

운영체제

금(12) 최상호 교수님

Assignment1

컴퓨터정보공학부

2018202076

이연걸

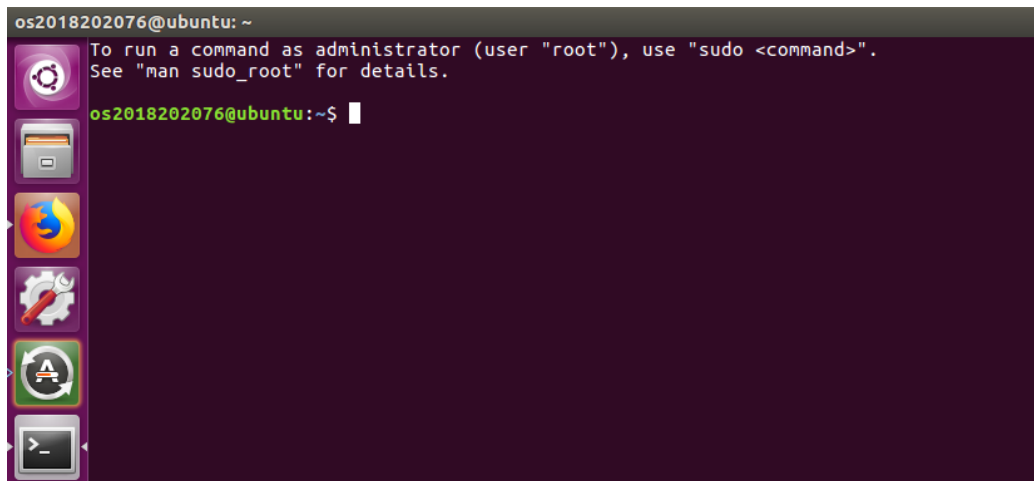
● Introduction

이번 과제는 VMware에 Ubuntu를 과제에서 요구하는 버전, 사양에 맞게 설치하고 과제에서 요구한 리눅스 명령어를 사용해 디렉토리 생성, 파일 생성, 파일 복사, 권한 설정, 파일 출력 등을 수행하는 것이었다. 그리고 kernel을 다운로드하고 커널 컴파일의 모든 과정을 터미널과 vi를 사용해 작성하는 것이었다. 마지막으로 ctags와 cscope를 사용해 Linux agp...이 실행되는 지점을 찾고 커널의 부팅 메시지가 문제에서 제시된 조건에 맞게 출력되도록 소스코드를 수정하는 것이다.

● Conclusion & Analysis

1. Assignment 1-1

리눅스 설치



리눅스 명령어

```
os2018202076@ubuntu: ~/assignment1
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

os2018202076@ubuntu:~$ mkdir assignment1
os2018202076@ubuntu:~$ cd assignment1/
os2018202076@ubuntu:~/assignment1$ touch os.txt
os2018202076@ubuntu:~/assignment1$ ls
os.txt
os2018202076@ubuntu:~/assignment1$ cp os.txt os_copy.txt
os2018202076@ubuntu:~/assignment1$ ls -al
total 8
drwxrwxr-x  2 os2018202076 os2018202076 4096 Sep 13 18:52 .
drwxr-xr-x 17 os2018202076 os2018202076 4096 Sep 13 18:50 ..
-rw-rw-r--  1 os2018202076 os2018202076   0 Sep 13 18:52 os_copy.txt
-rw-rw-r--  1 os2018202076 os2018202076   0 Sep 13 18:51 os.txt
os2018202076@ubuntu:~/assignment1$ chmod 444 os_copy.txt
os2018202076@ubuntu:~/assignment1$ ls -al
total 8
drwxrwxr-x  2 os2018202076 os2018202076 4096 Sep 13 18:52 .
drwxr-xr-x 17 os2018202076 os2018202076 4096 Sep 13 18:50 ..
-r--r--r--  1 os2018202076 os2018202076   0 Sep 13 18:52 os_copy.txt
-rw-rw-r--  1 os2018202076 os2018202076   0 Sep 13 18:51 os.txt
os2018202076@ubuntu:~/assignment1$ vi os.txt
os2018202076@ubuntu:~/assignment1$ vi os.txt
os2018202076@ubuntu:~/assignment1$ vim os.txt
os2018202076@ubuntu:~/assignment1$ cat os.txt
os_2018202076
os2018202076@ubuntu:~/assignment1$
```

