Environment Variable and Set-UID Program Lab

Name: Raman Srivastava
SUID: 946665605

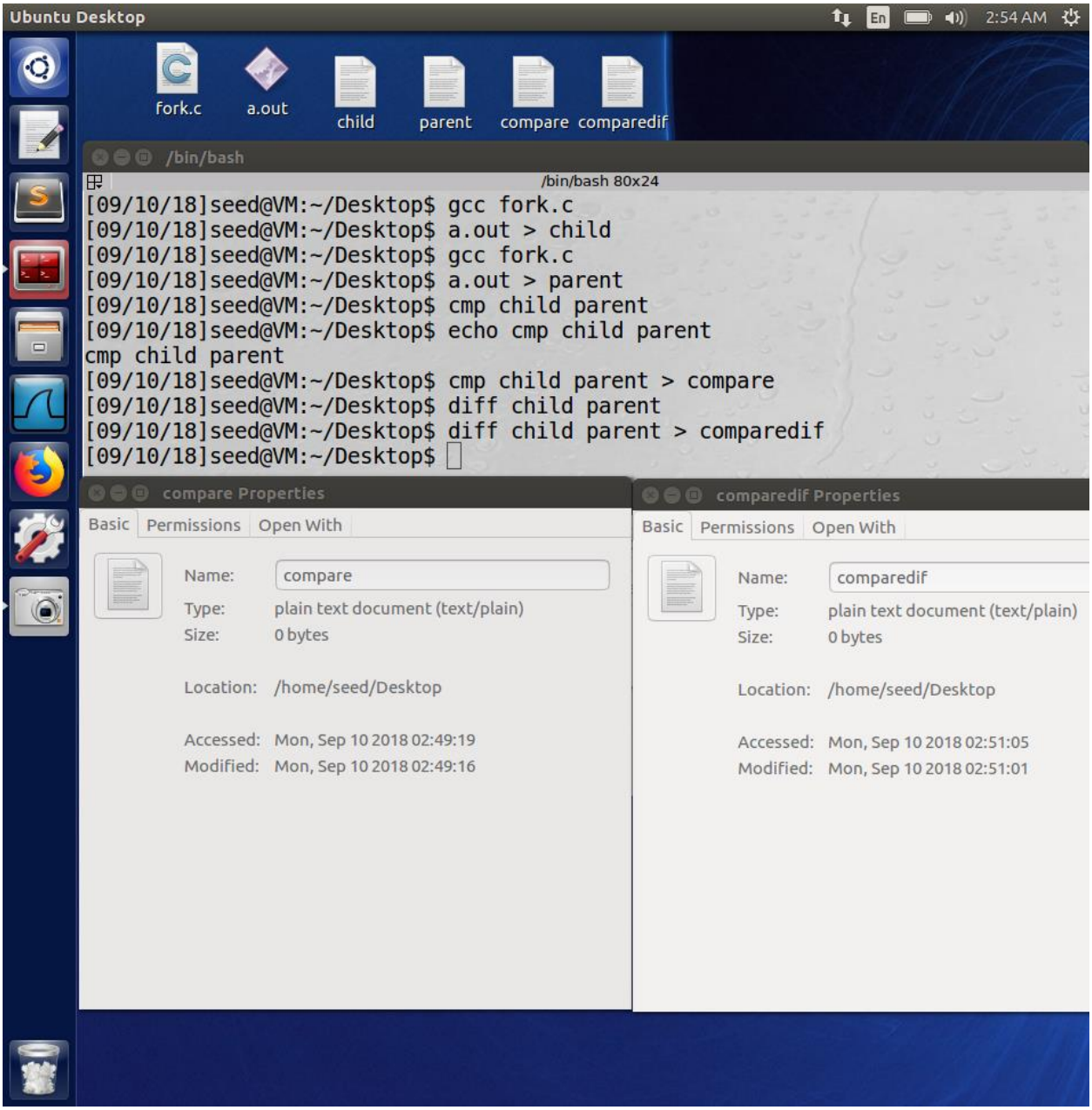
Task1: ManipulatingEnvironmentVariables

```
root@VM: /home/seed 80x42
root@VM:/home/seed# ./myprog
root@VM:/home/seed# clear
[3;J
root@VM:/home/seed# su seed
[09/12/18]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:74e60fc8-ddc7-4611-bbbc-7b63a4806fc8
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=2091
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/li
b/boost/libboost_filesystem.so.1.64.0:/home/seed/lib/boost/libboost_system.so.1.
64.0
WINDOWID=20971524
UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-session/1000/1064
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;4
```

```
root@VM: /home/seed 80x42
[09/12/18]seed@VM:~$ export RAMAN=SRIVASTAVA
[09/12/18]seed@VM:~$ env|grep RAMAN
RAMAN=SRIVASTAVA
[09/12/18]seed@VM:~$ unset RAMAN
[09/12/18]seed@VM:~$ env|grep RAMAN
[09/12/18]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:74e60fc8-ddc7-4611-bbbc-7b63a4806fc8
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=2091
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/li
b/boost/libboost_filesystem.so.1.64.0:/home/seed/lib/boost/libboost_system.so.1.
64.0
WINDOWID=20971524
UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-session/1000/1064
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;4
```

