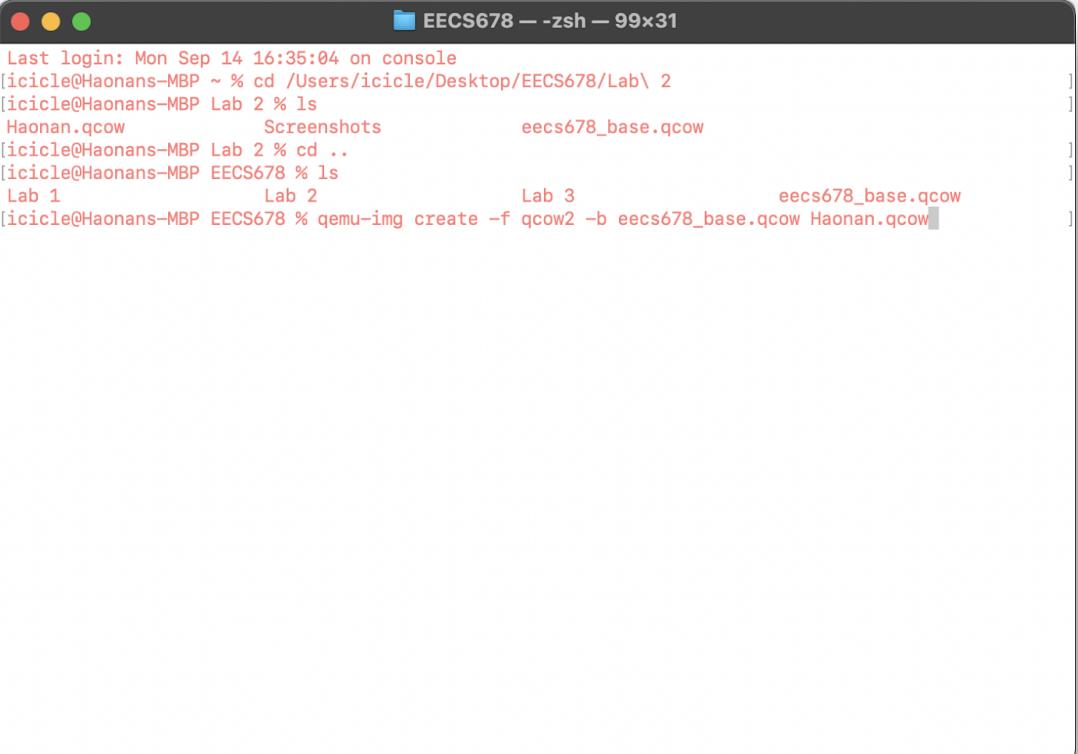


Haonan Hu

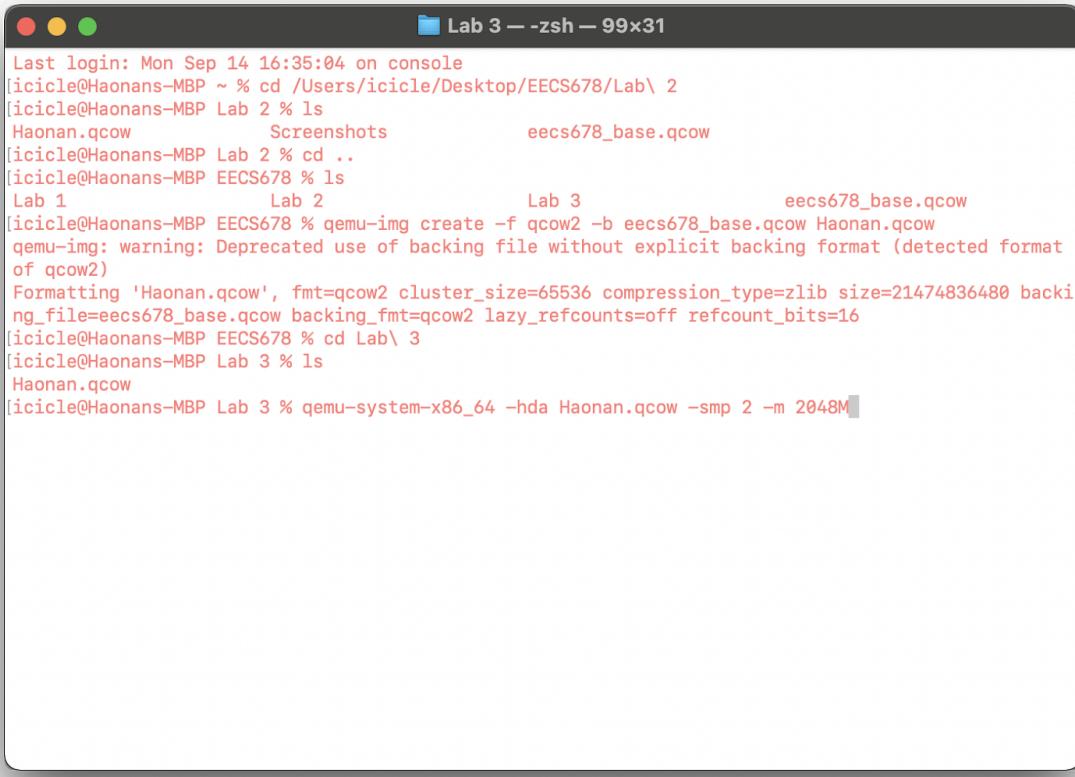
2863545

18, Sep, 2020



```
Last login: Mon Sep 14 16:35:04 on console
[icicle@Haonans-MBP ~ % cd /Users/icicle/Desktop/EECS678/Lab\ 2
[icicle@Haonans-MBP Lab 2 % ls
Haonan.qcow           Screenshots          eecs678_base.qcow
[icicle@Haonans-MBP Lab 2 % cd ..
[icicle@Haonans-MBP EECS678 % ls
Lab 1                 Lab 2                 Lab 3                 eecs678_base.qcow
[icicle@Haonans-MBP EECS678 % qemu-img create -f qcow2 -b eecs678_base.qcow Haonan.qcow
```

This will create a differential image of the base image in my first name



```
Last login: Mon Sep 14 16:35:04 on console
[icicle@Haonans-MBP ~ % cd /Users/icicle/Desktop/EECS678/Lab\ 2
[icicle@Haonans-MBP Lab 2 % ls
Haonan.qcow           Screenshots          eecs678_base.qcow
[icicle@Haonans-MBP Lab 2 % cd ..
[icicle@Haonans-MBP EECS678 % ls
Lab 1                 Lab 2                 Lab 3                 eecs678_base.qcow
[icicle@Haonans-MBP EECS678 % qemu-img create -f qcow2 -b eecs678_base.qcow Haonan.qcow
qemu-img: warning: Deprecated use of backing file without explicit backing format (detected format
of qcow2)
Formatting 'Haonan.qcow', fmt=qcow2 cluster_size=65536 compression_type=zlib size=21474836480 backi
ng_file=eecs678_base.qcow backing_fmt=qcow2 lazy_refcounts=off refcount_bits=16
