313551137 官劉翔

II. Compiling the Linux Kernel

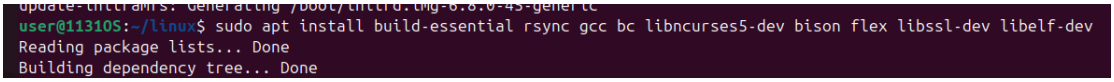
I will follow the following steps to implement:

1. Install tool package
2. Update Grub configuration.
3. Copy existing Linux kernel config file.
4. Custom kernel
5. Compile and build Linux kernel
6. Install Linux kernel and modules (drivers)
7. Reboot

1. Install tool package

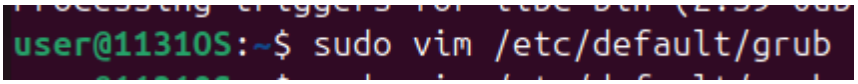
   First, I ensured all necessary tools were installed on my Ubuntu system by using the command:

   sudo apt install build-essential rsync gcc bc libncurses5-dev bison flex libssl-dev libelf-dev

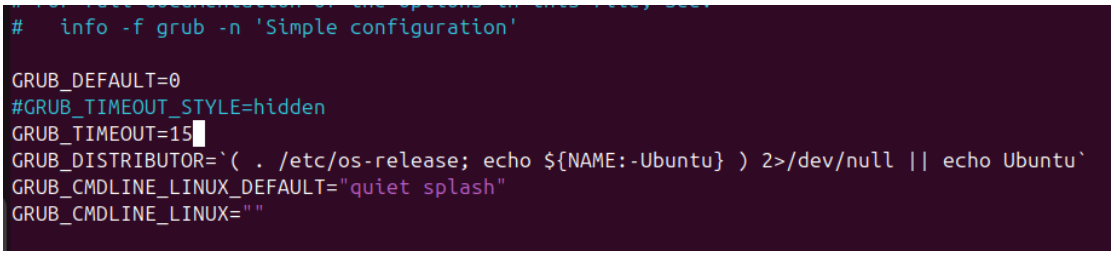


2. Update Grub configuration

   To allow more time for selecting the custom kernel during boot, I modified the GRUB timeout:



   I changed the GRUB_TIMEOUT value to 15 seconds and ignored the GRUB_TIMEOUT_STYLE=hidden command



   After saving the change, I updated GRUB

```
user@11310S:~$ sudo update-grub
Sourcing file `/etc/default/grub'
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-6.8.0-45-generic
Found initrd image: /boot/initrd.img-6.8.0-45-generic
Found memtest86+x64 image: /boot/memtest86+x64.bin
```

When we reboot, we can have 15 second to select which kernel we want to use.

3. Copy existing Linux kernel config file.

   To start with a known working configuration, I copied the existing kernel config to .config:

```
user@11310S:~/linux$ cp -v /boot/config-$(uname -r) .config
'/boot/config-6.8.0-45-generic' -> '.config'
user@11310S:~/linux$
```

   This step ensures that the new kernel will support the existing hardware and maintain system compatibility.

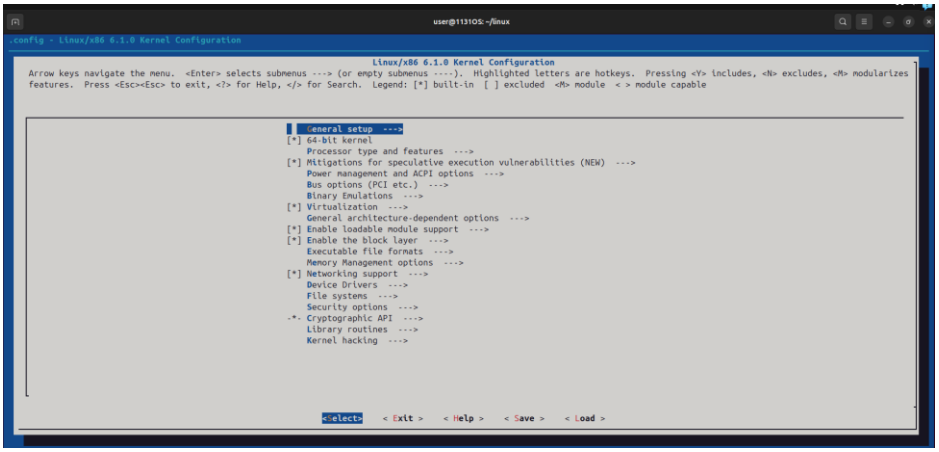
   Starting from scratch could lead to missing critical drivers or features, potentially rendering the system unbootable.

