

# Notice

## Delay

1 day : -10%

2 days : -15%

3 ~ 5 days: -50%

## Peer review (proj 2, 3)

## Today's presentation

Team 5, 8

# Project 4 Help

SNU Operating Systems

Nov 21, 2019

DCS Lab

# Overview

- **Geo-tagged file system** (based on **ext2**)
  - Attach a GPS tag to each regular file
- **Access control** with the tags
  - Files are accessible from the location where they are recently created/modified

# Key Challenges

- (1) Modify physical representation of **inode**
  - to embed GPS coordinates
- (2) Add **GPS-related inode operations** and **implement them for ext2 regular files**
  - `set_gps_location`
  - `get_gps_location`
- (3) Modify **access control mechanism** to realize location-based access control

# (1) Add GPS-related fields to inode structure

- fs/ext2/ext2.h
- There are two structs for ext2 inode.
  - **inode in the memory**
  - **inode on the disk**
- Add 5 fields.
- Pay attention to endianness of the fields of the physical inode
  - You may get a hint from other fields in the structures.

```
struct gps_location {  
    int lat_integer;  
    int lat_fractional;  
    int lng_integer;  
    int lng_fractional;  
    int accuracy;  
};
```

# (cf) Recall from Project 2 (WRR) ...

```
// kernel/sched/rt.c
const struct sched_class rt_sched_class = {
    .next = &fair_sched_class,
    .enqueue_task = enqueue_task_rt,
    .dequeue_task = dequeue_task_rt,
    .yield_task = yield_task_rt,

    .check_preempt_curr = check_preempt_curr_rt,
    .pick_next_task = pick_next_task_rt,
    .put_prev_task = put_prev_task_rt,

#ifdef CONFIG_SMP
    .select_task_rq = select_task_rq_rt,

    .set_cpus_allowed = set_cpus_allowed_rt,
    .rq_online = rq_online_rt,
    .rq_offline = rq_offline_rt,
    .pre_schedule = pre_schedule_rt,
    .post_schedule = post_schedule_rt,
    .task_woken = task_woken_rt,
    .switched_from = switched_from_rt,

    .set_curr_task = set_curr_task_rt,
    .task_tick = task_tick_rt,

    .get_rr_interval = get_rr_interval_rt,
    .prio_changed = prio_changed_rt,
    .switched_to = switched_to_rt,
#endif
};
```

**Interface**

**Implementation**

## Multiple implementation sets

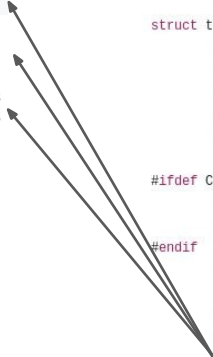
```
const struct sched_class rt_sched_class = {
const struct sched_class fair_sched_class = {
const struct sched_class idle_sched_class = {
```

## Pointing from each task\_struct

```
struct task_struct {
    volatile long state; /* -1 unrunnable
    void *stack;
    atomic_t usage;
    unsigned int flags; /* per process
    unsigned int ptrace;

#ifdef CONFIG_SMP
    struct llist_node wake_entry;
    int on_cpu;
#endif
    int on_rq;

    int prio, static_prio, normal_prio;
    unsigned int rt_priority;
    const struct sched_class *sched_class;
    struct sched_entity se;
    struct sched_rt_entity rt;
```



## (cf) Similar for inode\_operations

```
const struct inode_operations ext4_file_inode_operations = {
```

Interface

```
.setattr  
.getattr  
.setxattr  
.getxattr  
.listxattr  
.removexattr  
.get_acl  
.fiemap
```

Implementation

```
= ext4_setattr,  
= ext4_getattr,  
= generic_setxattr,  
= generic_getxattr,  
= ext4_listxattr,  
= generic_removexattr,  
= ext4_get_acl,  
= ext4_fiemap,
```

```
};
```

```
const struct inode_operations ext3_dir_inode_operations = {  
const struct inode_operations ext4_file_inode_operations = {  
const struct inode_operations ext4_special_inode_operations = {
```

```
struct inode {  
    umode_t            i_mode;  
    unsigned short     i_opflags;  
    kuid_t             i_uid;  
    kgid_t             i_gid;  
    unsigned int       i_flags;  
  
#ifdef CONFIG_FS_POSIX_ACL  
    struct posix_acl    *i_acl;  
    struct posix_acl    *i_default_acl;  
#endif
```

```
const struct inode_operations *i_op;  
struct super_block *i_sb;
```