[icicle@Haonans-MBP EECS678 % cd Lab\ 3
[icicle@Haonans-MBP Lab 3 % ls
Haonan.qcow
[icicle@Haonans-MBP Lab 3 % qemu-system-x86_64 -hda Haonan.qcow -smp 2 -m 2048M
```

The command will open up a GUI for booting the image and give it 2Gigabytes of ram to use

```
QEMU - (Press ctrl + alt + g to release Mouse)
SeaBIOS (version rel-1.13.0-48-gd9c812dda519-prebuilt.qemu.org)
Decompressing Linux... Parsing ELF... done.
Booting the kernel.
iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP PMM+7FF8F390+7FEF390 CA00
Debian GNU/Linux 6.0 debian tty1

debian login: root
Password: om Hard Disk...
Last login: Tue Nov 27 18:45:21 CST 2018 on tty1
Linux debian 2.6.32-5-686 #1 SMP Mon Sep 23 23:00:18 UTC 2013 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

[~]
root@debian$ adduser haonan
```

Add a user with my name from root

The screenshot shows a terminal window titled "QEMU" running a Vim editor. The file being edited is a sudoers configuration. The content of the file is as follows:

```
1 # User privilege specification
2 root ALL=(ALL) ALL
3 haonan ALL=(ALL) ALL
```

The cursor is positioned at the end of the third line. The status bar at the bottom left indicates "INSERT" mode. The status bar also shows the current line number (3), column number (21), and the word "All".