2. Assignment 1-2

Kernel Source 다운로드

```
root@ubuntu: /home/os2018202076/Downloads
os2018202076@ubuntu:~$ sudo su
root@ubuntu: /home/os2018202076$ ls
assignment1 Desktop Documents Downloads examples.desktop Music Pictures Public Templates Videos work
root@ubuntu: /home/os2018202076$ cd Downloads/
root@ubuntu: /home/os2018202076/Downloads$ sudo wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.19.67.tar.xz
--2022-09-13 19:01:58-- 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 77%[=====] 76.21M 3.15MB/s eta 7s
```

\$ sudo su

관리자 계정으로 전환하게 해준다.

```
root@ubuntu: /home/os2018202076/Downloads
os2018202076@ubuntu:~$ sudo su
root@ubuntu: /home/os2018202076$ ls
assignment1 Desktop Documents Downloads examples.desktop Music Pictures Public Templates Videos work
root@ubuntu: /home/os2018202076$ cd Downloads/
root@ubuntu: /home/os2018202076/Downloads$ sudo wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.19.67.tar.xz
--2022-09-13 19:01:58-- 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 3.27MB/s in 30s
2022-09-13 19:02:29 (3.23 MB/s) - 'linux-4.19.67.tar.xz' saved [103291756/103291756]
root@ubuntu: /home/os2018202076/Downloads$
```

\$ sudo wget http://cdn.kernel ...

전달된 url에서 파일 다운로드를 가능하게 해준다.

Kernel Source 압축 해제

```
root@ubuntu: /home/os2018202076/Downloads
linux-4.19.67/drivers/net/wireless/narvell/libertas/lf_cs.c
linux-4.19.67/drivers/net/wireless/narvell/libertas/lf_sdio.c
linux-4.19.67/drivers/net/wireless/narvell/libertas/lf_sdio.h
linux-4.19.67/drivers/net/wireless/narvell/libertas/lf_spi.c
linux-4.19.67/drivers/net/wireless/narvell/libertas/lf_spi.h
linux-4.19.67/drivers/net/wireless/narvell/libertas/lf_usb.c
linux-4.19.67/drivers/net/wireless/narvell/libertas/lf_usb.h
linux-4.19.67/drivers/net/wireless/narvell/libertas/main.c
linux-4.19.67/drivers/net/wireless/narvell/libertas/mesh.c
linux-4.19.67/drivers/net/wireless/narvell/libertas/mesh.h
linux-4.19.67/drivers/net/wireless/narvell/libertas/radiotap.h
linux-4.19.67/drivers/net/wireless/narvell/libertas/rx.c
linux-4.19.67/drivers/net/wireless/narvell/libertas/tx.c
linux-4.19.67/drivers/net/wireless/narvell/libertas/types.h
linux-4.19.67/drivers/net/wireless/narvell/libertas_tf/lf_usb.c
linux-4.19.67/drivers/net/wireless/narvell/libertas_tf/Kconfig
linux-4.19.67/drivers/net/wireless/narvell/libertas_tf/Makefile
linux-4.19.67/drivers/net/wireless/narvell/libertas_tf/cnd.c
linux-4.19.67/drivers/net/wireless/narvell/libertas_tf/dbg_defs.h
linux-4.19.67/drivers/net/wireless/narvell/libertas_tf/lf_usb.h
linux-4.19.67/drivers/net/wireless/narvell/libertas_tf/libertas_tf.h
linux-4.19.67/drivers/net/wireless/narvell/libertas_tf/main.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/11ac.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/11ac.h
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/11h.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/11n.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/11n.h
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/11n_aggr.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/11n_aggr.h
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/11n_rxreorder.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/11n_rxreorder.h
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/Kconfig
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/Makefile
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/README
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/cfg80211.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/cfg80211.h
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/cfp.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/cndevt.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/debugfs.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/decl.h
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/ethtool.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/fw.h
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/ie.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/init.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/ioctl.h
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/join.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/main.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/main.h
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/pcie.c
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/pcie.h
linux-4.19.67/drivers/net/wireless/narvell/mwifiex/scan.c

root@ubuntu: /home/os2018202076/Downloads
linux-4.19.67/usr/gltignore
linux-4.19.67/usr/Kconfig
linux-4.19.67/usr/Makefile
linux-4.19.67/usr/gen_init_cpio.c
linux-4.19.67/usr/gen_initramfs_list.sh
linux-4.19.67/usr/initramfs_data.S
linux-4.19.67/virt/
linux-4.19.67/virt/Makefile
linux-4.19.67/virt/kvm/Kconfig
linux-4.19.67/virt/kvm/arm/
linux-4.19.67/virt/kvm/arm/arch32.c
linux-4.19.67/virt/kvm/arm/arch_timer.c
linux-4.19.67/virt/kvm/arm/arm.c
linux-4.19.67/virt/kvm/arm/hyp/
linux-4.19.67/virt/kvm/arm/hyp/arch32.c
linux-4.19.67/virt/kvm/arm/hyp/timer-sr.c
linux-4.19.67/virt/kvm/arm/hyp/vgic-v3-sr.c
linux-4.19.67/virt/kvm/arm/mmio.c
linux-4.19.67/virt/kvm/arm/mmu.c
linux-4.19.67/virt/kvm/arm/perf.c
linux-4.19.67/virt/kvm/arm/pmu.c
linux-4.19.67/virt/kvm/arm/pvcl.c
linux-4.19.67/virt/kvm/arm/trace.h
linux-4.19.67/virt/kvm/arm/vgic/
linux-4.19.67/virt/kvm/arm/vgic/trace.h
linux-4.19.67/virt/kvm/arm/vgic/vgic-debug.c
linux-4.19.67/virt/kvm/arm/vgic/vgic-init.c
linux-4.19.67/virt/kvm/arm/vgic/vgic-lrqfd.c
linux-4.19.67/virt/kvm/arm/vgic/vgic-lis.c
linux-4.19.67/virt/kvm/arm/vgic/vgic-kvm-device.c
linux-4.19.67/virt/kvm/arm/vgic/vgic-mmio-v2.c
linux-4.19.67/virt/kvm/arm/vgic/vgic-mmio-v3.c
linux-4.19.67/virt/kvm/arm/vgic/vgic-mmio.c
linux-4.19.67/virt/kvm/arm/vgic/vgic-mmio.h
linux-4.19.67/virt/kvm/arm/vgic/vgic-v2.c
linux-4.19.67/virt/kvm/arm/vgic/vgic-v3.c
linux-4.19.67/virt/kvm/arm/vgic/vgic-v4.c
linux-4.19.67/virt/kvm/arm/vgic/vgic.c
linux-4.19.67/virt/kvm/arm/vgic/vgic.h
linux-4.19.67/virt/kvm/async_pf.c
linux-4.19.67/virt/kvm/async_pf.h
linux-4.19.67/virt/kvm/coalesced_mmio.c
linux-4.19.67/virt/kvm/coalesced_mmio.h
linux-4.19.67/virt/kvm/eventfd.c
linux-4.19.67/virt/kvm/irqchip.c
linux-4.19.67/virt/kvm/kvm_main.c
linux-4.19.67/virt/kvm/vfio.c
linux-4.19.67/virt/kvm/vfio.h
linux-4.19.67/virt/lib/
linux-4.19.67/virt/lib/Kconfig
linux-4.19.67/virt/lib/Makefile
linux-4.19.67/virt/lib/lrqbypass.c
root@ubuntu: /home/os2018202076/Downloads
```

