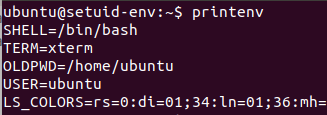
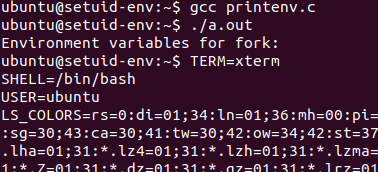
Environment Variable and Set-UID Program Lab

Task 1:

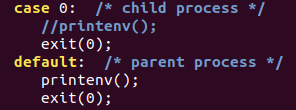


Ran printenv to print out environmental variables.

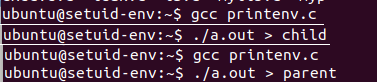
Task 2:



Compiled printenv.c and ran the resulting code.



Edited printenv.c to printenv() of parent process and not of the child process

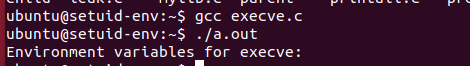


Saved both compiled versions into files of different names.

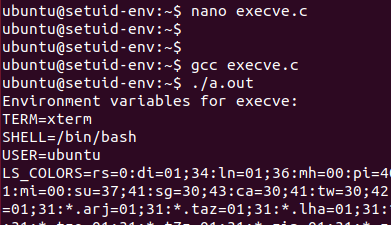


Ran diff which displayed nothing. This shows that the child inherited env variables from its parents.

Task 3:

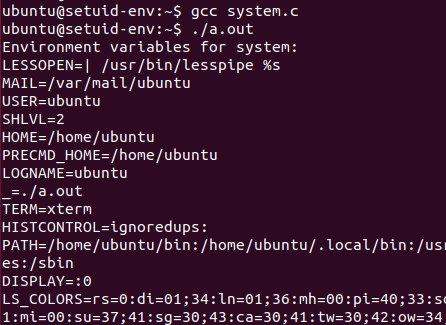


Compiled and ran execve.c. No environmental variables shown.



Edited exevce.c as instructed. Environmental variables are now appear in the new program.

Task 4:



Compiled and ran system.c.

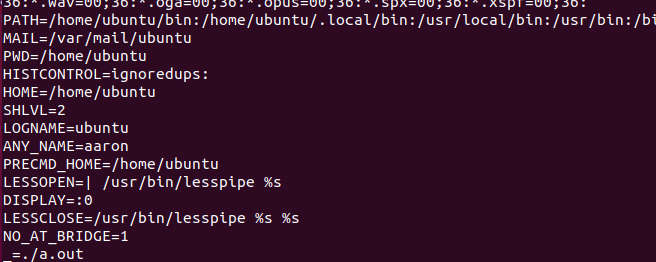
Task 5:



Compiled printall.c. Changed ownership to root and made it a set-uid program.

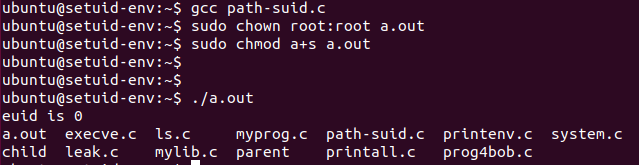


Exported PATH, LD\_LIBRARY\_PATH and my name “aaron”



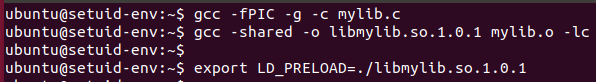
Upon running a.out, the exported variables are correctly shown.

Task 6:



Compiled path-suid.c. Changed ownership to root and made it a set-uid program. a.out ran the default ls command.

Task 7:



Compiled mylib.c and set LD\_PRELOAD environmental variable.



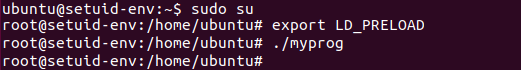
Compiled myprog.c



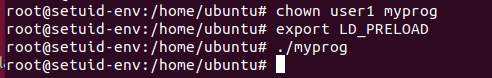
Ran myprog as normal user -> not sleeping.



Ran myprog as set-uid program -> slept for 1 second

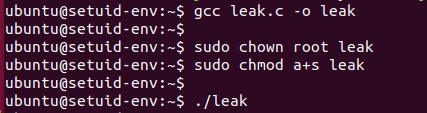


Switched user to root. Exported LD\_PRELOAD and ran program again -> slept for 1 second

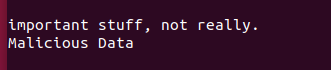


Set owner to user1. Exported LD\_PRELOAD and ran program again -> slept for 1 second.

Task 8:



Compiled leak.c. Made its owner root and made it a set-uid program.



After running it, file zzz had malicious data injected into itself.