Systems Programming

User-Space File System

H. Turgut Uyar Şima Uyar

2009-2011

Topics

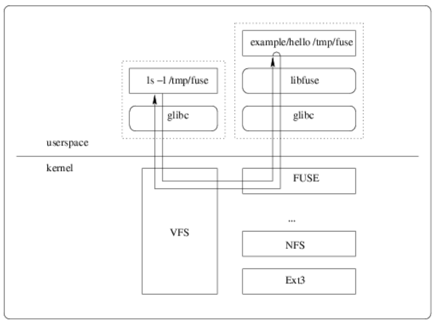
System Programming Levels

- compiling the kernel:
  best performance, every possible functionality
  risky, time-consuming
- kernel modules:
  very good performance, less risky, fast development
  can not do everything
- user-space:
  even less risky, fast development, can use external libraries
  poorer performance, can not do everything

FUSE

- Filesystem in Userspace
- develop a file system in user space on top of a kernel module
  - non-native filesystems (NTFS, ZFS, ...)
  - changing data storage (SQL, ...)
  - providing transparent functionality (compression, encryption, ...)

FUSE Structure



FUSE Development

- similar to device driver development:
  implement system calls
- needed package: libfuse-dev
- system calls:
  - file related:
    open, release, read, write, getattr, unlink, ...
  - directory related:
    readdir, mkdir, rmdir, ...

Example Filesystem: Hello world

- virtual filesystem with only one directory and one file
- the name of the file: `hello.txt`
- the contents of the file: `Hello, world!`

FUSE Development

Example (fuse operations)

```
static struct fuse_operations hello_oper = {
    .getattr = hello_getattr,
    .readdir = hello_readdir,
    .open    = hello_open,
    .read    = hello_read,
};
```

FUSE Development

Example

```
static const char *hello_path = "/hello.txt";
static const char *hello_str = "Hello,␣world!\n";
```

FUSE Development

directory listing: readdir

```
static int hello_readdir(
    const char *path,
    void *buf,
    fuse_fill_dir_t filler,
    off_t offset,
    struct fuse_file_info *fi
);
```

FUSE Development

Example (hello_readdir)

```
if (strcmp(path, "/") != 0)
    return -ENOENT;

filler(buf, ".", NULL, 0);
filler(buf, "..", NULL, 0);
filler(buf, hello_path + 1, NULL, 0);
```

FUSE Development

reading file attributes

```
static int hello_getattr(
    const char *path,
    struct stat *st_data
);
```

FUSE Development

Example (hello_getattr)

```
memset(stbuf, 0, sizeof(struct stat));
if (strcmp(path, "/") == 0) {
    stbuf->st_mode = S_IFDIR | 0755;
    stbuf->st_nlink = 2;
}
else if (strcmp(path, hello_path) == 0) {
    stbuf->st_mode = S_IFREG | 0444;
    stbuf->st_nlink = 1;
    stbuf->st_size = strlen(hello_str);
}
else
    res = -ENOENT;
```

FUSE Development

reading from a file

```
static int hello_read(
    const char *path,
    char *buf,
    size_t size,
    off_t offset,
    struct fuse_file_info *finfo
);
```

FUSE Development

Example (hello_read)

```
if (strcmp(path, hello_path) != 0)
    return -ENOENT;

len = strlen(hello_str);
if (offset < len) {
    if (offset + size > len)
        size = len - offset;
    memcpy(buf, hello_str + offset, size);
} else
    size = 0;

return size;
```

FUSE Development

- compiling:
  ```
  gcc -o hello -Wall -ansi -W -std=c99 -g -ggdb
      -D_GNU_SOURCE -D_FILE_OFFSET_BITS=64
      -lfuse hello.c
  ```
- mounting:
  ```
  ./hello <dir>
  ```
- unmounting:
  ```
  fusermount -u <dir>
  ```
- running in debug mode:
  ```
  ./hello -d <dir>
  ```

Example Filesystem: ROFS

- read-only filesystem
- access an underlying directory in read-only mode
- all read accesses are delegated to the underlying directory
- all write accesses are denied

FUSE Development

Example (fuse operations)

```
struct fuse_operations rofs_oper = {
    .getattr = rofs_getattr,
    .readdir = rofs_readdir,
    .mkdir   = rofs_mkdir,
    .unlink  = rofs_unlink,
    .rmdir   = rofs_rmdir,
    .rename  = rofs_rename,
    .open    = rofs_open,
    .read    = rofs_read,
    .write   = rofs_write,
    .release = rofs_release,
    ...
};
```

FUSE Development

Example (path translation)

```
char *rPath = malloc(sizeof(char)*
    (strlen(path) + strlen(rw_path) + 1));

strcpy(rPath, rw_path);
if (rPath[strlen(rPath)-1] == '/') {
    rPath[strlen(rPath)-1] = '\0';
}
strcat(rPath, path);

return rPath;
```

FUSE Development

Example (directory listing)

```
upath = translate_path(path);
dp = opendir(upath);  /* DIR *dp; */
free(upath);
if(dp == NULL) {
    res = -errno;
    return res;
}

/* fill in the directory info */

closedir(dp);
```

FUSE Development

Example (directory info)

```
/* struct dirent *de; */
while((de = readdir(dp)) != NULL) {
    struct stat st;
    memset(&st, 0, sizeof(st));
    st.st_ino = de->d_ino;
    st.st_mode = de->d_type << 12;
    if (filler(buf, de->d_name, &st, 0))
        break;
}
```

FUSE Development

Example (reading file attributes)

```
upath = translate_path(path);
res = lstat(upath, st_data);
free(upath);
if(res == -1) {
    return -errno;
}
```

FUSE Development

Example (reading from a file)

```
upath = translate_path(path);
fd = open(upath, O_RDONLY);
free(upath);
if(fd == -1) {
    res = -errno;
    return res;
}
res = pread(fd, buf, size, offset);
if(res == -1) {
    res = -errno;
}
close(fd);
```

FUSE Development

modification operations

```
static int rofs_mkdir(
    const char *path,
    mode_t mode
);

static int rofs_unlink(const char *path);

/* body */
return -EROFS;
```

FUSE Development

- compiling:
```
gcc -o rofs -Wall -ansi -W -std=c99 -g -ggdb
    -D_GNU_SOURCE -D_FILE_OFFSET_BITS=64
    -lfuse rofs.c
```
- mounting:
```
./rofs <rw_dir> <ro_dir>
```
- unmounting:
```
fusermount -u <ro_dir>
```
- running in debug mode:
```
./rofs -d <rw_dir> <ro_dir>
```