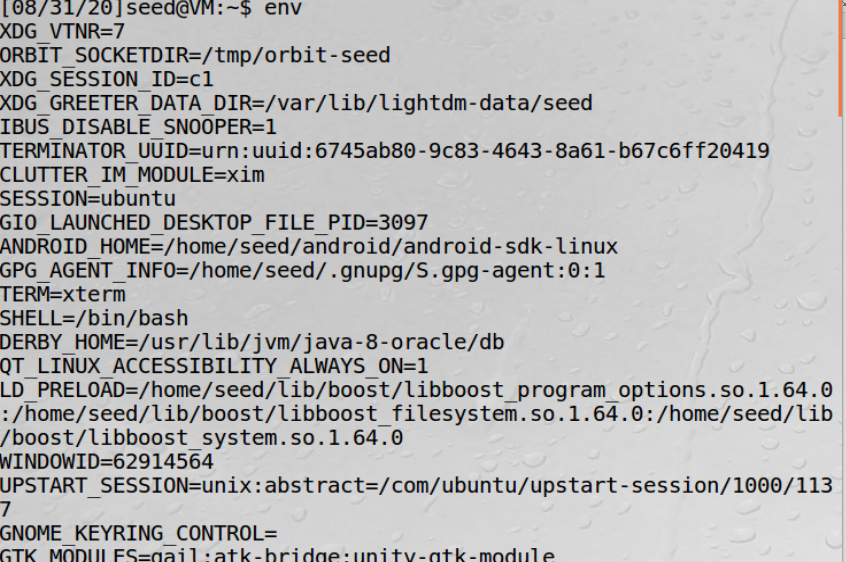
**Task 1**

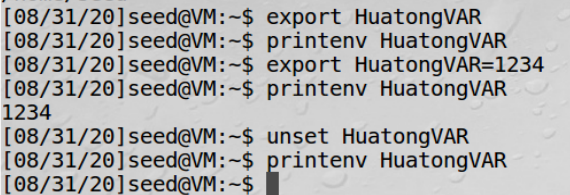
1.



