作業系統作業 7

經濟三 407510046 黃 X 雯

1.

set_system_trap_gate(SYSCALL_VECTOR, & system_call);

設置了系統調用的中斷向量（即系統調用的入口地址），即 system_call“函數”的地址綁定到中斷向量。所以在 int $0x80 之后，就轉到中斷向量，從而執行中斷處理程序（系統調用程序）。也就是說中斷向量指向的中斷處理程序就是 int $0x80 指令執行完之后接着執行的指令的地方。

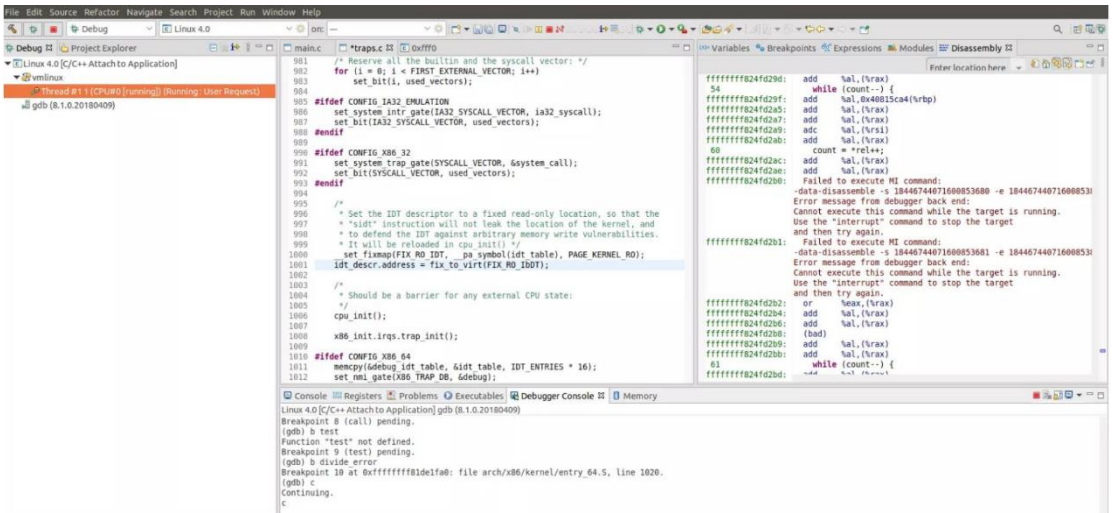
set_nmi_gate(X86_TRAP_BP, &int3)

X86_TRAP_BP:存觸發 int3 的 breakpoint 程式碼的編號

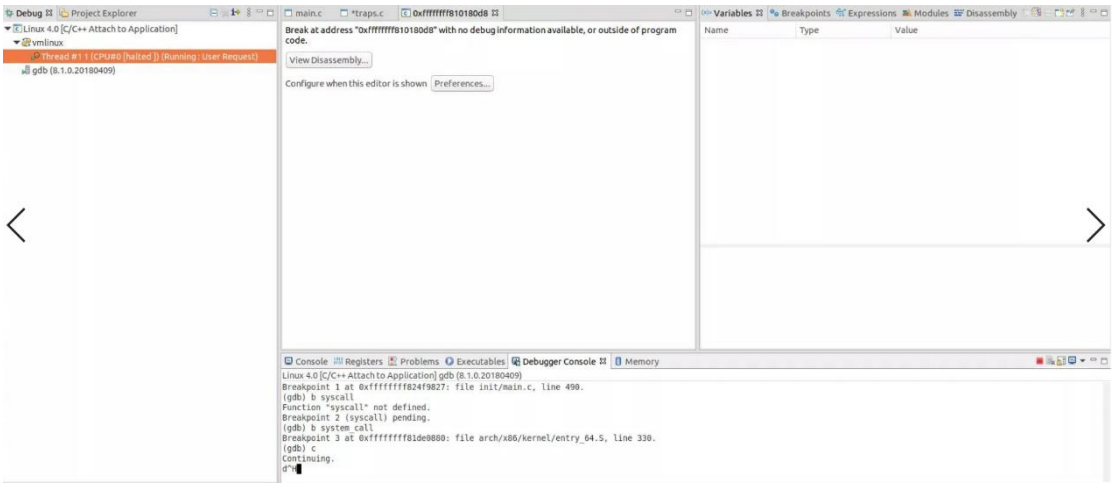
&int3:觸發 int3 中斷的位址

2.

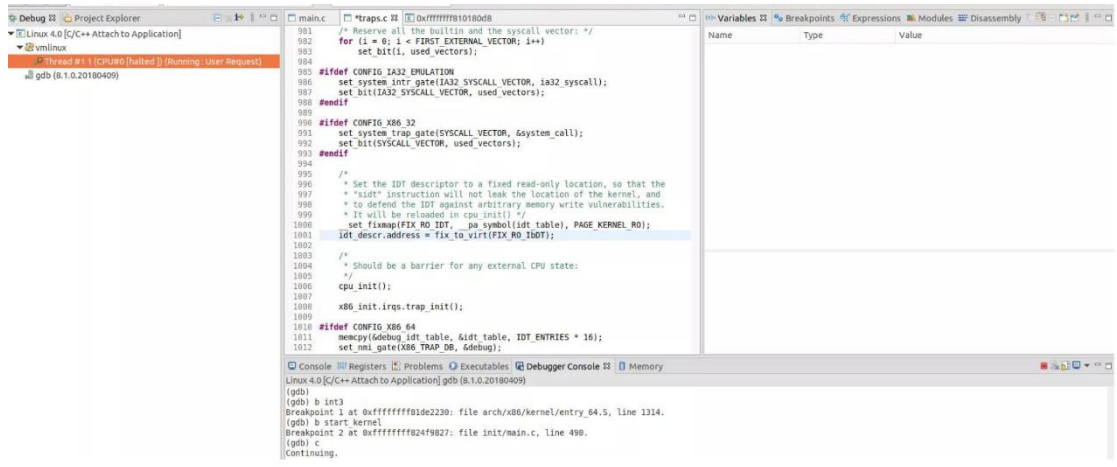
```c
#include <stdlib.h>
#include <stdio.h>

int main(int argc, char** argv) {
        int b = 0;
        int a = 10/b;
        int result;
        result = a/b;
        return 0;
}
~
```



```c
#include        <unistd.h>
#include        <stdio.h>
#include        <string.h>

int     main(int argc,char** argv){
        char* hello = "hello word\n";
        int len = strlen(hello)+1;
        long ret;

        __asm__ volatile(
                "mov $4, %%rax\n"
                "mov $2, %%rbx\n"
                "mov %1, %%rcx\n"
                "mov %2, %%rdx\n"
                "int $0x80\n"
                "mov %%rax, %0"
                : "=m"(ret)
                : "g" (hello), "g" (len)
                : "rax", "rbx", "rcx", "rdx");
        printf("return:%ld\n",ret);
}
~
```




```c
int     main(){
        asm("int3");
}
~
~
```



3.

RAX 暫存器，給一個 system call 內部的實現函數的位置。

system_call 從 system_call_table 拿到 call 的位址，並且將它存到 rax 暫存器中。