

운영체제실습

assignment 1

학 과: 컴퓨터정보공학부

담당교수: 김태석 교수님

학 번: 2019202050

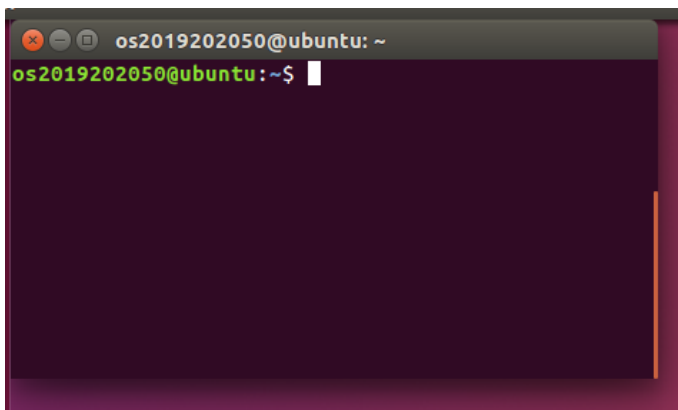
성 명: 이강현

1. Introduction

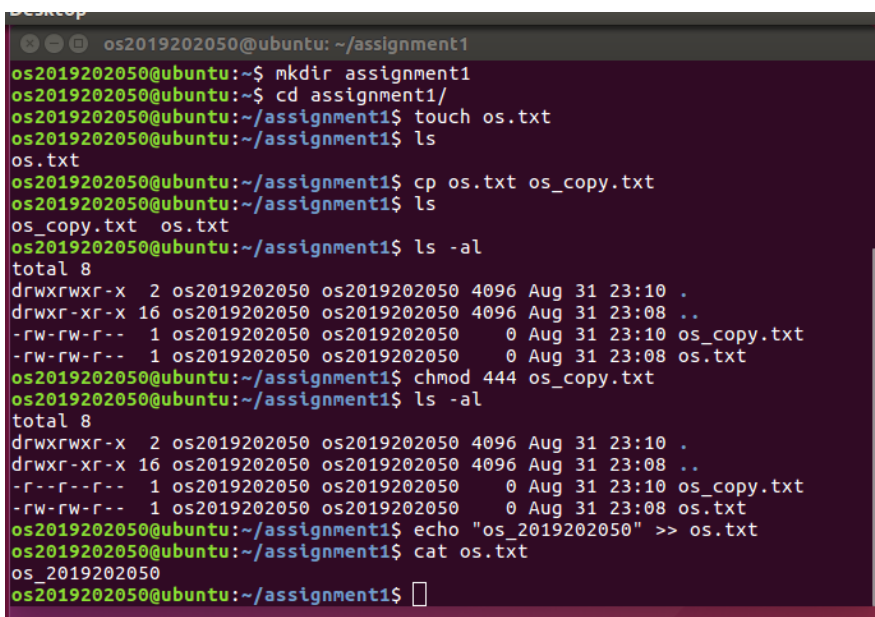
리눅스를 설치하고 기본적인 터미널 명령어에 대해 학습한다. 리눅스 커널을 터미널을 이용해 다운로드하고 이를 컴파일하는 법을 학습한다. 커널을 컴파일하는 과정에 대해 알아본다. ctag 나 cscope 와 같은 도구를 이용하여 변수나 함수를 코드내에서 찾는 법을 배우고 이를 이용한 커널 코드 수정을 진행해본다.

