NachOS-MP4

Lecturer: Jerry Chou
TA: Angus Lin, Jerry Tung
National Tsing Hua University

Outline

- Compilation flag
- Nachos file system
- Assignments & Bonus
- Sample Testcases
- Grading
- Hint

Before starting

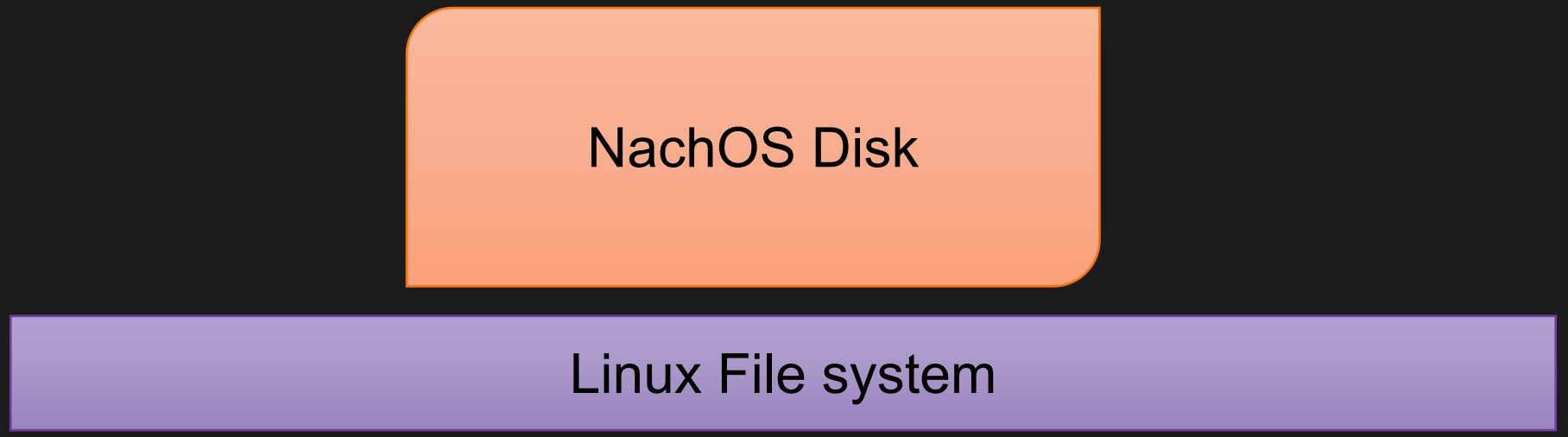
- Please copy /home/os2017/shared/NachOS-4.0_MP4 into your home directory
- Since we have updated some codes in the original NachOS, please DO NOT use your Nachos from MP1~MP3

Compilation Flag

- Note that in build.linux/Makefile
  - originally, `FILESYS_STUB` is defined during compilation (for MP1~3)
  - In MP4, we DO NOT define this macro (line 195 in the Makefile)
  - make clean and rebuild NachOS

- Now we are using Nachos native filesystem
  - filesys.cc, filehdr.cc, openfile.cc, …

NachOS File System



- NachOS simulates a raw disk on top of host file system
- The disk is just a file named “DISK_0” in the host file system

NachOS File System

- Basic commands
- `>../build.linux/nachos -f # format the disk`
- `>../build.linux/nachos -cp <file_on_LINUX> <path_on_NachOS> # copy a file to nachos`
- `>../build.linux/nachos -p <file_on_NachOS> # dump the content of the file`
- for more commands, refer to main.cc line226~275

Example

- >../build.linux/nachos -f
- >../build.linux/nachos -cp num_100.txt 100
- >../build.linux/nachos -l /
- >../build.linux/nachos -p 100

```
[test@lsalab test]$ ../build.linux/nachos -l /
100
[test@lsalab test]$ ../build.linux/nachos -p 100
000000001 000000002 000000003 000000004 000000005 000000006 000000007 000000008 000000009 000000010
000000011 000000012 000000013 000000014 000000015 000000016 000000017 000000018 000000019 000000020
000000021 000000022 000000023 000000024 000000025 000000026 000000027 000000028 000000029 000000030
000000031 000000032 000000033 000000034 000000035 000000036 000000037 000000038 000000039 000000040
000000041 000000042 000000043 000000044 000000045 000000046 000000047 000000048 000000049 000000050
000000051 000000052 000000053 000000054 000000055 000000056 000000057 000000058 000000059 000000060
000000061 000000062 000000063 000000064 000000065 000000066 000000067 000000068 000000069 000000070
000000071 000000072 000000073 000000074 000000075 000000076 000000077 000000078 000000079 000000080
000000081 000000082 000000083 000000084 000000085 000000086 000000087 000000088 000000089 000000090
000000091 000000092 000000093 000000094 000000095 000000096 000000097 000000098 000000099 000000100
```

