

운영체제

Assignment1

한준호

2018741035

최상호 교수님

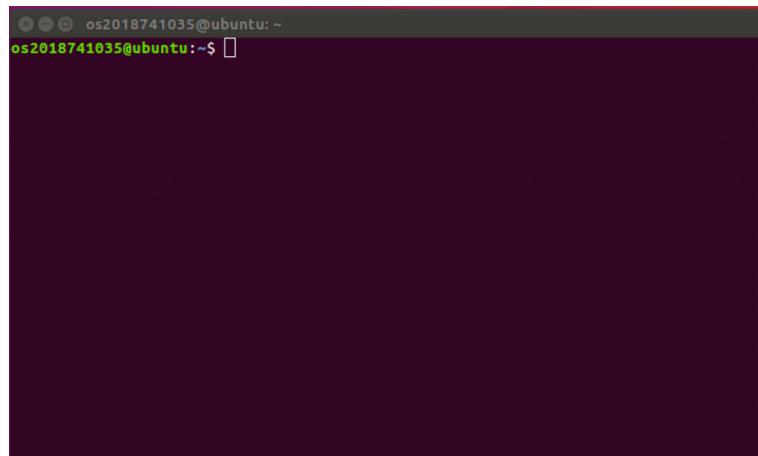
Assignment 1-1

Linux Installation

- Introduction

운영체제를 본격적으로 배우기 앞서 가상머신을 이용해 실습 환경을 구축했다. 맥북을 사용하기 때문에 교수님께서 추천해주신 VMWare 대신에 원래 사용하던 Virtual Box를 이용해서 Ubuntu를 설치해주었다. 교수님이 채점하시는 환경과 맞추기 위해 서 올려주신 링크를 통해 Ubuntu 16.04.5 Desktop 64bits 버전을 새로 설치했다.

- Conclusion & Analysis



Virtual Box를 통해 Ubuntu 16.04.5 Desktop 64bits를 설치하고 계정을 os학번으로 만들어주었다. 이 과정에서 Host name을 너무 길게 만들어 hostname 명령어를 통해 예시와 같이 Host name을 'ubuntu'로 변경해주었다.

- 고찰

예전에 Virtual Box를 통해 우분트를 설치해본적이 있었지만 실습 환경에 맞는 우분트 버전으로 새로 설치하는 과정에서 Virtual Box에 메모리 할당에 대한 부분에 어려움을 느꼈지만 추가적으로 인터넷 자료들을 참고하여 해결하였다.

설치 초기에는 계정ID와 hostname을 똑같이 설정해주었지만 hostname이 너무 길어 불편함을 느껴서 예시와 같이 'ubuntu'로 변경해주어 해결하였다.

- Reference

- [Mac] 버추얼박스(VirtualBox)에 우분투(Ubuntu) 설치하기, <https://til.younho9.dev/docs/etc/setting/mac-버추얼박스-virtualbox-에-우분투-ubuntu-설치하기/>
- 우분투 16.04 hostname 확인 및 변경 방법, <https://extrememanual.net/13014>

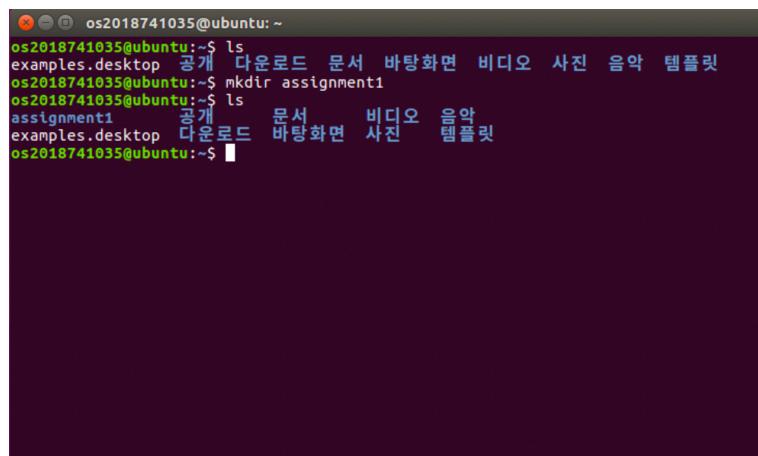
Linux command

- Introduction

리눅스의 기본적인 명령어들을 학습했다. 학습한 내용으로는 터미널을 이용해 새로운 디렉터리를 생성하고 디렉터리로 이동하는 명령어, 텍스트 파일을 생성하거나 복사하는 명령어, 파일을 수정 및 출력하는 명령어, 파일의 권한을 확인하고 변경해주는 명령어들을 학습했다.

- Conclusion & Analysis

1. 'assignment1'명의 디렉터리 생성하기



```
os2018741035@ubuntu:~$ ls
examples.desktop  공개  다운로드  문서  바탕화면  비디오  사진  음악  템플릿
os2018741035@ubuntu:~$ mkdir assignment1
os2018741035@ubuntu:~$ ls
assignment1      공개  문서  비디오  음악
examples.desktop  다운로드  바탕화면  사진  템플릿
os2018741035@ubuntu:~$ █
```

현재 디렉터리에 있는 내용을 확인하고 'assignment1'이라는 새로운 디렉터리를 생성해주었다.