## (2) Make a new syscall

- `int set_gps_location(struct gps_location__user *loc)`
- $\text{Latitude} = \text{lat\_integer} + \text{lat\_frac} \cdot 10^{-6}$
- $\text{Longitude} = \text{lng\_integer} + \text{lng\_frac} \cdot 10^{-6}$
- $0 \leq \text{lat\_frac}, \text{lng\_frac} \leq 999,999$
- $-90 \leq \text{latitude} \leq 90$
- $-180 \leq \text{longitude} \leq 180$
- Accuracy (meter): non-negative integer

```
struct gps_location {  
    int lat_integer;  
    int lat_fractional;  
    int lng_integer;  
    int lng_fractional;  
    int accuracy;  
};
```



← → ↺ google.com/maps/place/서울특별시+신림동+서울대학교+제2공학관/037.4487432,126.9503647/17z/data=!3m1!4b1!4m5!3m4!1s0x357b60C

☰ 서울특별시 신림동 서울대학교 제2공학관 🔍 ✕



## 서울대학교 제2공학관

서울특별시 신림동



경로



저장



주변



휴대전화로 보내기



공유



(Latitude,  
Longitude)

Accuracy (m)

## (2) Define GPS-related inode operations

- Add the following two function pointer fields to the **struct inode\_operations** structure in **include/linux/fs.h**
  - `int (*set_gps_location)(struct inode *);`
  - `int (*get_gps_location)(struct inode *, struct gps_location *);`

## (2) And implement them for ext2

- Implement the set/get functions for ext2.
  - `set_gps_location`: copy the current device location to the inode
  - `get_gps_location`: copy the inode location to the buffer
- Register the functions with ext2 file inode operations.

## (2) Update location info

- GPS info of **regular files** should be updated whenever they are **created or modified**
  - Use `set_gps_location` operation
    - \*\*\* You may assume that any GPS related operations are performed after properly setting the device location.
- look at
  - `fs/` - for file system code
  - `fs/ext2/` - for ext2 specific code

### (3) Access control

- Files of the modified ext2 can be only **accessible** from the location where they are **recently created/modified**.
- There is an **inode operation related to access control**.  
You can use it.
- Compare the geo-tag and current location.
  - You cannot use float or double operations.  
\*\*\* Note that the kernel does not have any floating point or double precision support.
  - You should consider accuracy of the geo-tag
  - **Compare the values with your own algorithm**. Document any assumptions or approximations on README.md

# Be careful!

- Current **device location** is shared mutable state, so you should use **proper synchronization mechanism** when accessing the state.
- Never access the memory by user-space addresses directly. Refer to guides and provided links in Project 1 help document(Linux Kernel Exploration Guide for OS projects).
- For parameters in struct `gps_location` in **set\_gps\_location syscall**, make sure they are in **appropriate range**.

# Testing with the modified file system

- To test your code, you should create a your modified ext2 file system. You will use **mke2fs** (in e2fsprogs).
- You need to modify ext2 inode structure to make mke2fs use your modified ext2.
  - e2fsprogs/lib/ext2fs/ext2\_fs.h
  - **There is a structure you should modify.**

# About Submission (IMPORTANT!)

- **Due: 2019-12-17 Tuesday 13:00:00 KST**
- Make sure your branch name is *proj4*
- Don't be late!
  - We will not grade the commits after the **deadline**
- Slides and Demo
  - Send it to the TA (**os-tas@dcslab.snu.ac.kr**) before the **deadline**
  - Title: **[OS-ProjX] TeamX slides&demo submission**
  - File name: **TeamX-slides.pptx(.ppt, .pdf), TeamX-demo.mp4(.avi, ...)**
- Please submit only one video
- Check for format : slides title / demo name / branch name and directory name