NachOS File System

- Disk can be viewed as a huge array, each element with several bytes and labeled with a sector number.
- each sector with size of 128 bytes

0	1	2	3	…					

NachOS File System

- NachOS file system maintains the following information
  - A free data block information
  - A directory information
  - A set of inodes, which record the information of a file
  - A set of data blocks, which store the actual data content of a file.
- All above information must be stored on the Nachos raw disk, and each of them must fit in the size of a SINGLE sector

MP4 - Assignments

- Part I: Trace the file system call and answer the following questions:
  1. Explain how does the NachOS FS manage and find free block space? Where is this information stored on the raw disk (which sector)?
  2. What is the maximum disk size can be handled by the current implementation, explain why?
  3. Explain how does the NachOS FS manage the directory data structure? Where is this information stored on the raw disk (which sector)?
  4. Explain what information is stored in an inode, and use a figure to illustrate the disk allocation scheme of current implementation.
  5. Why a file is limited to 4KB in the current implementation?

MP4 - Assignments

- Part II: Modify file system code to support large file
  - Limitations: you ARE NOT ALLOWED to modify sector size in disk.h
  - Goal:
    - Combine your MP1 file system call interface with Nachos file system, including file read, write and open.
    - Support up to 32KB file
  - Approach:
    - you can use any approach including modify the allocation scheme or extend the data block pointer structure, etc.
    - But you have to carefully verify the correctness of your implementation, and ensure both file read and write can be correctly performed.

MP4 - Assignments

- Part III - Modify file system code to support subdirectory
  - Goal
    - support subdirectory structures
    - at most 64 files/subdirectories in an directory
    - file name and directory name will not exceed 9 characters
    - use ‘/’ as the path name separator
    - support recursively list the file/directory in a directory

MP4 - Assignments

- Other - Memory Leakage Check
  - In this assignment, you are required to use valgrind to check whether your Nachos leaks memory.
  - `>valgrind ../build.linux/nachos <NachOS command>`

```
==22773==
==22773== HEAP SUMMARY:
==22773==     in use at exit: 652 bytes in 3 blocks
==22773==   total heap usage: 2,318 allocs, 2,315 frees, 179,604 bytes allocated
==22773==
==22773== LEAK SUMMARY:
==22773==    definitely lost: 0 bytes in 0 blocks
==22773==    indirectly lost: 0 bytes in 0 blocks
==22773==      possibly lost: 0 bytes in 0 blocks
==22773==    still reachable: 652 bytes in 3 blocks
==22773==         suppressed: 0 bytes in 0 blocks
==22773== Reachable blocks (those to which a pointer was found) are not shown.
==22773== To see them, rerun with: --leak-check=full --show-leak-kinds=all
==22773==
==22773== For counts of detected and suppressed errors, rerun with: -v
==22773== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 18 from 8)
```

