实验一 环境变量与特权程序实验

Task 1: 修改环境变量

（1）使用 printenv 或 env 指令查看当前进程的环境变量。

```
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2# printenv
SHELL=/bin/bash
COLORTERM=truecolor
SUDO_GID=1000
LANGUAGE=zh_CN:en_US:en
LC_ADDRESS=zh_CN.UTF-8
LC_NAME=zh_CN.UTF-8
SUDO_COMMAND=/bin/bash
LC_MONETARY=zh_CN.UTF-8
SUDO_USER=maskikeigo
PWD=/home/maskikeigo/桌面/ComputerSecurity/chap2
LOGNAME=root
XAUTHORITY=/run/user/1000/gdm/Xauthority
HOME=/root
LANG=zh_CN.UTF-8
LC_PAPER=zh_CN.UTF-8
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=4
0;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:*.l
z4=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:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.x
z=01;31:*.zst=01;31:*.tzst=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:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.jpg=01;35:*.jpeg=01
;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.p
pm=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:*.mp4
v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rm
vb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.x
cf=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;3
6:*.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:
LESSCLOSE=/usr/bin/lesspipe %s %s
LC_IDENTIFICATION=zh_CN.UTF-8
```

（2）使用 export 和 unset 指令添加或删除环境变量。

```
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2# export china=dragon
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2# echo $china
dragon
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2# unset china
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2# echo $china

root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2#
```

Task 2：将 EV 由父进程传递给子进程

（1）编译并运行程序，将程序打印的内容存放到文件 chid.txt 中。

```
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# ls -l
总用量 4
-rw-rw-r-- 1 maskikeigo maskikeigo 506 3月  28 09:50 printEV.c
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# gcc printEV.c -o printEV
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# ./printEV
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# SHELL=/bin/bash
COLORTERM=truecolor
SUDO_GID=1000
LANGUAGE=zh_CN:en_US:en
LC_ADDRESS=zh_CN.UTF-8
LC_NAME=zh_CN.UTF-8
SUDO_COMMAND=/bin/bash
LC_MONETARY=zh_CN.UTF-8
SUDO_USER=maskikeigo
PWD=/home/maskikeigo/桌面/ComputerSecurity/chap2/lab
LOGNAME=root
XAUTHORITY=/run/user/1000/gdm/Xauthority
HOME=/root
LANG=zh_CN.UTF-8
LC_PAPER=zh_CN.UTF-8
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;4
;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:*.
01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*
2=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=
1:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.jpg=01;35:
jpg=01;35:*.mjpeg=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:*
g=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:*.
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=0
:*.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:*.
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=
6:*.xspf=00;36:
```

```
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# ls
printEV  printEV.c
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# ./printEV > child
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab#
```

Child.txt 中保存的程序输出的内容：当前环境变量。

printEV.c child

lab > child

```
1  SHELL=/bin/bash
2  COLORTERM=truecolor
3  SUDO_GID=1000
4  LANGUAGE=zh_CN:en_US:en
5  LC_ADDRESS=zh_CN.UTF-8
6  LC_NAME=zh_CN.UTF-8
7  SUDO_COMMAND=/bin/bash
8  LC_MONETARY=zh_CN.UTF-8
9  SUDO_USER=maskikeigo
10 PWD=/home/maskikeigo/桌面/ComputerSecurity/chap2/lab
11 LOGNAME=root
12 XAUTHORITY=/run/user/1000/gdm/Xauthority
13 HOME=/root
14 LANG=zh_CN.UTF-8
```

（2）按文件修改源码后重新编译，查看结果并将结果保存到 child2.txt。

观察结果：两次运行的输出结果完全一致。

（3）使用 diff 指令输出的结果如下：无差别。结论：子进程将会拷贝了父进程 environ 指针数组指向的所有环境变量。

```
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# ls -l
总用量 32
-rw-r--r-- 1 root       root        2330 3月   28 10:19 child2.txt
-rw-r--r-- 1 root       root        2330 3月   28 10:07 child.txt
-rwxr-xr-x 1 root       root       16888 3月   28 10:16 printEV
-rw-rw-r-- 1 maskikeigo maskikeigo   532 3月   28 10:15 printEV.c
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# diff child.txt child2.txt
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab#
```

