

# exec和堆空间管理

一: The command parameter of showCmdParam

argv[0]: showCmdParam

argv[1]: arg1

argv[2]: arg2

二:

$$(1) P_1 = 0x401000 + \underbrace{0x1000}_{\text{代码段}} + \underbrace{0x1000}_{\text{数据段}} + \underbrace{0x8}_{\text{哑元}} + \underbrace{0x8}_{\text{新分配的flist}} = 0x403010$$

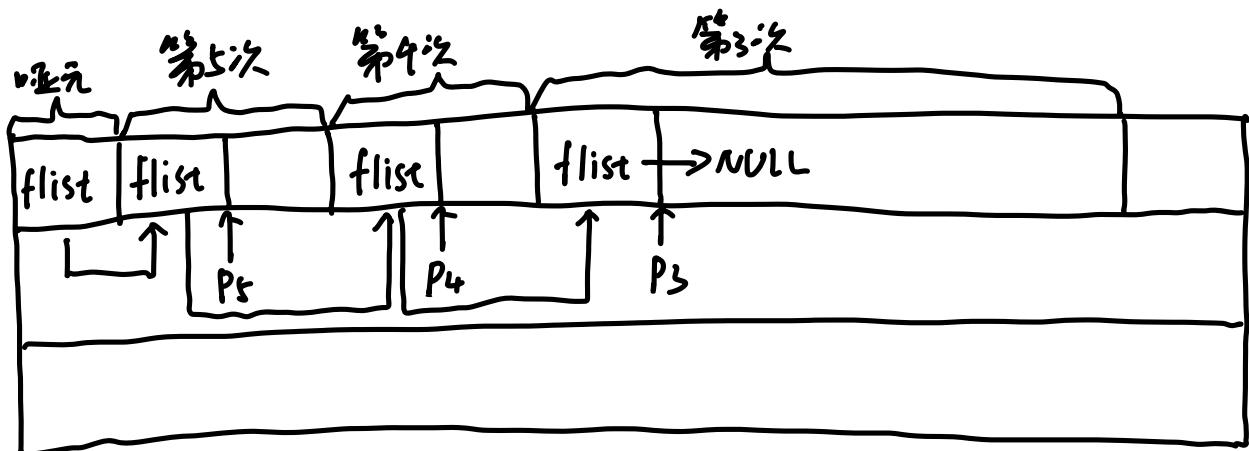
$$(2) P_2 = P_1 + \underbrace{0x8}_{\text{对齐后的第1个分配空间}} + \underbrace{0x8}_{\text{flist}} = 0x403020$$

$$(3) P_3 = P_2 + 0x8 + 0x8 = 0x403030 \quad \text{与 } P_2 \text{ 道理类似}$$

$$(4) P_4 = P_2 = 0x403020 \quad P_4 \text{ 刚好占有 } P_2 \text{ 原来的空间}$$

$$(5) P_5 = P_1 = 0x403010 \quad P_5 \text{ 刚好占有 } P_1 \text{ 原来的空间}$$

(6)



三:

① 为子进程准备命令行参数:

```
argv[0] = "showCmdParam";
```

```
argv[1] = "arg1";
```

```
argv[2] = "arg2";
```

```
argv[3] = 0
```

② 执行fork系统调用, 创建子进程

```
if (fork() == 0)
```

```
    execv(argv[0], argv);
```

```
else  
    ;
```

③ 子进程刚开始和父进程一样执行tryExec, 随后执行exec系统调用, 刷新用户空间, 装入新的程序的图像

④ exec系统调用返回, 回用户态, 从main函数第1条开始执行新程序, 即showCmdParam

四:

① 确认程序文件存在且有执行权限

② 读程序头并记录在PEParser对象中

③ 刷新进程的内存描述符

④ 检查虚空间大小(若不够大, 则放到盘交换区)

⑤ 复制命令行参数(为之后做准备)

⑥ 擦除用户空间

⑦ 重建Text结构

⑧ 为可执行部分分配物理内存, 刷新相对表, 写系统页表

⑨ 加载可执行程序代码, 常量、全局变量和初值

- ⑩在盘交换区为代码段留副本
- ⑪用户栈底构造main栈帧, 释放fakeStack
- ⑫初始化应用程序的执行环境
- ⑬返回用户态, 驱动main1函数

五:

showCmdParam正常终止, 执行内核函数Exit()终止自己:

- ①在盘交换区申请扇区, 启动IO, 将user结构
- ②释放资源
- ③唤醒父进程
- ④将子进程PPid改为1#进程
- ⑤唤醒1#进程