Add user to the sudoer file through vim then quit by command :wq



The screenshot shows a terminal window titled "QEMU" running on a Mac OS X host. The terminal is black with white text. It displays the following command being entered by the root user:

```
"/etc/sudoers" [New] 3L, 71C written
[~]
root@debian$ usermod -a -G sudo haonan
```

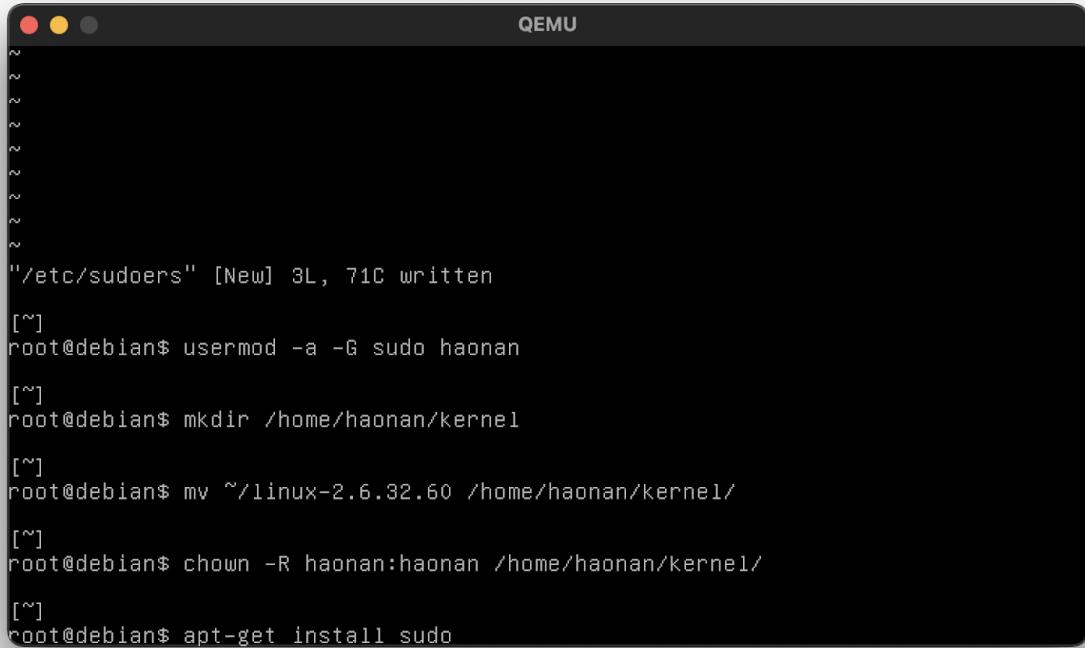
Add user haonan to the sudo group



The screenshot shows a terminal window titled "QEMU" running on a Mac OS X host. The terminal is a black box with white text. It displays the following command-line session:

```
"/etc/sudoers" [New] 3L, 71C written
[~]
root@debian$ usermod -a -G sudo haonan
[~]
root@debian$ mkdir /home/haonan/kernel
[~]
root@debian$ mv ~/linux-2.6.32.60 /home/haonan/kernel/
[~]
root@debian$ chown -R haonan:haonan /home/haonan/kernel/_
```

Move the kernel source and the config file to the user's directory and change the ownership to myself



The screenshot shows a terminal window titled "QEMU" running on a Linux host. The terminal is displaying a root shell session on a Debian system. The user has performed the following steps:

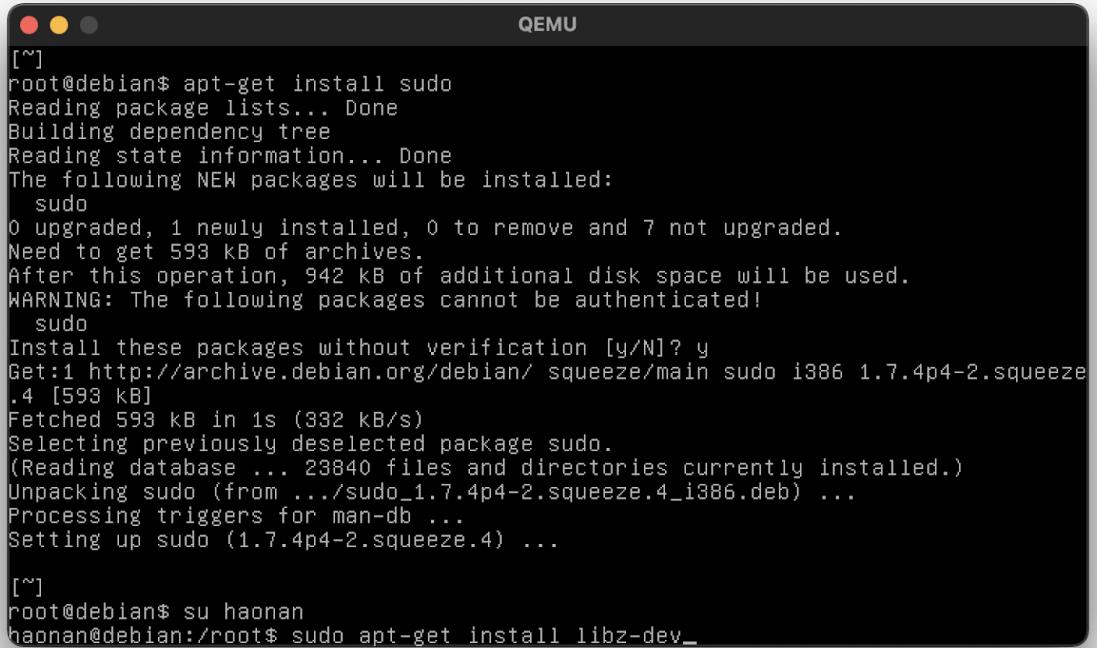
- Edited the "/etc/sudoers" file.
- Added the user "haonan" to the "sudo" group using the command "usermod -a -G sudo haonan".
- Made a directory structure at "/home/haonan/kernel".
- Moved a kernel file from the current directory to the new directory: "mv ~/linux-2.6.32.60 /home/haonan/kernel/".
- Changed ownership of the moved kernel file to the "haonan" user and group using "chown -R haonan:haonan /home/haonan/kernel/".
- Installed the "sudo" package using "apt-get install sudo".