2. Result

<assignment 1-1>



```
os2019202050@ubuntu: ~  
os2019202050@ubuntu:~$
```



```
os2019202050@ubuntu: ~/assignment1  
os2019202050@ubuntu:~$ mkdir assignment1  
os2019202050@ubuntu:~$ cd assignment1/  
os2019202050@ubuntu:~/assignment1$ touch os.txt  
os2019202050@ubuntu:~/assignment1$ ls  
os.txt  
os2019202050@ubuntu:~/assignment1$ cp os.txt os_copy.txt  
os2019202050@ubuntu:~/assignment1$ ls  
os_copy.txt  os.txt  
os2019202050@ubuntu:~/assignment1$ ls -al  
total 8  
drwxrwxr-x  2 os2019202050 os2019202050 4096 Aug 31 23:10 .  
drwxr-xr-x 16 os2019202050 os2019202050 4096 Aug 31 23:08 ..  
-rw-rw-r--  1 os2019202050 os2019202050   0 Aug 31 23:10 os_copy.txt  
-rw-rw-r--  1 os2019202050 os2019202050   0 Aug 31 23:08 os.txt  
os2019202050@ubuntu:~/assignment1$ chmod 444 os_copy.txt  
os2019202050@ubuntu:~/assignment1$ ls -al  
total 8  
drwxrwxr-x  2 os2019202050 os2019202050 4096 Aug 31 23:10 .  
drwxr-xr-x 16 os2019202050 os2019202050 4096 Aug 31 23:08 ..  
-r--r--r--  1 os2019202050 os2019202050   0 Aug 31 23:10 os_copy.txt  
-rw-rw-r--  1 os2019202050 os2019202050   0 Aug 31 23:08 os.txt  
os2019202050@ubuntu:~/assignment1$ echo "os_2019202050" >> os.txt  
os2019202050@ubuntu:~/assignment1$ cat os.txt  
os_2019202050  
os2019202050@ubuntu:~/assignment1$
```

1. Assignment1 명의 디렉토리를 생성하기 위해 mkdir 명령어를 사용하여 디렉토리를 생성하였다.
2. 생성한 디렉토리로 이동하기 위해 cd 명령어를 사용하였고 그후 os.txt 라는 빈 파일을 생성하기 위해 touch 명령어를 사용했다.
3. cp 명령어를 이용하여 os.txt 를 복사한 os_copy.txt 를 생성하였다.
4. chmod 명령어를 이용하여 os_copy.txt 파일의 접근권한을 모든 대상에게 읽기만 가능하게끔 수정하였다.
5. echo 명령어와 >>를 이용하여 os_2019202050 이라는 문자열을 os.txt 파일로 리다이렉션하여 파일에 쓰고 cat 명령어를 통해 터미널에 os.txt 파일의 내용을 출력하였다.

<assignment 1-2>

```
os2019202050@ubuntu:~$ sudo wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.19.67.tar.xz
--2023-09-17 04:34:26-- https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.19.67.tar.xz
Resolving cdn.kernel.org (cdn.kernel.org)... 146.75.49.176, 2a04:4e42:7c::432
Connecting to cdn.kernel.org (cdn.kernel.org)|146.75.49.176|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 103291756 (99M) [application/x-xz]
Saving to: 'linux-4.19.67.tar.xz'

linux-4.19.67.tar.xz      100%[=====] 98.51M  61.9MB/s   in 1.6s
2023-09-17 04:34:27 (61.9 MB/s) - 'linux-4.19.67.tar.xz' saved [103291756/103291756]

os2019202050@ubuntu:~$ ls
Desktop  Downloads  examples.desktop  linux-4.19.67.tar.xz  Music  Pictures  Public  Templates  Videos
os2019202050@ubuntu:~$ tar -Jxvf linux-4.19.67.tar.xz
```

위와 같이 커널을 다운받고 tar 압축을 해제한다.

```
os2019202050@ubuntu: ~/Downloads/linux-4.19.67
os2019202050@ubuntu:~$ ls
Desktop  Downloads  Music  Public  Videos
Documents  examples.desktop  Pictures  Templates
os2019202050@ubuntu:~$ cd Downloads/
os2019202050@ubuntu:~/Downloads$ ls
linux-4.19.67  linux-4.19.67.tar.xz
os2019202050@ubuntu:~/Downloads$ cd linux-4.19.67/
os2019202050@ubuntu:~/Downloads/linux-4.19.67$ ls
arch      CREDITS      firmware  ipc         lib         mm          scripts    usr
block     crypto       fs        Kbuild     LICENSES    net         security   virt
certs     Documentation  include   Kconfig    MAINTAINERS  README     sound
COPYING   drivers      init      kernel     Makefile     samples    tools
os2019202050@ubuntu:~/Downloads/linux-4.19.67$ vi Makefile
os2019202050@ubuntu:~/Downloads/linux-4.19.67$
```

커널의 extra version 을 수정하기 위해 다운받은 커널에서 makefile 를 vi 로 수정한다.

```
os2019202050@ubuntu: ~/Downloads/linux-4.19.67
# SPDX-License-Identifier: GPL-2.0
VERSION = 4
PATCHLEVEL = 19
SUBLEVEL = 67
EXTRAVERSION = -2019202050
NAME = "People's Front"
```

