第一次实验报告

Environment Variable and Set-UID Program Lab

实验人: 57117104 孙舒雯

实验时间：9.01-9.03

1. Task 1: Manipulating Environment Variables

实验过程：

1.使用 printenv 或者 env 命令打印环境变量；

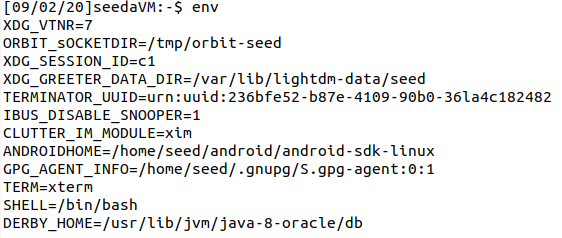
2.使用 export 和 unset 命令设置和取消环境变量

实验结果如图:

1.输入 printenv 命令



输入 env 命令

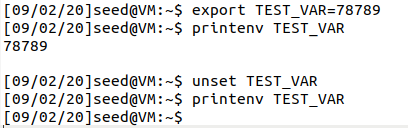


1. 用 export 导入一个环境变量 TEST\_VAR(:/home)



用 printenv 查看设置有没有成功：

unset 取消环境变量，再次查看 TEST\_VAR



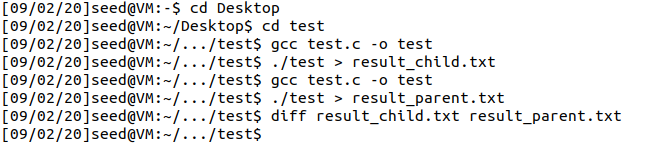
由于环境变量已被取消，并无打印结果。



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

实验过程：

打印父进程、子进程的环境变量，比较结果

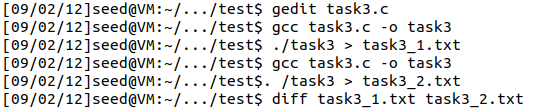


结果：用 diff 命令对比结果，无任何输出，表示父进程和子进程的环境变量完全相同， fork()系统调用生成的子进程继承了父进程的环境变量。

1. Task 3: Environment Variables and execve()

实验过程：用 execve 方法执行程序/usr/bin/env，改变第三个参数(envp)对比输出结果。

改变 envp 参数编译程序



结果：当 envp 参数为 NULL 时，出结果为空。当 envp 参数被指定

时，环境变量为 envp 数组指定的参数序列。总之，execve()在覆写调用进程时，会根据传入参数的情况改变进程的环境变量值。

1. Task 4: Environment Variables and system()

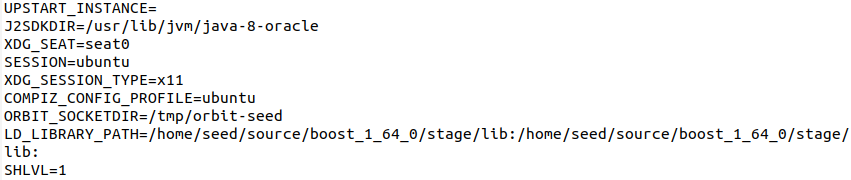
实验过程：

用 system()函数执行一个新程序，验证新程序的环境变量是否与调用程序相同。

参考手册编译运行如下代码：



部分结果：



用 system()函数执行一个新程序后，新程序环境变量确实与调用程序的环境变量相同。

1. Task 5: Environment Variable and Set-UID Programs

实验过程：改变编译程序拥有者，设置为 Set-UID 程序，观察环境变量变化。

编译手册中代码，改变可执行程序的权限和所有者：



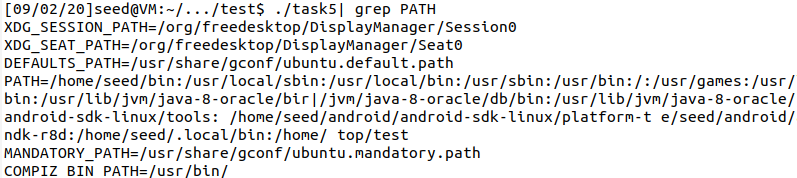
修改环境变量 PATH,LD\_LIBRARY\_PATH 的值，添加 SUN







结果：运行 task5 程序如图