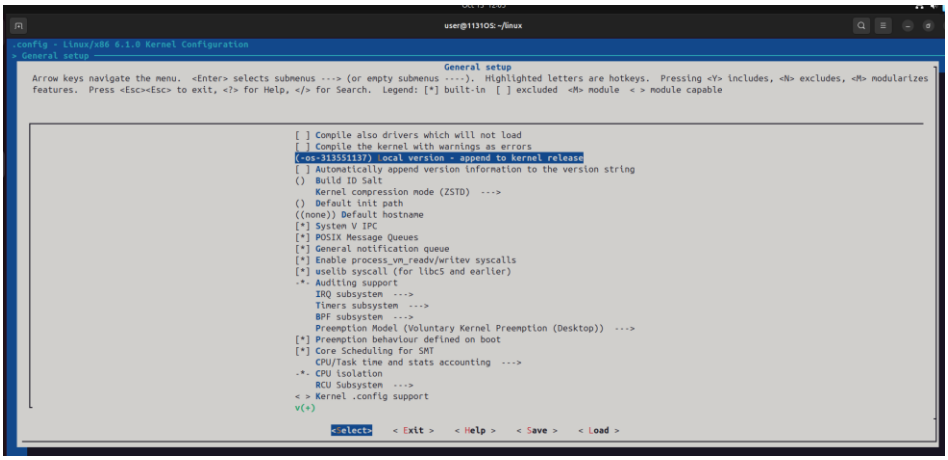
4. Custom kernel

   I customized the kernel configuration by:

```
user@11310S:~/linux$ make menuconfig
  HOSTCC  scripts/basic/fixdep
  UPD     scripts/kconfig/mconf-cfg
  HOSTCC  scripts/kconfig/mconf.o
  HOSTCC  scripts/kconfig/lxdialog/checklist.o
  HOSTCC  scripts/kconfig/lxdialog/inputbox.o
  HOSTCC  scripts/kconfig/lxdialog/menubox.o
  HOSTCC  scripts/kconfig/lxdialog/textbox.o
  HOSTCC  scripts/kconfig/lxdialog/util.o
  HOSTCC  scripts/kconfig/lxdialog/yesno.o
  HOSTCC  scripts/kconfig/confdata.o
  HOSTCC  scripts/kconfig/expr.o
  LEX     scripts/kconfig/lexer.lex.c
```

   In the configuration menu, I navigated to "General setup" -> "Local version - append to kernel release" and set it to "-os-313551137".





5. Compile and build Linux kernel

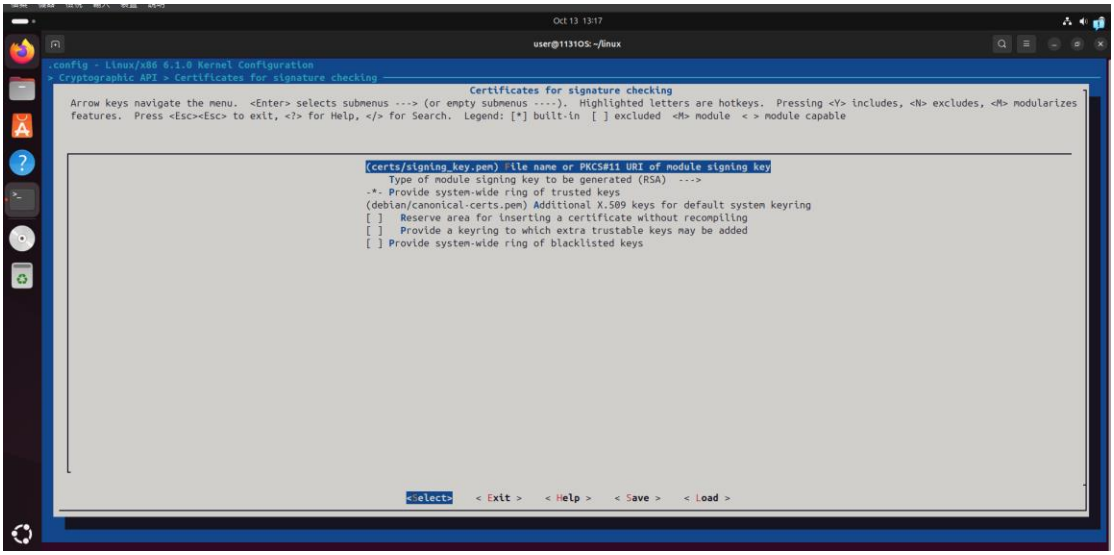
I experimented with three different approaches for this critical step:

