



# 경기과학고등학교 연구용 서버 사용 및 관리 설명서

version 1.03

리눅스 사용자 협회

관리자 gs19007

모둠장 gs19032

경기과학고등학교  
정보과학과  
Gyeonggi Science High School

2021. 9. 9.

이 설명서는 비영리적 목적에 한해 제한 없이 배포할 수 있습니다.



과학영재학교 경기과학고등학교  
Gyeonggi Science High School for the Gifted

Copyright ©2021 Gyeonggi Science High School

[HTTPS:/ /WWW.GS.HS.KR/](https://www.gs.hs.kr/)

*Compiled at September 15, 2021*



## 과학영재학교 경기과학고등학교

Gyeonggi Science High School for the Gifted

### 서버 사용 규칙

#### (a) 사용 목적

경기과학고등학교의 연구용 서버는 7층 서버실에 있다. 이 서버는 정보과학 연구 또는 이외 학교의 공적인 목적으로만 이용할 수 있으며 암호화폐 채굴, 게임 연산 보조 등 사적인 목적으로 이용할 수 없다.

#### (b) 사용 신청

연구용 서버의 GPU를 사용하지 않는 경우 별도의 사용 신청이 필요하지 않다. 서버의 GPU를 사용하고자 하는 학생은 반드시 송죽학사를 통해 미리 신청해야 한다. 신청하고 승인받은 이후 사용하고자 할 때는 반드시 정해진 GPU를 정해진 기한 동안만 사용해야 한다.

#### (c) 관리

연구용 서버는 GM 봉사모둠이 관리한다. 또한, GM 봉사모둠원만이 관리자 계정을 가질 수 있으며 root 계정의 비밀번호는 각 기수의 기장이 관리한다.

#### (d) 현재 서버 정보

<b>OS</b>	Ubuntu 20.04.2 LTS
<b>CPU</b>	Intel(R) Xeon(R) CPU E5-2680 v2 @ 2.80GHz × 40
<b>GPU</b>	Nvidia Tesla K40c × 6
<b>IP</b>	115.23.235.135
<b>SSH Port</b>	—
<b>Location</b>	SRC 720
<b>Admin</b>	19007 권준우
<b>Last modified</b>	2021.5.11.

*To The Researchers of Gyeonggi Science High School*

*Calculation is the best way to prove.*

## Acknowledgements

서버의 효율적인 이용을 위해 아래와 같은 규칙들이 있다. 반드시 숙지하고 준수하기 바란다.

- 서버는 연구 및 공적 목적으로만 사용 가능하다.
- 서버는 함께 사용한다.
- 계정명은 반드시 gs00000 (학번)으로 한다.
- 계정 생성시 자신의 홈 디렉토리에 whoami.txt를 만들어 자신의 이름, 연락처, 이메일 주소를 기록해야 한다.
- GPU는 송죽학사를 통해 신청한 자만이 사용할 수 있다.
- GPU 1개는 최장 12일 동안, 2개는 최장 6일 동안, ..., 6개는 최장 2일 동안 사용 가능하다.
- GPU를 신청하지 않은 상태에서 사용할 시 프로세스가 강제 종료될 수 있다.
- 과도한 CPU/GPU/RAM 사용으로 서버에 과부하를 유발하는 프로세스는 강제로 종료될 수 있으며 사용자는 경고를 받을 수 있다. 경고 3회 누적시 GPU 신청이 제한될 수 있다.
- 관리자 권한이 없는 계정은 졸업 시 삭제를 원칙으로 한다.
- 관리자 계정은 GM 봉사모둠원만이 소유한다.
- 허가 없이 reboot 명령어를 사용해서는 안 된다.
- 모든 사용자는 자신의 가상환경을 구성해 사용해야 한다.

아래는 유용한 링크들이다.

- 경기과학고등학교 연구용 서버 사용 및 관리 설명서 (Overleaf 주소)
- 경기과학고등학교 리눅스 사용자 협회 (Github 주소)
- 경기과학고등학교 리눅스 사용자 협회 오픈채팅방 (카카오톡 주소)
- 현재 연구용 서버 IP (서버 주소)

## Preface

4차 산업혁명이 시작되고, 수많은 데이터가 정보화되어 처리되고 있다. 또, 인공지능과 같은 첨단 프로그램들은 많은 양의 데이터를 빠르게 처리해야 하기에 고성능 컴퓨터를 필요로 한다. 본 설명서는 리눅스 서버를 처음 접하는 정보과학 연구자들을 위해 작성되었다. 특히, 리눅스 서버 이용을 위해 필수적인 ssh, sftp 등의 사용 방법, 공용 서버 사용 규칙, 인공지능 연구를 위해 필수적인 Python, CUDA, TensorFlow, Torch, venv 등의 사용 방법을 다뤘다. 또, 함께 쓰는 공동 연구 환경에서 어떻게 자신의 연구를 해나가야 하는지 방향을 제시했다.

