

운영체제

담당 교수님: 김태석 교수님

학번: 2016722074

이름: 김영태

실습번호: Assignment1

Assignment 1-1

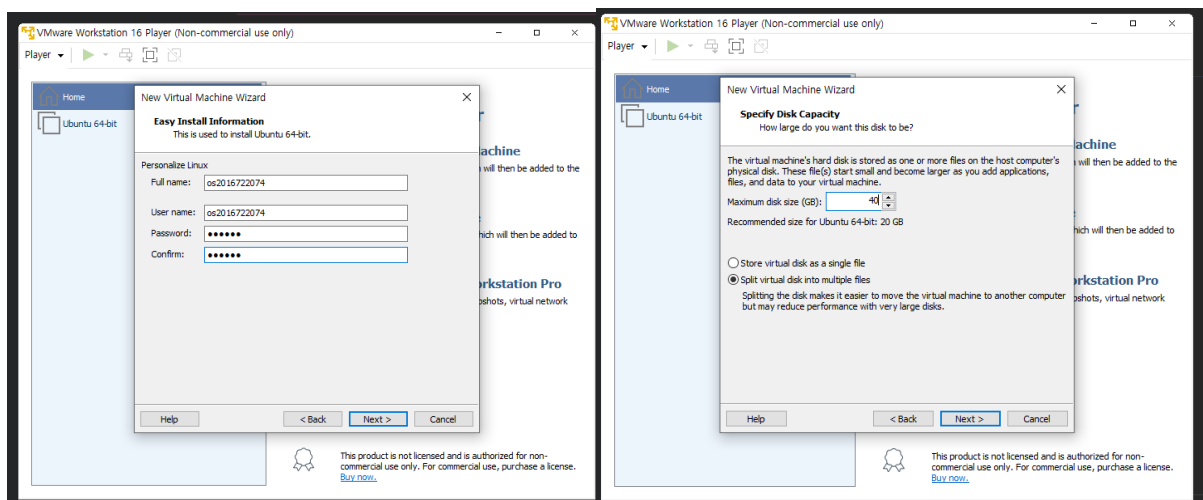
Introduction

가상 환경에서 OS를 설치할 수 있는 VMware를 설치하고 그 위에 Ubuntu를 설치한다.

