

운영체제 실습 Report

실습 제목: Assignment 1

실습일자: 2022년 09월 01일 (목)

제출일자: 2022년 09월 22일 (목)

학 과: 컴퓨터정보공학부

담당교수: 최상호 교수님

실습분반: 금요일 5,6교시

학 번: 2018202065

성 명: 박 철 준

● Introduction

1. 제목

Assignment 1

2. 목적

가. Assignment 1-1

과제 요구 사항

리눅스를 설치하고 계정 ID "os학번"으로 나오게 한다.

터미널 화면 캡처 후 보고서에 첨부하여야 한다.

각 해당하는 리눅스 명령어 사용하여 캡처 후 보고서에 첨부

1. "assignment1"명의 디렉터리 생성
2. assignmnet1 디렉터리 이동 후 "os.txt"명의 빈파일 생성
3. os.txt를 os_copy.txt명으로 복사
4. os_copy.txt의 권한을 모든 대상에게 읽기만 부여
5. os.txt에 os_본인학번 작성 후 터미널에 출력

나. Assignment 1-2

과제 요구 사항

커널 컴파일 과정을 terminal과 vi를 사용하여 캡처하여야 한다.

각 명령어가 어떠한 기능을 하였는 지 간단히 서술하여야 한다.

4.19.67 커널로 재 부팅 후 버전 확인한다.

"uname -r" 의 결과가 "4.19.67-본인학번" 이 나오도록 진행하여야 한다.

다. Assignment 1-3

과제 요구사항

Linux agp...이 실행되는 지점에서 Linux Kernel Message 출력하여야 한다.

Linux agp...이 실행되는 지점에 커널 메시지가 다음과 같이 출력되도록 커널 코드 수정하여야 한다.

Linux agp... 을 실행시키는 함수의 함수명과 argument의 값을 출력하도록 커널 코드 수정한다.

Linux Kernel Message 양식은 아래와 같다.

os본인학번_Linux agpgart interface v0.103

os본인학번_arg in function_name (argument)

printk() 함수 사용하여야 한다. 로그 레벨은 KERN_INFO 사용한다. (강의 자료 "Appendix A" 참고) 출력 메시지는 7 page의 [Linux Kernel Message 양식] 참고한다.

dmesg , grep으로 확인 시 (1), (2) 메시지가 연달아 출력되어야 한다.(강의 자료 "Appendix B" 참고)

주의사항 1, 2를 참고하여 보고서에 다음의 내용을 필히 포함

1. 리눅스 커널 코드에서 수정한 부분을 명시한다.
2. 소스코드 path 작성한다.
3. 검색한 캡처 화면 첨부
- 4.결과 화면 캡처

● Conclusion & Analysis

- Assignment 1-1

```
os2018202065@ubuntu: ~  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
os2018202065@ubuntu:~$
```

계정 ID가 os2018202065로 os학번으로 설정 후 설치 완료하였다.

```
os2018202065@ubuntu: ~/assignment1  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
os2018202065@ubuntu:~$ mkdir assignment1  
os2018202065@ubuntu:~$ cd assignment1  
os2018202065@ubuntu:~/assignment1$ touch os.txt  
os2018202065@ubuntu:~/assignment1$ ls  
os.txt  
os2018202065@ubuntu:~/assignment1$ cp os.txt os_copy.txt  
os2018202065@ubuntu:~/assignment1$ ls  
os_copy.txt  os.txt  
os2018202065@ubuntu:~/assignment1$ ls -al  
total 8  
drwxrwxr-x  2 os2018202065 os2018202065 4096 Sep 14 08:08 .  
drwxr-xr-x 18 os2018202065 os2018202065 4096 Sep 14 08:06 ..  
-rw-rw-r--  1 os2018202065 os2018202065    0 Sep 14 08:08 os_copy.txt  
-rw-rw-r--  1 os2018202065 os2018202065    0 Sep 14 08:07 os.txt  
os2018202065@ubuntu:~/assignment1$ chmod 444 os_copy.txt  
os2018202065@ubuntu:~/assignment1$ ls -al  
total 8  
drwxrwxr-x  2 os2018202065 os2018202065 4096 Sep 14 08:08 .  
drwxr-xr-x 18 os2018202065 os2018202065 4096 Sep 14 08:06 ..  
-r--r--r--  1 os2018202065 os2018202065    0 Sep 14 08:08 os_copy.txt  
-rw-rw-r--  1 os2018202065 os2018202065    0 Sep 14 08:07 os.txt  
os2018202065@ubuntu:~/assignment1$ echo os_2018202065 > os.txt  
os2018202065@ubuntu:~/assignment1$ cat os.txt  
os_2018202065  
os2018202065@ubuntu:~/assignment1$
```