Install Sudo on kernel in orer to grant user right to execute some command

```
[~]
root@debian$ apt-get install sudo
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  sudo
0 upgraded, 1 newly installed, 0 to remove and 7 not upgraded.
Need to get 593 kB of archives.
After this operation, 942 kB of additional disk space will be used.
WARNING: The following packages cannot be authenticated!
  sudo
Install these packages without verification [y/N]? y
Get:1 http://archive.debian.org/debian/ squeeze/main sudo i386 1.7.4p4-2.squeeze
.4 [593 kB]
Fetched 593 kB in 1s (332 kB/s)
Selecting previously deselected package sudo.
(Reading database ... 23840 files and directories currently installed.)
Unpacking sudo (from .../sudo_1.7.4p4-2.squeeze.4_i386.deb) ...
Processing triggers for man-db ...
Setting up sudo (1.7.4p4-2.squeeze.4) ...

[~]
root@debian$ su haonan_
```

Switch user from root to the new user(me)



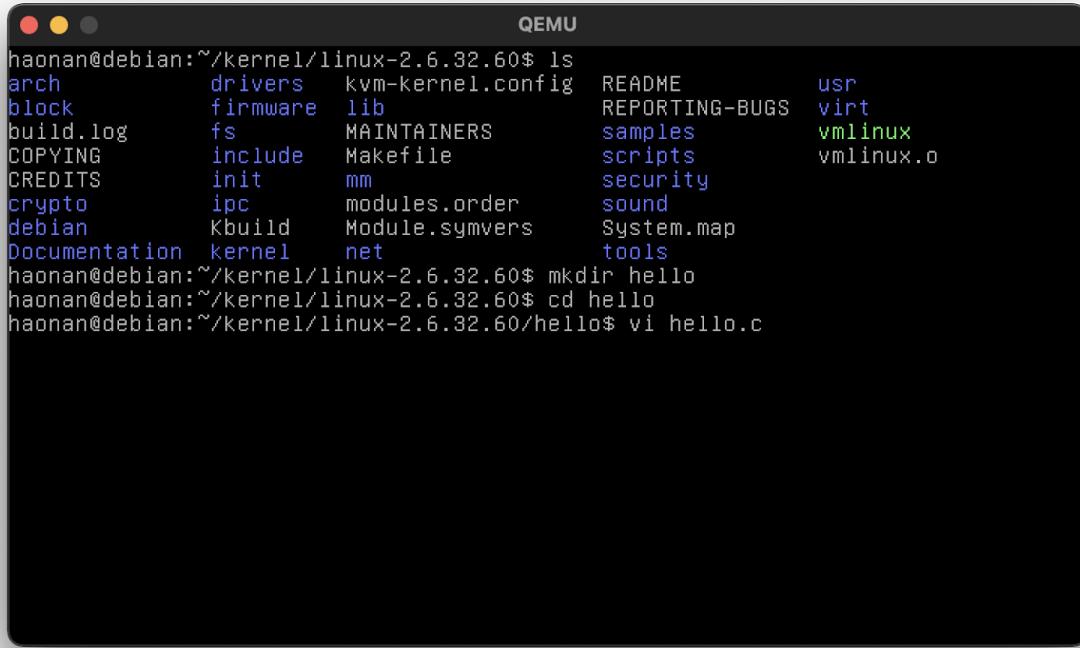
```
[~]
root@debian$ apt-get install sudo
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  sudo
0 upgraded, 1 newly installed, 0 to remove and 7 not upgraded.
Need to get 593 kB of archives.
After this operation, 942 kB of additional disk space will be used.
WARNING: The following packages cannot be authenticated!
  sudo
Install these packages without verification [y/N]? y
Get:1 http://archive.debian.org/debian/ squeeze/main sudo i386 1.7.4p4-2.squeeze
.4 [593 kB]
Fetched 593 kB in 1s (332 kB/s)
Selecting previously deselected package sudo.
(Reading database ... 23840 files and directories currently installed.)
Unpacking sudo (from .../sudo_1.7.4p4-2.squeeze.4_i386.deb) ...
Processing triggers for man-db ...
Setting up sudo (1.7.4p4-2.squeeze.4) ...

[~]
root@debian$ su haonan
haonan@debian:/root$ sudo apt-get install libbz-dev
```

