

Name: Bharath Karumudi
Assignment: Lab 1 – Computer Security

Task 1: Manipulating Environment Variables

```
[10/08/19]seed@VM:~$ printenv
XDG_VTNR=7
ORBIT_SOCKETDIR=/tmp/orbit-seed
XDG_SESSION_ID=c1
XDG_GREETER_DATA_DIR=/var/lib/lightdm-data/seed
IBUS_DISABLE_SNOOPER=1
TERMINATOR_UUID=urn:uuid:ffa3415d-001b-4a92-a903-b5a3004eb102
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=2889
ANDROID_HOME=/home/seed/android/android-sdk-linux
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1
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
WINDOWID=60817412
UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-session/1000/1130
GNOME_KEYRING_CONTROL=
GTK_MODULES=gail:atk-bridge:unity-gtk-module
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:cd=40;33:or=01;31: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:*.lzh=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.Z=01;31:*.diz=01;31:*.gz=01;31:*.lrz=01;31:*.lzo=01;31:*.xz=01;31:*.bzip2=01;31:*.bz=01;31:*.tbz=01;31:*.taz=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:*.jpg=01;35:*.jpeg=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=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:*.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:
QT_ACCESSIBILITY=1
LD_LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib:
XDG_SESSION_PATH=/org/freedesktop/DisplayManager/Session0
XDG_SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
DEFAULTS_PATH=/usr/share/gconf/ubuntu.default.path
GIO_LAUNCHED_DESKTOP_FILES=/usr/share/applications/terminator.desktop
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/usr/share/upstart/xdg:/etc/xdg
DESKTOP_SESSION=ubuntu
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/a
```

```
[10/08/19]seed@VM:~$ env | grep HOME
ANDROID_HOME=/home/seed/android/android-sdk-linux
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
JAVA_HOME=/usr/lib/jvm/java-8-oracle
HOME=/home/seed
[10/08/19]seed@VM:~$
```

Observation: I used `printenv` command to print all the environment variables that are set. In the second screenshot, I used `env` command which is similar to `printenv` and to see only the variables that are with word `HOME`, so sent the `env` output to `grep`.

Explanation: The `printenv` command gets all the environment variables that are set and displays on the terminal. The `env` also do the same and with `grep`, I filtered the output which has the word “HOME”.

```

[10/08/19]seed@VM:~$
[10/08/19]seed@VM:~$ env | grep HOME
ANDROID_HOME=/home/seed/android/android-sdk-linux
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
JAVA_HOME=/usr/lib/jvm/java-8-oracle
HOME=/home/seed
[10/08/19]seed@VM:~$
[10/08/19]seed@VM:~$ export global_Home=/home
[10/08/19]seed@VM:~$
[10/08/19]seed@VM:~$ env | grep global_Home
global_Home=/home
[10/08/19]seed@VM:~$
[10/08/19]seed@VM:~$ unset global_Home
[10/08/19]seed@VM:~$ env | grep global_Home
[10/08/19]seed@VM:~$
[10/08/19]seed@VM:~$
```

Observation: In this, I first printed all the HOME variables as check and then defined a new variable called “global_Home” with “export” and assigned a value as /home. This was verified by printing the **env** and with **grep**.

Later, the same variable was unset with “unset” command and verified with **env** again and we can see nothing was shown.

Explanation: The export command with set the new environment variable and the unset will unassign the variable, which we can see from the implementation.

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

Step 1:

```
[10/08/19]seed@VM:~/.../Lab1$ vi parent-child.c
[10/08/19]seed@VM:~/.../Lab1$ gcc -o child.out parent-child.c
[10/08/19]seed@VM:~/.../Lab1$ ls -lrt
total 12
-rw-rw-r-- 1 seed seed 342 Oct  8 21:52 parent-child.c
-rwxrwxr-x 1 seed seed 7500 Oct  8 21:52 child.out
[10/08/19]seed@VM:~/.../Lab1$ child.out > child
[10/08/19]seed@VM:~/.../Lab1$ more child
XDG_VTNR=7
ORBIT_SOCKETDIR=/tmp/orbit-seed
XDG_SESSION_ID=c1
XDG_GREETER_DATA_DIR=/var/lib/lightdm-data/seed
IBUS_DISABLE_SNOOPER=1
TERMINATOR_UUID=urn:uuid:0b8988f5-claf-4ea4-8837-6a4144f9775f
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=3124
ANDROID_HOME=/home/seed/android/android-sdk-linux
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
```

Observation: Created a program file with given code in lab instructions with the name *parent-child.c* and compiled with gcc with executable as *child.out* to get the child process output.

When compiled and ran the program, the output of the program was all the environment variables of child process and are saved to a file called *child*.

Explanation: When forked from the main process, the child process will be created and with `printenv()`, it will print all the environment variables that are set for the child process. Generally, the child process will inherit all the environment variables from parent process.

Step 2:

```
[10/08/19]seed@VM:~/.../Lab1$ vi parent-child.c
[10/08/19]seed@VM:~/.../Lab1$ gcc -o parent.out parent-child.c
[10/08/19]seed@VM:~/.../Lab1$
[10/08/19]seed@VM:~/.../Lab1$ ./parent.out > parent
[10/08/19]seed@VM:~/.../Lab1$ more parent
XDG_VTNR=7
ORBIT_SOCKETDIR=/tmp/orbit-seed
XDG_SESSION_ID=c1
XDG_GREETER_DATA_DIR=/var/lib/lightdm-data/seed
IBUS_DISABLE_SNOOPER=1
TERMINATOR_UUID=urn:uuid:0b8988f5-claf-4ea4-8837-6a4144f9775f
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=3124
ANDROID_HOME=/home/seed/android/android-sdk-linux
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1
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
WINDOWID=68817412
UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-session/1000/1130
GNOME_KEYRING_CONTROL=
GTK_MODULES=gail:atk-bridge:unity-gtk-module
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:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32*:tar=01;31*:taz=01;31*:arc=01;31*:arj=01;31*:taz=01;31*:lha=01;31*:lza=01;31*:lzh=01;31*:lzm=01;31*:tlz=01;31*:txz=01;31*:tzo=01;31*:t7z=01;31*:zip=01;31*:z=01;31*:Z=01;31*:dz=01;31*:gz=01;31*:lrz=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*:rz=01;31*:cab=01;31*:jpg=01;35*:jpeg=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=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*: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*:
QT_ACCESSIBILITY=1
LD_LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib:
XDG_SESSION_PATH=/org/freedesktop/DisplayManager/Session0
XDG_SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
DEFAULTS_PATH=/usr/share/gconf/ubuntu.default.path
GIO_LAUNCHED_DESKTOP_FILE=/usr/share/applications/terminator.desktop
```

Observation: This time, the same program *parent-child.c* was recompiled by modifying the program by commenting the `printenv()` under child and uncommented the `printenv()` under default process. The program was compiled and created a new executable called "*parent.out*" and ran the process and stored the output to a file called *parent*.

Explanation: This time the parent process executed the `printenv()` and printed its environment variables and are saved to a file "parent".