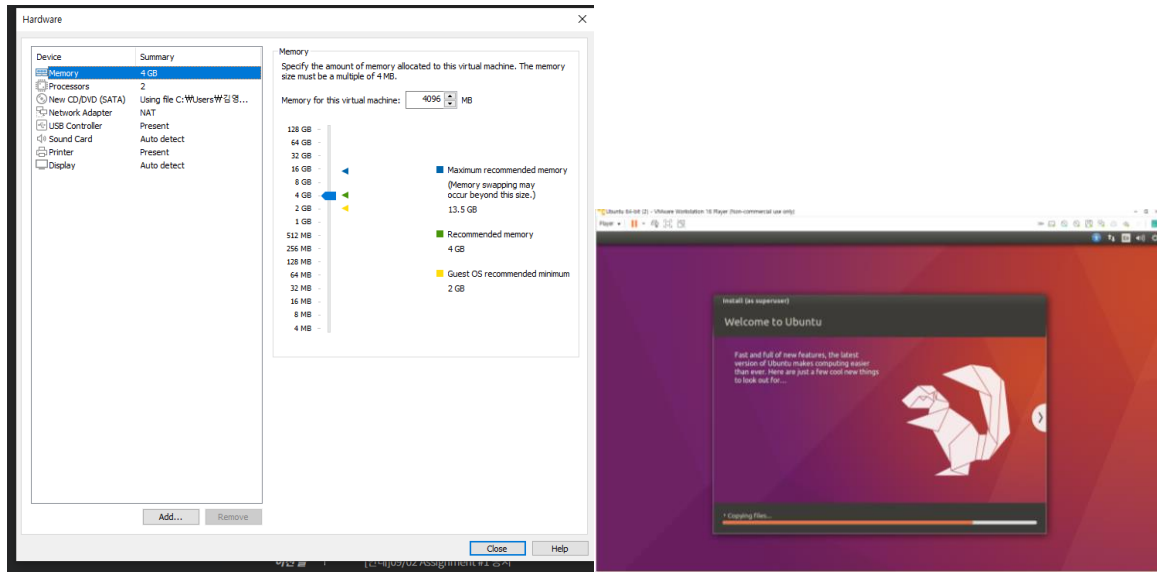
Reference

강의자료

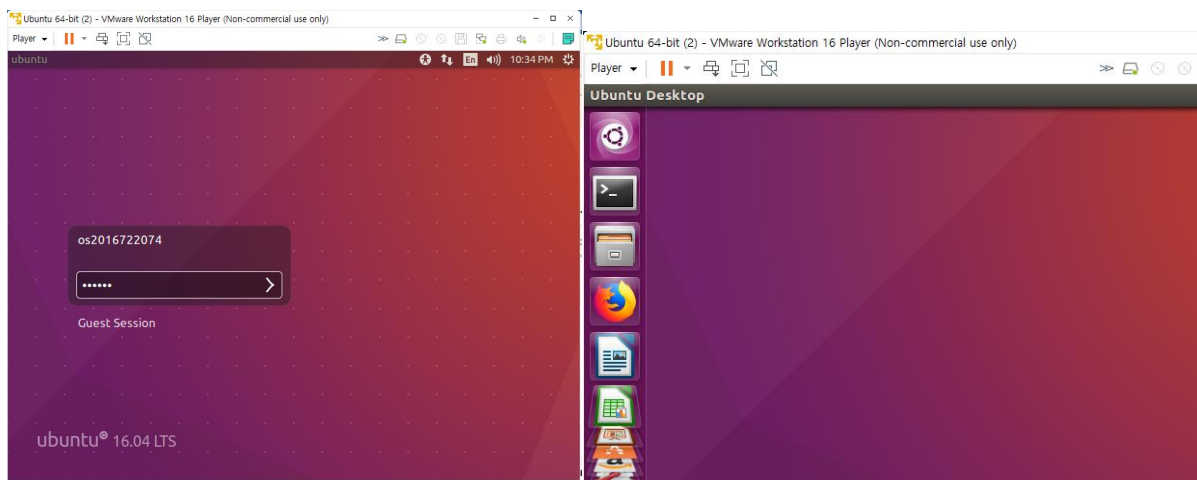
Conclusion



먼저 VMware를 실행시키고 강의자료를 통해 다운받은 iso파일을 이용해 우분투를 설치한다. 사용자 이름과 비밀번호를 설정한다. 그 후 가상머신에 할당할 저장공간 크기를 설정한다. Disk Size는 40Gbytes로 설정했다.



메모리 크기와 CPU개수를 설정한 후 Finish버튼을 누르면 Ubuntu 설치가 진행된다.



Ubuntu 설치 완료 후 방금 전 설정했던 사용자 비밀번호를 입력한다. 터미널을 dock에 고정한다.

Assignment 1-2

Introduction

가상머신에 설치된 우분투에 Kernel을 다운로드 후 kernel을 컴파일 한다. 커널 컴파일 과정을 확인해 본다.