- ls

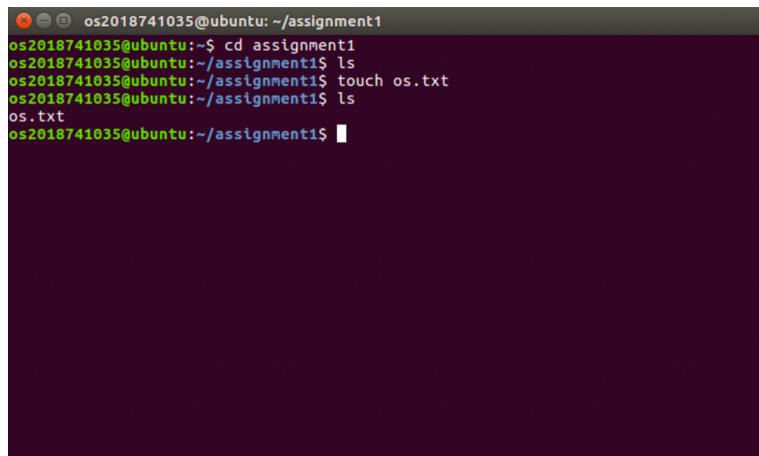
현재 디렉터리에 있는 내용(디렉터리, 파일 등)을 출력하는 명령어

- l : 파일들을 나열할때 자세히 출력한다.
- a : 경로안의 모든 파일을 나열한다.(숨김파일도 포함)
- R : 위치한 디렉토리 하부 디렉토리의 파일까지 모두 출력한다.
- h : 파일크기를 해석파기 편하게 출력한다.
- r : 출력 결과를 내림차순으로 정렬한다.
- t : 출력 결과를 파일이 수정된 시간을 기준으로 정렬한다.

- mkdir [디렉터리명]

현재 디렉터리에 새로운 디렉터리를 생성하는 명령어

2. assignment1 디렉터리 이동 후 'os.txt'명의 빈 파일 생성



```
os2018741035@ubuntu:~/assignment1
os2018741035@ubuntu:~$ cd assignment1
os2018741035@ubuntu:~/assignment1$ ls
os2018741035@ubuntu:~/assignment1$ touch os.txt
os2018741035@ubuntu:~/assignment1$ ls
os.txt
os2018741035@ubuntu:~/assignment1$
```

위에서 생성한 디렉터리로 이동해 'os.txt'라는 빈 파일을 생성하였다.

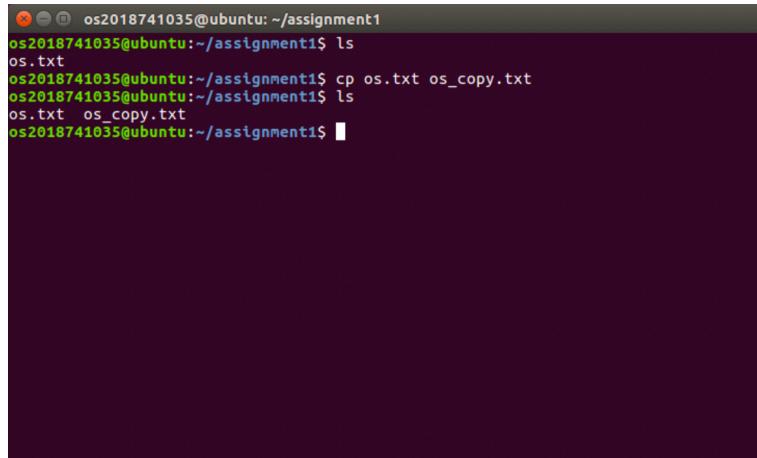
- cd [디렉터리 경로]

디렉터리를 이동하는 명령어

- touch [파일명]

빈 텍스트 파일을 만드는 명령어

3. os.txt를 os_copy.txt명으로 복사

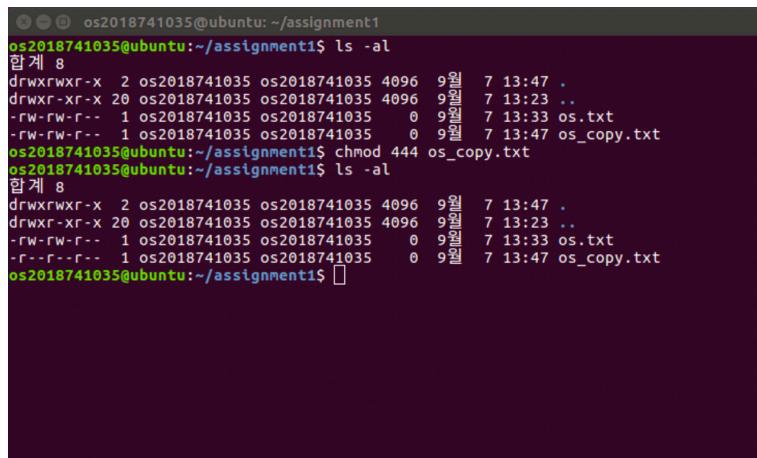


```
os2018741035@ubuntu:~/assignment1
os2018741035@ubuntu:~/assignment1$ ls
os.txt
os2018741035@ubuntu:~/assignment1$ cp os.txt os_copy.txt
os2018741035@ubuntu:~/assignment1$ ls
os.txt  os_copy.txt
os2018741035@ubuntu:~/assignment1$
```