1. mkdir assignment1으로 디렉터리 생성
2. cd assignment1으로 디렉터리 접근
3. touch os.txt로 빈파일 생성
4. cp os.txt os_copy.txt로 복사
5. chmod 444 os_copy.txt로 모든 대상에 읽기 권한만 부여
6. echo os_2018202065 > os.txt를 통 os_2018202065를 리다이렉션한 os.txt에 저장한다.

- Assignment 1-2

```
root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67
SPDX-License-Identifier: GPL-2.0
VERSION = 4
PATCHLEVEL = 19
SUBLEVEL = 67
EXTRAVERSION = -2018202065
NAME = "People's Front"

# *DOCUMENTATION*
# To see a list of typical targets execute "make help"
# More info can be located in ./README
# Comments in this file are targeted only to the developer, do not
# expect to learn how to build the kernel reading this file.

# That's our default target when none is given on the command line
PHONY := _all
_all:

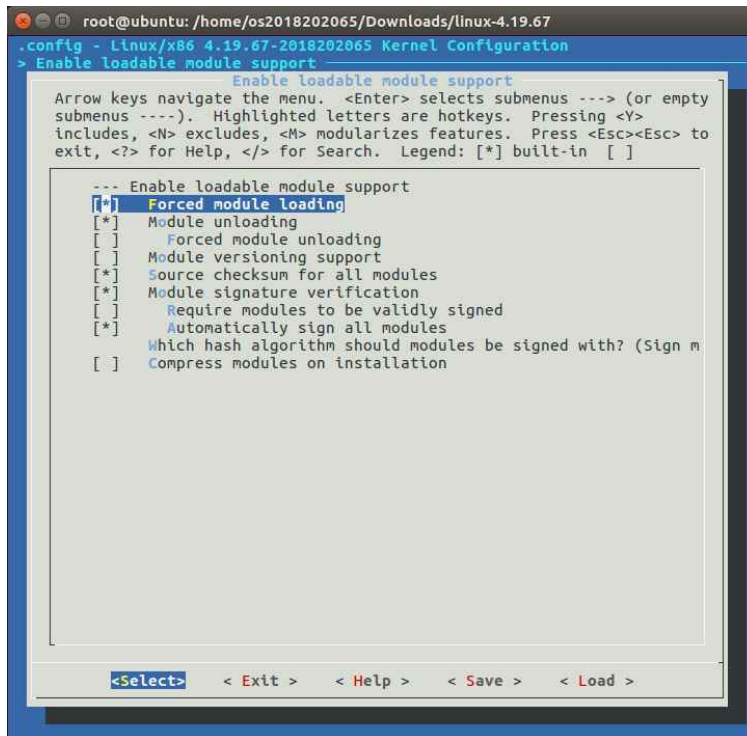
# o Do not use make's built-in rules and variables
#   (this increases performance and avoids hard-to-debug behaviour);
# o Look for make include files relative to root of kernel src
MAKEFLAGS += -rR --include-dir=$(CURDIR)

# Avoid funny character set dependencies
"Makefile" 1736L, 60021C
```