Reference

강의 자료

Conclusion

```
499 packages can be upgraded. Run 'apt list --upgradable' to see them.
os2016722074@ubuntu:~$ cd /home/os2016722074/Downloads/
os2016722074@ubuntu:~/Downloads$ sudo wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.19.67.tar.xz
--2021-09-17 17:20:55-- 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'

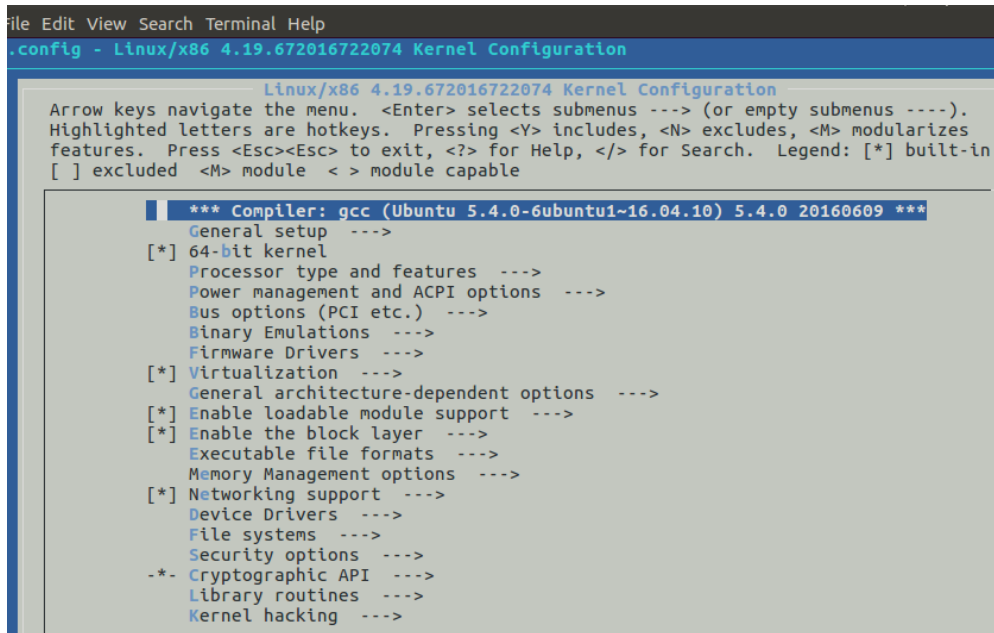
linux-4.19.67.tar.x 100%[=====] 98.51M 10.8MB/s in 9.3s

2021-09-17 17:21:05 (10.6 MB/s) - 'linux-4.19.67.tar.xz' saved [103291756/103291756]
os2016722074@ubuntu:~/Downloads$
```

먼저 우분투의 Downloads에 커널 압축 파일을 다운받는다. 이때 sudo명령어를 사용해 root권한을 얻는다.

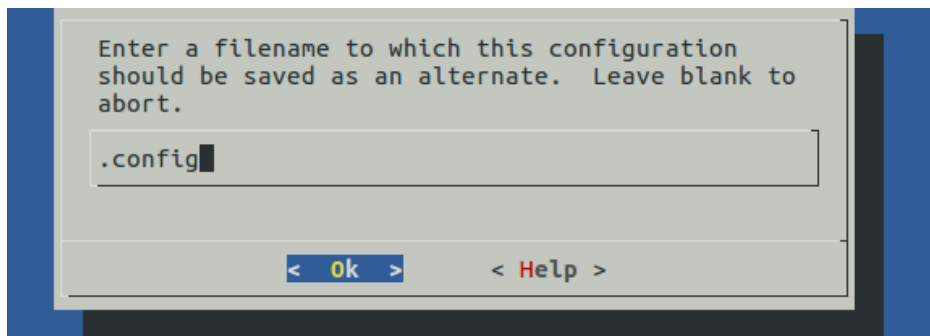
```
block CREDITS drivers include Kbuild lib Makefile README security usr
certs crypto firmware init Kconfig LICENSES mm samples sound virt
os2016722074@ubuntu:~/Downloads/linux-4.19.67$ vi Makefile
os2016722074@ubuntu:~/Downloads/linux-4.19.67$ vi Makefile
os2016722074@ubuntu:~/Downloads/linux-4.19.67$ 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 libelf1 libfl-dev libsigsegv2 libssl-doc libssl1.0.0 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
```

그 후 압축 파일을 압축 풀기 후 vi에디터인 vim을 사용해 Makefile의 코드를 수정한다.