위와 같이 extraversion 을 학번으로 수정한다.

```
os2019202050@ubuntu:~/Downloads/linux-4.19.67$ sudo apt install build-essential
libncurses5-dev bison flex libssl-dev libelf-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
build-essential is already the newest version (12.1ubuntu2).
The following additional packages will be installed:
  libbison-dev libfl-dev libsigsegv2 libssl-doc libtinfo-dev m4 zlib1g-dev
Suggested packages:
  bison-doc ncurses-doc
The following NEW packages will be installed:
  bison flex libbison-dev libelf-dev libfl-dev libncurses5-dev libsigsegv2
  libssl-dev libssl-doc libtinfo-dev m4 zlib1g-dev
0 upgraded, 12 newly installed, 0 to remove and 0 not upgraded.
Need to get 4,006 kB of archives.
After this operation, 15.5 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

커널을 컴파일할 도구들을 위와같이 설치하고

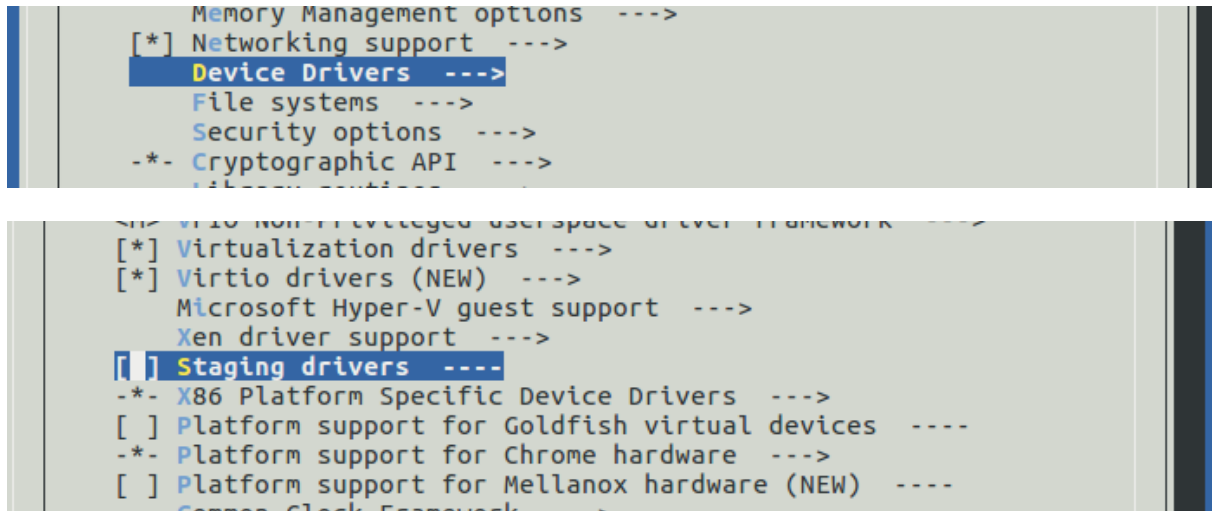
Sudo make menuconfig 를 터미널에 입력하여 아래와 같이 커널 환경설정을 진행한다.