• cp [파일1 or 디렉토리1] [파일2 or 디렉터리2]

파일을 복사하는 명령어

4. os_copy.txt의 권한을 모든 대상에게 읽기만 부여



```
os2018741035@ubuntu:~/assignment1
os2018741035@ubuntu:~/assignment1$ ls -al
합계 8
drwxrwxr-x  2 os2018741035 os2018741035 4096 9월  7 13:47 .
drwxr-xr-x 20 os2018741035 os2018741035 4096 9월  7 13:23 ..
-rw-rw-r--  1 os2018741035 os2018741035    0 9월  7 13:33 os.txt
-rw-rw-r--  1 os2018741035 os2018741035    0 9월  7 13:47 os_copy.txt
os2018741035@ubuntu:~/assignment1$ chmod 444 os_copy.txt
os2018741035@ubuntu:~/assignment1$ ls -al
합계 8
drwxrwxr-x  2 os2018741035 os2018741035 4096 9월  7 13:47 .
drwxr-xr-x 20 os2018741035 os2018741035 4096 9월  7 13:23 ..
-rw-rw-r--  1 os2018741035 os2018741035    0 9월  7 13:33 os.txt
-r--r--r--  1 os2018741035 os2018741035    0 9월  7 13:47 os_copy.txt
os2018741035@ubuntu:~/assignment1$
```

ls -al 명령어를 통해 현재 디렉터리에 있는 파일의 권한을 확인하고 chmod 명령어를 이용해서 os_copy.txt의 권한을 변경해주었다.

리눅스는 여러 사용자가 들어와 사용하는 멀티유저 시스템이기 때문에 각각의 파일에 대한 보안 상에 문제 때문에 각각 파일에 대한 권한(퍼미션)을 변경해줄 수 있다.

● 퍼미션(권한) 종류

- 읽기 (r): 파일의 읽기 권한
- 쓰기 (w): 파일의 쓰기 권한
- 실행 (x): 파일의 실행 권한

● 퍼미션의 사용자 지정

- 소유자: 소유자에 대한 퍼미션 지정
- 그룹: 소유 그룹에 대한 퍼미션 지정
- 공개: 모든 사용자들에 대한 퍼미션 지정

초기 os_copy.txt 권한 -rw-rw-r— : 소유자: rw-, 그룹: rw-, 공개: r-

-> 이 파일에 대해서는 소유자는 읽기(r), 쓰기(w)를 허용.

-> 파일의 소유 그룹에 속하고 있는 사용자들은 읽기(r), 쓰기(w)를 허용.

-> 이외의 나머지 모든 사용자들은 읽기(r)를 허용.

- chmod [변경될 퍼미션 값] [변경할 파일]

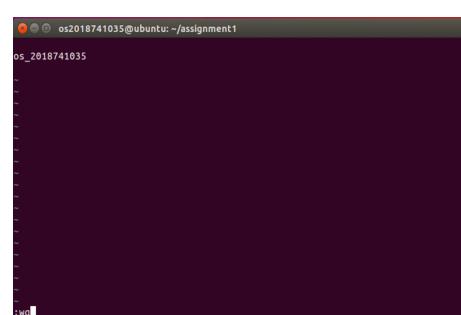
권한을 변경해주는 명령어 (퍼미션 값: 읽기(r) = 4, 쓰기(w) = 2, 실행(x) = 1)

5. os.txt에 os_본인학번 작성 후 터미널에 출력

```
os2018741035@ubuntu:~/assignment1
os2018741035@ubuntu:~/assignment1$ vi os.txt
os2018741035@ubuntu:~/assignment1$ cat os.txt

os_2018741035

os2018741035@ubuntu:~/assignment1$
```



〈vi [파일명] 명령 실행 후〉

- vi 명령어 - 읽기, 수정 가능

- 파일 열기: vi [파일명]

- : 커서 맨 아래로 이동
 - w 저장하기
 - q 나가기
 - wq 저장하고 종료하기
 - q! 저장하지 않고 나가기

- cat 명령어 - 출력만 가능 (읽기, 보기)
 - 파일 열기(출력, 보기): cat [파일명]
 - 각 행에 번호 붙여서 출력하기: cat -b [파일명]
 - 빈 행에 번호 붙여서 출력하기: cat -n [파일명]

- 고찰

리눅스의 여러 가지 명령어들을 학습하고 써보면서 풀더 만들기와 같은 사소한 것에서 권한 변경, 설치까지 모든 것을 터미널을 통해 조작하고 상호작용하는 것에 대해 익숙해졌다. 지금까지는 사용하지 않았던 권한을 변경을 하는 명령어를 통해 파일 각각의 권한까지 설정해 줄 수 있다는 것이 리눅스의 파일 관리 체계가 세밀한 것 같다는 생각이 들었다. 처음에 vim 패키지를 설치하지 않고 기본 설치된 vi를 통해 과제를 수행하였는데 vim 패키지를 설치하고 다시 과제를 수행해 본 결과 편집이 훨씬 편했다.

