# Lab. Session 1 - Env Setup

Computer Security Lab.

Name: Sangyun Kim

Email: sangyun.kim@snu.ac.kr

# Contents

# Contents

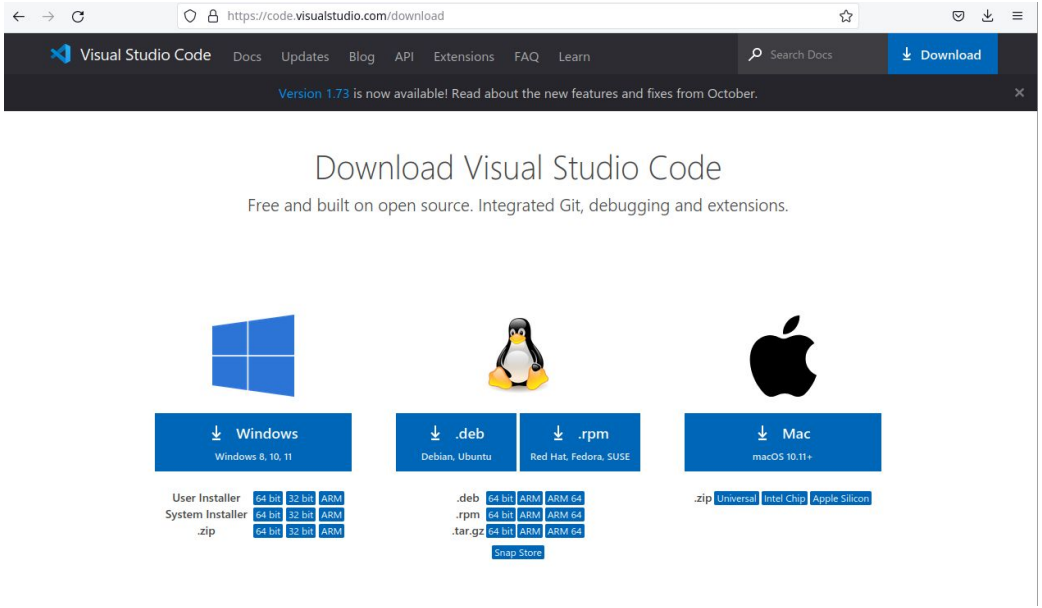
# Environment Setup

- VSCode Editor
  - 실습 진행을 하기 위해 사용할 에디터
    - 모든 OS 환경에서 사용 가능 (e.g., Windows, macOS, Linux)

  - 다양한 무료 extension 제공을 통한 개발 환경 customize 용이
    - SSH extension을 사용해 Compsec서버에 접속 후 실습 진행
    - 실습 서버: kayle.snu.ac.kr

# Environment Setup

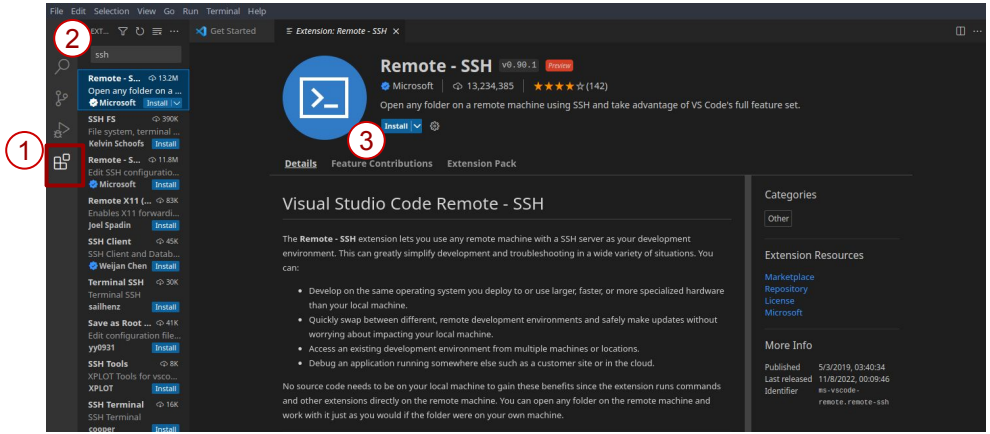
- VSCode Editor 설치 (링크)



# Environment Setup

- VSCode Editor 설치 방법
  - 아래의 링크를 통해 설치 진행
  - [Windows] : https://penguingoon.tistory.com/185
  - [Mac] : https://www.lainyzine.com/ko/article/how-to-install-visual-studio-code-on-macos/