```

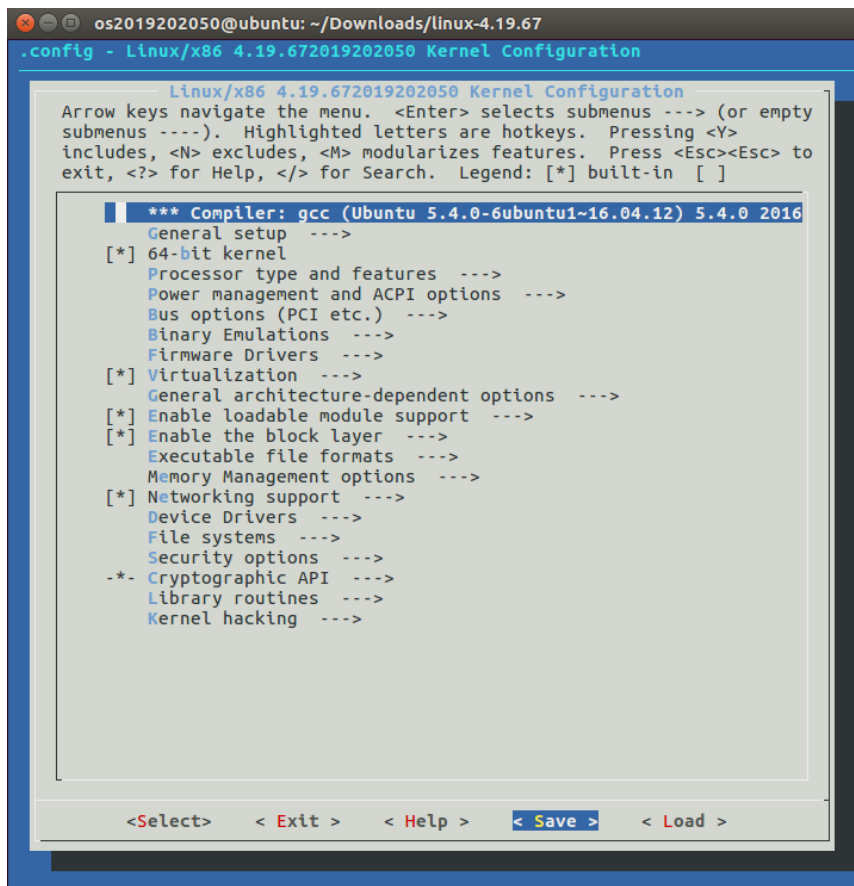
[*] Virtualization --->
General architecture-dependent options --->
[*] Enable loadable module support --->
[*] Enable the block layer --->
Executable file formats --->
Memory Management options --->
[*] Networking support --->
```

```
exit, <?> for Help, </> for Search. Legend: [*] built-in [ ]
--- Enable loadable module support
[*] Forced module loading
[*] Module unloading
[ ] Forced module unloading
[ ] Module versioning support
[*] Source checksum for all modules
[*] Module signature verification
[ ] Require modules to be validly signed
[*] Automatically sign all modules
Which hash algorithm should modules be signed with? (Sign m
[ ] Compress modules on installation
```

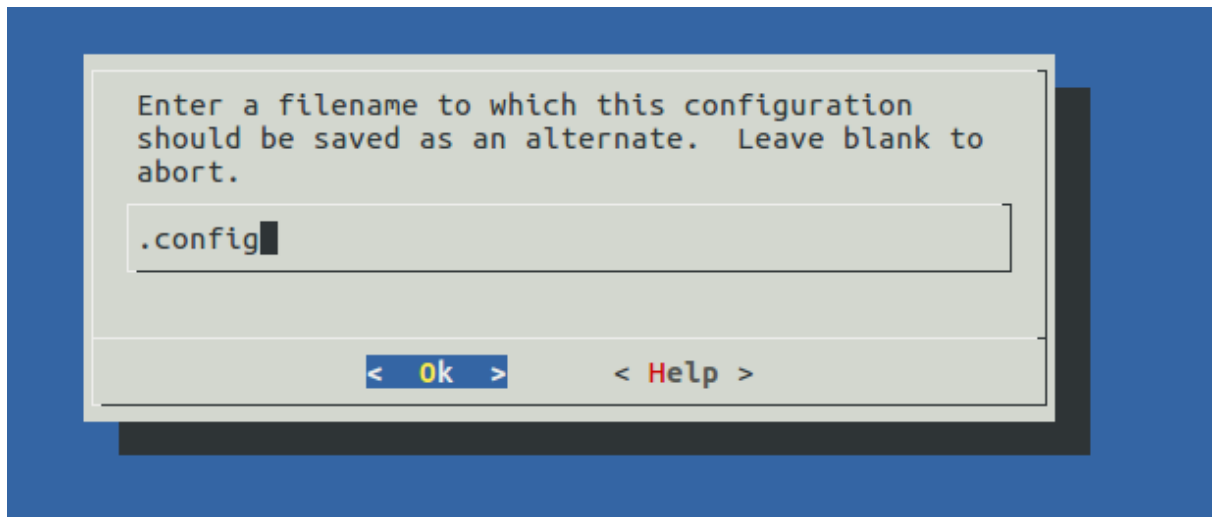
커널 모듈 적재시 발생하는 문제를 해결하는 과정이다.



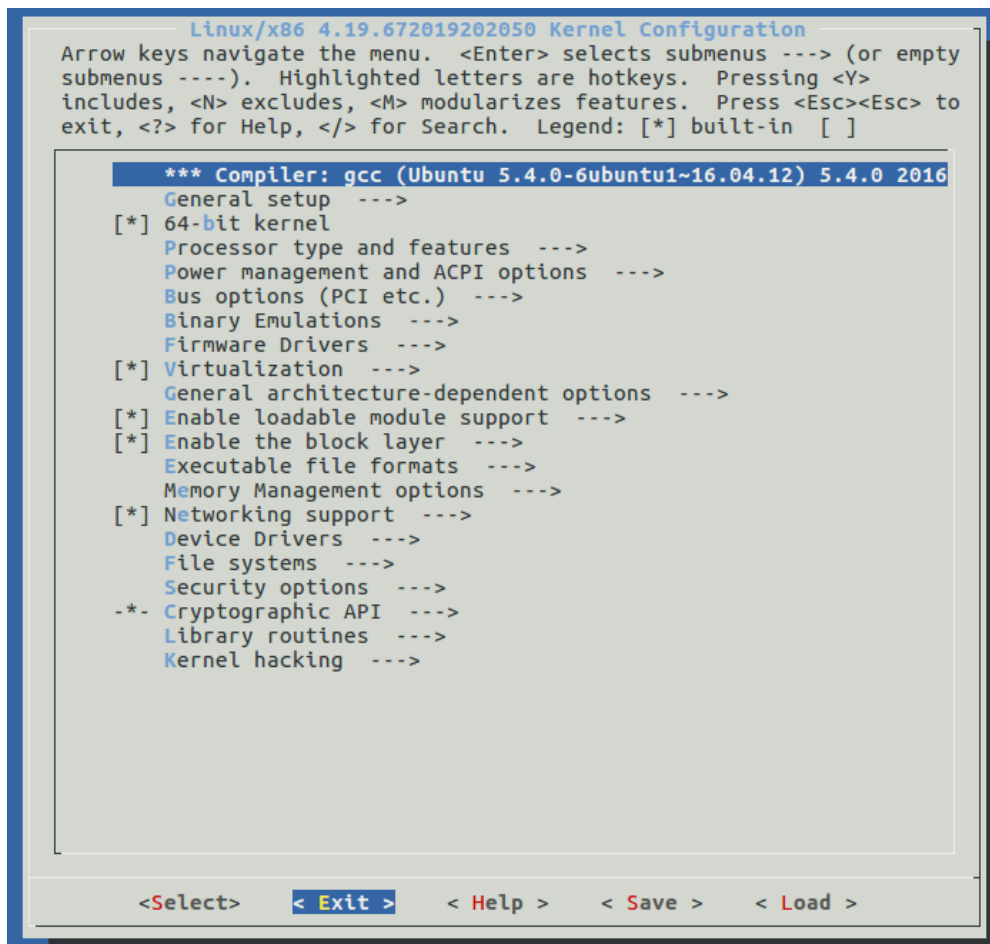
커널 컴파일시 문제가 될 수 있는 모듈을 제거한다.



Save 를 누르고



.config 로 설정을 저장하고



Exit 를 눌러 빠져나오면

