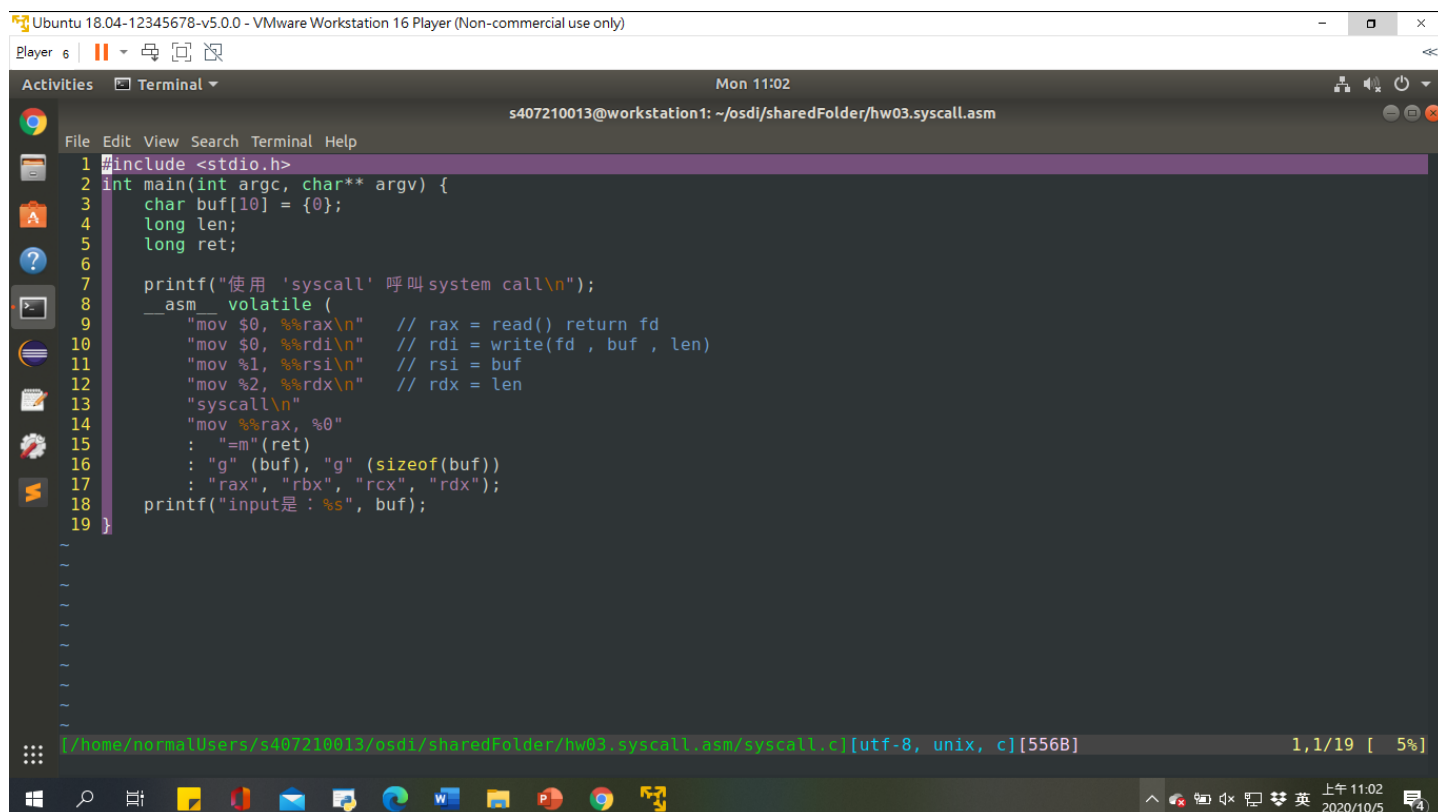
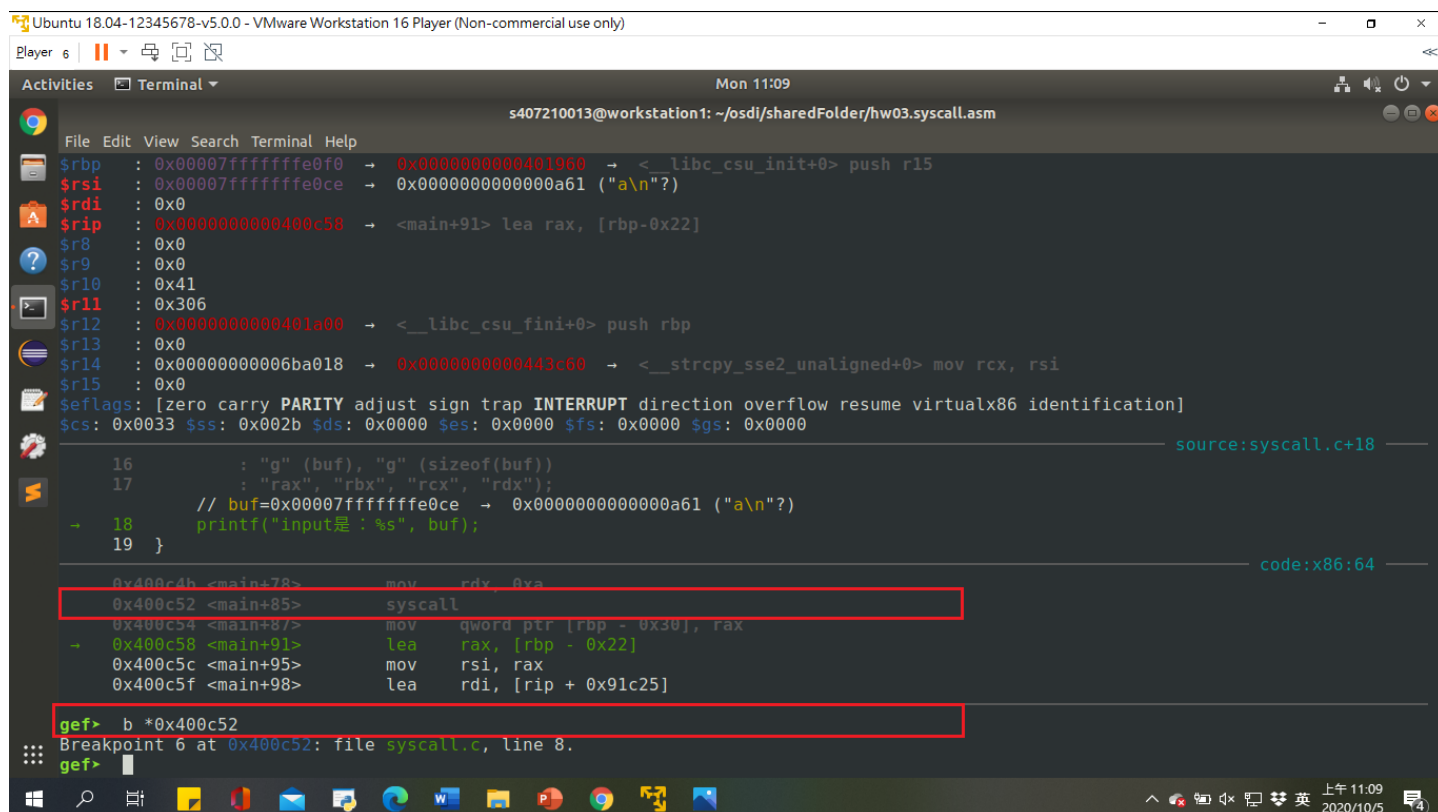