정보과학 연구의 세계는 깊고도 넓다. 그렇기에, 연구의 의미와 가치를 잘 이해하고 자신에게 맞는 연구를 하는 것이 중요하다. 연구 방법을 익히고, 다양한 사람들의 연구를 경험해봐야 한다. 저자는 본 설명서를 통해, 서버를 이용한 정보과학 연구에 꼭 필요한 것들을 가려 뽑아 연구의 모든 것을 간명하게 전달하려고 노력했다. 본 설명서가 귀하의 미래를 밝히는 길잡이가 되기를 기원한다.

37기 정보과 일동

2021년 5월 17일, 송죽골에서.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	Ubuntu . . . . .	1
<b>2</b>	<b>Connection</b>	<b>2</b>
2.1	SSH를 이용한 서버 접속 . . . . .	2
2.2	SFTP를 이용한 파일 관리 . . . . .	4
<b>3</b>	<b>Command</b>	<b>5</b>
3.1	파일 관리 . . . . .	5
3.1.1	절대경로와 상대경로 . . . . .	5
3.1.2	디렉토리 및 탐색 . . . . .	5
3.1.3	디렉토리 및 파일 관리 . . . . .	6
3.2	기타 기능 . . . . .	6
3.2.1	프로세스 관리 . . . . .	6
3.2.2	권한 관리 . . . . .	6
3.2.3	Vim . . . . .	7
3.2.4	Pipeline . . . . .	7
<b>4</b>	<b>Environments</b>	<b>8</b>
4.1	주의사항 . . . . .	8
4.2	간편 설치 . . . . .	8
4.3	가상환경 설치 및 이용 . . . . .	8
4.4	Jupyter Notebook 설치 . . . . .	9
4.5	tensorflow 설치(선택) . . . . .	11

4.6 CUDA 설치 . . . . .	11
4.7 셸 바꾸기(선택) . . . . .	12
<b>5 Other</b>	<b>13</b>
5.1 NewKOI . . . . .	13
5.2 CMS . . . . .	13
5.3 DB . . . . .	13
5.4 GCC . . . . .	13
5.5 Robocode . . . . .	13
<b>6 Admin</b>	<b>14</b>
6.1 공지 . . . . .	14
6.1.1 유저 관리 . . . . .	14
6.2 초기화 . . . . .	14
6.2.1 설치디스크 준비 . . . . .	14
6.2.2 Ubuntu 설치 . . . . .	15
6.3 Drivers . . . . .	15
6.4 CUDA 및 cuDNN . . . . .	16
6.4.1 CUDA 설치 . . . . .	16
6.4.2 환경변수 설정 . . . . .	17
6.4.3 cuDNN 설치 . . . . .	18
6.5 Storage . . . . .	19
<b>Appendix A Troubleshooting</b>	<b>20</b>
A.1 Graphic Driver . . . . .	20
A.1.1 nvidia-smi . . . . .	20
A.2 Storage . . . . .	20
A.3 Python Environments . . . . .	20
A.4 Jupyter . . . . .	20
A.5 SSH . . . . .	20
A.6 Database . . . . .	21
A.7 Nginx . . . . .	21
<b>Appendix B GSHS CSLAB</b>	<b>22</b>
<b>Appendix C Contributors</b>	<b>23</b>

# List of Figures

2.1	앱 및 기능 . . . . .	2
2.2	OpenSSH 서버 설치 . . . . .	3
2.3	SSH 접속창 . . . . .	3
2.4	FileZilla 접속 . . . . .	4
2.5	파일질라 접속 . . . . .	4
4.1	활성화된 venv 환경 . . . . .	9
4.2	Jupyter Notebook 버전 확인 . . . . .	9
4.3	Jupyter Notebook config 파일 생성 . . . . .	10
4.4	Jupyter Notebook 실행 모습 . . . . .	10
4.5	Jupyter Notebook 화면 . . . . .	11
4.6	bash로 변경 . . . . .	12
6.1	nvidia-smi . . . . .	15
6.2	CUDA 설치 . . . . .	16
6.3	CUDA 설치 . . . . .	17
6.4	CUDA 설치 - Drivers . . . . .	17
6.5	cuDNN 설치 . . . . .	18

---

## List of Tables

## List of Abbreviations

**GSHS** Gyeonggi Science High School

**GPU** Graphic Processing Unit

**CPU** Central Processing Unit

**GUI** Graphic User Interface

**CLI** Command Line Interface

**CUDA** Compute Unified Device Architecture

# Introduction

서버는 모두가 함께 사용하는 것입니다.

