# Assignment1: SEED Labs – Environment Variable and Set-UID Program

Mayur Suresh

CSC 302-01: Computer Security

Date: February 19th, 2023

## Task 4: Environment Variables and system()

For this task we will see the behavior of using the system() function with another program. We will use this function to execute a command. Unlike the execve() function which will directly execute a command the system() function will invoke the `/bin/`sh -c` command.

We will first run and compile a program that utilizes the system() function and we see the environment variables below.

Text

Description automatically generatedText

Description automatically generated

## Task 5: Environment Variables and Set-UID Programs

Set-UID is a critical security mechanism in Unix OS, as it assumes the owner’s privileges when running. It can gain root privileges then this program is run if the owner is in root.

We first write a program that will print out the environment variables with this process and compiled as shown below.

Text

Description automatically generated Text

Description automatically generated

We then run the following commands with the binary file outputted from the program.

Graphical user interface, text

Description automatically generated with medium confidence

We all so see that we have switched the ownership to root below.

Text

Description automatically generated

When then using the export commands set the PATH, LD\_LIBRARY\_PATH and a PATH defined by use like below and then see it listed in the environment variables.

Text

Description automatically generated

We now run the Set-UID again with the same process as before and wee see that nothing happens. I am very surprised by this because I was under the assumption that I would be sent over to root, but I stayed in seed. I now run the binary file again and only see the environment variables displayed again.

## Task 6: Environment Variables and Set-UID Programs

For this task we will change the path address as it is dangerous to call system() inside a Set-UID program.

First, we will change the PATH using export, setting it to the `/home/seed` directory to the beginning of the PATH environment variable. As shown with the environment variables below.

Text

Description automatically generated

We now set up a Set-UID program that executes the `/bin/sh` command. We compile the program, this program changes the owner to the root with the chown/chmod commands and this makes it a Set-UID program.

Text

Description automatically generated

When we run the binary file produced by the program, we see that it executes as it supposed showing the files under the current directory shown below.

Text

Description automatically generated

We next go ahead and change the command called from `/bin/sh` to one of our own. I will choose the `pwd` command to run this program as shown below. We see that with the system() and changing the owner from seed to root that it acts just like a Set-UID program would behave.

Text

Description automatically generated Text

Description automatically generated

## Task 7: The LD\_PRELOAD Environment Variable and Set-UID Programs

For this task we will see how the Set-UID program deals with many environment variables such as LD\_LIBRARY\_PATH and the LD\_PRELOAD.

We will now see how these environment variables changes the behavior of dynamic loader/linker when running a program. First, we build a dynamic link library with a program that overrides the sleep() function in libc.

Text

Description automatically generated

Then we set the LD\_RELOAD to the ` ./libmylib.so.1.0.1` address. Next, we compile the program the program myprog.

We now run the myprog program which is a normal program in many ways to see the difference. We first run the program as a normal user, then make the myprog a Set-UID root program by switching its owner from seed to root and then export the LD\_PRELOAD environment variable and then run the program.

Text

Description automatically generated 1) We first run it normally

Text

Description automatically generated 2) We then make it a Set-UID program and run it as a normal user.

Text

Description automatically generated 3) Finally, we export the variable LD\_PRELOAD in the root and run it again

There is a difference in between all these scenarios because it is all about the environment variables. In the first step be export the LD\_PRELOAD when we are in seed and then we run it and this is why we are able to see the string. However, when we switch the program’s owner to the root and run it like in second step we are not able to see the string as we only exported the environment variable in the seed at that time. So we go into the root export the environment variable and run it we see that string again. So, it all comes down to then the LD\_PRELOAD is set.

## Task 8: The LD\_PRELOAD Environment Variable and Set-UID Programs

In this task we will work with a scenario where we use the program given is compiled and used to explain how to help Bob.

First we create a catall.c and compile the program and change its owner to root. We see below running the program after making it a Set-UID program by giving it a sample file. We are able to see the file from the root.

Text

Description automatically generated

But with just with the program provided Bob is not able to modify or edit any files. Hence, Bob is not able to delete a file that is not writable to us. This program does not give us permission or the ability when the program is on in the root, because the file specified is not owned by the root.

We no switch from the system() to the execve() and compile it after making it a Set-UID program.

Text

Description automatically generated Text

Description automatically generated

Even after replacing the system() with execve() there is no change as all it does is start a new process but does not give anymore permissions. So, this program still only gives us the access to read a file and not edit files or delete them as we are not root users. So, the abilities with using the system() is shared by execve().