Attacks Through Environment Variables

2.1. What is the difference between environment variables and shell variables?

2.2. In Bash, if we run "`export foo=bar`", does it change the environment variable of the current process?

2.3. The followings are two different ways to print out environment variables. Please describe their differences:

```
$ /usr/bin/env
$ /usr/bin/strings /proc/$$/environ
```

2.4. In our code, when we use `execve()` to execute an external program `xyz` , we pass `NULL` in the third argument. How many environment variables will the process running `xyz` has?

2.5. Bob says that he never uses any environment variable in his code, so he does not need to worry about any security problem caused by environment variables. Is he correct?

2.6. A program `abc` invokes an external program `xyz` using `system()`, which is affected by the `PATH` environment variable. When we invoke `abc` from a shell prompt, how does the shell variable `PATH` in the current shell end up affecting the behavior of the `system()` function?

2.7. Please explain why using `secure_getenv()` is better than using `getenv()`.

2.8. A privileged `Set-UID` program needs to find out which directory it is currently in. There are two typical approaches. One is to use the `PWD` environment variable, which contains the full path of the current directory. Another approach is to use the `getcwd()` function (you can find its manual online). Please describe which approach you would like to take and why.

2.9. In Linux, many environment variables are ignored if the program by the dynamic linker if the program to be executed is a `Set-UID` program. Two such examples are `LD_PRELOAD` and `LD_LIBRARY_PATH`. Please read the manual of `ld-linux` (`https://linux.die.net/man/8/ld-linux`) and explain why the following environment variables are also ignored:

- `LD_AUDIT`
- `LD_DEBUG_OUTPUT`

2.10. There are two typical approaches for letting normal users do privileged tasks, one is to write a root-owned `Set-UID` program, and let the user run; another approach is to use a dedicated root daemon to do those privileged tasks for users. Please compare the attack surfaces of these two approaches, and describe which one is more secure.