- Reference

- [Linux] 리눅스 mkdir 명령어(디렉토리 생성) 사용법 & 옵션 정리, <https://coding-factory.tistory.com/753>
- [Linux] 파일 관련 명령어 총정리(생성, 출력, 복사, 이동, 삭제), <https://coding-factory.tistory.com/501>
 - [기본] 리눅스의 퍼미션(권한)을 조정하기 (chmod, chown), <https://conory.com/blog/19194>
 - [Linux] vi, cat 파일 편집, 파일 읽기 명령어, <https://javaoop.tistory.com/43>

Assignment 1-2

Kernel 4.19.67 Compile

- Introduction

System을 구성하는 Memory, File, Device와 같은 중요한 자원들의 관리하고 Process management, Time management, CPU scheduling 등 OS의 핵심 역할을 수행하는 Linux의 Kernel을 터미널을 통해 다운로드 해본다. 이 과정에서 사용되는 명령어들의 기능을 학습한다.

- Conclusion & Analysis

1. Kernel 설치전 apt(Advanced Package Tool) update 하기

```
os2018741035@ubuntu:~$ apt update
파키지 목록을 읽는 중입니다... 완료
W: chmod 0700 of directory /var/lib/apt/lists/partial failed - SetupAPTPartialDirectory (1: 명령을 해용하지 않음)
E: 잠금 파일 /var/lib/apt/lists/lock 파일을 열 수 없습니다 - open (13: 허가 거부)
E: /var/lib/apt/lists/ 디렉터리를 잠글 수 없습니다
W: /var/cache/apt/pkgcache.bin 파일을 삭제하는데 문제가 있습니다 - RemoveCaches (13: 허가 거부)
W: /var/cache/apt/srcpkgcache.bin 파일을 삭제하는데 문제가 있습니다 - RemoveCaches (13: 허가 거부)

os2018741035@ubuntu:~$ sudo apt update
[sudo] password for os2018741035:
[download] 1 http://security.ubuntu.com/ubuntu xenial-security InRelease [99.8 kB]
[download] 2 http://security.ubuntu.com/ubuntu xenial-updates InRelease [99.8 kB]
[download] 3 http://kr.archive.ubuntu.com/ubuntu xenial-backports InRelease [97.4 kB]
[download] 4 http://security.ubuntu.com/ubuntu xenial-security/main amd64 DEP-11 Metadata [93.0 kB]
[download] 5 http://kr.archive.ubuntu.com/ubuntu xenial-updates/main amd64 DEP-11 Metadata
[download] 6 http://security.ubuntu.com/ubuntu xenial-security/universe amd64 DEP-11 Metadata [136 kB]
[download] 7 http://security.ubuntu.com/ubuntu xenial-security/multiverse amd64 DEP-11 Metadata [2,404 kB]
[download] 8 http://kr.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 DEP-11 Metadata [285 kB]
[download] 9 http://kr.archive.ubuntu.com/ubuntu xenial-updates/multiverse amd64 DEP-11 Metadata [5,956 kB]
[download] 10 http://kr.archive.ubuntu.com/ubuntu xenial-backports/main amd64 DEP-11 Metadata [32 kB]
[download] 11 http://kr.archive.ubuntu.com/ubuntu xenial-backports/universe amd64 DEP-11 Metadata [6,612 kB]
[download] 12 http://kr.archive.ubuntu.com/ubuntu xenial-backports/multiverse amd64 DEP-11 Metadata [391 kB]
파키지 목록을 읽는 중입니다... 완료
업데이트된 정보를 확인하려면 'apt-get update'를 실행하세요.
497 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

설치에는 관리자 권한이 필요하기 때문에 sudo를 붙여서 관리자 권한으로 명령어를 실행하여 dpkg 패키징 설치에 필요한 apt를 update 해줬다. 아래는 계정의 sudo 상태를 확인하고 루트 계정으로 로그인하는 명령어이다.