- VSCode Extension 설치



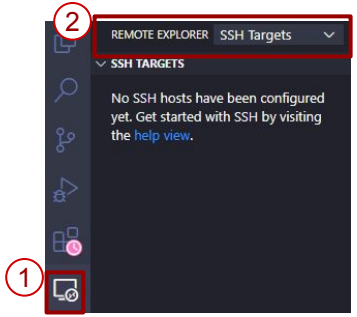

① Extension tab으로 이동

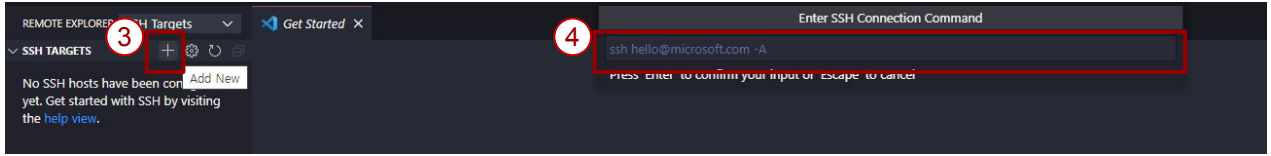
② SSH 검색

③ Remote-SSH extension 설치

# Environment Setup

- SSH 접속




① Remote Explorer tab으로 이동

② SSH Targets로 설정

③ Add New 클릭

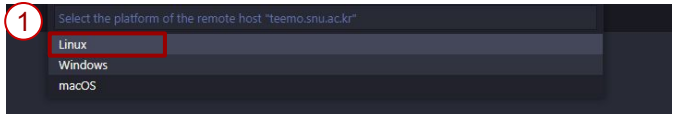
④ ssh -p XXXX compsec@kayle.snu.ac.kr 입력 (XXXX: port # to be used)

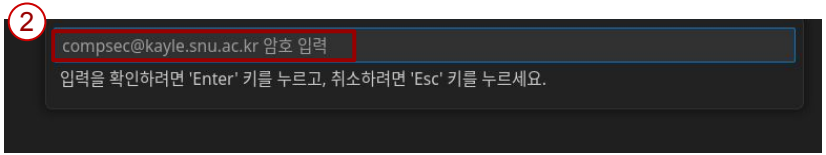
**NOTE**: port 번호 (-p) 는 추후에 공지 예정 (개인 당 포트 1개 지급)

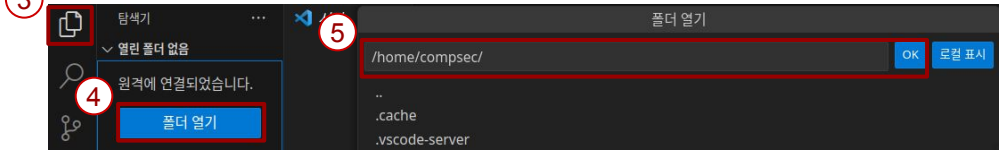
⑤ 접속 후 반드시 passwd 명령어로 패스워드 변경!

# Environment Setup

- SSH 접속 후 설정




① Linux platform 설정

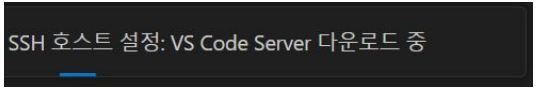
② Password 설정 (“compsec” 입력)