Method1(error):

Initially, I attempted to use the make deb-pkg command, which is designed to create Debian packages for the kernel. However, this method encountered errors related to certificate files:

```
user@11310S:~/linux$ make deb-pkg
  SYNC    include/config/auto.conf.cmd
  CC      certs/system_keyring.o
make[5]: *** No rule to make target 'debian/canonical-certs.pem', needed by 'certs/x509_certificate_list'.  Stop.
make[4]: *** [scripts/Makefile.build:500: certs] Error 2
make[3]: *** [Makefile:1992: .] Error 2
make[2]: *** [debian/rules:7: build-arch] Error 2
dpkg-buildpackage: error: debian/rules binary subprocess returned exit status 2
make[1]: *** [scripts/Makefile.package:80: deb-pkg] Error 2
make: *** [Makefile:1636: deb-pkg] Error 2
user@11310S: ~/linux$ make menuconfig
```

Even after disabling related configurations in menuconfig, the errors persisted.



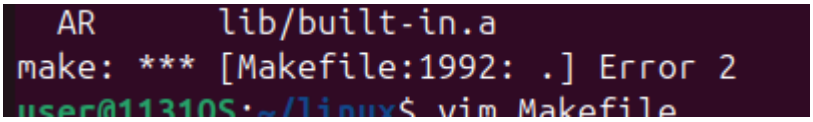
Method2(error?):

Next, I tried using make defconfig to generate a fresh configuration, followed by make -j

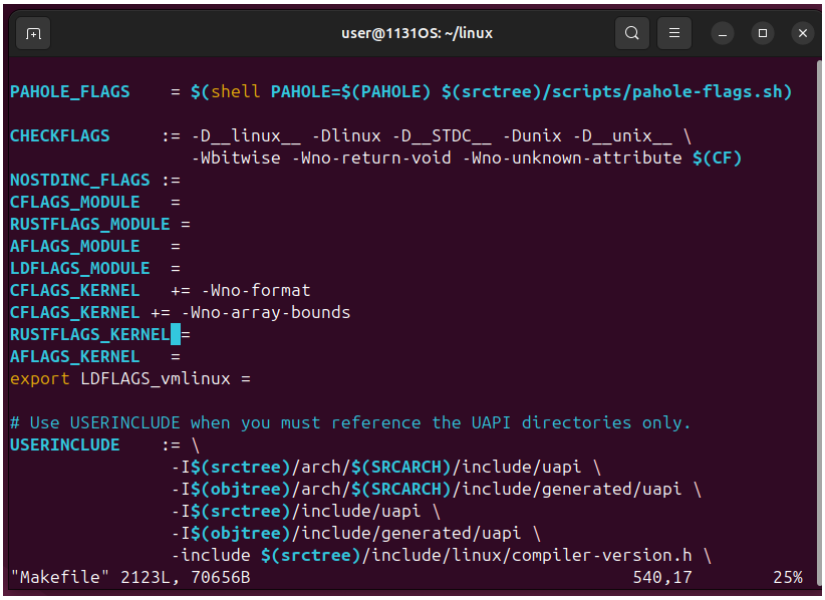
make defconfig

make -j$(nproc)

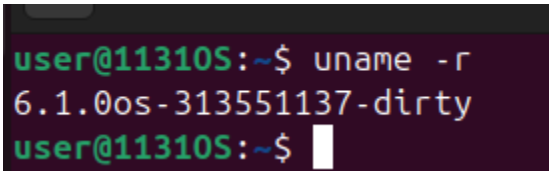
This method resets the kernel configuration to default values, which can be beneficial for starting with a clean slate. However, it resulted in an error in Makefile, and I think it is because default might be incompatible with our specific system or compilation requirements.



Although I change the Makefile and force it to ignore some error by CFLAGS_KERNEL += -Wno-format -Wno-array-bounds.

