

# CSGE602055 Operating Systems

## CSF2600505 Sistem Operasi

### Week 03: File System & FUSE

Rahmat M. Samik-Ibrahim (ed.)

University of Indonesia

<https://os.vlsm.org/>

Always check for the latest revision!

REV194 14-Feb-2019

# Operating Systems 2019-1

A (Rm 3114) [Tu/Th 10-12] — B (Rm 3114) [Tu/Th 13-15] — C (Rm 3114)

[Tu/Th 16-18] — D (Rm 2401) [Tu/Th 10-12] — E (Rm 2306) [Tu/Th 13-15]

Week	Schedule	Topic	OSC10
Week 00	07 Feb - 13 Feb 2019	Overview 1, Virtualization & Scripting	Ch. 1, 2, 18.
Week 01	14 Feb - 20 Feb 2019	Overview 2, Virtualization & Scripting	Ch. 1, 2, 18.
Week 02	21 Feb - 27 Feb 2019	Security, Protection, Privacy, & C-language	Ch. 16, 17
Week 03	28 Feb - 06 Mar 2019	File System & FUSE	Ch. 13, 14, 15
Week 04	12 Mar - 18 Mar 2019	Addressing, Shared Lib, & Pointer	Ch. 9
Week 05	19 Mar - 25 Mar 2019	Virtual Memory	Ch. 10
Mid-Term	23-30 Mar 2019 (tba)	MidTerm (UTS)	
Week 06	02 Apr - 08 Apr 2019	Concurrency: Processes & Threads	Ch. 3, 4
Week 07	09 Apr - 15 Apr 2019	Synchronization & Deadlock	Ch. 6, 7, 8
Week 08	16 Apr - 22 Apr 2019	Scheduling	Ch. 5
Week 09	23 Apr - 29 Apr 2019	Storage, BIOS, Loader, & Systemd	Ch. 11
Week 10	30 Apr - 06 May 2019	I/O & Programming	Ch. 12
Reserved	07 May - 17 May 2019		
Final	18-25 May 2019 (tba)	Final (UAS)	This schedule is subject to change.
Extra	27 Jun 2019	Extra assignment confirmation	

# The Weekly Check List

- ☐ **Resources:** <https://os.vlsm.org/>
  - ☐ **Download Slides and Demos from GitHub.com**  
<https://github.com/UI-FASILKOM-OS/SistemOperasi/>
  - ☐ **Problems** — <https://rms46.vlsm.org/2/>:  
195.pdf (Week 00), 196.pdf (Week 01), 197.pdf (Week 02),  
198.pdf (Week 03), 199.pdf (Week 04), 200.pdf (Week 05),  
201.pdf (Week 06), 202.pdf (Week 07), 203.pdf (Week 08),  
204.pdf (Week 09), 205.pdf (Week 10).
  - ☐ **Badak All in One** — [BADAK.cs.ui.ac.id:///extra/](http://badak.cs.ui.ac.id///extra/)
- ☐ **Text Book:** any recent/decent OS book. Eg. (**OSC10**) Silberschatz et. al.: **Operating System Concepts**, 10<sup>th</sup> Edition, 2018. See also <http://codex.cs.yale.edu/avi/os-book/OS10/>.
- ☐ Encode your **QRC** with size upto 7cm x 7cm (ca. 400x400 pixels):  
**"OS191 CLASS ID SSO-ACCOUNT Your-Full-Name"**
- ☐ Write your Memo (with QRC) **every week**.
- ☐ Login to [badak.cs.ui.ac.id](http://badak.cs.ui.ac.id) via [kawung.cs.ui.ac.id](http://kawung.cs.ui.ac.id) for at least **10 minutes** every week. Copy the weekly demo folders into your own badak home directory.  
Eg.: `cp -r /extra/Demos/* ~/mydemos/`

# Agenda

- 1 Start
- 2 Schedule
- 3 Agenda
- 4 Week 03
- 5 File System Interface
- 6 File System Implementation
- 7 Devices
- 8 FUSE
- 9 The End