## 1.1 | Motivation

경기과학고등학교(GSHS)에는 고성능 GPU 6개, 그리고 40개의 CPU 코어를 가진 서버가 있다. 이 서버는 강력하기에 학생들은 자신의 연구에 큰 도움을 받고 있다. 그러나, 그 서버의 무분별한 사용과 관리자 부재로 인해 연구활동에 어려움을 겪게 되었으며 신입생들의 경우 구전으로 전해지는 사용방법에 의존해야 했다. 이에 문제의식을 느낀 37기 정보과 일동은 경기과학고등학교 리눅스 사용자 협회, 그리고 교내 봉사모임인 GM을 만들어 서버 사용 및 관리를 체계화하는 한편 본 설명서를 통해 처음 서버를 이용하는 학생들을 돋고자 한다.

## 1.2 | Ubuntu

Ubuntu Linux는 윈도우와 같은 운영체제로, 윈도우와 달리 무료이며 공개 소스이다. 서버에 일반적으로 이용하는 Ubuntu Server는 GUI가 없는 Ubuntu로, 마우스 없이 명령줄만으로 제어하는 CLI 환경이다. 윈도우의 명령 프롬프트(cmd)와 유사하다.

# Connection

이 장은 서버와의 연결, 그리고 파일 송수신 방법을 다룬다.

## 2.1 | SSH를 이용한 서버 접속

서버는 학교에 있으므로 원격으로만 이용할 수 있다. 이를 위하여 SSH(Secure SHell)를 쓴다. Putty와 같은 전용 프로그램을 이용해도 되고, OpenSSH를 설치한 후 명령 프롬프트와 같은 환경에서 SSH 명령어를 이용해도 된다. 이 문서에서는 후자를, Windows 10 기준으로 설명하도록 하겠다.

먼저 윈도우에서 시작 > 설정 > 앱 > 앱 및 기능에 들어간다.

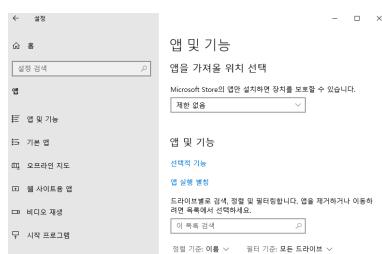


Figure 2.1: 앱 및 기능

여기서 선택적 기능을 클릭하고 SSH를 검색해 OpenSSH를 찾아 설치한다.



Figure 2.2: OpenSSH 서버 설치

경기과학고등학교 연구용 서버는 Ubuntu 18.04로 구성되어 있다. 이 운영체제는 Windows처럼 사용자 이름(username)과 암호(password)가 있어야 한다. 만약 본인이 서버를 이용하고 싶다면 근처의 정보전공 친구에게 사이다를 사주고 계정을 얻어내도록 하자. 일련의 과정을 통해 서버에 접속할 계정명과 암호, 주소와 포트까지 알게 되었다면 다음과 같은 명령어를 통해 서버에 접속할 수 있다. 이후 암호를 입력하면 접속에 성공한 것이다. 참고로 root 계정은 보통 이용하지 않는다.

```
$ ssh -p [port] [username]@[address]
ex) ssh -p 22 root@127.0.0.1
```

로그인하면 다음과 같은 화면을 볼 수 있다. 참고로 서버는 TUI 환경이기에 접속된 창으로 모든 작업을 수행하게 된다.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

세로운 코모스 플랫폼 PowerShell 사용 https://aka.ms/powershell
PS C:\SLAB\root> whoami
root
root@: ~ password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 5.8.0-37-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

 * Canonical Livepatch is available for installation.
   To reduce system reboot and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

204 packages can be updated.
100 updates are security updates.

New release '20.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2023.

GYEONGGI SCIENCE HIGH SCHOOL
COMPUTER SCIENCE LAB

=====
*** Use Anaconda for Python programming. ***
*** Anaconda is an open-source distribution of Python ***
*** and R for scientific computing. ***
*** THIS SERVER IS CONFIGURED WITH ANACONDA. ***
*** The Default version is ***
*** tensorflow=2.1.0, pytorch=1.3.0, CUDA 10.2 ***
*** cuDNN = 7.6.5.14, cuBLAS = 10.2.1.3.1 ***
*** compute compat. is 3.5.5. build=force-source ***
*** Use conda environment if other versions are needed. ***
*** Please, do not update driver or cuda ***
*** without teacher's permission. ***
=====