```
os2019202050@ubuntu: ~/Downloads/linux-4.19.67
HOSTLD scripts/kconfig/mconf
scripts/kconfig/mconf Kconfig
#
# using defaults found in /boot/config-4.15.0-29-generic
#
/boot/config-4.15.0-29-generic:890:warning: symbol value 'm' invalid for HOTPLUG
PCI_SHPC
/boot/config-4.15.0-29-generic:1144:warning: symbol value 'm' invalid for NF_NAT
REDIRECT
/boot/config-4.15.0-29-generic:1147:warning: symbol value 'm' invalid for NF_TAB
LES_INET
/boot/config-4.15.0-29-generic:1148:warning: symbol value 'm' invalid for NF_TAB
LES_NETDEV
/boot/config-4.15.0-29-generic:1331:warning: symbol value 'm' invalid for NF_TAB
LES_IPV4
/boot/config-4.15.0-29-generic:1336:warning: symbol value 'm' invalid for NF_TAB
LES_ARP
/boot/config-4.15.0-29-generic:1343:warning: symbol value 'm' invalid for NF_NAT
MASQUERADE_IPV4
/boot/config-4.15.0-29-generic:1378:warning: symbol value 'm' invalid for NF_TAB
LES_IPV6
/boot/config-4.15.0-29-generic:1388:warning: symbol value 'm' invalid for NF_NAT
MASQUERADE_IPV6
/boot/config-4.15.0-29-generic:1416:warning: symbol value 'm' invalid for NF_TAB
LES_BRIDGE
/boot/config-4.15.0-29-generic:3992:warning: symbol value 'm' invalid for HW_RAN
DOM_TPM
/boot/config-4.15.0-29-generic:4941:warning: symbol value 'm' invalid for LIRC
/boot/config-4.15.0-29-generic:6167:warning: symbol value 'm' invalid for SND_SO
C_INTEL_SST_TOPLEVEL
/boot/config-4.15.0-29-generic:6172:warning: symbol value 'm' invalid for SND_SO
C_INTEL_MACH
/boot/config-4.15.0-29-generic:7725:warning: symbol value 'm' invalid for DELL_S
MBIOS_WMI
/boot/config-4.15.0-29-generic:7726:warning: symbol value 'm' invalid for DELL_S
MBIOS_SMM