# Week 03 File System & FUSE: Topics<sup>1</sup>

- Files: data, metadata, operations, organization, buffering, sequential, nonsequential
- Directories: contents and structure
- File systems: partitioning, mount/unmount, virtual file systems
- Standard implementation techniques
- Memory-mapped files
- Special-purpose file systems
- Naming, searching, access, backups
- Journaling and log-structured file systems

---

<sup>1</sup>Source: ACM IEEE CS Curricula 2013

# Week 03 File System & FUSE: Learning Outcomes<sup>1</sup>

- Describe the choices to be made in designing file systems. [Familiarity]
- Compare and contrast different approaches to file organization, recognizing the strengths and weaknesses of each. [Usage]
- Summarize how hardware developments have led to changes in the priorities for the design and the management of file systems. [Familiarity]
- Summarize the use of journaling and how log-structured file systems enhance fault tolerance. [Familiarity]

---

<sup>1</sup>Source: ACM IEEE CS Curricula 2013

# File System Interface

- File Concept

- File Attributes: Name, Id, Type, Location, Size, Protection, Time Stamp: create, last modified, last accessed.
- File Operation
  - Create/Delete/Truncate
  - Open/Close
  - Read/Write
- File Types: Executable, Object, Source Code, Library, Markup, Markdown, Archive, Compressed.
- File Structure
- Access Methods: Sequential vs Direct Access

- Directory and Disk Structure

- Three-Structured Directories
- FS Mounting vs. Volume Based System

- File Sharing

- Protection: Access Control

# File Systems Implementation

- File System Layers / Structure
  - Application Programs
  - Logical File Systems
  - File-Organization Module
  - Basic File Systems
  - I/O Control
  - Hardware Device
- File System Implementation
- File Control Block
- FS In Memory Structure
- VFS: Virtual File Systems
  - How to support multiple File Systems
  - I.e. How to support multiple `open()/close()` `read()/write()` operations



# Implementation and Allocation Method

- Directory Implementation
  - Linear List
  - Hash Table
- Allocation Method
  - Contiguous
  - Linked
  - Indexed
  - Combined Scheme
- Free Space Management
- Efficiency & Performance
- Recovery

- the /dev/ directory
  - /etc/fstab: configuration of filesystems
  - /etc/mtab → /proc/mounts: mounted filesystems
  - /proc/swaps: swap filesystems
  - df: checking disk space and filesystems
  - Device Major and Minor Numbers
  - UUID - Universally Unique Identifier (128 bits)
  - GUID - Globally Unique Identifiers: `ls -al /dev/disk/by-uuid`
  - practically is NOT guaranteed unique
  - FUSE: Filesystem in Userspace
  - BBFS: Big Brother File System
- More Storage Structure
  - tmpfs
  - objfs
  - ctfs
  - lofs
  - procfs
  - ufs
  - zfs

# FUSE

```
>>>>> $ ls -al
total 172
drwxr-xr-x 4 demo demo 4096 Apr 25 17:30 .
drwx----- 4 demo demo 4096 Apr 25 17:30 ..
-rw-r--r-- 1 demo demo 2214 Apr 25 17:30 1-READ-THIS-FIRST.txt
drwxr-xr-x 2 demo demo 4096 Apr 25 17:30 disk-images
drwxr-xr-x 5 demo demo 4096 Apr 25 17:30 fuse-tutorial-2018-02-04
-rw-r--r-- 1 demo demo 151814 Apr 25 17:30 fuse-tutorial-2018-02-04.tgz
```

```
>>>>> $ cat 1-READ-THIS-FIRST.txt
```

This demo is available in badak:///extra/.

See also: <https://github.com/UI-FASILKOM-OS/os181>.

You should copy this folder to your own working folder.

DO NOT work inside the /extra folder!

