CON101: Software Vulnerabilities

Akash Suryawanshi

November 4, 2020

1 Buffer Overflow

1.1 Description

Buffer Overflow exists when we write a program which attempts to put more data in a buffer than its capacity or which attempts to put the data into a unit which is not allocated to buffer. In spite of being such a popular form of software security vulnerability, buffer overflow attacks are still common, mainly because buffer overflow can occur in so many ways and are not easy to discover. Even if discovered, they are very difficult to exploit. What happens in a buffer overflow exploit is that the attacker sends data to a program which is stored in an stack buffer of smaller size so that the data on stack is overwritten. The data sets the value of return pointer such that when function returns, it executes malicious code from the attacker. Buffer overflow typically occurs in a code that : whose functionality depends on external code. is very complex that even the programmer cannot predict its behaviour. that depends upon properties which are outside the scope of the code.

1.2 Attacks and Consequences

Attackers use buffer overflow to corrupt the stack execution of a web application. They send some specific input to a web application, which then makes the web application run some malicious code, effectively taking over the machine. Generally, buffer overflow leads to crashes, incorrect results or errors. Almost all web severs and applications are susceptible to buffer overflow.

2 Directory Traversal

2.1 Description

A directory traversal aims to access files which are stored outside the root folder. It is usually done by manipulating variables that reference files with ../ sequences and its variations or by using file paths. Directory traversal attacks are executed through web browsers. Web applications have to include local resources such as images, scripts, themes etc.

2.2 Attacks and consequences

There is a risk that the attacker might be able to include a file we did'nt authorize every time a file is included by the application. If the web server/application is vulnerable to directory traversal attack, then the attacker can the system to reach root directory access files and directories which are not meant to be accessed by others. Once accessed they can even modify or delete important files which might include keys, password files, source code etc. This can be prevented if the input is properly filtered which includes : URL, FORM Parameters, cookies, http request headers.

3 Cross-site scripting

3.1 Description

Cross-site scripting denoted by XSS is a complete group of software weaknessess that allows hackers to execute thier own malicious code in the browsers of the website visitors. The attack occurs when the victim visits the web page/application that executes the unwanted code. XSS attacks are carried out in various different languages like VBSsript, Flash, CSS etc but mostly it is javascript. The flaws that allow these attacks to succeed are wideapread and might occur where a web application uses input from a user withn the output it generates without encoding or validating it.

3.2 Attacks and consequences

Attackers usually send malicious scripts using XSS to user. The user's browser has no way to know that weather the script is harmful or not, and it will just execute it as it assumes the script came from a trused source. Hence it can access any sensitive information. XSS can cause severe problems like diclosing of the user's session cookie, allowing an attacker to hijack the users session and take over the account, disclosing of end user files, installation of Trojan horse programs, redirecting the user to some other site etc.

References

1. https://owasp.org/www-community/attacks/Path$_{T}raversal$
2. https://owasp.org/www-community/attacks/xss/
3. https://www.acunetix.com/websitesecurity/cross-site-scripting/
4. https://www.synopsys.com/blogs/software-security/detect-prevent-and-mitigate-buffer-overflow-attacks/