\$ tar -Jxvf li ...

여러 개의 파일을 하나로 묶거나 풀 때 사용한다.

-를 사용해 옵션을 붙였는데 앞에서부터 xz관련 옵션 명시, tar파일 풀기, 실행 대상의 파일 내용 명시, 작성 대상이 되는 tar 파일의 이름 지정이다.

Kernel Extra Version 수정

```

root@ubuntu:/home/os2018202076/Downloads# ls
linux-4.19.67  linux-4.19.67.tar.xz  splab_commands
root@ubuntu:/home/os2018202076/Downloads# cd linux-4.19.67
root@ubuntu:/home/os2018202076/Downloads/linux-4.19.67# ls
arch  certs  CREDITS  Documentation  firmware  include  ipc  Kconfig  lib  MAINTAINERS  mm  README  scripts  sound  usr
block  COPYING  crypto  drivers  fs  init  kbuild  kernel  LICENSES  Makefile  net  samples  security  tools  virt
root@ubuntu:/home/os2018202076/Downloads/linux-4.19.67#

```

```

root@ubuntu:/home/os2018202076/Downloads/linux-4.19.67
# SPDX-License-Identifier: GPL-2.0
VERSION = 4
PATCHLEVEL = 19
SUBLEVEL = 67
EXTRAVERSION = "-2018202076"
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 += -FR --include-dir=$(CURDIR)

# Avoid funny character set dependencies
unexport LC_ALL
LC_CTYPE=C

```

Vi editor를 사용해 Makefile의 설정을 변경한다.