```
os2018741035@ubuntu:~$ whoami
os2018741035
os2018741035@ubuntu:~$ sudo su
sudo: unable to resolve host ubuntu
root@ubuntu:/home/os2018741035#
```

2. Kernel Source Download

```
os2018741035@ubuntu:~$ cd /home/os2018741035/다운로드
os2018741035@ubuntu:~/다운로드$ sudo wget http://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.19.67.tar.xz
sudo: unable to resolve host ubuntu
[sudo] password for os2018741035:
--2022-09-11 20:47:19--  http://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
접속 cdn.kernel.org (cdn.kernel.org)|146.75.49.176|:80... 접속됨.
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 11.1MB/s   in 9.0s
2022-09-11 20:47:28 (10.9 MB/s) - 'linux-4.19.67.tar.xz' saved [103291756/103291756]

os2018741035@ubuntu:~/다운로드$
```

Kernel을 compile 하기 위해 터미널을 통해 kernel 홈페이지에서 Kernel Source를 다운로드 해줬다.

3. Kernel Source 압축 해제

```
os2018741035@ubuntu:~/다운로드$ ls  
linux-4.19.67.tar.xz  
os2018741035@ubuntu:~/다운로드$ tar -Jxvf linux-4.19.67.tar.xz
```

다운로드한 Kernel Source를 압축 해제 해줬다.

4. Kernel Extra Version 수정

```
os2018741035@ubuntu:~/다운로드$ ls  
linux-4.19.67 linux-4.19.67.tar.xz  
os2018741035@ubuntu:~/다운로드$ cd linux-4.19.67/  
os2018741035@ubuntu:~/다운로드/linux-4.19.67$ ls  
COPYING Kconfig README crypto include lib scripts usr  
CREDITS LICENSES arch drivers init mm security virt  
Documentation MAINTAINERS block firmware ipc net sound  
Kbuild Makefile certs fs kernel samples tools  
os2018741035@ubuntu:~/다운로드/linux-4.19.67$ vi Makefile  
os2018741035@ubuntu:~/다운로드/linux-4.19.67$ █  
  
# SPDX-License-Identifier: GPL-2.0  
VERSION = 4  
PATCHLEVEL = 19  
SUBLEVEL = 67  
EXTRAVERSION = os2018741035  
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
```

5,27

꼭대기

위에서 압축 해제한 디렉터리에 들어가 vi 편집기를 이용해 Makefile의 Extra Version 을 ‘-학번’으로 수정해주었다.

5. Kernel Compile시 필요한 라이브러리 설치

```
os2018741035@ubuntu:~/linux-4.19.67$ sudo apt install build-essential libncurses5-dev bison flex libssl-dev libelf-dev  
sudo: unable to resolve host ubuntu  
[sudo] password for os2018741035:  
패키지 목록을 읽는 중입니다... 완료  
의존성 트리를 만드는 중입니다  
상태 정보를 읽는 중입니다... 완료  
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 zlib1g-dev  
제안하는 패키지:  
  bison-doc ncurses-doc  
다음 새 패키지를 설치할 것입니다:  
  bison flex libbison-dev libelf-dev libfl-dev libncurses5-dev libsigsegv2  
  libssl-dev libssl-doc libtinfo-dev m4 zlib1g-dev  
다음 패키지를 업그레이드 할 것입니다:  
  libelf1 libssl1.0.0 zlib1g  
3개 업그레이드, 12개 새로 설치, 0개 제거 및 494개 업그레이드 안 함.  
4,006 kB바이트 / 5,183 kB바이트 아카이브를 받아야 합니다.  
이 작업 후 15.5 MB바이트의 디스크 공간을 더 사용하게 됩니다.  
계속 하시겠습니까? [Y/n]
```

6. Kernel 환경 설정

```
os2018741035@ubuntu:~$ cd 다운로드
os2018741035@ubuntu:~/다운로드$ cd linux-4.19.67
os2018741035@ubuntu:~/다운로드/linux-4.19.67$ sudo make menuconfig
sudo: unable to resolve host ubuntu
[sudo] password for os2018741035: 
```

```
.config - Linux/x86 4.19.67os2018741035 Kernel Configuration
> Enable loadable module support
  Enable loadable module support
    Arrow keys navigate the menu. <Enter> selects submenus --> (or empty submenus ----). Highlighted letters are hotkeys. Pressing <V> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <P> 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
      L(+)
      Select < Exit > < Help > < Save > < Load >
```

```
.config - Linux/x86 4.19.67os2018741035 Kernel Configuration
> Device Drivers
  Device Drivers
    Arrow keys navigate the menu. <Enter> selects submenus --> (or empty submenus ----). Highlighted letters are hotkeys. Pressing <V> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <P> for Help, </> for Search. Legend: [*] built-in [ ]
  [*] Device Drivers
    [*] Parallel port LCD/Keypad Panel support
    [*] Userspace I/O drivers
    [*] VFIO Non-Privileged userspace driver framework
    [*] Virtualization drivers
    [*] Virtio drivers (NEW)
    [*] Microsoft Hyper-V guest support
    [*] Xen driver support
    [*] Staging drivers
    [*] i86 Platform Specific Device Drivers
    [*] Platform support for Goldfish virtual devices
    L(+)
    Select < Exit > < Help > < Save > < Load >
```