2



3



**Task 2**

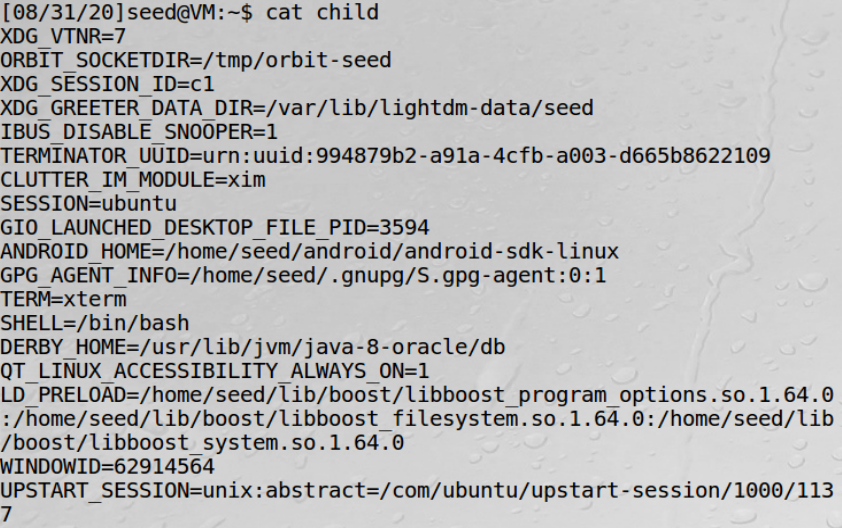
1.编写、编译、执行代码，并将输出重定向到child文件中



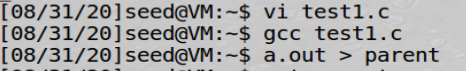




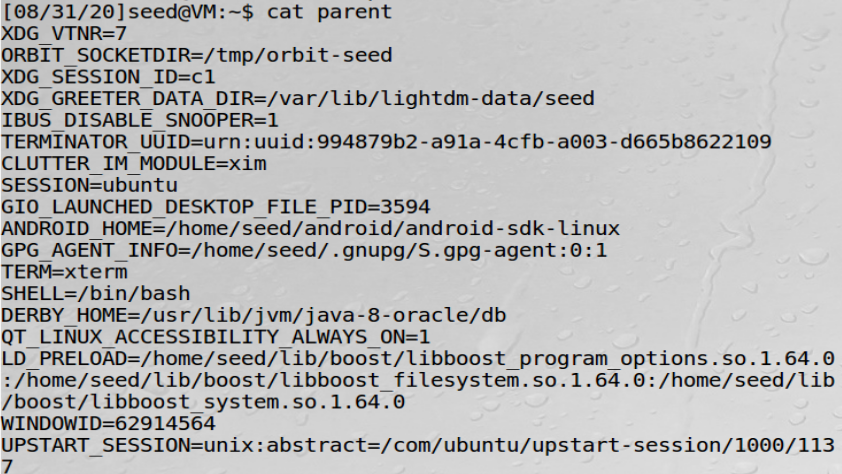
2.查看child文件中的内容，即查看子进程的环境变量



3.按题目要求修改代码，重新编译执行，并将输出重定向到parent中



4.查看parent文件中的内容，即查看父进程进程的环境变量



5.比较child和parent两个文件的内容



结论：子进程和父进程环境变量相同

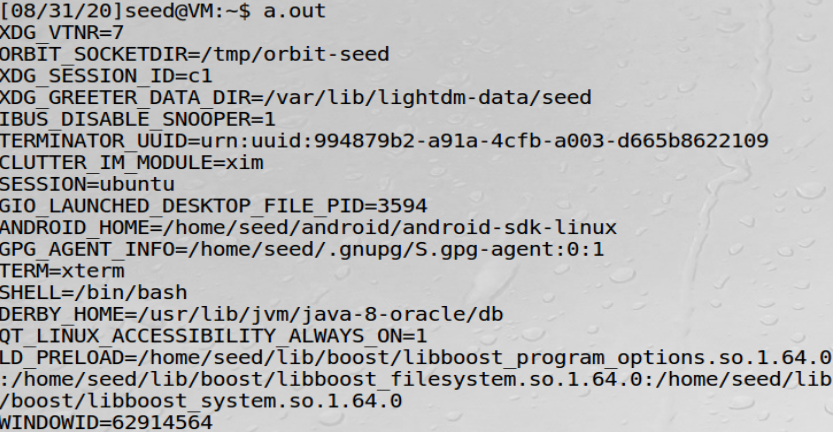
**Task 3**





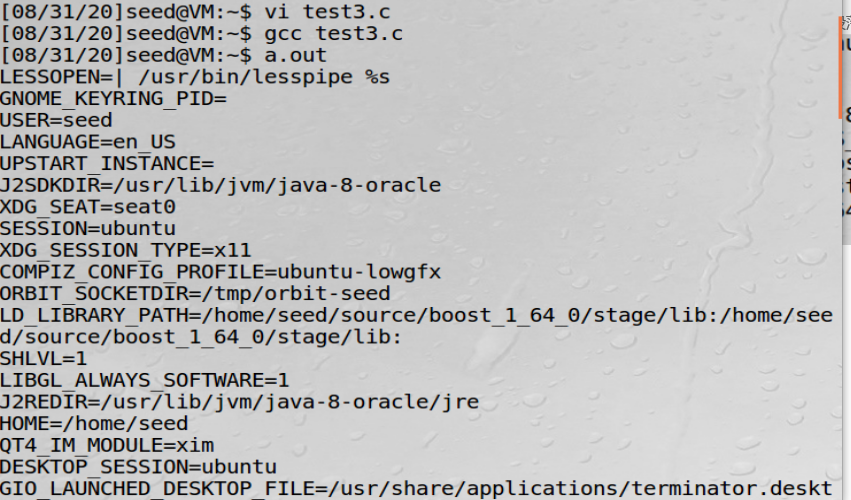
没有输出环境变量

按题目要求修改代码后，程序输出环境变量



结论：execve（）函数通过参数设置环境变量

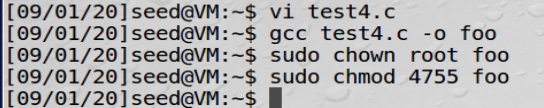
**Task 4**

1. 

