Lab5

516030910101 罗宇辰

Question

1.

Do you have to do anything else to ensure that this I/O privilege setting is saved and restored properly when you subsequently switch from one environment to another? Why?

不需要。因为在switch的时候old env的trap frame的信息会被push到栈上，再把new env的trap frame 信息pop出来，并没有对页表的I/O权限进行过显式的修改

Challenge

> *Challenge!* Implement Unix-style `exec`.

- **方法**：在spawn的代码上进行修改。按照spawn中的流程创建了child process之后，把child env和father env的属性进行swap（相当于把program的信息从child搬运到father中）然后把child给destroy掉，在father中运行program
- **代码流程**：
    - 文件：`lib/exec.c`

```
// 同spawnl，解析参数后调用exec
int
execl(const char *prog, const char *arg0, ...)
{
    ...

    va_start(vl, arg0);
    unsigned i;
    for(i=0;i<argc;i++)
        argv[i+1] = va_arg(vl, const char *);
    va_end(vl);
    return exec(prog, argv);
}

// 同spawn，先把prog装载到child中，最后调用sys_env_swap把child和father进行交换，然后
destroy child
int
exec(const char *prog, const char **argv)
{
    ...
    // swap env
    if ((r = sys_env_swap(child)) < 0)
        panic("sys_env_set_status: %e", r);
    // destroy child
    sys_env_destroy(child);
    return thisenv->env_id;

error:
    sys_env_destroy(child);
    close(fd);
    return r;
}
```

- 文件：`kern/syscall.c`

```
tatic int
sys_env_swap(envid_t envid)
{
    struct Env *e;
    int r = envid2env(envid, &e, 1);
    if(r < 0)
        return -E_BAD_ENV;
    // 将tf， pgfaultcall，brk改为新环境的设置
    curenv->env_tf = e->env_tf;
    curenv->env_pgfault_upcall = e->env_pgfault_upcall;
    curenv->env_brk = e->env_brk;
    // 交换页表，目的是将来destroy child的时候可以释放father的memory
    pde_t *tmp = curenv->env_pgdir;
    curenv->env_pgdir = e->env_pgdir;
    e->env_pgdir = tmp;
    // 使用新页表
    lcr3(PADDR(curenv->env_pgdir));

    return 0;
}
```

- **测试**：
  - 文件：`user/exechello.c`

```
void
umain(int argc, char **argv)
{
    int r;
    cprintf("i am parent environment %08x\n", thisenv->env_id);
    if ((r = execl("hello", "hello", 0)) < 0)
        panic("exec(hello) failed: %e", r);
}

////////////////////////////////////////
//////        hello 的代码如下      //////
////////////////////////////////////////

void
umain(int argc, char **argv)
```

```
{
    cprintf("hello, world\n");
    cprintf("i am environment %08x\n", thisenv->env_id);
}
```

- 结果:

  ```
  jos@cosmic:~/jos-2019-spring$ make run-exechello
  ...
  i am parent environment 00001001    # currenv is 00001001
  hello, world
  i am environment 00001001           # new prog runs in env 00001001
  ```

BUG

1. lib/exit.c中不注释掉close_all就会time out -> pull之后解决
2. start the shell : TIMEOUT