Install libbz-dev by sudo command



Go to the directory where linux-2.6.32.60 located



The screenshot shows a terminal window titled "QEMU" running on a Debian system. The user is navigating through the Linux kernel source code directory at "/kernel/linux-2.6.32.60". The terminal output is as follows:

```
haonan@debian:~/kernel/linux-2.6.32.60$ ls
arch      drivers  kvm-kernel.config  README      usr
block     firmware lib          REPORTING-BUGS  virt
build.log   fs      MAINTAINERS    samples    vmlinuz
COPYING    include  Makefile       scripts    vmlinuz.o
CREDITS    init     mm          security
crypto     ipc      modules.order sound
debian     Kbuild  Module.symvers System.map
Documentation  kernel  net          tools
haonan@debian:~/kernel/linux-2.6.32.60$ mkdir hello
haonan@debian:~/kernel/linux-2.6.32.60$ cd hello
haonan@debian:~/kernel/linux-2.6.32.60/hello$ vi hello.c
```

Create a directory called hello and then create a c program called hello.c under this directory.

```
QEMU - (Press ctrl + alt + g to release Mouse)
#include <linux/kernel.h>
asm linkage long sys_hello(void)
{
    printk("Hello world\n");
    return 100;
}
-- INSERT --          7,2          All
```

Add lines to print hello world in hello.c file

The screenshot shows a terminal window titled "QEMU". Inside the terminal, the file "hello.c" is displayed. The code defines a function "sys_hello" with an assembly linkage, which prints "Hello world\n" and returns 100. The terminal also shows the file was written and lists the current directory as "/kernel/linux-2.6.32.60/hello\$".

```
asm linkage long sys_hello(void)
{
    printk("Hello world\n");
    return 100;
}

"hello.c" [New] 7L, 102C written
haonan@debian:~/kernel/linux-2.6.32.60/hello$ vi Makefile_
```

Create a makefile



```
QEMU
obj-y := hello.o
-- INSERT --          1,17      All
```

Add content to Makefile

```
mod_strip_cmd = $(STRIP) --strip-debug
else
mod_strip_cmd = $(STRIP) $(INSTALL_MOD_STRIP)
endif # INSTALL_MOD_STRIP=1
else
mod_strip_cmd = true
endif # INSTALL_MOD_STRIP
export mod_strip_cmd

ifeq ($(KBUILD_EXTMOD),)
core-y      += kernel/ mm/ fs/ ipc/ security/ crypto/ block/ hello/
vmlinux-dirs := $(patsubst %/,%,$(filter %, $(init-y) $(init-m) \
$(core-y) $(core-m) $(drivers-y) $(drivers-m) \
$(net-y) $(net-m) $(libs-y) $(libs-m)))
vmlinux-alldirs := $(sort $(vmlinux-dirs) $(patsubst %/,%,$(filter %/, \
$(init-n) $(init-) \
$(core-n) $(core-) $(drivers-n) $(drivers-) \
$(net-n) $(net-) $(libs-n) $(libs-))))
init-y      := $(patsubst %/, %/built-in.o, $(init-y))
core-y      := $(patsubst %/, %/built-in.o, $(core-y))
-- INSERT --
```