In this task, I've first printed out all the environment variables to see which one's are available. I later exported a new environment variable by the name RAMAN and I stored SRIVASTAVA in it. On searching for RAMAN using grep in the environment variable list, I found the environment variable we searched for. I later unset the RAMAN environment variable and verified its removal from the list by looking for it in the printenv result.

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



Here, I wrote the program in “gedit” and names the file “fork.c”. In the first iteration of “fork.c”, I commented the printenv() command under default statement to see the environment variables that the child process has. In the second iteration of “fork.c”, I commented the printenv() command of the child process to see the environment variables of the parent process. I compiled each iteration of fork.c separately.

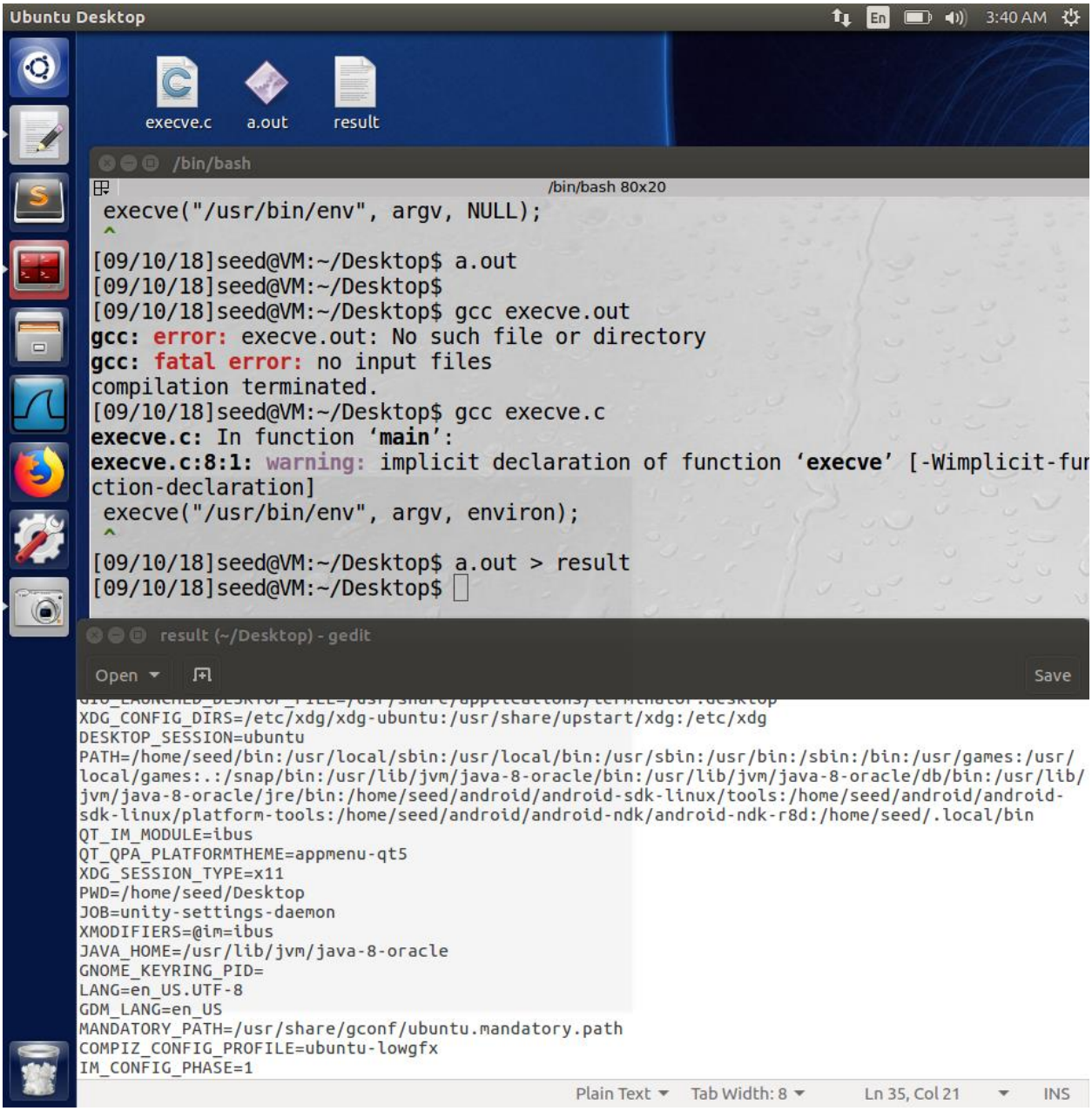
OBSERVATION

I saved the output of the first iteration as “child.txt” which has the list of environment variables of the

child process. After making the change in the program and compiling it again, I saved the output of the second iteration as “parent.txt”. I used the “cmp” command to compare “child.txt” and “parent.txt” files and saved the result of that as “compare.txt”. We can see the file size is 0 bytes, which means there’s no difference in the environment variables of the child and parent processes. This means when the “fork()” command was performed from the parent process, a child process was called and all the environment variables of the parent process were inherited by the child process.

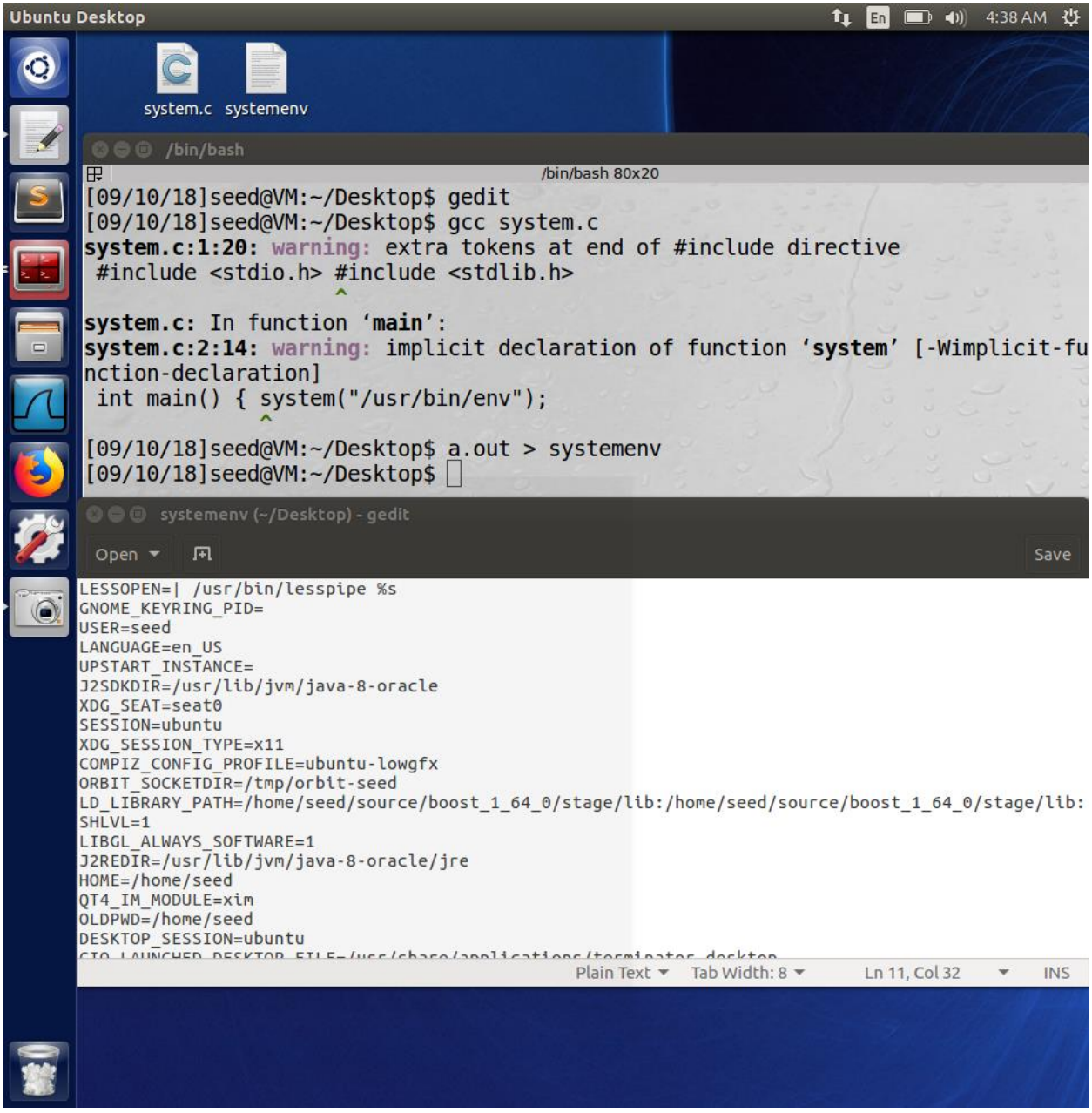
I also used the diff command for similar purpose and saved the result of that as “comparediff.txt”.

