## 同济大学计算机网络 实验报告



姓名: \_\_涂远鹏-1652262\_\_

三.守护进程再次分裂子进程

### 编写 test3-1.c:

```
#include <stdio.h>
#include <sys/types.h>
int main()
     int i,j;
pid_t pid1;
pid1=fork();
if(pid1==-1)
return 0;
if(pid1==0)
          for (i=0; i<10;)
      pid_t pid = fork();
if (pid == 0)
           for(j=0; j<3; j++)
                 printf("%d %d 1652262 sub\n",(int)getppid(),(int)getpid());
                 sleep(25);
           return 0;
      sleep(3);
      while(1)
           printf("%d %d 1652262 main\n",(int)getppid(),(int)getpid());
           sleep(5);
         return 0;
      }
if(pid1>0)
      return 0;
} ~
```

#### 编写 makefile:

生成的可执行文件测试结果:

```
[root@RHEL74-SVR 03]# 2314 2315 1652262 sub
2314
      2316
           1652262
                     sub
2314
      2317
            1652262
                     sub
2314
      2318
            1652262
2314
      2319
            1652262
                     sub
2314
      2320
            1652262
2314
      2321
            1652262
                     sub
2314
      2322
            1652262
2314
      2323
            1652262
                     sub
2314
      2315
            1652262
            1652262
2314
      2324
                     sub
2314
      2316
            1652262
                     sub
  2314 1652262 main
           1652262
2314 2317
                     sub
2314
      2318
            1652262
                     sub
  2314 1652262 main
2314 2319 1652262
                     sub
     2320
2314
           1652262
                     sub
  2314 1652262 main
           1652262
2314 2321
                     sub
1 2314 1652262 main
     2322
            1652262
2314
                     sub
      2323
2314
            1652262
                     sub
    2315
2314
            1652262
                     sub
1 2314 1652262 main
2314 2324
2314 2316
            1652262
                     sub
            1652262
  2314 1652262 main
2314 2317
           1652262
2314
      2318
            1652262
                     sub
  2314 1652262 main
2314 2319 1652262
                     sub
2314 2320 1652262
  2314 1652262 main
2314 2321 1652262
                     sub
1 2314 1652262 main
2314 2322 1652262
2314 2323 1652262
                     sub
                     sub
  2314 1652262 main
2314 2324 1652262 sub
  2314 1652262
                 main
   2314
         1652262
                  main
   2314
         1652262
                  main
   2314
         1652262
                  main
  2314
         1652262
                  main
  2314
         1652262
                  main
```

2.分裂出的子进程和守护进程之间的进程 id 有何关联?

答:分裂出的子进程的 id 均比其父进程(守护进程)的 id 要大并且是从守护进程的 id 开始进行+1 增长的:

```
[root@RHEL74-SVR 03]# ./test3-1
[root@RHEL74-SVR 03]# 2314 231
                                2315 1652262 sub
2314
             1652262
       2316
                        sub
2314
       2317
             1652262
                        sub
2314
       2318
             1652262
                        sub
2314
       2319
             1652262
                        sub
2314
             1652262
       2320
                        sub
2314
       2321
             1652262
                        sub
2314
       2322
              1652262
                        sub
2314
       2323
             1652262
                        sub
2314
       2315
             1652262
                        sub
2314
       2324
             1652262
                        sub
2314
       2316
             1652262
                        sub
  2314
          1652262
                   main
             1652262
2314
       2317
                        sub
       2318
             1652262
2314
                        sub
   2314
          1652262
                   main
             1652262
2314
       2319
                        sub
2314
      2320
             1652262
   2314
          1652262
                    main
             1652262
2314
       2321
                        sub
1
   2314 1652262 main
```

3.什么叫僵尸进程?僵尸进程的产生原因?