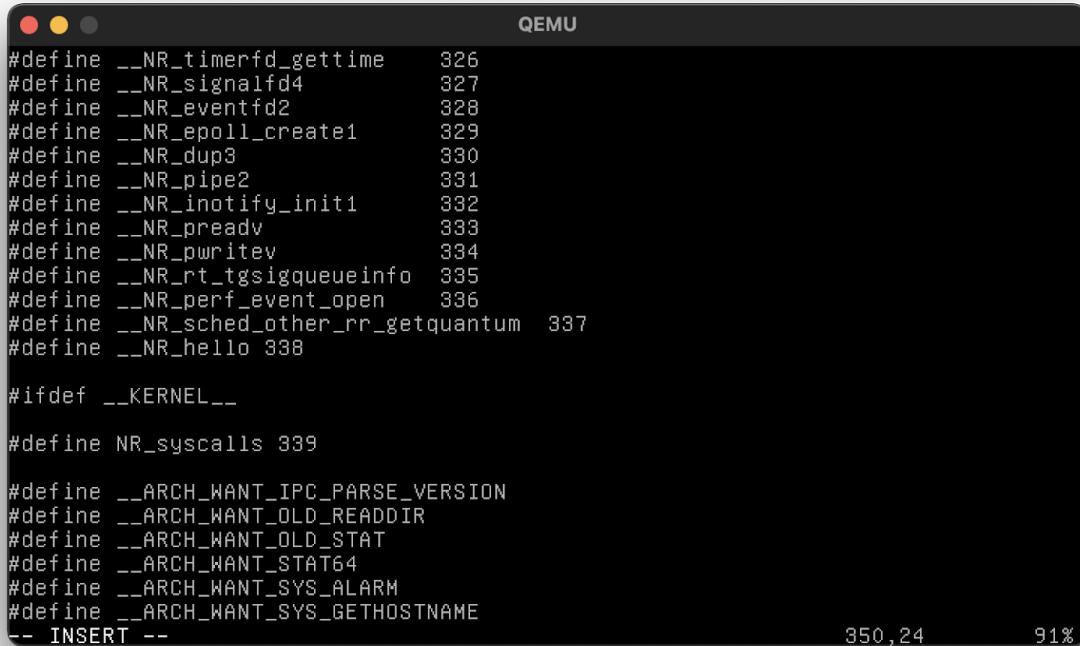
650,64-72 40%

Locate Makefile under linux-2.6.32.60 directory and open it then searching for core-y then edit it with adding hello at the end

The screenshot shows a terminal window titled "QEMU" displaying a list of system call entries. Each entry consists of a .long instruction followed by a symbol name and a comment indicating its address. The symbols listed are: sys_tee, sys_vmsplice, sys_move_pages, sys_getcpu, sys_epoll_pwait, sys_utimensat, sys_signalfd, sys_timerfd_create, sys_eventfd, sys_fallocate, sys_timerfd_settime, sys_timerfd_gettime, sys_signalfd4, sys_eventfd2, sys_epoll_create1, sys_dup3, sys_pipe2, sys_inotify_init1, sys_preadv, sys_pwritev, sys_rt_tgsigqueueinfo, sys_perf_event_open, sys_sched_other_rr_getquantum, and sys_hello_. At the bottom left, there is a prompt "-- INSERT --". On the right side, the text "340,17-24" and "Bot" are displayed.

```
.long sys_tee          /* 315 */
.long sys_vmsplice
.long sys_move_pages
.long sys_getcpu
.long sys_epoll_pwait
.long sys_utimensat     /* 320 */
.long sys_signalfd
.long sys_timerfd_create
.long sys_eventfd
.long sys_fallocate
.long sys_timerfd_settime    /* 325 */
.long sys_timerfd_gettime
.long sys_signalfd4
.long sys_eventfd2
.long sys_epoll_create1
.long sys_dup3           /* 330 */
.long sys_pipe2
.long sys_inotify_init1
.long sys_preadv
.long sys_pwritev
.long sys_rt_tgsigqueueinfo /* 335 */
.long sys_perf_event_open
.long sys_sched_other_rr_getquantum
.long sys_hello_
```

Go to directory “ arch/x86/kernel” find the file `syscall_table_32.S`, add `.long sys_hello` at the end of the file.



