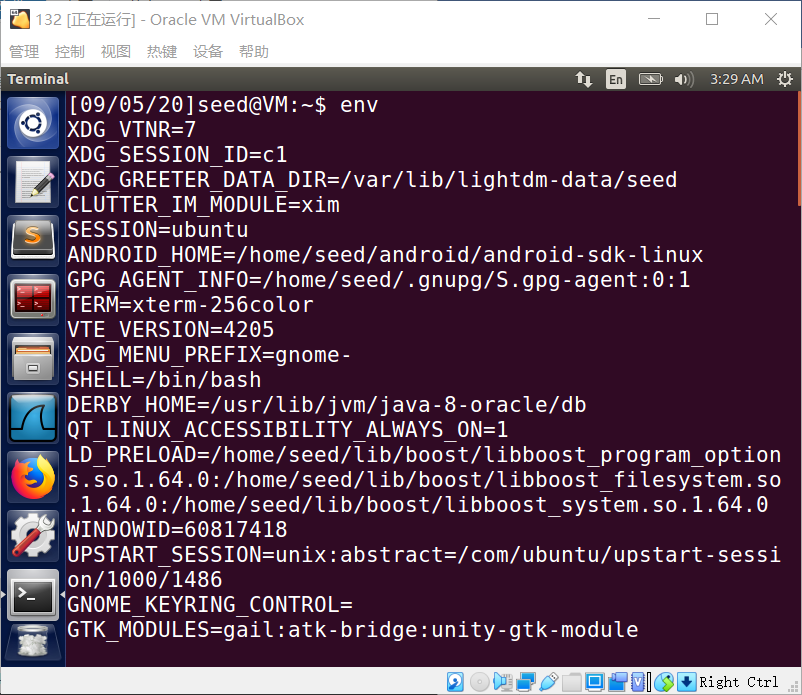
**网络空间安全实训实验报告**

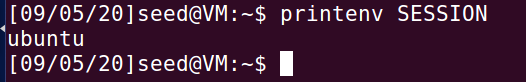
学号：57118213 姓名：陈洪杰

**Task1**

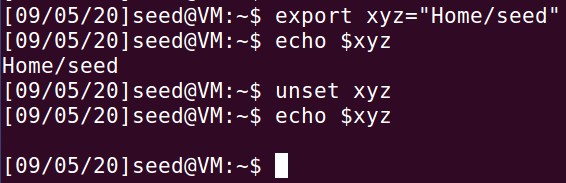
1. 输入env后，得到如下结果：



1. 随意查看一个变量，如下