MP4 - Assignments

- How to de-leakage?
  - `>valgrind --leak-check=full ...`
- You are responsible to eliminate these leakage.
- N block(s) of leakage receives $\lceil \log_2(N+1) \rceil$ point(s) penalty

```
==21161== HEAP SUMMARY:
==21161==     in use at exit: 10,268 bytes in 45 blocks
==21161==   total heap usage: 2,318 allocs, 2,273 frees, 179,604 bytes allocated
==21161==
==21161== 248 (8 direct, 240 indirect) bytes in 1 blocks are definitely lost in loss record 5 of 9
==21161==    at 0x4008465: operator new(unsigned int) (vg_replace_malloc.c:327)
==21161==    by 0x805BBEB: FileSystem::FileSystem(bool) (filesys.cc:117)
==21161==    by 0x8053BAC: Kernel::Initialize() (kernel.cc:115)
==21161==    by 0x8055443: main (main.cc:299)
==21161==
==21161== 4,560 bytes in 19 blocks are definitely lost in loss record 8 of 9
==21161==    at 0x4008465: operator new(unsigned int) (vg_replace_malloc.c:327)
==21161==    by 0x805AF90: FileHeader::AllocateInternal(PersistentBitmap*, int, int) (filehdr.cc:114)
==21161==    by 0x805AC44: FileHeader::Allocate(PersistentBitmap*, int) (filehdr.cc:75)
==21161==    by 0x805BA64: FileSystem::FileSystem(bool) (filesys.cc:100)
==21161==    by 0x8053BAC: Kernel::Initialize() (kernel.cc:115)
==21161==    by 0x8055443: main (main.cc:299)
==21161==
==21161== 4,808 (8 direct, 4,800 indirect) bytes in 1 blocks are definitely lost in loss record 9 of 9
==21161==    at 0x4008465: operator new(unsigned int) (vg_replace_malloc.c:327)
==21161==    by 0x805BBAA: FileSystem::FileSystem(bool) (filesys.cc:116)
==21161==    by 0x8053BAC: Kernel::Initialize() (kernel.cc:115)
==21161==    by 0x8055443: main (main.cc:299)
==21161==
==21161== LEAK SUMMARY:
==21161==    definitely lost: 4,576 bytes in 21 blocks
==21161==    indirectly lost: 5,040 bytes in 21 blocks
==21161==      possibly lost: 0 bytes in 0 blocks
==21161==    still reachable: 652 bytes in 3 blocks
==21161==         suppressed: 0 bytes in 0 blocks
==21161== Reachable blocks (those to which a pointer was found) are not shown.
==21161== To see them, rerun with: --leak-check=full --show-leak-kinds=all
```

MP4 – Bonus

- Bonus I (5pt)
  - Extend the disk from 128KB to 64MB
  - support up to 64MB single file
- Bonus II (5pt)
  - Remove a file or recursively remove the directory

Sample Test Case

- We have prepared some sample test cases for you guys to debug
- Simple Run:
  - `>./FS_partII_a.sh`
  - `>./FS_partII_b.sh`
  - `>./FS_partIII.sh`
  - `>./FS_bonusI.sh`
  - `>./FS_bonusII.sh`
- TA will use some hidden test case to verify your correctness.

Result of Sample Test Case

- FS_partII_a.sh

```
/FS_test1
abcdefghijklmnopqrstuvwxyz
/FS_test2
Passed! ^_^
```

- FS_partII_b.sh

```
000000901 000000902 000000903 000000904 000000905 000000906 000000907 000000908 000000909 000000910
000000911 000000912 000000913 000000914 000000915 000000916 000000917 000000918 000000919 000000920
000000921 000000922 000000923 000000924 000000925 000000926 000000927 000000928 000000929 000000930
000000931 000000932 000000933 000000934 000000935 000000936 000000937 000000938 000000939 000000940
000000941 000000942 000000943 000000944 000000945 000000946 000000947 000000948 000000949 000000950
000000951 000000952 000000953 000000954 000000955 000000956 000000957 000000958 000000959 000000960
000000961 000000962 000000963 000000964 000000965 000000966 000000967 000000968 000000969 000000970
000000971 000000972 000000973 000000974 000000975 000000976 000000977 000000978 000000979 000000980
000000981 000000982 000000983 000000984 000000985 000000986 000000987 000000988 000000989 000000990
000000991 000000992 000000993 000000994 000000995 000000996 000000997 000000998 000000999 000001000
========================================
000000001 000000002 000000003 000000004 000000005 000000006 000000007 000000008 000000009 000000010
000000011 000000012 000000013 000000014 000000015 000000016 000000017 000000018 000000019 000000020
000000021 000000022 000000023 000000024 000000025 000000026 000000027 000000028 000000029 000000030
000000031 000000032 000000033 000000034 000000035 000000036 000000037 000000038 000000039 000000040
000000041 000000042 000000043 000000044 000000045 000000046 000000047 000000048 000000049 000000050
000000051 000000052 000000053 000000054 000000055 000000056 000000057 000000058 000000059 000000060
000000061 000000062 000000063 000000064 000000065 000000066 000000067 000000068 000000069 000000070
000000071 000000072 000000073 000000074 000000075 000000076 000000077 000000078 000000079 000000080
000000081 000000082 000000083 000000084 000000085 000000086 000000087 000000088 000000089 000000090
000000091 000000092 000000093 000000094 000000095 000000096 000000097 000000098 000000099 000000100
========================================
[0] 100 F
[1] 1000 F
```