The screenshot shows a terminal window titled "QEMU". The content of the terminal is as follows:

```
#define __NR_timerfd_gettime    326
#define __NR_signalfd4          327
#define __NR_eventfd2           328
#define __NR_epoll_create1       329
#define __NR_dup3                330
#define __NR_pipe2               331
#define __NR_inotify_init1       332
#define __NR_preadv              333
#define __NR_pwritev             334
#define __NR_rt_tgsigqueueinfo   335
#define __NR_perf_event_open      336
#define __NR_sched_other_rr_getquantum 337
#define __NR_hello 338

#ifndef __KERNEL__
#define NR_syscalls 339

#define __ARCH_WANT_IPC_PARSE_VERSION
#define __ARCH_WANT_OLD_REaddir
#define __ARCH_WANT_OLD_STAT
#define __ARCH_WANT_STAT64
#define __ARCH_WANT_SYS_ALARM
#define __ARCH_WANT_SYS_GETHOSTNAME
-- INSERT --
```

The terminal also displays status information at the bottom right: "350, 24" and "91%".

Now go to “include/asm” directory and find the file “unistd_32.h” then go to the bottom by typing in G and add #define __NR_hello 338 at the bottom of #define
__NR_sched_other_rr_getquantum 337, then change #define NR_sysvals 338 to 339.

```
QEMU
asmlinkage long sys_old_readdir(unsigned int, struct old_linux_dirent __user *,
unsigned int);
asmlinkage long sys_pselect6(int, fd_set __user *, fd_set __user *,
fd_set __user *, struct timespec __user *,
void __user *);
asmlinkage long sys_ppoll(struct pollfd __user *, unsigned int,
struct timespec __user *, const sigset_t __user *,
size_t);

int kernel_execve(const char *filename, char *const argv[], char *const envp[]);

asmlinkage long sys_perf_event_open(
    struct perf_event_attr __user *attr_uptr,
    pid_t pid, int cpu, int group_fd, unsigned long flags);

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_sched_other_rr_getquantum(void);

asmlinkage long sys_hello(void);
#endif
-- INSERT --
```

892,7

Bot

Locate "include/linux", open file syscalls.h and add asmlinkage long sys_hello(void) the the end of the file

```
QEMU
asm linkage long sys_pselect6(int, fd_set __user *, fd_set __user *,
                               fd_set __user *, struct timespec __user *,
                               void __user *);
asm linkage long sys_ppoll(struct pollfd __user *, unsigned int,
                           struct timespec __user *, const sigset_t __user *,
                           size_t);
int kernel_execve(const char *filename, char *const argv[], char *const envp[]);

asm linkage long sys_perf_event_open(
    struct perf_event_attr __user *attr_uptr,
    pid_t pid, int cpu, int group_fd, unsigned long flags);

asm linkage long sys_mmap_pgoff(unsigned long addr, unsigned long len,
                                unsigned long prot, unsigned long flags,
                                unsigned long fd, unsigned long pgoff);

asm linkage long sys_sched_other_rr_getquantum(void);

asm linkage long sys_hello(void);
#endif
"syscalls.h" 892L, 38632C written
haonan@debian:~/Kernel/linux-2.6.32.60/include/linux$ sudo vi /usr/bin/kvm-kerne
l-build
```

Add level of concurrency to the building process

The screenshot shows a terminal window titled "QEMU - (Press ctrl + alt + g to release Mouse)". The window contains a shell script with the following content:

```
1#!/bin/sh
2 rev=$1
3 if [ -z "$rev" ]; then
4     echo "Usage: build64 <revision>"
5     exit 1
6 fi
7 make-kpkg -j 2 --rootcmd fakeroot --initrd --revision=$rev kernel_image 2>&1
```

The command "make-kpkg" is highlighted in blue. The window has a dark background with light-colored text. The bottom right corner shows the status bar with "7,14" and "All".