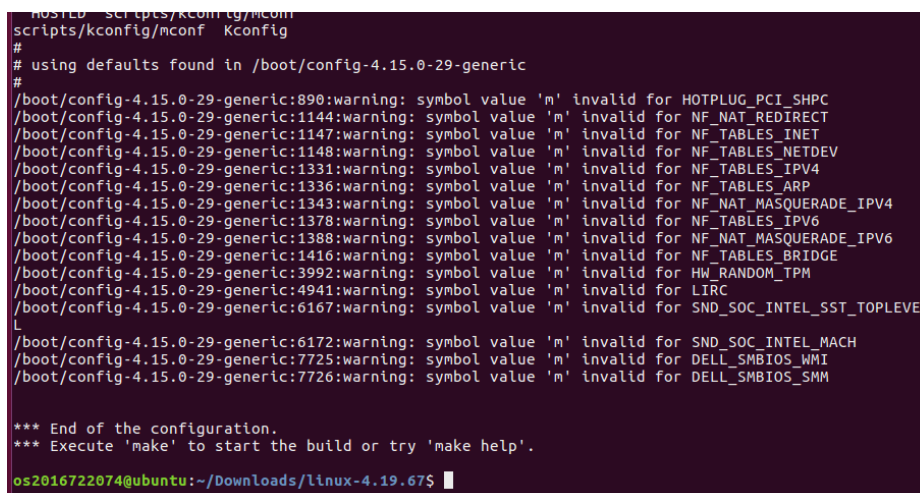


다음으로 커널의 환경설정을 위해 root권한으로 파일을 설치 후 커널환경을 설정한다.

커널 모듈 적재 시 발생할 수 있는 문제를 해결하기 위해 "Enable loadable module support" 의 "Forced module loading"을 체크하고, 컴파일 시 문제가 될 수 있는 모듈을 제거하기 위해 "Device Drivers"의 "Staging drivers"를 체크 해제한다.



모든 설정을 마친 후 .config파일에 저장하고 종료한다.



```
os2016722074@ubuntu:~$ cd ./Downloads/linux-4.19.67/
os2016722074@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
os2016722074@ubuntu:~/Downloads/linux-4.19.67$ make -j4
Makefile:590: include/config/auto.conf: No such file or directory
Makefile:621: include/config/auto.conf.cmd: No such file or directory
HOSTCC    scripts/kconfig/conf.o
HOSTLD    scripts/kconfig/conf
scripts/kconfig/conf --syncconfig Kconfig
SYSHDR    arch/x86/include/generated/asm/unistd_32_ia32.h
SYSTRIP   arch/x86/include/generated/asm/syscall_32.h
```

Make 명령어를 사용해서 Kernel을 컴파일 한다. 이때 인자로 스레드의 개수를 받는데, 스레드의 개수를 4개로 설정해서 컴파일을 진행했다.

```
os2016722074@ubuntu: ~/Downloads/linux-4.19.67
os2016722074@ubuntu:~$ uname -r
4.15.0-29-generic
os2016722074@ubuntu:~$ cd ./Downloads/linux-4.19.67/
os2016722074@ubuntu:~/Downloads/linux-4.19.67$ make modules_install
mkdir: cannot create directory '/lib/modules/4.19.672016722074': Permission denied
Makefile:1250: recipe for target '_modinst_' failed
make: *** [_modinst_] Error 1
os2016722074@ubuntu:~/Downloads/linux-4.19.67$ sudo make modules_install
[sudo] password for os2016722074:
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_ede-x86_64.ko
INSTALL arch/x86/crypto/ghash-clmulni-intel.ko
```

