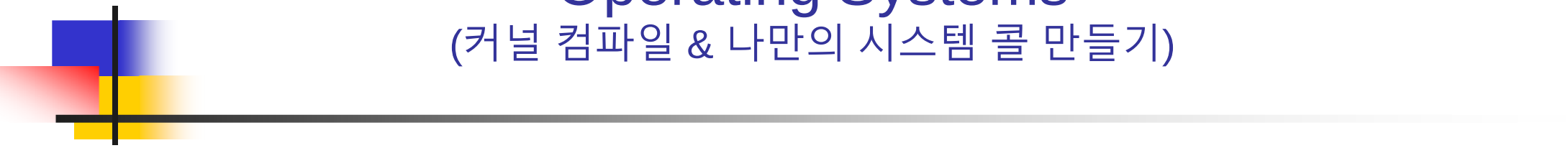
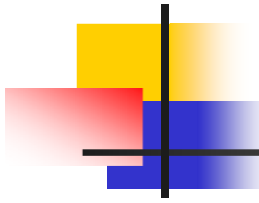


Operating Systems
(커널 컴파일 & 나만의 시스템 콜 만들기)

Day 5

These lecture materials are modified from the lecture notes written by A. Silberschatz, P. Galvin and G. Gagne.

August, 2022



주의할 점

- 가상머신 세팅 고칠 것 (가상머신 꺼진 상태서만 가능함)
  - 용량 20GB로는 너무 부족하니 더 늘리기
    - (전 200GB로 했는데 100GB면 충분할 것으로 보임)
  - RAM도 8GB까지 늘리기
  - core 개수도 늘리기
    - (본인 컴퓨터 사양보고 늘리기)

- 이때 용량은 파티션 설정을 다시 해 줘야하는데 아래를 참조
  - [Ubuntu] VMware Ubuntu 디스크 공간 확장 (tistory.com)
    - sudo apt-get install gparted
    - sudo gparted

소스 다운로드 & 압축 풀기

- 리눅스 커널 소스 코드

```
wget https://mirrors.edge.kernel.org/pub/linux/kernel/v5.x/linux-5.18.16.tar.xz
```

```
tar xvf linux-5.18.16.tar.xz
```

<=압축 풀기

cd linux-5.18.16

소스에 시스템 콜 추가

- cd kernel

```
/kernel$ nano mycall.c
```

```c
#include<linux/kernel.h>
#include<linux/linkage.h>
#include<linux/syscalls.h>
int value;
SYSCALL_DEFINE1(mycall1, int, x){
        printk("INSERT value: %d", x);
        value = x;
        return 0;
}
SYSCALL_DEFINE0(mycall2){
        printk("Current value: %d", value);
        return value;
}
```

- printk 는 커널 메시지를 출력해 주는 함수
- 커널 메시지는 커맨드 demsg로 확인할 수 있음

추가한 시스템 콜을 컴파일 하기 위한 환경 세팅

- nano Makefile

```
obj-y     = fork.o exec_domain.o panic.o \
            cpu.o exit.o softirq.o resource.o \
            sysctl.o capability.o ptrace.o user.o \
            signal.o sys.o umh.o workqueue.o pid.o task_work.o \
            extable.o params.o \
            kthread.o sys_ni.o nsproxy.o \
            notifier.o ksysfs.o cred.o reboot.o \
            async.o range.o smpboot.o ucount.o regset.o mycall.o
```

추가!

추가한 시스템 콜을 컴파일 하기 위한 환경 세팅

- cd ..
- cd ./arch/x86/entry/syscalls

```
/arch/x86/entry/syscalls$ nano syscall_64.tbl
```

추가!

```
451     common  mycall1                 sys_mycall1
452     common  mycall2                 sys_mycall2
```

탭키임
띄어쓰기 아님

추가한 시스템 콜을 컴파일 하기 위한 환경 세팅

- cd ../../../..
- cd ./include/linux

```
/include/linux$ nano syscalls.h
```

```
#include <asm/syscall_wrapper.h>

asmlinkage long sys_mycall1(int value);
asmlinkage long sys_mycall2(void);
#endif /* CONFIG_ARCH_HAS_SYSCALL_WRAPPER */
```

추가!