커널 소스파일을 다운로드받고 vi makefile을 통해 -학번으로 수정

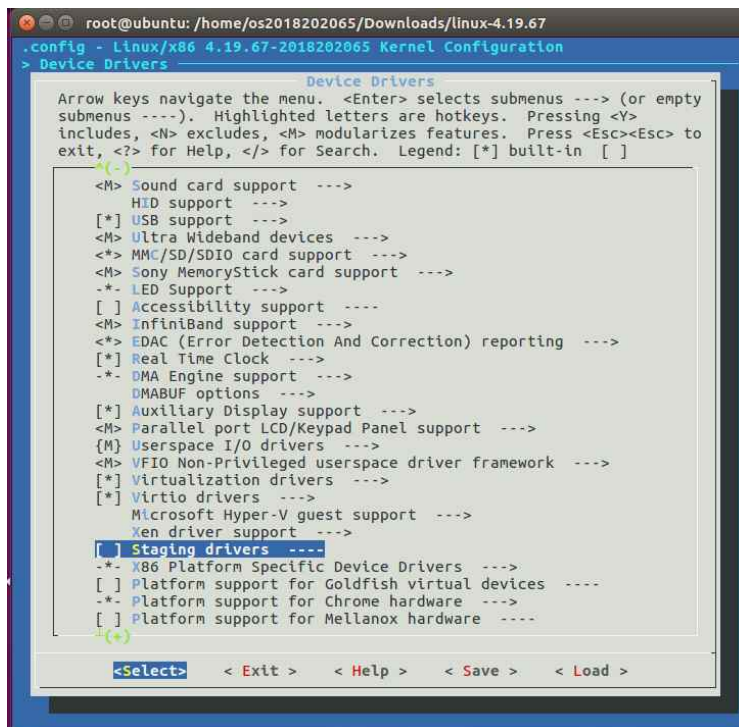
```
root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67
os2018202065@ubuntu:~$ sudo su
[sudo] password for os2018202065:
root@ubuntu:/home/os2018202065# cd Downloads/
root@ubuntu:/home/os2018202065/Downloads# cd linux-4.19.67
root@ubuntu:/home/os2018202065/Downloads/linux-4.19.67# sudo apt install build-essential libncurses5-dev bison flex libssl-dev libelf-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
bison is already the newest version (2:3.0.4.dfsg-1).
build-essential is already the newest version (12.1ubuntu2).
flex is already the newest version (2.6.0-11).
libncurses5-dev is already the newest version (6.0+20160213-1ubuntu1).
libelf-dev is already the newest version (0.165-3ubuntu1.2).
libssl-dev is already the newest version (1.0.2g-1ubuntu4.20).
0 upgraded, 0 newly installed, 0 to remove and 494 not upgraded.
root@ubuntu:/home/os2018202065/Downloads/linux-4.19.67#
```

관리자 권한으로 build-essential libncurses5-dev bison flex libssl-dev libelf-dev을 설치



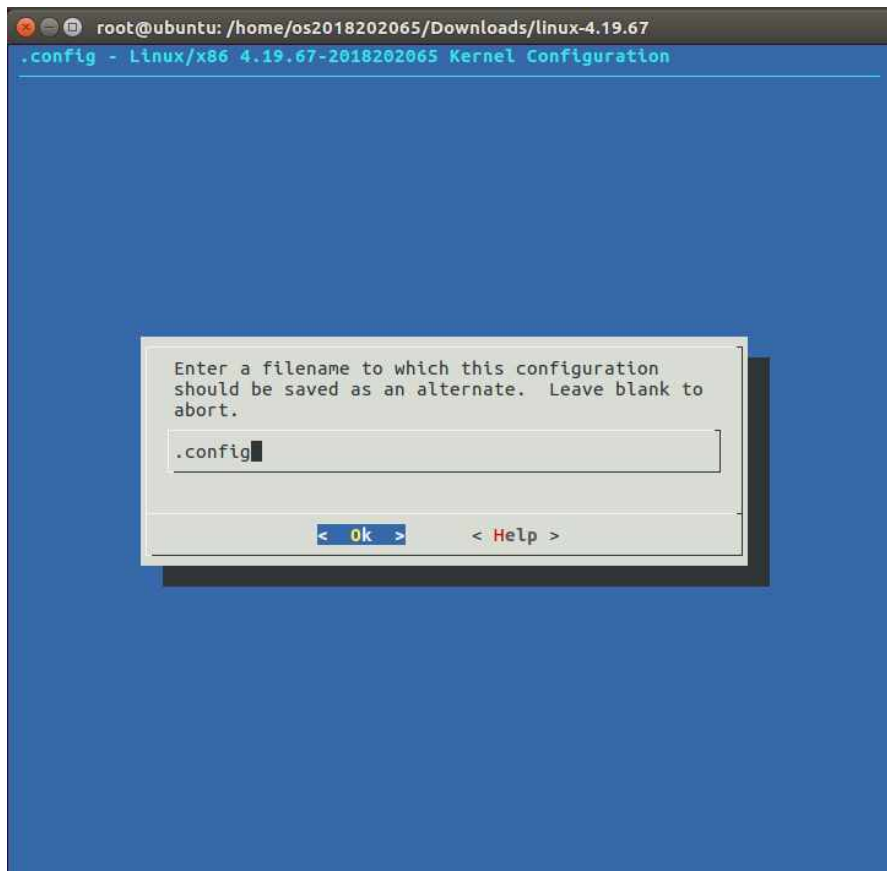
sudo make menuconfig을 통해 커널 환경 설정 준비하고 커널 모듈 적재 시 발생할 수 있는 문제 해결을 위해

"Enable loadable module support" → "Forced module loading" 체크한다.



