

CSE: 5382-001: SECURE PROGRAMMING

ASSIGNMENT 1

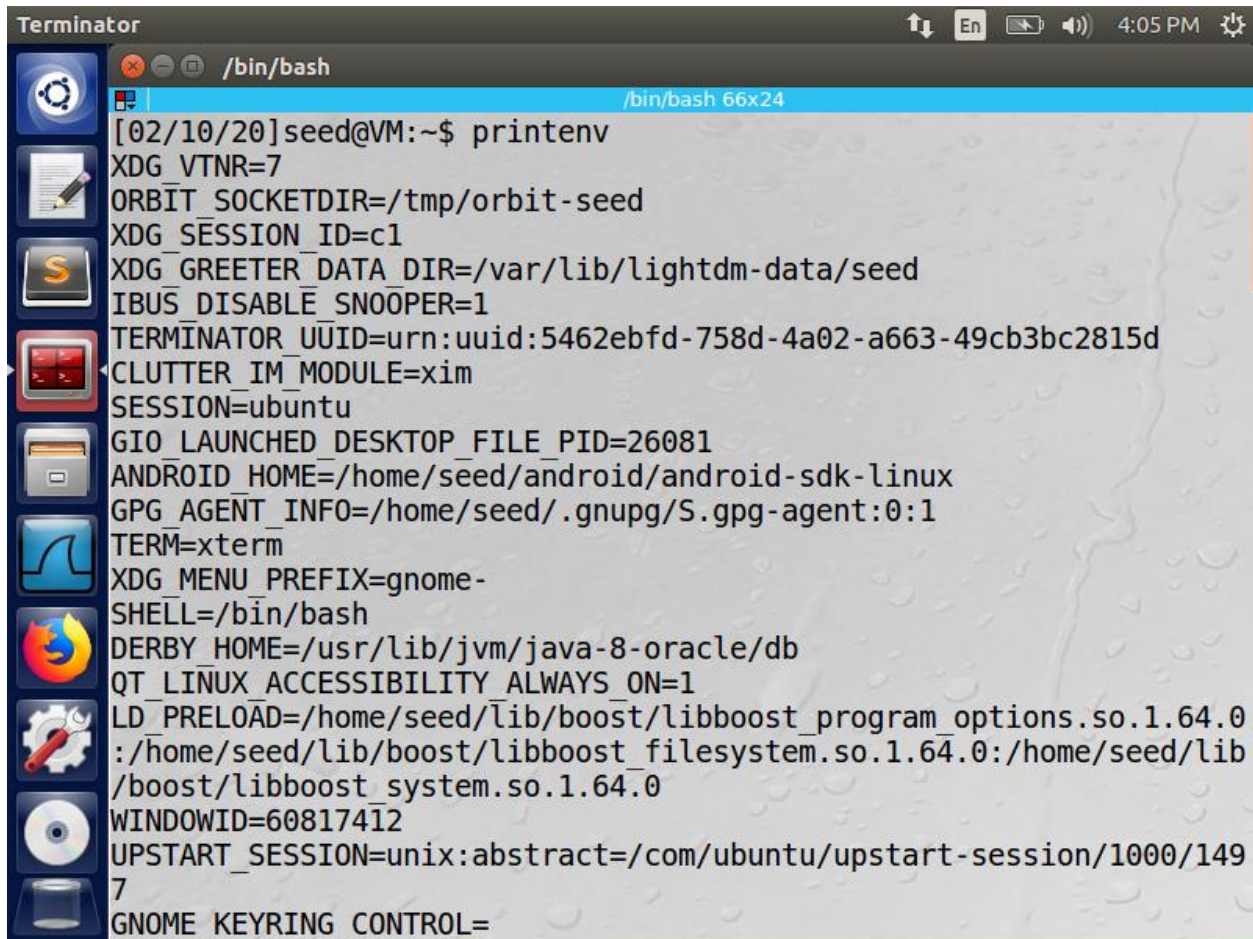
Tharoon T Thiagarajan
1001704601

Task 1: Manipulating Environment Variables

Step 1:

Output:

printenv or env command is used to print all environment variables.



The screenshot shows a Terminator terminal window with a dark theme. The title bar reads "Terminator" and the window title is "/bin/bash". The terminal content shows the command "printenv" being executed, which lists all environment variables. The output includes variables like XDG_VTNR, ORBIT_SOCKETDIR, XDG_SESSION_ID, XDG_GREETER_DATA_DIR, IBUS_DISABLE_SNOOPER, TERMINATOR_UUID, CLUTTER_IM_MODULE, SESSION, GIO_LAUNCHED_DESKTOP_FILE_PID, ANDROID_HOME, GPG_AGENT_INFO, TERM, XDG_MENU_PREFIX, SHELL, DERBY_HOME, QT_LINUX_ACCESSIBILITY_ALWAYS_ON, LD_PRELOAD, WINDOWID, UPSTART_SESSION, and GNOME_KEYRING_CONTROL.

```
Terminator /bin/bash
[02/10/20]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:5462ebfd-758d-4a02-a663-49cb3bc2815d
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=26081
ANDROID_HOME=/home/seed/android/android-sdk-linux
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm
XDG_MENU_PREFIX=gnome-
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/1000/149
7
GNOME_KEYRING_CONTROL=
```

Terminator

4:07 PM

```
/bin/bash
;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
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-r8d:/home/seed/.local/bin
QT_IM_MODULE=ibus
QT_QPA_PLATFORMTHEME=appmenu-qt5
XDG_SESSION_TYPE=x11
PWD=/home/seed
JOB=unity-settings-daemon
```

Terminator

/bin/bash

/bin/bash 66x24

GNOME_DESKTOP_SESSION_ID=this-is-deprecated
UPSTART_INSTANCE=
XDG_SESSION_DESKTOP=ubuntu
UPSTART_EVENTS=xsession started
LOGNAME=seed
COMPIZ_BIN_PATH=/usr/bin/
DBUS_SESSION_BUS_ADDRESS=unix:abstract=/tmp/dbus-cp92DdjWZn
J2SDKDIR=/usr/lib/jvm/java-8-oracle
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/share/gnome:/usr/local/share/
:/usr/share/./var/lib/snapd/desktop
QT4_IM_MODULE=xim
LESSOPEN=| /usr/bin/lesspipe %s
INSTANCE=
UPSTART_JOB=unity7
XDG_RUNTIME_DIR=/run/user/1000
DISPLAY=:0
XDG_CURRENT_DESKTOP=Unity
GTK_IM_MODULE=ibus
J2REDIR=/usr/lib/jvm/java-8-oracle/jre
LESSCLOSE=/usr/bin/lesspipe %s %s
XAUTHORITY=/home/seed/.Xauthority
COLORTERM=gnome-terminal
=/usr/bin/printenv
[02/10/20]seed@VM:~\$

Using grep:

By using grep, I am able to view the particular directory.

The screenshot displays a Kali Linux desktop environment. On the left side, there is a vertical sidebar containing several application icons: a gear (Settings), a document with a pencil (Text Editor), a folder with a dollar sign (File Manager), a terminal window (Terminal), a folder (File Manager), a web browser (Firefox), a gear with a wrench (System Tools), and a CD/DVD icon. The main area of the desktop is occupied by a terminal window titled "/bin/bash". The terminal shows the following commands and output:

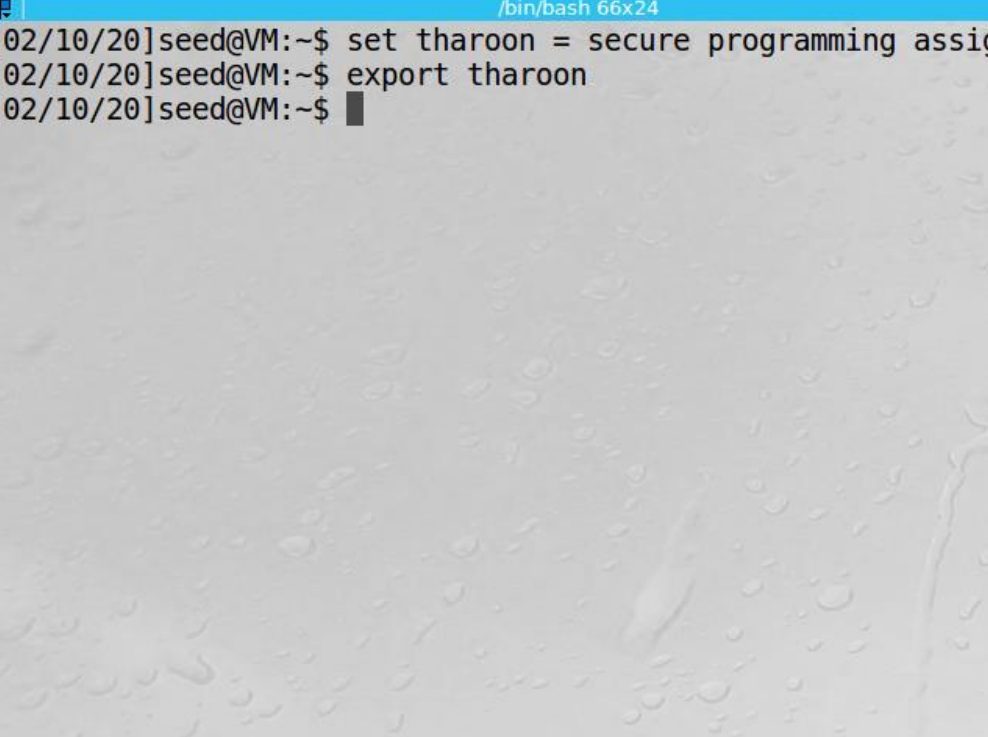
```
[02/10/20]seed@VM:~$ env | grep PWD
PWD=/home/seed
[02/10/20]seed@VM:~$
```

The terminal window has a blue title bar and a light blue background. The desktop background is a light gray with a subtle pattern of water droplets.

Step 2:

Output:

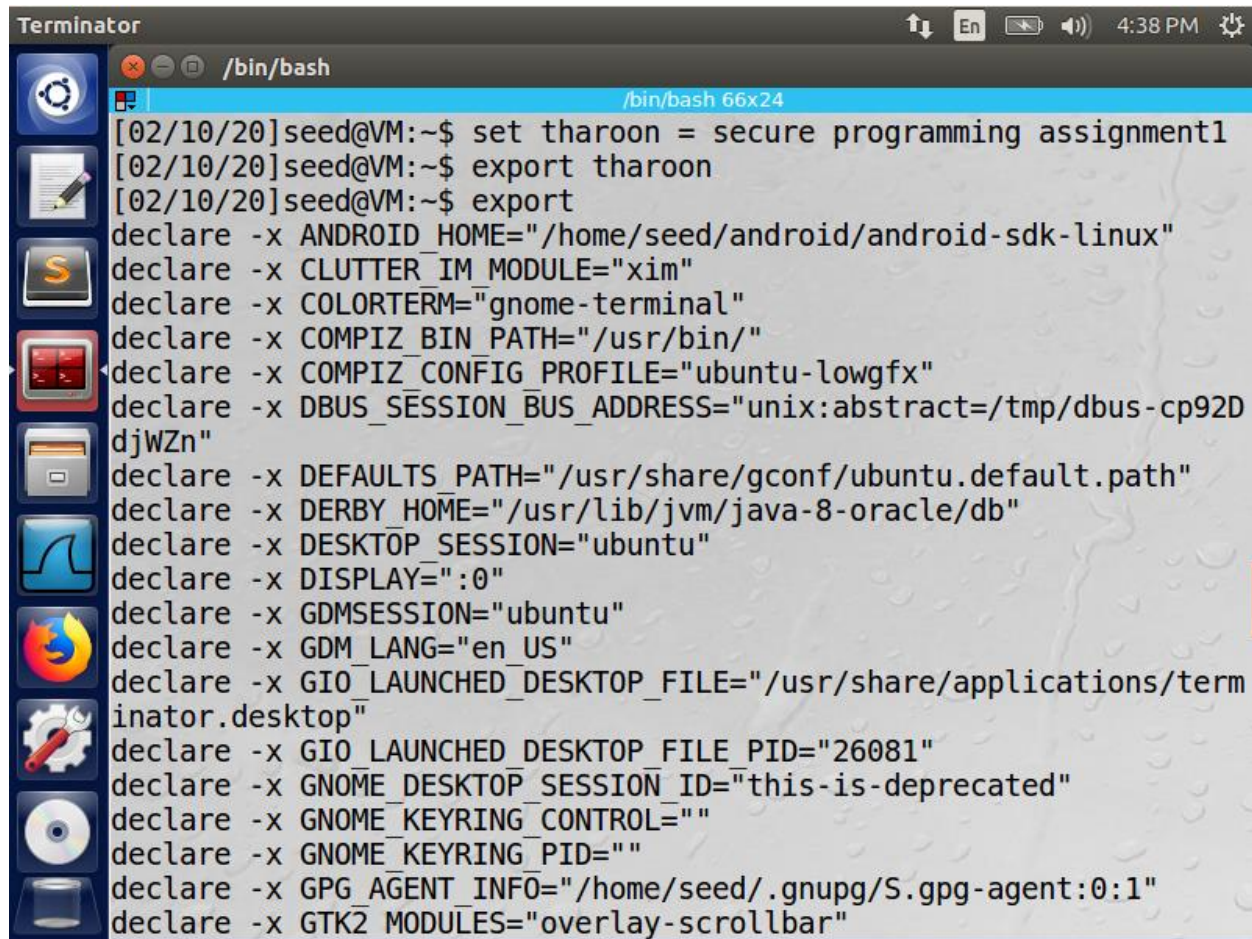
We use the set command to create an environment variable in the bash. Then, we use export command to set the newly created environment variable.



The screenshot shows a Kali Linux desktop environment. The desktop background is a dark, textured surface with water droplets. On the left side, there is a vertical dock containing several application icons: a blue gear icon, a document icon, a yellow 'S' icon, a red square icon, a folder icon, a blue square icon with a white 'A', a Firefox icon, a gear icon with a red wrench, a CD icon, and a USB icon. The top of the screen features a dark gray panel with the text 'Terminator' on the left and system status icons (up/down arrow, 'En', battery, volume, and clock) on the right. The clock shows '4:36 PM'. A terminal window is open in the center, titled '/bin/bash 66x24'. The terminal shows the following commands and output:

```
/bin/bash
[02/10/20]seed@VM:~$ set tharoon = secure programming assignment1
[02/10/20]seed@VM:~$ export tharoon
[02/10/20]seed@VM:~$
```



Now we use the export command to check if the newly created environment variable is set to the operating system. We will be able to see the newly created environment variable.