*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.

os2019202050@ubuntu:~/Downloads/linux-4.19.67$
```

이렇게 환경설정이 마무리된 것을 확인할 수 있다.

```
os2019202050@ubuntu:~/Downloads/linux-4.19.67$ make -j16
```

바뀐 환경설정을 기반으로 커널을 컴파일 한다. Vmware 로 사용할 프로세서의 수를 8로 설정해주었기에 이것의 2 배정도로 컴파일을 수행할 thread 수를 결정하였다.

```
os2019202050@ubuntu:~/Downloads/linux-4.19.67$ sudo make modules_install
```

컴파일된 모듈을 /lib/module 로 이동해주고

```

os2019202050@ubuntu:~/Downloads/linux-4.19.67$ sudo make install
sh ./arch/x86/boot/install.sh 4.19.67-2019202050 arch/x86/boot/bzImage \
    System.map "/boot"
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 4.19.67-2019202050
/boot/vmlinuz-4.19.67-2019202050
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 4.19.67-2019202050 /
boot/vmlinuz-4.19.67-2019202050
update-initramfs: Generating /boot/initrd.img-4.19.67-2019202050
run-parts: executing /etc/kernel/postinst.d/pm-utils 4.19.67-2019202050 /boot/vm
linuz-4.19.67-2019202050
run-parts: executing /etc/kernel/postinst.d/unattended-upgrades 4.19.67-20192020
50 /boot/vmlinuz-4.19.67-2019202050
run-parts: executing /etc/kernel/postinst.d/update-notifier 4.19.67-2019202050 /
boot/vmlinuz-4.19.67-2019202050
run-parts: executing /etc/kernel/postinst.d/zz-update-grub 4.19.67-2019202050 /b
oot/vmlinuz-4.19.67-2019202050
Generating grub configuration file ...
Warning: Setting GRUB_TIMEOUT to a non-zero value when GRUB_HIDDEN_TIMEOUT is se
t is no longer supported.
Found linux image: /boot/vmlinuz-4.19.67-2019202050
Found initrd image: /boot/initrd.img-4.19.67-2019202050
Found linux image: /boot/vmlinuz-4.15.0-29-generic
Found initrd image: /boot/initrd.img-4.15.0-29-generic
Found memtest86+ image: /boot/memtest86+.elf
Found memtest86+ image: /boot/memtest86+.bin
done

```

컴파일된 커널을 부트로더에 등록해준다.

```

os2019202050@ubuntu:~/Downloads/linux-4.19.67$ sudo vi /etc/default/grub

```

grub 설정을 열어 GRUB_HIDDEN_TIMEOUT=0 을 주석처리하여 부팅시 grub 을 확인할 수 있게끔 설정하고 해당 기능을 사용하지 않으니 GRUB_HIDDEN_TIMEOUT_QUIET 설정 또한 false 로 설정한다.

```

GRUB_DEFAULT=0
#GRUB_HIDDEN_TIMEOUT=0
GRUB_HIDDEN_TIMEOUT_QUIET=false
GRUB_TIMEOUT=10
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX_DEFAULT="quiet"
GRUB_CMDLINE_LINUX="find_preseed=/preseed.cfg auto noprompt priority=critical lo
cale=en_US"

```

```

os2019202050@ubuntu:~/Downloads/linux-4.19.67$ reboot

```

설정을 마무리하고 재부팅을 진행한다.


```
os2019202050@ubuntu: ~  
os2019202050@ubuntu:~$ uname -r  
4.19.67-2019202050  
os2019202050@ubuntu:~$
```

재부팅 후 `uname -r`을 이용하여 커널의 버전을 확인하면 다운받은 커널의 버전이 사용되었고 extra 버전 또한 학번으로 잘 변경된 것을 확인한다.

<assignment 1-3>

커널내 Linux agp...이 실행되는 지점을 찾기 위해서 `cscope`를 사용하였다.