## 1. Syscall.c 程式碼



```
1 #include <stdio.h>
2 int main(int argc, char** argv) {
3     char buf[10] = {0};
4     long len;
5     long ret;
6
7     printf("使用 'syscall' 呼叫 system call\n");
8     __asm__ volatile (
9         "mov $0, %%rax\n"      // rax = read() return fd
10        "mov $0, %%rdi\n"      // rdi = write(fd, buf, len)
11        "mov %1, %%rsi\n"      // rsi = buf
12        "mov %2, %%rdx\n"      // rdx = len
13        "syscall\n"
14        "mov %%rax, %0"
15        : "=m"(ret)
16        : "g" (buf), "g" (sizeof(buf))
17        : "rax", "rbx", "rcx", "rdx");
18    printf("input是: %s", buf);
19 }
```

## 2. 執行程式，並使用 disass /m main，先找到「"syscall\n"」的指標位址，然後設立新的中斷點。



```
$rbp : 0x00007fffffffe0f0 → 0x000000000000401960 → <_libc_csu_init+0> push r15
$rsi : 0x00007fffffffe0ce → 0x00000000000000a61 ("a\n")
$rdi : 0x0
$rip : 0x000000000000400c58 → <main+91> lea rax, [rbp-0x22]
$rs8 : 0x0
$rs9 : 0x0
$rs10 : 0x41
$rs11 : 0x306
$rs12 : 0x000000000000401a00 → <_libc_csu_fini+0> push rbp
$rs13 : 0x0
$rs14 : 0x0000000000006ba018 → 0x000000000000443c60 → <__strcpy_sse2_unaligned+0> mov rcx, rsi
$rs15 : 0x0
$eflags: [zero carry PARITY adjust sign trap INTERRUPT direction overflow resume virtualx86 identification]
$cs: 0x0033 $ss: 0x002b $ds: 0x0000 $es: 0x0000 $fs: 0x0000 $gs: 0x0000

16 : "g" (buf), "g" (sizeof(buf))
17 : "rax", "rbx", "rcx", "rdx");
// buf=0x00007fffffffe0ce → 0x00000000000000a61 ("a\n")
- 18 printf("input是: %s", buf);
19 }
```