The image shows a screenshot of a Terminator terminal window. The title bar at the top reads "Terminator" and includes standard window controls and system status icons (network, battery, volume, and time 4:38 PM). The terminal's title bar shows the current directory as "/bin/bash" and the window size as "66x24". The terminal content shows a series of commands entered by a user named "seed" at a prompt "seed@VM:~\$". The commands are: "set tharoon = secure programming assignment1", "export tharoon", and "export". This is followed by a list of "declare -x" commands for various system environment variables, including ANDROID_HOME, CLUTTER_IM_MODULE, COLORTERM, COMPIZ_BIN_PATH, COMPIZ_CONFIG_PROFILE, DBUS_SESSION_BUS_ADDRESS, DEFAULTS_PATH, DERBY_HOME, DESKTOP_SESSION, DISPLAY, GDMSESSION, GDM_LANG, GIO_LAUNCHED_DESKTOP_FILE, GIO_LAUNCHED_DESKTOP_FILE_PID, GNOME_DESKTOP_SESSION_ID, GNOME_KEYRING_CONTROL, GNOME_KEYRING_PID, GPG_AGENT_INFO, and GTK2_MODULES. A vertical sidebar on the left contains icons for various applications, with the terminal icon currently selected.

```
Terminator /bin/bash
[02/10/20]seed@VM:~$ set tharoon = secure programming assignment1
[02/10/20]seed@VM:~$ export tharoon
[02/10/20]seed@VM:~$ export
declare -x ANDROID_HOME="/home/seed/android/android-sdk-linux"
declare -x CLUTTER_IM_MODULE="xim"
declare -x COLORTERM="gnome-terminal"
declare -x COMPIZ_BIN_PATH="/usr/bin/"
declare -x COMPIZ_CONFIG_PROFILE="ubuntu-lowgfx"
declare -x DBUS_SESSION_BUS_ADDRESS="unix:abstract=/tmp/dbus-cp92D
djWZn"
declare -x DEFAULTS_PATH="/usr/share/gconf/ubuntu.default.path"
declare -x DERBY_HOME="/usr/lib/jvm/java-8-oracle/db"
declare -x DESKTOP_SESSION="ubuntu"
declare -x DISPLAY=":0"
declare -x GDMSESSION="ubuntu"
declare -x GDM_LANG="en_US"
declare -x GIO_LAUNCHED_DESKTOP_FILE="/usr/share/applications/term
inator.desktop"
declare -x GIO_LAUNCHED_DESKTOP_FILE_PID="26081"
declare -x GNOME_DESKTOP_SESSION_ID="this-is-deprecated"
declare -x GNOME_KEYRING_CONTROL=""
declare -x GNOME_KEYRING_PID=""
declare -x GPG_AGENT_INFO="/home/seed/.gnupg/S.gpg-agent:0:1"
declare -x GTK2_MODULES="overlay-scrollbar"
```

Terminator

↑ ↓ En 🔊 4:38 PM ⚙

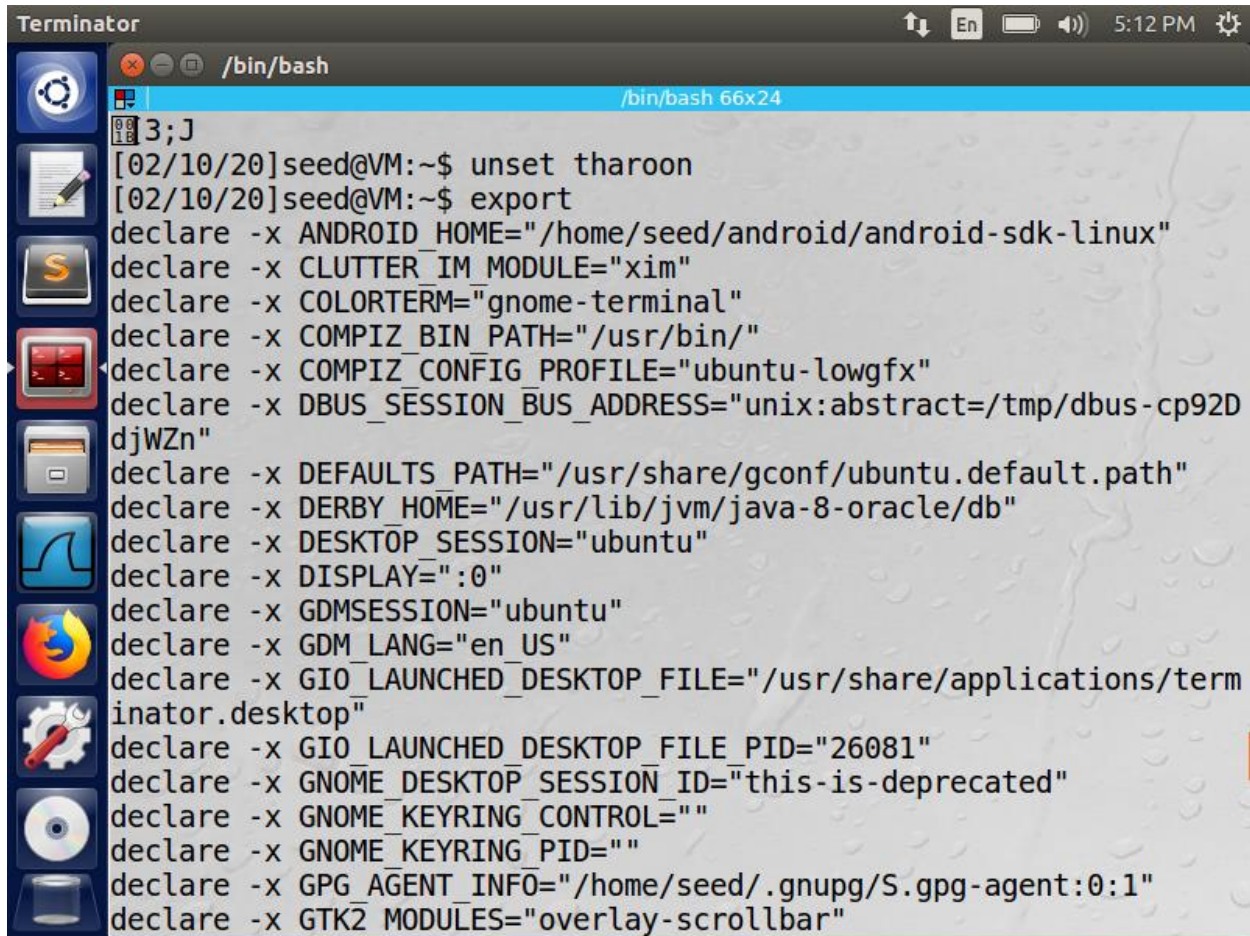
  /bin/bash

 /bin/bash 66x24

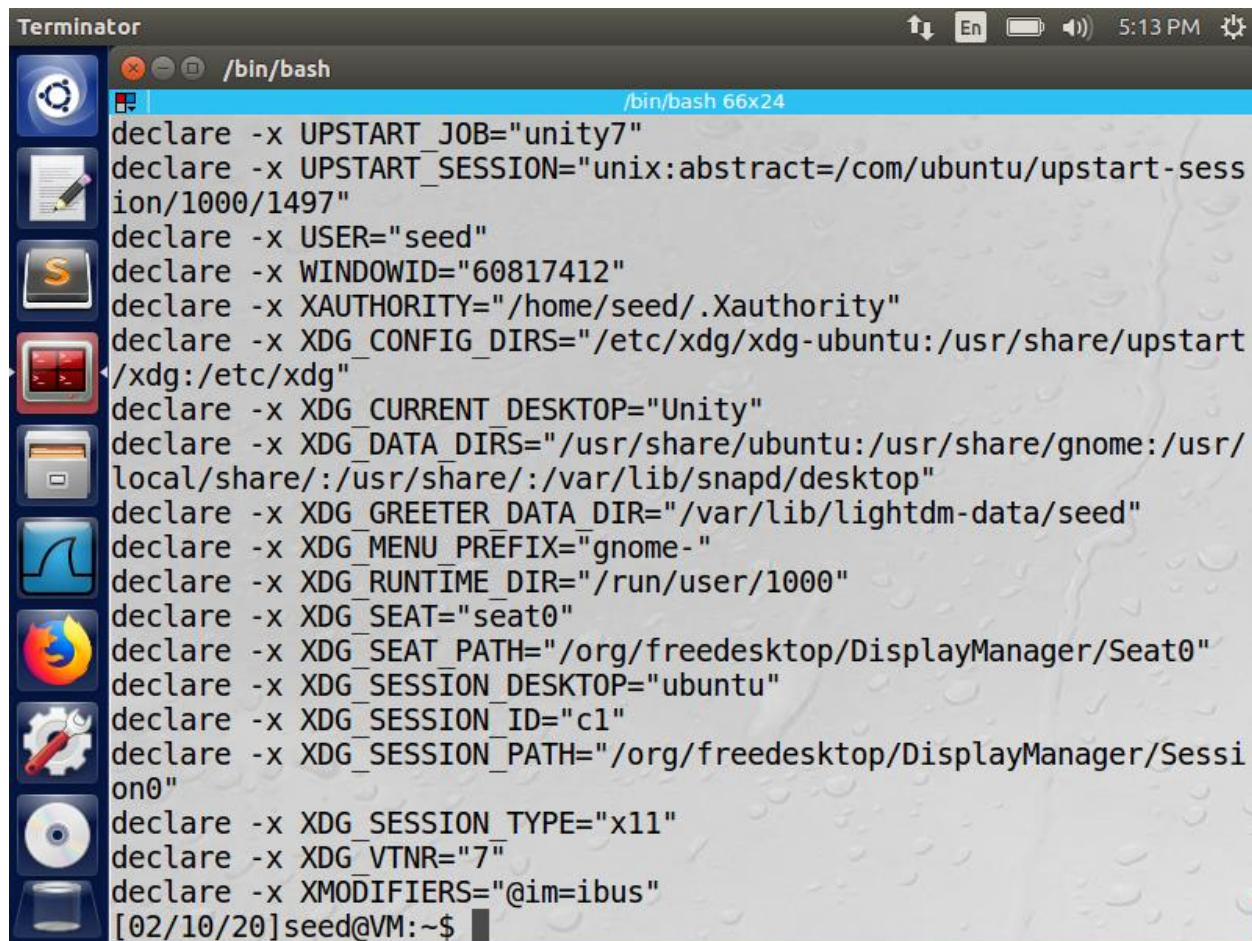
```
declare -x UPSTART_SESSION="unix:abstract=/com/ubuntu/upstart-session/1000/1497"
declare -x USER="seed"
declare -x WINDOWID="60817412"
declare -x XAUTHORITY="/home/seed/.Xauthority"
declare -x XDG_CONFIG_DIRS="/etc/xdg/xdg-ubuntu:/usr/share/upstart/xdg:/etc/xdg"
declare -x XDG_CURRENT_DESKTOP="Unity"
declare -x XDG_DATA_DIRS="/usr/share/ubuntu:/usr/share/gnome:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop"
declare -x XDG_GREETER_DATA_DIR="/var/lib/lightdm-data/seed"
declare -x XDG_MENU_PREFIX="gnome-"
declare -x XDG_RUNTIME_DIR="/run/user/1000"
declare -x XDG_SEAT="seat0"
declare -x XDG_SEAT_PATH="/org/freedesktop/DisplayManager/Seat0"
declare -x XDG_SESSION_DESKTOP="ubuntu"
declare -x XDG_SESSION_ID="c1"
declare -x XDG_SESSION_PATH="/org/freedesktop/DisplayManager/Session0"
declare -x XDG_SESSION_TYPE="x11"
declare -x XDG_VTNR="7"
declare -x XMODIFIERS="@im=ibus"
declare -x tharoon
```

[02/10/20]seed@VM:~\$

Now we will unset the environment variable which we created in the above step, using the unset command. Now if we use export command to look for environment variable, we will not be able to see the created environment variable.



```
Terminator
/bin/bash
[02/10/20]seed@VM:~$ unset tharoon
[02/10/20]seed@VM:~$ export
declare -x ANDROID_HOME="/home/seed/android/android-sdk-linux"
declare -x CLUTTER_IM_MODULE="xim"
declare -x COLORTERM="gnome-terminal"
declare -x COMPIZ_BIN_PATH="/usr/bin/"
declare -x COMPIZ_CONFIG_PROFILE="ubuntu-lowgfx"
declare -x DBUS_SESSION_BUS_ADDRESS="unix:abstract=/tmp/dbus-cp92D
djWZn"
declare -x DEFAULTS_PATH="/usr/share/gconf/ubuntu.default.path"
declare -x DERBY_HOME="/usr/lib/jvm/java-8-oracle/db"
declare -x DESKTOP_SESSION="ubuntu"
declare -x DISPLAY=":0"
declare -x GDMSESSION="ubuntu"
declare -x GDM_LANG="en_US"
declare -x GIO_LAUNCHED_DESKTOP_FILE="/usr/share/applications/term
inator.desktop"
declare -x GIO_LAUNCHED_DESKTOP_FILE_PID="26081"
declare -x GNOME_DESKTOP_SESSION_ID="this-is-deprecated"
declare -x GNOME_KEYRING_CONTROL=""
declare -x GNOME_KEYRING_PID=""
declare -x GPG_AGENT_INFO="/home/seed/.gnupg/S.gpg-agent:0:1"
declare -x GTK2_MODULES="overlay-scrollbar"
```

A screenshot of a Terminator terminal window. The title bar shows 'Terminator' and system icons. The terminal has a blue header bar with '/bin/bash' and a window title bar with '/bin/bash 66x24'. The main area contains a list of 'declare -x' commands for environment variables. On the left, there is a vertical sidebar with icons for various applications like a gear, notepad, terminal, and others.

```
declare -x UPSTART_JOB="unity7"
declare -x UPSTART_SESSION="unix:abstract=/com/ubuntu/upstart-session/1000/1497"
declare -x USER="seed"
declare -x WINDOWID="60817412"
declare -x XAUTHORITY="/home/seed/.Xauthority"
declare -x XDG_CONFIG_DIRS="/etc/xdg/xdg-ubuntu:/usr/share/upstart/xdg:/etc/xdg"
declare -x XDG_CURRENT_DESKTOP="Unity"
declare -x XDG_DATA_DIRS="/usr/share/ubuntu:/usr/share/gnome:/usr/local/share/:/usr/share:/var/lib/snapd/desktop"
declare -x XDG_GREETER_DATA_DIR="/var/lib/lightdm-data/seed"
declare -x XDG_MENU_PREFIX="gnome-"
declare -x XDG_RUNTIME_DIR="/run/user/1000"
declare -x XDG_SEAT="seat0"
declare -x XDG_SEAT_PATH="/org/freedesktop/DisplayManager/Seat0"
declare -x XDG_SESSION_DESKTOP="ubuntu"
declare -x XDG_SESSION_ID="c1"
declare -x XDG_SESSION_PATH="/org/freedesktop/DisplayManager/Session0"
declare -x XDG_SESSION_TYPE="x11"
declare -x XDG_VTNR="7"
declare -x XMODIFIERS="@im=ibus"
[02/10/20]seed@VM:~$
```

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

Step 1:

Output:

I run the task2 file which contains the program with uncommented `printenv()` method in the child process, and the `printenv()` method in the parent process is commented. Now, I compile and run the task2 file and save it in a text file task2.txt. I use cat command to view the contents of the text file task2.txt. I am able to see list of all environment variables being displayed.