Kernel 환경 설정 다운로드

```

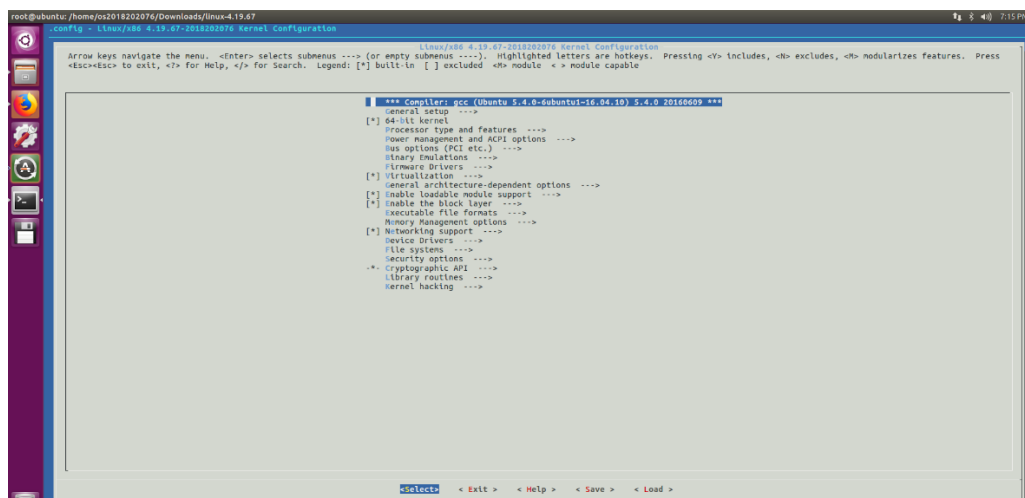
root@ubuntu:/home/os2018202076/Downloads/linux-4.19.67# sudo apt-get install build-essential libncurses5-dev bison flex libssl-dev libelf-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
bison is already the newest version (2:3.0.4.dfsg-1).
build-essential 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).
The following additional packages will be installed:
  libelf1 libssl-doc libssl1.0.0 zlib1g zlib1g-dev
The following NEW packages will be installed:
  libelf-dev libssl-dev libssl-doc zlib1g-dev
The following packages will be upgraded:
  libelf1 libssl1.0.0 zlib1g
3 upgraded, 4 newly installed, 0 to remove and 494 not upgraded.
Need to get 3,823 kB of archives.
After this operation, 10.9 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y

```

\$ sudo apt-get install build-ess ...

Apt-get 패키지 관리자를 사용해 필요한 패키지들을 다운받는다.

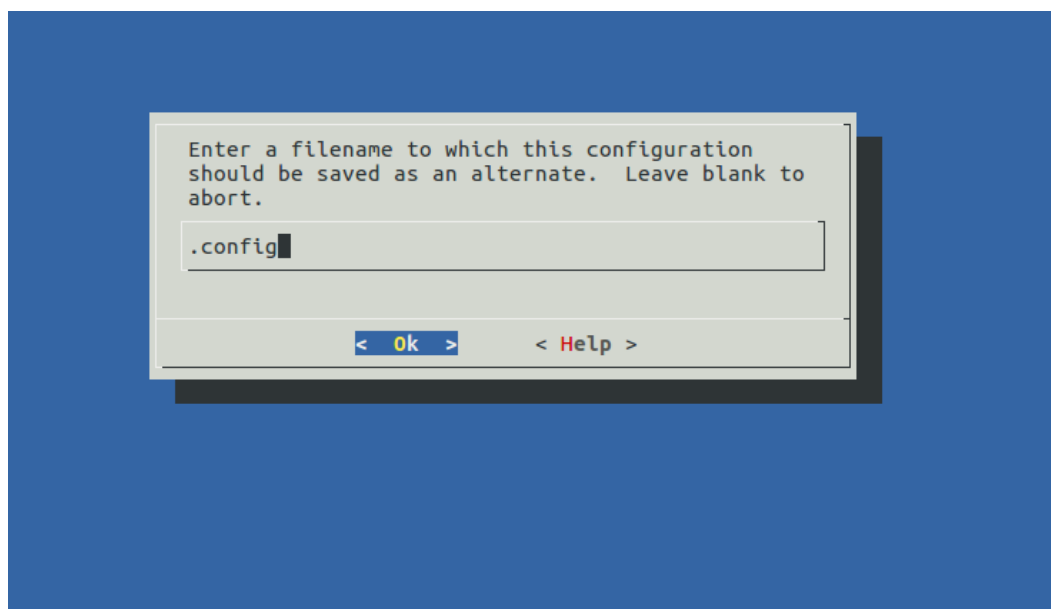
Kernel 환경 설정 \$ make menuconfig



```
--- 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 modules with SHA-512) --->
[ ] Compress modules on installation

<Select> < Exit > < Help > < Save > < Load >
```

```
DMABUF options --->
[*] Auxiliary Display support --->
<M> Parallel port LCD/Keypad Panel support --->
{M} Userspace I/O drivers --->
<M> VFIO Non-Privileged userspace driver framework --->
[*] Virtualization drivers --->
[*] Virtio drivers --->
Microsoft Hyper-V guest support --->
Xen driver support --->
[ ] Staging drivers ----
-- X86 Platform Specific Device Drivers --->
[ ] Platform support for Goldfish virtual devices ----
+ (+)
```



```
root@ubuntu:/home/os2018202076/Downloads/linux-4.19.67# sudo make menuconfig
scripts/kconfig/mconf Kconfig

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

root@ubuntu:/home/os2018202076/Downloads/linux-4.19.67#
```

\$ make menuconfig

커널 환경 설정을 해준다.