```
.config - Linux/x86 4.19.67os2018741035 Kernel Configuration
> scripts/kconfig/Kconfig
# using defaults found in /boot/config-4.15.0-29-generic
# /boot/config-4.15.0-29-generic:899:warning: symbol value 'm' invalid for HOTPLUG_PCI_SPC
# /boot/config-4.15.0-29-generic:c1144:warning: symbol value 'm' invalid for NF_NAT_REDIRECT
# /boot/config-4.15.0-29-generic:c1147:warning: symbol value 'm' invalid for NF_TABLES_INET
# /boot/config-4.15.0-29-generic:c1148:warning: symbol value 'm' invalid for NF_TABLES_NETDE
# /boot/config-4.15.0-29-generic:c1331:warning: symbol value 'm' invalid for NF_TABLES_IPV4
# /boot/config-4.15.0-29-generic:c1336:warning: symbol value 'm' invalid for NF_TABLES_ARP
# /boot/config-4.15.0-29-generic:c1343:warning: symbol value 'm' invalid for NF_NAT_MASQUERA
# DE_IPv4
# /boot/config-4.15.0-29-generic:c1378:warning: symbol value 'm' invalid for NF_TABLES_IPV6
# /boot/config-4.15.0-29-generic:c1388:warning: symbol value 'm' invalid for NF_NAT_MASQUERA
# DE_IPv6
# /boot/config-4.15.0-29-generic:c1416:warning: symbol value 'm' invalid for NF_TABLES_BRIDGE
# /boot/config-4.15.0-29-generic:c3992:warning: symbol value 'm' invalid for HW_RANDOM_TPM
# /boot/config-4.15.0-29-generic:c4941:warning: symbol value 'm' invalid for LIRC
# /boot/config-4.15.0-29-generic:c10167:warning: symbol value 'm' invalid for SND_SOC_INTEL_ST_TPLATEV
# /boot/config-4.15.0-29-generic:c6172:warning: symbol value 'm' invalid for SND_SOC_INTEL_M_ACH
# /boot/config-4.15.0-29-generic:c7725:warning: symbol value 'm' invalid for DELL_SMBIOS_WMI
# /boot/config-4.15.0-29-generic:c7726:warning: symbol value 'm' invalid for DELL_SMBIOS_SMM

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

Kernel을 Compile 하기 전에 어떻게 Compile할 것인지 환경 설정해줬다.

7. Kernel Compile

```
os2018741035@ubuntu:~/다운로드/linux-4.19.67$ make -j6
```

가상 머신에 CPU core 수를 4개로 할당해뒀기 때문에 컴파일을 나누어 수행할 thread 수를 6으로 설정해주었다.

8. Module Install

```
os2018741035@ubuntu:~/Downloads/linux-4.19.67$ sudo make modules_install
```

Kernel의 Module을 설치해주었다.

9. Compile된 Kernel을 Boot Loader에 등록

```
os2018741035@ubuntu:~/Downloads/linux-4.19.67$ sudo make install
```

Kernel Image를 /boot로 복사하고 System Map을 /boot로 복사해주었다.

10. Compile한 Kernel로 재부팅

```
os2018741035@ubuntu:~$ uname -r  
4.19.67-2018741035  
os2018741035@ubuntu:~$
```

Kernel 4.19.67로 재부팅한 후 ‘uname -r’ 명령어를 이용해 버전을 확인하였고 그 결과가 ‘4.19.67-학번’으로 나온 것을 확인하였다.

- 고찰

강의자료를 참고하여 하나씩 따라한 결과 어렵지 않게 원하는 Kernel을 Compile 할 수 있었다. 하지만 과제1-1에서 우분투를 설치할 때 가상 머신을 ‘VirtualBox’를 사용했고 우분투 용량을 직접 Partition 했던 과정에서 ‘/home’와 ‘/boot’의 용량을 작게하여 설치해서 마지막에 Kernel을 Compile하는 과정에서 용량이 부족해서 Compile이 중단되었고 Kernel Panic으로 설치한 Kernel로 재 부팅할 수 없었다. 따라서 ‘/home’, ‘/boot’의 용량을 늘려서 우분투를 재설치했고 이후에 Kernel을 Compile할 수 있었다.

- Reference

- 준호씨의 블로그[ubuntu OS 업그레이드. 18.04.1 LTS. 커널패닉시 /boot 영역 공간이 충분한지 확인.], <https://junho85.pe.kr/1057>

Assignment 1-3

Linux apg…이 실행되는 지점에서 Linux Kernel Message 출력

- Introduction

Linux Vim Editor에서 변수, 함수, 매크로, 구조체 등을 검색하기 위해 cscope, ctags를 설치하고 이것들을 이용해서 검색한 함수가 선언된 부분을 찾고 이동한다. 이동한 부분에서 커널에서 메시지를 출력하는 함수인 printk()를 이용해 원하는 커널 메시지를 출력한다. 이때, printk()로 출력된 메시지는 dmesg 명령어를 통해 확인할 수 있다.