Terminator

↑ ↓ En 🔋 🔊 9:23 PM ⚙

/bin/bash

/bin/bash 66x24

```
[02/10/20]seed@VM:~$ vi task2.c
[02/10/20]seed@VM:~$ gcc -Wall task2.c -o task2
task2.c:13:6: warning: return type of 'main' is not 'int' [-Wmain]
void main()
    ^
[02/10/20]seed@VM:~$ ./task2 > task2.txt
[02/10/20]seed@VM:~$ cat task2.txt
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:d012f22b-9364-4911-88c8-082cf67baf34
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=27239
ANDROID_HOME=/home/seed/android/android-sdk-linux
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm
XDG_MENU_PREFIX=gnome-
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
```


Terminator

↑ ↓ En 9:24 PM

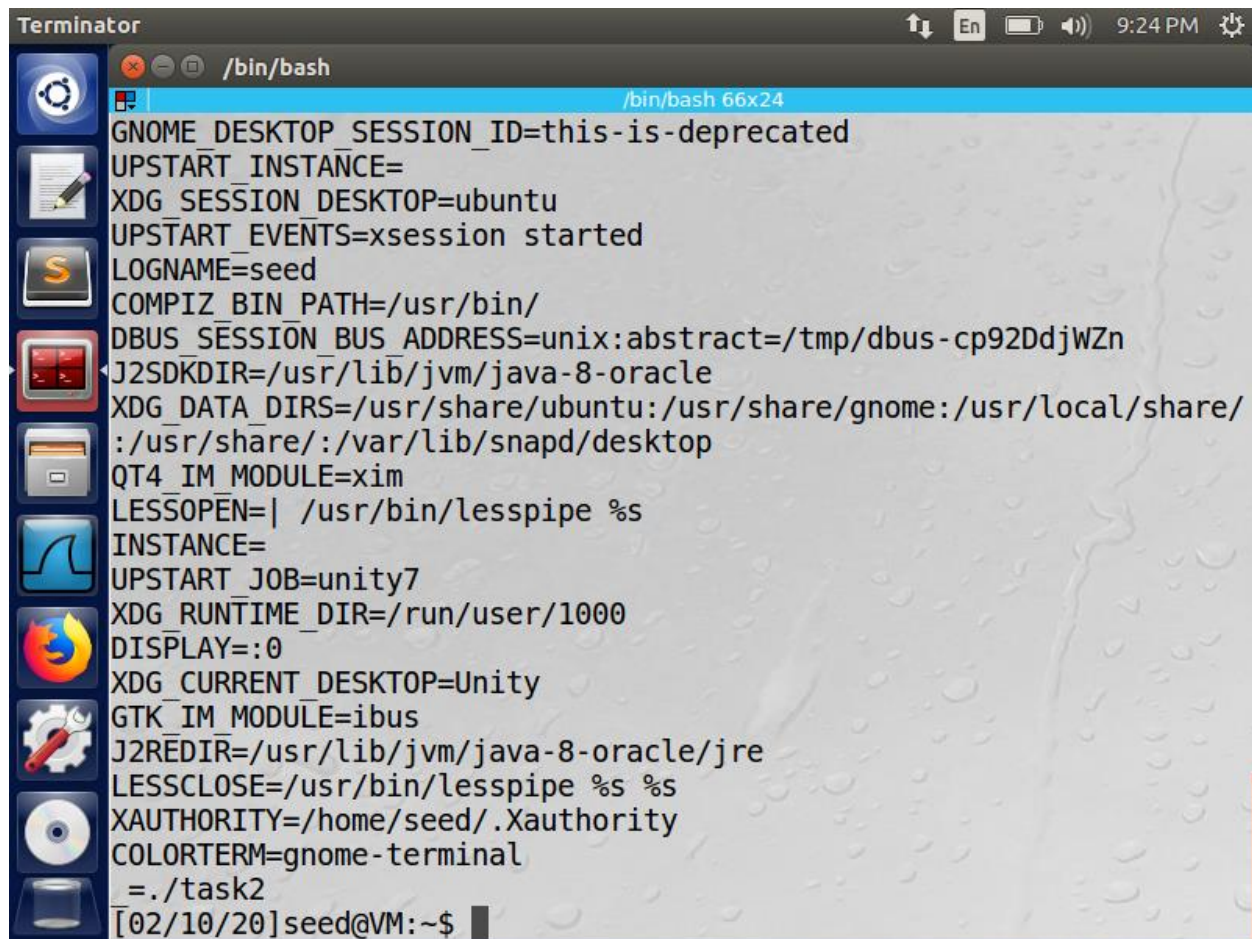
/bin/bash

/bin/bash 66x24

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:*.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:*.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:

A screenshot of a Terminator terminal window. The title bar shows 'Terminator' and system icons. The terminal has a blue header bar with '/bin/bash' and a window icon. The main area displays a list of environment variables in a monospaced font. The background has a faint, textured pattern. The prompt at the bottom is '[02/10/20] seed@VM: ~\$'.

```
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
UPSTART_INSTANCE=
XDG_SESSION_DESKTOP=ubuntu
UPSTART_EVENTS=xsession started
LOGNAME=seed
COMPIZ_BIN_PATH=/usr/bin/
DBUS_SESSION_BUS_ADDRESS=unix:abstract=/tmp/dbus-cp92DdjWZn
J2SDKDIR=/usr/lib/jvm/java-8-oracle
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/share/gnome:/usr/local/share/
:/usr/share/;/var/lib/snapd/desktop
QT4_IM_MODULE=xim
LESSOPEN=| /usr/bin/lesspipe %s
INSTANCE=
UPSTART_JOB=unity7
XDG_RUNTIME_DIR=/run/user/1000
DISPLAY=:0
XDG_CURRENT_DESKTOP=Unity
GTK_IM_MODULE=ibus
J2REDIR=/usr/lib/jvm/java-8-oracle/jre
LESSCLOSE=/usr/bin/lesspipe %s %s
XAUTHORITY=/home/seed/.Xauthority
COLORTERM=gnome-terminal
=./task2
[02/10/20] seed@VM: ~$
```

Step 2:

Output:

I run the task2Parent file which contains the program with uncommented `printenv()` method in the parent process, and the `printenv()` method in the child process is commented. Now, I compile and run the task2Parent file and saved it in a text file task2Parent.txt. I use `cat` command to view the contents of the text file task2Parent.txt. I am able to see list of all environment variables being displayed.

Terminator

/bin/bash

/bin/bash 66x24

```
[02/10/20]seed@VM:~$ vi task2Parent.c
[02/10/20]seed@VM:~$ gcc -Wall task2Parent.c -o task2Parent
task2Parent.c:13:6: warning: return type of 'main' is not 'int' [-Wmain]
void main()
    ^
[02/10/20]seed@VM:~$ ./task2Parent > task2Parent.txt
[02/10/20]seed@VM:~$ cat task2Parent.txt
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:195ad1cf-dc3f-43b4-9172-e76cd51f7bfd
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=27614
ANDROID_HOME=/home/seed/android/android-sdk-linux
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm
XDG_MENU_PREFIX=gnome-
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1
```

Terminator

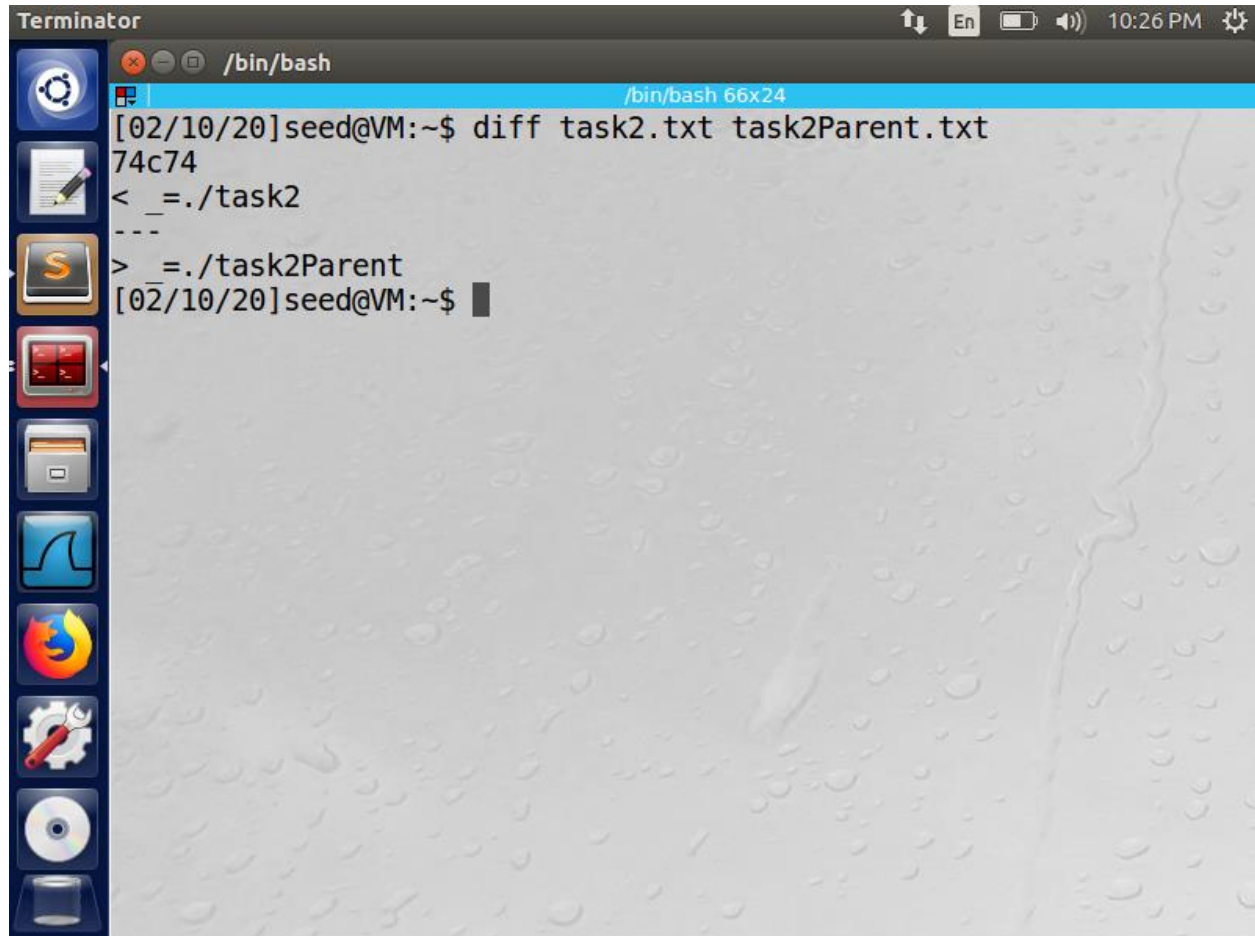
/bin/bash

/bin/bash 66x24

GNOME_DESKTOP_SESSION_ID=this-is-deprecated
UPSTART_INSTANCE=
XDG_SESSION_DESKTOP=ubuntu
UPSTART_EVENTS=xsession started
LOGNAME=seed
COMPIZ_BIN_PATH=/usr/bin/
DBUS_SESSION_BUS_ADDRESS=unix:abstract=/tmp/dbus-cp92DdjWZn
J2SDKDIR=/usr/lib/jvm/java-8-oracle
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/share/gnome:/usr/local/share/
:/usr/share/./var/lib/snapd/desktop
QT4_IM_MODULE=xim
LESSOPEN=| /usr/bin/lesspipe %s
INSTANCE=
UPSTART_JOB=unity7
XDG_RUNTIME_DIR=/run/user/1000
DISPLAY=:0
XDG_CURRENT_DESKTOP=Unity
GTK_IM_MODULE=ibus
J2REDIR=/usr/lib/jvm/java-8-oracle/jre
LESSCLOSE=/usr/bin/lesspipe %s %s
XAUTHORITY=/home/seed/.Xauthority
COLORTERM=gnome-terminal
=./task2Parent
[02/10/20] seed@VM:~\$

Step 3:

I used the diff command to compare the two text files task2.txt and task2Parent.txt. I am able to see that there is no any differences between the two text files except for the file name which results in 74c74. I conclude that the child process inherits the environment variable of the parent process.



The image shows a screenshot of a Terminator terminal window. The title bar at the top reads "Terminator" and includes standard window controls and system status icons (network, volume, battery, and time 10:26 PM). The terminal window has a dark background with a light blue title bar that says "/bin/bash 66x24". The terminal content shows a user prompt "[02/10/20]seed@VM:~\$" followed by the command "diff task2.txt task2Parent.txt". The output of the command is "74c74", indicating a single line difference. Below this, the terminal shows the contents of the two files being compared: "< _=./task2" and "> _=./task2Parent". The prompt returns to "[02/10/20]seed@VM:~\$". On the left side of the terminal window, there is a vertical dock containing several application icons: a gear (settings), a document with a pencil (editor), a dollar sign (terminal), a red square (file manager), a folder (file manager), a graph (plotter), a Firefox logo (web browser), a wrench and gear (system tools), a CD (media), and a laptop (terminal).

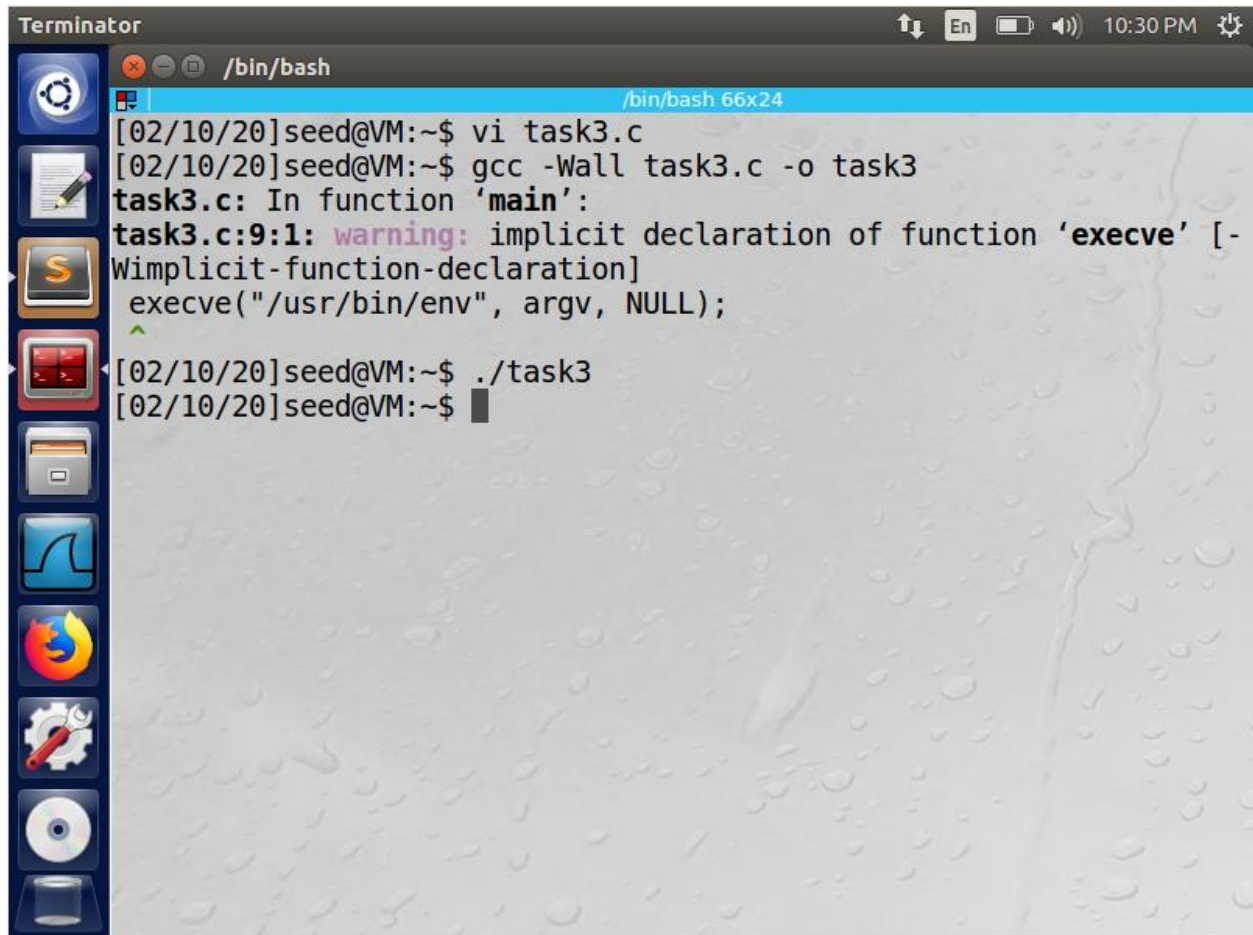
```
Terminator
/bin/bash
[02/10/20]seed@VM:~$ diff task2.txt task2Parent.txt
74c74
< _=./task2
---
> _=./task2Parent
[02/10/20]seed@VM:~$
```


Task 3: Environment Variables and `execve()`

Step 1:

Output:

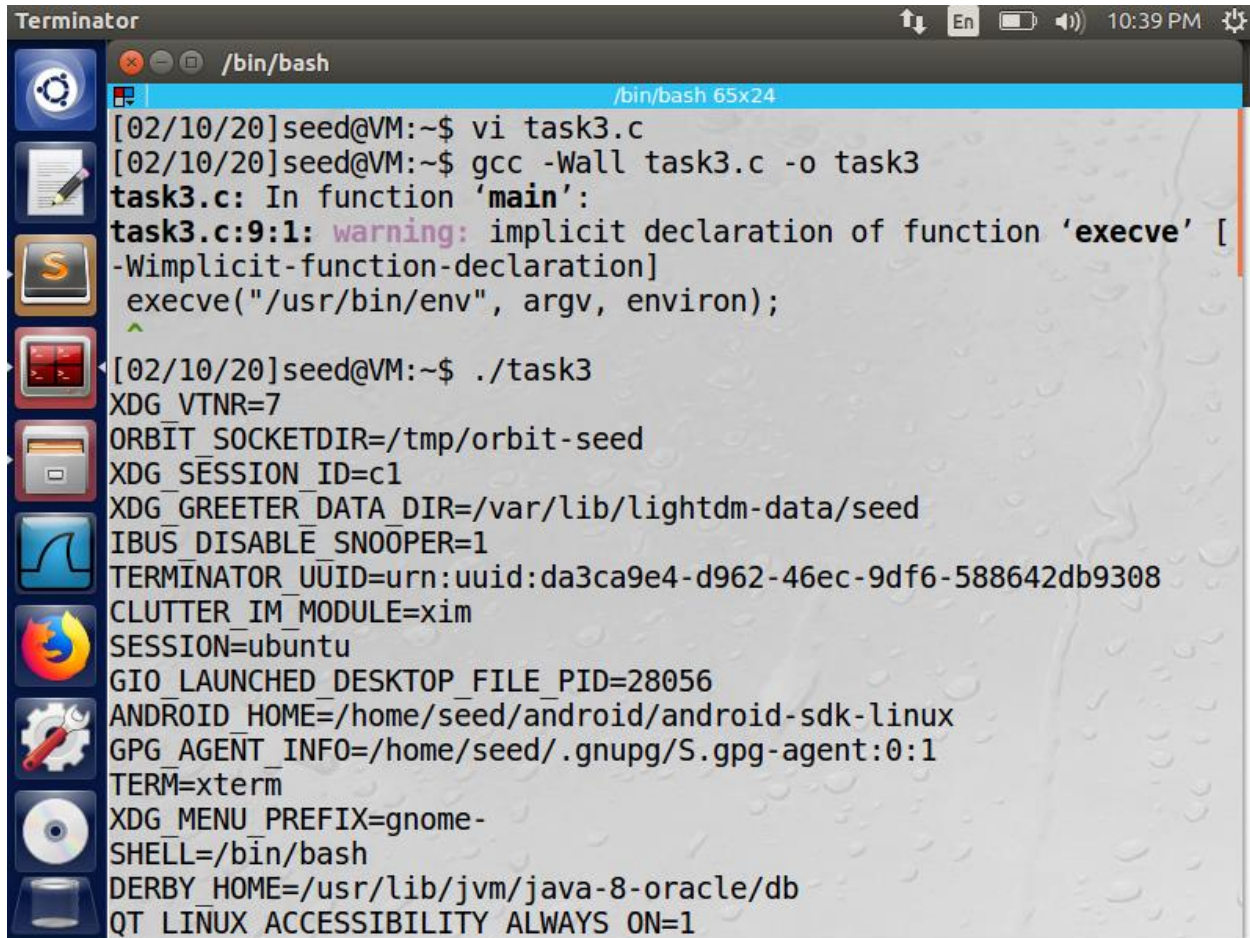
When I compile and run the given program, I am able to see that nothing gets executed. This is because NULL is passed as the third parameter to the `execve()` method, which results in empty output. This is because the environment variable is not stored in the memory.



```
Terminator /bin/bash
[02/10/20]seed@VM:~$ vi task3.c
[02/10/20]seed@VM:~$ gcc -Wall task3.c -o task3
task3.c: In function 'main':
task3.c:9:1: warning: implicit declaration of function 'execve' [-Wimplicit-function-declaration]
  execve("/usr/bin/env", argv, NULL);
  ^
[02/10/20]seed@VM:~$ ./task3
[02/10/20]seed@VM:~$
```

Step 2:

When I pass the third parameter from NULL to environ, after compiling and running the program I am able to see the list of environment variables displayed. This is because the environment variables are loaded into the memory.



```
Terminator
/bin/bash
[02/10/20]seed@VM:~$ vi task3.c
[02/10/20]seed@VM:~$ gcc -Wall task3.c -o task3
task3.c: In function 'main':
task3.c:9:1: warning: implicit declaration of function 'execve' [-Wimplicit-function-declaration]
  execve("/usr/bin/env", argv, environ);
  ^
[02/10/20]seed@VM:~$ ./task3
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:da3ca9e4-d962-46ec-9df6-588642db9308
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=28056
ANDROID_HOME=/home/seed/android/android-sdk-linux
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm
XDG_MENU_PREFIX=gnome-
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1
```

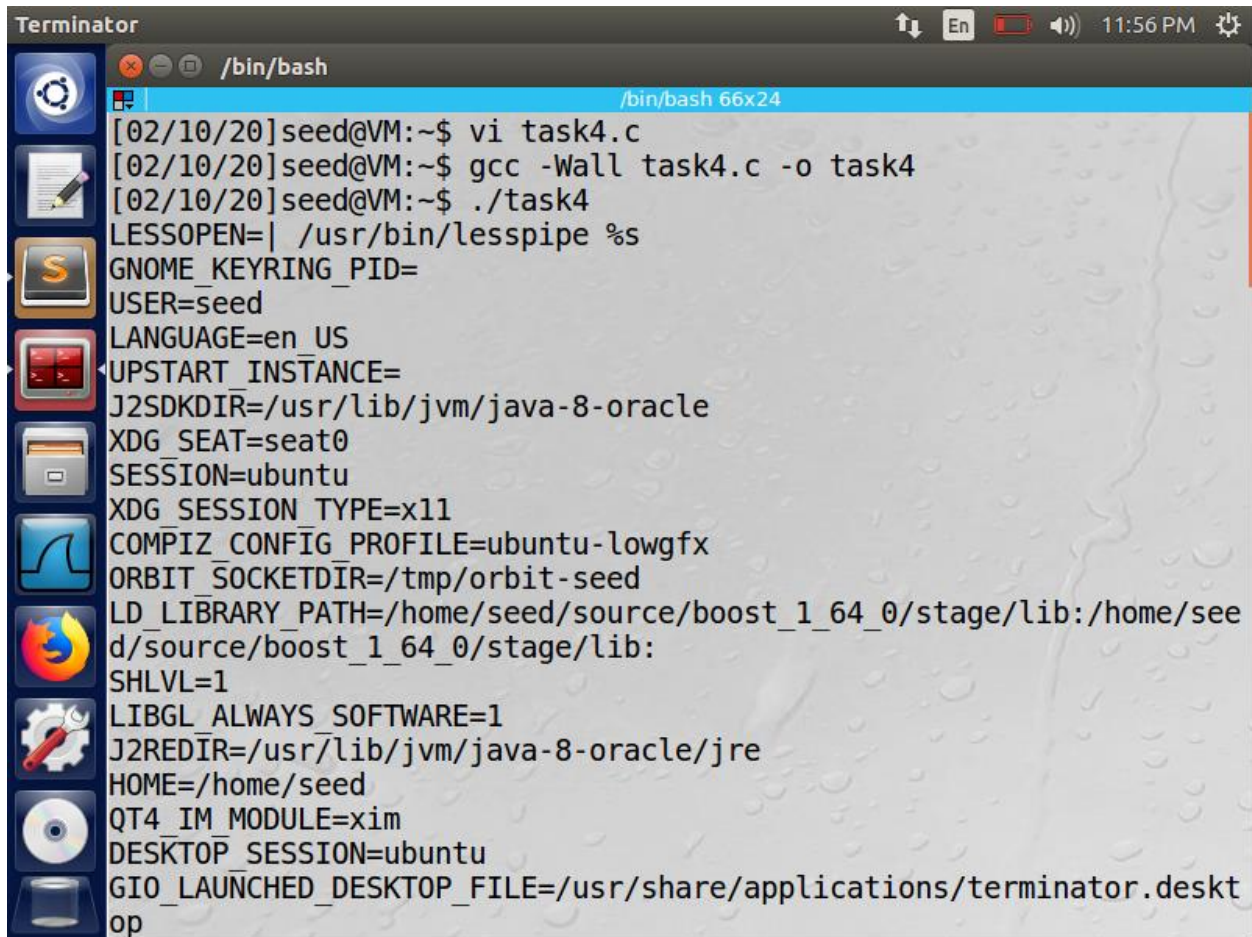
Step 3:

The reason that the new program is able to print the environment variables is because the environ variable is passed as the third parameter to the execve() method which prints the list of environment variables.

Task 4: Environment Variables and system()

Output:

I can see that environmental variable is passed when executed through system("/usr/bin/env"). Basically, system() does not execute the command directly and when I use this system("/usr/bin/env"), environmental variables are passed that are getting displayed and hence verified.

A screenshot of a Terminator terminal window. The title bar says "Terminator" and the window title is "/bin/bash". The terminal content shows a series of commands and their outputs. The user runs 'vi task4.c', 'gcc -Wall task4.c -o task4', and './task4'. The output of './task4' is a list of environment variables: 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, DESKTOP_SESSION=ubuntu, and GIO_LAUNCHED_DESKTOP_FILE=/usr/share/applications/terminator.desktop. The terminal has a dark background with a light blue header bar and a light gray background with a subtle water droplet pattern.

```
Terminator /bin/bash
[02/10/20]seed@VM:~$ vi task4.c
[02/10/20]seed@VM:~$ gcc -Wall task4.c -o task4
[02/10/20]seed@VM:~$ ./task4
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
DESKTOP_SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE=/usr/share/applications/terminator.desktop
```


Terminator

En 11:56 PM

/bin/bash

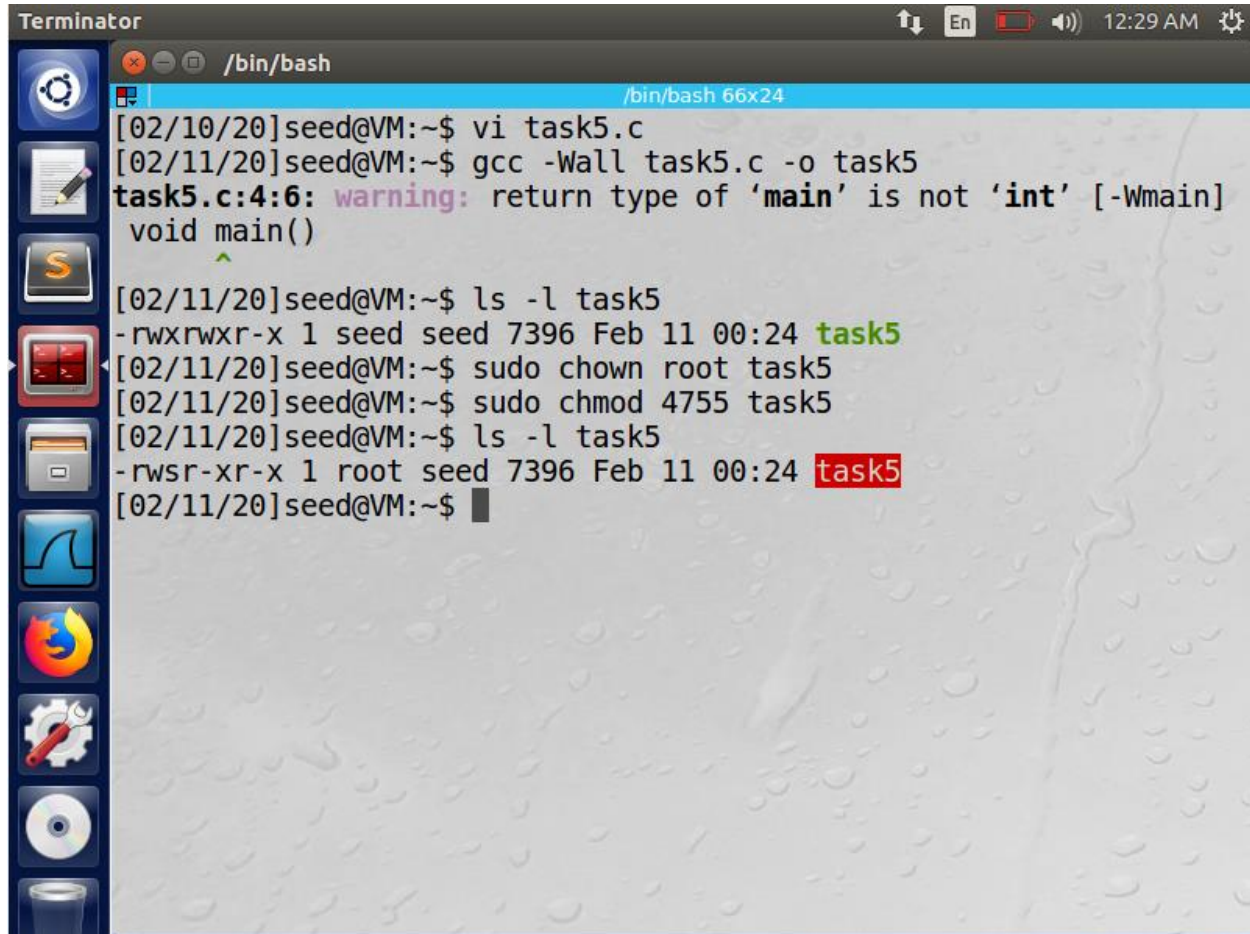
/bin/bash 66x24

XAUTHORITY=/home/seed/.Xauthority
XDG_GREETER_DATA_DIR=/var/lib/lightdm-data/seed
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
TERMINATOR_UUID=urn:uuid:eda1ca6e-d97e-4585-979a-244ef779115d
SHELL=/bin/bash
QT_ACCESSIBILITY=1
GDMSESSION=ubuntu
LESSCLOSE=/usr/bin/lesspipe %s %s
UPSTART_EVENTS=xsession started
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-session/1000/1497
XDG_VTNR=7
QT_IM_MODULE=ibus
PWD=/home/seed
JAVA_HOME=/usr/lib/jvm/java-8-oracle
CLUTTER_IM_MODULE=xim
ANDROID_HOME=/home/seed/android/android-sdk-linux
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/usr/share/upstart/xdg:/etc/xdg
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/share/gnome:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
JOB=unity-settings-daemon
[02/10/20]seed@VM:~\$

Task 5: Environment Variable and Set-UID Programs

Step 1 and Step 2:

Compiling the given program and changing its owner to root and making it a SET-UID program.