Add -j flag with value to the make-kpkg command at the end

Start build the kernel

```
QEMU
make[2]: Leaving directory `/home/haonan/kernel/linux-2.6.32.60'
make[1]: Leaving directory `/home/haonan/kernel/linux-2.6.32.60'
haonan@debian:~/kernel/linux-2.6.32.60$ su
Password:

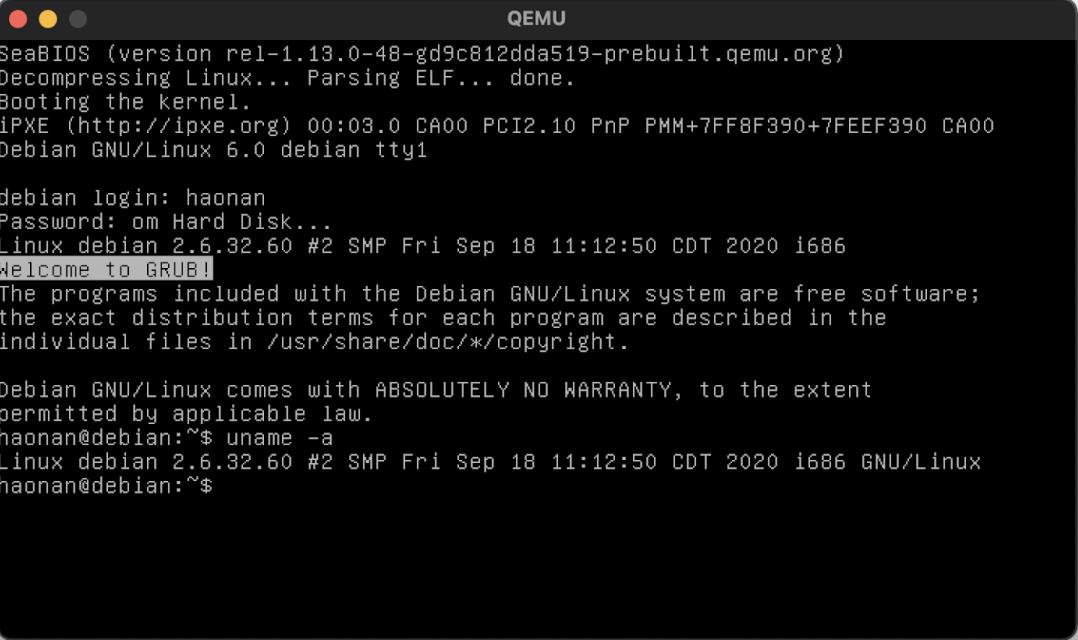
[/home/haonan/kernel/linux-2.6.32.60]
root@debian$ ls
arch/           drivers/   kernel/      net/          tools/
block/          firmware/ kvm-kernel.config README        usr/
build.log       fs/        lib/         REPORTING-BUGS virt/
COPYING         hello/    MAINTAINERS  samples/      vmlinu*x
CREDITS         include/   Makefile    scripts/     vmlinux.o
crypto/         init/     mm/         security/
debian/         ipc/      modules.order sound/      System.map
Documentation/ Kbuild    Module.symvers

[/home/haonan/kernel/linux-2.6.32.60]
root@debian$ cd ..

[/home/haonan/kernel]
root@debian$ ls
linux-2.6.32.60/  linux-image-2.6.32.60_1_i386.deb

[/home/haonan/kernel]
root@debian$ dpkg -i linux-image-2.6.32.60_1_i386.deb _
```

Go back to previous directory, and then start installing



The screenshot shows a terminal window titled "QEMU" running on a SeaBIOS boot interface. The screen displays the following text:

```
SeaBIOS (version rel-1.13.0-48-gd9c812dda519-prebuilt.qemu.org)
Decompressing Linux... Parsing ELF... done.
Booting the kernel.
iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP PMM+7FF8F390+7FEF390 CA00
Debian GNU/Linux 6.0 debian tty1

debian login: haonan
Password: om Hard Disk...
Linux debian 2.6.32.60 #2 SMP Fri Sep 18 11:12:50 CDT 2020 i686
Welcome to GRUB!
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
haonan@debian:~$ uname -a
Linux debian 2.6.32.60 #2 SMP Fri Sep 18 11:12:50 CDT 2020 i686 GNU/Linux
haonan@debian:~$
```

After rebooting, showed my identity and my system version

```
QEMU - (Press ctrl + alt + g to release Mouse)

#include <stdio.h>
#include <sys/syscall.h>
#include <unistd.h>
#include <linux/kernel.h>

int main()
{
    long int syscall_val = syscall(388);
    printf("System call sys_hello returned %ld\n", syscall_val);
    return 0;
}

-- INSERT --           11,2          All
```

Make a test file to test syscall

The screenshot shows a terminal window titled "QEMU" running on a Mac OS X desktop. The terminal displays the source code of a C program named "test_syscall.c". The code includes a system call to "sys_hello" and prints its return value. Below the code, the terminal shows the compilation command "gcc test_syscall.c -o test_syscall", the execution command "./test_syscall", and the resulting output "System call sys_hello returned 100".

```
int main()
{
    long int syscall_val = syscall(338);
    printf("System call sys_hello returned %ld\n", syscall_val);
    return 0;
}

"test_syscall.c" [New] 10L, 216C written
haonan@debian:~$ gcc test_syscall.c -o test_syscall
haonan@debian:~$ ./test_syscall
System call sys_hello returned 100
haonan@debian:~$ _
```

Compiling test file and run it, it appears 100 which is good, means our syscall is working

In the kernel log, It prints Hello world.