컴파일 시 문제가 될 수 있는 모듈 제거를 위해

"Device Drivers" → "Staging drivers" 체크 해제한다.



환경 설정을 save한다.

```
scripts/kconfig/mconf Kconfig

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

root@ubuntu:/home/os2018202065/Downloads/linux-4.19.67#
```

이후 다시 터미널로 돌아온 후 저장을 확인한다.

```
root@ubuntu:/home/os2018202065/Downloads/linux-4.19.67# make -j6
DESCEND objtool
CALL scripts/checksyscalls.sh
CHK include/generated/compile.h
CC [M] net/nfc/hci/core.o
AR net/netfilter/built-in.a
LD [M] net/netfilter/nf_conntrack.o
CC [M] net/nfc/hci/hcp.o
LD [M] net/netfilter/nf_conntrack_h323.o
LD [M] net/netfilter/nf_nat.o
LD [M] net/netfilter/nf_tables.o
CC [M] drivers/infiniband/hw/cxgb3/cxio_resource.o
CC [M] net/netfilter/xt_connlabel.o
CC [M] net/nfc/hci/command.o
CC [M] net/nfc/hci/llc.o
CC [M] net/nfc/hci/llc_nop.o
```

make -j6를 하여 커널을 컴파일 한다.

```
root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67
LD [M] sound/soc/intel/boards/snd-soc-sst-bytcr-rt5651.ko
LD [M] sound/soc/intel/boards/snd-soc-sst-bytcr-rt5640.ko
LD [M] sound/soc/intel/boards/snd-soc-sst-cht-bsw-max98090_ti.ko
LD [M] sound/soc/intel/boards/snd-soc-sst-cht-bsw-rt5645.ko
LD [M] sound/soc/intel/boards/snd-soc-sst-cht-bsw-rt5672.ko
LD [M] sound/soc/intel/boards/snd-soc-sst-haswell.ko
LD [M] sound/soc/intel/common/snd-soc-acpi-intel-match.ko
LD [M] sound/soc/intel/common/snd-soc-sst-acpi.ko
LD [M] sound/soc/intel/common/snd-soc-sst-dsp.ko
LD [M] sound/soc/intel/common/snd-soc-sst-firmware.ko
LD [M] sound/soc/intel/common/snd-soc-sst-ipc.ko
LD [M] sound/soc/intel/haswell/snd-soc-sst-haswell-pcm.ko
LD [M] sound/soc/intel/skylake/snd-soc-skl.ko
LD [M] sound/soc/intel/skylake/snd-soc-skl-ssp-clk.ko
LD [M] sound/soc/intel/skylake/snd-soc-skl-ipc.ko
LD [M] sound/soc/snd-soc-acpi.ko
LD [M] sound/soc/snd-soc-core.ko
LD [M] sound/soc/xtensa/snd-soc-xtfpga-i2s.ko
LD [M] sound/soc/zte/zx-tdm.ko
LD [M] sound/synth/emux/snd-emux-synth.ko
LD [M] sound/soundcore.ko
LD [M] sound/synth/snd-util-mem.ko
LD [M] sound/usb/6fire/snd-usb-6fire.ko
LD [M] sound/usb/bcd2000/snd-bcd2000.ko
LD [M] sound/usb/caiaq/snd-usb-caiaq.ko
LD [M] sound/usb/hiface/snd-usb-hiface.ko
LD [M] sound/usb/line6/snd-usb-line6.ko
LD [M] sound/usb/line6/snd-usb-pod.ko
LD [M] sound/usb/line6/snd-usb-podhd.ko
LD [M] sound/usb/line6/snd-usb-toneport.ko
LD [M] sound/usb/line6/snd-usb-variak.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/x86/snd-hdmi-lpe-audio.ko
LD [M] virt/lib/iqbybypass.ko
root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67#
root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67#
```

