进程创建



系统允许一个进程创建新进程,新进程即为子进程,子进程还可以创建新的子进程,形成

进程树结构模型。

```
#include <sys/types.h>
#include <unistd.h>
pid_t fork(void);
```

返回值:

- 成功: 子进程中返回 O, 父进程中返回子进程 ID
- 失败:返回 -1

失败的两个主要原因:

- 1. 当前<mark>系统的进程数已经达到了系统规定的上限</mark>,这时 errno 的值被设置 为 EAGAIN
- 2. <mark>系统内存不足</mark>,这时 errno 的值被设置为 ENOMEM



```
int main() {
                                                   父进程
  pid_t pid = fork();
  if(pid > 0) {
    printf("i am parent process, pid: %d, ppid: %d\n", getpid(), getppid());
  } else if(pid == 0) {
    printf("i am child process, pid: %d, ppid: %d\n", getpid(),getppid());
  for(int i = 0; i < 3; i++) {
    printf("i: %d, pid: %d\n", i, getpid());
    sleep(1);
  return 0.
```

```
int main() {
                                                 子进程
  pid_t pid = fork();
  if(p+> 0) {
    printf("i am parent process, pid: %d, ppid: %d\n", getpid(), getppid());
  } else if(pid == 0) {
    printf("i am child process, pid : %d, ppid : %d\n", getpid(),getppid());
  for(int i = 0; i < 3; i++) {
    printf("i: %d, pid: %d\n", i, getpid());
    sleep(1);
  return 0;
```

内核区 Linux kernel PID=10086 环境变量 命令行参数 PID是局部变量 在栈空间 父进程为子进程 的PID; 子进程为0_{...} 栈空间 共享库 堆空间 .bss .data .text 受保护的地址 (0~4k)



Linux kernel pid=10089 环境变量 命令行参数 栈空间 共享库 堆空间 bss 710199063 data .text 受保护的地址 (0~4k)

fork()以后,子进程的 用户区数据和父进程 一样。内核区也会拷 贝过来,但是 pid



02/父子进程虚拟地址空间

物理内存

写时拷贝 num 11 num 10 num 12 内核区 内核区 用户区 用户区 栈空间 栈空间 num = 11 num = 12 共享库 共享库 堆空间 堆空间

使用 GDB 调试的时候, GDB 默认只能跟踪一个进程, 可以在 fork 函数调用之前, 通过指令设置 GDB 调试工具跟踪父进程或者是跟踪子进程, 默认跟踪父进程。

show follow-fork-mode 查看当前调试的是什么进程

设置调试父进程或者子进程: set follow-fork-mode [parent (默认) | child]

设置调试模式: set detach-on-fork [on | off]

默认为 on,表示调试当前进程的时候,<mark>其它的进程继续运行</mark>,如果为 off,调试当前进程的时候,其它进程被 GDB 挂起。

查看调试的进程: info inferiors

切换当前调试的进程: inferior id

使进程脱离 GDB 调试: detach inferiors id





THANKS



关注【牛客大学】公众号 回复"牛客大学"获取更多求职资料