Currently maintained by: gs19087
Last login: Sat Jan 16 13:35:35 2021 from 127.0.0.1
root@root:~#
```

Figure 2.3: SSH 접속창

## 2.2 | SFTP를 이용한 파일 관리

SFTP는 SSH File Transfer Protocol의 약어다. 즉, 이미 다른 SSH 통신을 이용해 파일을 전송하는 것이다. 대표적인 프로그램으로 FileZilla가 있다. 파일질라를 설치한 후 좌측 상단 사이트 관리자에서 새 로그인 정보를 만들어 저장하고 연결하면 된다.

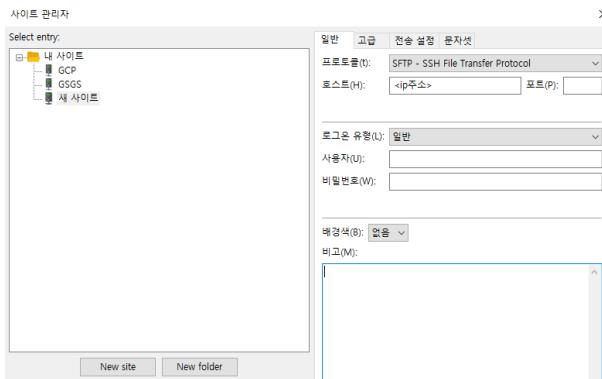


Figure 2.4: FileZilla 접속

접속에 성공하면 화면 양쪽에 파일 탐색기가 생길 것이다. 좌측은 현재 사용자의 컴퓨터이며 우측은 서버다. 이제 사용자는 평소 윈도우 파일 탐색기를 이용하듯 파일을 끌어 옮기거나 다운로드하여 편집할 수 있다. 다만 파일을 다운로드 후 편집하고 다시 업로드를 해야 서버의 파일이 수정된다는 점에 유의해야 한다.

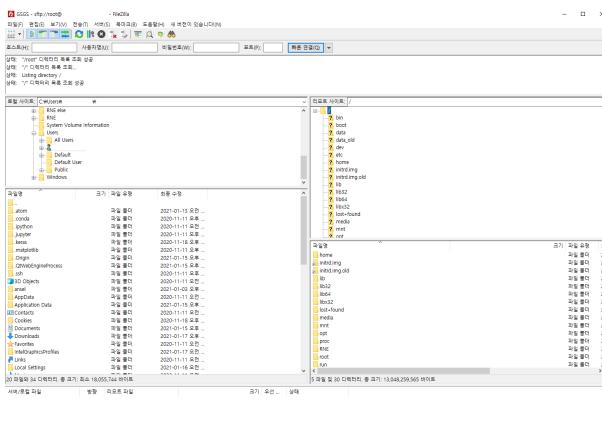


Figure 2.5: 파일질라 접속

# Command

이 장은 기본 리눅스 명령어를 다룬다.

## 3.1 | 파일 관리

### 3.1.1 | 절대경로와 상대경로

절대경로는 /home/gs19000/dir1/dir2와 같은 것이다. 상대경로는 현재 내 위치에서 시작하는 경로다. 즉, 내가 지금 /home/gs19000/dir1에 있다면 여기서 /home/gs19000/dir1/dir2의 상대 경로는 그냥 dir2다.

### 3.1.2 | 디렉토리 및 탐색

cd 명령어를 사용해서 디렉토리를 탐색할 수 있다.

..은 상위 디렉토리, -은 이전 디렉토리, ~는 홈 디렉토리를 의미한다. cd (디렉토리)라 치면 해당 디렉토리로 이동된다. 디렉토리를 표현할 때 절대경로, 상대경로 둘 다 가능하며, 실행시 주소쪽 값이 변경된다.

```
$ cd dir1  입력시  
~/dir1$ ..으로 바뀐다 .  
$ cd /home/gs19000/dir1  
$ cd ..  
$ cd -  
$ cd ~
```

### 3.1.3 | 디렉토리 및 파일 관리

`rm` 명령어를 사용해서 파일을 삭제할 수 있다.

`rm (file)` : 해당 파일을 삭제한다. `rm -r (dir)` : 해당 디렉토리를 삭제한다. 또한, `mkdir` 명령어를 사용해서 디렉터리를 만들 수 있다.

`mkdir (dir)` : 해당 이름의 디렉토리가 존재하지 않을 경우 새로 `dir`이름의 디렉터리를 생성한다.

다음으로 `cp`, `mv` 명령어를 통해 파일을 옮기거나 복사할 수 있다.

`cp (file1) (file2)` : `file1`의 이름을 `file2` 이름으로 바꾸어 복사한다. (`file1`은 삭제되지 않음 `file2`가 이미 존재할 경우 덮어씌워진다)

`cp -r (dir1) (dir2)` : `dir1`이름의 디렉토리를 `dir2` 디렉토리 내부에 복사한다. (`dir1`은 반드시 존재해야 하며 `dir2`는 존재하지 않을 경우 새로 만들어진다)

`mv (file1) (file2)` : `file1`의 이름을 `file2`로 바꾼다. (`file2`가 이미 존재할 경우 덮어씌워진다)

`mv (dir1) (dir2)` : `dir1`의 이름을 `dir2`로 바꾼다. (`dir1`은 반드시 존재해야 하며 `dir2`가 존재할 경우 `dir2` 내부에 `dir1` 파일을 이동시킨다.)

마지막으로, `cat`을 이용해 내용을 읽어올 수 있다. `cat (file1)` : `file1`의 내용을 불러온다.