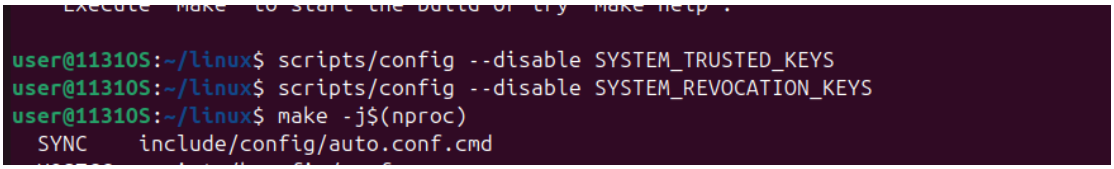


It compile successfully. However, I don't know whether this change effect the program in the future, so I changed to Method 3.



Method3(successful):

This method worked because it combined a known working configuration with targeted disabling of problematic features. The SYSTEM_TRUSTED_KEYS and SYSTEM_REVOCATION_KEYS options, which were causing issues in Method 1, were disabled without compromising the overall kernel functionality. Thus, I execute the following command.



Finally, it is working. I was crying and successfully compiled the kernel.

```
  LD [M]  sound/usb/line6/snd-usb-toneport.ko
  LD [M]  sound/usb/line6/snd-usb-variax.ko
  LD [M]  sound/usb/misc/snd-ua101.ko
  LD [M]  sound/usb/snd-usb-audio.ko
  LD [M]  sound/usb/snd-usbmidi-lib.ko
  LD [M]  sound/usb/usx2y/snd-usb-us122l.ko
  LD [M]  sound/usb/usx2y/snd-usb-usx2y.ko
  LD [M]  sound/virtio/virtio_snd.ko
  LD [M]  sound/x86/snd-hdmi-lpe-audio.ko
  LD [M]  sound/xen/snd_xen_front.ko
  LD [M]  virt/lib/irqbypass.ko
user@11310S:~/linux$
```

6. Install Linux kernel and modules:

   After successful compilation,

   a. Install the required modules:

      sudo make modules_install

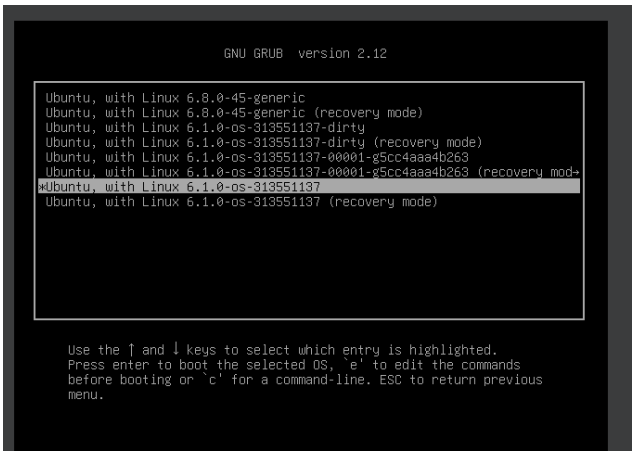
   b. Install the kernel by typing:

      sudo make install

7. Reboot and Result:

   Finally, I rebooted the system to load the new kernel.

   After I reboot, the selection menu show



   In the end, I check my kernel version.

```
user@1131OS: ~
user@1131OS:~$ uname -a
Linux 1131OS 6.1.0-os-313551137 #1 SMP PREEMPT_DYNAMIC Mon Oct 14 12:01:46 UTC 2024 x86_64 x86_64 x86_64 GNU/Linux
user@1131OS:~$ 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
user@1131OS:~$
```

III. Implementing a new System Calls

Step1~4 execute in ~/linux.

1. Kernel Implementation:

   In **kernel/sys.c,** implement the system call:

   Besides, I also add some code to prevent common errors happening

```c
}

SYSCALL_DEFINE2(NR_revstr, char __user *, str, size_t, n)
{
    char *k_str;
    int i, j;
    char temp;

    k_str = kmalloc(n + 1, GFP_KERNEL);
    if (!k_str)
        return -ENOMEM;

    if (copy_from_user(k_str, str, n)) {
        kfree(k_str);
        return -EFAULT;
    }
    k_str[n] = '\0';

    printk(KERN_INFO "The origin string: %s\n", k_str);

    for (i = 0, j = n - 1; i < j; i++, j--) {
        temp = k_str[i];
        k_str[i] = k_str[j];
        k_str[j] = temp;
    }

    printk(KERN_INFO "The reversed string: %s\n", k_str);

    if (copy_to_user(str, k_str, n)) {
        kfree(k_str);
        return -EFAULT;
    }

    kfree(k_str);
    return 0;
}
#endif /* CONFIG_COMPAT */
"kernel/sys.c" 2828L, 67553B
```

   a. SYSCALL_DEFINE2(NR_revstr, char __user *, str, size_t, n):
      1. This macro defines a system call named NR_revstr with two arguments.
      2. char __user *str: A pointer to a user-space string that needs to be reversed.
      3. size_t n: The length of the string.
   b. Variable declarations:
      1. char *k_str: A kernel-space buffer to store the string.