Kernel Compile

```
root@ubuntu:/home/os2018202076/Downloads/linux-4.19.67# make -j12
HOSTLD  scripts/kconfig/conf
scripts/kconfig/conf  --syncconfig Kconfig
SYSHDR  arch/x86/include/generated/uapi/asm/unistd_64.h
SYSHDR  arch/x86/include/generated/uapi/asm/unistd_x32.h
SYSHDR  arch/x86/include/generated/uapi/asm/unistd_32.h
HYPERCALLS arch/x86/include/generated/asm/xen-hypercalls.h
SYSTBL  arch/x86/include/generated/asm/syscalls_64.h
SYSHDR  arch/x86/include/generated/asm/unistd_64_x32.h
SYSHDR  arch/x86/include/generated/asm/unistd_32_ia32.h
SYSTBL  arch/x86/include/generated/asm/syscalls_32.h
WRAP    arch/x86/include/generated/uapi/asm/bpf_perf_event.h
WRAP    arch/x86/include/generated/uapi/asm/poll.h
UPD     include/generated/uapi/linux/version.h
UPD     include/config/kernel.release
DESCEND objtool
HOSTCC  /home/os2018202076/Downloads/linux-4.19.67/tools/objtool/fixdep.o
```

```
LD [M]  sound/soc/24c/2x-ltm.ko
LD [M]  sound/synth/emux/snd-emux-synth.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-variax.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/irqbypass.ko
root@ubuntu:/home/os2018202076/Downloads/linux-4.19.67#
```

\$ make -j12

컴파일을 12개의 스레드에 분할해서 실행한다.

Module 설치

```
root@ubuntu:/home/os2018202076/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
```

\$ make modules_install

Makefile의 modules_install 규칙을 실행한다.

Compile된 Kernel을 Boot Loader에 등록

```
root@ubuntu:/home/os2018202076/Downloads/linux-4.19.67# make install
sh ./arch/x86/boot/install.sh 4.19.67-2018202076 arch/x86/boot/bzImage \
System.map "/boot"
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 4.19.67-2018202076 /boot/vmlinuz-4.19.67-2018202076
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 4.19.67-2018202076 /boot/vmlinuz-4.19.67-2018202076
update-initramfs: Generating /boot/initrd.img-4.19.67-2018202076
run-parts: executing /etc/kernel/postinst.d/pm-utils 4.19.67-2018202076 /boot/vmlinuz-4.19.67-2018202076
run-parts: executing /etc/kernel/postinst.d/unattended-upgrades 4.19.67-2018202076 /boot/vmlinuz-4.19.67-2018202076
run-parts: executing /etc/kernel/postinst.d/update-notifier 4.19.67-2018202076 /boot/vmlinuz-4.19.67-2018202076
run-parts: executing /etc/kernel/postinst.d/zz-update-grub 4.19.67-2018202076 /boot/vmlinuz-4.19.67-2018202076
Generating grub configuration file ...
Warning: Setting GRUB_TIMEOUT to a non-zero value when GRUB_HIDDEN_TIMEOUT is set is no longer supported.
Found linux image: /boot/vmlinuz-4.19.67-2018202076
Found initrd image: /boot/initrd.img-4.19.67-2018202076
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/os2018202076/Downloads/linux-4.19.67#
```

\$ make install

/arch/x86/boot/ 경로에 있는 bzImage라는 이름의 Kernel image를 /boot에 복사하고 System Map(System.map)을 /boot로 복사한다 .그리고 Grub 부트로더에 자동으로 등록해준다.

이 세가지 기능을 가능하게 하는 install 규칙을 실행한다.

