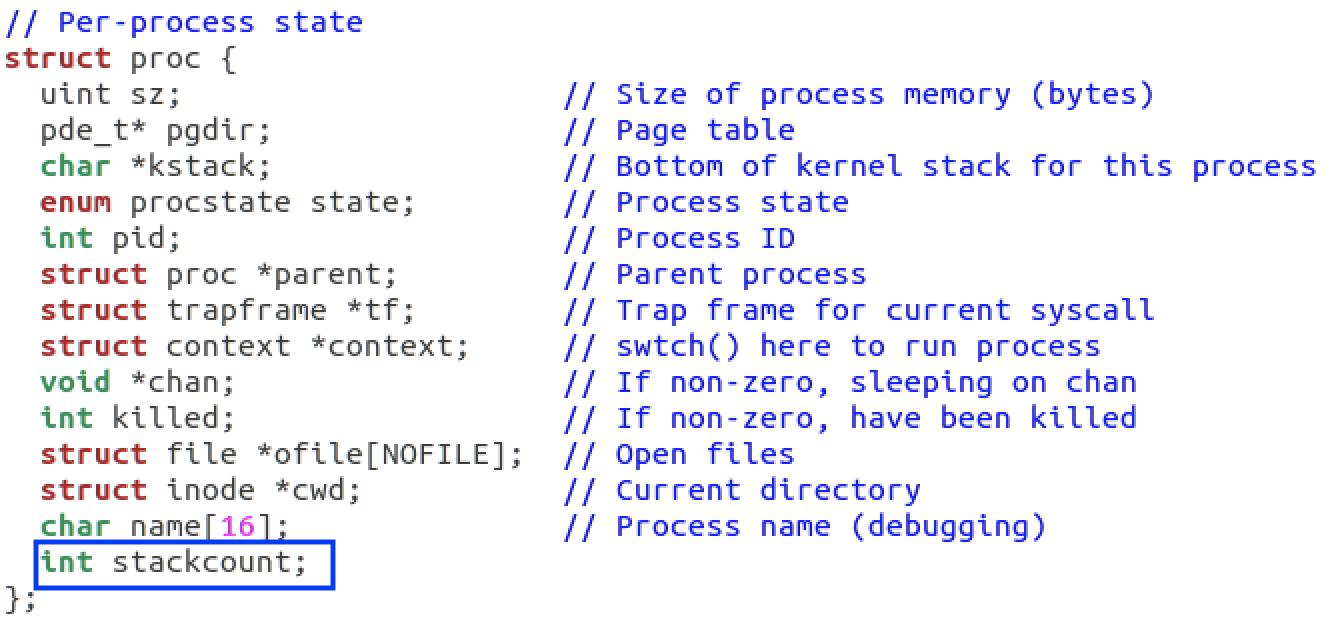
<Project 5. Page Fault Handler> 2015310884 박소현

1. proc.h

To count the number of stack, add int stackcount; in struct proc.



2. proc.c

When allocating a process, stackcount should be 0.

static struct proc\*

allocproc(void)

{

...

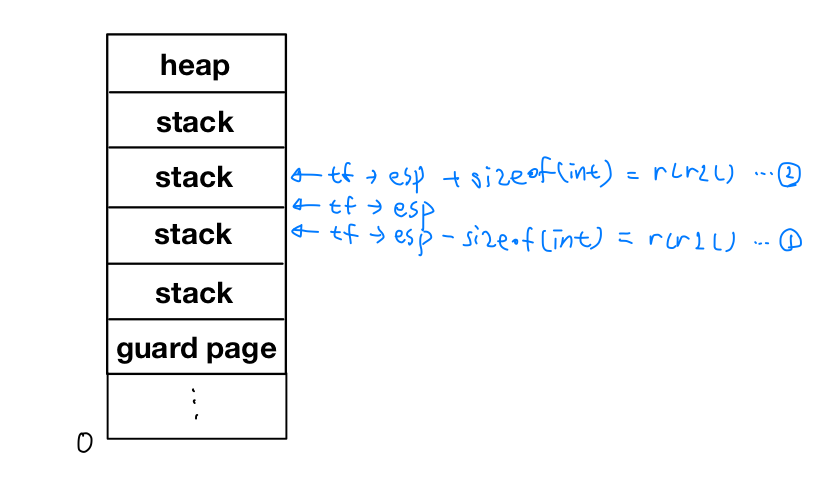
found:

...

p->stackcount = 0;

}

3. trap.c



As you can see in the above picture, I divided into two cases.

>case1: tf->esp - sizeof(int) == rcr2()

if stackcount == 3,

- cprintf("Invalid access\n");

- kill the process

else

- allocate page tables and physical memory to grow process from oldsz to newsz

- cprintf("Allocate page\n");

- stackcount++;

>case2: tf->esp + sizeof(int) == rcr2()

- stackcount --;

>others:

- cprintf("Invalid access\n");

- kill the process

That can be expressed into below codes.

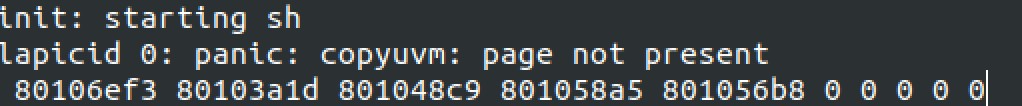


4. exec.c

I changed the code to allocate one page at the next page boundary.



5. vm.c



To solve kernel panic when copying user vm, I inserted codes related counting stack in "copyuvm" function.

pde\_t\*

copyuvm(pde\_t \*pgdir, uint sz)

{

...

int cnt = 0;

...

if( !(\*pte & PTE\_P))

{

cnt += 1;

if(cnt > 3)

panic("copyuvm: page not present");

continue;

}

...

}