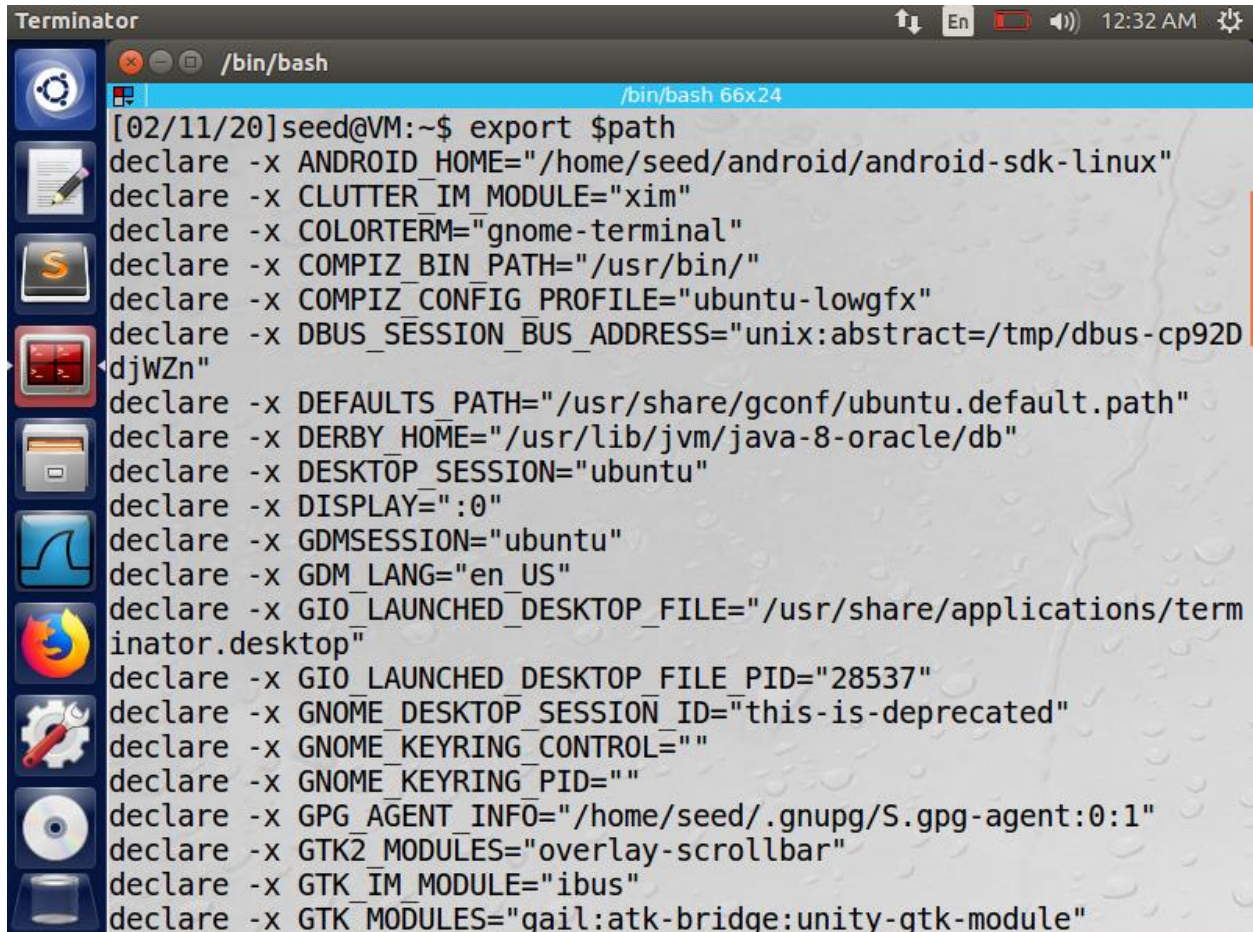


```
Terminator /bin/bash
[02/10/20]seed@VM:~$ vi task5.c
[02/11/20]seed@VM:~$ gcc -Wall task5.c -o task5
task5.c:4:6: warning: return type of 'main' is not 'int' [-Wmain]
void main()
    ^
[02/11/20]seed@VM:~$ ls -l task5
-rwxrwxr-x 1 seed seed 7396 Feb 11 00:24 task5
[02/11/20]seed@VM:~$ sudo chown root task5
[02/11/20]seed@VM:~$ sudo chmod 4755 task5
[02/11/20]seed@VM:~$ ls -l task5
-rwsr-xr-x 1 root seed 7396 Feb 11 00:24 task5
[02/11/20]seed@VM:~$
```

Step 3:

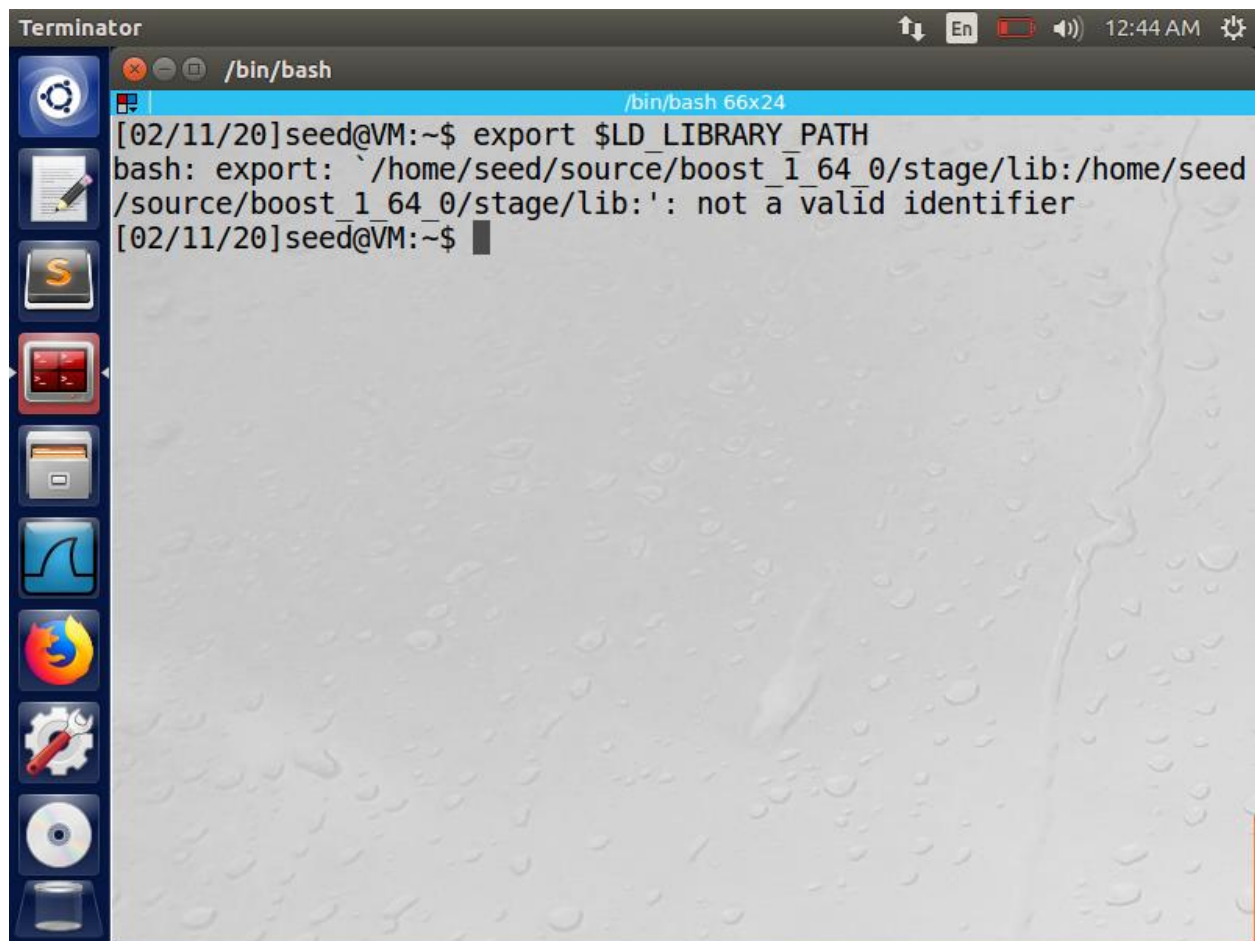
Output:

Using the export command to set PATH:



```
Terminator
/bin/bash
[02/11/20]seed@VM:~$ export $path
declare -x ANDROID_HOME="/home/seed/android/android-sdk-linux"
declare -x CLUTTER_IM_MODULE="xim"
declare -x COLORTERM="gnome-terminal"
declare -x COMPIZ_BIN_PATH="/usr/bin/"
declare -x COMPIZ_CONFIG_PROFILE="ubuntu-lowgfx"
declare -x DBUS_SESSION_BUS_ADDRESS="unix:abstract=/tmp/dbus-cp92D
djWZn"
declare -x DEFAULTS_PATH="/usr/share/gconf/ubuntu.default.path"
declare -x DERBY_HOME="/usr/lib/jvm/java-8-oracle/db"
declare -x DESKTOP_SESSION="ubuntu"
declare -x DISPLAY=":0"
declare -x GDMSESSION="ubuntu"
declare -x GDM_LANG="en_US"
declare -x GIO_LAUNCHED_DESKTOP_FILE="/usr/share/applications/term
inator.desktop"
declare -x GIO_LAUNCHED_DESKTOP_FILE_PID="28537"
declare -x GNOME_DESKTOP_SESSION_ID="this-is-deprecated"
declare -x GNOME_KEYRING_CONTROL=""
declare -x GNOME_KEYRING_PID=""
declare -x GPG_AGENT_INFO="/home/seed/.gnupg/S.gpg-agent:0:1"
declare -x GTK2_MODULES="overlay-scrollbar"
declare -x GTK_IM_MODULE="ibus"
declare -x GTK_MODULES="gail:atk-bridge:unity-gtk-module"
```

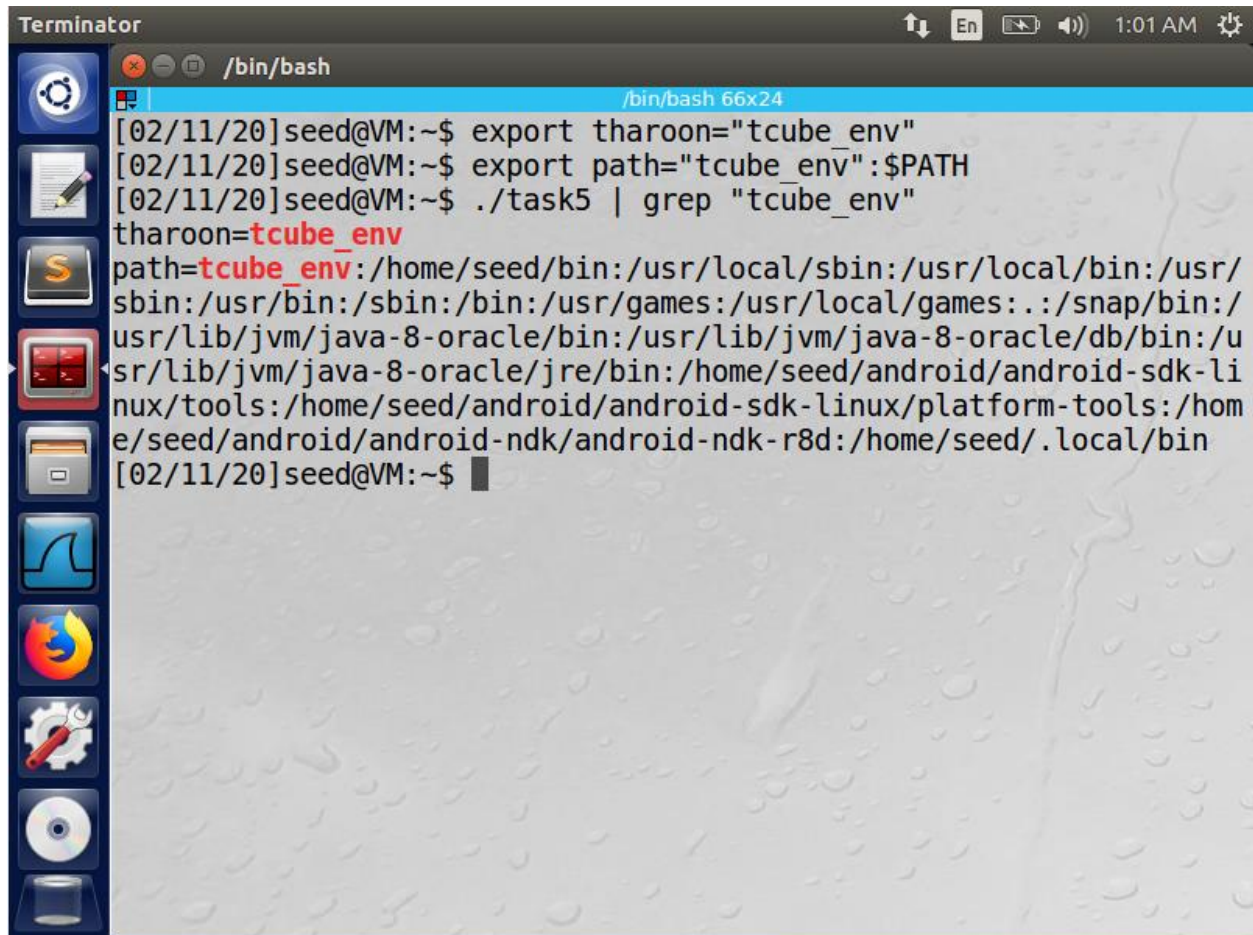
Using the export command to set LD_LIBRARY_PATH:



The image shows a screenshot of a Terminator terminal window. The title bar at the top reads "Terminator" and includes standard window controls (minimize, maximize, close) and system status icons (language, battery, volume, time 12:44 AM, and settings). The terminal window has a blue title bar that says "/bin/bash" and a status bar at the bottom that says "/bin/bash 66x24". The terminal content shows a user prompt "[02/11/20]seed@VM:~\$" followed by the command "export \$LD_LIBRARY_PATH". The next line shows the error message "bash: export: `/home/seed/source/boost_1_64_0/stage/lib:/home/seed/source/boost_1_64_0/stage/lib:': not a valid identifier". The prompt then returns to "[02/11/20]seed@VM:~\$". On the left side of the terminal window, there is a vertical dock with several application icons: a gear, a document with a pencil, a yellow 'S' icon, a red square icon, a folder icon, a blue square icon with a white 'M', the Firefox logo, a gear with a wrench, a CD icon, and a laptop icon.

```
[02/11/20]seed@VM:~$ export $LD_LIBRARY_PATH
bash: export: `/home/seed/source/boost_1_64_0/stage/lib:/home/seed
/source/boost_1_64_0/stage/lib:': not a valid identifier
[02/11/20]seed@VM:~$
```

I now create a new environment variable and set it into the new path using the export command. Export command will allow a child variable to inherit all the variables from the parent. Now, I create a new environment variable and append it to the PATH.

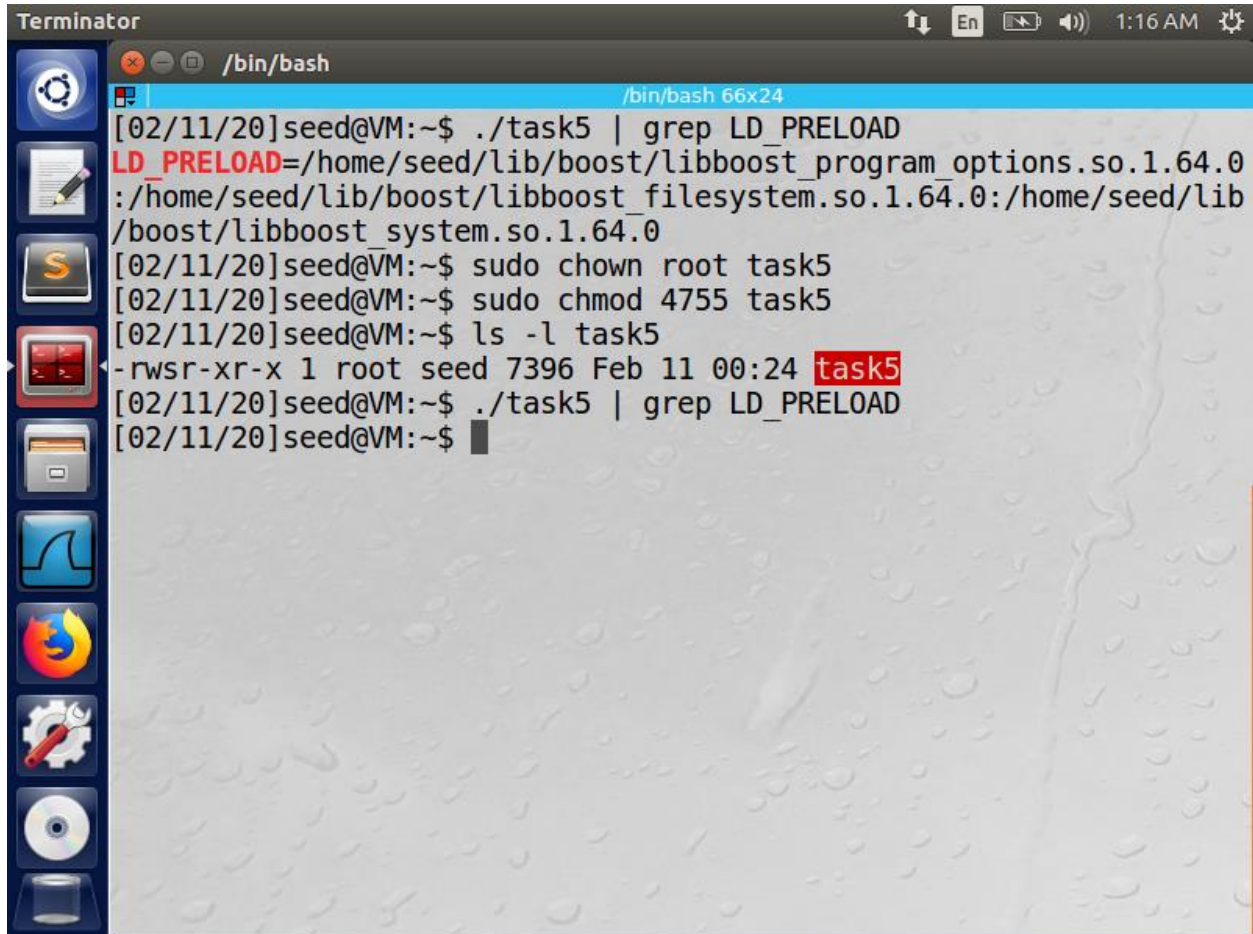


The image shows a Terminator terminal window with a dark theme. The title bar reads "Terminator" and includes standard window controls and system status icons (volume, network, battery, time 1:01 AM). The terminal window has a blue title bar that says "/bin/bash" and a subtitle "/bin/bash 66x24". The terminal content shows the following commands and output:

```
[02/11/20]seed@VM:~$ export tharoon="tcube_env"
[02/11/20]seed@VM:~$ export path="tcube_env":$PATH
[02/11/20]seed@VM:~$ ./task5 | grep "tcube_env"
tharoon=tcube_env
path=tcube_env:/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:/u
sr/lib/jvm/java-8-oracle/jre/bin:/home/seed/android/android-sdk-li
nux/tools:/home/seed/android/android-sdk-linux/platform-tools:/hom
e/seed/android/android-ndk/android-ndk-r8d:/home/seed/.local/bin
[02/11/20]seed@VM:~$
```

A vertical dock on the left side of the terminal window contains icons for various applications: a gear (settings), a notepad, a terminal, a terminal with a red cursor, a folder, a graph, a Firefox browser, a settings gear with a wrench, a CD/DVD, and a laptop.