```
$ rm file1
$ rm -r dir1
$ mkdir dir2
$ cp file1 file2
$ cp -r dir1 dir2
$ mv file1 file2
$ mv dir1 dir2
$ cat file3.txt
```

## 3.2 | 기타 기능

### 3.2.1 | 프로세스 관리

`htop`을 이용해 실행중인 프로세스를 확인할 수 있다. `kill`, `pkill`을 통해 프로세스를 제거할 수 있다.

```
$ htop
$ kill -9 <pid>
```

### 3.2.2 | 권한 관리

`chmod`을 이용해 권한을 관리할 수 있다. 권한에 대한 자세한 정보는 ...

```
$ sudo chmod 777 file1
```

### 3.2.3 | Vim

기본 메모장과 같다. i를 눌러 편집 모드에 들어갈 수 있으며 esc로 나올 수 있다. :w를 이용해 저장할 수 있고, :q를 이용해 종료할 수 있다.

### 3.2.4 | Pipeline

파이프라인을 통해 출력값에서 원하는 문자열을 확인할 수 있다. 보통 아래와 같이 |와 grep을 이용해 사용한다. 아래의 경우 file2.txt를 출력하는데, 그중 "abc"가 포함된 부분을 찾는 것이다.

```
$ cat file2.txt | grep abc
```

# Environments

이 장은 기본 연구 환경을 다룬다.

## 4.1 | 주의사항

- **sudo reboot**은 그 어떠한 경우에도 사용하면 안 된다. 새로운 프로그램을 설치하는 등 작업 도중 블로그 글을 따라쓰게 될 수 있는데, 그중에 reboot 명령어가 있으면 건너뛰자. 타인의 프로세스가 비정상적으로 종료될 위험이 크다.
- 모든 사용자는 반드시 가상환경을 구성하고, 그 가상환경 내에 여러가지 연구에 필요한 프로그램들을 설치해 사용해야 한다. 이렇게 하면 관리자 권한 없이 원하는 대로 프로그램의 버전이나 설정을 바꿀 수 있다.

## 4.2 | 간편 설치

관리자가 미리 만들어 둔 계정의 폴더를 복사한다.

```
$ cp -r /home/sampleUser/* ~/
```

이 방법을 사용하면 4.3와 4.4로 향하면 된다.

## 4.3 | 가상환경 설치 및 이용

만약 가상환경을 추가로 설치하려 하거나 4.2의 간편 설치를 이용하지 않았다면 다음 명령어로 가상환경을 설치하자.

```
$ virtualenv venv
$ virtualenv venv --python=python3.6
```

여기서 3.5 대신 원하는 다른 버전을 다운받아도 된다.

이후 가상환경을 사용할 때에는 다음 명령어를 사용하면 된다.

```
$ source ~/venv/bin/activate
```

Figure 4.1: 활성화된 venv 환경

## 4.4 | Jupyter Notebook 설치

가상환경을 켠 상태로 다음 명령어를 입력하자.

```
$ pip3 install jupyter
```

설치가 제대로 되었는지를 확인하기 위해 다음 명령어를 입력하여 Jupyter Notebook 버전을 확인하자.

```
$ jupyter --version
```

Component	Version
jupyter core	4.6.3
jupyter-notebook	6.2.0
qtconsole	4.7.7
ipython	7.9.0
ipykernel	5.5.5
jupyter client	6.1.12
jupyter lab	not installed
nbconvert	5.6.1
ipywidgets	7.6.3
nbformat	5.1.3
traitlets	4.3.3

Figure 4.2: Jupyter Notebook 버전 확인

다음 명령어를 입력하여 ~/jupyter 디렉토리에 접근하자.

```
$ cd ~/.jupyter
```

만약 해당 디렉토리가 존재하지 않는다면, 다음 명령어를 입력하여 해당 디렉토리를 생성한 후 해당 디렉토리에 접근하면 된다.

```
$ mkdir ~/.jupyter
```

이제 다음 명령어를 이용하여 config 파일을 생성하자.

```
$ jupyter notebook --generate-config
```

```
(venv) tester@GSGS:~/jupyter$ jupyter notebook --generate-config
Writing default config to: /home/tester/.jupyter/jupyter_notebook_config.py
```

Figure 4.3: Jupyter Notebook config 파일 생성

이제 ~/jupyter 디렉토리에 jupyter\_notebook\_config.py라는 파일이 생성되었을 것이다. 해당 파일의 내용을 지우고 다음 내용을 입력하자. 단, username에는 자신의 아이디를, student\_number에는 자신의 학번을 입력한다.

```
c = get_config()
c.NotebookApp.allow_origin='*'
c.NotebookApp.notebook_dir='/home/username'
c.NotebookApp.ip='115.23.235.135'
c.NotebookApp.port=student_number
c.NotebookApp.open_browser=False
```

이제 jupyter notebook을 실행시키기 위해 다음 명령어를 입력하자.