1. 用export和unset设置删除环境变量

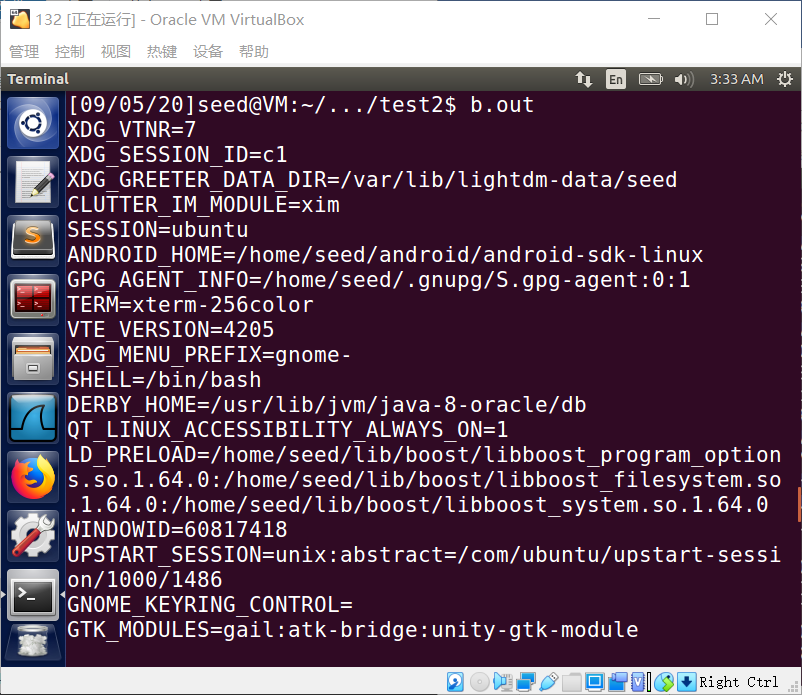
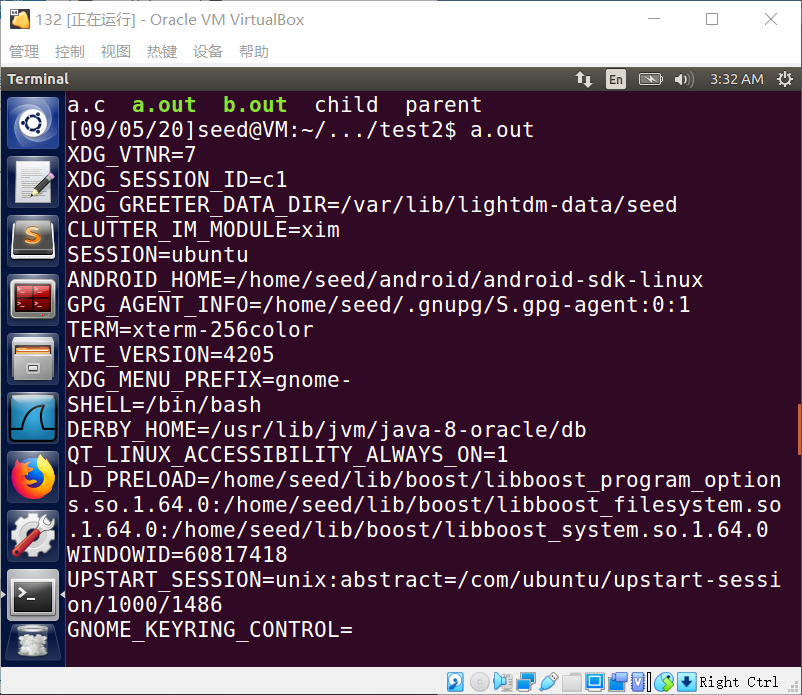


**Task2**

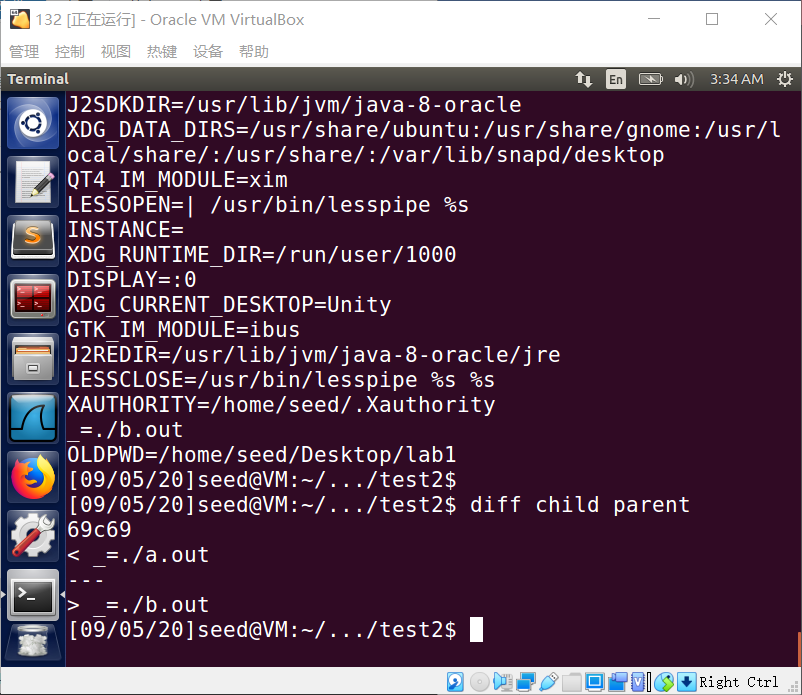
1. 输入所给代码，保存为a.c，进行step1操作后，编译为a.out，输出文件为child