答:

#### 僵尸进程:

一个子进程在其父进程还没有调用 wait()或 waitpid()的情况下退出。这个子进程就 是僵尸进程

#### 产生僵尸进程的原因:

一个进程在调用 exit 命令结束自己的生命的时候,其实它并没有真正的被销毁,而是留下一个称为僵尸进程(Zombie)的数据结构(系统调用 exit,它的作用是使进程退出,但也仅仅限于将一个正常的进程变成一个僵尸进程,并不能将其完全销毁)。在 Linux 进程的状态中,僵尸进程是非常特殊的一种,它已经放弃了几乎所有内存空间,没有任何可执行代码,也不能被调度,仅仅在进程列表中保留一个位置,记载该进程的退出状态等信息供其他进程收集,除此之外,僵尸进程不再占有任何内存空间。它需要它的父进程来为它收尸,如果他的父进程没安装 SIGCHLD 信号处理函数调用 wait 或 waitpid ()等待子进程结束,又没有显式忽略该信号,那么它就一直保持僵尸状态,如果这时父进程结束了,那么 init 进程自动会接手这个子进程,为它收尸,它还是能被清除的。但是如果父进程是一个循环,不会结束,那么子进程就会一直保持僵尸状态,这就是为什么系统中有时会有很多的僵尸进程。

查看 test3-1 生成的僵尸进程(ps aux |grep -w 'Z'):

```
ps aux |grep -w
root 2330
                     0.0
                                                                                  [test3-1]
                                              0 pts/0
                                                pts/0
pts/0
                                                                  22:59
22:59
22:59
              2331
2332
root
                     0.0
                           0.0
                                                           Z
                                                                            0:00
                                                                                   test3-1
                                                                                               <defunct>
                                       ō
                                                                            0:00
                     0.0
                                              0
root
                           0.0
                                                                                   test3-1
                                                                                               <defunct>
                                                pts/0
                                                           Z
                                                                                   test3-1
root
                                                                                               <defunct>
                                                           Z
                                                                  22:59
22:59
23:00
root
              2336
                     0.0
                           0.0
                                       0
                                              0
                                                pts/0
                                                                            0:00
                                                                                   test3-1
                                                                                               <defunct>
                                       ō
                                              0
                                                pts/0
root
              2337
                     0.0
                           0.0
                                                                            0:00
                                                                                   test3-1
                                                                                               <defunct>
                                                pts/0
                                                                            0:00
                                                                                   test3-1
root
                                                                                               <defunct>
                                                                  23:00
23:00
23:00
root
              2345
                     0.0
                           0.0
                                       0
                                              0
                                                pts/0
                                                                            0:00
                                                                                   [test3-1]
                                                                                              <defunct>
                           0.0
                                       0
                                              0
                                                pts/0
root
              2346
                     0.0
                                                                            0:00
                                                                                   test3-1
                                                                                              <defunct>
                     0.0
                           0.0
                                                pts/0
                                                                            0:00
                                                                                  [test3-1]
                                                                                              <defunct>
root 2366 0.0 0.
[root@RHEL74-SVR 03]# 1
                           0.0 114880
                                          1024
                                                                  23:01
                                                                            0:00 grep --color=auto -w Z
                                      1652262
                               2329
                                                  main
```

4.如何杀死僵尸进程?

答.