Grub 설정

```
root@ubuntu: ~
# 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'

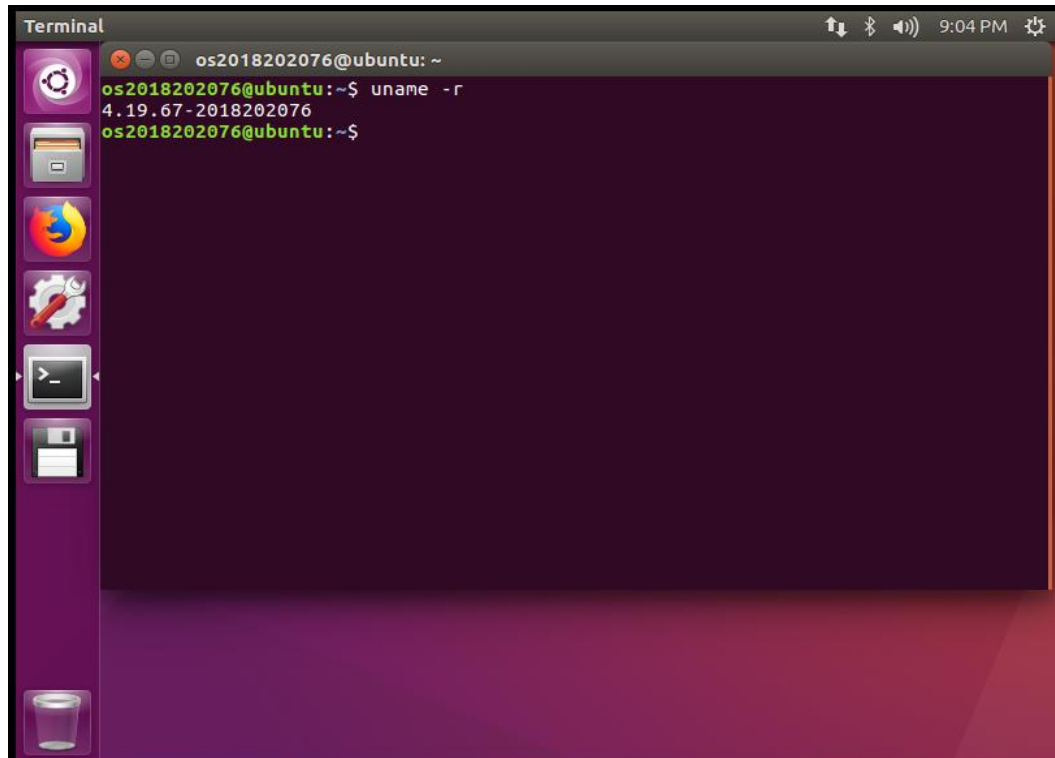
GRUB_DEFAULT=0
#GRUB_HIDDEN_TIMEOUT=0
GRUB_HIDDEN_TIMEOUT_QUIET=false
GRUB_TIMEOUT=10
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX_DEFAULT="quiet"
GRUB_CMDLINE_LINUX="find_preseed=/preseed.cfg auto noprompt priority=critical locale=en_US"

# 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

# The resolution used on graphical terminal
# (only when using terminal=console or terminal=gfx)
```

재부팅 후 커널 버전 확인



```
$ uname -r
```

uname은 시스템 정보를 출력해준다. -r 옵션으로 커널의 릴리즈 정보를 출력했다.

3. Assignment 1-3

리눅스 커널 코드에서 수정한 부분 명시

```
static int __init agp_init(void)
{
    if (!agp_off) {
        printk(KERN_INFO "os2018202076_Linux agpgart interface v%d.%d\n",
            AGPGART_VERSION_MAJOR, AGPGART_VERSION_MINOR);
        printk(KERN_INFO "os2018202076_arg in agp_init (void)\n");
    }
    return 0;
}
```

Linux agp...이 실행되는 지점은 agp_init()이다. 그렇기 때문에 과제에서 요구한 조건대로 Linux agp...이 실행되는 지점에 커널 메시지가 과제에서 요구한 조건에 부합하게 "Linux apart interface v%d.%d\n" 앞에 os2018202076_을 추가해 주었다.

그리고 Linux의 agp를 실행시키는 함수의 함수명인 agp_init과 argument의 값인 void를 출력하도록 커널 코드를 수정했다.

수정한 코드는 다음과 같다.

```
printk(KERN_INFO "os2018202076_arg in agp_init (void)\n");
```