커널 컴파일이 끝나면 컴파일된 모듈을 /lib/modules로 이동한다.

```
make: *** [install] Error 2
os2016722074@ubuntu:~/Downloads/linux-4.19.67$ sudo make install
sh ./arch/x86/boot/install.sh 4.19.672016722074 arch/x86/boot/bzImage \
System.map "/boot"
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 4.19.672016722074
boot/vmlinuz-4.19.672016722074
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 4.19.672016722074
boot/vmlinuz-4.19.672016722074
update-initramfs: Generating /boot/initrd.img-4.19.672016722074
run-parts: executing /etc/kernel/postinst.d/pm-utils 4.19.672016722074 /boot/v
inuz-4.19.672016722074
run-parts: executing /etc/kernel/postinst.d/unattended-upgrades 4.19.672016722
4 /boot/vmlinuz-4.19.672016722074
run-parts: executing /etc/kernel/postinst.d/update-notifier 4.19.672016722074
boot/vmlinuz-4.19.672016722074
run-parts: executing /etc/kernel/postinst.d/zz-update-grub 4.19.672016722074 /
ot/vmlinuz-4.19.672016722074
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-4.19.672016722074
Found initrd image: /boot/initrd.img-4.19.672016722074
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
os2016722074@ubuntu:~/Downloads/linux-4.19.67$
```

다시 root권한으로 명령어를 입력해 컴파일된 커널을 boot loader에 등록한다.

```
os2016722074@ubuntu: ~/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'

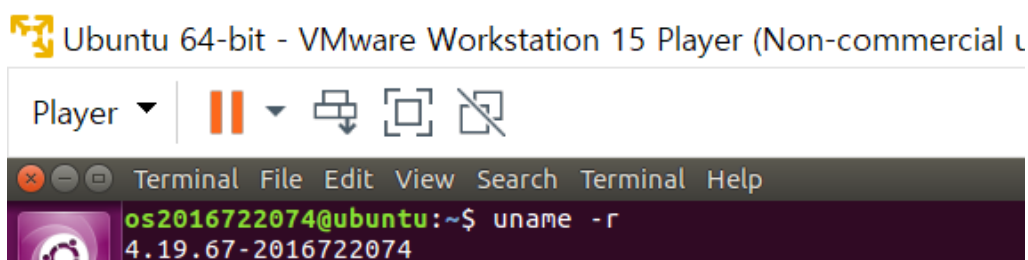
GRUB_DEFAULT=0
#GRUB_HIDDEN_TIMEOUT=0
GRUB_HIDDEN_TIMEOUT_QUIET=false
GRUB_TIMEOUT=10
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX_DEFAULT="quiet"
GRUB_CMDLINE_LINUX="find_preseed=/preseed.cfg auto noprompt priority=critical locale=en_US"

# 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
```

다음으로 Grub 파일을 수정했다. 이때 root권한이 없다면 파일이 readonly로 열리기 때문에 sudo 명령어를 사용해서 파일을 수정할 수 있게 한다.

위의 모든 과정을 마치고 터미널에서 reboot명령어를 입력해서 시스템을 재부팅 했다. 부팅 후 터미널에서 uname -r명령어를 입력해 현재 커널의 버전을 확인했다.