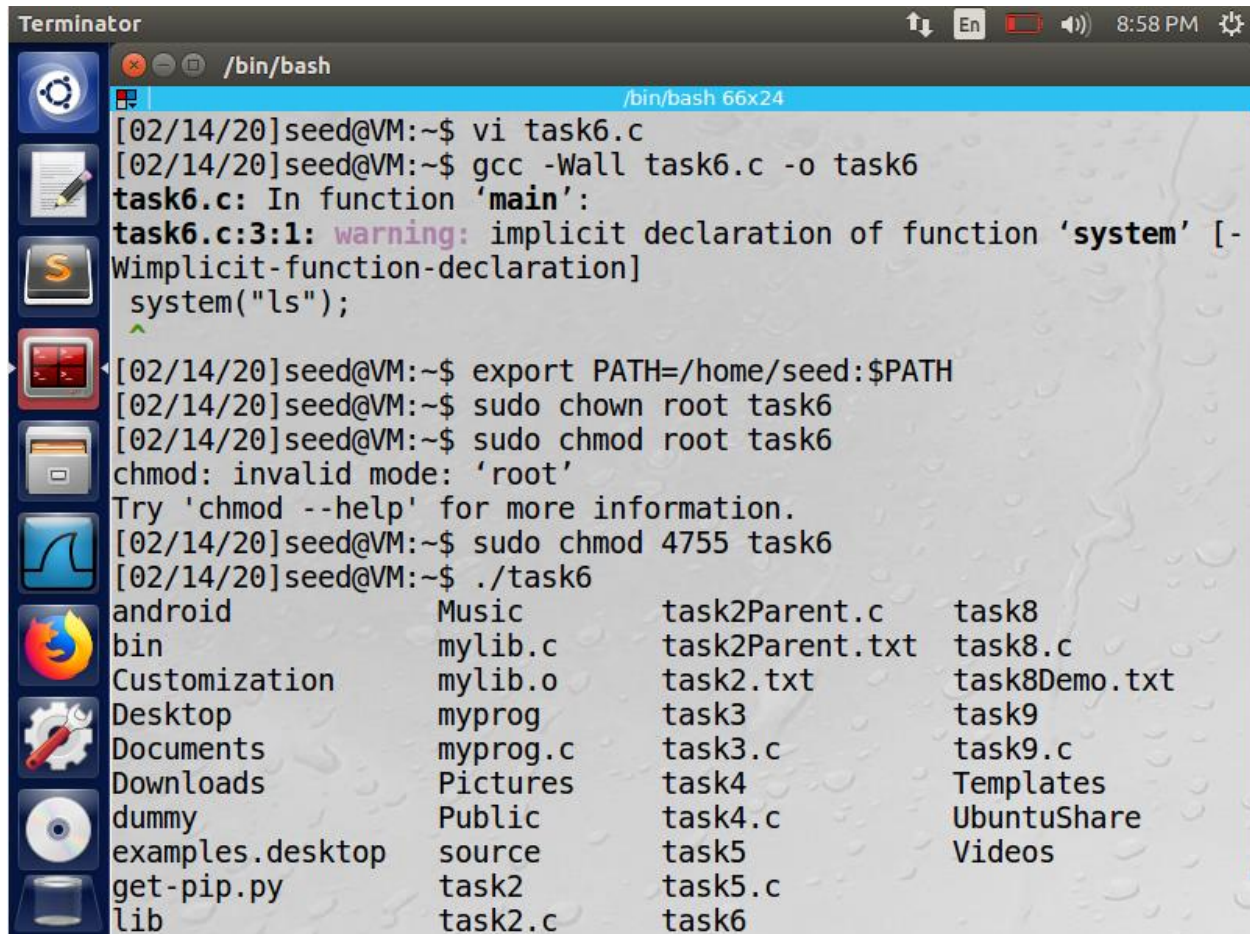
The Surprise thing I noticed is that when I run the task5.file and grep the LD_PRELOAD I am able to see the environment variable. If I change the task5.c file to root and change it as the SET-UID program and grep the LD_PRELOAD I am not able to find the environment variable. This shows that LD_* is not inheritable when run as root.



```
Terminator /bin/bash
/bin/bash 66x24
[02/11/20]seed@VM:~$ ./task5 | grep LD_PRELOAD
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
[02/11/20]seed@VM:~$ sudo chown root task5
[02/11/20]seed@VM:~$ sudo chmod 4755 task5
[02/11/20]seed@VM:~$ ls -l task5
-rwsr-xr-x 1 root seed 7396 Feb 11 00:24 task5
[02/11/20]seed@VM:~$ ./task5 | grep LD_PRELOAD
[02/11/20]seed@VM:~$
```

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

I compiled the given program and made it as root owned SET-UID program. Then I export the given environment variable `export PATH=/home/seed:$PATH`. Now when I run the program, I am able to run the `ls` command and I am able to see the list of all files and file directory. This is because of the countermeasure by the `/bin/sh`. Since the EUID and the RUID is different.



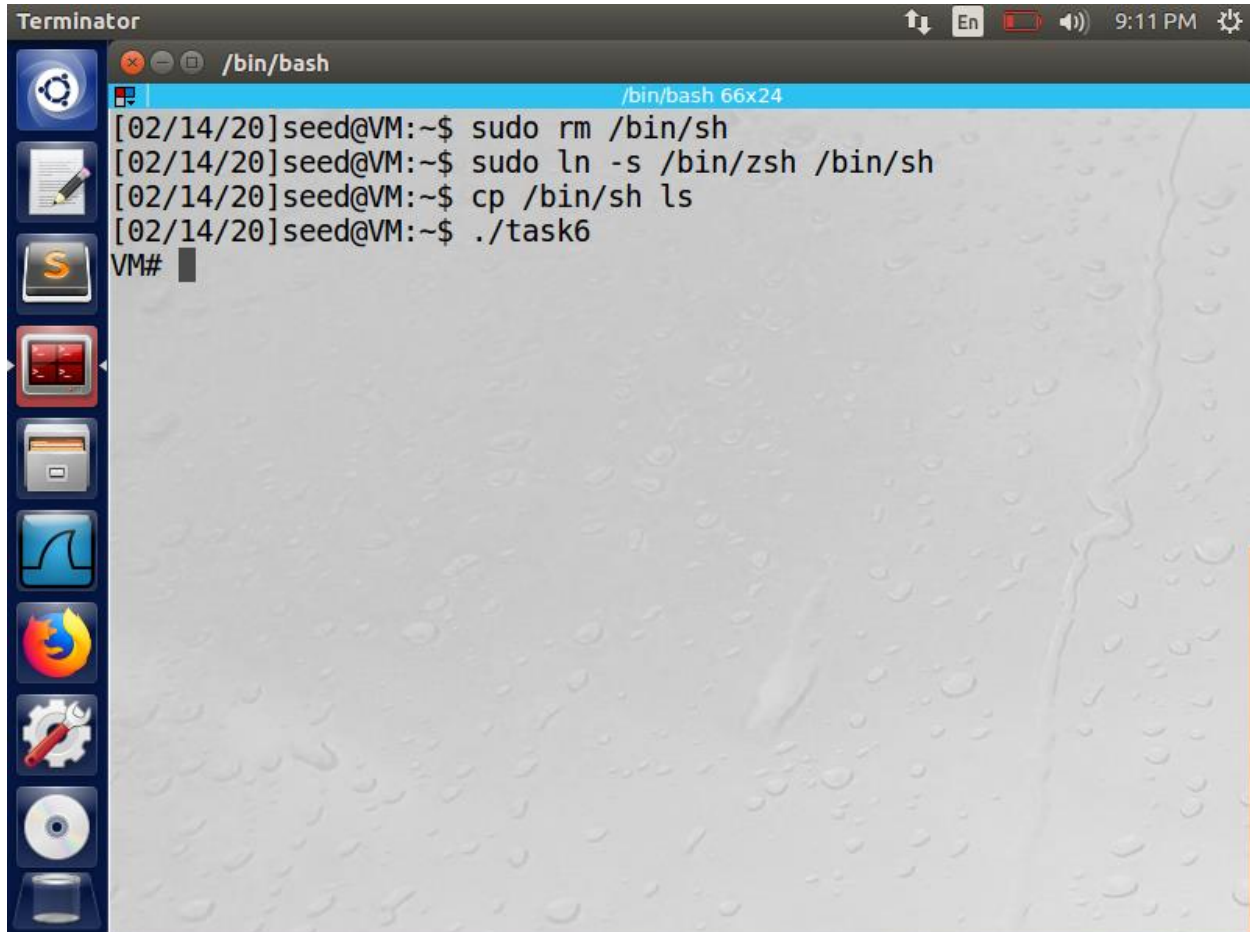
The screenshot shows a Terminator terminal window with a dark theme. The title bar reads "Terminator" and the window title is "/bin/bash". The terminal content shows the following sequence of commands and output:

```
[02/14/20]seed@VM:~$ vi task6.c
[02/14/20]seed@VM:~$ gcc -Wall task6.c -o task6
task6.c: In function 'main':
task6.c:3:1: warning: implicit declaration of function 'system' [-Wimplicit-function-declaration]
  system("ls");
  ^
[02/14/20]seed@VM:~$ export PATH=/home/seed:$PATH
[02/14/20]seed@VM:~$ sudo chown root task6
[02/14/20]seed@VM:~$ sudo chmod root task6
chmod: invalid mode: 'root'
Try 'chmod --help' for more information.
[02/14/20]seed@VM:~$ sudo chmod 4755 task6
[02/14/20]seed@VM:~$ ./task6
```

The output of the `./task6` command is a directory listing:

android	Music	task2Parent.c	task8
bin	mylib.c	task2Parent.txt	task8.c
Customization	mylib.o	task2.txt	task8Demo.txt
Desktop	myprog	task3	task9
Documents	myprog.c	task3.c	task9.c
Downloads	Pictures	task4	Templates
dummy	Public	task4.c	UbuntuShare
examples.desktop	source	task5	Videos
get-pip.py	task2	task5.c	
lib	task2.c	task6	

Now I linked the `/bin/sh` to another shell that does not have any countermeasure, using the given commands, I point to the shell with no countermeasure. And I have copied the `/bin/sh` to `ls`. Now when I run the given program, I am able to gain access to the root. This is because the shell which I changed has no countermeasure and this makes the program to gain the access to root.



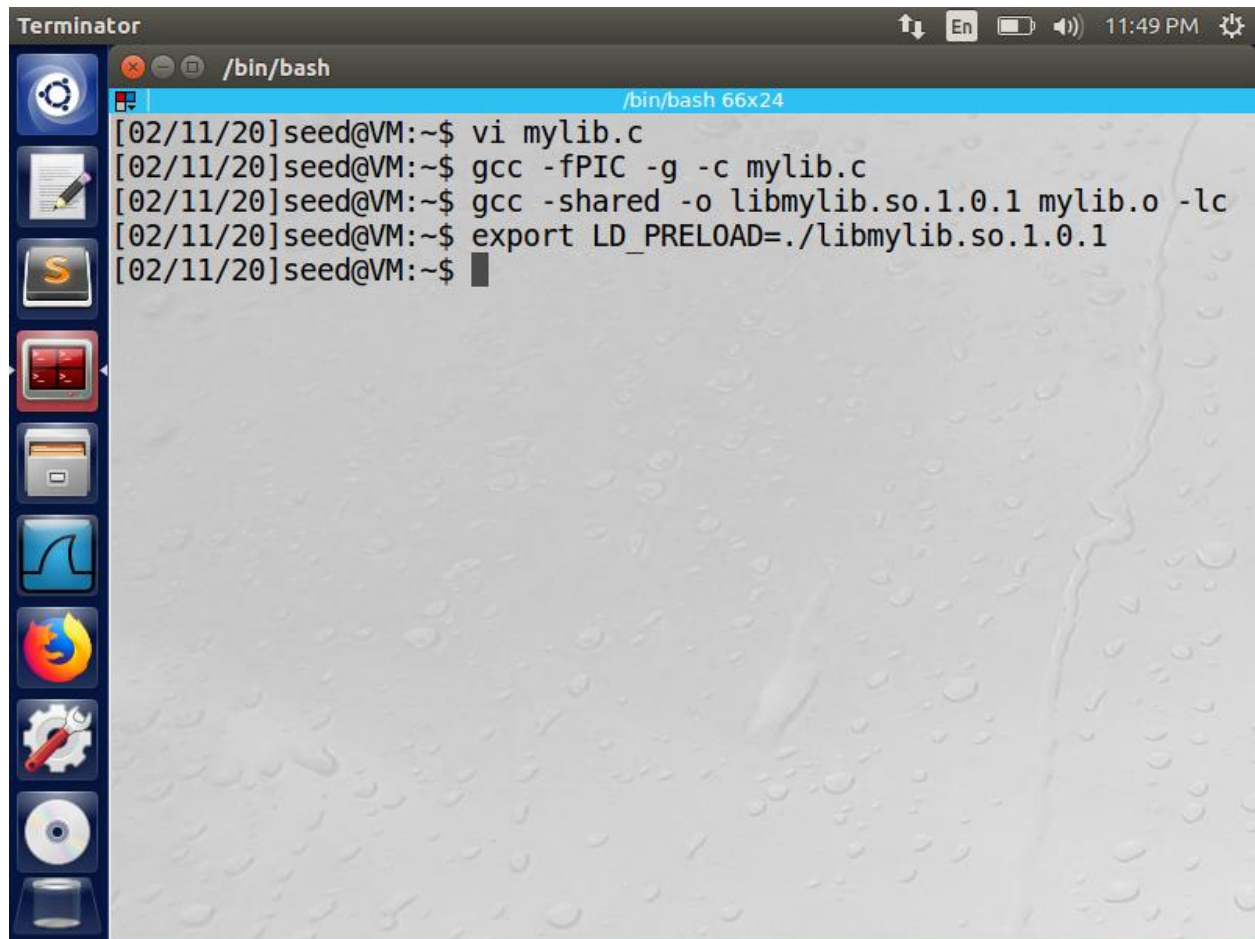
```
Terminator /bin/bash
[02/14/20]seed@VM:~$ sudo rm /bin/sh
[02/14/20]seed@VM:~$ sudo ln -s /bin/zsh /bin/sh
[02/14/20]seed@VM:~$ cp /bin/sh ls
[02/14/20]seed@VM:~$ ./task6
VM#
```


Task 7: The LD_PRELOAD Environment Variable and Set-UID Programs

Step 1:

Output:

Compiled and executed as given. Then created the shared path and exported the LD_PRELOAD path.



