Project 4

조교: 박지웅, 김화정, 민철기

os-tas@dcslab.snu.ac.kr

Project 4

- Geo-tagged File System
  - embed location information into ext2 file system metadata and use it for access control
  1. Tracking device location (10 pts.)
  2. Add GPS-related operations to inode (20 pts.)
  3. Update location information for files (15 pts.)
  4. User-space testing for location info. (15 pts.)

- Due: 12.11 (월) 14:00PM

Tracking device location (10 pts.)

- Artik10 doesn't have GPS sensors to acquire loc. info.
- Write the following definition of struct gps_location on include/linux/gps.h

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

- Write a new system call to update kernel's current loc. of the device on kernel/gps.c
  - int set_gps_location(struct gps_location __user *loc); /* 380 */
    - latitude = loc->lat_integer + loc->lat_fractional * (10^-6)
    - longitude = loc->lng_integer + loc->lng_fractional * (10^-6)
    - 0 <= Fractional parts for 위도(lat.), 경도(long.) < 999,999
    - -90 <= 위도(latitude) < 90, -180 <= 경도(longitude) <= 180

- Write a user space program *gpsupdate* on test/gpsupdate.c

Add GPS-related operations to inode (20 pts.)

- Modify the linux inode operations interface to include location getter/setter functionality
  - Add the following two members to the struct inode_operations definition in include/linux/fs.h

```
int (*set_gps_location)(struct inode *);
int (*get_gps_location)(struct inode *, struct gps_location *);
```

  - *set_gps_location* use the latest GPS data in the kernel
- Implement this location-related operations for ext2
  - Change the physical representation of an ext2 inode on disk
    by appending the following fields

```
i_lat_integer (32-bits)
i_lat_fractional (32-bits)
i_lng_integer (32-bits)
i_lng_fractional (32-bits)
i_accuracy (32-bits)
```

  - Pay close attention to endianness of the fields

Update location information for files (15 pts.)

- Modify ext2 to update location information of regular files when they are created or modified
  - Call *set_gps_location* whenever a regular file is created or modified
  - “Modified” means that file content have changed
  - You don’t have to care about directories and symbolic links
- Feel free to update directories and symbolic links
  - *You may earn extra points!*

User-space testing for location information (15 pts.)

- Modify *e2fsprogs* which creates our modified ext2
  - Modify the appropriate files in e2fsprogs/lib/ext2fs/
  - Compile the modified *e2fsprogs*
  - Create a modified ext2 using the modified *mke2fs* tool
  - Push the proj4.fs file to your device and mount it
- Write a new system call (381) on kernel/gps.c

```
int get_gps_location(const char __user *pathname, struct gps_location __user *loc);
```

- Write a user space utility *file_loc* on test/file_loc.c to output a file's location including the GPS coordinates and a Google Maps link
  - One command line arg. = the path to a file
  - Wirte a Makefile in test directory to compile *gpsupdate* and *file_loc*

Location-based file access (15 pts.)

- Modify our ext2 so that files can be only readable from the location where they are created or modified
  - You have to consider Accuracy to allow some errors in loc.
  - Note that kernel doesn’t have any floating point or double precision support
  - Location-based filed access is an extra checklist for ext2 rather than the replacement of existing access control mechanism