进行step2操作，编译为b.out，输出文件为parent，

然后分别执行a.out和b.out得到下面两种结果，如下图：



2.使用diff命令比较child和parent文件的不同，得到下面的结果。

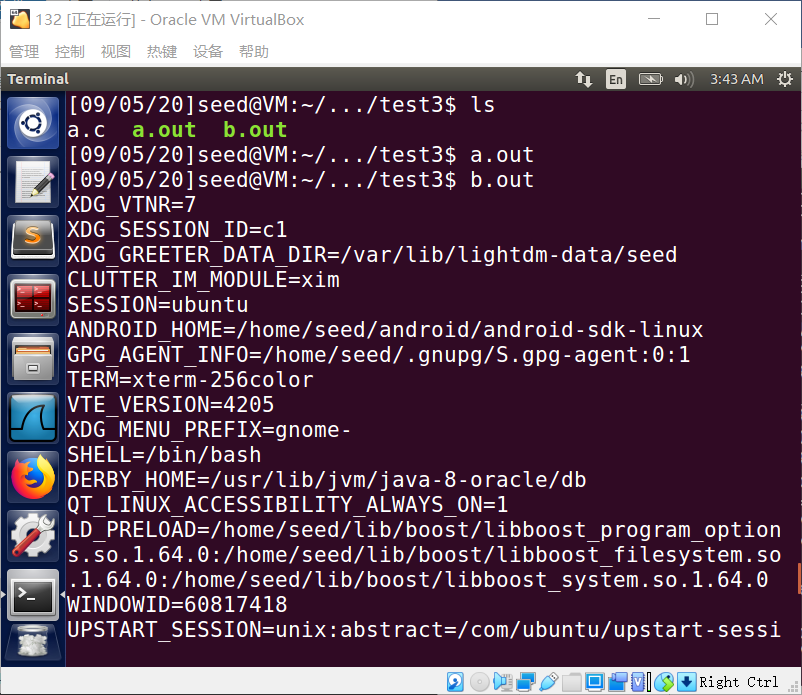


从图中可以发现，这两个输出文件除了69行以外都相同，且第69行不同的只有两个文件的名字。这说明说明原环境变量被子进程完全继承。

**Task3**

1. 输入所给代码，保存为a.c，进行step1操作后，编译为a.out

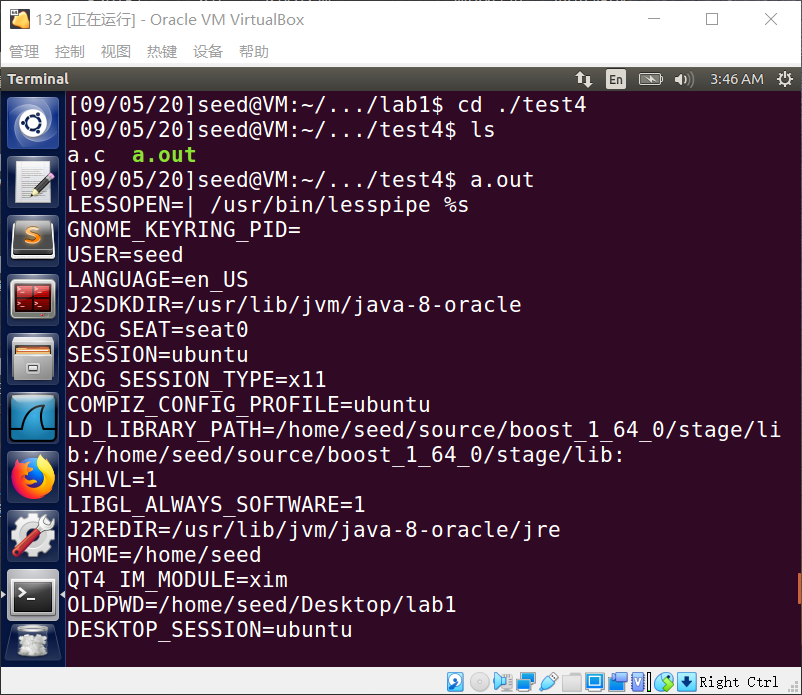
进行step2操作后，编译为b.out。执行a.out和b.out，结果如下图：