Step 3:

```
[10/08/19]seed@VM:~/.../Lab1$ ls -lrt
total 36
-rwxrwxr-x 1 seed seed 7500 Oct  8 21:52 child.out
-rw-rw-r-- 1 seed seed  342 Oct  8 21:55 parent-child.c
-rwxrwxr-x 1 seed seed 7500 Oct  8 21:56 parent.out
-rw-rw-r-- 1 seed seed 4274 Oct  8 22:06 parent
-rw-rw-r-- 1 seed seed 4273 Oct  8 22:07 child
[10/08/19]seed@VM:~/.../Lab1$
[10/08/19]seed@VM:~/.../Lab1$ diff parent child
75c75
< _=./parent.out
---
> _=./child.out
[10/08/19]seed@VM:~/.../Lab1$
```

Observation: Compared the outputs of child from step1 and parent from step 2 with a `diff` command, which shows the differences between two files. The output was no difference except the executing process name.

Explanation: This was due to Child process inherits the environment variables from parent process. So all the environment variables are same between two output files. The only difference noticed is executing process name. In our case, step 1 and step2 has two different executable names, hence the difference is in executing process names. If both are same, then we could see no differences with `diff` output.

Task 3: Environment Variables and `execve()`

Step 1:

```
[10/08/19]seed@VM:~/.../Lab1$ vi execve_pgm.c
[10/08/19]seed@VM:~/.../Lab1$ gcc -o execve_pgm.out execve_pgm.c
execve_pgm.c: In function 'main':
execve_pgm.c:11:2: warning: implicit declaration of function 'execve' [-Wimplicit-function-declaration]
    execve("/usr/bin/env", argv, NULL);
    ^
[10/08/19]seed@VM:~/.../Lab1$ █
[10/08/19]seed@VM:~/.../Lab1$ ./execve_pgm.out
[10/08/19]seed@VM:~/.../Lab1$
```

Observation: A new c program has been created – `execve_pgm.c`, compiled and created the executable as `execve_pgm.out`, where it has `NULL` as third argument for `execve` command. The output for this was `NULL` or empty result.

Explanation: This was due to the third argument where generally we have to pass environ then it will get the content. But in this case, we told env to get from NULL. So nothing was printed or empty result.

Step 2:

```
08/08/19]seed@M:-/.../Lab15 ./execve_pgm_2_out
KDG VTRN=r
ORBIT SOCKETDIR=/tmp/orbit-seed
KDG SESSION ID=1
KDG GREETER DATA DIR=/var/lib/lightdm-data/seed
IDUS DISABLE SNOOPER=1
TERMINATOR UUID=urn:uuid:0ba986f5-claf-4ea4-8337-6a4144f9775f
CLUTTER IN MODULE=win
SESSION=ubuntu
$IO LAUNCHED DESKTOP FILE PID=3124
ANDROID_HOME=/home/seed/android/android-sdk-linux
PKG_AGENT_INFO=/home/seed/.gnupg/S/gpg-agent:0:1
TERM=xterm
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
QT_LINUX ACCESSIBILITY ALWAYS_On=1
LD PRELOAD=/home/seed/lib/boost/libboost_program_options.so.1.64.0:/home/seed/lib/boost/libboost_filesystem.so.1.64.0:/lib64/ld-linux-x86_64.so.2
UPSTART SESSION=unix:abstract:/com/ubuntu/upstart-session/1000/1130
$HOME KEYRING CONTROL=
GTK_MODULES=gail:atk-bridge:unity-gtk-module
JSEB=seed
LS COLORS=sr:di=0;1;34:ln=0;1;36:mh=0;pi=40;33:sow=0;1;35:dow=0;1;35:bdw=0;33;0;1:cd=40;33;0;1:or=40;31;0;1:mi=
0;31;31::tze=0;1;31::ar=0;1;31::arj=0;1;31::az=0;1;31::lha=0;1;31::lza=0;1;31::lzh=0;1;31::lzm=
0;31;31::z=0;1;31::Z=0;1;31::d=0;1;31::gz=0;1;31::lrz=0;1;31::lz=0;1;31::lzo=0;1;31::xz=0;1;31::bz2=0;1;31::
0;1;31::jar=0;1;31::war=0;1;31::ear=0;1;31::sar=0;1;31::rar=0;1;31::alz=0;1;31::ace=0;1;31::zoo=0;1;31::
0;1;35::gi=0;1;35::bmp=0;1;35::pbm=0;1;35::pgm=0;1;35::ppm=0;1;35::tga=0;1;35::xbm=0;1;35::xpm=0;1;35::
nq=0;1;35::pcx=0;1;35::m=0;v=0;1;35::mpg=0;1;35::mpeg=0;1;35::mzv=0;1;35::mkv=0;1;35::webm=0;1;35::ogm=0;1;35::
nuv=0;1;35::wmv=0;1;35::asf=0;1;35::rm=0;1;35::rmvb=0;1;35::flc=0;1;35::avi=0;1;35::fli=0;1;35::flv=0;1;35::
0;1;35::emf=0;1;35::svg=0;1;35::aac=0;0;36::au=0;0;36::flac=0;0;36::m4a=0;0;36::mid=0;0;36::
ra=0;0;36::wav=0;0;36::oga=0;0;36::opus=0;0;36::spx=0;0;36::xspf=0;0;36:
LD ACCESSIBILITY=1
LD LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib:
KDG SESSION PATH=/org/freedesktop/DisplayManager/Seat0
KDG SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
SSH_AUTH_SOCKETS/run/user/1000/keyring/gpg
DEFAULTS_PATH=/usr/share/gconf/ubuntu.default.path
$IO LAUNCHED DESKTOP FILE=/usr/share/applications/terminator.desktop
KDG CONFIG DIRS=/etc/cxdg-ubuntu:/usr/share/upstart/xdg:/etc/cxdg
DESKTOP_SESSION=ubuntu
PATH=/home/seed/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/games:/usr/local/ga
-e -n /usr/lib/jvm/java-8-oracle/jre bin:/usr/bin/android/android-sdk-linux/tools:/home/se
android-nrk/android-nrk-rdr:/home/seed/_local/bin
```

Observation: Modified the same “execve_pgm.c” and this time added the “environ” as third argument and recompiled. The output was all environment variables that are set.

Explanation: This time, the env command was executed with the argument “environ” which is a pointer to envp, which has all the environment variables in name, value pairs. So the env command executed and printed all the environment variables.

Step 3:

Explanation/Conclusion: The conclusion from step 1 and step 2 is, in step 1 we told env to execute with NULL so it didn't print anything and in step 2, we told env to get the content of environ, where it has all the environment variables. That is the reason, in step 2 gets all the environment variables.