Assignment 1-3

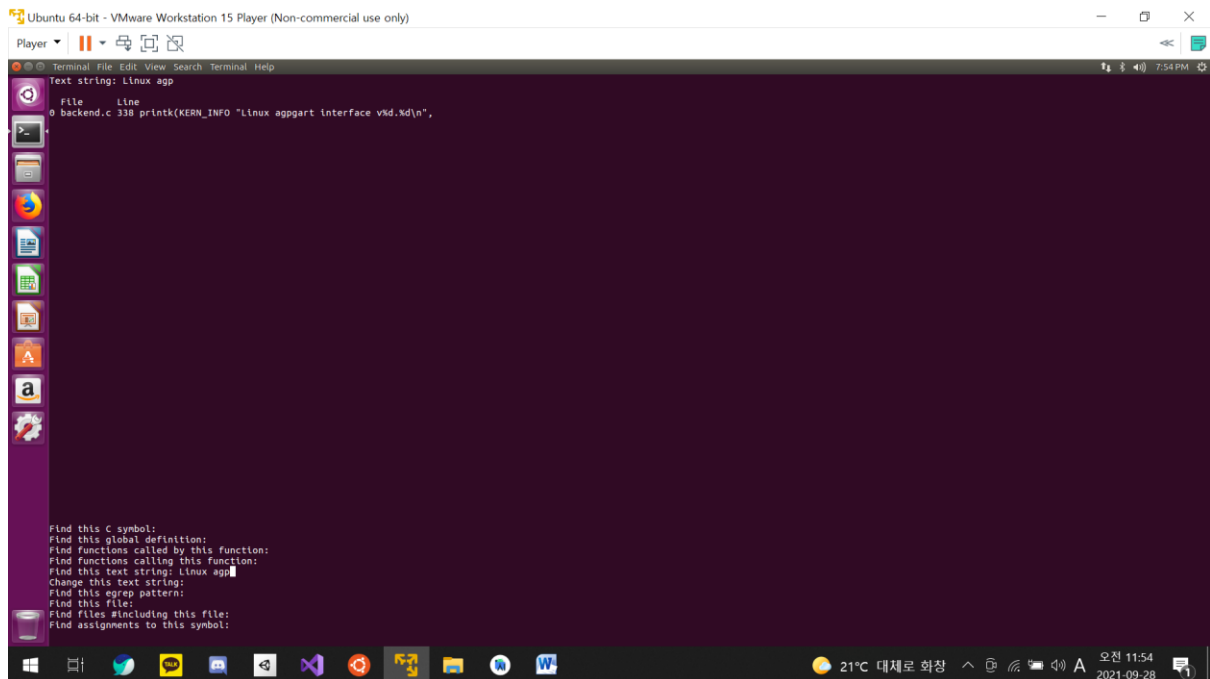
Introduction

cscope를 사용해 커널 코드 내의 코드를 수정하여 Linux agp...이 실행되는 지점에 커널 메시지가 출력되도록 커널 코드를 수정한다. 이때 Linux agp...을 실행 시키는 함수의 함수명과 argument의 값을 출력하도록 커널코드를 수정한다.

