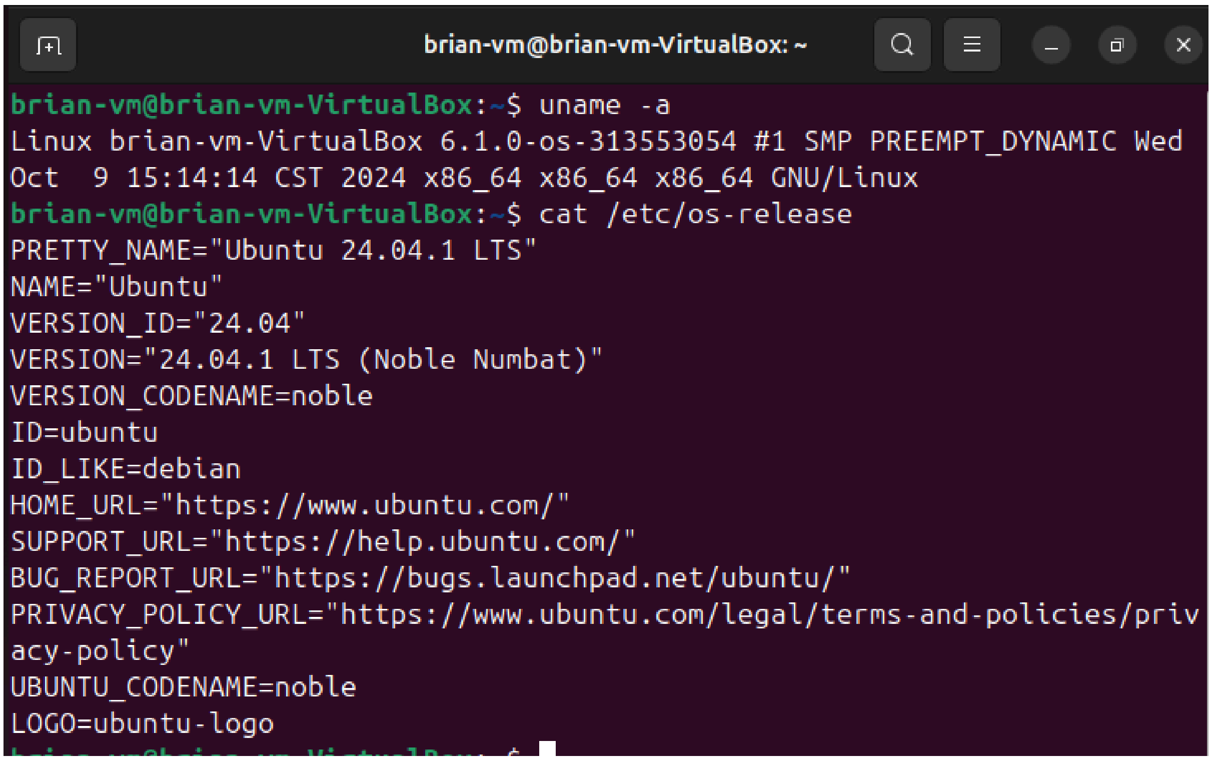
OS Lab1 Report
313553054 陳冠霖

1.Compile linux kernel

After setting up the environment and change local version into my student number(`make menuconfig`) , I started compile kernel from the following url , from step 3 to step 7 . Additionally, I changed the grub settings so that everytime I login my virtual machine ,I can select my own linux local version(6.1).

https://phoenixnap.com/kb/build-linux-kernel



```
brian-vm@brian-vm-VirtualBox:~$ uname -a
Linux brian-vm-VirtualBox 6.1.0-os-313553054 #1 SMP PREEMPT_DYNAMIC Wed
Oct  9 15:14:14 CST 2024 x86_64 x86_64 x86_64 GNU/Linux
brian-vm@brian-vm-VirtualBox:~$ cat /etc/os-release
PRETTY_NAME="Ubuntu 24.04.1 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04.1 LTS (Noble Numbat)"
VERSION_CODENAME=noble
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/priv
acy-policy"
UBUNTU_CODENAME=noble
LOGO=ubuntu-logo
```

2.Implement a new system call

(1)Create a directory named sys_revstr and change the directory to sys_rev/: `mkdir sys_rev` `cd sys_revstr`