Task 4: Environment Variables and system()

```
[10/08/19]seed@VM:~/.../Lab1$ vi system_func.c
[10/08/19]seed@VM:~/.../Lab1$
[10/08/19]seed@VM:~/.../Lab1$ gcc -o system_func.out system_func.c
[10/08/19]seed@VM:~/.../Lab1$
[10/08/19]seed@VM:~/.../Lab1$ ./system_func.out
LESSOPEN=| /usr/bin/lesspipe %s
GNOME_KEYRING_PID=
USER=seed
LANGUAGE=en_US
UPSTART_INSTANCE=
J2SDKDIR=/usr/lib/jvm/java-8-oracle
XDG_SEAT=seat0
SESSION=ubuntu
XDG_SESSION_TYPE=x11
COMPIZ_CONFIG_PROFILE=ubuntu-lowgfx
ORBIT_SOCKETDIR=/tmp/orbit-seed
LD_LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib:
SHLVL=1
LIBGL_ALWAYS_SOFTWARE=1
J2REDIR=/usr/lib/jvm/java-8-oracle/jre
HOME=/home/seed
QT4_IM_MODULE=xim
OLDPWD=/home/seed/Documents
DESKTOP_SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE=/usr/share/applications/terminator.desktop
QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1
GTK_MODULES=gail:atk-bridge:unity-gtk-module
XDG_SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
INSTANCE=
DBUS_SESSION_BUS_ADDRESS=unix:abstract=/tmp/dbus-d0DvYXZ44A
GIO_LAUNCHED_DESKTOP_FILE_PID=3124
COLORTERM=gnome-terminal
GNOME_KEYRING_CONTROL=
QT_QPA_PLATFORMTHEME=appmenu-qt5
MANDATORY_PATH=/usr/share/gconf/ubuntu.mandatory.path
IM_CONFIG_PHASE=1
SESSIONTYPE=gnome-session
UPSTART_JOB=unity7
LOGNAME=seed
GTK_IM_MODULE=ibus
WINDOWID=60817412
=./system_func.out
```

Observation: Created the given program to “system_func.c” and compiled to “system_func.out”. When the program executed, it printed all the environment variables.

Explanation: With system() the shell (/bin/sh -c /usr/bin/env) was called and executed the env command and the output was displayed with all the environment variables on the terminal as shown in the screenshot.

Task 5: Environment Variable and Set-UID Programs

Step 1:

```
/bin/bash
[10/14/19]seed@VM:~/.../Lab1$ vi set_uid_1.c
[10/14/19]seed@VM:~/.../Lab1$ gcc -o set_uid_1.o set_uid_1.c
[10/14/19]seed@VM:~/.../Lab1$ ls -l set_uid_1.*
-rw-rw-r-- 1 seed seed 160 Oct 14 19:18 set_uid_1.c
-rwxrwxr-x 1 seed seed 7400 Oct 14 19:18 set_uid_1.o
[10/14/19]seed@VM:~/.../Lab1$ ./set_uid_1.o | more
XDG_VTNR=7
ORBIT_SOCKETDIR=/tmp/orbit-seed
XDG_SESSION_ID=c1
XDG_GREETER_DATA_DIR=/var/lib/lightdm-data/seed
IBUS_DISABLE_SNOOPER=1
TERMINATOR_UUID=urn:uuid:0ce21642-3c18-4d12-9e80-83400fb2a9e5
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=15074
ANDROID_HOME=/home/seed/android/android-sdk-linux
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1
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
WINDOWID=25165828
UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-session/1000/1119
GNOME_KEYRING_CONTROL=
GTK_MODULES=gail:atk-bridge:unity-gtk-module
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:cd=40;33: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:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tzo=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:*.jpg=01;35:*.jpeg=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=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:*.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:
QT_ACCESSIBILITY=1
LD_LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib:
XDG_SESSION_PATH=/org/freedesktop/DisplayManager/Session0
XDG_SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
```

Observation: Created a executable with the given program called “set_uid_1.o” and when executed with current user and as a normal program, it printed all the environment variables.

Explanation: The program was executed with current user and printed the environment variables for the current user.

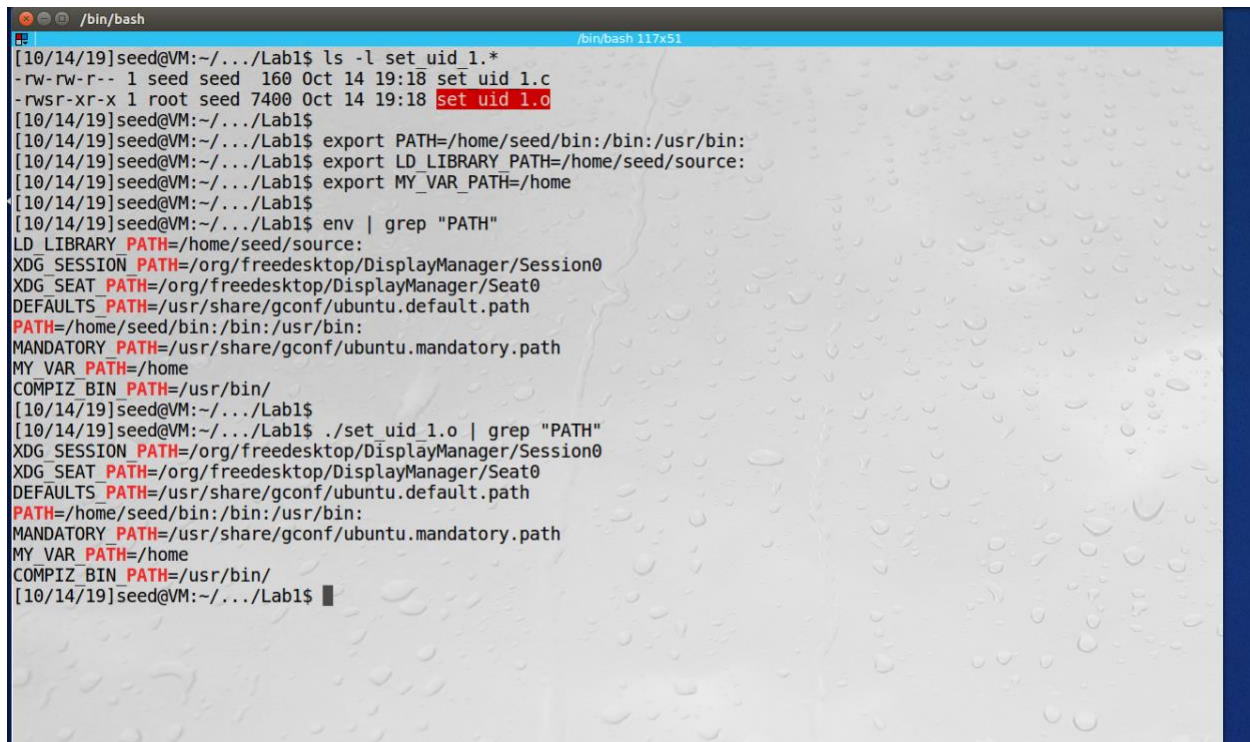
Step 2:

```
/bin/bash
/bin/bash 80x52
ANDROID_HOME=/home/seed/android/android-sdk-linux
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1
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
WINDOWID=25165828
UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-session/1000/1119
GNOME_KEYRING_CONTROL=
GTK_MODULES=gail:atk-bridge:unity-gtk-module
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:cd=40;33: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:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.taz=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:*.jpg=01;35:*.jpeg=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=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:*.dcm=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:
QT_ACCESSIBILITY=1
LD_LIBRARY_PATH=/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib:
XDG_SESSION_PATH=/org/freedesktop/DisplayManager/Session0
XDG_SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
DEFAULTS_PATH=/usr/share/gconf/ubuntu.default.path
GIO_LAUNCHED_DESKTOP_FILE=/usr/share/applications/terminator.desktop
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/usr/share/upstart/xdg:/etc/xdg
DESKTOP_SESSION=ubuntu
[10/14/19]seed@VM:~/.../Lab1$ ls -l set_uid 1.*
-rw-rw-r-- 1 seed seed 160 Oct 14 19:18 set_uid 1.c
-rwxrwxr-x 1 seed seed 7400 Oct 14 19:18 set_uid 1.o
[10/14/19]seed@VM:~/.../Lab1$
[10/14/19]seed@VM:~/.../Lab1$ sudo chown root set_uid 1.o
[10/14/19]seed@VM:~/.../Lab1$ sudo chmod 4755 set_uid 1.o
[10/14/19]seed@VM:~/.../Lab1$ ls -l set_uid 1.*
-rw-rw-r-- 1 seed seed 160 Oct 14 19:18 set_uid 1.c
-rwsr-xr-x 1 root seed 7400 Oct 14 19:18 set_uid 1.o
[10/14/19]seed@VM:~/.../Lab1$
```

Observation: The set_uid_1.o was now changed to root owned set-UID program with chown and chmod commands as shown in screenshot and the result can be seen in red color.