Task 3: Environment Variables and execve()



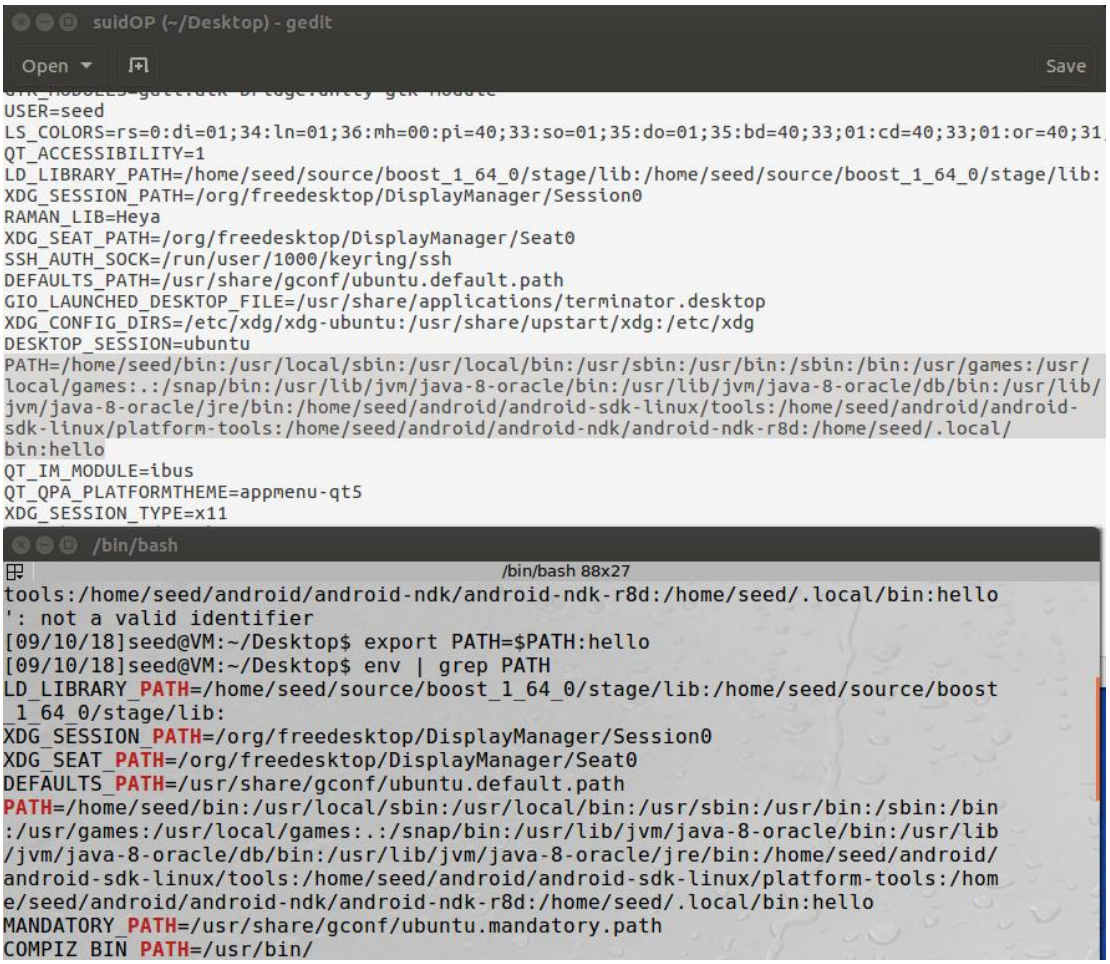
Here, in the first iteration, the environment variables are not passed to the new program. So it does not print anything. On the second iteration when we use “execve("/usr/bin/env", argv, environ);”, we’re passing environ as environment variables to the new program. So execve() command has the environment variables passed to “/usr/bin/env” which is why all the environment variables get printed. Execve() asks the operating system to execute the command passed to it as an argument, instead of asking the shell to do it, which is unsafe.