(2)Create a file sys_rev.c using gedit: `gedit sys_revstr.c`

(3)Finish C code in sys_revstr.c:

```c
#include <linux/kernel.h>     // Needed for kernel log functions
#include <linux/syscalls.h>   // Needed for system calls
#include <linux/uaccess.h>    // Needed for copy_from_user and copy_to_user

// The sys_revstr system call implementation
SYSCALL_DEFINE2(revstr, char __user *, user_str, int, len)
{
    char *kernel_buffer;
    int i;

    // Allocate memory for the kernel buffer to store the string
    kernel_buffer = kmalloc(len + 1, GFP_KERNEL);
    if (!kernel_buffer)
        return -ENOMEM;  // Return error if memory allocation fails

    // Copy the string from user space to kernel space
    if (copy_from_user(kernel_buffer, user_str, len)) {
        kfree(kernel_buffer);
        return -EFAULT;  // Return error if copying from user space fails
    }

    kernel_buffer[len] = '\0';  // Null-terminate the string

    // Print the original string to the kernel ring buffer
    printk(KERN_INFO "The origin string: %s\n", kernel_buffer);

    // Reverse the string in kernel_buffer
    for (i = 0; i < len / 2; i++) {
        char temp = kernel_buffer[i];
        kernel_buffer[i] = kernel_buffer[len - i - 1];
        kernel_buffer[len - i - 1] = temp;
    }
```

```c
    // Print the reversed string to the kernel ring buffer
    printk(KERN_INFO "The reversed string: %s\n", kernel_buffer);

    // Copy the reversed string from kernel space back to user space
    if (copy_to_user(user_str, kernel_buffer, len)) {
        kfree(kernel_buffer);
        return -EFAULT;  // Return error if copying to user space fails
    }

    kfree(kernel_buffer);  // Free the kernel buffer memory
    return 0;  // Return success
}
```

(4)Create a “Makefile” in the sys_revstr directory:`gedit Makefile`
and add the following line to it:`obj-y := sys_revstr.o`

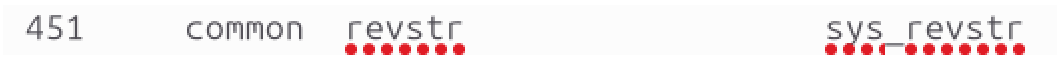
(5)Add a new entry for reverse string system call at syscall table’s entry 451 in /include/uapi/asm-generic/unistd.h file:

```
#define __NR_revstr 451
int syscall(__NR_revstr, char *str, size_t n);
#undef __NR_syscalls
#define __NR_syscalls 451
```

(6)Add an entry(last) in ~/linux/Kbuild to include the created sys_revstr file

```
obj-y                       += init/
obj-y                       += usr/
obj-y                       += arch/$(SRCARCH)/
obj-y                       += $(ARCH_CORE)
obj-y                       += kernel/
obj-y                       += certs/
obj-y                       += mm/
obj-y                       += fs/
obj-y                       += ipc/
obj-y                       += security/
obj-y                       += crypto/
obj-$(CONFIG_BLOCK)         += block/
obj-$(CONFIG_IO_URING)      += io_uring/
obj-$(CONFIG_RUST)          += rust/
obj-y                       += $(ARCH_LIB)
obj-y                       += drivers/
obj-y                       += sound/
obj-$(CONFIG_SAMPLES)       += samples/
obj-$(CONFIG_NET)           += net/
obj-y                       += virt/
obj-y                       += $(ARCH_DRIVERS)
obj-y                       += sys_revstr/
```

(7)Add an entry in arch/x86/entry/syscalls/syscall_64.tbl

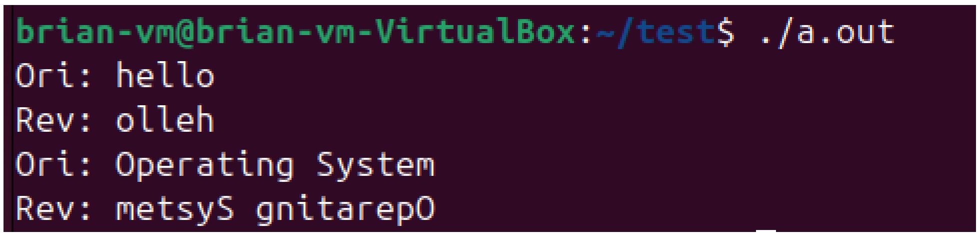


```
451     common  revstr                  sys_revstr
```

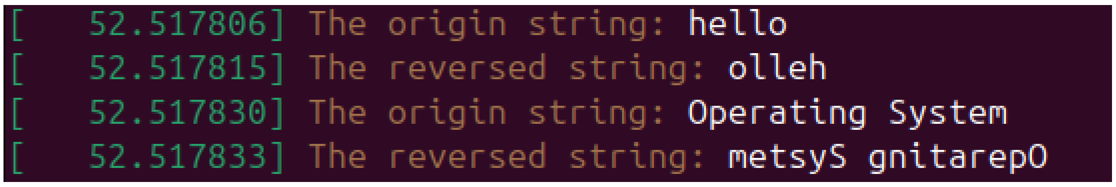
(8)Modify Syscall.h to add asmlinkage

(9)Compile and update kernel again

(10)Create a test directory in userspace and use code in lab spec to test the implemented system calls.



```
brian-vm@brian-vm-VirtualBox:~/test$ ./a.out
Ori: hello
Rev: olleh
Ori: Operating System
Rev: metsyS gnitarepO
```



```
[   52.517806] The origin string: hello
[   52.517815] The reversed string: olleh
[   52.517830] The origin string: Operating System
[   52.517833] The reversed string: metsyS gnitarepO
```