CIS 751 Lecture Assignment 6

Chuck Zumbaugh

October 15, 2025

1 Buffer Overflows

- Fail safe defaults - The default behavior for the functions that create these vulnerabilities is to just write past the buffer. While it is difficult to remove language features, modern compilers and IDEs can provide warnings, or in some configurations, throw a compile-time error when this is done.
- Least common mechanism - In the OS we were working with, the stack (and heap) is readable, writable, and executable. This, along with the ubiquitous use of these throughout program execution provides many opportunities for something to go wrong.
- Work factor - This is dependent on the operating system and any safeguards that have been put in place to prevent this. On RedHat 8, it is quite simple to execute a buffer overflow as the buffer size is easy to obtain and the addresses are stable. Features such as ASLR, a non-executable stack, and canaries make this more difficult.

2 Environment Variables

- Least Privilege - We are exploiting a vulnerability (either through the dynamic linker or by changing the PATH variable) in a set UID program to obtain a root shell. This can be resolved by responsible use of privilege escalation within a program, using execve() rather than system(), and preventing the child process from inheriting environment variables such as LD_PRELOAD in a set UID program.

- Complete mediation - There is no re-authentication required when the root shell is created. This could be fixed by requiring an admin password every time a root shell is created.
- Work factor - Without the protections in modern operating systems, it is trivial to exploit a set UID program using this. Strategies such as preventing the child process from inheriting certain environment variables can help mediate this.

3 Format Strings

- Least Privilege - In the context of obtaining a root shell through a set UID program. The counter measure for this is to use escalated privileges judiciously to prevent an attacker from exploiting a vulnerability and obtaining a root shell.
- Fail safe defaults - By default, printf() and va_arg() are not aware if they reach the end of their argument list, and an attacker can therefore access and manipulate data on the stack in certain cases. This can be mitigated by not using user inputs for format strings, compiler warnings/errors, and measures that protect the stack and heap from exploitation (ASLR, canary, NX-bit).

4 Race Conditions

- Least Privilege - We are relying on the fact that the race condition is occurring in a situation when privileges are escalated. The countermeasure to this is to use the minimal privileges required.
- Work factor - exploiting a single race condition can be done relatively easily/quickly. By increasing the number of checks it becomes more difficult for the attacker to win.
- Complete mediation - open() only checks the EUID and not the RUID. If open were to check that the EUID == RUID, particularly for sensitive files such as passwd or shadow, this could be prevented.
- Fail safe defaults - By default, symbolic links will be followed. Preventing symbolic links with sensitive files could help to prevent this.