Task 4: Environment Variables and system()



In this task, we've printed all the environment variables using the system() command. What has happened here is, system() has acted as a bridge or the middleman between the program and the shell. So system has just called the shell and given the shell a command, which was passed to it as a parameter in the C program.

Task 5: Environment Variables and Set-UID Programs

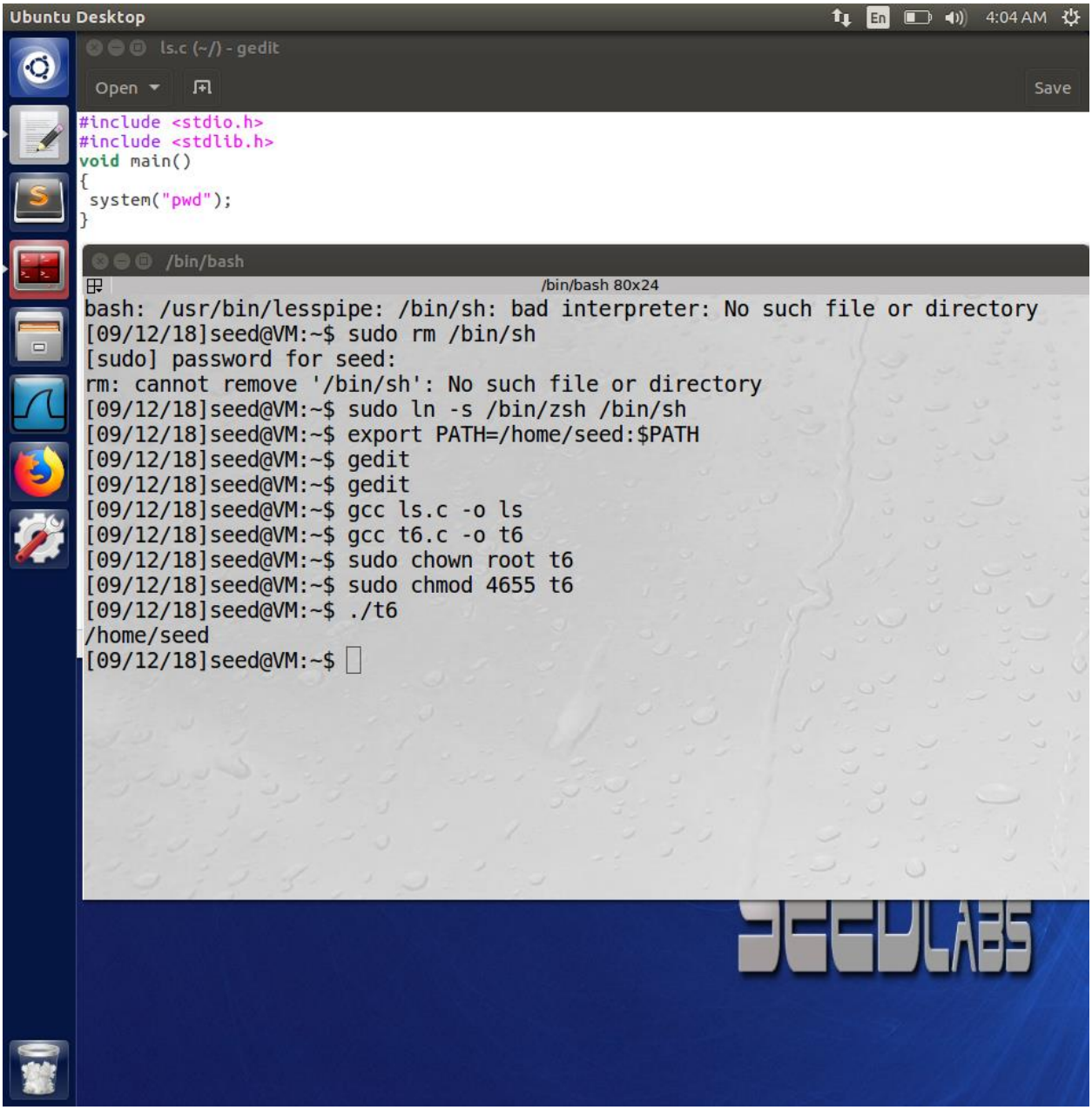


```
suidOP (~/Desktop) - gedit
Open ▾                                                                    Save
GIO_LAUNCHED_DESKTOP_FILE_PID=2213
ANDROID_HOME=/home/seed/android/android-sdk-linux
GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1
TERM=xterm
SHELL=/bin/bash
DERBY_HOME=/usr/lib/jvm/java-8-oracle/db
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=20971524
OLDPWD=/home/seed
UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-session/1000/1097
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
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
RAMAN_LIB=Heya
```

```
/bin/bash
/bin/bash 88x27
COMPIZ_BIN_PATH=/usr/bin/
[09/10/18]seed@VM:~/Desktop$ export LD_LIBRARY_PATH
[09/10/18]seed@VM:~/Desktop$ export RAMAN_LIB=Heya
[09/10/18]seed@VM:~/Desktop$ a.out
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:8531fcab-d7ea-4364-9a55-bdf8e5468c78
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=2213
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/li
```