커널 컴파일이 완료된 모습이다.

```
root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67#
root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67# make modules_install
INSTALL arch/x86/crypto/aes-x86_64.ko
INSTALL arch/x86/crypto/aesni-intel.ko
INSTALL arch/x86/crypto/blowfish-x86_64.ko
INSTALL arch/x86/crypto/camellia-aesni-avx-x86_64.ko
INSTALL arch/x86/crypto/camellia-aesni-avx2.ko
INSTALL arch/x86/crypto/camellia-x86_64.ko
INSTALL arch/x86/crypto/cast5-avx-x86_64.ko
INSTALL arch/x86/crypto/cast6-avx-x86_64.ko
INSTALL arch/x86/crypto/chacha20-x86_64.ko
INSTALL arch/x86/crypto/crc32-pclmul.ko
INSTALL arch/x86/crypto/crct10dif-pclmul.ko
INSTALL arch/x86/crypto/des3_edc-x86_64.ko
INSTALL arch/x86/crypto/ghash-clmulni-intel.ko
INSTALL arch/x86/crypto/glue_helper.ko
INSTALL arch/x86/crypto/poly1305-x86_64.ko
```

make modules_install을 통해 System.map을 /boot으로 복사


```
root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67
INSTALL sound/soc/intel/boards/snd-soc-sst-bytcr-rt5640.ko
INSTALL sound/soc/intel/boards/snd-soc-sst-bytcr-rt5651.ko
INSTALL sound/soc/intel/boards/snd-soc-sst-cht-bsw-max98090_ti.ko
INSTALL sound/soc/intel/boards/snd-soc-sst-cht-bsw-rt5645.ko
INSTALL sound/soc/intel/boards/snd-soc-sst-cht-bsw-rt5672.ko
INSTALL sound/soc/intel/boards/snd-soc-sst-haswell.ko
INSTALL sound/soc/intel/common/snd-soc-acpi-intel-match.ko
INSTALL sound/soc/intel/common/snd-soc-sst-acpi.ko
INSTALL sound/soc/intel/common/snd-soc-sst-dsp.ko
INSTALL sound/soc/intel/common/snd-soc-sst-firmware.ko
INSTALL sound/soc/intel/common/snd-soc-sst-ipc.ko
INSTALL sound/soc/intel/haswell/snd-soc-sst-haswell-pcm.ko
INSTALL sound/soc/intel/skylake/snd-soc-skl-ipc.ko
INSTALL sound/soc/intel/skylake/snd-soc-skl-ssp-clk.ko
INSTALL sound/soc/intel/skylake/snd-soc-skl.ko
INSTALL sound/soc/snd-soc-acpi.ko
INSTALL sound/soc/snd-soc-core.ko
INSTALL sound/soc/xtensa/snd-soc-xtfpga-i2s.ko
INSTALL sound/soc/zte/zx-tdm.ko
INSTALL sound/soundcore.ko
INSTALL sound/synth/emux/snd-emux-synth.ko
INSTALL sound/synth/snd-util-mem.ko
INSTALL sound/usb/6fire/snd-usb-6fire.ko
INSTALL sound/usb/bcd2000/snd-bcd2000.ko
INSTALL sound/usb/caiaq/snd-usb-caiaq.ko
INSTALL sound/usb/hiface/snd-usb-hiface.ko
INSTALL sound/usb/line6/snd-usb-line6.ko
INSTALL sound/usb/line6/snd-usb-pod.ko
INSTALL sound/usb/line6/snd-usb-podhd.ko
INSTALL sound/usb/line6/snd-usb-toneport.ko
INSTALL sound/usb/line6/snd-usb-variiax.ko
INSTALL sound/usb/misc/snd-ua101.ko
INSTALL sound/usb/snd-usb-audio.ko
INSTALL sound/usb/snd-usbmidi-lib.ko
INSTALL sound/usb/usx2y/snd-usb-us122l.ko
INSTALL sound/usb/usx2y/snd-usb-usx2y.ko
INSTALL sound/x86/snd-hdmi-lpe-audio.ko
INSTALL virt/lib/irqbypass.ko
DEPMOD 4.19.67-2018202065
root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67#
```

make modules_install이 완료된 모습이다.

```
root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67# make install
sh ./arch/x86/boot/install.sh 4.19.67-2018202065 arch/x86/boot/bzImage \
    System.map "/boot"
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 4.19.67-2018202065
/boot/vmlinuz-4.19.67-2018202065
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 4.19.67-2018202065 /
boot/vmlinuz-4.19.67-2018202065
update-initramfs: Generating /boot/initrd.img-4.19.67-2018202065
```