图中可以发现，a.out没有输出，而b.out正常输出。因为execve()这个函数的第三个参数表示的是传递的新环境变量。查询所给代码得知，a.out里面第三个参数为NULL，所以输出为空，而b.out里面第三个参数为environ，所以输出了环境变量。

**Task4**

1. 输入代码保存为a.c，编译为a.out，执行a.out，得到以下结果：



system（）函数执行了三步操作：

1）fork一个子进程

2）在子进程中调用exec函数去执行command

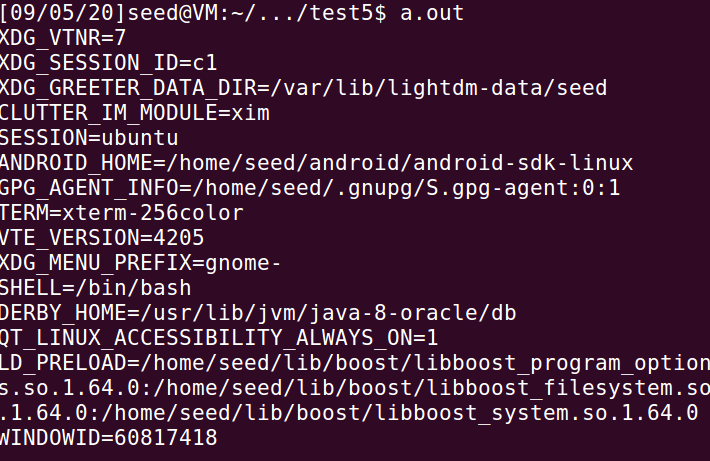
3）在父进程中调用wait去等待子进程结束。

如果exec执行成功，也即command顺利执行完毕，则返回command通过exit或return返回的值。

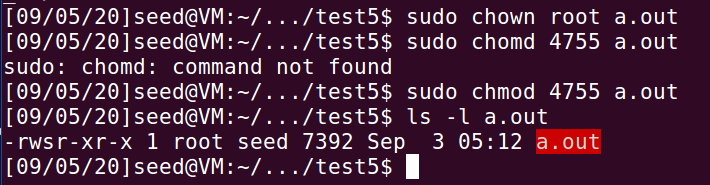
这里我们看到代码里是“/usr/bin/env”，所以返回的就是环境变量的一些值。