③ Explorer tab 이동

④ Open Folder 클릭

⑤ /home/compsec/ 디렉토리로 설정

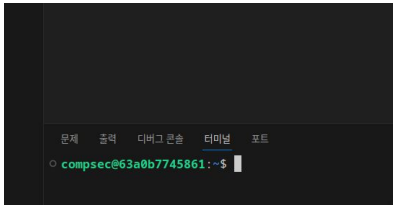
- Hang Issue: VS Code Server 다운로드 중




  - 조교에게 문의

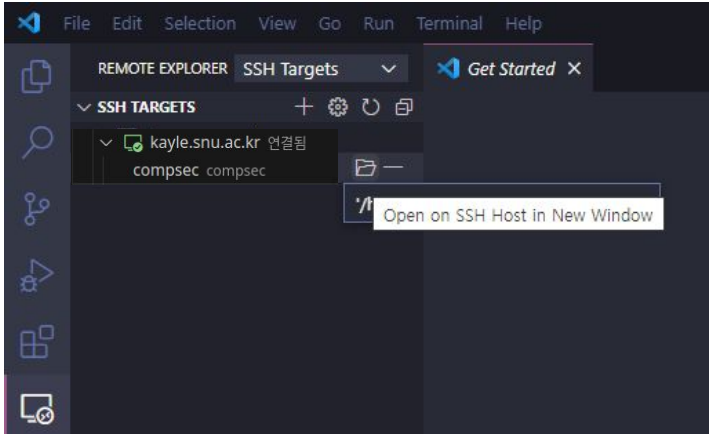
# Environment Setup

- SSH 접속 완료 (터미널)
  - Ctrl + ` : 터미널 창 단축키



# Environment Setup

- SSH 재접속
  - Remote Explorer tab에서 compsec 클릭 시 재접속 가능



# Contents

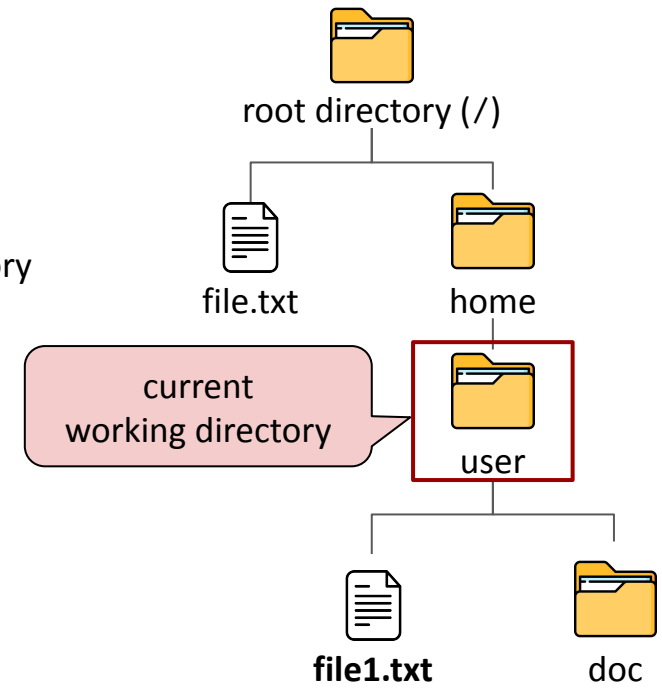
# Linux Environment

- Bash Shell
  - Default user interface
  - Every Linux User has its own home directory


- Home directory (~)
  - “~” : 현재 사용 중인 user의 home directory 의미
    - “/home/compsec” == “~”

# Linux Path

- Path
  - Two ways of representing location
  - Absolute Path
    - points to location
      - regardless of the current working directory
    - include the root directory (/)
    - example) `/home/user/fil1.txt`
  - Relative Path
    - path starts from current working directory
    - . means current working directory
    - .. means parent directory
    - example) `./file1.txt` or `~/file1.txt`



# Linux Command List

| Command | Description |
|---|---|
| pwd | print current working directory |
| ls | list current location |
| touch [new filename] | make new empty file |
| mkdir [new directory name] | make new directory |
| cd [location] | move to another location |
| cp [source] [destination] | copy source file to destination |
| cp -r [source] [destination] | copy directories recursively |
| mv [source] [destination] | move source file to destination |
| rm [filename] | remove file |
| rm -r [directory name] | remove directories and their contents recursively |

# Linux Commands in Bash Shell

- mkdir (make directory)
- cd (change directory)
- ls (list)
- rm (remove)

```
user@XXX:~ $ mkdir temp
```

# Linux Commands in Bash Shell

- mkdir (make directory)
- cd (change directory)
- ls (list)
- rm (remove)

```
user@XXX:~ $ cd temp
user@XXX:~/temp $ pwd
/home/user/temp
user@XXX:~/temp $ cd ~
user@XXX:~ $ pwd
/home/user
```

# Linux Commands in Bash Shell

- mkdir (make directory)
- cd (change directory)
- ls (list)
- rm (remove)

```
user@XXX:~/temp $ cd ..
user@XXX:~ $ ls
temp
user@XXX:~ $ ls -l
drwxrwxr-x 3 compsec compsec 4096
Jun 1 00:00 temp
```

# Linux Commands in Bash Shell

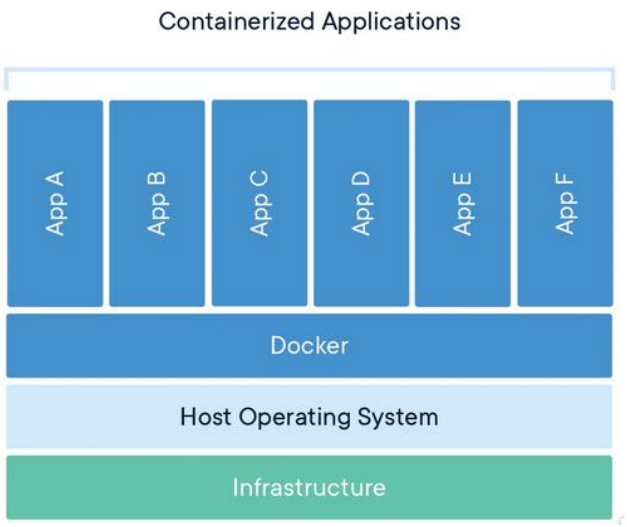
- mkdir (make directory)
- cd (change directory)
- ls (list)
- rm (remove)

```
user@XXX:~ $ touch file
user@XXX:~ $ ls
file temp
user@XXX:~ $ rm file
user@XXX:~ $ ls
temp
user@XXX:~ $ rm -r temp
user@XXX:~ $ ls
```