没找到LD\_LIBRARY\_PATH 。

1. Task 6: The PATH Environment Variable and Set-UID Programs

实验过程：改变path环境变量，观察编译程序运行结果。

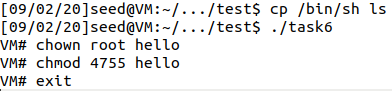
编译运行 task6.c 文件，修改环境变量PATH，在最前端插入当前工作目录。



修改/bin/sh并复制到当前目录，命名成ls

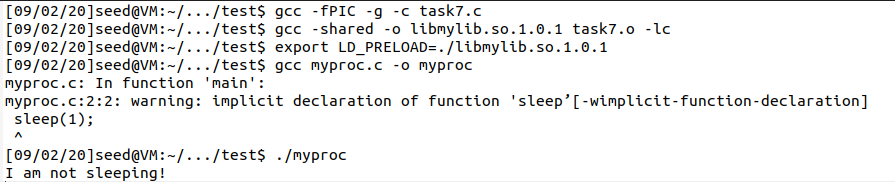


此时不需sudo命令可直接执行chown和chmod命令，因为setUID 程序生成的shell具有root 权限。

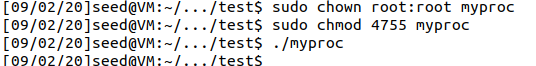


1. Task 7: The LD PRELOAD Environment Variable and Set-UID Programs

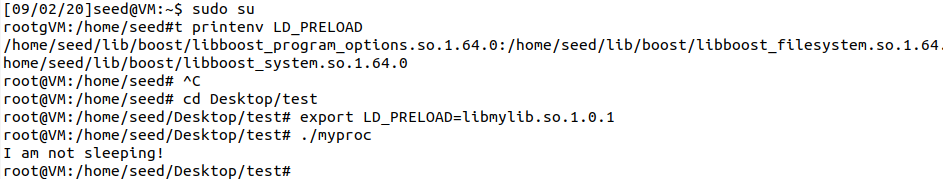
实验过程：写出sleep 函数并且生成动态链接库，修改LD\_PRELOAD 环境变量的值，编译运行 myproc 程序



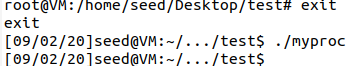
将 myproc 可执行程序变为 set-UID root 程序



由于myproc程序对应的LD\_PRELOAD环境变量并非seed用户下所设置的LD\_PRELOAD值所以没有输出。进入root 用户，修改root用户下的LD\_PRELOAD值，再次运行myproc 。

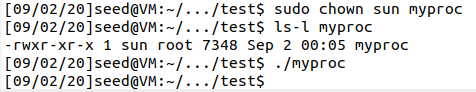


退出 root 用户



myproc是root Set-UID 程序，是真正的 sleep 函数

把程序的所有者改为SUN运行 myproc



输出为空。因此，只有ruid和euid相同时，LD\_PRELOAD变量才有效

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

实验过程：参考实验手册内容编译 task8 文件，设置为 set-root-UID 程序。为了做对比试验，在 task8 可执行程序的工作目录下创建con1 目录，存放一个名为testing 的普通文件，将 con1 和testing的权限设为 700，所有用户为 root。

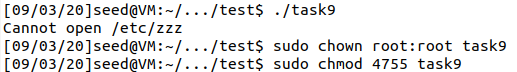
实验结果：搞不太清楚了。。。

1. Task 9: Capability Leaking

实验过程：在/etc下创建 zzz 文件，只有root 用户才可以写zzz。

对task9.c 进行编译、权限修改，Set-UID。

结果：



因为在 task9 可执行程序运行时打开了zzz 文件，在执行完相关任务之后没有在撤销特权前及时关闭 zzz 的文件描述符 fd，fd 还具有 root 特权 ，因此zzz 文件被修改了。