Explanation: The program was now root-owned set UID program and the impact will be in following step3.

Step 3:

A terminal window titled '/bin/bash' showing a series of commands and their outputs. The user is in a directory ~/.../Lab1. They run 'ls -l set uid 1.*' which shows two files: 'set uid 1.c' and 'set uid 1.o'. Then they run 'export PATH=/home/seed/bin:/bin:/usr/bin:', 'export LD_LIBRARY_PATH=/home/seed/source:', and 'export MY_VAR_PATH=/home'. Next, they run 'env | grep "PATH"', which outputs several environment variables including LD_LIBRARY_PATH, XDG_SESSION_PATH, XDG_SEAT_PATH, DEFAULTS_PATH, PATH, MANDATORY_PATH, MY_VAR_PATH, and COMPIZ_BIN_PATH. Finally, they run './set uid 1.o | grep "PATH"', which outputs the same set of variables as the previous command, but notably, LD_LIBRARY_PATH is missing from this output.

```
[10/14/19]seed@VM:~/.../Lab1$ ls -l set uid 1.*
-rw-rw-r-- 1 seed seed 160 Oct 14 19:18 set uid 1.c
-rwsr-xr-x 1 root seed 7400 Oct 14 19:18 set uid 1.o
[10/14/19]seed@VM:~/.../Lab1$
[10/14/19]seed@VM:~/.../Lab1$ export PATH=/home/seed/bin:/bin:/usr/bin:
[10/14/19]seed@VM:~/.../Lab1$ export LD_LIBRARY_PATH=/home/seed/source:
[10/14/19]seed@VM:~/.../Lab1$ export MY_VAR_PATH=/home
[10/14/19]seed@VM:~/.../Lab1$
[10/14/19]seed@VM:~/.../Lab1$ env | grep "PATH"
LD_LIBRARY_PATH=/home/seed/source:
XDG_SESSION_PATH=/org/freedesktop/DisplayManager/Session0
XDG_SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
DEFAULTS_PATH=/usr/share/gconf/ubuntu.default.path
PATH=/home/seed/bin:/bin:/usr/bin:
MANDATORY_PATH=/usr/share/gconf/ubuntu.mandatory.path
MY_VAR_PATH=/home
COMPIZ_BIN_PATH=/usr/bin/
[10/14/19]seed@VM:~/.../Lab1$
[10/14/19]seed@VM:~/.../Lab1$ ./set uid 1.o | grep "PATH"
XDG_SESSION_PATH=/org/freedesktop/DisplayManager/Session0
XDG_SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
DEFAULTS_PATH=/usr/share/gconf/ubuntu.default.path
PATH=/home/seed/bin:/bin:/usr/bin:
MANDATORY_PATH=/usr/share/gconf/ubuntu.mandatory.path
MY_VAR_PATH=/home
COMPIZ_BIN_PATH=/usr/bin/
[10/14/19]seed@VM:~/.../Lab1$
```

Observation: The PATH, LD_LIBRARY_PATH, MY_VAR_PATH are set to new values and then validated with env and piped to grep command. Where I can observe all the three variables are there. But when executed our program which prints all the environment variables, then the all are printed, but LD_LIBRARY_PATH was not printed out.

Explanation:

When the program ran with Set-UID, the new PATH and MY_VAR_PATH are inherited to child process, but LD_LIBRARY_PATH was not. The above screenshot shows the difference between the shell env and the program output environment variables. Because, the LD_LIBRARY_PATH will not be inherited when executed with a set-UID program.

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

```
/bin/bash
[10/14/19]seed@VM:~$ cd Documents/
[10/14/19]seed@VM:~/Documents$ cd Lab1/
[10/14/19]seed@VM:~/.../Lab1$ pwd
/home/seed/Documents/Lab1
[10/14/19]seed@VM:~/.../Lab1$ ls -lrt
total 156
-rwxrwxr-x 1 seed seed 7500 Oct 8 21:52 child.out
-rw-rw-r-- 1 seed seed 342 Oct 8 21:55 parent-child.c
-rwxrwxr-x 1 seed seed 7500 Oct 8 21:56 parent.out
-rw-rw-r-- 1 seed seed 4274 Oct 8 22:06 parent
-rw-rw-r-- 1 seed seed 4273 Oct 8 22:07 child
-rwxrwxr-x 1 seed seed 7400 Oct 8 22:18 execve_pgm.out
-rw-rw-r-- 1 seed seed 191 Oct 8 22:20 execve_pgm.c
-rwxrwxr-x 1 seed seed 7452 Oct 8 22:21 execve_pgm_2.out
-rw-rw-r-- 1 seed seed 92 Oct 8 22:29 system_func.c
-rwxrwxr-x 1 seed seed 7352 Oct 8 22:29 system_func.out
-rw-rw-r-- 1 seed seed 725 Oct 8 22:47 path.bkp
-rw-rw-r-- 1 seed seed 160 Oct 8 22:49 set_uid_1.c
-rwsr-xr-x 1 root seed 7400 Oct 8 22:51 set_uid_1.out
-rw-rw-r-- 1 seed seed 42 Oct 8 23:25 ls.c
-rwsr-xr-x 1 root seed 7352 Oct 8 23:25 ls
-rw-rw-r-- 1 seed seed 149 Oct 9 21:54 mylib.c
-rw-rw-r-- 1 seed seed 2588 Oct 9 21:55 mylib.o
-rwxrwxr-x 1 seed seed 7928 Oct 9 21:55 libmylib.so.1.0.1
-rw-rw-r-- 1 seed seed 52 Oct 9 21:57 myprog.c
-rwsr-xr-x 1 user1 seed 7348 Oct 9 21:57 myprog.out
-rwsr-xr-x 1 root seed 7548 Oct 10 00:22 syst.o
-rw-rw-r-- 1 seed seed 431 Oct 10 00:34 sys_exe.c
-rwsr-xr-x 1 root seed 7548 Oct 10 00:34 exect.o
-rw-rw-r-- 1 seed seed 888 Oct 11 14:36 capability_leak.c
-rwsr-xr-x 1 root seed 7648 Oct 11 14:37 capability_leak.o
[10/14/19]seed@VM:~/.../Lab1$
[10/14/19]seed@VM:~/.../Lab1$ export PATH=$PWD:$PATH
[10/14/19]seed@VM:~/.../Lab1$ ls
^C[10/14/19]seed@VM:~/.../Lab1$
[10/14/19]seed@VM:~/.../Lab1$
```

Observation: Created the new program “set_uid_1.c” with the given code and compiled to an executable with name “ls”. Changed the program to a set-UID root program with chmod 4755 on “ls”. Modified the PATH variable, by prepending the current path. When executed the ls, the command didn’t work as actual ls command. Before making the changes to PATH the command was executing normally, where we can see that in the screenshot.