소스 코드 옮기고 계속 컴파일 환경 설정

sudo apt-get install git fakeroot build-essential ncurses-dev xz-utils libssl-dev flex libelf-dev bison

cd ~ (tar 파일 받은 경로로 가면됨)
sudo cp -r linux-5.18.16 /usr/src/ linux-5.18.16

```
cd /usr/src/linux-5.18.16
```

```
sudo cp /boot/config-$(uname -r) .config
```

```
sudo make menuconfig
```

***만약 아래와 같이 나오면 디스플레이 크기 변경하고,
해상도도 100%로 줄일 것.

```
Your display is too small to run Menuconfig!
It must be at least 19 lines by 80 columns.
```

***제대로 창이 뜨면, load->save->exit

소스 코드 옮기고 계속 컴파일 환경 설정

```
sudo nano .config
```

(1)

CONFIG_SYSTEM_TRUSTED_KEYS="debian/canonical-certs.pem“

CONFIG_SYSTEM_TRUSTED_KEYS=""

(2)

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

CONFIG_SYSTEM_REVOCATION_KEYS=""

컴파일

```
sudo make -j8
```

여기서 8은 사용할 코어 개수
만약 –j [코어개수]를 안 하고, sudo make만 하면
진짜로 컴파일 10시간 걸림

```
  LD      arch/x86/boot/setup.elf
  OBJCOPY arch/x86/boot/setup.bin
  BUILD   arch/x86/boot/bzImage
Kernel: arch/x86/boot/bzImage is ready  (#6)
```

중간에 뜸!
끝 아님!

이렇게 뜨면 컴파일 잘 된거지만 이게 안 뜨면 **어디에 오타가 있으니 재 확인 할 것**
시간은 시간대로 몇시간 소비하고 컴파일 실패할 수도 있음

컴파일

```
$ sudo make modules_install -j8
  SIGN    /lib/modules/5.18.16/kernel/sound/usb/s
  SIGN    /lib/modules/5.18.16/kernel/sound/xen/s
  DEPMOD  /lib/modules/5.18.16
os@os-virtual-machine:/usr/src/linux-5.18.16$
```

컴파일한 커널을 OS에 인스톨

```
os@os-virtual-machine:/usr/src/linux-5.18.16$ uname -r
5.15.0-43-generic
```

그 전의 커널 버전

sudo make install

```
os@os-virtual-machine:/usr/src/linux-5.18.16$ sudo update-initramfs -c -k 5.18.16
update-initramfs: Generating /boot/initrd.img-5.18.16
```

새로운 커널 인스톨

```
os@os-virtual-machine:/usr/src/linux-5.18.16$ sudo update-grub
Sourcing file `/etc/default/grub'
```

컴파일한 커널을 OS에 인스톨

sudo reboot

```
os@os-virtual-machine:~$ uname -r
5.18.16
```

추가한 시스템 콜 사용해보기

추가한 시스템콜이 잘 돌아가는지 확인하기 위한 코딩 (유저 레벨)

nano systest.c

```c
#include<stdio.h>
#include<linux/kernel.h>
#include<sys/syscall.h>
#include<unistd.h>
int main(){
        int result1 = syscall(451, 123456789);
        printf("mycall1 returned %d\n", result1);
        int result2 = syscall(452);
        printf("mycall2 returned %d\n", result2);

        return 0;
}
```

gcc –o systest.out systest.c

추가한 시스템 콜 사용해보기

```
os@os-virtual-machine:~$ ./systest.out
mycall1 returned 0
mycall2 returned 123456789
```

```
~$ sudo dmesg
```
커널 메시지 확인

```
[  610.443565] INSERT value: 123456789
[  610.443642] Current value: 123456789
```

의도한 대로 시스템 콜이 잘 작동함!

Github에 업로드 해보자!

일단 repository부터 만들 것

```
git init
```

```
git config --global user.name "dayoung08"
```

```
git config --global user.email "cecci08@naver.com"
```

코드를 git init한 디렉토리에 옮기거나 cp 할 것.

```
mv ../linux-5.18.16 linux-5.18.16
```

```
git add -A
```

새 파일이 추가되거나, 파일이 수정될 경우 add를 해줘야 함

Github에 업로드 해보자!

여기 “first commit” 대신 변경/추가 사항 간단히 작성할 것

```
git commit -m "first commit"
```

```
git branch -M main
git remote add origin http://github.com/dayoung08/kernel.git
git push -u origin main
```

```
Username for 'https://github.com': dayoung08
Password for 'https://dayoung08@github.com':
warning: https://github.com/dayoung08/kernel.git/(으)로 리다이렉트
```

여기서 패스워드는 토큰을 입력해주면 됨
[GitHub] 깃허브 토큰(Token) 생성하는 법 :: 대두코기 (tistory.com)
토큰 복붙할 때 공백 안 들어가도록 주의할 것

Day 5
END