make install을 통해 Grub 부트 로더에 자동 등록


```

root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67
sh ./arch/x86/boot/install.sh 4.19.67-2018202065 arch/x86/boot/bzImage \
    System.map "/boot"
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 4.19.67-2018202065
/boot/vmlinuz-4.19.67-2018202065
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 4.19.67-2018202065 /
boot/vmlinuz-4.19.67-2018202065
update-initramfs: Generating /boot/initrd.img-4.19.67-2018202065
run-parts: executing /etc/kernel/postinst.d/pm-utils 4.19.67-2018202065 /boot/vm
linuz-4.19.67-2018202065
run-parts: executing /etc/kernel/postinst.d/unattended-upgrades 4.19.67-20182020
65 /boot/vmlinuz-4.19.67-2018202065
run-parts: executing /etc/kernel/postinst.d/update-notifier 4.19.67-2018202065 /
boot/vmlinuz-4.19.67-2018202065
run-parts: executing /etc/kernel/postinst.d/zz-update-grub 4.19.67-2018202065 /b
oot/vmlinuz-4.19.67-2018202065
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-4.19.67-2018202065
Found initrd image: /boot/initrd.img-4.19.67-2018202065
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
root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67#

```

make install이 완료된 모습이다.

```

root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67
# If you change this file, run 'update-grub' afterwards to update
# /boot/grub/grub.cfg.
# For full documentation of the options in this file, see:
#   info -f grub -n 'Simple configuration'

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"

# Uncomment to enable BadRAM filtering, modify to suit your needs
# This works with Linux (no patch required) and with any kernel that obtains
# the memory map information from GRUB (GNU Mach, kernel of FreeBSD ...)
#GRUB_BADRAM="0x01234567,0xfefefefe,0x89abcdef,0xefefefef"

# Uncomment to disable graphical terminal (grub-pc only)
#GRUB_TERMINAL=console

# The resolution used on graphical terminal
# note that you can use only modes which your graphic card supports via VBE
# you can see them in real GRUB with the command 'vbeinfo'
#GRUB_GFXMODE=640x480

# Uncomment if you don't want GRUB to pass "root=UUID=xxx" parameter to Linux
#GRUB_DISABLE_LINUX_UUID=true

# Uncomment to disable generation of recovery mode menu entries
#GRUB_DISABLE_RECOVERY="true"

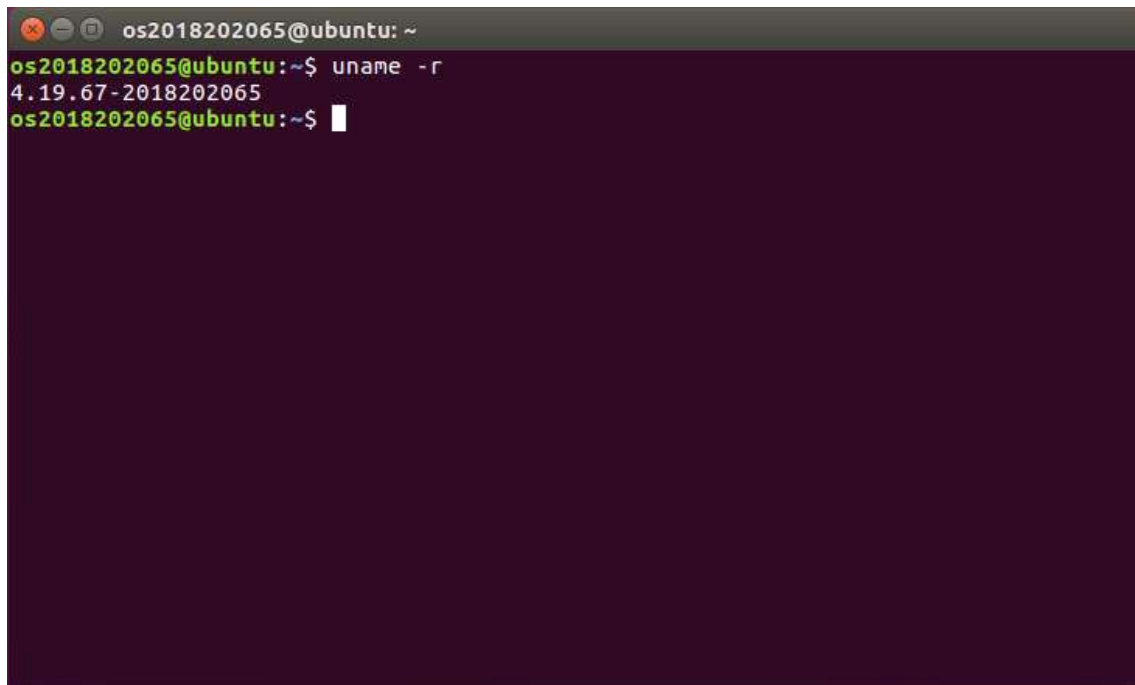
# Uncomment to get a beep at grub start
#GRUB_INIT_TUNE="480 440 1"

~
~
~
/etc/default/grub" 34L, 1302C 1,1 All