- Conclusion & Analysis

```
os2018741035@ubuntu:~/Downloads/linux-4.19.67$ sudo apt install exuberant-ctags
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  exuberant-ctags
0 upgraded, 1 newly installed, 0 to remove and 494 not upgraded.
Need to get 126 kB of archives.
After this operation, 341 kB of additional disk space will be used.
Get:1 http://kr.archive.ubuntu.com/ubuntu xenial/main amd64 exuberant-ctags amd64 4:1:5.9-svn20110310-11 [126 kB]
Fetched 126 kB in 2s (60.2 kB/s)
Selecting previously unselected package exuberant-ctags.
(Reading database ... 182808 files and directories currently installed.)
Preparing to unpack .../exuberant-ctags_1%3a5.9~svn20110310-11_amd64.deb ...
Unpacking exuberant-ctags (1:5.9-svn20110310-11) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up exuberant-ctags (1:5.9-svn20110310-11) ...
update-alternatives: using /usr/bin/ctags-exuberant to provide /usr/bin/ctags (c
tags) in auto mode
update-alternatives: using /usr/bin/ctags-exuberant to provide /usr/bin/etags (e
tags) in auto mode
os2018741035@ubuntu:~/Downloads/linux-4.19.67$ sudo apt install cscope
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  cscope-el
The following NEW packages will be installed:
  cscope
0 upgraded, 1 newly installed, 0 to remove and 494 not upgraded.
Need to get 207 kB of archives.
After this operation, 1,239 kB of additional disk space will be used.
Get:1 http://kr.archive.ubuntu.com/ubuntu xenial/universe amd64 cscope amd64 15.
8b-1build1 [207 kB]
Fetched 207 kB in 3s (58.2 kB/s)
Selecting previously unselected package cscope.
(Reading database ... 182814 files and directories currently installed.)
Preparing to unpack .../cscope_15.8b-1build1_amd64.deb ...
Unpacking cscope (15.8b-1build1) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up cscope (15.8b-1build1) ...
```

ctags 와 cscope를 설치했다.

```
os2018741035@ubuntu:~/Downloads/linux-4.19.67$ sudo ctags -R
[sudo] password for os2018741035:

os2018741035@ubuntu:~/Downloads/linux-4.19.67$ ls
arch      drivers   kernel      Module.symvers  tags
block     firmware lib        net            tools
built-in.a fs       LICENSES  README          usr
certs     include  MAINTAINERS samples        virt
COPYING   init    Makefile   scripts        vmlinux
CREDITS   ipc     mm        security      vmlinux-gdb.py
crypto    Kbuild  modules.builtin sound        vmlinux.o
Documentation Kconfig modules.order  System.map
```

Source code의 tag(global variable, function, macro···)들의 정보를 담고 있는 database를 생성하는 명령어로, 생성된 tag DB는 vim, emac 같은 editor에서 symbol 검색에 이용한다.

```
os2018741035@ubuntu:~$ cscope -R
```

ctags보다 다양한 검색 태입을 지원하는 cscope를 실행하는 명령어로 cscope는 실행 시 database를 구축한다.

C symbol: start_kernel		
File	Function	Line
0 process.c	<global>	16 extern void start_kernel(void);
1 start_kernel.h	<global>	11 extern asmlinkage void __init start_kernel(void);
2 bootp.c	start_kernel	135 start_kernel(void)
3 bootpz.c	start_kernel	263 start_kernel(void)
4 main.c	start_kernel	152 void start_kernel(void)
5 relocate.c	relocate_kernel	305 void *kernel_entry = start_kernel;
6 relocate.c	relocate_kernel	399 kernel_entry = RELOCATED(start_kernel);
7 setup.c	start_parisc	425 start_kernel();
8 setup_32.c	sparc32_start_kernel	294 start_kernel();
9 setup_64.c	start_early_boot	386 start_kernel();
a process.c	start_kernel_proc	28 start_kernel();
b head32.c	i386_start_kernel	56 start_kernel();
c head64.c	x86_64_start_reservations	472 start_kernel();
d main.c	start_kernel	531 asmlinkage __visible void __init start_kernel(void)

start_kernel()에서 'Linux apg···'이 실해되는 지점을 찾기위해 먼저 cscope를 실행 후 'start_kernel'을 검색했다.

```

 * page_ext requires contiguous pages,
 * bigger than MAX_ORDER unless SPARSEMEM.
 */
page_ext_init_flatmem();
mem_init();
kmem_cache_init();
pgttable_init();
vmalloc_init();
ioremap_huge_init();
/* Should be run before the first non-init thread is created */
init_espfix_bsp();
/* Should be run after espfix64 is set up. */
pti_init();
}

asm linkage __visible void __init start_kernel(void)
{
    char *command_line;
    char *after_dashes;

    set_task_stack_end_magic(&init_task);
    smp_setup_processor_id();
    debug_objects_early_init();

    cgroup_init_early();

    local_irq_disable();
    early_boot_irqs_disabled = true;

    printk(KERN_INFO "2018741035 in start_kernel()\n");

    /*
"Downloads/linux-4.19.67/init/main.c" 1180L, 29573C