2. int i, j: Loop counters for string reversal.
3. char temp: Temporary variable for swapping characters.

c. k_str = kmalloc(n + 1, GFP_KERNEL):
   1. Allocates n + 1 bytes of memory in kernel space. (The extra byte is for the null terminator.)
   2. GFP_KERNEL is the allocation flag for normal kernel allocations.

d. if (!k_str) return -ENOMEM:
   1. If memory allocation fails, return an "Out of memory" error.

e. if (copy_from_user(k_str, str, n)) { ... }:
   1. Copies n bytes from the user-space string to the kernel-space buffer.
   2. If copying fails, free the allocated memory and return an error.

f. k_str[n] = '\0':
   1. Null-terminates the kernel-space string.

g. printk(KERN_INFO "The origin string: %s\n", k_str):
   1. Logs the original string to the kernel log.

h. String reversal loop:
   1. for (i = 0, j = n - 1; i < j; i++, j--) {
   2. Reverses the string by swapping characters from both ends towards the middle.

i. printk(KERN_INFO "The reversed string: %s\n", k_str):
   1. Logs the reversed string to the kernel log.

j. if (copy_to_user(str, k_str, n)) { ... }:
   1. Copies the reversed string back to user space.
   2. If copying fails, free the allocated memory and return an error.

k. kfree(k_str):
   1. Frees the allocated kernel memory.

l. return 0:
   1. Returns 0 to indicate successful execution of the system call.

2. System Call Prototype:

   Add the system call prototype in **include/linux/syscalls.h**

```
/* obsolete: kernel/sys.c */
asmlinkage long sys_gethostname(char __user *name, int len);
asmlinkage long sys_uname(struct old_utsname __user *);
asmlinkage long sys_olduname(struct oldold_utsname __user *);
asmlinkage long sys_NR_revstr(char __user *str, size_t n);
#ifdef __ARCH_WANT_SYS_OLD_GETRLIMIT
asmlinkage long sys_old_getrlimit(unsigned int resource, struct rlimit __user *rl
#endif

/* obsolete: ipc */
asmlinkage long sys_ipc(unsigned int call, int first, unsigned long second,
                unsigned long third, void __user *ptr, long fifth);

/* obsolete: mm/ */
asmlinkage long sys_mmap_pgoff(unsigned long addr, unsigned long len,
                       unsigned long prot, unsigned long flags,
                       unsigned long fd, unsigned long pgoff);
asmlinkage long sys_old_mmap(struct mmap_arg_struct __user *arg);


/*
 * Not a real system call, but a placeholder for syscalls which are
 * not implemented -- see kernel/sys_ni.c
"include/linux/syscalls.h" 1389L, 56929B
```

3. Set system Call Table Entry:

   Based on the context of the kernel document for x86

> x86 System Call Implementation
>
> To wire up your new system call for x86 platforms, you need to update the master syscall tables. Assuming your new system call isn't special in some way (see below), this involves a "common" entry (for x86_64 and x32) in arch/x86/entry/syscalls/syscall_64.tbl:

   Add the system call number in **arch/x86/entry/syscalls/syscall_64.tbl**:

```
448     common  process_mrelease        sys_process_mrelease
449     common  futex_waitv             sys_futex_waitv
450     common  set_mempolicy_home_node sys_set_mempolicy_home_node
451     common  NR_revstr               sys_NR_revstr
#
# Due to a historical design error, certain syscalls are numbered diff
# in x32 as compared to native x86_64.  These syscalls have numbers 51
# Do not add new syscalls to this range.  Numbers 548 and above are av
# for non-x32 use.
#
512     x32     rt_sigaction            compat_sys_rt_sigaction
513     x32     rt_sigreturn            compat_sys_x32_rt_sigreturn
514     x32     ioctl                   compat_sys_ioctl
515     x32     readv                   sys_readv
516     x32     writev                  sys_writev
517     x32     recvfrom                compat_sys_recvfrom
518     x32     sendmsg                 compat_sys_sendmsg
519     x32     recvmsg                 compat_sys_recvmsg
520     x32     execve                  compat_sys_execve
521     x32     ptrace                  compat_sys_ptrace
522     x32     rt_sigpending           compat_sys_rt_sigpending
523     x32     rt_sigtimedwait         compat_sys_rt_sigtimedwait_tim
"arch/x86/entry/syscalls/syscall_64.tbl" 419L, 14891B
```

   We set the new system call in 451 entry.