```
os2019202050@ubuntu:~/Downloads/linux-4.19.67$ sudo apt install cscope
```

위와 같이 `cscope`를 설치해주고 `cscope -R` 명령어를 이용하여 데이터베이스를 형성해주면 자동으로 아래와 같이 파일내에서 함수나 변수를 찾을 수 있게 된다.

```
os2019202050@ubuntu: ~/Downloads/linux-4.19.67  
Text string: agpgart  


| File             | Line                                                     |
|------------------|----------------------------------------------------------|
| 0 agp.h          | 34 #define PFX "agpgart: "                               |
| 1 ali-agp.c      | 399 .name = "agpgart-ali",                               |
| 2 amd-k7-agp.c   | 543 .name = "agpgart-amdk7",                             |
| 3 amd64-agp.c    | 741 .name = "agpgart-amd64",                             |
| 4 ati-agp.c      | 559 .name = "agpgart-ati",                               |
| 5 backend.c      | 38 #include <linux/agpgart.h>                            |
| 6 backend.c      | 338 printk(KERN_INFO "Linux agpgart interface v%d.%d\n", |
| 7 compat_ioctl.c | 32 #include <linux/agpgart.h>                            |

  
* Lines 23-31 of 58, 28 more - press the space bar to display more *  
Find this C symbol:  
Find this global definition:  
Find functions called by this function:  
Find functions calling this function:  
Find this text string:  
Change this text string:  
Find this egrep pattern:  
Find this file:  
Find files #including this file:  
Find assignments to this symbol:
```

관련된 부분이 출력문이므로 find this text string 을 이용해 agpgart 를 찾도록 하였다.

그 결과 커널 출력부분인 printk 부분을 다운받은 커널 폴더내의 drivers/char/agp/backend.c 라는 파일에서 찾을 수 있었고 출력문을 수정하기 위해 해당 파일로 이동하였다. 추가적으로 출력해야 하는 부분은 Linux agp...을 출력시키는 함수의 함수명과 argument 이므로 functions called by this function 을 검색할 수 있게 해주는 cscope 의 기능을 이용하여 해당부분을 agp_init 이 호출하는 것이 맞는지 다시한번 확인하였다.

```
os2019202050@ubuntu: ~/Downloads/linux-4.19.67
Functions called by this function: agp_init

File      Function Line
0 backend.c printk 338 printk(KERN_INFO "Linux agpgart interface v%d.%d\n",

static int __init agp_init(void)
{
    if (!agp_off)
        printk(KERN_INFO "os2019202050_Linux agpgart interface v%d.%d\n",
                    AGPGART_VERSION_MAJOR, AGPGART_VERSION_MINOR);
        printk(KERN_INFO "os2019202050_arg in agp_init(void)");
    return 0;
}
```

따라서 위처럼 출력문을 추가해주었다. 로그레벨은 KERN_INFO 로 설정하였다.

```
os2019202050@ubuntu:~/Downloads/linux-4.19.67$ dmesg | grep "os2019202050" -n
1:[    0.000000] Linux version 4.19.67-2019202050 (os2019202050@ubuntu) (gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.10)) #2 SMP Sun Sep 17 03:27:43 PDT 2023
1407:[    6.293847] os2019202050_Linux agpgart interface v0.103
1408:[    6.293847] os2019202050_arg in agp_init(void)
os2019202050@ubuntu:~/Downloads/linux-4.19.67$
```

메시지가 연달아서 중복없이 하나의 set 으로 출력되는 것을 확인하였다.

3. 고찰

리눅스의 기본적인 명령어부터 리눅스 커널의 코드를 수정하고 이를 컴파일하여 적용하는 방법까지 학습해보았다. 간단한 내용들이었지만 라이브러리나 단순 api 를 사용하는 것이 아니라 kernel level 의 코드를 다루는 부분이라 수정에 있어서 신중할 필요가 있었다. 또한 많이 복잡하고 많은 함수가 있어 원하는 부분을 찾는게 쉽지 않았고 그 과정에서 사용되는 ctags 나 cscope 을 잘 이용하는게 중요할 듯 하였다.

4. Reference

운영체제실습 강의자료 참조