# Contents

# What is Docker?

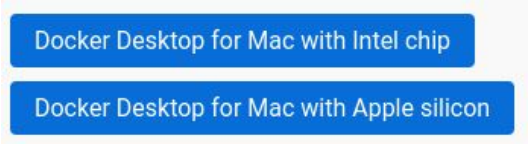
- 컨테이너 기반의 가상화 소프트웨어
  - 컨테이너 – 프로세스를 분리하여 독립적 실행을 가능케 함
    - cgroup, namespaces 기술 등을 통해 분리 및 독립
  - 이미지 배포 모델 제공
    - 개별 App. 에 실행에 필요한 실행환경을 이미지로 구성
      - OS, 라이브러리 등
- Docker의 주요 특징
  - 애플리케이션 레벨의 격리된 환경 제공
  - 팀원 모두에게 동일한 개발 환경 제공 **(Best!)**




Reference - How docker works?

# Docker Installation - Mac

- 홈페이지에서 현재 환경에 맞는 설치파일 (Docker.dmg) 다운로드 및 설치




- Docker 설치 확인

  $ docker -v

  Docker version X.X.X, build XX

# Docker Installation - Windows

- 홈페이지에서 현재 환경에 맞는 설치파일 (Docker Desktop Installer.exe) 다운로드
- Windows terminal 설치 및 실행
- Windows terminal 에서 다음 커맨드를 실행

  $ wsl  --install

  $ wsl --set-default-version 2

- 다운받은 Docker Desktop Installer.exe 실행 및 설치

  Configuration

  ☑ Use WSL 2 instead of Hyper-V (recommended)

  ☐ Add shortcut to desktop

- Windows terminal 에서 다음 커맨드를 통해 설치 확인

  $ docker -v

  Docker version X.X.X, build XX

```
PS C:\Users\malco> docker -v
Docker version 20.10.17, build 100c701
```

# Docker Installation - Linux

- 홈페이지에서 현재 환경에 맞는 설치파일 다운로드
  - 공식적으로 제공하는 배포판
    - Ubuntu
    - Debian
    - Fedora
    - Arch (Test)

- 다음 커맨드를 통해 설치 (Ubuntu 기준)

```
$ sudo apt install gnome-terminal
$ sudo apt-get update
$ sudo apt-get install ./docker-desktop-<version>-<arch>.deb
```

- 다음 커맨드를 통해 설치 확인

```
$ docker -v
Docker version X.X.X, build XX
```

# Docker Image Build

- Dockerfile을 이용하여 Docker Image 생성
  - **$ docker build -t <tag_name> -f <dockerfile_path> .**
  - **tag_name: 이미지를 관리할 이름**
  - **dockerfile_path: Dockerfile 경로**

- Build한 Docker Image 확인
  - **$ docker images**

```
REPOSITORY          TAG                 IMAGE ID            CREATED                 SIZE
<image name>        <tag>               <image id>          <created time>          <image size>
```

- *example*
  - **$ docker build -t myimage:0.1 -f /tmp/Dockerfile/**
  - **$ docker images**

```
REPOSITORY          TAG                 IMAGE ID            CREATED                 SIZE
myimage             0.1                 asdf123asd          1 seconds ago           230MB
```

# Docker Run

- 생성된 Docker Image를 통한 Container 실행 reference

```
docker run --name <contain_name> --privileged -d -p <host_port>:22 <tag_name>
```

OPTIONS

IMAGE

- OPTIONS
  - --name: identify a container by a name (e.g., container-test)
  - --privileged : Give extended privileges to this container
  - -d (detached) : the container is removed when it exits **or** when the daemon exits, whichever happens first.
  - -p (incoming ports): Publish all exposed ports to the host interfaces (e.g., -p 1234:22)

    host container
- IMAGE
  - specify a version of an image (e.g., <tag_name>, myimage:0.1)

# Example – Dockerfile Reference

```
FROM ubuntu:22.04
```

FROM: creates a layer from the ubuntu:22.04 Docker image.

```
RUN adduser --disabled-password --gecos '' compsec
```

RUN: Register a new user "**compsec**" with a Linux command adduser

```
USER compsec
```

USER: sets the user name (or UID)

```
WORKDIR /home/compsec
```

WORKDIR: sets the working directory for any instructions RUN, CMD, and etc.

```
ARG DEBIAN_FRONTEND=noninteractive
```

ARG: defines a variable that users can pass at build-time to the builder with the docker build command

```
RUN sudo apt-get install openssh-server -y
```

```
CMD ["sudo", "/usr/sbin/sshd", "-D"]
```

CMD: define what command gets executed when running a container
(example: runs a ssh daemon to allow SSH connection)