

## HW3: OS\_6233\_HW3\_File\_System

1. Add functions based on HelloWorld example
2. All the information of processes' id is retrieved from /proc/ directory.
3. The details of one process are retrieved from 'status' in /proc/PROCESS\_ID/status file.

### Added Functions

#### 1. listdir (#138 ~ #167)

```
// get the content of given directory
void listdir(const char *name, int level, void *buf, fuse_fill_dir_t filler) {
    DIR *dir;
    struct dirent *entry;

    if (!(dir = opendir(name)))
        return;
    if (!(entry = readdir(dir)))
        return;

    do {
        if (entry->d_type == DT_DIR) {
            char path[1024];
            int len = snprintf(path, sizeof(path)-1, "%s/%s", name, entry->d_name);
            path[len] = 0;
            if (strcmp(entry->d_name, ".") == 0 || strcmp(entry->d_name, "..") == 0)
                continue;

            // Get the process directory
            if (checkNum(entry->d_name)) {
                filler(buf, entry->d_name, NULL, 0);
                // printf("%s[%s], %d\n", level*2, "", entry->d_name, checkNum(entry-
>d_name));
            }
            // listdir(path, level + 1);
        }
        else {
            // printf("%s- %s\n", level*2, "", entry->d_name);
        }
    } while (entry = readdir(dir));
    closedir(dir);
}
```

Get all the content of given directory. I use this function to get all the process folders under /proc/ directory.

## 2. checkNum

```
// Check if the string is consisted of numbers
int checkNum(const char *p) {
    while(*p) {
        if (*p != '/' && (*p < '0' || *p > '9')) return 0;
        else p++;
    }
    return 1;
}
```

Because the names of processes folder are only consisted of 0~9, by using this function and `listdir`, we can get all the process folders.

## 3. ReadProcInfo (#179 ~ #207)

```
// read process info
char* readProcInfo(const char *filename) {
    char full_path[100] = "";

    const char *path = "/proc";
    const char *sepreator = "/";
    const char *target = "status";

    strcat(full_path, path);
    strcat(full_path, filename);
    strcat(full_path, sepreator);
    strcat(full_path, target);

    char source[BUFF_SIZE + 1];

    FILE *fp = fopen(full_path, "r");
    if (fp != NULL) {
        size_t newLen = fread(source, sizeof(char), BUFF_SIZE, fp);
        if (newLen == 0) {
            fputs("Error reading file", stderr);
        } else {
            source[++newLen] = '\0'; /* Just to be safe. */
        }

        fclose(fp);
    }

    return source;
}
```

Use readProcInfo function to load the detail info of one process. Here is an example of the output:

```

ubuntu@ubuntu-VirtualBox: ~/Desktop
ubuntu@ubuntu-VirtualBox:~/Desktop$ sudo cat /tmp/myproc/1777
Name:      system-service-
State:     S (sleeping)
Tgid:     1777
Pid:      1777
PPid:     1
TracerPid: 727 0 1970-01-01 01:00 580
Uid:      0 0 0 0 1970-01-01 01:00 590
Gid:      0 0 0 0 1970-01-01 01:00 6
FDSize:   256 504 1970-01-01 01:00 7
Groups:   0 0 745 1970-01-01 01:00 706
VmPeak:   root 14760 kB 0-01-01 01:00 735
VmSize:   root 14760 kB 0-01-01 01:00 737
VmLck:    root 720 kB 0-01-01 01:00 748
VmHWM:    root 8452 kB 0-01-01 01:00 757
VmRSS:    root 8452 kB 0-01-01 01:00 761
VmData:   root 3740 kB 0-01-01 01:00 775
VmStk:    root 144 kB 0-01-01 01:00 781
VmExe:    root 1980 kB 0-01-01 01:00 786
VmLib:    root 8100 kB 0-01-01 01:00 793
VmPTE:    root 740 kB 0-01-01 01:00 794
VmSwap:   root 710 kB 0-01-01 01:00 796
Threads:  root 715 1 1970-01-01 01:00 798
SigQ:     0/15993 1970-01-01 01:00 808
SigPnd:   0000000000000000 01 01:00 821
ShdPnd:   0000000000000000 01 01:00 823
SigBlk:   0000000000000000 01 01:00 885
SigIgn:   0000000001001000 01 01:00 894
SigCgt:   0000000180000002 01 01:00 9
CapInh:   0000000000000000 sudo ls -la /tmp/myproc/
CapPrm:   ffffffffffffffff st_mode = S_IRWXU | 0444;
CapEff:   ffffffffffffffff buf->st_nlink = 1;
CapBnd:   ffffffffffffffff buf->st_size = strlen(hello_str);
Cpus_allowed: 3 3 1 (checkNum(path)) {
Cpus_allowed_list: 0-1 stbuf->st_mode = S_IFREG | 0444;
Nfds_allowed: 1 stbuf->st_nlink = 1;
Nfds_allowed_list: 0 stbuf->st_size = strlen(readProcInfo(path))
voluntary_ctxt_switches: 28
nonvoluntary_ctxt_switches: 6

```

## Modified Functions

### 1. hello\_getattr

```
static int hello_getattr(const char *path, struct stat *stbuf)
{
    int res = 0;

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

    return res;
}
```

**checkNum(path)**: checking if it is the process folder.

**strlen(readProcInfo(path))**: getting the size of the file. So we can see the size of each file when we use 'ls -la' to list all the processes file.