Task 3：环境变量与 execve()

（1）编译并执行程序，观察程序执行结果如下：没有打印环境变量。

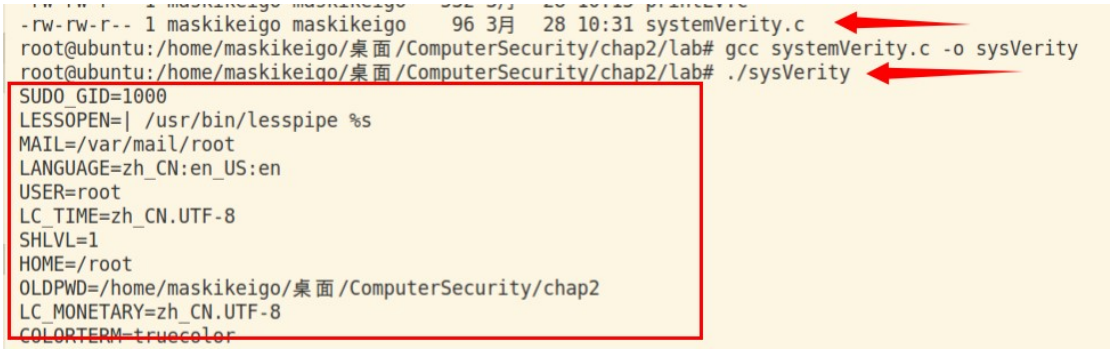
```
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task3$ ls -l
总用量 24
-rwxrwxr-x 1 maskikeigo maskikeigo 16752 3月   28 16:02 task3
-rw-rw-r-- 1 maskikeigo maskikeigo   223 3月   28 16:02 task3.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task3$ ./task3
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task3$
```

（2）修改 execve()函数参，编译后再执行：输出当前环境变量。

（3）指令修改前后的区别在于，第二次向 execve()中第三个位置传入了参数 environ，这是个全局变量，用来指向当前进程的环境变量数组。所以指令执行后将当前环境变量传递给了执行“usr/bin/env”的新进程，进程输出了我们所见的内容。

Task 4：环境变量与 system()函数

System()函数和 execve()函数的不同之处在于，前者通过调用 shell 程序来执行命令，而非自己直接执行。更多的路径程序，使得 system()将自己的环境变量传递给了 shell，这增大了攻击面。编译结果如下：



Task 5：环境变量和 Set-UID 特权程序

（1）编写程序并编译，将其修改为特权程序。如下：

```
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# gcc printCurrEv.c -o pCE
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# sudo chown root pCE
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# sudo chmod 4755 pCE
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# ls -l
总用量 84
-rw-r--r-- 1 root       root        2330 3月   28 10:19 child2.txt
-rw-r--r-- 1 root       root        2332 3月   28 10:34 child3.txt
-rw-r--r-- 1 root       root        2330 3月   28 10:07 child.txt
-rwsr-xr-x 1 root       root       16776 3月   28 10:41 pCE
-rw-rw-r-- 1 maskikeigo maskikeigo   179 3月   28 10:41 printCurrEv.c
-rwxr-xr-x 1 root       root       16888 3月   28 10:16 printEV
-rw-rw-r-- 1 maskikeigo maskikeigo   532 3月   28 10:15 printEV.c
-rw-rw-r-- 1 maskikeigo maskikeigo    96 3月   28 10:31 systemVerity.c
-rwxr-xr-x 1 root       root       16704 3月   28 10:32 sysVerity
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab#
```

