exec 和 堆空间管理

同济大学计算机系　操作系统作业　　2023-12-7

学号 2151769　　姓名 吕博文

一、这个程序的输出是什么?

```
#include <stdio.h>
#include <stdlib.h>

int main1( )   // tryExec.c
{
   char* argv[4];
   argv[0] = "showCmdParam";
   argv[1] = "arg1";
   argv[2] = "arg2";
   argv[3] = 0;

   if ( fork( ) ==0 )
         ;
   else{
      execv(argv[0] , argv) ;
      printf("Over!\n") ;
   }
   exit(0);
}
```

```
#include <stdio.h>
#include <stdlib.h>

int main1(int argc, char *argv[])   // showCmdParam.c
{
   int i;

   printf("The command parameter of showCmdParam\n");

   for(i = 0; i < argc; i++)
      printf("argv[%d]:\t%s\n", i, argv[i]);

   exit(0);
}
```

程序输出：

The command parameter of showCmdParam
argv[0]:showCmdParam
argv[1]:arg1
argv[2]:arg2

注意（1）输出没有“over!” 因为，execv 不是普通的子程序调用，装载新程序 showCmdParam 后，原先用户空间中的 tryExec 程序没了，进程执行不到 printf(“Over!\n”)这条语句。(2) 本题，是父进程在转换进程图像

二、现运行进程 PA，1 页代码，1 页数据，没有只读数据 和 bss，1 页堆栈。代码段起始 0x401000。进程依次执行下列动态内存分配释放操作。

```
char *p1= malloc(4) ;
char *p2= malloc(4) ;
char *p3= malloc(32) ;
free(p2);
char *p4= malloc(8) ;
free(p1);
char *p5= malloc(8) ;
```

（1） 指针 p1 的值是多少 ?

首先，进程首次执行 malloc，在数据段首端执行 brk 系统调用分配三个页的大小，并在开头 0x403000 分配一个 flist 哑元，之后 p1 申请 4 字节空间，加上 flist 大小为 8 字节一共 12 个字节，实行 8 字节对齐，所以紧接着 malloc_begin 之后分配一段长度为 16 字节的空间，p1 = 0x403010.

(2) 指针 p2 的值是多少 ？

P2 申请 4 字节空间，加上 flist 的大小一共 12 字节，实行 8 字节对齐，一共 16 字节，在空闲区中寻找到第一个大于等于 16 字节的空间，分配给 p2 = 0x403020.

(3) 指针 p3 的值是多少 ？

P3 申请 32 字节的空间，加上 flist 大小一共 40 字节，在空闲区中寻找到这样一段空间，分配给 p3 = 0x403030.

(4) 指针 p4 的值是多少 ？

Free(p2)操作之后，p1 的 nlink 直接指向 p3，在所有空闲区中寻找 16 字节的空间时直接找到了上次 p2 被释放的空间，于是 p4 占有原来 P2 的空间，p4 = 0x403020.

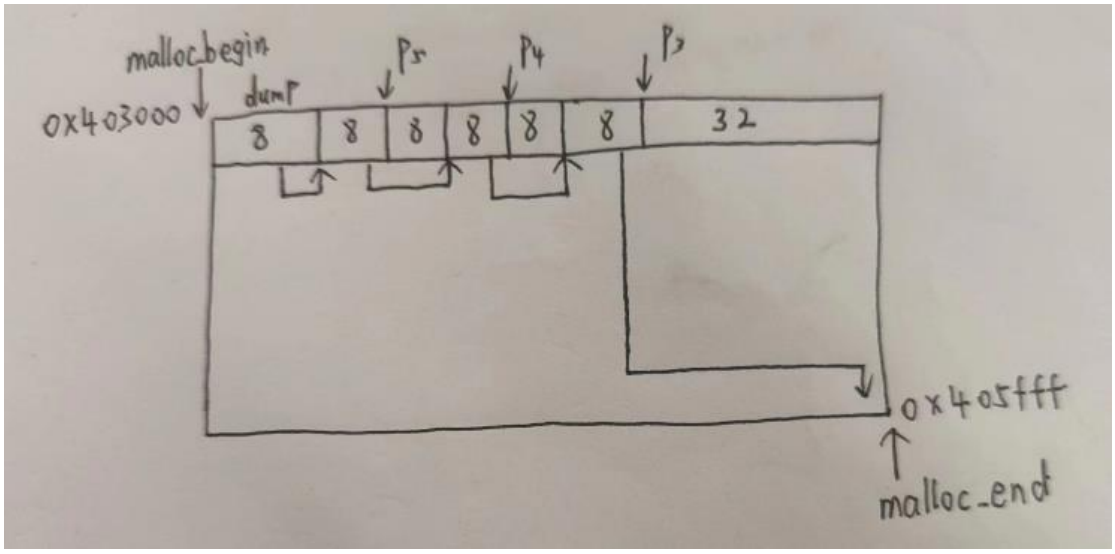
(5) 指针 p5 的值是多少 ？

Free(p1)之后，malloc_begin 的 nlink 直接指向 p4，在所有的空闲区中寻找长度为 16 字节的空间时，直接找到了上次被释放的 p1 的空间，于是 p5 占有 p1 的空间，p5 = 0x403010.

(6) 情景分析

上述求得指针值的过程即为具体的情景分析

(7) 画最终的堆结构图



p1 = 0x403010（内存片浪费 4 字节）

p2 = 0x403020（内存片浪费 4 字节）

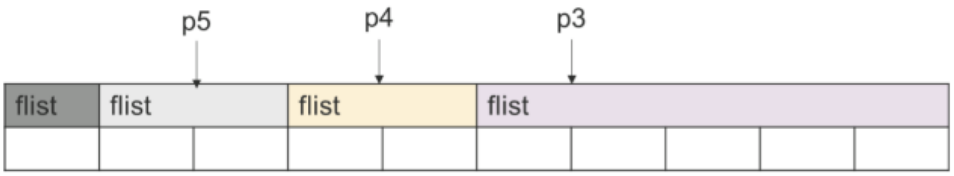
p3= 0x403030

p4 = 0x403020（p2 释放的内存片刚好够用）

p5 = 0x403010（同上）

p5 执行 malloc，发现哑元和分配给 p4 的内存片之间的空闲区（16 字节）刚好够用，用该空闲区装新内存片（没有浪费空间），返回地址是 0x403010。

最终的堆结构图：



三、简述 tryExec 进程创建子进程的过程

tryExec 进程执行 fork 2 号系统调用，为子进程分配空闲的 process 项，调用 Newproc() 过程创建子进程，将父进程图像完整复制到子进程图像中，之后父进程返回新创建的子进程的 pid 号，子进程返回 0.

四、简述程序 showCmdParam 的加载过程

TryExec 进程执行 execv 11 号系统调用，首先系统调用入口函数分析命令行参数个数和具体内容传给 Exec 函数，Exec 函数寻找可执行文件并确定其运行权限，初始化工作完成之后重构进程图像包括 text 结构，数据段，堆栈段等等，系统调用返回用户态之后，之心 runtime 函数驱动应用程序，新的进程从 main 函数的第一条指令开始执行 showCmdParam 进程

五、简述子进程 showCmdParam 进程的终止过程

子进程 showCmdParam 进程正常执行完毕以后，执行 eixt(0)系统调用，由父进程执行 wiat 系统调用回收子进程的 PCB，返回终止码，进程结束。

二、三、四，不必过于详细。要求掌握相关 PPT 的标题和主干步骤。