

HW2 M11315051 林子新

Hardware :

1. CPU : i5-12400
2. GPU : RTX4070
3. RAM : 32GB (partition 512 MB for Virtual Machine)
4. SSDd : 1TB (partition 60GB for Virtual Machine)


Software :

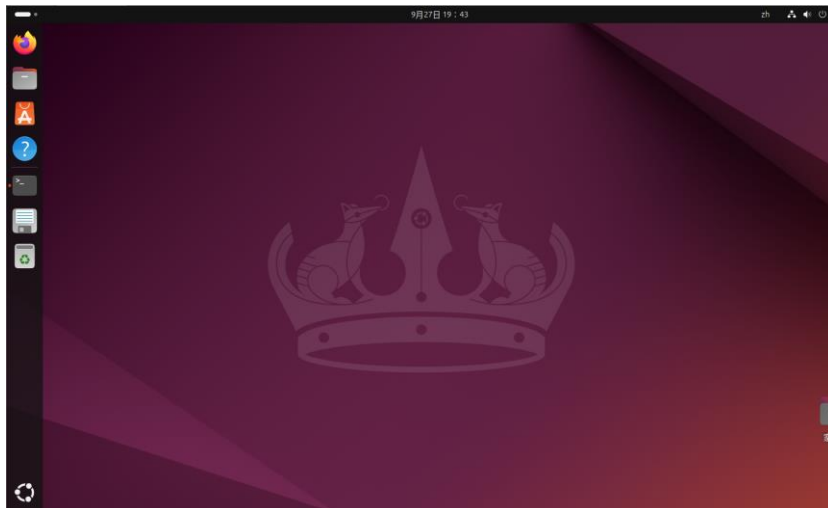
1. VMware(虛擬環境)
2. ubuntu-24.04.1-desktop-amd64(虛擬作業系統)

此次作業沒有不能使用 compiled image of any packaged distribution 的限制，因此我們選定 ubuntu 做為我們的作業系統

實作步驟：

1. 在 VMware 上執行 Linux 系統

- (1) 下載 Ubuntu ISO 檔 下載網址：[Download Ubuntu Desktop | Ubuntu](#)
- (2) 開啟後點選 Create a New Virtual Machine -> typical -> I will install operating system later -> Linux -> Other Linux 2.6.x kernel
- (3) Disk size 我設定 60GB -> finish
- (4) 點選  Edit virtual machine settings -> 在 CD/DVD 選擇 use ISO image file -> 選取 ubuntu-24.04.1-desktop-amd64.iso
- (5) 執行 虛擬環境 如果沒有 error 就可以看到 Linux 系統



2. 在 VMware 上面進行 Linux kernel 的製作 – Part1 環境準備

- (1) sudo apt-get update 更新包 apt-get

```
linzixin@linzixin-VMware-Virtual-Platform:~$ sudo apt update
已有:1 http://tw.archive.ubuntu.com/ubuntu noble InRelease
```

(2) sudo apt-get install build-essential 安裝 GCC 編譯器

```
linzixin@linzixin-VMware-Virtual-Platform:~$ sudo apt-get install build-essential
```

(3) sudo apt-get install libncurses-dev bison flex libssl-dev libelf-dev

```
linzixin@linzixin-VMware-Virtual-Platform:~$ sudo apt-get install libncurses-dev bison flex libssl-dev libelf-dev
```

(4) git clone git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux-2.6.git

(5) cd linux-2.6

```
linzixin@linzixin-VMware-Virtual-Platform:~$ cd linux-2.6/
linzixin@linzixin-VMware-Virtual-Platform:~/linux-2.6$ ls
arch          init          modules.builtin      System.map
block         io_uring      modules.builtin.modinfo test_syscall
built-in.a    ipc          modules.order        test_syscall.c
certs         Kbuild       Module.symvers       tools
```

3. 在 VMware 上面進行 Linux kernel 的製作 – Part2 修改 kernel

進行 kernel 的修改我們需要修改三個部分

1. 修改 syscall_64.tbl -> 做為 系統呼叫的索引號、名稱及對應的內核函數 (給新增函數 id)
2. 修改 sys.c -> 實現系統呼叫功能 (實現函數)
3. 修改 syscalls.h -> 聲明系統呼叫的原型。(告訴 kernel 新增函數)