Reference

강의 자료

Conclusion

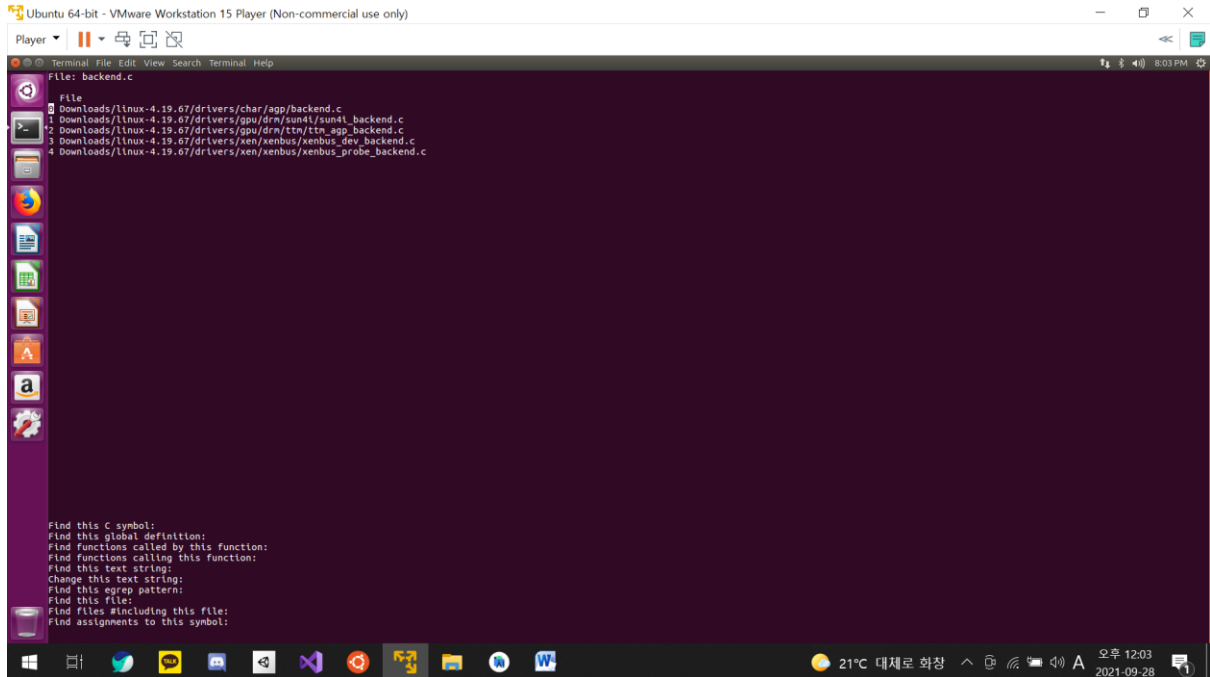


scope에서 Linux agp 라는 문자열을 검색했다. Backend.c 라는 파일의 338번째 줄에 Linux agp가 적혀있는 것을 확인 했다. tab키를 눌러 검색된 파일 목록의 선택 화면으로 바꾸고 엔터 키를 눌러 파일을 확인한다.

```
static int __init agp_init(void)
{
    if (!agp_off)
        printk(KERN_INFO "os2016722074_Linux agpgart interface v%d.%d\\n",
                AGPGART_VERSION_MAJOR, AGPGART_VERSION_MINOR);
    printk(KERN_INFO "os2016722074_arg in agp_init(void)\\n");

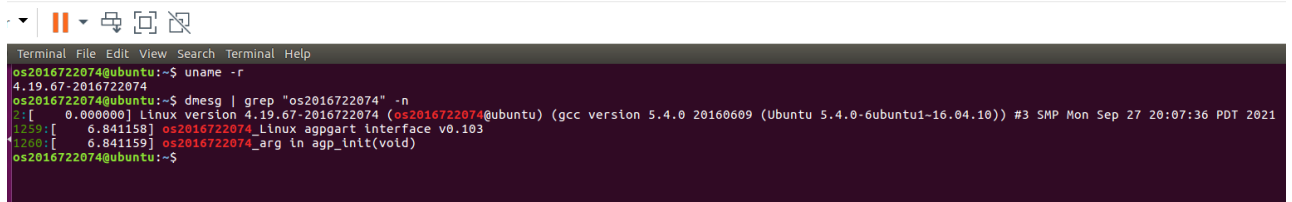
    return 0;
}
```

agp_init 함수 내용을 수정한다. 파일의 내부를 확인해 보니 agp_init함수의 내부에서 Linux agp라는 문자열이 적혀있는 것을 확인할 수 있었다. printk함수는 커널메시지를 출력하는 함수 이므로 해당 함수 내에 적힌 문자열을 수정하여 과제의 요구사항을 만족시켰다. 그 후 다시 printk함수를 호출하여 Linux agp문자열이 적혀있는 함수의 이름과 argument인 void를 적어 커널메시지로 출력하게 했다.



조금전 위에서 확인했다시피 Linux agp문자열이 적혀있던 파일은 backend.c 파일 단 하나뿐이었다. Vi 에디터에서 :wq를 입력해 수정한 코드를 저장하고 다시 cscope화면으로 돌아왔다. Find this file: 에서 backend.c를 입력해서 이 파일이 어느 경로에 위치해 있는지 검색했다.

검색한 소스코드의 path는 Downloads/linux-4.10.67/drivers/char/agp/backend.c 인 것을 확인했다.



다시 커널컴파일의 모든과정을 진행하고 터미널에 reboot명령어를 입력해 시스템을 재부팅 시켰다. 그 후 dmesg | grep 명령어를 사용해 printk()로 출력한 내용 중 특정 문자열이 포함된 메시지를 확인했다. 위에서 backend.c 파일을 수정해서 적었던 내용이 출력된 것을 확인했다.