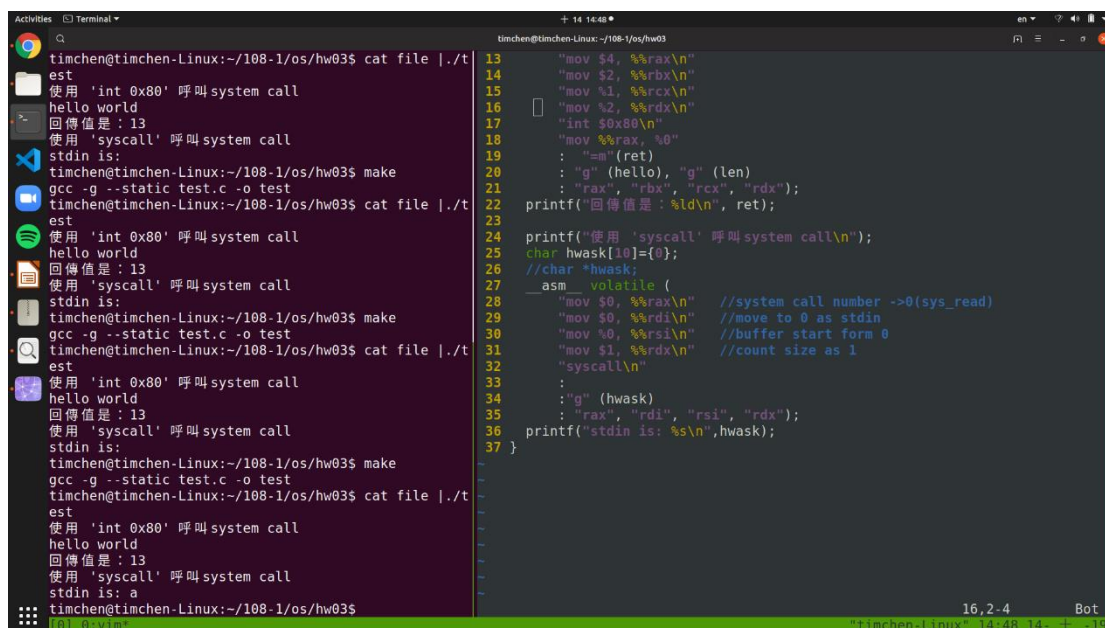


資工三 406410061 陳威樺



```
timchen@timchen-Linux:~/108-1/os/hw03$ cat file |./t
est
使用 'int 0x80' 呼叫system call
hello world
回傳值是: 13
使用 'syscall' 呼叫system call
stdin is:
timchen@timchen-Linux:~/108-1/os/hw03$ make
gcc -g --static test.c -o test
timchen@timchen-Linux:~/108-1/os/hw03$ cat file |./t
est
使用 'int 0x80' 呼叫system call
hello world
回傳值是: 13
使用 'syscall' 呼叫system call
stdin is:
timchen@timchen-Linux:~/108-1/os/hw03$ make
gcc -g --static test.c -o test
timchen@timchen-Linux:~/108-1/os/hw03$ cat file |./t
est
使用 'int 0x80' 呼叫system call
hello world
回傳值是: 13
使用 'syscall' 呼叫system call
stdin is: a
timchen@timchen-Linux:~/108-1/os/hw03$

13  "mov $4, %%rax\n"
14  "mov $2, %%rbx\n"
15  "mov %1, %%rcx\n"
16  "mov %2, %%rdx\n"
17  "int $0x80\n"
18  "mov %%rax, %0"
19  : "=m"(ret)
20  : "g" (hello), "g" (len)
21  : "rax", "rbx", "rcx", "rdx");
22  printf("回傳值是: %ld\n", ret);
23
24  printf("使用 'syscall' 呼叫system call\n");
25  char hwask[10]={0};
26  //char *hwask;
27  __asm__ volatile (
28      "mov $0, %%rax\n" //system call number ->0(sys_read)
29      "mov $0, %%rdi\n" //move to 0 as stdin
30      "mov %0, %%rsi\n" //buffer start form 0
31      "mov $1, %%rdx\n" //count size as 1
32      "syscall\n"
33      :
34      : "g" (hwask)
35      : "rax", "rdi", "rsi", "rdx");
36  printf("stdin is: %s\n",hwask);
37 }
```

由網站提供的資訊，一一將 `rax rdi rsi rdx` 修改成跟 `sys_read` 相同的格式，且將輸出的東西(`hwask`)用成陣列(一開始用 `*char` 會失敗)，最後用 `g` 來接輸出的資料，若用 `m` 的話不會成功，可能因為 `g` 會固定拿 64 位元的東西，而 `m` 不會，因此可能造成拿取的資料錯誤。最後直接輸出 `hwask`。