经验证，原进程的环境变量被传递给了使用system（）创建的新程序

**Task 5**

1.编译代码，将程序拥有者改为root，并使他成为Set-UID程序

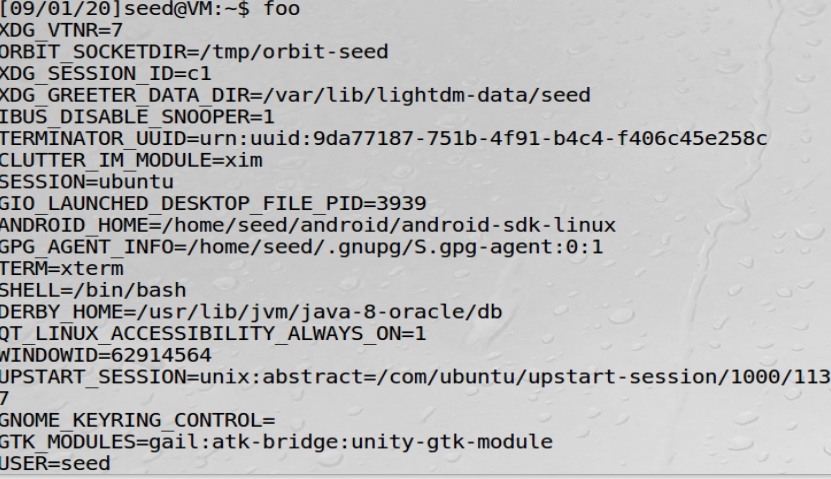


2.使用export命令设置以下环境变量





3.运行SET-UID程序foo，查看输出的环境变量：



4.



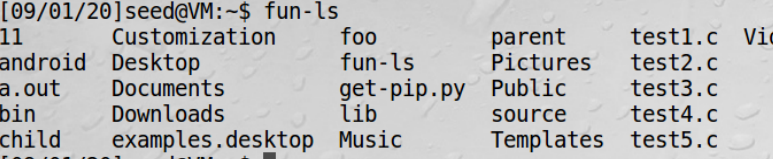
发现：在shell进程中设置的环境变量传递给了SET-UID子进程

**Task 6**

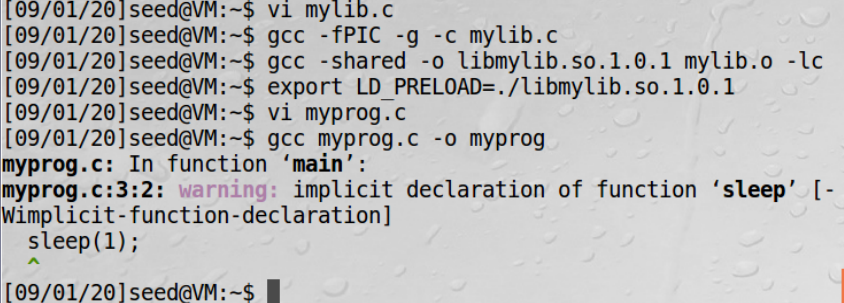




运行程序，可以发现程序执行的是ls命令



**Task 7**

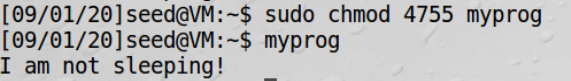


**4个情形**

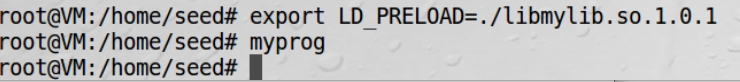
1. Make myprog a regular program, and run it as a normal user



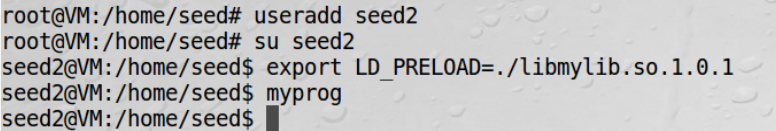
1. Make myprog a Set-UID root program, and run it as a normal user



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



1. 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.



导致他们不同的原因就在于LD\_PRELOAD环境变量以及程序所拥有的用户权限不一样。只有用户自己创建的程序自己去运行，才会使用LD\_PRELOAD环境变量，重载sleep函数，否则的话忽略LD\_PRELOAD环境变量，不会重载sleep函数。

**Task 8**

1.这个命令不安全，Bob可能会出于好奇或者个人利益驱使阅读或者修改只有root用户才可以运行的一些文件。

2.不会有效

**Task 9**

1. 创建文件



2.



3.





查看zzz



文件被改变

4.文件被修改的原因在于设置uid前，zzz文件就已经被打开了。只要将语句setuid(getuid())移至调用open函数之前，就能避免这个问题。