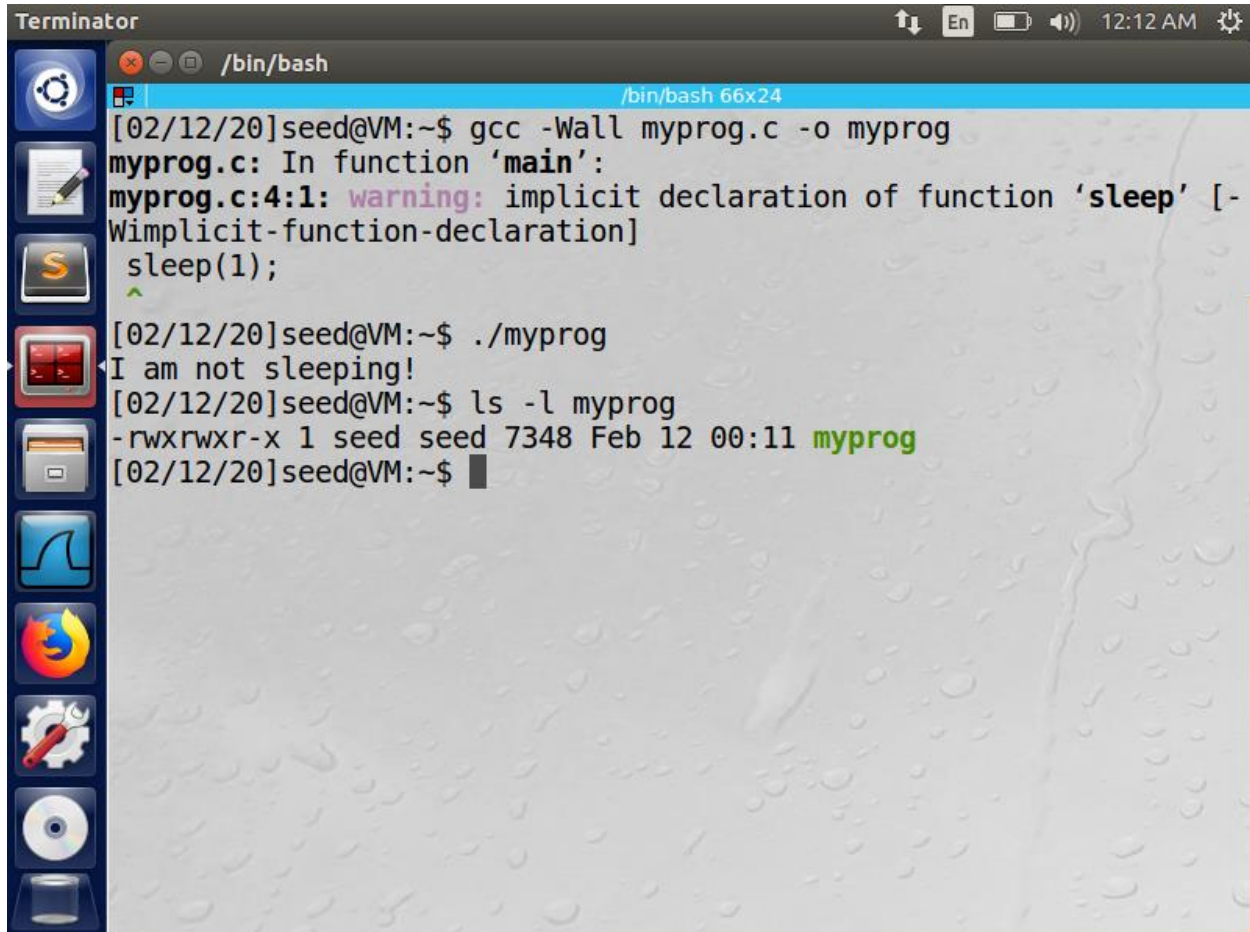
The screenshot shows a Terminator terminal window with a dark theme. The title bar reads "Terminator" and includes standard window controls and system status icons (network, volume, battery, time 11:49 PM). The terminal window has a blue header bar with "/bin/bash" and a smaller blue bar below it with "/bin/bash 66x24". The terminal content shows a series of commands and their outputs:

```
[02/11/20]seed@VM:~$ vi mylib.c
[02/11/20]seed@VM:~$ gcc -fPIC -g -c mylib.c
[02/11/20]seed@VM:~$ gcc -shared -o libmylib.so.1.0.1 mylib.o -lc
[02/11/20]seed@VM:~$ export LD_PRELOAD=./libmylib.so.1.0.1
[02/11/20]seed@VM:~$
```

A vertical sidebar on the left contains icons for various applications: a gear, a notepad, a terminal, a terminal with a red border, a folder, a graph, a Firefox browser, a settings gear, a CD/DVD, and a laptop.

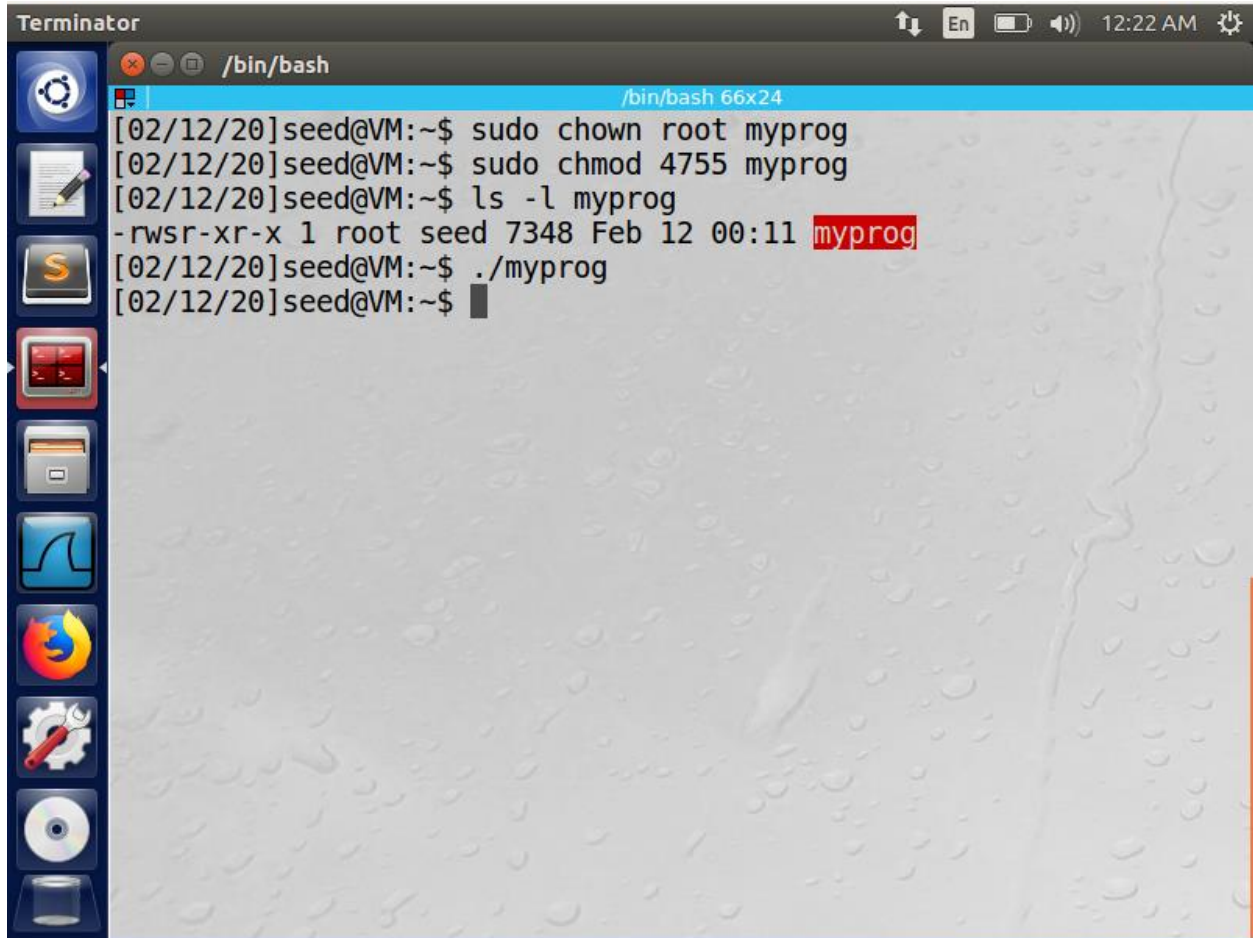
Step 2:

1. Now I made myprog as a regular program and ran it as normal user. I am able to see the output 'I am not sleeping!'. This is due to running the program in the normal user, which has EUID and RUID as same. Here the bash does not perform the counter measure.



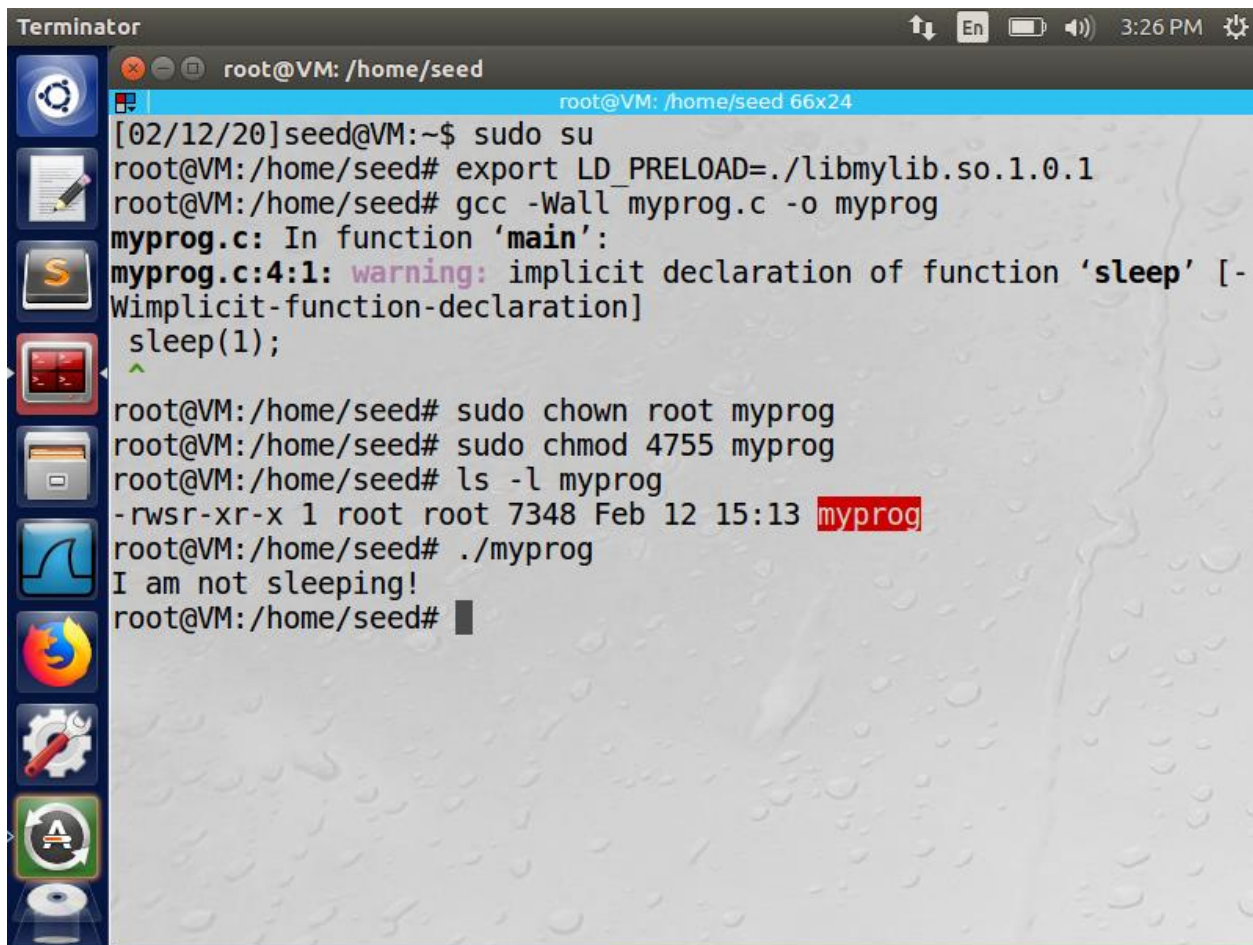
```
Terminator
/bin/bash
[02/12/20]seed@VM:~$ gcc -Wall myprog.c -o myprog
myprog.c: In function 'main':
myprog.c:4:1: warning: implicit declaration of function 'sleep' [-Wimplicit-function-declaration]
  sleep(1);
[02/12/20]seed@VM:~$ ./myprog
I am not sleeping!
[02/12/20]seed@VM:~$ ls -l myprog
-rwxrwxr-x 1 seed seed 7348 Feb 12 00:11 myprog
[02/12/20]seed@VM:~$
```

2. I made myprog to a SET-UID root program and I ran it as normal user. I was able to see that when I ran myprog as normal user, the sleep function of myprog is called. This is because of the SET-UID program. The bash takes the countermeasure and sees that EUID and RUID is different and does not execute the mylib program.



```
Terminator
/bin/bash
[02/12/20]seed@VM:~$ sudo chown root myprog
[02/12/20]seed@VM:~$ sudo chmod 4755 myprog
[02/12/20]seed@VM:~$ ls -l myprog
-rwsr-xr-x 1 root seed 7348 Feb 12 00:11 myprog
[02/12/20]seed@VM:~$ ./myprog
[02/12/20]seed@VM:~$
```

3. I now get into the root account and export the LD_ PRELOAD environment variable. Then I compile the myprog.c and change it to SET-UID program and ran the program, I was able to see the output getting printed. This is because both EUID and RUID is same which is the root user.



```
Terminator
root@VM: /home/seed
root@VM: /home/seed 66x24
[02/12/20]seed@VM:~$ sudo su
root@VM:/home/seed# export LD_PRELOAD=./libmylib.so.1.0.1
root@VM:/home/seed# gcc -Wall myprog.c -o myprog
myprog.c: In function 'main':
myprog.c:4:1: warning: implicit declaration of function 'sleep' [-Wimplicit-function-declaration]
sleep(1);
^
root@VM:/home/seed# sudo chown root myprog
root@VM:/home/seed# sudo chmod 4755 myprog
root@VM:/home/seed# ls -l myprog
-rwsr-xr-x 1 root root 7348 Feb 12 15:13 myprog
root@VM:/home/seed# ./myprog
I am not sleeping!
root@VM:/home/seed#
```

4. I created a new user called tharoon and made myprog owner as tharoon and made myprog a SET-UID program and I exit from the root. Now I am in the seed user and I export the LD_PRELOAD from the seed user. Now when I run the myprog I am not able to see the output. This is because myprog is now SET-UID program with its ownership as tharoon user.