```
$ jupyter notebook --config ~/jupyter/jupyter_notebook_config.py
```

이제 터미널 창에 뜨는 url을 복사하여 접속하면 jupyter notebook을 사용할 수 있다.

```
(base) dobbby@GSGS:~$ jupyter notebook --config .jupyter/jupyter_notebook_config.py
[16:18:14.799 NotebookApp] JupyterLab extension loaded from /opt/anaconda3/lib/python3.7/site-packages/jupyterlab
[16:18:14.800 NotebookApp] JupyterLab application directory is /opt/anaconda3/share/jupyter/lab
[16:18:14.800 NotebookApp] Serving notebooks from local directory: /home/dobby
[16:18:14.800 NotebookApp] The Jupyter Notebook is running at:
[16:18:14.800 NotebookApp] http://115.23.235.135:9876/
[16:18:14.800 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
```

Figure 4.4: Jupyter Notebook 실행 모습



Figure 4.5: Jupyter Notebook 화면

터미널 창의 세션이 종료되어도 jupyter notebook이 계속 실행되도록 하려면 nohup 명령어를 이용하면 된다.

```
$ nohup jupyter notebook --config ~/.jupyter/jupyter_notebook_config.py
```

## 4.5 | tensorflow 설치(선택)

가상환경을 켜 상태로 다음 명령어를 입력하자.

```
$ pip3 install --upgrade tensorflow
```

## 4.6 | CUDA 설치

만약 서버에 없는 CUDA가 필요하다면 관리자에게 문의해야 한다. 이후 자신의 환경변수에 설치된 특정 버전의 CUDA를 추가하면 된다.

관리자는 다음 링크에 접속해 CUDA를 설치한다. 이 부분은 반드시 관리자 권한으로 수행해야 한다. 그래픽 드라이버나 CuDNN은 다시 설치하지 않는다.

[https://developer.nvidia.com/cuda-downloads?target\\_os=Linux](https://developer.nvidia.com/cuda-downloads?target_os=Linux)

## 4.7 | 쉘 바꾸기(선택)

쉘이 뭔지에 대한 장황한 설명을 쓰지는 않겠다. 쉘에는 sh, bash 등이 있는데, 처음 아이디를 만들면 쉘이 sh로 설정이 되어있을 수 있다. bash가 여러가지 좋은 기능(위 아래 방향키로 이전 명령 불러오기 등)이 있으므로 다음 명령어를 쳐서 bash로 쉘을 바꿔주자.

```
$ chsh -s /bin/bash
```

이후 서버와의 연결을 한 번 끊었다가 재접속했을 때 아래처럼 \$ 옆에 사용자명이 뜨면 성공한 것이다.

```
defaultsettings@GSGS:~$
```

Figure 4.6: bash로 변경

현재 서버에서 bash 쉘을 사용하는 상태에서 GPU를 사용하는 프로그램을 실행시키는 경우, GPU에 메모리만 할당되고 GPU를 사용하지 않는 현상이 종종 나타나고 있으니 만약 bash 쉘을 사용하는데 GPU 사용량이 이상할 만큼 저조하다면, 이를 한 번 의심해보자. 해당 프로그램을 실행시킬 때에만 bash를 사용하지 않으면 이러한 문제를 해결할 수 있다.

## Other

이 장은 서버의 다른 이용 방법들을 다룬다.

5.1 | NewKOI

5.2 | CMS

5.3 | DB

5.4 | GCC

5.5 | Robocode

# Admin

이 장은 관리자 작업과 관련된 것들을 다룬다.

## 6.1 | 공지

서버에는 여러 학생들의 연구 및 논문 데이터가 있다. 그러므로 서버에서의 중대한 작업이 진행될 경우 사전에 공지하여 백업하도록 하는 것이 바람직하다. GM 오픈채팅방 등을 이용하면 된다.

### 6.1.1 | 유저 관리

adduser를 이용해 유저를 생성할 수 있다. usermod를 이용해 유저의 그룹을 관리할 수 있다. su를 이용해 사용자를 바꿀 수 있다.