```

vi 편집기를 사용하여 리눅스를 재부팅 하였을 시 컴파일한 커널을 적용시켜 실행하기 위한 설정을 한다.

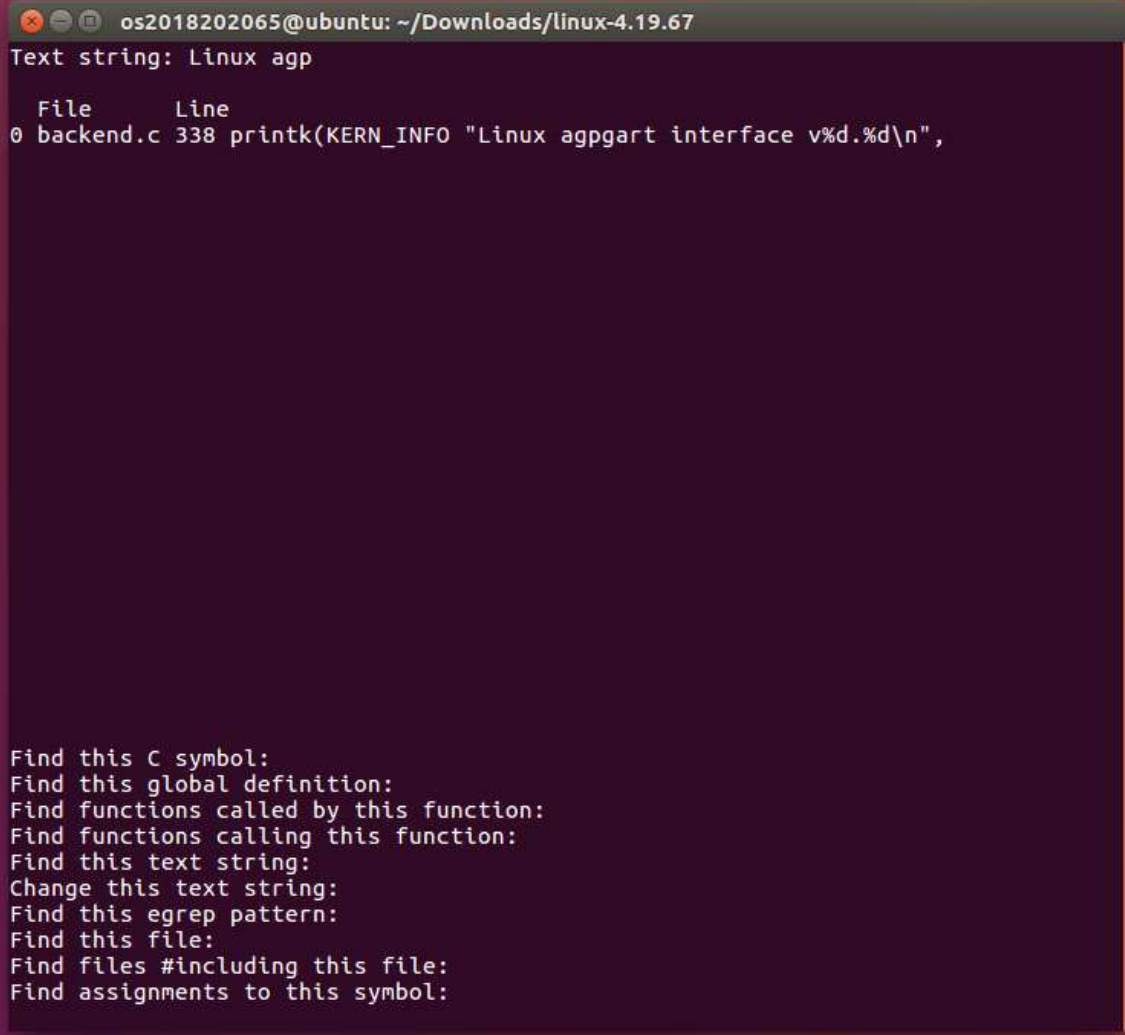
이때 사용한 명령어는 `sudo vi /etc/default/grub`이다.