Terminator

root@VM: /home/seed

root@VM: /home/seed 66x24

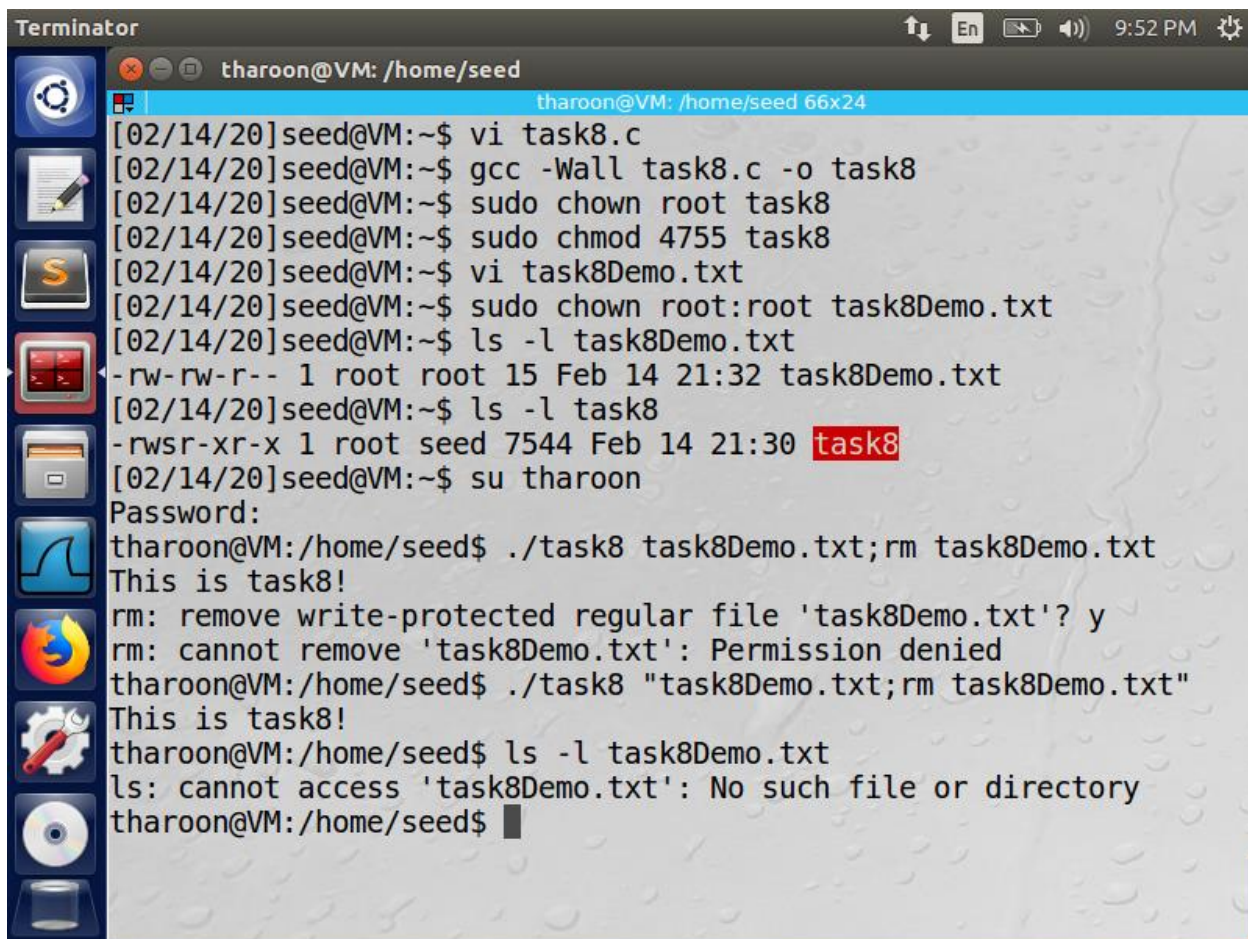
```
[02/12/20]seed@VM:~$ sudo su
root@VM:/home/seed# useradd -d /usr/tharoon -m tharoon
root@VM:/home/seed# chown tharoon myprog
root@VM:/home/seed# ls -l myprog
-rwxr-xr-x 1 tharoon root 7348 Feb 12 15:13 myprog
root@VM:/home/seed# chmod 4755 myprog
root@VM:/home/seed# ls -l myprog
-rwsr-xr-x 1 tharoon root 7348 Feb 12 15:13 myprog
root@VM:/home/seed# exit
exit
[02/12/20]seed@VM:~$ export LD_PRELOAD=./libmylib.so.1.0.1
[02/12/20]seed@VM:~$ ./myprog
[02/12/20]seed@VM:~$
```

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

Step 1:

Output:

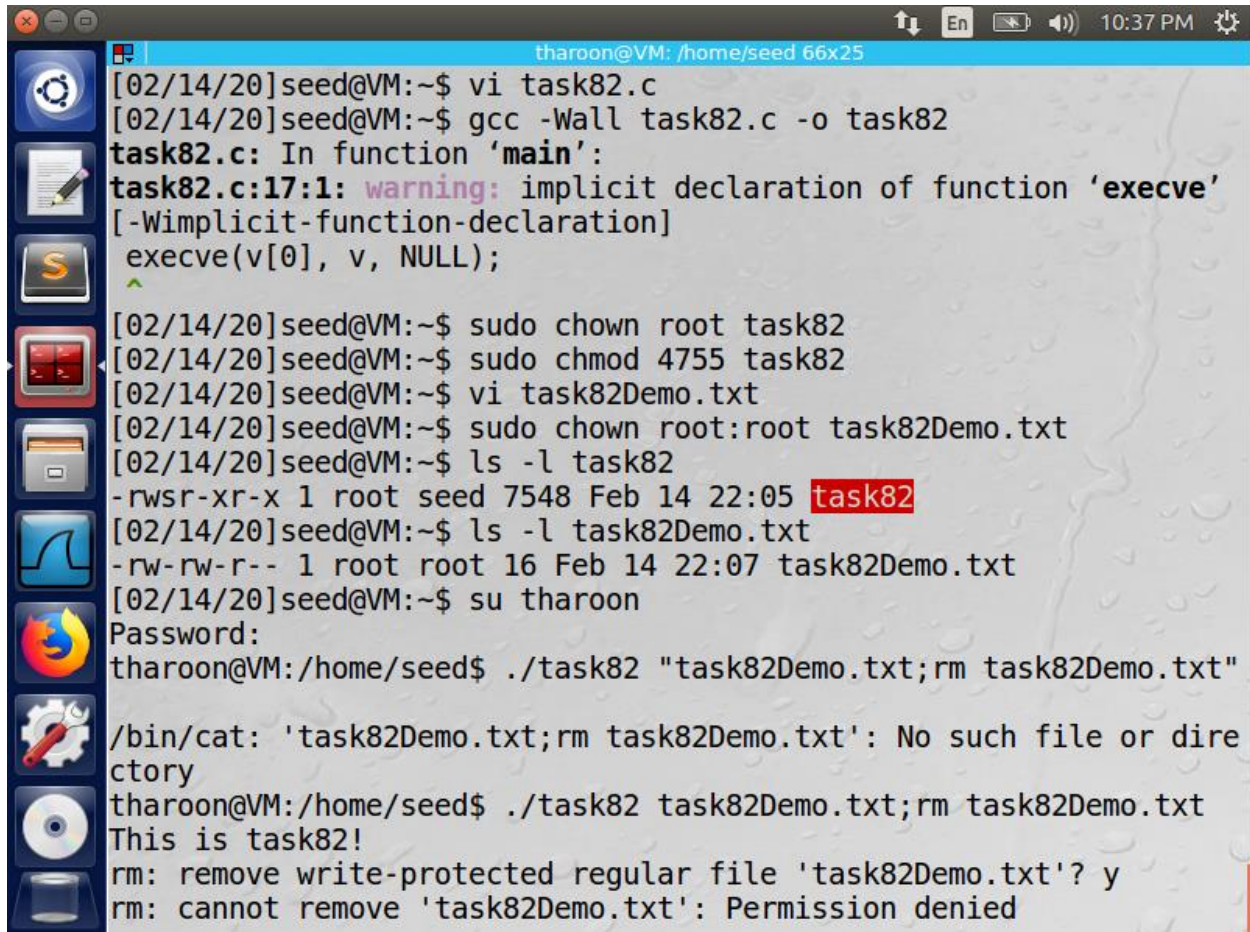
I compiled the given program and made it a SET-UID program and changed the owner to root. I also created a file called task8Demo.txt. Now, I enter into the user tharoon and run the given program and try removing the created file task8Demo.txt by giving the commands in double quotes. I am able to delete the file. This is because when system() executes, it doesn't execute the command directly. Instead it calls the execl() function, this execl() function eventually calls execve() to run /bin/sh. So, if the program is a Set-UID program, the user will have temporary root privileges and can remove any file with root privileges. And I also show that when removing the file without giving it as an argument I am not able to delete the file.



```
Terminator
tharoon@VM: /home/seed
tharoon@VM: /home/seed 66x24
[02/14/20]seed@VM:~$ vi task8.c
[02/14/20]seed@VM:~$ gcc -Wall task8.c -o task8
[02/14/20]seed@VM:~$ sudo chown root task8
[02/14/20]seed@VM:~$ sudo chmod 4755 task8
[02/14/20]seed@VM:~$ vi task8Demo.txt
[02/14/20]seed@VM:~$ sudo chown root:root task8Demo.txt
[02/14/20]seed@VM:~$ ls -l task8Demo.txt
-rw-rw-r-- 1 root root 15 Feb 14 21:32 task8Demo.txt
[02/14/20]seed@VM:~$ ls -l task8
-rwsr-xr-x 1 root seed 7544 Feb 14 21:30 task8
[02/14/20]seed@VM:~$ su tharoon
Password:
tharoon@VM:/home/seed$ ./task8 task8Demo.txt;rm task8Demo.txt
This is task8!
rm: remove write-protected regular file 'task8Demo.txt'? y
rm: cannot remove 'task8Demo.txt': Permission denied
tharoon@VM:/home/seed$ ./task8 "task8Demo.txt;rm task8Demo.txt"
This is task8!
tharoon@VM:/home/seed$ ls -l task8Demo.txt
ls: cannot access 'task8Demo.txt': No such file or directory
tharoon@VM:/home/seed$
```


Step2:

Now I commented out the `system()` function and uncommented the `execve()` function in the given program. I also created a file called `task82Demo.txt`. Now, I enter into the user `tharoon` and run the given program and try removing the created file `task82Demo.txt` by giving the commands in double quotes. I am not able to remove the file. This is because of the `execve()` command, which directly runs the commands.

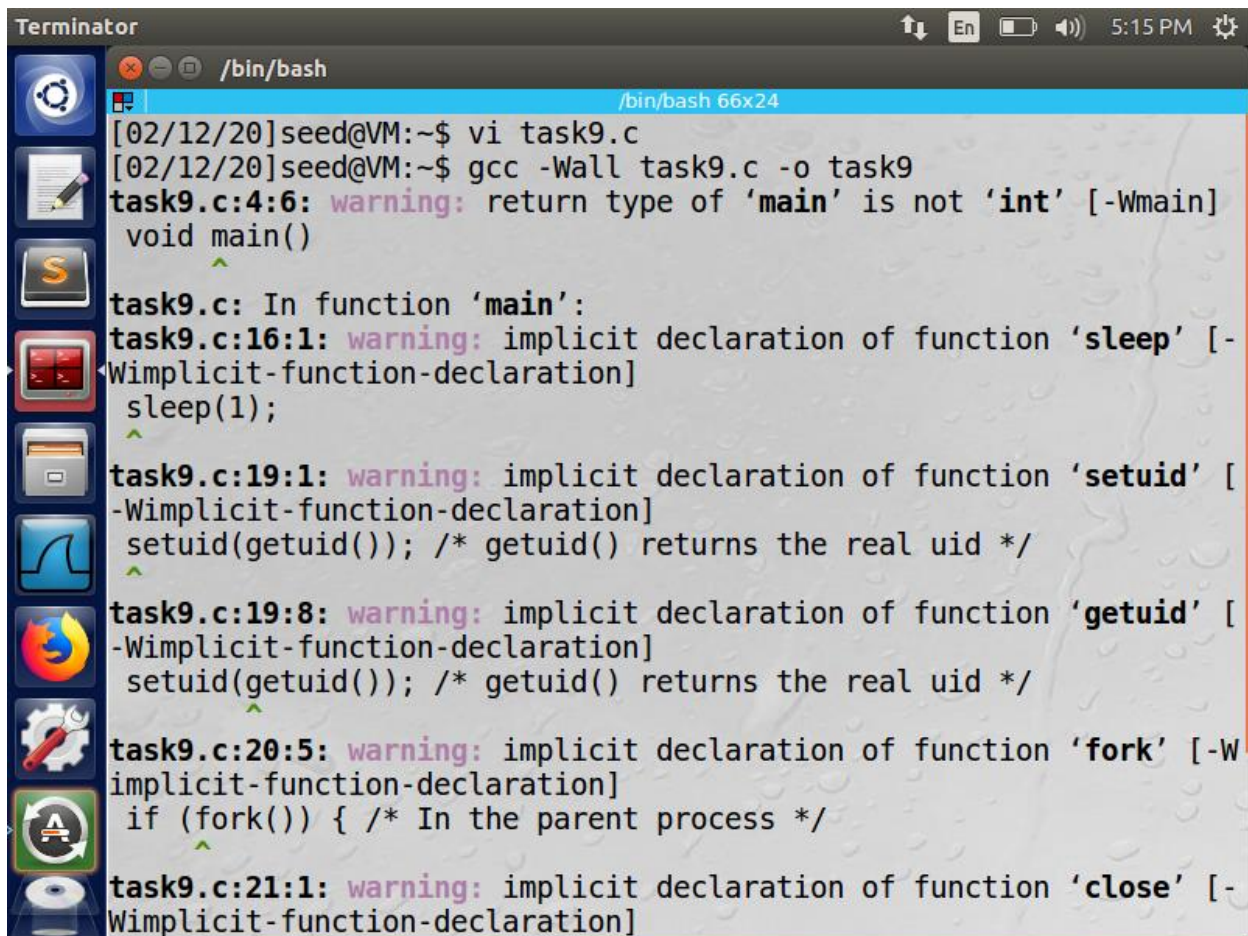


```
tharoon@VM: /home/seed 66x25
[02/14/20]seed@VM:~$ vi task82.c
[02/14/20]seed@VM:~$ gcc -Wall task82.c -o task82
task82.c: In function 'main':
task82.c:17:1: warning: implicit declaration of function 'execve'
[-Wimplicit-function-declaration]
  execve(v[0], v, NULL);
[02/14/20]seed@VM:~$ sudo chown root task82
[02/14/20]seed@VM:~$ sudo chmod 4755 task82
[02/14/20]seed@VM:~$ vi task82Demo.txt
[02/14/20]seed@VM:~$ sudo chown root:root task82Demo.txt
[02/14/20]seed@VM:~$ ls -l task82
-rwsr-xr-x 1 root seed 7548 Feb 14 22:05 task82
[02/14/20]seed@VM:~$ ls -l task82Demo.txt
-rw-rw-r-- 1 root root 16 Feb 14 22:07 task82Demo.txt
[02/14/20]seed@VM:~$ su tharoon
Password:
tharoon@VM:/home/seed$ ./task82 "task82Demo.txt;rm task82Demo.txt"
/bin/cat: 'task82Demo.txt;rm task82Demo.txt': No such file or directory
tharoon@VM:/home/seed$ ./task82 task82Demo.txt;rm task82Demo.txt
This is task82!
rm: remove write-protected regular file 'task82Demo.txt'? y
rm: cannot remove 'task82Demo.txt': Permission denied
```

Task 9: Capability Leaking

Observation:

I have compiled the given program and made the compiled a SET-UID program and changed its owner to root. I have also created a file /etc/zzz using the sudo touch command which makes the owner as root. Now when I run the program, I am able to see the output getting printed. This is because the child process refers to the file descriptor of the parent process. So, the privileges of the parent did not downgrade. Downgrading of privileges happens using the `setuid(getuid())` in the program. Since the privileges were not downgraded, the child process is able to access the created file. Also, the reason is file descriptor is not closed before the fork process.



```
Terminator /bin/bash
[02/12/20]seed@VM:~$ vi task9.c
[02/12/20]seed@VM:~$ gcc -Wall task9.c -o task9
task9.c:4:6: warning: return type of 'main' is not 'int' [-Wmain]
void main()
    ^
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]
```

Terminator

/bin/bash

/bin/bash 66x24

```
[02/12/20]seed@VM:~$ sudo chown root task9
[02/12/20]seed@VM:~$ sudo chmod 4755 task9
[02/12/20]seed@VM:~$ ls -l task9
-rwsr-xr-x 1 root seed 7640 Feb 12 17:12 task9
[02/12/20]seed@VM:~$ sudo touch /etc/zzz
[02/12/20]seed@VM:~$ ./task9
[02/12/20]seed@VM:~$ cat /etc/zzz
Malicious Data
[02/12/20]seed@VM:~$
```

tharoon@VM: /home/seed 66x25

```
[02/15/20]seed@VM:~$ ls -l /etc/zzz
-rw-r--r-- 1 root root 15 Feb 12 17:20 /etc/zzz
[02/15/20]seed@VM:~$
```