```
$ adduser gs21000  
$ usermod -aG GROUP gs21000  
$ su gs20000
```

## 6.2 | 초기화

서버에 중대한 문제가 발생하여 해결이 불가하거나 심히 어려운 경우 초기화가 그 해결책이 될 수 있다. 본 문서는 Ubuntu Server 20.04.2 LTS를 기반으로 작성되었다.

### 6.2.1 | 설치디스크 준비

1. 2GB 이상의 USB 준비
2. 우분투 이미지 사이트(링크)에서 Ubuntu Server 20.04 준비

3. Rufus(링크) 준비
4. Rufus 실행 후 준비한 iso 선택
5. 준비한 USB를 선택하고 '시작' 클릭

## 6.2.2 | Ubuntu 설치

연구용 서버는 SRC 7층 서버실에 위치해 있다. 먼저 7층 교무실에 계시는 정덕교 선생님께 서버 초기화를 위해 왔음을 알리자.

연구용 서버는 서버실에서 가장 가까운 서버랙의 제일 아래에 있다. 후면에 USB와 VGA 단자가 있고 안쪽에는 키보드, 마우스와 모니터가 있으므로 연결하여 작업하면 된다. 먼저 전원 버튼을 꾹 눌러 종료시킨 뒤 다시 눌러 시스템을 시작한다. 이후 바이오스 창이 나올 때 F2 또는 Del 키를 누르면 부팅 디스크를 선택할 수 있게 된다.

이후 설치 과정을 따라하면 된다. 이때 OpenSSH-Server를 설치하는 것을 추천한다. 또, 중간에 네트워크를 설정하는 창이 나오면 설정표(비공개 링크)를 보고 정덕교 선생님께 여쭤보며 작성하면 된다.

## 6.3 | Drivers

이 Section은 어떤 드라이버도 깔려 있지 않은 상태에서의 설치 상황을 다룬다.

먼저 아래 명령어를 통해 현재 설치된 GPU에 적합한 드라이버를 찾을 수 있다.

```
$ ubuntu-drivers devices
```

이후 그중 하나를 골라(recommended) 설치하면 된다.

```
$ sudo apt install nvidia-driver-XXX (숫자)
```

적용하려면 재부팅이 필요하다.

```
$ sudo reboot
```

Nvidia-smi를 통해 설치를 확인할 수 있다.

```
$ nvidia-smi
```

GPU Name	Persistence-MI	Bus-Id	Disp A	Volatile Uncorr. ECC		
Fan	Temp	Perf	Mem:Usage/Cap	GPU-Util	Compute M.	MIG M.
0 Tesla K40c	Off	0000:00:00:00:00:00	00:00.0 Off		0%	N/A
23%	26C	P8	19W / 235W	8GB / 12206MB		

Figure 6.1: nvidia-smi

여기서 CUDA Version은 권장 사항으로, 실제 CUDA 버전 및 설치 유무와는 상관 없다.

## 6.4 | CUDA 및 cuDNN

### 6.4.1 | CUDA 설치

기본적으로 알아야 할 사실은 CUDA가 라이브러리라는 것이다. 그렇기에 여러 CUDA 버전을 동시에 설치할 수 있다. 또, debian package로 설치하지만 않았다면 다음 명령어를 통해 쉽게 제거할 수 있음을 기억하자.

```
$ sudo rm -rf /usr/local/cuda*
```

CUDA 설치를 위해서는 gcc가 필요하다.

```
$ sudo apt-get install build-essential
```

이후 CUDA 설치를 위해 먼저 Nvidia Developer 사이트(링크)에 들어간다.

여기서 현재 서버의 상황에 맞는 버튼들을 누른다. 참고로 CUDA 12 이후 버전들은 점진적으로 Compute Compatability가 3.x대인 GPU들에 대한 지원을 제한하고 있는데, 연구용 서버에 장착된 GPU인 Tesla K40c의 Compute Compatability는 3.5다. 따라서 11.2 이하 버전의 설치를 권장한다.

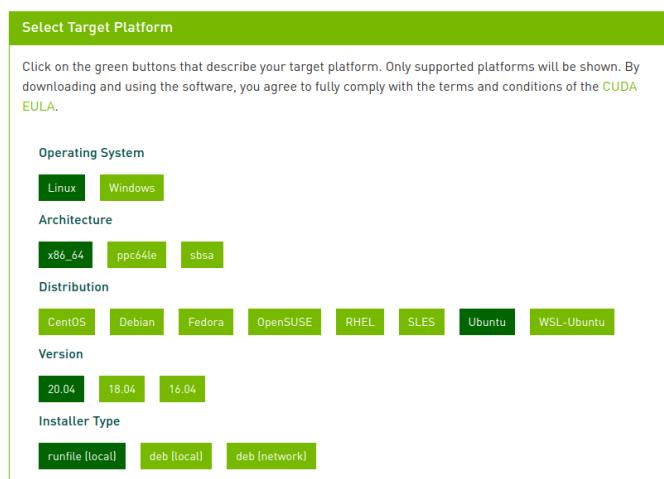


Figure 6.2: CUDA 설치

이후 지시사항에 따라 명령어를 입력하면 된다.