（2）在普通用户下，使用 export 指令设置环境变量:export EVexample=IAmChinese，运行特权程序 pCE 结果如下所示：

```
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/la
b# export EVexample=IAmChinese
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/la
b# ./pCE
SHELL=/bin/bash
COLORTERM=truecolor
SUDO_GID=1000
EVexample=IAmChinese
LANGUAGE=zh_CN:en_US:en
LC_ADDRESS=zh_CN.UTF-8
LC_NAME=zh_CN.UTF-8
SUDO_COMMAND=/bin/bash
LC_MONETARY=zh_CN.UTF-8
SUDO_USER=maskikeigo
PWD=/home/maskikeigo/桌面/ComputerSecurity/chap2/lab
LOGNAME=root
XAUTHORITY=/run/user/1000/gdm/Xauthority
HOME=/root
LANG=zh_CN.UTF-8
LC_PAPER=zh_CN.UTF-8
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
=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=0
01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;3
:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;3
1:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;3
```

使用指令 diff 对程序输出结果进行比较：如下。可知，在 pCE 的 EV 中多了上面由 export 指令新建的环境变量 EVexample。亦即，子进程 pCE 继承了父进程中 shell 经 export 导出的 EV。

```
-rwsr-xr-x 1 root       root       16776 3月   28 10:41 pCE
-rw-rw-r-- 1 maskikeigo maskikeigo   179 3月   28 10:41 printCurrEv.c
-rwxr-xr-x 1 root       root       16888 3月   28 10:16 printEV
-rw-rw-r-- 1 maskikeigo maskikeigo   532 3月   28 10:15 printEV.c
-rw-rw-r-- 1 maskikeigo maskikeigo    96 3月   28 10:31 systemVerity.c
-rwxr-xr-x 1 root       root       16704 3月   28 10:32 sysVerity
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# ./pCE > child3.txt
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# diff child3.txt child2.txt
4d3
< EVexample=IAmChinese
34c33
< _=./pCE
---
> _=./printEV
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab#
```

Task 6：路径环境变量与 Set-UID 特权程序

（1）编译程序，并运行。结果如下图所示：

```
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# ls
child2.txt  child.txt  printCurrEv.c  printEV.c       sysVerity  task6.c
child3.txt  pCE        printEV        systemVerity.c  task6
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# ./task6
child2.txt  child.txt  printCurrEv.c  printEV.c       sysVerity  task6.c
child3.txt  pCE        printEV        systemVerity.c  task6
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab#
```

（2）将该程序升级为特权程序，并试图令其执行其他命令。

```
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# sudo chown root task6
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# sudo chmod 4755 task6
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab# ls
child2.txt  child.txt  printCurrEv.c  printEV.c       sysVerity  task6.c
child3.txt  pCE        printEV        systemVerity.c  task6
root@ubuntu:/home/maskikeigo/桌面/ComputerSecurity/chap2/lab#
```

方法：由于该程序并没有提供访问 ls 指令的绝对路径，我们可以通过修改 PATH 环境变量使得特权程序执行其他指令而非真正的 ls 命令。如，直接获取 root 权限。

首先，新建我们想要执行的间谍程序 cal.c，如下所示：

```c
#include <stdlib.h>

int main(){

    system("/bin/bash -p");
    return 0;
}
```

没有执行预想中的攻击效果，重复试验也是一样。

```
-rw-rw-r-- 1 maskikeigo maskikeigo    77 3月   28 13:22 cal.c
-rwsr-xr-x 1 root       maskikeigo 16696 3月   28 15:18 task6
-rw-rw-r-- 1 maskikeigo maskikeigo    86 3月   28 13:03 task6.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task6$ sudo ln -sf /bin/zsh /bin/sh
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task6$ ./task6
cal.c  task6  task6.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task6$ gcc cal.c -o cal
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task6$ export PATH=.:$PATH
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task6$ echo $PATH
ja:/home/maskikeigo/gn:/home/maskikeigo/opt/gcc_riscv32/bin:/home/maskikeigo/ninja:/home/maskikeigo/gn
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task6$ ./task6
cal  cal.c  task6  task6.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task6$ sudo rm /bin/sh
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task6$ sudo ln -s /bin/zsh /bin/sh
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task6$ ./task6
cal  cal.c  task6  task6.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task6$ export PATH=.:$PATH
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task6$ echo $PATH
.:.:/opt/ros/noetic/bin:/home/maskikeigo/opt/gcc_riscv32/bin:/home/maskikeigo/ninja:/home/maskikeigo/g
/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task6$ ./task6
cal  cal.c  task6  task6.c
```