소스코드 경로 작성

```
static int __init agp_init(void)
{
    if (!agp_off) {
        printk(KERN_INFO "os2018202076_Linux agpgart interface v%d.%d\n",
            AGPGART_VERSION_MAJOR, AGPGART_VERSION_MINOR);
        printk(KERN_INFO "os2018202076_arg in agp_init (void)\n");
    }
    return 0;
}

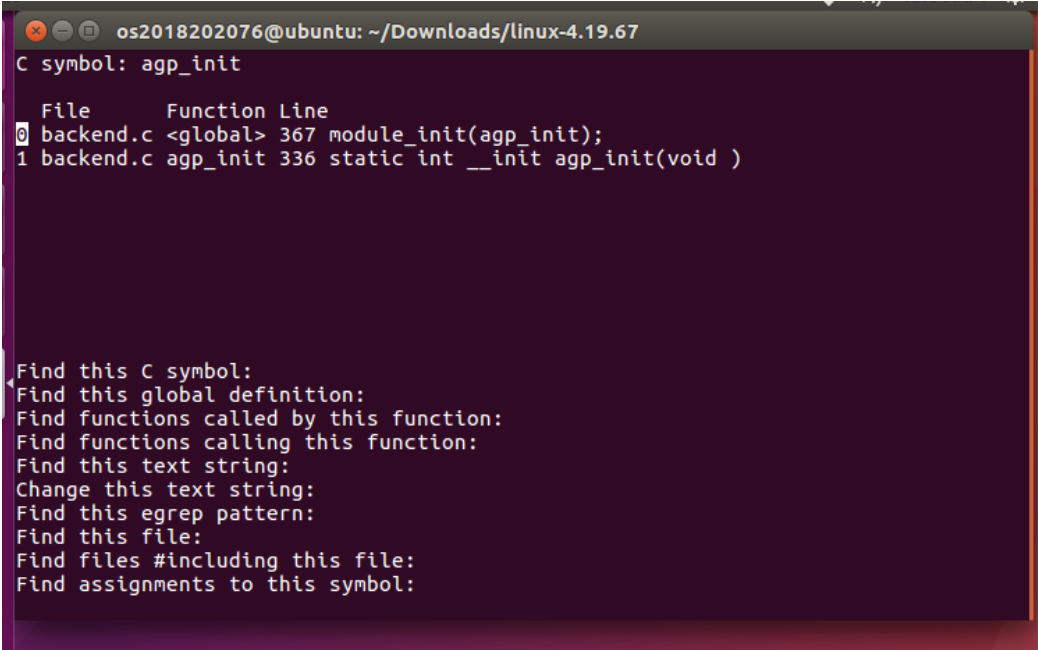
static void __exit agp_exit(void)
"drivers/char/agp/backend.c" 369L, 9187C 336,1 93%
```

더 정확한 경로는 다음과 같다.

```
os2018202076@ubuntu:~/Downloads/linux-4.19.67/drivers/char/agp$ ls | grep back
backend.c
backend.o
os2018202076@ubuntu:~/Downloads/linux-4.19.67/drivers/char/agp$ pwd
/home/os2018202076/Downloads/linux-4.19.67/drivers/char/agp
os2018202076@ubuntu:~/Downloads/linux-4.19.67/drivers/char/agp$
```

/home/os2018202076/Downloads/linux-4.19.67/drivers/char/agp/backend.c

검색한 캡처 화면 첨부



```
os2018202076@ubuntu: ~/Downloads/linux-4.19.67
C symbol: agp_init

File      Function Line
0 backend.c <global> 367 module_init(agp_init);
1 backend.c agp_init 336 static int __init agp_init(void )

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:
```

결과 화면 캡처

```
os2018202076@ubuntu:~/Downloads/linux-4.19.67$ dmesg | grep "os2018202076" -n
1402:[ 14.019871] os2018202076_Linux agpgart interface v0.103
1403:[ 14.019872] os2018202076_arg in agp_init (void)
os2018202076@ubuntu:~/Downloads/linux-4.19.67$
```