Explanation: As the PATH has the current directory as first place to check. The shell found the ls which is in the current directory and executed our custom ls program rather than /bin/ls and also ran with root as effective user id. The PATH variable impacted the functionality.

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

Step 1:

(1, 2, 3, 4):

```
[10/09/19]seed@VM:~/.../Lab1$ vi mylib.c
[10/09/19]seed@VM:~/.../Lab1$ gcc -fPIC -g -c mylib.c
[10/09/19]seed@VM:~/.../Lab1$ gcc -shared -o libmylib.so.1.0.1 mylib.o -lc
[10/09/19]seed@VM:~/.../Lab1$ export LD_PRELOAD=./libmylib.so.1.0.1
[10/09/19]seed@VM:~/.../Lab1$ vi myprog.c
[10/09/19]seed@VM:~/.../Lab1$ gcc -o myprog.out myprog.c
myprog.c: In function 'main':
myprog.c:4:2: warning: implicit declaration of function 'sleep' [-Wimplicit-function-declaration]
  sleep(1);
  ^
[10/09/19]seed@VM:~/.../Lab1$
[10/09/19]seed@VM:~/.../Lab1$ ls -lrt | tail -5
-rw-rw-r-- 1 seed seed 149 Oct 9 21:54 mylib.c
-rw-rw-r-- 1 seed seed 2588 Oct 9 21:55 mylib.o
-rwxrwxr-x 1 seed seed 7928 Oct 9 21:55 libmylib.so.1.0.1
-rw-rw-r-- 1 seed seed 52 Oct 9 21:57 myprog.c
-rwxrwxr-x 1 seed seed 7348 Oct 9 21:57 myprog.out
[10/09/19]seed@VM:~/.../Lab1$
```

Observation: Created a mylib.c and myprog.c file and compiled and also exported the LD_PRELOAD environment variable as instructed and also created the executables as mylib.o and myprog.out respectively.

Explanation: Setup the pre-work as instructed and program compilation which compiled and also modified the environment variable LD_PRELOAD value to the custom one by using export. The impact will be seen in following step 2.

Step 2:

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

```
tor
/bin/bash
[10/14/19]seed@VM:~/.../Lab1$ ls -lrt my*
-rw-rw-r-- 1 seed seed 149 Oct 9 21:54 mylib.c
-rw-rw-r-- 1 seed seed 2588 Oct 9 21:55 mylib.o
-rw-rw-r-- 1 seed seed 52 Oct 9 21:57 myprog.c
-rwxr-xr-x 1 seed seed 7348 Oct 9 21:57 myprog.out
[10/14/19]seed@VM:~/.../Lab1$ whoami
seed
[10/14/19]seed@VM:~/.../Lab1$ export LD_PRELOAD=./libmylib.so.1.0.1
[10/14/19]seed@VM:~/.../Lab1$
[10/14/19]seed@VM:~/.../Lab1$ ./myprog.out
I am not sleeping!
[10/14/19]seed@VM:~/.../Lab1$
```

Observation: When executed the program as normal and with seed user, the program took the custom sleep function rather from libc.

Explanation: Because we forced the LD_PRELOAD environment variable to take the custom library.

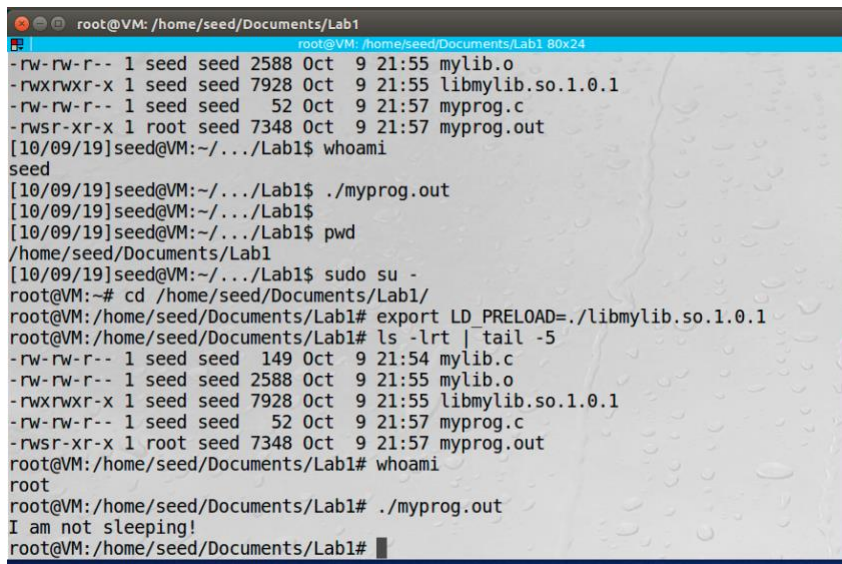
Case 2: Make myprog a Set-UID root program and run it as a normal user.

```
/bin/bash
[10/09/19]seed@VM:~/.../Lab1$ ls -lrt my*
-rwxrwxr-x 1 seed seed 7352 Oct 8 22:29 system_func.out
-rw-rw-r-- 1 seed seed 725 Oct 8 22:47 path.bkp
-rw-rw-r-- 1 seed seed 160 Oct 8 22:49 set uid 1.c
-rwsr-xr-x 1 root seed 7400 Oct 8 22:51 set uid 1.out
-rw-rw-r-- 1 seed seed 42 Oct 8 23:25 ls.c
-rwsr-xr-x 1 root seed 7352 Oct 8 23:25 ls
-rw-rw-r-- 1 seed seed 149 Oct 9 21:54 mylib.c
-rw-rw-r-- 1 seed seed 2588 Oct 9 21:55 mylib.o
-rwxrwxr-x 1 seed seed 7928 Oct 9 21:55 libmylib.so.1.0.1
-rw-rw-r-- 1 seed seed 52 Oct 9 21:57 myprog.c
-rwxrwxr-x 1 seed seed 7348 Oct 9 21:57 myprog.out
[10/09/19]seed@VM:~/.../Lab1$ ./myprog.out
[10/09/19]seed@VM:~/.../Lab1$ sudo chown root myprog.out
[10/09/19]seed@VM:~/.../Lab1$ sudo chmod 4755 myprog.out
[10/09/19]seed@VM:~/.../Lab1$ ls -lrt | tail -5
-rw-rw-r-- 1 seed seed 149 Oct 9 21:54 mylib.c
-rw-rw-r-- 1 seed seed 2588 Oct 9 21:55 mylib.o
-rwxrwxr-x 1 seed seed 7928 Oct 9 21:55 libmylib.so.1.0.1
-rw-rw-r-- 1 seed seed 52 Oct 9 21:57 myprog.c
-rwsr-xr-x 1 root seed 7348 Oct 9 21:57 myprog.out
[10/09/19]seed@VM:~/.../Lab1$ whoami
seed
[10/09/19]seed@VM:~/.../Lab1$ ./myprog.out
[10/09/19]seed@VM:~/.../Lab1$
```

Observation: When executed the program as root owned set-UID program with seed user the program went to sleep for a second.