=====

Folder Name:

Week09/

# FUSE (2)

To copy the folder to your home directory:

```
cp -r /extra/Week09/W09-demos/ W09-demos/
```

=====

File Listing:

```
* 1-READ-THIS-FIRST.txt (this file)
* disk-images/ (directory)
* fuse-tutorial-2018-02-04 (fuse tutorial)
* fuse-tutorial-2018-02-04.tgz (source).
```

A. disk-images

```
cd disk-images/
```

=====

B. fuse-tutorial

1. UBUNTU's deb packages (privilege):

```
sudo apt-get install autoconf automake build-essential \
    fuse libfuse-dev pkg-config sshfs
```

2. Get the tarball with

```
wget http://www.cs.nmsu.edu/~pfeiffer/fuse-tutorial.tgz
```

3. List and open the tarball with

```
tar tfz fuse-tutorial.tgz
tar xzf fuse-tutorial.tgz
```

4. Enter the directory (yours may be a different version)

```
cd fuse-tutorial-2018-02-04/
ls -al
```

# FUSE (3)

5. Read the manual with  
lynx index.html

---

Writing a FUSE Filesystem: a Tutorial

Joseph J. Pfeiffer, Jr., Ph.D. (pfeiffer@cs.nmsu.edu)  
Emeritus Professor  
Department of Computer Science, New Mexico State University

Version of 2018-02-04

One of the real contributions of Unix has been the view that "everything is a file". A tremendous number of radically different sorts of objects, from data storage to file format conversions to internal operating system data structures, have been mapped to the file abstraction.

One of the more recent directions this view has taken has been Filesystems in User Space, or FUSE (no, the acronym really doesn't work. Oh well). The idea here is that if you can envision your interaction with an object in terms of a directory structure and filesystem operations, you can write a FUSE file system to provide that interaction. You just write code that implements file operations like `open()`, `read()`, and `write()`; when your filesystem is mounted, programs are able to access the data using the standard file operation system calls, which call your code.

FUSE filesystems have been written to do everything from providing remote access to files on a different host without using NFS or CIFS (see SSHFS at [2]<https://github.com/libfuse/sshfs>) to implementing a filesystem to talk to devices using the Media Transfer protocol (see

[.....])

# FUSE (4)

```
6. Run
   ./configure
   make
```

```
7 cd example
```

TO TRY:

```
$ ls -al rootdir
$ ls -al mountdir
$ df
$ ../src/bbfs rootdir/ mountdir/
$ df
$ ls -al rootdir
$ ls -al mountdir
```

TO PLAY:

```
$ cd mountdir
$ touch blah-blah-blah.txt
$ ls -al
$ cd ..
$ ls -al rootdir
```

TO FINISH:

```
$ fusermount -u mountdir
```

EXTRA:

```
# /etc/fstab: configuration of filesystems
# /etc/mtab --> /proc/mounts: mounted filesystems
# /proc/swaps: swap filesystems
# df: checking disk space and filesystems
# GUID (Globally Unique Identifiers) ls -al /dev/disk/by-uuid
RMS
```

# FUSE (5)

```
>>>> $ ./configure
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking for style of include used by make... GNU
checking dependency style of gcc... gcc3
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /bin/grep
checking for egrep... /bin/grep -E
checking for ANSI C header files... yes
[...]
checking for fdatasync... yes
checking that generated files are newer than configure... done
configure: creating ./config.status
config.status: creating Makefile
config.status: creating html/Makefile
config.status: creating src/Makefile
config.status: creating src/config.h
config.status: executing depfiles commands
```