使用 Kill -HUP 僵尸进程 ID 来杀死僵尸进程,往往此种情况无法杀死僵尸进程,此时就

需要杀死僵尸进程的父进程,僵尸进程则被 init 进程,处理 kill -HUP 僵尸进程父 ID: 通过杀死其父进程的方法杀死其子进程:

```
ps aux |grep -w
root 2330
root 2331
                         0.0
                                                                                22:59
22:59
22:59
22:59
                                 0.0
                                                                                             0:00
                                                                                                    [test3-1]
                                                                                                                   <defunct>
                                                                        ZZZZZZZZZZZ
                                                        0 pts/0
0 pts/0
                                 0.0
                                                                                            0:00
                          0.0
                                               0
                                                                                                     [test3-1]
[test3-1]
                                                                                                                   <defunct>
                 2332
root
                                                                                                                   <defunct>
                                                           pts/0
root
                 2333
                          0.0
                                 0.0
                                               0
                                                        0
                                                                                             0:00
                                                                                                     test3-1]
                                                                                                                    <defunct>
                 2336
                                                          pts/0
pts/0
pts/0
                                                                                22:59
root
                          0.0
                                 0.0
                                               0
                                                        0
                                                                                             0:00
                                                                                                     [test3-1]
[test3-1]
                                                                                                                   <defunct>
                 2337
                          0.0
                                               Ö
                                                                                             0:00
root
                                                                                                                   <defunct>
                                                                                                    [test3-1]
[test3-1]
[test3-1]
[test3-1]
root
                 2338
                          0.0
                                 0.0
                                               0
                                                        0
                                                                                 23:00
                                                                                             0:00
                                                                                                                    <defunct>
root
                 2345
                          0.0
                                 0.0
                                               0
                                                        0
                                                          pts/0
pts/0
                                                                                23:00
                                                                                             0:00
                                                                                                                   <defunct>
                 2346
                                                                                             0:00
root
                                                                                                                   <defunct>
root
                 2347
                          0.0
                                 0.0
                                               0
                                                        0
                                                           pts/0
                                                                                 23:00
                                                                                             0:00
root 2366
[root@RHEL74-SVR
1 2329 1652262
                                 0.0 114880
                         0.0 0.
03]# 1
                                                   1024 pts/0
                                                                                23:01
                                                                                             0:00 grep --color=auto -w Z
                                     2329
                                              1652262
                                                             main
    2329
2329
2329
             1652262
1652262
                           main
                           main
     2329
              1652262
     2329
              1652262
                           main
     2329
              1652262
                           main
     2329
              1652262
                           main
             1652262
1652262
1
     2329
                           main
     2329
                           main
              1652262
                           main
             1652262
1652262
     2329
                           main
     2329
                           main
    2329
             1652262
             1652262 main
1652262 main
1652262 main
0rep -w 'Z'1 2329 1652262 main
                           main
    2329
    2329
T 2329 1032202 main

ps aux lorep -w 'Z'1 2329 1652262 main

kill 2329 |

[root@RHEL74-SVR 03]# ps aux |grep -w 'Z'

root 2370 0.0 0.0 114880 1024 pts/0

[root@RHEL74-SVR 03]# |
                                                                      R+ 23:03 0:00 grep --color=auto -w Z
```

# 5.写 test3-2.c, 要求同 test3-1, 但是子进程退出后不能存在僵尸进程: 答:

子进程在终止时会向父进程发 SIGCHLD 信号,Linux 默认是忽略该信号的,我们可以显式安装该信号,在信号处理函数中调用 wait等函数来为其收尸,从而实现要求

编写的 test3-2.c 如下:

```
#include <stdio.h>
#include <sys/types.h>
#include <signal.h>
#include <string.h>
#include <sys/wait.h>
sig_atomic_t child_exit_status;
void clean_up_child_process(int signal_num)
      int status;
     wait (&status);
child_exit_status = status;
}
int main()
     struct sigaction sigchild_action;
memset(&sigchild_action, 0, sizeof(sigchild_action));
sigchild_action.sa_handler = &clean_up_child_process;
sigaction(SIGCHLD, &sigchild_action, NULL);
int i,j;
pid_t pid1;
pid1=fork();
if(pid1==-1)
return 0:
     return 0;
if(pid1==0)
      for(i=0;i<10;)
          pid_t pid = fork();
if (pid == 0)
              for (j=0; j<3; j++)
                 return 0;
     sleep(3);
      while(1)
           printf("%d %d 1652262 main\n",(int)getppid(),(int)getpid());
           sleep(5);
      return 0;
      if(pid1>0)
      return 0;
}
```

暂停查看僵尸进程发现并没有产生僵尸进程,测试结果,并未产生任何僵尸进程:

```
[root@RHEL7-SVR 03]# 2375 2376 1652262 sub
2375 2377 1652262 sub
2375 2378 1652262 sub
2375 2378 1652262 sub
2375 2380 1652262 sub
2375 2381 1652262 sub
2375 2382 1652262 sub
2375 2383 1652262 sub
2375 2384 1652262 sub
2375 2384 1652262 sub
2375 2385 1652262 sub
2375 2387 1652262 sub
2375 2387 1652262 sub
2375 2378 1652262 sub
2375 2378 1652262 sub
2375 2378 1652262 sub
2375 2378 1652262 sub
2375 2381 1652262 sub
2375 2381 1652262 sub
2375 2378 1652262 sub
2375 2381 1652262 sub
2375 2383 1652262 sub
2375 2381 1652262 sub
2
```