After saving the program, I compiled it, changed the ownership to root and made it a Set-UID program. Upon exporting Path, I could see the changes being reflected in the environment variables printed (highlighted in the text file). Similar was the case with a user defined environment variable, RAMAN in this example. Both the environment variables got inherited by the child process from the parent. However, the LD_LIBRARY_PATH variable remained uninherited. This shows that even if the program is a Set-UID Program, LD_LIBRARY_PATH refuses to get inherited to the child process for security purposes, because before the execution of any program, it first check the LD_LIBRARY_PATH for libraries.

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



In this task, I wrote a program which passed “pwd” as a string parameter to system(). We manipulated the environment variable PATH so that when ‘ls’ is passed as a relative path, it looks up at PATH to determine where to look for ‘ls’. I created a program and names the file as ls, so when the program in file t6 calls for ‘ls’ which is a relative path, it looks up the environment variable PATH and looks for ‘ls’ there, which in this directory happens to be an executable file to print the present work directory (pwd). In short words, the ls command has been replaced by pwd and the program has been compromised by exploiting the PATH environment variable.

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

```
root@VM: /home/seed 80x42
[09/12/18]seed@VM:~$ gedit
[09/12/18]seed@VM:~$ gcc -fPIC -g -c mylib.c
[09/12/18]seed@VM:~$ gcc -shared -o libmylib.so.1.0.1 mylib.o -lc
[09/12/18]seed@VM:~$ export LD_PRELOAD=./libmylib.so.1.0.1
[09/12/18]seed@VM:~$ gedit
[09/12/18]seed@VM:~$ gcc myprog.c -o myprog
myprog.c: In function ‘main’:
myprog.c:5:1: warning: implicit declaration of function ‘sleep’ [-Wimplicit-func
tion-declaration]
 sleep(1);
 ^
[09/12/18]seed@VM:~$ ./myprog
I am not sleeping!
[09/12/18]seed@VM:~$ sudo chown root myprog
[sudo] password for seed:
[09/12/18]seed@VM:~$ sudo chmod 4655 myprog
[09/12/18]seed@VM:~$ ls -l myprog
-rwSr-xr-x 1 root seed 7348 Sep 12 06:58 myprog
[09/12/18]seed@VM:~$ ./myprog
[09/12/18]seed@VM:~$ su
Password:
root@VM:/home/seed# sudo chown root myprog
root@VM:/home/seed# chmod 4655 myprog
root@VM:/home/seed# export LD_PRELOAD=./libmylib.so.1.0.1
root@VM:/home/seed# ./myprog
I am not sleeping!
root@VM:/home/seed# adduser user1
Adding user `user1' ...
Adding new group `user1' (1002) ...
Adding new user `user1' (1002) with group `user1' ...
Creating home directory `/home/user1' ...
Copying files from `/etc/skel' ...
ERROR: ld.so: object './libmylib.so.1.0.1' from LD_PRELOAD cannot be preloaded (
cannot open shared object file): ignored.
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for user1
Enter the new value, or press ENTER for the default
        Full Name []: user1
        Room Number []:
        Work Phone []:
```

```
root@VM: /home/seed 80x42
[09/12/18]seed@VM:~$ su
Password:
root@VM:/home/seed# sudo chown root myprog
root@VM:/home/seed# chmod 4655 myprog
root@VM:/home/seed# export LD_PRELOAD=./libmylib.so.1.0.1
root@VM:/home/seed# ./myprog
I am not sleeping!
root@VM:/home/seed# adduser user1
Adding user `user1' ...
Adding new group `user1' (1002) ...
Adding new user `user1' (1002) with group `user1' ...
Creating home directory `/home/user1' ...
Copying files from `/etc/skel' ...
ERROR: ld.so: object './libmylib.so.1.0.1' from LD_PRELOAD cannot be preloaded (
cannot open shared object file): ignored.
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for user1
Enter the new value, or press ENTER for the default
        Full Name []: user1
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] y
root@VM:/home/seed# su user1
user1@VM:/home/seed$ sudo chown user1 myprog
[sudo] password for user1:
user1 is not in the sudoers file.  This incident will be reported.
user1@VM:/home/seed$ su
Password:
root@VM:/home/seed# sudo chown user1 myprog
root@VM:/home/seed# sudo chmod 4655 myprog
root@VM:/home/seed# export LD_PRELOAD=./libmylib.so.1.0.1
```