# FUSE (6)

```
>>>> $ make
Making all in example
make[1]: Entering directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/example'
mkdir -p mountdir
mkdir -p rootdir
echo "bogus file" > rootdir/bogus.txt
make[1]: Leaving directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/example'
Making all in html
make[1]: Entering directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/html'
make[1]: Nothing to be done for 'all'.
make[1]: Leaving directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/html'
Making all in src
make[1]: Entering directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/src'
make all-am
make[2]: Entering directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/src'
gcc -DHAVE_CONFIG_H -I. -D_FILE_OFFSET_BITS=64 -I/usr/include/fuse -g -O2 -MT bbfs.o -MD -MP -MF
.deps/bbfs.Tpo -c -o bbfs.o bbfs.c
mv -f .deps/bbfs.Tpo .deps/bbfs.Po
gcc -DHAVE_CONFIG_H -I. -D_FILE_OFFSET_BITS=64 -I/usr/include/fuse -g -O2 -MT log.o -MD -MP -MF
.deps/log.Tpo -c -o log.o log.c
mv -f .deps/log.Tpo .deps/log.Po
gcc -D_FILE_OFFSET_BITS=64 -I/usr/include/fuse -g -O2 -o bbfs bbfs.o log.o -lfuse -pthread
make[2]: Leaving directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/src'
make[1]: Leaving directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/src'
make[1]: Entering directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04'
make[1]: Nothing to be done for 'all-am'.
make[1]: Leaving directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04'
>>>> $
```



# FUSE (7)

```
>>>> $ cd example/
>>>> $ ls -al rootdir/
total 12
drwxr-xr-x 2 demo demo 4096 Apr 25 18:23 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:23 ..
-rw-r--r-- 1 demo demo 11 Apr 25 18:23 bogus.txt
>>>> $ ls -al mountdir/
total 8
drwxr-xr-x 2 demo demo 4096 Apr 25 18:23 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:23 ..
>>>> $ df
Filesystem      1K-blocks      Used Available Use% Mounted on
udev              10240          0       10240   0% /dev
tmpfs            1639412    103116    1536296   7% /run
/dev/vda2        9515660    1677648    7331596  19% /
/dev/vdc1        32895760  12093508    19108200  39% /usr
tmpfs            4098528          0    4098528   0% /dev/shm
tmpfs            5120          0       5120   0% /run/lock
tmpfs            4098528          0    4098528   0% /sys/fs/cgroup
/dev/vdb1        515929528  38454128  451244668   8% /home
tmpfs            819708          0     819708   0% /run/user/1002
>>>> $ ../src/bbfs rootdir/ mountdir/
Fuse library version 2.9
about to call fuse_main
>>>> $ df
Filesystem      1K-blocks      Used Available Use% Mounted on
udev              10240          0       10240   0% /dev
[...]
tmpfs            819708          0     819708   0% /run/user/1002
bbfs            515929528  38454136  451244660   8% /home/demo/mydemo/W09-demos/
                                     fuse-tutorial-2018-02-04/example/mountdir
>>>> $
```

# FUSE (8)

```
>>>> $ ls -al rootdir/
total 12
drwxr-xr-x 2 demo demo 4096 Apr 25 18:23 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:26 ..
-rw-r--r-- 1 demo demo 11 Apr 25 18:23 bogus.txt
>>>> $ ls -al mountdir/
total 12
drwxr-xr-x 2 demo demo 4096 Apr 25 18:23 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:26 ..
-rw-r--r-- 1 demo demo 11 Apr 25 18:23 bogus.txt
>>>> $ cd mountdir/
>>>> $ touch blah-blah-blah.txt
>>>> $ ls -al
total 12
drwxr-xr-x 2 demo demo 4096 Apr 25 18:30 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:26 ..
-rw-r--r-- 1 demo demo 0 Apr 25 18:30 blah-blah-blah.txt
-rw-r--r-- 1 demo demo 11 Apr 25 18:23 bogus.txt
>>>> $ cd ..
>>>> $ ls -al rootdir/
total 12
drwxr-xr-x 2 demo demo 4096 Apr 25 18:30 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:26 ..
-rw-r--r-- 1 demo demo 0 Apr 25 18:30 blah-blah-blah.txt
-rw-r--r-- 1 demo demo 11 Apr 25 18:23 bogus.txt
>>>> $ fusermount -u mountdir
>>>> $ ls -al mountdir/
total 8
drwxr-xr-x 2 demo demo 4096 Apr 25 18:23 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:26 ..
>>>> $
```

# The End

- ☐ This is the end of the presentation.
- ☒ This is the end of the presentation.
  - This is the end of the presentation.