### 2. hello\_readdir

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

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

    filler(buf, ".", NULL, 0);
    filler(buf, "..", NULL, 0);

    // filler(buf, myproc_path + 1, NULL, 0);

    // char *temp_path = "l143";
    // filler(buf, temp_path + 1, NULL, 0);

    char *proc_path = "/proc/";
    listdir(proc_path, 0, buf, filler);

    return 0;
}
```

**listdir(proc\_path, 0, buf, filler):** after we list all the process, we write the content into buffer, 'buf'.

### 3. hello\_open

```
static int hello_open(const char *path, struct fuse_file_info *fi)
{
    if (!checkNum(path + 1)) return -ENOENT;
    // if (strcmp(path, myproc_path) != 0) return -ENOENT;

    if ((fi->flags & 3) != 0_RDONLY)
        return -EACCES;

    return 0;
}
```

**!checkNum(path + 1):** if it is not a process folder, we return -ENOENT. By doing this, user can only access the processes folder.

### 4. hello\_read

```
static int hello_read(const char *path, char *buf, size_t size, off_t offset,
                     struct fuse_file_info *fi)
{
    size_t len;
    (void) fi;

    if (checkNum(path + 1)) {
        const char *p = readProcInfo(path);

        len = strlen(p);
        // len = strlen(hello_str);

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

        return size;
    }
    else {
        return 0;
    }
}
```

We read the process info from 'proc/PROCESS\_ID/status' file and write the content into buffer.

## Compile & Run

```

ubuntu@ubuntu-VirtualBox: ~/Desktop
ubuntu@ubuntu-VirtualBox:~/Desktop$ gcc -Wall hello.c `pkg-config fuse --cflags --libs` -o hello
hello.c: In function 'listdir':
hello.c:165:5: warning: suggest parentheses around assignment used as truth value
hello.c: In function 'readProcInfo':
hello.c:206:5: warning: function returns address of local variable
ubuntu@ubuntu-VirtualBox:~/Desktop$ sudo ./hello /tmp/myproc/
ubuntu@ubuntu-VirtualBox:~/Desktop$ sudo cat /tmp/myproc/1777
Name:      System-Service-
State:     S (sleeping)
Tgid:      1777
Pid:       1777
PPid:      1
TracerPid: 0
Uid:       0
Gid:       0
FDSizes:  256
Groups:    0
VmPeak:    14760 kB
VmSize:    14760 kB
VmLck:     0 kB
VmHWM:     8452 kB
VmRSS:     8452 kB
VmData:    3740 kB
VmStk:     144 kB
VmExe:     1980 kB
VmLib:     8100 kB
VmPTE:     40 kB
VmSwap:    0 kB
Threads:   1
sigQ:      0/15993
sigPnd:    0000000000000000
shdPnd:    0000000000000000
sigBlk:    0000000000000000
sigIgn:    0000000010010000
sigCgt:    0000000180000002
CapInh:    0000000000000000
CapPrm:    ffffffffffffffff
CapEff:    ffffffffffffffff

```

1(yellow): compile source file

2(red): mount the file system to /tmp/myproc/

3(green): read the file(process) 1777

## Another screenshot of “ls -la” command

```

ubuntu@ubuntu-VirtualBox:~/Desktop$ sudo ls -la /tmp/myproc
total 4
drwxr-xr-x  2 root root    0 1970-01-01 01:00 .
drwxrwxrwt 15 root root 4096 2014-05-10 22:30 ..
-r--r--r--  1 root root   715 1970-01-01 01:00 1
-r--r--r--  1 root root   513 1970-01-01 01:00 1049
-r--r--r--  1 root root   717 1970-01-01 01:00 1094
-r--r--r--  1 root root   501 1970-01-01 01:00 11
-r--r--r--  1 root root   726 1970-01-01 01:00 1124
-r--r--r--  1 root root   719 1970-01-01 01:00 1127
-r--r--r--  1 root root   733 1970-01-01 01:00 1135
-r--r--r--  1 root root   731 1970-01-01 01:00 1157
-r--r--r--  1 root root   502 1970-01-01 01:00 12
-r--r--r--  1 root root   772 1970-01-01 01:00 1220
-r--r--r--  1 root root   724 1970-01-01 01:00 1226
-r--r--r--  1 root root   745 1970-01-01 01:00 1230
-r--r--r--  1 root root   500 1970-01-01 01:00 13
-r--r--r--  1 root root   782 1970-01-01 01:00 1314
-r--r--r--  1 root root   783 1970-01-01 01:00 1333
-r--r--r--  1 root root   776 1970-01-01 01:00 1393
-r--r--r--  1 root root   505 1970-01-01 01:00 14
-r--r--r--  1 root root   776 1970-01-01 01:00 1403
-r--r--r--  1 root root   775 1970-01-01 01:00 1412
-r--r--r--  1 root root   775 1970-01-01 01:00 1417
-r--r--r--  1 root root   778 1970-01-01 01:00 1423
-r--r--r--  1 root root   775 1970-01-01 01:00 1426
-r--r--r--  1 root root   782 1970-01-01 01:00 1427
-r--r--r--  1 root root   778 1970-01-01 01:00 1432
-r--r--r--  1 root root   785 1970-01-01 01:00 1448
-r--r--r--  1 root root   771 1970-01-01 01:00 1451
-r--r--r--  1 root root   779 1970-01-01 01:00 1456
-r--r--r--  1 root root   783 1970-01-01 01:00 1459
-r--r--r--  1 root root   779 1970-01-01 01:00 1463
-r--r--r--  1 root root   781 1970-01-01 01:00 1467
-r--r--r--  1 root root   783 1970-01-01 01:00 1469
-r--r--r--  1 root root   786 1970-01-01 01:00 1480
-r--r--r--  1 root root   785 1970-01-01 01:00 1484

```

## Environment

OS: Ubuntu 11.04 (Linux kernel 2.6.38)

Language: C