Result of Sample Test Case

- FS_partIII.sh

```
[0] t0 D
[1] t1 D
[2] t2 D
========================================
[0] f1 F
[1] aa D
[2] bb D
[3] cc D
========================================
[0] t0 D
        [0] f1 F
        [1] aa D
        [2] bb D
                [1] f2 F
                [2] f3 F
                [3] f4 F
        [3] cc D
[1] t1 D
[2] t2 D
========================================
000000001 000000002 000000003 000000004 000000005 000000006 000000007 000000008 000000009 000000010
000000011 000000012 000000013 000000014 000000015 000000016 000000017 000000018 000000019 000000020
000000021 000000022 000000023 000000024 000000025 000000026 000000027 000000028 000000029 000000030
000000031 000000032 000000033 000000034 000000035 000000036 000000037 000000038 000000039 000000040
000000041 000000042 000000043 000000044 000000045 000000046 000000047 000000048 000000049 000000050
000000051 000000052 000000053 000000054 000000055 000000056 000000057 000000058 000000059 000000060
000000061 000000062 000000063 000000064 000000065 000000066 000000067 000000068 000000069 000000070
000000071 000000072 000000073 000000074 000000075 000000076 000000077 000000078 000000079 000000080
000000081 000000082 000000083 000000084 000000085 000000086 000000087 000000088 000000089 000000090
000000091 000000092 000000093 000000094 000000095 000000096 000000097 000000098 000000099 000000100
========================================
000000001 000000002 000000003 000000004 000000005 000000006 000000007 000000008 000000009 000000010
000000011 000000012 000000013 000000014 000000015 000000016 000000017 000000018 000000019 000000020
000000021 000000022 000000023 000000024 000000025 000000026 000000027 000000028 000000029 000000030
000000031 000000032 000000033 000000034 000000035 000000036 000000037 000000038 000000039 000000040
000000041 000000042 000000043 000000044 000000045 000000046 000000047 000000048 000000049 000000050
000000051 000000052 000000053 000000054 000000055 000000056 000000057 000000058 000000059 000000060
000000061 000000062 000000063 000000064 000000065 000000066 000000067 000000068 000000069 000000070
000000071 000000072 000000073 000000074 000000075 000000076 000000077 000000078 000000079 000000080
000000081 000000082 000000083 000000084 000000085 000000086 000000087 000000088 000000089 000000090
000000091 000000092 000000093 000000094 000000095 000000096 000000097 000000098 000000099 000000100
```

Result of Sample Test Case

- Note that you are free to have your own format of showing the structures of subdirectories as long as you can:
  - Show the relationship of upper and lower layer of directories
  - Show that the entry is a File or Directory
- It is recommend for you to study the script and write your own one to test your implementation.

Grading

- Program correctness - Demo (on server)
  - partII - 30%, partIII - 30%
- Report
  - Describe Nachos Native File System & answer the questions of PartI assignment - 20%
  - Explain your implementation for PartII and III in details – 20%
  - Code modifications, group contribution, etc.
- Bonus: 10%

Grading

- Deadline: 2017/1/17(Wednesday) 08:00, penalty for late submission
- You can discuss, but do not copy. 0 will be given to cheaters.

Hint

- Files to be heavily modified:
  - filehdr.cc/h, directory.cc/h, filesys.cc/h, MP1 related files
- Files to be slightly modified:
  - main.cc/h
- File system related files
  - filesys.cc/h: top-level file system functions (e.g. format, create, remove, …)
  - directory.cc/h: structures of one directory
  - openfile.cc/h: similar to file descriptor(e.g. open, read, write, close)
  - filehdr.cc/h: inode information (e.g. sectors on disk)
  - synchdisk.cc/h: operation on disks

Hint

- Part II
  - Add some indirect blocks in FileHeader.
  - Think some data structure to store information larger than a sector size
    - linked list, heap, tree, …
  - Be very sure what each function in filehdr.cc done
  - Read hint in the comment, they might be helpful!

Hint

- Part III
  - you might want to add some information in DirectoryEntry and Directory
  - the relationship between openfile and directory is important!
  - Read hint in the comment, they might be helpful!