Task 7：LD_PRELOAD 环境变量与特权程序

（1）将程序 mylib.c 编译成 DLL。

```
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$ ls
mylib.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$ gcc -c mylib.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$ gcc -shared -o libmylib.so.1.0.1 mylib.o
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$ export LD_PRELOAD=./libmylib.so.1.0.1
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$
```

（2）在 4 种情况下，运行 prog 程序，如下所示：

1，普通用户，普通程序：可以调用 mylib 链接库。

```
-rw-rw-r-- 1 maskikeigo maskikeigo  1696 3月   28 13:44 mylib.o
-rwxrwxr-x 1 maskikeigo maskikeigo 16696 3月   28 13:41 myprog
-rw-rw-r-- 1 maskikeigo maskikeigo    82 3月   28 13:41 myprog.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$ ./myprog
I am not sleeping!
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$
```

2，普通用户，特权程序：不可调用 mylib 链接库。

```
-rw-rw-r-- 1 maskikeigo maskikeigo  1696 3月   28 13:44 mylib.o
-rwsr-xr-x 1 root       maskikeigo 16696 3月   28 13:41 myprog
-rw-rw-r-- 1 maskikeigo maskikeigo    82 3月   28 13:41 myprog.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$ ./myprog
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$
```

3，root 用户，特权程序：不可调用 mylib 链接库。

```
-rwsr-xr-x 1 root       maskikeigo 16696 3月   28 13:41 myprog
-rw-rw-r-- 1 maskikeigo maskikeigo    82 3月   28 13:41 myprog.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$ export LD_PRELOAD=./libmylib.so.1.0.1
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$ sudo ./myprog
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$
```

4，export 普通用户，特权程序：可以执行。（这和 1 有啥区别？）

```
-rwxr-xr-x 1 maskikeigo maskikeigo 16696 3月  28 13:41 myprog
-rw-rw-r-- 1 maskikeigo maskikeigo    82 3月  28 13:41 myprog.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$ export LD_PRELOAD=./libmylib.so.1.0.1
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/loader$ ./myprog
I am not sleeping!
```

（3）针对步骤（2）中现象的解释，可以通过分别执行下面 4 种情况来分析：

1，不使用 export，运行普通程序 myprog→不调用；

2，不使用 export，运行特权程序 myprog→不调用；

3，使用 export，运行普通程序 myprog→调用；

4，使用 export，运行特权程序 myprog→不调用。

由此可以知道，由于在 shell 中运行的程序是通过 shell 执行 fork()生成的子进程，其可以继承自 shell 的 EV 只有 2 种：一是从环境变量转换而来的 shell 变量，一是 export 导出的 shell 变量。故 myprog 只有继承了 export 的 shell 变量，才能调用 mylib 库。特权程序由于存在保护机制，其在链接时会默认屏蔽 LD_PRELOAD 等有风险的环境变量，因而始终不会调用库。