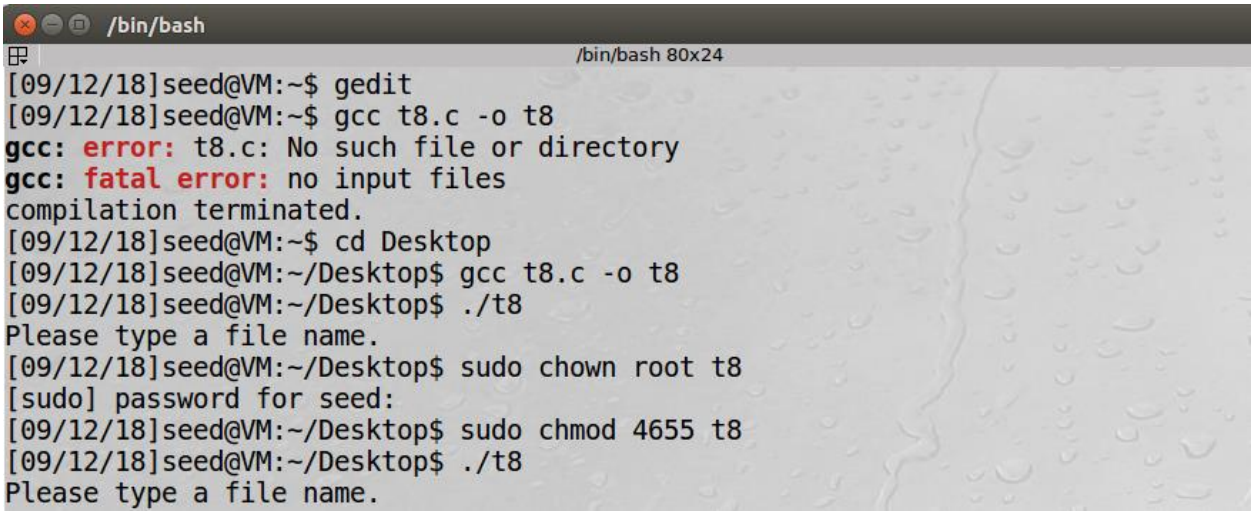
```
user1@VM: /home/seed
user1@VM: /home/seed 80x14
COLORTERM=gnome-terminal
_=/usr/bin/printenv
[09/12/18]seed@VM:~$ su user1
Password:
user1@VM:/home/seed$ ./myprog
bash: ./myprog: Permission denied
user1@VM:/home/seed$ ls -l myprog
-rwSr-xr-x 1 user1 seed 7348 Sep 12 06:58 myprog
user1@VM:/home/seed$ su seed
Password:
[09/12/18]seed@VM:~$ ./myprog
[09/12/18]seed@VM:~$ export LD_PRELOAD=./libmylib.so.1.0.1
[09/12/18]seed@VM:~$ ./myprog
[09/12/18]seed@VM:~$
```

- Make myprog a regular program, and run it as a normal user.
  Ans: In this step, the program does not sleep. The main function calls the sleep function which is in the dynamic link library we created and exported. The library was exported to LD_PRELOAD environment variable.
- Make myprog a Set-UID root program, and run it as a normal user.
  Ans: In this step, the program myprog has been made to a Set-UID program which is owned by the root. The program sleeps here and the dynamic link library we created is not is not invoked.
- Make myprog a Set-UID root program, export the LD_PRELOAD environment variable again in the root account and run it.
  Ans: Here the program does not sleep. The dynamic link library was accessed because we exported the library to the LD_PRELOAD environment variable. Despite myprog being a Set-UID program, the dynamic link library works because the RUID and EUID is the same. Here, the program is executed from the root, and the owner of the program is also the root.

- 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.
  Ans: Here I made a new user 'user1' and made it the owner of the Set-UID 'myprog'. I then exported the LD_PRELOAD environment variable in the seed account and ran it from there. The program went to sleep.

  I observed that the dynamic link library loaded to the LD_PRELOAD environment variable gets loaded when the real UID and effective UID are from the root.

