hw1 Report
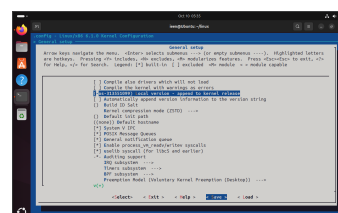
313551099 李以恩

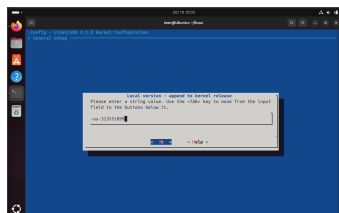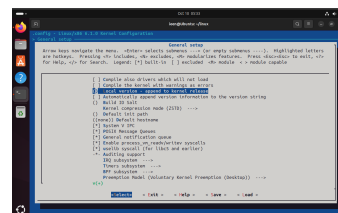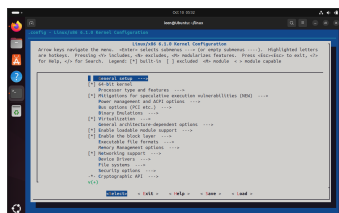## I. Compiling the Linux Kernel

1. **Install Required Packages**

```
sudo apt install build-essential libncurses-dev bison flex libssl-dev libelf-dev
```

2. **Configure Kernel**

```
cp -v /boot/config-$(uname -r) .config
make menuconfig
```

- Append my student-ID to the kernel release.

3. **Build the Kernel**

- It takes up to 3 hours to finish compiling.

```
sudo make
sudo make modules_install
sudo make install
```

4. **Reboot and Verify Kernel Version**

- When the system boots up, I access the GRUB menu to choose the kernel I just compiled.
- GRUB Menu → Advanced options for Ubuntu → Ubuntu with Linux 6.1.0-os-313551099

My result

```
ieen@Ubuntu:~/linux$ make kernelrelease
6.1.0-os-313551099
ieen@Ubuntu:~/linux$ uname -a
Linux Ubuntu 6.1.0-os-313551099 #1 SMP PREEMPT_DYNAMIC Thu Oct 10 09:02:02 UTC 2024 x86_64 x86_64 x86_64 GNU/Linux
ieen@Ubuntu:~/linux$ 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/privacy-policy"
UBUNTU_CODENAME=noble
LOGO=ubuntu-logo
```

II. Implementing a new System Calls

1. **Modify system call number definition**

   - Open the system call table file

     ```
     vim arch/x86/entry/syscalls/syscall_64.tbl
     ```

   - Add the system call number definition to the file

     ```
     451     common     revstr     sys_revstr
     ```

2. **Modify system call declaration**

   - Open the system call declaration file

     ```
     vim include/linux/syscalls.h
     ```

   - Add the declaration of my system call

     ```
     asmlinkage long sys_revstr(char __user *str, int length);
     ```

3. **Implement a new system call**

   - Create my own system call file: revstr/revstr.c

     ```
     vim revstr/revstr.c
     ```

   - Write my system call file:

     ```
     #include <linux/syscalls.h>
     #include <linux/linkage.h>
     #include <linux/kernel.h>
     #include <linux/uaccess.h>

     SYSCALL_DEFINE2(revstr, char __user *, str, int, length)
     {
         int i;
         char original_str[256];
         char reversed_str[256];

         // Copy from user
         if(copy_from_user(original_str, str, length))
         {
             return -EFAULT;
         }
         original_str[length] = '\0';
     ```

```
    // Reverse
    for(i=0; i<length; i++)
    {
        reversed_str[i] = original_str[length - i - 1];
    }
    reversed_str[length] = '\0';

    // Print
    printk("The original string: %s\n", original_str);
    printk("The reversed string: %s\n", reversed_str);

    // Copy to user
    if (copy_to_user(str, reversed_str, length))
    {
        return -EFAULT;
    }
    return 0;
}
```

4. **Add the system call to the kernel build system**

   - Create the makefile of my system call

     ```
     vim revstr/Makefile
     ```

   - Add the following code, to let the revstr.c be compiled as revstr.o

     ```
     obj-y := revstr.o
     ```

5. **Modify the Makefile**

   - Open the Makefile from the linux folder

     ```
     vim Makefile
     ```

   - Append the following code after "core-y"

     ```
     core-y += kernel/ certs/ mm/ fs/ ipc/ security/ crypto/ revstr/
     ```

6. **Recompile**

   ```
   sudo make
   sudo make modules_install
   sudo make install
   sudo update-grub
   sudo reboot
   ```

7. **Test**

   - Use the test code provided from the TAs. Create **revstr_test.c** in user-space.
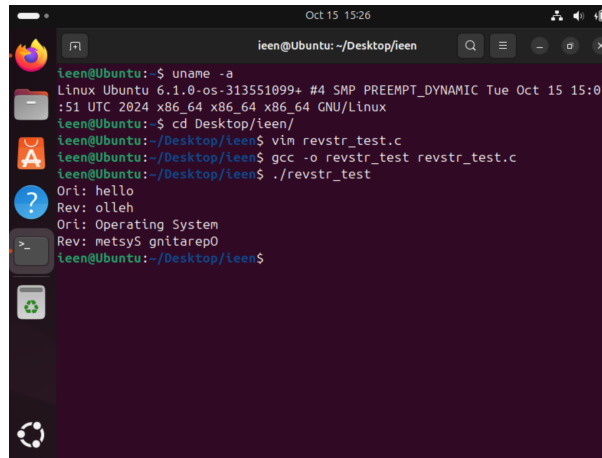
   - Compile and run it.

     ```
     vim revstr_test.c
     gcc -o revstr_test revstr_test.c
     ```
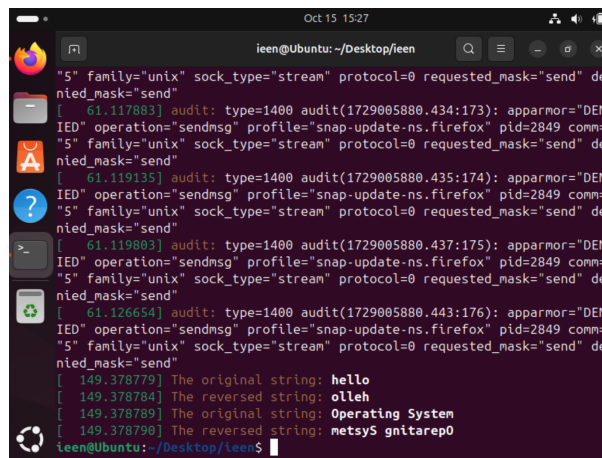
```
./revstr_test
dmesg
```

My result

- The output of the above user-space program:

- the kernel ring buffer for the output messages:

III. Patch

IV. Package list

- Package list

```
dpkg --get-selections > packages_list_313551099.txt
```

- config

```
find ~/linux -name ".config"
```