Explanation: Because when you run as set-UID program, the LD_PRELOAD we exported for seed user will not get effective and took the default library.

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

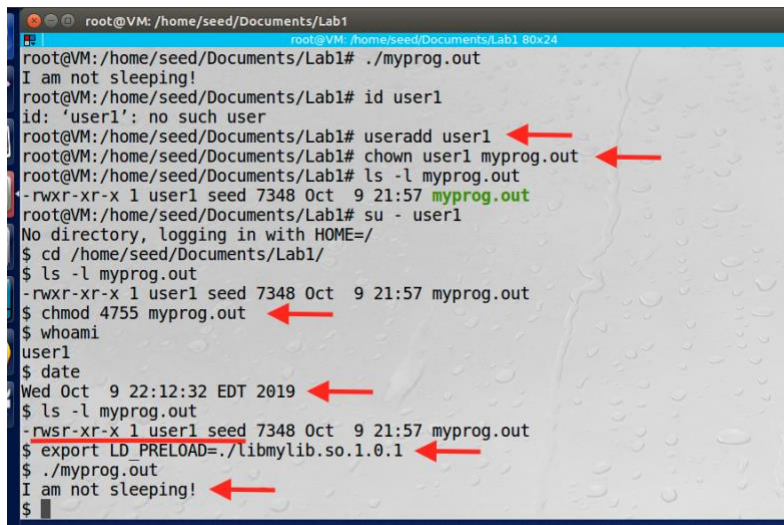
A terminal window titled 'root@VM: /home/seed/Documents/Lab1' with a blue header bar. The terminal shows a series of commands and their outputs. It starts with a file listing showing permissions for 'mylib.o', 'libmylib.so.1.0.1', 'myprog.c', and 'myprog.out'. Then, the user 'seed' runs 'whoami' (output: seed), './myprog.out', 'pwd' (output: /home/seed/Documents/Lab1), and 'sudo su -' to become root. As root, they run 'cd /home/seed/Documents/Lab1/', 'export LD_PRELOAD=./libmylib.so.1.0.1', 'ls -lrt | tail -5' (showing the same file list as before), and 'whoami' (output: root). Finally, they run './myprog.out' as root, which outputs 'I am not sleeping!' before returning to the root prompt.

```
root@VM: /home/seed/Documents/Lab1
root@VM: /home/seed/Documents/Lab1 80x24
-rw-rw-r-- 1 seed seed 2588 Oct 9 21:55 mylib.o
-rwxrwxr-x 1 seed seed 7928 Oct 9 21:55 libmylib.so.1.0.1
-rw-rw-r-- 1 seed seed 52 Oct 9 21:57 myprog.c
-rwsr-xr-x 1 root seed 7348 Oct 9 21:57 myprog.out
[10/09/19]seed@VM:~/.../Lab1$ whoami
seed
[10/09/19]seed@VM:~/.../Lab1$ ./myprog.out
[10/09/19]seed@VM:~/.../Lab1$ pwd
/home/seed/Documents/Lab1
[10/09/19]seed@VM:~/.../Lab1$ sudo su -
root@VM:~# cd /home/seed/Documents/Lab1/
root@VM:/home/seed/Documents/Lab1# export LD_PRELOAD=./libmylib.so.1.0.1
root@VM:/home/seed/Documents/Lab1# ls -lrt | tail -5
-rw-rw-r-- 1 seed seed 149 Oct 9 21:54 mylib.c
-rw-rw-r-- 1 seed seed 2588 Oct 9 21:55 mylib.o
-rwxrwxr-x 1 seed seed 7928 Oct 9 21:55 libmylib.so.1.0.1
-rw-rw-r-- 1 seed seed 52 Oct 9 21:57 myprog.c
-rwsr-xr-x 1 root seed 7348 Oct 9 21:57 myprog.out
root@VM:/home/seed/Documents/Lab1# whoami
root
root@VM:/home/seed/Documents/Lab1# ./myprog.out
I am not sleeping!
root@VM:/home/seed/Documents/Lab1#
```

Observation: When the root owned set-UID program was ran with root user by switching to root account. The regular sleep from libc was not executed, but our custom sleep program was. As a proof we can see our custom message “I am not sleeping” was displayed.

Explanation: Because when executing the root owned set-UID program the effective user will be root and it will take root environment variables. In this case, for root we already defined a custom library for LD_PRELOAD, so the custom sleep function was executed rather from libc.

Case 4: Make myprog a Set-UID user1 program (i.e., the owner is user1, which is another user account), export the LD_PRELOAD environment variable again in a different user's account (not-root user) and run it.



```
root@VM: /home/seed/Documents/Lab1
root@VM: /home/seed/Documents/Lab1# ./myprog.out
I am not sleeping!
root@VM: /home/seed/Documents/Lab1# id user1
id: 'user1': no such user
root@VM: /home/seed/Documents/Lab1# useradd user1
root@VM: /home/seed/Documents/Lab1# chown user1 myprog.out
root@VM: /home/seed/Documents/Lab1# ls -l myprog.out
-rwxr-xr-x 1 user1 seed 7348 Oct  9 21:57 myprog.out
root@VM: /home/seed/Documents/Lab1# su - user1
No directory, logging in with HOME=/
$ cd /home/seed/Documents/Lab1/
$ ls -l myprog.out
-rwxr-xr-x 1 user1 seed 7348 Oct  9 21:57 myprog.out
$ chmod 4755 myprog.out
$ whoami
user1
$ date
Wed Oct  9 22:12:32 EDT 2019
$ ls -l myprog.out
-rwsr-xr-x 1 user1 seed 7348 Oct  9 21:57 myprog.out
$ export LD_PRELOAD=./libmylib.so.1.0.1
$ ./myprog.out
I am not sleeping!
```

Observation: A user was added with useradd command and then modified the myprog.out ownership to user1 and then made the set-UID program. Also exported the LD_PRELOAD environment variable. When executed the program, the custom sleep was executed instead of regular sleep in libc and we can see the message from our program “I am not sleeping” was displayed.

Explanation: The program has both effective and real user id are same and the environment variable we set for LD_PRELOAD was effective and executes the custom library.

Step 3:

