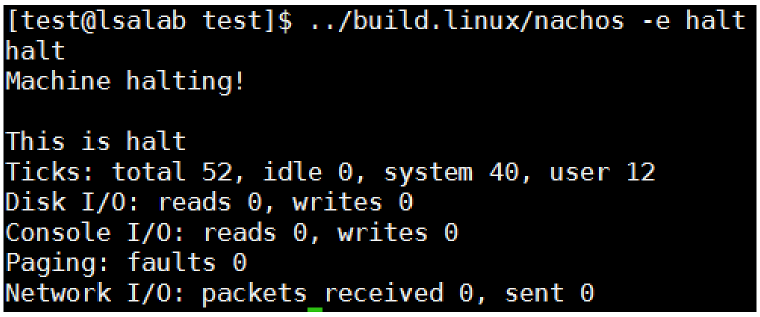
   Explain how system calls go through Nachos **in detail.**

   (c). Explain your implementation

   Modification of the code

III. Instruction

1. Login server
   - 140.114.78.227 port:22
   - Username: 2017osteam + your teamID (e.g. 2017osteam01)
   - Passwd: You will be ask to set up your password once you login
2. Install NachOS
   - cp –r /home/os2017/shared/NachOS-4.0_MP1 .
   - cd NachOS-4.0_MP1/code/build.linux
   - make clean
   - make
3. Compile/Rebuild NachOS
   - cd NachOS-4.0_MP1/code/build.linux
   - make clean
   - make
4. Test NachOS
   - cd NachOS-4.0_MP1/code/test
   - make clean
   - make halt
   - ../build.linux/nachos -e halt



```
[test@lsalab test]$ ../build.linux/nachos -e halt
halt
Machine halting!

This is halt
Ticks: total 52, idle 0, system 40, user 12
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0
```

IV. Grading

1. Implementation correctness – 60%
   (a). Console I/O system call – 20%
   (b). File I/O system call – 40%
2. Trace system call – 25%
   (a). Explain how system calls work in detail (ex. Halt() , Create())
   (b). Filename: MP1_trace_[yourGroupNumber].pdf
3. Report – 15%
   (a). Explain your work (modifications of the code, team member contribution, …)
   (b). Filename: MP1_code_[yourGroupNumber].pdf

V. Hint

1. Implementation Part I
   - Trace how Halt() system call works, this will help you a lot.
   - Do not trace Add() system call, this is not a console IO system call.
   - Files to modify
     - ➢ userprog/syscall.h, exception.cc, ksyscall.h, synchconsole
     - ➢ machine/console, interrupt
     - ➢ test/start.S
     - ➢ threads/kernel

2. Implementation Part II
   - Trace how Create() system call works, this will help you a lot.
   - Pay attention to the return value.
   - Files to modify
     - ➢ userprog/syscall.h, exception.cc, ksyscall.h
     - ➢ machine/interrupt
     - ➢ filesys/filesys, openfile
     - ➢ threads/kernel

VI. Reminder

1. iLMS
   (a). Upload your report files in **PDF** format to iLMS.
   (b). You **DO NOT** need to upload NachOS code to iLMS, but we will use your latest modification time as your submission time.
2. Demo policy
   (a). Demo will take place on our server.
   (b). You are responsible to make sure your code works on our server.
   (c). **Limit 10 mins** for each team, so please be well prepared for it.
3. Refer to syllabus for late submission penalty.
4. 0 will be given to cheaters.
5. More details can be found from the homework slides.