實做步驟 修改 syscall_64.tbl

(1) cd arch/x86/entry/syscalls/

(2) nano syscall_64.tbl

(3) 增加 548 64 print_school_id sys_print_school_id

548 是系統呼叫的號碼

64 是系統的位元數，表示 64 位

print_school_id 給系統呼叫分配的名稱

sys_print_school_id 是內核中實際實現系統呼叫的函數名

GNU nano 7.2			syscall_64.tbl *
543	x32	io_setup	compat_sys_io_setup
544	x32	io_submit	compat_sys_io_submit
545	x32	execveat	compat_sys_execveat
546	x32	preadv2	compat_sys_preadv64v2
547	x32	pwritev2	compat_sys_pwritev64v2
548	64	print_school_id	sys_print_school_id

實做步驟 修改 sys.c

(1) cd /kernel

(2) nano sys.c

(3) 找個位置寫入函數

```
GNU nano 7.2 sys.c

SYSCALL_DEFINE0(print_school_id){
    printk(KERN_EMERG "School ID : M11315051\n");
    return 0;
}
```

實做步驟 修改 syscalls.h

- (1) cd include/linux/
- (2) nano syscalls.h
- (3) 寫入 asmlinkage long sys_print_school_id(void);

```
GNU nano 7.2 syscalls.h

/*
 * Architecture-specific system calls
 */

/* x86 */
asmlinkage long sys_ioperm(unsigned long from, unsigned long num, int on);

asmlinkage long sys_uretprobe(void);

asmlinkage long sys_print_school_id(void); // new syscall
```

4. 在 VMware 上面進行 Linux kernel 的製作 – Part3 make time

- (1) make menuconfig

```
General setup --->
[*] 64-bit kernel (NEW)
Processor type and features --->
[*] Mitigations for CPU vulnerabilities (NEW) --->
Power management and ACPI options --->
Bus options (PCI etc.) --->
Binary Emulations --->
[*] Virtualization (NEW) --->
```

會看到像是這樣的畫面

- (2) Save & exit
- (3) nano .config (後續在 實作過程問題 會說明)
- (4) control + w 進入搜尋模式
- (5) 輸入 CONFIG_SYSTEM_TRUST_KEYS，會看到下圖，並且在附近也會看到 CONFIG_SYSTEM_REVOCATION_KEYS

```
CONFIG_SYSTEM_TRUSTED_KEYS="debian/canonical-certs.pem"
```

```
CONFIG_SYSTEM_REVOCATION_KEYS="debian/canonical-revoked-certs.pem"
```

- (6) 將 CONFIG_SYSTEM_TRUST_KEYS 以及 CONFIG_SYSTEM_REVOCATION_KEYS 改成 ""

```
CONFIG_SYSTEM_TRUSTED_KEYS=""
```

```
CONFIG_SYSTEM_REVOCATION_KEYS=""
```

- (7) 保存後退出
 - (8) make j\$(nproc)
 - (9) sudo make modules_install
 - (10) sudo make install
 - (11) 完成後重啟系統
5. 在 VMware 上面檢查是否已經新增 system call

- (1) 撰寫 C 程式來調用新增的系統呼叫
- (2) nano test_syscall.c

```
linzixin@linzixin-VMware-Virtual-Platform:~/linux-2.6$ nano test_syscall.c
```

- (3) 輸入程式

```
GNU nano 7.2 test_syscall.c
#include <stdio.h>
#include <unistd.h>

int main(){
    syscall(548);
    return 0;
}
```

- (4) gcc test_syscall.c -o test_syscall (編譯這個程式)
- (5) ./test_syscall (運行測試程式)

```
linzixin@linzixin-VMware-Virtual-Platform:~/linux-2.6$ gcc test_syscall.c -o test_syscall
linzixin@linzixin-VMware-Virtual-Platform:~/linux-2.6$ ./test_syscall
```