```

실습하면서 기술한 start_kernel()이 있는 'main.c'로 이동했다.

```

asm linkage __visible void __init start_kernel(void)
{
    char *command_line;
    char *after_dashes;

    set_task_stack_end_magic(&init_task);
    smp_setup_processor_id();
    debug_objects_early_init();

    cgroup_init_early();

    local_irq_disable();
    early_boot_irqs_disabled = true;

    printk(KERN_INFO "2018741035 in start_kernel()\n");

    /*
:cscope add cscope.out
531,1      44%
asm linkage __visible void __init start_kernel(void)
{
    char *command_line;
    char *after_dashes;

    set_task_stack_end_magic(&init_task);
    smp_setup_processor_id();
    debug_objects_early_init();

    cgroup_init_early();

    local_irq_disable();
    early_boot_irqs_disabled = true;

    printk(KERN_INFO "2018741035 in start_kernel()\n");

    /*
:cs find t Linux agp

```

vim 명령모드 내에서 cscope 명령을 수행하기 위해 'cscope add cscope.out' 명령어를 입력했다. 추가적으로 'Linux agp...'이 실행되는 지점을 찾기위해 'cs find t Linux agp'를 입력했다. 이때, 질의 종류(검색 유형)는 원하는 문자열을 찾기위해 '4 or t'를 사용했다.

```

list_del(&bridge->list);
if (!list_empty(&agp_bridges))
    agp_frontend_cleanup();
module_put(bridge->driver->owner);
}
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 "Linux agpgart interface v%d.%d\n",
               AGPGART_VERSION_MAJOR, AGPGART_VERSION_MINOR);*/
        printk(KERN_INFO "os2018741035_Linux agpgart interface v0.103");
        printk(KERN_INFO "os2018741035_arg in __init agp_init(void)");
    return 0;
}

static void __exit agp_exit(void)
{
}

#ifndef MODULE
static __init int agp_setup(char *s)
{
    if (!strcmp(s, "off"))
        agp_off = 1;
    if (!strcmp(s, "try_unsupported"))

```

342,10-17 95%

'Linux agp...'이 실행되는 지점에서 printk() 함수를 이용해 Linux Kernel Message 양식에 맞춰 커널 메시지를 수정했다. 이때, 로그레벨은 'Informational'을 의미하는 'KERN_INFO'를 사용했다.

```
os2018741035@ubuntu:~/Downloads/linux-4.19.67$ sudo make
```

```
os2018741035@ubuntu:~/Downloads/linux-4.19.67$ sudo make modules_install
```

```
os2018741035@ubuntu:~/Downloads/linux-4.19.67$ sudo make install
```

module compile 후 reboot했다.

```
os2018741035@ubuntu:~$ dmesg | grep "os2018741035" -n
322:[ 6.939447] os2018741035_Linux agpgart interface v0.103
323:[ 6.939448] os2018741035_arg in __init agp_init(void)
os2018741035@ubuntu:~$
```

dmesg, grep 명령어를 이용해 Linux Kernel Message를 확인했다. 이때, 특정 문자가 포함된 열이 몇 번째 열인지 알기위해 명령어 뒤쪽에 '-n'을 붙였다.

```
C symbol: agp_init
  File      Function Line
0 backend.c <global> 366 module_init(agp_init);
1 backend.c agp_init 335 static int __init agp_init(void )
```

추가적으로, 수정한 소스코드의 path를 알기위해 cscope에서 'agp_init'을 검색하였고 'backend.c'라는 수정된 커널 코드 파일이 결과로 나온 것을 확인했다.

```
static int __init agp_init(void)
{
    if (!agp_off)
        /*printk(KERN_INFO "Linux agpgart interface v%d.%d\n",
               AGPGART_VERSION_MAJOR, AGPGART_VERSION_MINOR);*/
        printk(KERN_INFO "os2018741035_Linux agpgart interface v0.103");
        printk(KERN_INFO "os2018741035_agp in __init agp_init(void)");
    return 0;
}

static void __exit agp_exit(void)
{
<nloads/linux-4.19.67/drivers/char/agp/backend.c" 368L, 9218C
```

수정된 커널 소스코드의 path가 'Downloads/linux-4.19.67/drivers/char/agp/backend.c'라는 것을 확인하였다.

- 고찰

처음 과제를 시작할 때, 'Linux agp…'가 시작하는 지점이라는 말을 잘 이해 못하고 'agp'에 대해 검색해보았지만 만족스러운 정보가 나오지 않아서 어려움을 느꼈지만 여러 시도 끝에 'cs find' 명령어를 쓸 때 검색 유형을 '4 or t'로 바꿔서 검색해 본 결과 'Linux agp…'로 시작하는 부분을 찾을 수 있었다.

실습과 과제를 해보면서 ctags와 cscope를 둘 다 사용해보았는데 ctags보다 cscope를 사용하는 방법이 더 쉽고 직관적이고 편한것 같다는 생각이 들었다. 하지만 ctags는 vim 편집기를 사용할 때 초기에 해당 디렉터리에 'tag DB'만 만들어뒀다면 항상 연동되어 있기 때문에 필요할 때 바로바로 ctags의 검색 명령어만을 이용해서 원하는 검색이 가능하다는 장점이 있었다.

- Reference

[Linux] 소스 분석 툴 cscope 사용법, <https://harryp.tistory.com/131>