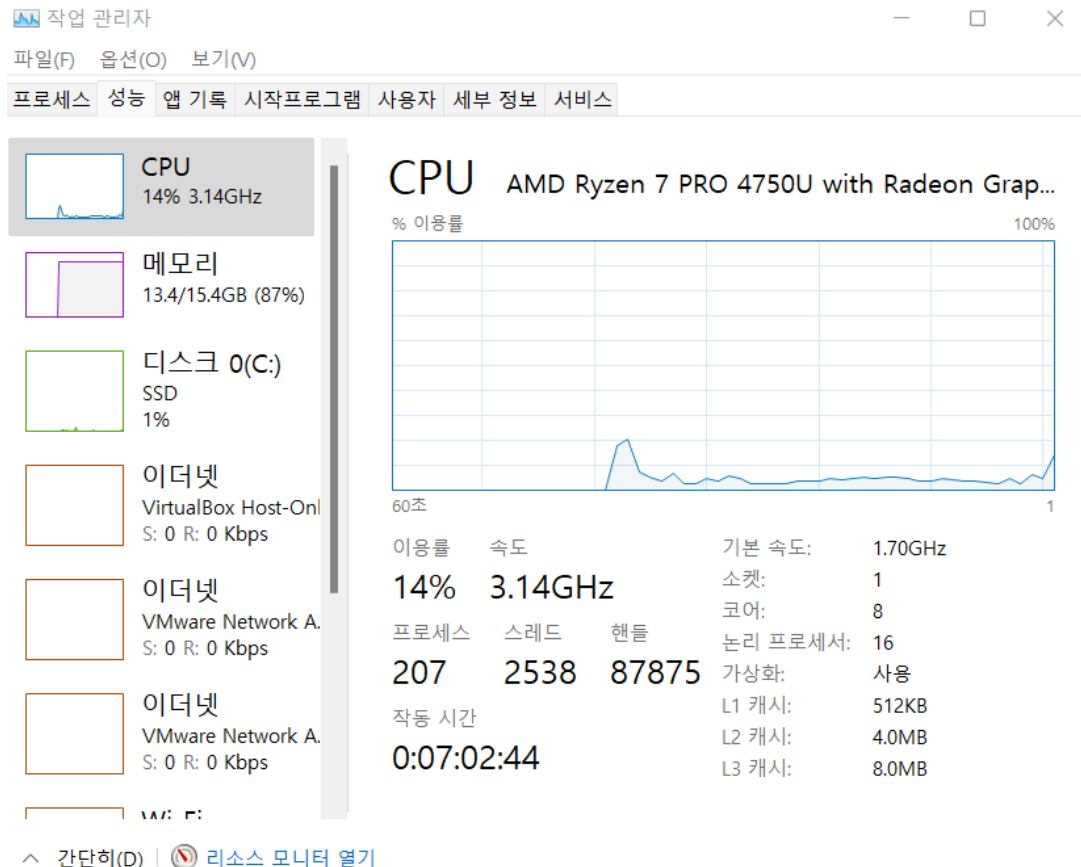
● 고찰

이번 과제에선 가상 머신에 Linux를 설치하고 명령어를 사용해 파일, 디렉토리 생성, 수정, 삭제 권한 변경, 커널 컴파일, start_kernel의 이해 등을 요구했다.

과제를 진행하다가 의문사항이 한가지 있었다. 과제 조건에 “가상 머신에 할당된 CPU core 수의 1.5배 ~ 2배 정도면 적합” 이라는 말이 쓰여 있었다. 왜 thread의 수가 CPU 코어 수의 2배인 것일까? 그에 대한 해답은 하이퍼스레딩이었다.

하이퍼 스레딩은 1코어를 가진 CPU 프로세서를 논리적인 2코어로 만드는 기술이다. 커널 컴파일과 같은 처리할 데이터가 많은 작업을 진행할 때 프로세서의 처리량을 높여 성능을 높이는 기술이다.

내 노트북에 설치된 CPU도 하이퍼스레딩을 지원하는지 여부가 궁금해서 확인해 보았다.



작업 관리자의 성능 탭의 CPU를 코어와 논리 프로세서가 나뉘어져 있다. 코어 개수는 8개, 논리 프로세서 개수는 16개로 적혀 있다. 하이퍼 스레딩을 지원하는 CPU 프로세서는 코어 수와 논리 프로세서 수가 다르게 표시되기에 현재 CPU 프로세서가 하이퍼스레딩을 지원함을 확인할 수 있었다.

실수했던 부분도 있었는데 ctags, cscope은 Man Page를 참고해 어렵지 않게 사용할 수 있었지만 과제의 조건이었던 "본인이 기술한 start_kernel()에서 Linux agp...이 실행되는 지점에 기반하여 수정한 코드"를 제대로 이해하지 못했다.

리눅스의 부팅 과정 중 GRUB를 거쳐 Kernel단계에 들어가면 start_kernel()이 실행되는데 Linux_agp이 실행되는 지점은 start_kernel이 아닌 agp_init이다. 처음에 start_kernel() 함수에 두 문구를 출력하는 printk문을 넣어서 두 문구가 중복되어 출력되었는데 agp_init 함수로 옮겨서 해결할 수 있었다.

● Reference

-

<https://jinstory.net/2462#:~:text=%ED%95%98%EC%9D%B4%ED%8D%BC%EC%8A%A4%EB%A0%88%EB%94%A9%20%ED%99%95%EC%9D%B8%20%EB%B0%A9%EB%B2%95,%EC%96%B4%EB%A0%B5%EC%A7%80%20%EC%95%8A%EA%B2%8C%20%ED%99%95%EC%9D%B8%ED%95%A0%20%EC%88%98%20%EC%9E%88%EC%8A%B5%EB%8B%88%EB%8B%A4.>