- (6) sudo dmesg | tail (檢查核心日誌中的訊息) 看到學號了，可喜可賀！

```
linzixin@linzixin-VMware-Virtual-Platform:~/linux-2.6$ sudo dmesg | tail
[sudo] linzixin 的密碼：
[ 340.950647] audit: type=1400 audit(1727411680.926:181): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="snap.firefox.hook.post-refresh" pid=2239 comm="apparmor_parser"
[ 341.110536] audit: type=1400 audit(1727411681.085:182): apparmor="DENIED" operation="open" class="file" profile="snap-update-ns.firefox" name="/usr/local/share/" pid=2252 comm="6" requested_mask="r" denied_mask="r" fsuid=0 ouid=0
[ 419.920900] systemd-journald[373]: /var/log/journal/dcdf8916b8c04cbc99c5dc087173ffab/user-1000.journal: Journal file uses a different sequence number ID, rotating.
[ 420.139780] audit: type=1400 audit(1727411760.112:183): apparmor="DENIED" operation="open" class="file" profile="snap-update-ns.snapd-desktop-integration" name="/proc/2384/maps" pid=2384 comm="5" requested_mask="r" denied_mask="r" fsuid=1000 ouid=0
[ 420.302078] rfkill: input handler enabled
[ 422.377514] rfkill: input handler disabled
[ 422.983917] audit: type=1400 audit(1727411762.956:184): apparmor="DENIED" operation="open" class="file" profile="snap-update-ns.snapd-desktop-integration" name="/proc/3035/maps" pid=3035 comm="5" requested_mask="r" denied_mask="r" fsuid=1000 ouid=0
[ 425.186334] input: VMware DnD UInput pointer as /devices/virtual/input/input6
[ 497.008925] School ID : M11315051
[ 497.009171] audit: type=1400 audit(1727411836.980:185): apparmor="DENIED" operation="open" class="file" profile="syslogd" name="/run/systemd/sessions/" pid=1202 comm="72733A6061696E20513A526567" requested_mask="r" denied_mask="r" fsuid=102 ouid=0
```

實作過程問題 (由處理花費的時間排序)：

1. CONFIG_SYSTEM_TRUST_KEYS 以及 CONFIG_SYSTEM_REVOCATION_KEYS 認證問題

在實作過程中，我遇到了以下 error，這是編譯過程中，與 TRUST_KEYS 以及 REVOCATION_KEYS 證書有關的 error，所以我去.config 檔案裡面進行修改，然後修改成""之後就可以正常 make 了

```
linzixin@linzixin-VMware-Virtual-Platform:~/linux-2.6$ make -j$(nproc) CONFIG_MODULE_
SIG=n CONFIG_SYSTEM_TRUSTED_KEYS=""
mkdir -p /home/linzixin/linux-2.6/tools/objtool && make O=/home/linzixin/linux-2.6 su
bdir=tools/objtool --no-print-directory -C objtool
INSTALL libsubcmd_headers
CALL scripts/checksyscalls.sh
HOSTCC certs/extract-cert
CC mm/kfence/core.o
COPY certs/x509.genkey
CC certs/blacklist.o
make[3]: *** 沒有規則可製作目標「debian/canonical-revoked-certs.pem」，由「certs/x509
_revocation_list」需求。 停止。
make[3]: *** 正在等待未完成的作業....
GEN certs/blacklist_hash_list
make[2]: *** [scripts/Makefile.build:485: certs] 錯誤 2
```

2. 從 HW1 系統轉到 Ubuntu

其實一開始是打算使用 HW1 的系統進行 HW2 的作業，但是遇到主要的問題是 gcc 的編譯器無法安裝在簡易系統，雖然可以在 WSL(windows support linux)上先編譯完再丟到簡易系統，但是我後來選擇直接使用 compiled image of Ubuntu，用以解決 gcc 的編譯器無法安裝在簡易系統的問題。

3. VMware 容量問題，我一開始是給 20GB，結果再編譯的時候容量不足，後來改 40GB 還是一樣，最後是改 60GB 才足夠，推薦一開始就先設 60GB

HW comment

這次 HW 比起上次 HW 遇到的問題，處理的時間短很多，不知道是因為更了解 linux OS 還是這次遇到的問題比較簡單，不過經過了多次的練習，現在對於如何編譯 system call 已經有一定程度的了解，我覺得這是很好的練習。