第十三章 守护进程

习题：

13-1、一旦文件从路径打开后，通过访问文件描述符就可以对其操作了，无须再次读其路径。chroot后，并不关闭以打开的文件描述符，但是如果先chroot，再打开之前的路径，可能会找不到路径中的目标；

```
        man 2 chroot
This call does not close open file descriptors, and such file descriptors may allow access to files outside the chroot tree.
```

```c
#include <stdio.h>
#include <syslog.h>

int main(int argc, const char *argv[]) {
    if (2 <= argc) {
        syslog( LOG_ERR, "Syslog befor chroot %m");
    }
    chroot("/opt");
    syslog( LOG_ERR, "Syslog after chroot %m");
    return 0;
}
```

```
tongue aupe # gcc 13-1.c -o 13-1
tongue aupe # ./13-1 x; tail -2 /var/log/messages
Mar 26 11:30:56 tongue 13-1: Syslog befor chroot Success
Mar 26 11:30:56 tongue 13-1: Syslog after chroot Success
tongue aupe # ./13-1; tail -2 /var/log/messages
Mar 26 11:30:56 tongue 13-1: Syslog befor chroot Success
Mar 26 11:30:56 tongue 13-1: Syslog after chroot Success
tongue aupe # ./13-1 x; tail -2 /var/log/messages
Mar 26 11:31:01 tongue 13-1: Syslog befor chroot Success
Mar 26 11:31:01 tongue 13-1: Syslog after chroot Success
```

前后两次都正常写入

两次都没写入，输出刚才的结果

再次写入

13-2、介绍核心的几个吧

ksoftirqd：内核软中断；

kworker：进程队列处理器；

migration：进程多cpu间的调度迁移；

kthreadd：线程管理；

13-3、没了登录名；

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <syslog.h>
#include <string.h>
#include <signal.h>
#include <fcntl.h>
#include <sys/resource.h>

void daemonize(const char *cmd) {
    int i, fd0, fd1, fd2;
    pid_t pid;
    struct rlimit rl;
    struct sigaction sa;
    umask(0);
    if (0 > getrlimit( RLIMIT_NOFILE, &rl)) {
        printf("%s: can't get file limit\n", cmd); _exit(-1);
    }
    if (0 > (pid = fork())) {
        printf("%s: can't fork", cmd); _exit(-1);
    } else if (0 != pid) {
        exit(0);
    }
    setsid();
    sa.sa_handler = SIG_IGN;
    sigemptyset(&sa.sa_mask);
    sa.sa_flags = 0;
    if (0 > sigaction( SIGHUP, &sa, NULL)) {
        printf("%s: can't ignore SIGHUP", cmd); _exit(-1);
    }
```

```
    if (0 > (pid = fork())) {
        printf("%s: can't fork", cmd); _exit(-1);
    } else if (0 != pid) {
        exit(0);
    }
    if (0 > chdir("/")) {
        printf("%s: can't change directory to /", cmd); _exit(-1);
    }
    if (rl.rlim_max == RLIM_INFINITY) {
        rl.rlim_max = 1024;
    }
    for (i = 0; i < rl.rlim_max; i++) {
        close(i);
    }
    fd0 = open("/dev/null", O_RDWR);
    fd1 = dup(0);
    fd2 = dup(0);
}

int main(int argc, const char *argv[]) {
    daemonize(strrchr( argv[0], '/'));
    syslog( LOG_INFO, "login name %s", getlogin());
    while(1) {
        sleep(1);
    }
    return 0;
}
/opt/drill_ground/aupe/13-3.c [FORMAT=unix:utf-8] [TYPE=C] [COL=001
tongue aupe # gcc 13-3.c -o 13-3    如绿框
tongue aupe # ./13-3; tail -1 /var/log/messages
Mar 26 16:02:15 tongue 13-3: login name (null)
```