INJECTION ATTACKS: HOW TO PROTECT OUR DATA

Natalia GONCEARU

Universitatea Tehnică a Moldovei

Abstract: *Injection attacks are amongst the oldest and most dangerous web application attacks. Injection attacks are based on a single problem that persists in many technologies: namely, no strict separation exists between program instructions and user data. There are different kind of injection attacks like HTML Injection, JavaScript Injection and SQL Injection, by far the most dangerous because its affect our personal data. There are different kind of methods to help us to protect our data and check if our data was changed.*

Keywords: *Injection attacks, attacks, data, HTML, JavaScript, SQL.*

Introduction

Injection attacks refer to a broad class of attack vectors that allow an attacker to supply untrusted input to a program, which gets processed by an interpreter as part of a command or query which alters the course of execution of that program.

Injection attacks are amongst the oldest and most dangerous web application attacks. They can result in data theft, data loss, loss of data integrity, denial of service, as well as full system compromise.

Injection attacks are based on a single problem that persists in many technologies: namely, no strict separation exists between program instructions and user data (also referred to as user input). This problem allows for attackers to sneak program instructions into places where the developer expected only benign data. By sneaking in program instructions, the attacker can instruct the program to perform actions of the attacker's choosing.

A successful attack requires three elements:

- *Identifying the technology that the web application is running.* Injection attacks are heavily dependent on the programming language or hardware possessing the problem. This can be accomplished with some reconnaissance or by simply trying all common injection attacks. To identify technologies, an attacker can look at web page footers, view error pages, view page source code, and use tools such as nessus, nmap, THC-amap, and others.
- *Identifying all possible user inputs.* Some user input is obvious, such as HTML forms. However, an attacker can interact with a web application in many ways.
- *Finding the user input that is susceptible to the attack.*

1. HTML injection

The essence of this type of injection attack is injecting HTML code through the vulnerable parts of the website. The Malicious user sends HTML code through any vulnerable field with a purpose to change the website's design or any information, that is displayed to the user. This injection attack can be performed with two different purposes: to change displayed website's appearance and to steal other person's identity. Main types are: stored HTML injection and reflected HTML injection.

2. JavaScript Injection

JavaScript is one of the most popular technologies and is most widely used for web pages and web applications. It can be used for realizing different website functionalities. However, this technology can bring some security issues, which the developer and tester should be conscious about.

Javascript can be used not only for good purposes but for some malicious attacks too. One among that is Javascript Injection. Injection is to change the website's appearance and manipulate the parameters. Consequences of JS Injection can be very different – from damaging website‘s design to accessing someone else's account. When you are starting to test against JS Injection, the first thing you should do is to check if JS Injection is possible or not. Checking for this type of Injection possibility is very easy – when navigated to the website, you have to type in the browser's address bar code like this: *javascript:alert('Executed!');*

If a popup window with a message 'Executed!' appears, then the website is vulnerable to JS Injection. Then in the website‘s address bar, you can try various Javascript commands. It should be mentioned, that JS Injection is not only possible from the website's address bar.

There are various other website‘s elements, that may be vulnerable to JS Injection. The most important. Thing is to know exactly the parts of the website which can be affected by Javascript Injection and how to check it.

During this injection attack a malicious user can gain parameters information or change any parameters value (Example, cookie settings).

This can cause quite serious risks as a malicious user can gain sensitive content. Such type of injection can be performed using some Javascript commands. Firstly, in order to prevent this attack, every received input should be validated. Input should be validated every time, and not just when the data is initially accepted. It is highly recommended not to rely on the client side validation. Also, it is recommended to perform an important logic in the server side.

3. SQL Injection

SQL Injection (SQLi) is a type of an injection attack that makes it possible to execute malicious SQL statements. These statements control a database server behind a web application. Attackers can use SQL Injection vulnerabilities to bypass application security measures. They can go around authentication and authorization of a web page or web application and retrieve the content of the entire SQL database. They can also use SQL Injection to add, modify, and delete records in the database.

An SQL Injection vulnerability may affect any website or web application that uses an SQL database such as MySQL, Oracle, SQL Server, or others. Criminals may use it to gain unauthorized access to your sensitive data: customer information, personal data, trade secrets, intellectual property, and more. SQL Injection attacks are one of the oldest, most prevalent, and most dangerous web application vulnerabilities. The OWASP organization (Open Web Application Security Project) lists injections in their OWASP Top 10 2017 document as the number one threat to web application security. To make an SQL Injection attack, an attacker must first find vulnerable user inputs within the web page or web application. A web page or web application that has an SQL Injection vulnerability uses such user input directly in an SQL query. The attacker can create input content. Such content is often called a malicious payload and is the key part of the attack. After the attacker sends this content, malicious SQL commands are executed in the database.

SQL is a query language that was designed to manage data stored in relational databases. You can use it to access, modify, and delete data. Many web applications and websites store all the data in SQL databases. In some cases, you can also use SQL commands to run operating system commands. Therefore, a successful SQL Injection attack can have very serious consequences.

The only sure way to prevent SQL Injection attacks is input validation and parametrized queries including prepared statements. The application code should never use the input directly. The developer must sanitize all input, not only web form inputs such as login forms. They must remove potential malicious code elements such as single quotes. It is also a good idea to turn off the visibility of database errors on your production sites. Database errors can be used with SQL Injection to gain information about your database.

Bibliografie

1. Justin Clarke, SQL Injection Attacks and Defense, 2015.
2. https://www.acunetix.com/websitesecurity/sql-injection2/
3. https://www.softwaretestinghelp.com/html-injection-tutorial/
4. *https://www.softwaretestinghelp.com/javascript-injection-tutorial/*