wib. | Use Case

Injection attacks

North America large University

What is an injection attack?

Injection attacks can occur in web applications as well as APIs. They happen when an attacker sends malicious input (string or data) to a web application or API, with the intention of changing its operation, and the application treats this input as legitimate and executes it as part of a normal process flow.

This can result in unintended consequences, such as executing malicious code or retrieving sensitive data. The most common types of injection attacks include:

- **Structured Query Language (SQL) injection**
- **No SQL queries**
- **Operating system (OS) commands**
- **Extensible Markup Language (XML)**
- **Lightweight Directory Access Protocol (LDAP)**
- **Object-Relational Mapping (ORM)**

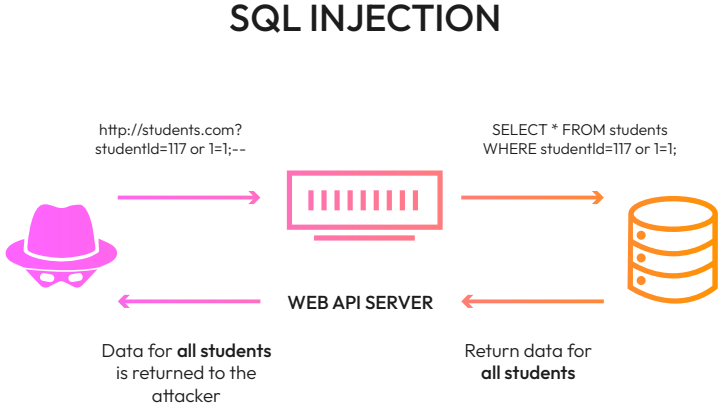
It's important for developers to properly validate and sanitize user input to prevent these types of attacks which as mentioned are very common and why injection attacks 'earned' their place on OWASP's top 10 list of web application security risks.

How it works

The basic idea behind an Injection attack is to inject malicious data into an application, either as part of a request or as input to a form. This malicious data can then be executed by the application which results in unintended consequences such as data disclosure, code execution, and system compromise.

The vulnerability occurs when the application fails to validate user input properly, allowing an attacker to inject malicious data into the request or input. The attacker can then use this to force the back-end system, such as an SQL database to disclose data or execute malicious code, despite them not having credentials to access to it.

The example here illustrates an SQL injection at a university. An attacker sends a request to the database server; altering it so that the server returns not the user data but the entire database.



Case example: top North American university

Wib's attack research team where invited to evaluate API security at a large North American university.

Objective

The objective was to identify any API vulnerabilities within the organization with a view to introducing necessary remedial controls.

To undertake our work, we received a comprehensive export of JSON file documentation from the university including all APIs known to the DevOps group. We were also provided with access credentials for a low-privilege user and an admin user.

What we did

During the test, our Wib code analysis and traffic inspection engines discovered unmanaged 'shadow' APIs communicating with endpoints. Initially, we assumed that the university forgot to include those APIs in its documentation. On closer inspection, these API were not included in the documentation, but we did notice that the shadow APIs had a similar structure to the known APIs. We were able to confirm that the shadow APIs were in fact old versions of the same API that were still active and receiving API calls. The old API versions caused endpoints to handle the calls differently to the current API version.

We were able to insert a JavaScript code to execute a Cross-Site Scripting (XSS) attack, which was successful because of the lack of proper validation on user input. This in return enabled our red team attackers to steal the admin cookie and authenticate themselves (account takeover); allowing them to perform actions without the server being able to identify the legitimate user and the attacker.

What we found

The potential impact on the company was high. We were able to convert a low-privilege user to an admin user in a few clicks, without detection. Admins can perform any action in the system, potentially stealing data, manipulating other accounts, deleting information, etc. As well as the obvious commercial impact, the university's reputation would have been irrevocably tarnished and it would face stiff penalties for breaking compliance regulations such as GDPR and CCPA.

How to avoid API injection attacks

- **Develop a safe API** that provides a parameterized interface.
- **Special characters should be escaped** using the specific syntax for the target interpreter.
- **Always limit the number of returned records** to prevent mass disclosure in case of injection.
- **Define data types and strict patterns** for all string parameters.
- **Validate incoming data** using sufficient filters to only allow valid values for each input parameter.
- **Perform data validation** using a single, trustworthy, and actively maintained library.
- **Validate, filter and sanitize all client-provided data,** or other data coming from integrated systems.

The Wib holistic approach

Wib advanced API Security Platform ('Fusion Platform') is comprehensive, holistic solution for securing APIs across an organization's entire ecosystem. It utilizes a comprehensive, multi-lens approach, powered by Wib's proprietary Fusion Engine – to assess the security posture of APIs, minimize risk, and quickly address cybersecurity incidents throughout the entire development process. From code to testing and production – further enabling incident response and vulnerability management via Wib Fusion Defense.

It focuses on three engines – code analysis, traffic inspection and attack simulation – provide advanced perspectives to better gain a holistic view of each organization's unique API gaps and flaws.

The code analysis engine maps all endpoints, APIs, patterns, and designs in the code by connecting to the API repositories and inventories.
At this point, we can identify the APIs we expect to see. The engine will identify any endpoints unable to properly validate the authentication tokens.

The traffic inspection engine harnesses the code analysis data to verify and compare. This avoids false positives and can identify where mitigation controls may be in place if there is no authentication enforcement. It also enables other potentially hidden APIs to be inspected for similar defects

The attack simulation engine allows us to 'red team' various attacks against the endpoints and APIs. A human dimension is critical here as there is no automated tool capable of exploiting business logic. Some attacks are used to eliminate false positive feedback.

Through this approach, Wib's Fusion Platform is uniquely equipped to optimize defence against API logic attacks and effective detection of potential blind spots where traditional rule base detection will fail.

Secure. Liberate. Innovate.

API Penetration Testing as a Service (PTaaS)

APIs expose more attack surface than User Interfaces, but most penetration testing lacks rigorous testing of APIs by experienced offensive API attackers.

Our industry-first API penetration testing service is a quick and simple way to ensure visibility, protection, and compliance by providing full pen testing capabilities or augmenting your existing pen testing solution with our API-specific security expertise.

What you get:

- **Full risk and vulnerability assessment of your critical APIs (can include black, grey, or white box testing)**
- **A risk severity score based on NIST cyber matrix calculator**
- **Contextual remediation report for all identified vulnerabilities**
- **Consultancy and remediation roadmap plan with Wib's security team experts**

Testing tailored to PCI DSS 4.0, GDPR, CCPA, SOC-2, ISO, NIST 800-30, HIPAA, CMA and other regulatory Frameworks