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



```
/bin/bash
/bin/bash 80x24
[09/12/18]seed@VM:~$ cd Desktop
[09/12/18]seed@VM:~/Desktop$ /bin/ls
a.out  compare     execve.c  parent  result  suidOP    systemenv  t8.c
child  comparedif  fork.c    Raman   suid.c  system.c  t8         toremove.txt
[09/12/18]seed@VM:~/Desktop$ ./t8 "aa;rm toremove.txt"
/bin/cat: aa: No such file or directory
[09/12/18]seed@VM:~/Desktop$ /bin/ls
a.out  compare     execve.c  parent  result  suidOP    systemenv  t8.c
child  comparedif  fork.c    Raman   suid.c  system.c  t8
```

```
/bin/bash
/bin/bash 80x24
[09/12/18]seed@VM:~/Desktop$ gcc t8.c -o t8
t8.c: In function 'main':
t8.c:18:3: warning: implicit declaration of function 'execve' [-Wimplicit-funct
on-declaration]
   execve(v[0], v, NULL);
   ^
[09/12/18]seed@VM:~/Desktop$ sudo chown root t8
[sudo] password for seed:
[09/12/18]seed@VM:~/Desktop$ sudo chmod 4655 t8
[09/12/18]seed@VM:~/Desktop$ /bin/ls -l t8
-rwSr-xr-x 1 root seed 7544 Sep 12 05:32 t8
[09/12/18]seed@VM:~/Desktop$ ./t8
Please type a file name.
[09/12/18]seed@VM:~/Desktop$ /bin/ls
a.out  compare     execve.c  parent  removexecve.txt  suid.c  system.c   t8
child  comparedif  fork.c    Raman   result           suidOP  systemenv  t8.c
[09/12/18]seed@VM:~/Desktop$ ./t8 "aa;rm removexecve.txt"
/bin/cat: 'aa;rm removexecve.txt': No such file or directory
[09/12/18]seed@VM:~/Desktop$
```

For this task, I created 2 files. 'toremove.txt' which should be removed by system() and 'removexecve.txt' that should be removed by execve(). Because system() invokes /bin/sh, we can pass multiple commands because shell can execute multiple commands at once. So we were successful in removing 'toremove.txt' However in the case of execve(), the three arguments in the function stands for command to be executed, parameters for the command and the environment variables to be passed with the command. So if we take try to take multiple commands, it'll consider it as 1 string and the removing process will fail. So we're unable to delete the 'removexecve.txt' file.

Task 9: Capability Leaking

```
root@VM: /etc
root@VM: /etc 80x40
[09/12/18]seed@VM:/etc$ su
Password:
root@VM:/etc# touch zzz
root@VM:/etc# ls
```

```
root@VM: /etc
root@VM: /etc 80x40
dhcp                    mailcap.order           sysctl.conf
dictionaries-common     manpath.config          sysctl.d
dnsmasq.d               mime.types              systemd
doc-base                mke2fs.conf             terminfo
dpkg                    modprobe.d              thermald
drirc                   modules                 thunderbird
emacs                   modules-load.d          timezone
environment             mono                    tmpfiles.d
firefox                 mtab                    ucf.conf
fonts                   mtools.conf             udev
fstab                   mysql                   udisks2
ftpusers                nanorc                  ufw
fuse.conf               network                 updatedb.conf
fwupd.conf              NetworkManager          update-manager
gai.conf                networks                update-motd.d
gconf                   newt                    update-notifier
gdb                     nsswitch.conf           UPower
ghostscript             opt                     upstart-xsessions
gnome                   os-release              usb_modeswitch.conf
gnome-app-install       pam.conf                usb_modeswitch.d
gnome-vfs-2.0           pam.d                   vim
groff                   papersize               vsftpd.conf
group                   passwd                  vtrgb
group-                  passwd-                 wgetrc
grub.d                  pcmcia                  whoopsie
gshadow                 perl                    wireshark
gshadow-                php                     wpa_supplicant
gss                     phpmyadmin              X11
gtk-2.0                 pki                     xdg
gtk-3.0                 pm                      xml
guest-session           pnm2ppa.conf            zsh
hdparm.conf             polkit-1                zsh command not found
host.conf               popularity-contest.conf zzz
hostname                ppp
[09/12/18]seed@VM:/etc$ gedit zzz
[09/12/18]seed@VM:/etc$ cat zzz
Malicious Data
```

In this task, when I wrote the program, I saved a blank text file ‘zzz.txt’. When the setuid(getuid()) function is called, it should release its capabilities. However when the fork() operation is done, the child inherits from the parent. Also the file zzz.txt got Malicious Data written in it, which means the file was modifiable.