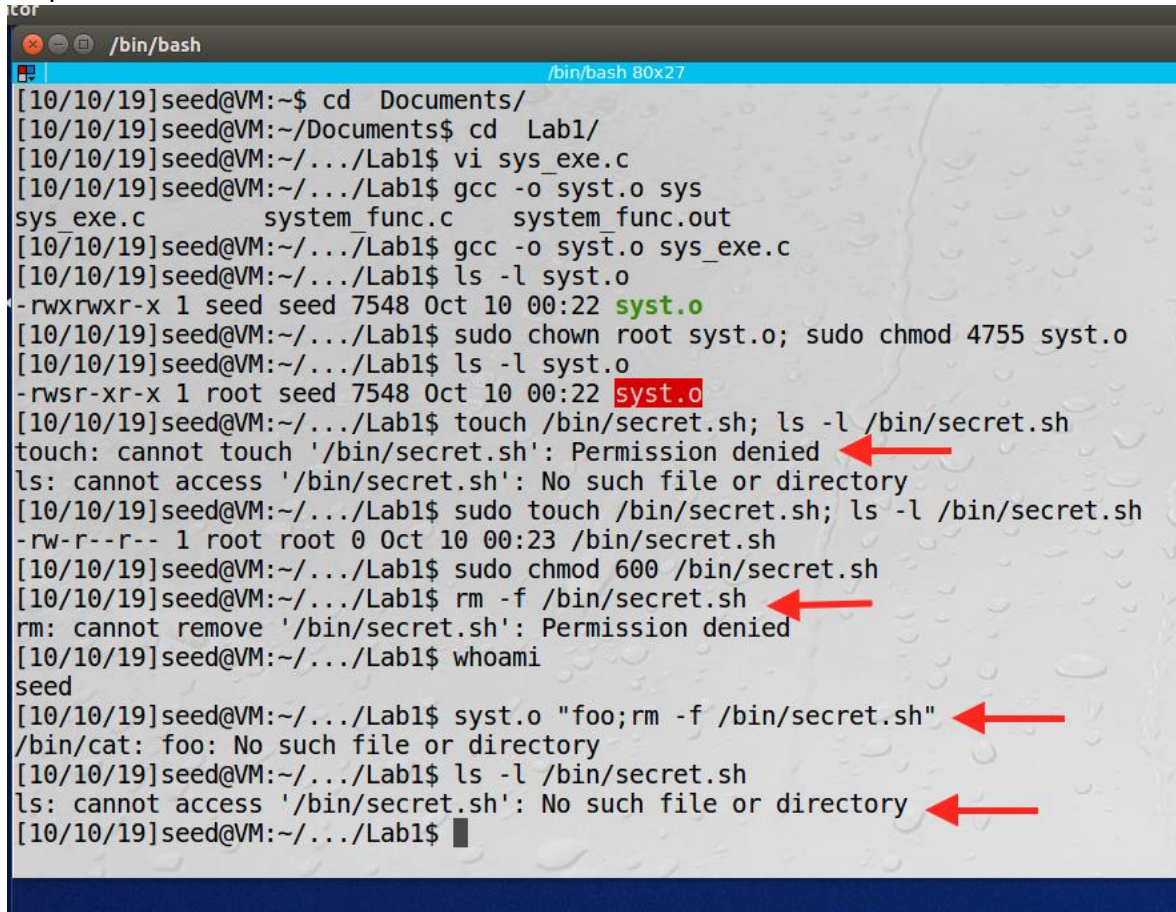
Overall task 7 explanation and conclusion:

From above four cases in the experiment, what I observed is the environment variables plays the key role here and especially for LD_PRELOAD.

For set-UID program, when running with a different real user id the custom LD_PRELOAD defined in the real user id environment will not effective. But when real and effective user ids are same, then the LD_PRELOAD defined will be effective. This is a security measure, so a normal user can not alter the environment when running a set-UID program.

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

Step 1:

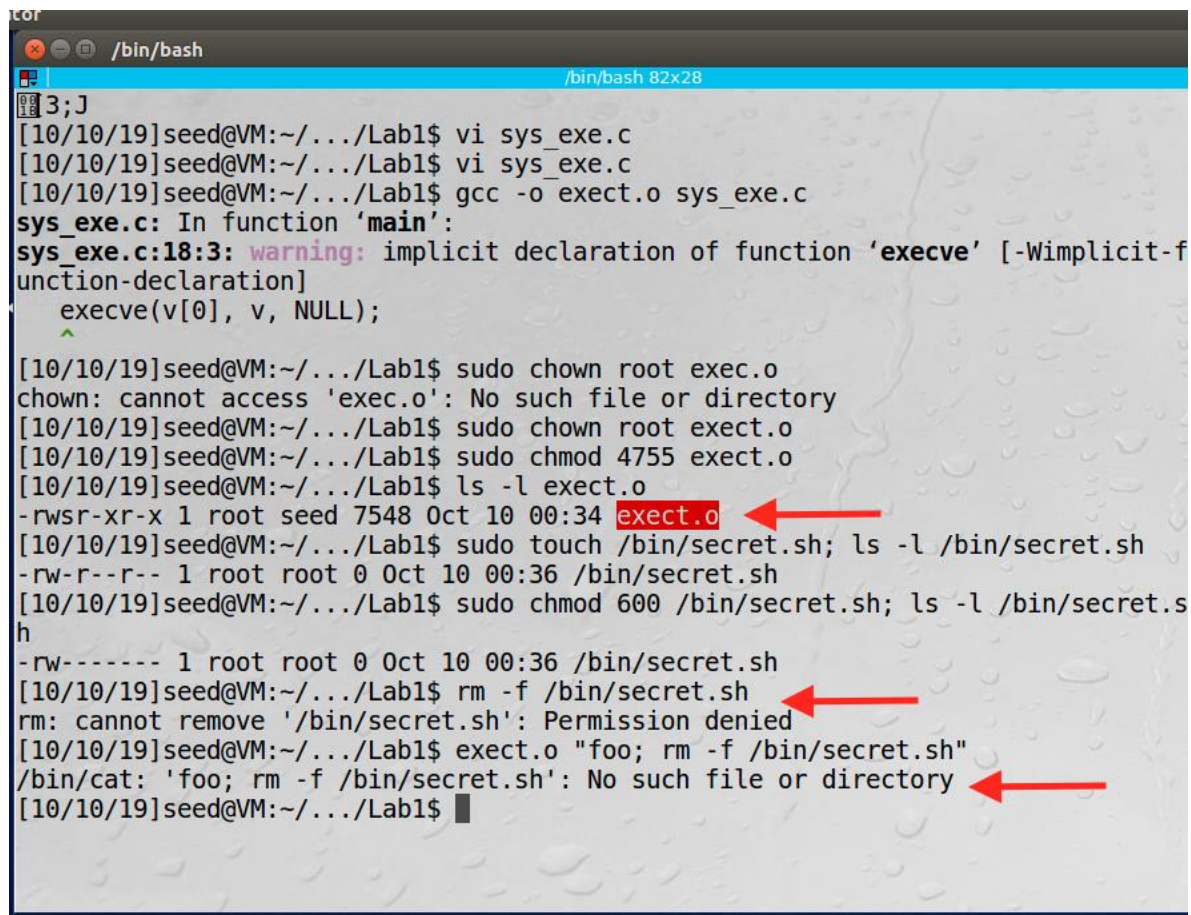


```
[10/10/19]seed@VM:~$ cd Documents/
[10/10/19]seed@VM:~/Documents$ cd Lab1/
[10/10/19]seed@VM:~/.../Lab1$ vi sys_exe.c
[10/10/19]seed@VM:~/.../Lab1$ gcc -o syst.o sys
sys_exe.c      system_func.c      system_func.out
[10/10/19]seed@VM:~/.../Lab1$ gcc -o syst.o sys_exe.c
[10/10/19]seed@VM:~/.../Lab1$ ls -l syst.o
-rwxrwxr-x 1 seed seed 7548 Oct 10 00:22 syst.o
[10/10/19]seed@VM:~/.../Lab1$ sudo chown root syst.o; sudo chmod 4755 syst.o
[10/10/19]seed@VM:~/.../Lab1$ ls -l syst.o
-rwsr-xr-x 1 root seed 7548 Oct 10 00:22 syst.o
[10/10/19]seed@VM:~/.../Lab1$ touch /bin/secret.sh; ls -l /bin/secret.sh
touch: cannot touch '/bin/secret.sh': Permission denied
ls: cannot access '/bin/secret.sh': No such file or directory
[10/10/19]seed@VM:~/.../Lab1$ sudo touch /bin/secret.sh; ls -l /bin/secret.sh
-rw-r--r-- 1 root root 0 Oct 10 00:23 /bin/secret.sh
[10/10/19]seed@VM:~/.../Lab1$ sudo chmod 600 /bin/secret.sh
[10/10/19]seed@VM:~/.../Lab1$ rm -f /bin/secret.sh
rm: cannot remove '/bin/secret.sh': Permission denied
[10/10/19]seed@VM:~/.../Lab1$ whoami
seed
[10/10/19]seed@VM:~/.../Lab1$ syst.o "foo;rm -f /bin/secret.sh"
/bin/cat: foo: No such file or directory
[10/10/19]seed@VM:~/.../Lab1$ ls -l /bin/secret.sh
ls: cannot access '/bin/secret.sh': No such file or directory
[10/10/19]seed@VM:~/.../Lab1$
```