**Task5**

1. 输入代码保存为a.c，编译为a.out，执行a.out，得到下面结果：

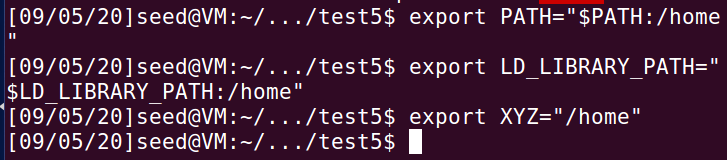


1. 利用所给的命令更改a.out的权限，如下图

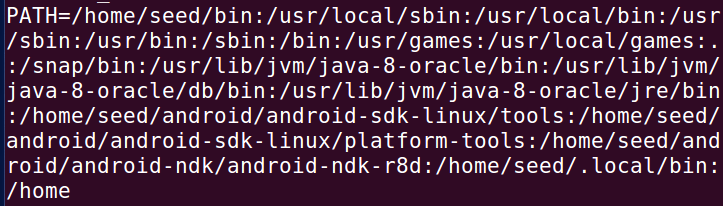


发现已经成为一个Set-UID程序。

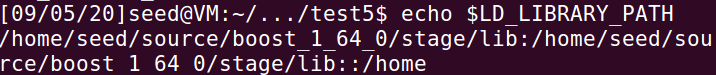
1. 用export设置三个环境变量。



再次执行a.out，得到



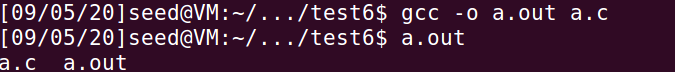




从上图中发现，更改的变量都成了我们设置的路径。

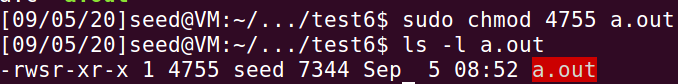
**Task6**

1. 输入代码存为a.c，编译为a.out，执行a.out，得到



更改所有权

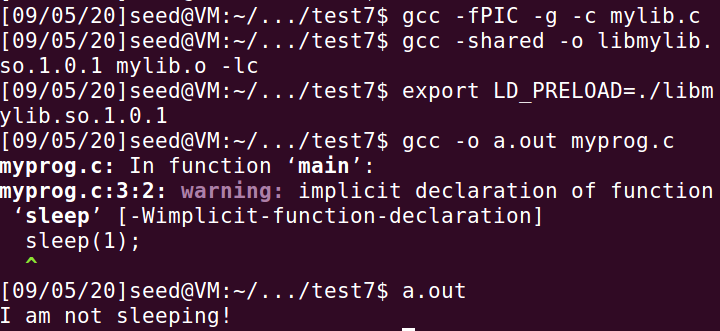




如果要让这个代码运行自己的ls的话，应该在该目录下新建一个ls，这样运行的时候先找到的就是这个ls。

**Task7**

1. 先建立mylib.c，然后根据所给命令设置好，将myprog.c编译为a.out，最后执行a.out，结果如图



正常输出了myprog.c的内容。

1. 以普通用户的身份运行程序时，输出：I am not sleeping!

以普通用户运行拥有root权限的Set-UID程序时，无输出。

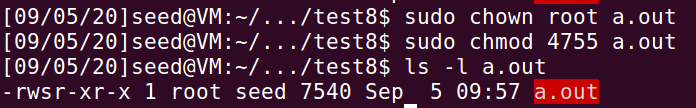
将a.out设置为拥有root权限的Set-UID程序，在root用户下重设LD PRELOAD环境变量并运行，无输出。

在user2用户中再次设置LD\_PRELOAD环境变量并运行拥有root权限的Set-UID程序时，输出：I am not sleeping!

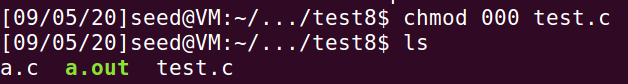
1. 输出不同主要是因为LD\_PRELOAD这个环境变量，设置了自己的路径，所以链接的动态库解释自己写的mylib.c，所以会出现I am not sleeping!而如果有了root权限，就会使用原来的sleep（）函数。

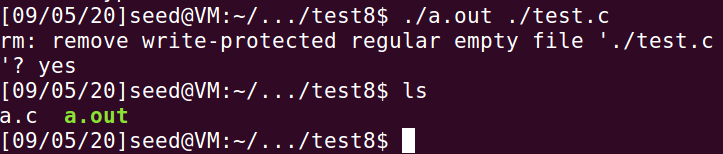
**Task8**

1. 输入代码保存为a.c，编译为a.out，将a.out改为有root权限的set-uid程序



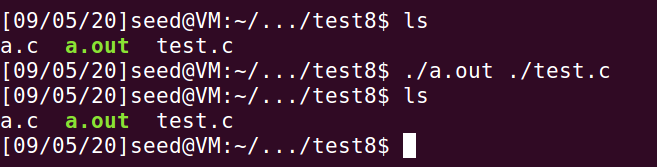
1. v[0] = "/bin/cat"改为v[0] = "rm",即删除命令。并新建一个test.c文件，将其权限改为000





可以发现test.c被删除掉了。

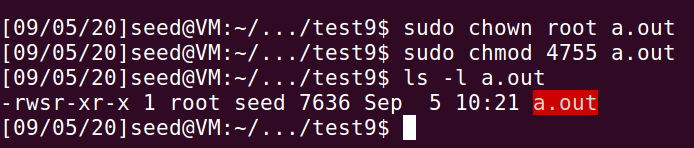
1. 修改注释的部分，再次进行删除操作



发现这一次没有删除成功。

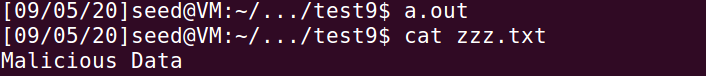
**Task9**

1. 将代码保存为a.c，编译为a.out，将a.out改为有root权限的set-uid程序



1. 创建zzz.txt，执行a.out，观察zzz.txt的内容





发现能够成功写进去了，因为最后没有关闭文件，所以可以写进去。