A terminal window with a dark purple background. The title bar at the top shows window control icons and the text 'os2018202065@ubuntu: ~'. The terminal content shows the command 'uname -r' being executed, with the output '4.19.67-2018202065' displayed on the next line. The prompt 'os2018202065@ubuntu:~\$' is visible at the bottom of the terminal area.

```
os2018202065@ubuntu: ~  
os2018202065@ubuntu:~$ uname -r  
4.19.67-2018202065  
os2018202065@ubuntu:~$
```

이후 reboot을 통해 재부팅하고 나서 `uname -r`을 통해 현재 사용중인 커널을 확인 한 결과 4.19.67-2018202065로 요구하는 바와 결과가 일치하는 것을 알 수 있다.

- Assignment 1-3



```
os2018202065@ubuntu: ~/Downloads/linux-4.19.67
Text string: Linux agp

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

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:
```

cscope -R을 실행하고 find this test string:에 Linux agp까지만 입력하여도 backend.c 파일에서 과제 요구사항에서 요구하는 구문이 검색되게 되는데 sudo su를 권한을 가지고 있다면 엔터를 눌러 해당 C파일에 접근한다.


```
root@ubuntu: /home/os2018202065/Downloads/linux-4.19.67
}
EXPORT_SYMBOL_GPL(agp_remove_bridge);

int agp_off;
int agp_try_unsupported_boot;
EXPORT_SYMBOL(agp_off);
EXPORT_SYMBOL(agp_try_unsupported_boot);

static int __init agp_init(void)
{
    if (!agp_off)
        printk(KERN_INFO "os2018202065_Linux agpgart interface v%d.%d\n",
                AGPGART_VERSION_MAJOR, AGPGART_VERSION_MINOR);
        printk(KERN_INFO "os2018202065_arg in agp_init(void)\n");
    return 0;
}

static void __exit agp_exit(void)
{
}

#ifdef MODULE
static __init int agp_setup(char *s)

```

해당 구문을 다음과 같이 과제에서 요구하는 형식으로 수정을 한다.

```
os2018202065@ubuntu: ~
os2018202065@ubuntu:~$ dmesg | grep "os2018202065" -n
1404:[ 10.570331] os2018202065_Linux agpgart interface v0.103
1405:[ 10.570333] os2018202065_arg in agp_init(void)
os2018202065@ubuntu:~$
```

이후 module compile 후 reboot을 진행하고

dmesg | grep "os2018202065" -n을 입력하게 되면 과제 요구에 맞게 결과가 출력됨을 알 수 있다.

```
os2018202065@ubuntu:~$ sudo find / -name backend.c
[sudo] password for os2018202065:
/home/os2018202065/Downloads/linux-4.19.67/drivers/char/agp/backend.c
find: '/run/user/1000/gvfs': Permission denied
os2018202065@ubuntu:~$
```

또한 해당 소스코드는 backend.c파일이며 해당 파일을 find 명령어를 통해 찾게 되면 (이때 사용한 명령어는 sudo find / -name backend.c이다.)

/home/os2018202065/Downloads/linux-4.19.67/drivers/char/agp/backend.c의 file path를 가지는 것을 알 수 있다.

- 고찰

이번 과제를 진행하면서 GUI환경에 익숙한 나에게 CLI환경인 리눅스에서 콘솔모드로 효율적인 작업 수행을 하기위해선 과제를 통해 리눅스 명령어 습득이 가장 중요함을 알게되었다. 또한 서버나 임베디드 소프트웨어 등 리눅스를 사용하는 CLI환경에서 텍스트 기반 파일의 작업을 할 때 윈도우에서 흔하게 쓰는 텍스트 편집기를 사용할 수 없기에 vi 편집기를 이용해서 작성을 해야한다. 이를 위해서는 vi편집기의 사용을 원활하게 할 수 있어야 하고 이번 과제를 통해 vi 편집기의 활용을 익힐 수 있었다. 또한 리눅스 커널을 컴파일 하는 법과 커널 내부 파일을 수정하는 법을 익혀 운영체제 실습 조건을 갖추 수 있었다.