```
$ wget https://developer.download.nvidia.com/compute/cuda/11.2.0/local_installers/cuda_11.2.0_460.27.04_linux.run
$ sudo sh cuda_11.2.0_460.27.04_linux.run
```

Figure 6.3: CUDA 설치

중간에 설치 프롬프트가 뜨는데, 스페이스 키를 눌러 드라이버는 설치에서 제외한다. 이후 설치를 진행하면 된다.

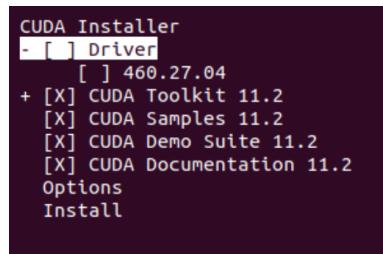


Figure 6.4: CUDA 설치 - Drivers

## 6.4.2 | 환경변수 설정

먼저 다음 명령어를 통해 CUDA를 환경변수에 등록한다. 11.2 자리에 다른 버전을 넣을 수도 있다.

```
$ sudo sh -c "echo 'export PATH=$PATH:/usr/local/cuda-11.2/bin' >> /etc/profile"

$ sudo sh -c "echo 'export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/cuda-11.2/lib64' >> /etc/profile"

$ sudo sh -c "echo 'export CUDADIR=/usr/local/cuda-11.2' >> /etc/profile"

$ source /etc/profile
```

이후 다음 명령어를 입력했을 때 CUDA 버전이 올바르게 출력된다면 설치 성공이다.

```
$ nvcc -V
```

### 6.4.3 | cuDNN 설치

Nvidia Developer 사이트(링크)에 들어간다. 로그인 또는 회원가입 후 약관 및 서약에 동의하면 링크가 생긴다. 여기서 Archived cuDNN Releases에 들어가 cuDNN v8.1.1 (February 26th, 2021), for CUDA 11.0, 11.1 and 11.2를 다운로드 받는다.

## cuDNN Download

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural net

I Agree To the Terms of the cuDNN Software License Agreement

Note: Please refer to the Installation Guide for release prerequisites, includir

For more information, refer to the cuDNN Developer Guide, Installation Guide

[Download cuDNN v8.2.2 \[July 6th, 2021\], for CUDA 11.4](#)

[Download cuDNN v8.2.2 \[July 6th, 2021\], for CUDA 10.2](#)

[Archived cuDNN Releases](#)

Figure 6.5: cuDNN 설치

이 파일은 서버에서는 다운받을 수 없다. 따라서 sftp 등의 방법으로 서버에 업로드해야 한다. 이후 다음 명령어를 차례로 입력한다.

```
$ tar xvzf cudnn-11.2-linux-x64-v8.1.1*.tgz  
  
$ sudo cp cuda/include/cudnn* /usr/local/cuda/include  
  
$ sudo cp cuda/lib64/libcudnn* /usr/local/cuda/lib64  
  
$ sudo chmod a+r /usr/local/cuda/include/cudnn.h /usr/local/cuda/lib64/libcudnn*  
  
$ sudo ln -sf /usr/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn_adv_train.so.8.1.1 /usr  
/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn_adv_train.so.8
```

```
$ sudo ln -sf /usr/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn_ops_infer.so.8.1.1 /usr  
/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn_ops_infer.so.8  
  
$ sudo ln -sf /usr/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn_cnn_train.so.8.1.1 /usr  
/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn_cnn_train.so.8  
  
$ sudo ln -sf /usr/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn_adv_infer.so.8.1.1 /usr  
/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn_adv_infer.so.8  
  
$ sudo ln -sf /usr/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn_ops_train.so.8.1.1 /usr  
/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn_ops_train.so.8  
  
$ sudo ln -sf /usr/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn_cnn_infer.so.8.1.1 /usr  
/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn_cnn_infer.so.8  
  
$ sudo ln -sf /usr/local/cuda-11.2/targets/x86_64-linux/lib/libcudnn.so.8.1.1 /usr/local/  
cuda-11.2/targets/x86_64-linux/lib/libcudnn.so.8  
  
$ sudo ldconfig
```

이후 다음 명령어를 입력했을 때 버전이 정확하게 출력되면 된다.

```
$ ldconfig -N -v $(sed 's/:/ /' <<< $LD_LIBRARY_PATH) 2>/dev/null | grep libcudnn
```

## 6.5 | Storage

# Troubleshooting

## A.1 | Graphic Driver

### A.1.1 | nvidia-smi

nvidia driver library mismatch

cuda

## A.2 | Storage

dev1

## A.3 | Python Environments

pip, pip3, venv

## A.4 | Jupyter

config

## A.5 | SSH

sshd\_config

## A.6 | Database

mongodb, mariadb

## A.7 | Nginx

nginx.conf

## GSHS CSLAB

### 정보과 연구용 기기 목록

- Nvidia RTX 2070 eGPU × 2
- Nvidia RTX 2080 Super eGPU × 1
- Nvidia RTC 3070 × 1
- Intel NUC × 2
- 본 서버.

# C

## Contributors

19007 권준우, 19011 김도영, 19032 노현서, 19042 박영민