Observation: Created the program and compiled to “syst.o” and made the executable as root owned set-UID program. Then created a secret file “/bin/secret.sh” with 600 permissions and verified that seed user cannot delete the file. When executed the program syst.o with the parameter “foo; rm -f /bin/secret.sh” ; then the secret.sh was deleted.

Explanation: From the above, as a Bob, the integrity was compromised by passing the malicious input as argument to a root set-UID program. With system() call, the command will be executed on shell and with a semi column, it will be treated the latter part as another command, where we passed the rm command and deleted a write protected file “/bin/secrets.sh” and as it a root owned set-UID program the file got deleted and integrity was compromised.

Step 2:



```
cor
/bin/bash
/bin/bash 82x28
3;J
[10/10/19]seed@VM:~/.../Lab1$ vi sys_exe.c
[10/10/19]seed@VM:~/.../Lab1$ vi sys_exe.c
[10/10/19]seed@VM:~/.../Lab1$ gcc -o exect.o sys_exe.c
sys_exe.c: In function 'main':
sys_exe.c:18:3: warning: implicit declaration of function 'execve' [-Wimplicit-f
unction-declaration]
    execve(v[0], v, NULL);
    ^
[10/10/19]seed@VM:~/.../Lab1$ sudo chown root exec.o
chown: cannot access 'exec.o': No such file or directory
[10/10/19]seed@VM:~/.../Lab1$ sudo chown root exect.o
[10/10/19]seed@VM:~/.../Lab1$ sudo chmod 4755 exect.o
[10/10/19]seed@VM:~/.../Lab1$ ls -l exect.o
-rwsr-xr-x 1 root seed 7548 Oct 10 00:34 exect.o
[10/10/19]seed@VM:~/.../Lab1$ sudo touch /bin/secret.sh; ls -l /bin/secret.sh
-rw-r--r-- 1 root root 0 Oct 10 00:36 /bin/secret.sh
[10/10/19]seed@VM:~/.../Lab1$ sudo chmod 600 /bin/secret.sh; ls -l /bin/secret.s
h
-rw----- 1 root root 0 Oct 10 00:36 /bin/secret.sh
[10/10/19]seed@VM:~/.../Lab1$ rm -f /bin/secret.sh
rm: cannot remove '/bin/secret.sh': Permission denied
[10/10/19]seed@VM:~/.../Lab1$ exect.o "foo; rm -f /bin/secret.sh"
/bin/cat: 'foo; rm -f /bin/secret.sh': No such file or directory
[10/10/19]seed@VM:~/.../Lab1$
```

Observation: Modified the same sys_exec.c by commenting the system command and uncommented the execve and created the executable as exect.o and made it as root owned set-UID program. This time also created a write protected file with “/bin/secret.sh” and verified the file removal by rm command and seed user was unable to delete it directly. But when the program was executed with exect.o with arguments as “foo; rm -f /bin/secret.sh” also the file was not removed and the message was cat didn’t find the file.

Explanation: The attack that was performed with system() did NOT worked this time. As the execve() does not invoke the shell, rather makes a system call with complete arguments. So, the whole argument was taken as one and passed to cat command. So the cat tried to open a file of name “foo; rm -f /bin/secret.sh”, which is not exist. So the attack was not worked in this case.

Task 9: Capability Leaking

```
/bin/bash
[10/11/19]seed@VM:~/.../Lab1$ pwd
/home/seed/Documents/Lab1
[10/11/19]seed@VM:~/.../Lab1$ vi capability_leak.c
[10/11/19]seed@VM:~/.../Lab1$ gcc -o capability_leak.o capability_leak.c
capability_leak.c: In function 'main':
capability_leak.c:17:1: warning: implicit declaration of function 'sleep' [-Wimplicit-function-declaration]
sleep(1);
^
capability_leak.c:21:1: warning: implicit declaration of function 'setuid' [-Wimplicit-function-declaration]
setuid(getuid()); /* getuid() returns the real uid */
^
capability_leak.c:21:8: warning: implicit declaration of function 'getuid' [-Wimplicit-function-declaration]
setuid(getuid()); /* getuid() returns the real uid */
^
capability_leak.c:22:5: warning: implicit declaration of function 'fork' [-Wimplicit-function-declaration]
if (fork()) { /* In the parent process */
^
capability_leak.c:23:1: warning: implicit declaration of function 'close' [-Wimplicit-function-declaration]
close (fd);
^
capability_leak.c:29:1: warning: implicit declaration of function 'write' [-Wimplicit-function-declaration]
write (fd, "Malicious Data\n", 15);
^
[10/11/19]seed@VM:~/.../Lab1$ ls -l capability_leak.o
-rwxrwxr-x 1 seed seed 7648 Oct 11 14:37 capability_leak.o
[10/11/19]seed@VM:~/.../Lab1$ sudo chown root capability_leak.o; sudo chmod 4755 capability_leak.o
[10/11/19]seed@VM:~/.../Lab1$ ls -l capability_leak.o
-rwsr-xr-x 1 root seed 7648 Oct 11 14:37 capability_leak.o
[10/11/19]seed@VM:~/.../Lab1$ whoami
seed
[10/11/19]seed@VM:~/.../Lab1$ capability_leak.o
Cannot open /etc/zzz
[10/11/19]seed@VM:~/.../Lab1$ sudo touch /etc/zzz
[10/11/19]seed@VM:~/.../Lab1$ capability_leak.o
[10/11/19]seed@VM:~/.../Lab1$ cat /etc/zzz
Malicious Data
[10/11/19]seed@VM:~/.../Lab1$
[10/11/19]seed@VM:~/.../Lab1$
[10/11/19]seed@VM:~/.../Lab1$
```

Observation:

The given code was compiled into “capability_leak.o” and then made it to root owned set-UID program with chown and chmod commands and also created an empty file /etc/zzz with touch command.

When the executable was ran with normal user (seed), the program was able to write the content in the root owned file “/etc/zzz” with the new line “Malicious Data”.

Explanation:

Though the program was executed with normal user and can see the capability of writing the content to /etc/zzz. Because the file descriptor was not closed before setting the real UID. So capability still continued and the attack was successful. Typically, before setting the real UID using setuid(getuid()) the file descriptor (fd) should be closed.