## 3. 當運行到 Syscall 前時，觀察 buf 及 reg 的值，發現 buf 仍為空、rax 暫存器也還未寫入。

```

Registers
$rax : 0x0
$rbx : 0x0000000000400400 → <_init+0> sub rsp, 0x8
$rcx : 0x0000000000449751 → 0x5777ffff0003d48 ("H=?")
$rdx : 0xa
$rsp : 0x00007ffffffe0b0 → 0x00007ffffffe218 → 0x00007ffffffe4df → "/home/normalUsers/s407210013/osdi/sharedFolder/hw0[
...] "
$rbp : 0x00007ffffffe0f0 → 0x0000000000401960 → <__libc_csu_init+0> push r15
$rsi : 0x00007ffffffe0ce → 0x0000000000000000
$rdi : 0x0
$rip : 0x0000000000400c52 → <main+85> syscall
$r8 : 0x0
$r9 : 0x0
$r10 : 0x41
$r11 : 0x246
$r12 : 0x0000000000401a00 → <__libc_csu_fini+0> push rbp
$r13 : 0x0
$r14 : 0x00000000006ba018 → 0x0000000000443c60 → <__strcpy_sse2_unaligned+0> mov rcx, rsi
$r15 : 0x0
$eflags: [zero carry PARITY adjust sign trap INTERRUPT direction overflow resume virtualx86 identification]
$cs: 0x0033 $ss: 0x002b $ds: 0x0000 $es: 0x0000 $fs: 0x0000 $gs: 0x0000
source:syscall.c:8

6
7     printf("使用 'syscall' 呼叫system call\n");
→ 8     __asm__ volatile (
9         "mov $0, %%rax\n"    // rax = read() return fd
10        "mov $0, %%rdi\n"    // rdi = write(fd, buf, len)
                                code:x86:64
0x400c41 <main+68>    mov     rdi, 0
0x400c48 <main+75>    mov     rsi, rsi
0x400c4b <main+78>    mov     rdx, 0xa
→ 0x400c52 <main+85>    syscall
0x400c54 <main+87>    mov     qword ptr [rbp - 0x30], rax
0x400c58 <main+91>    lea     rax, [rbp - 0x22]

gef> p buf
$1 = "\000\000\000\000\000\000\000\000\000"
gef>

```

4. 當 Sycall 後，發現 buf 的值變成 a、且 rax 也完成寫入。

```

Registers
$rax : 0x2
$rbx : 0x0000000000400400 → <_init+0> sub rsp, 0x8
$rcx : 0x0000000000400ae4 → <_start+4> rcr DWORD PTR [rsi+0x48], 1
$rdx : 0xa
$rsp : 0x00007ffffffe0b0 → 0x00007ffffffe218 → 0x00007ffffffe4df → "/home/normalUsers/s407210013/osdi/sharedFolder/hw0[
...] "
$rbp : 0x00007ffffffe0f0 → 0x0000000000401960 → <__libc_csu_init+0> push r15
$rsi : 0x00007ffffffe0ce → 0x0000000000000a61 ("a\n"? )
$rdi : 0x0
$rip : 0x0000000000400c58 → <main+91> lea rax, [rbp-0x22]
$r8 : 0x0
$r9 : 0x0
$r10 : 0x41
$r11 : 0x306
$r12 : 0x0000000000401a00 → <__libc_csu_fini+0> push rbp
$r13 : 0x0
$r14 : 0x00000000006ba018 → 0x0000000000443c60 → <__strcpy_sse2_unaligned+0> mov rcx, rsi
$r15 : 0x0
$eflags: [zero carry PARITY adjust sign trap INTERRUPT direction overflow resume virtualx86 identification]
$cs: 0x0033 $ss: 0x002b $ds: 0x0000 $es: 0x0000 $fs: 0x0000 $gs: 0x0000
source:syscall.c:10

16         : "g" (buf), "g" (sizeof(buf))
17         : "rax", "rbx", "rcx", "rdx");
    // buf=0x00007ffffffe0ce → 0x0000000000000a61 ("a\n"? )
→ 18     printf("input是: %s", buf);
19 }
                                code:x86:64
0x400c4b <main+78>    mov     rdx, 0xa
0x400c52 <main+85>    syscall
0x400c54 <main+87>    mov     qword ptr [rbp - 0x30], rax
→ 0x400c58 <main+91>    lea     rax, [rbp - 0x22]
0x400c5c <main+95>    mov     rsi, rax
0x400c5f <main+98>    lea     rdi, [rip + 0x91c25]

gef> p buf
$2 = "a\n\000\000\000\000\000\000\000\000"
gef>

```

5. 利用組語，輸入 a 英文字母，輸出執行結果 a




Last login: Mon Oct 5 10:47:15 2020 from 140.123.222.116