Task 8：使用 system()或 execve()执行外部程序调用

（1）编译程序，并将其升级为特权程序。你能保证系统完整性吗？

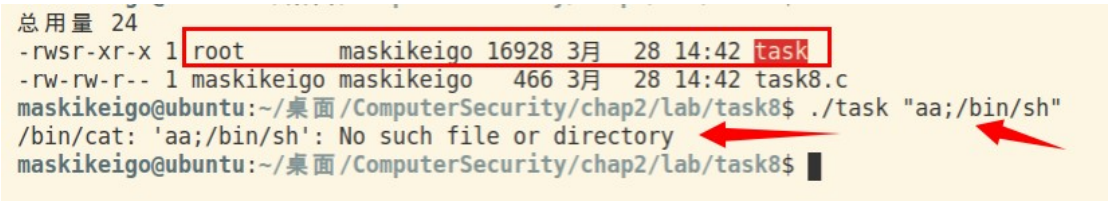
```
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task8$ sudo chown root task8
[sudo] maskikeigo 的密码：
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task8$ sudo chmod 4755 task8
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task8$ ls -l
总用量 24
-rwsr-xr-x 1 root       maskikeigo 16928 3月  28 14:24 task8
-rw-rw-r-- 1 maskikeigo maskikeigo   416 3月  28 14:21 task8.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task8$
```

无法保证，因为程序使用 system()执行指令，事实上 system()本身并不执行，而是调用 shell 执行该命令。Shell 的特性在于可以输入多条指令，且这些指令用分号分隔即可。理论是这样的，应该会弹出 root shell，这里没有。可能是我的系统版本为 20.04.

如下所示：

```
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task8$ ls -l
总用量 24
-rwsr-xr-x 1 root       maskikeigo 16928 3月  28 14:37 task8
-rw-rw-r-- 1 maskikeigo maskikeigo   446 3月  28 14:37 task8.c
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task8$ ./task8 "aa;/bin/sh"
maskikeigo@ubuntu:~/桌面/ComputerSecurity/chap2/lab/task8$
```

（2）将 system()调用改为 execve()调用后，步骤 1 中攻击方式失效。如下所示：

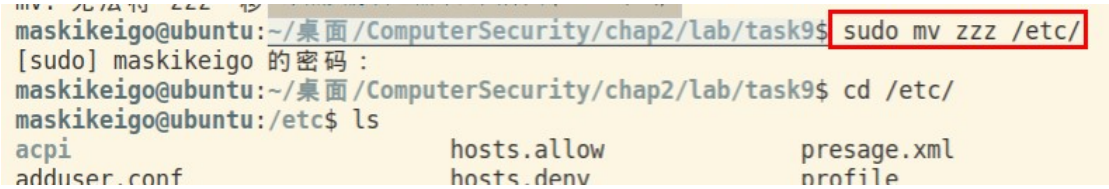


原因在于，execve()直接请求 OS 执行命令（而不是调用 shell 执行），它将整个输入命令当做参数，而不是多个指令。故查找不到“aa;/bin/sh”指令。

Task 9：权限泄漏

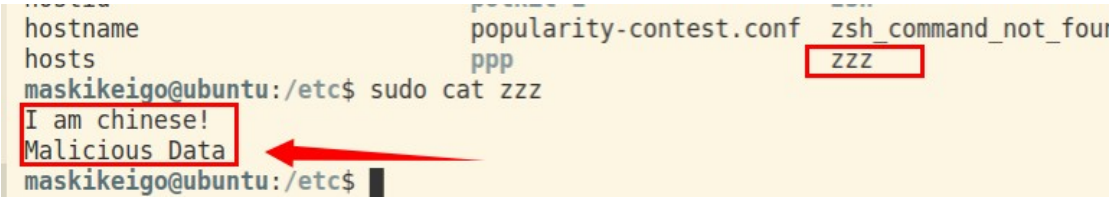
（1）编译程序，并将其修改为特权程序。在普通用户下运行并观察。

先创建文件 zzz，并存放在/etc/目录下。



编译并运行程序，将其修改为特权程序。

执行结果：程序暂停 1s 后，对 zzz 文件进行了写入操作，增加量一行文字，如下图所示：



原因：观察源码可知，在执行 write 操作之前，程序没有执行 close(fd)命令关闭文件描述符，导致权限泄漏，可以利用特权程序执行写入操作。