4. Re-compile kernel:

   I use the method in Part I step3~step7. Recap:

   Step 3: cp -v /boot/config-$(uname -r) .config

   Step 4: make menuconfig

   Step 5: make -j$(nproc)

   Step 6: sudo make modules_install

   sudo make install

   Step 7: reboot

   Recompile result

```
user@1131OS:~$ uname -a
Linux 1131OS 6.1.0-os-313551137+ #2 SMP PREEMPT_DYNAMIC Mon Oct 14 16:26:10 UTC
2024 x86_64 x86_64 x86_64 GNU/Linux
```

5. Create && Compile && Run the Test Program:

   I use vim to create a test program test_NR_revstr.c

```
user@1131OS:~/HW1$ vim test_NR_revstr.c
user@1131OS:~/HW1$ gcc -o test_NR_revstr test_NR_revstr.c
user@1131OS:~/HW1$ ./test_NR_revstr
```

   Running Test result:

```
user@1131OS:~/HW1$ ./test_NR_revstr
Ori: hello
Rev: olleh
Ori: Operating System
Rev: metsyS gnitarepO
```

6. Check Kernel Logs

```
user@1131OS:~/HW1$ sudo dmesg | tail
[sudo] password for user:
[  305.992096] audit: type=1400 audit(1728926796.486:165): apparmor="DENIED" ope
ration="open" profile="snap.snapd-desktop-integration.snapd-desktop-integration"
 name="/usr/share/glib-2.0/schemas/" pid=8376 comm="desktop-launch" requested_ma
sk="r" denied_mask="r" fsuid=1000 ouid=0
[  306.005374] audit: type=1400 audit(1728926796.492:166): apparmor="DENIED" ope
ration="open" profile="snap.snapd-desktop-integration.snapd-desktop-integration"
 name="/usr/share/glib-2.0/schemas/" pid=8379 comm="desktop-launch" requested_ma
sk="r" denied_mask="r" fsuid=1000 ouid=0
[  306.446389] audit: type=1400 audit(1728926796.944:167): apparmor="DENIED" ope
ration="open" profile="snap.snapd-desktop-integration.snapd-desktop-integration"
 name="/etc/fonts/snap-override/" pid=8400 comm=5B70616E676F5D204663496E6974 req
uested_mask="r" denied_mask="r" fsuid=1000 ouid=0
[  307.192204] rfkill: input handler disabled
[  307.381632] ISO 9660 Extensions: Microsoft Joliet Level 3
[  307.382127] ISO 9660 Extensions: RRIP_1991A
[  641.407387] The origin string: hello
[  641.407393] The reversed string: olleh
[  641.407408] The origin string: Operating System
[  641.407409] The reversed string: metsyS gnitarepO
```

IV. Patch

1. Check initial status:

```
user@1131OS:~/linux$ git status
HEAD detached at v6.1
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   arch/x86/entry/syscalls/syscall_64.tbl
        modified:   include/linux/syscalls.h
        modified:   kernel/sys.c
```

2. Stage all changes:

```
no changes added to commit (use "git add" and/or "git commit -a")
user@11310S:~/linux$ git add .
user@11310S:~/linux$ git commit -m "Add NR_revstr system call"
[detached HEAD 38cb96a0a631] Add NR_revstr system call
 3 files changed, 37 insertions(+), 1 deletion(-)
user@11310S:~/linux$ git format-patch -1 HEAD
0001-Add-NR_revstr-system-call.patch
```

I. Git add .:

This command stages all modified and new files in the current directory and its subdirectories. The dot (.) means "current directory and everything below it".

II. git commit -m "Add revstr system call":

This makes a new commit with all your staged changes.

III. git format-patch -1 HEAD:

creates a patch file for the most recent commit. The -1 flag means "create a patch for the last 1 commit", and HEAD refers to the most recent commit on the current branch.

3. Final check:

```
user@11310S:~/linux$ git status
HEAD detached from v6.1
nothing to commit, working tree clean
```