



Penetration Test Report

Prepared By [REDACTED] Chouifi

Inhaltsverzeichnis

1 Actions Taken.....3

2 Explanation of attack:.....5

3 Rating By [REDACTED] Chouifi.....6

1 Actions Taken

To determine and practically demonstrate the feasibility of expanding access given a foothold from external network (internet), [REDACTED] conducted the following activities:

From Zone: External network (internet)

To Zone: [REDACTED]

Method: SQL-Injection

SQL-Injection into Website (SQL type :MySQL)

SQL injection is a common type of cyber attack that targets the security vulnerabilities in web applications utilizing SQL (Structured Query Language) databases. In SQL injection attacks, malicious actors exploit input fields within web forms or URL parameters to inject malicious SQL code into the application's backend database.

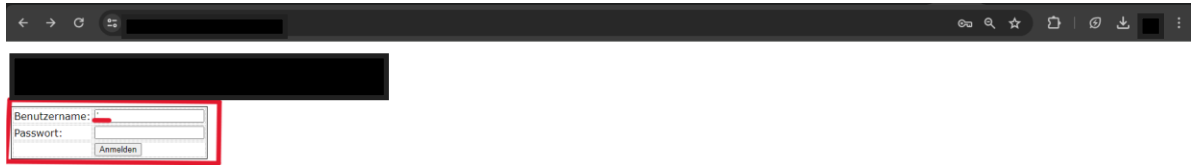
This injected SQL code can then be executed by the application's database server, allowing the attacker to perform a variety of unauthorized actions, such as retrieving sensitive data, modifying or deleting database records, or even gaining administrative access to the entire database.

SQL injection attacks can have severe consequences, including unauthorized access to confidential information, data breaches, and manipulation or destruction of data. They pose a significant threat to the security and integrity of web applications and the sensitive data they handle.

Preventing SQL injection requires careful coding practices, such as using parameterized queries or prepared statements, input validation, and sanitization of user input to ensure that all data passed to the database is properly formatted and cannot be interpreted as SQL commands. Additionally, regular security audits and updates to address any potential vulnerabilities are crucial for protecting against SQL injection attacks.



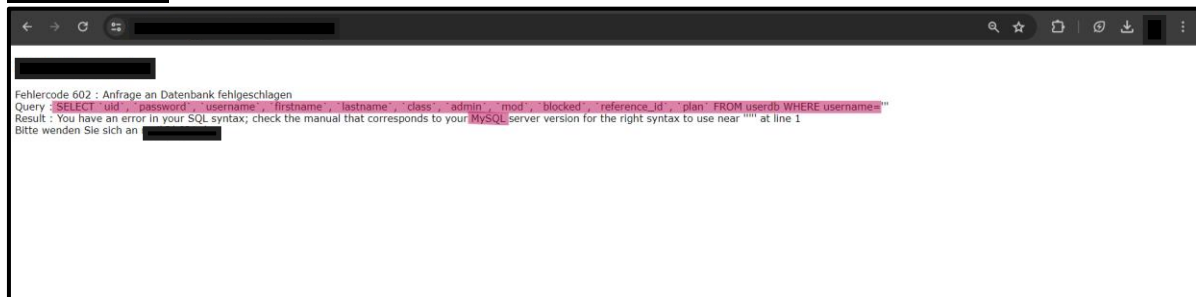
INPUT:



Benutzername:
Passwort:

“ ”

OUTPUT:



Fehlercode 602 : Anfrage an Datenbank fehlgeschlagen
Query : SELECT `uid`, `password`, `username`, `firstname`, `lastname`, `class`, `admin`, `mod`, `blocked`, `reference_id`, `plan` FROM userdb WHERE username=""
Result : You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near "" at line 1
Bitte wenden Sie sich an [REDACTED]

“ Fehlercode 602 : Anfrage an Datenbank fehlgeschlagen

Query : SELECT `uid`, `password`, `username`, `firstname`, `lastname`, `class`, `admin`,
`mod`, `blocked`, `reference_id`, `plan` FROM userdb WHERE username=""

Result : You have an error in your SQL syntax; check the manual that corresponds to your
MySQL server version for the right syntax to use near "" at line 1

Bitte wenden Sie sich an [REDACTED] ”



2 Explanation of attack:

By inserting a single quote (') into the input field, I triggered an error message. This seemingly innocuous action is, in fact, a critical vulnerability indicative of a potential SQL injection attack. The error message returned by the application is alarmingly revealing, as it contains vital information about the structure of the database, including the names of its columns.

This seemingly benign error message serves as a roadmap for a malicious actor, offering insight into the underlying database schema. Armed with this knowledge, an attacker could craft SQL queries to extract sensitive information from the database with ease. Each column name exposed in the error message serves as a potential gateway to accessing a wealth of confidential data stored within the database.

What makes this error message particularly dangerous is its unintended disclosure of the database schema. With this information, an attacker could construct targeted SQL queries to retrieve, modify, or even delete data from the database. In the wrong hands, this exploit could lead to severe consequences, including data breaches, unauthorized access to sensitive information, and potential financial or reputational damage to the organization.

Therefore, it is imperative to address this SQL injection vulnerability promptly. Implementing robust input validation and sanitization measures, such as parameterized queries or prepared statements, can mitigate the risk of SQL injection attacks. Additionally, conducting regular security audits and updates to patch any vulnerabilities in the application's codebase is essential for safeguarding against such exploits.



3 Rating By [REDACTED] Chouifi

Technical Rating:

Max CVSS (Common Vulnerability Scoring System): The SQL injection vulnerability poses a significant threat to the security of the system, potentially allowing attackers to gain unauthorized access to sensitive data or execute arbitrary SQL commands. Considering the potential impact on confidentiality, integrity, and availability of the system, the vulnerability could be assigned a high CVSS score, possibly in the range of **6.8**.

EPSS (Exploitability Score): The exploitability of the SQL injection vulnerability depends on various factors, including the complexity of the SQL queries required to exploit the vulnerability and the level of access controls implemented within the system. Given the simplicity of launching SQL injection attacks and the widespread availability of automated tools for reconnaissance and exploitation, the vulnerability could receive a moderately high EPSS score (**2.83%**).

Non-Technical Rating:

Considering the information provided (name and birth date) may not be highly valuable on its own, the risk posed by the SQL injection attack could be assessed as moderately dangerous on a scale of 1 to 10. While the immediate impact of accessing this limited data may seem relatively low, the potential for escalation exists.

This escalation could involve compromising the entire system, allowing attackers to exfiltrate more sensitive data, disrupt services, or even deploy malicious payloads such as malware or ransomware. Therefore, even though the initial data may not be highly valuable, the possibility of further exploitation raises the overall risk level.

Given these considerations, the non-technical rating for the danger posed by the SQL injection attack is around **8**. While not the most severe threat, it warrants immediate attention and remediation to prevent potential escalation and mitigate the risk of more significant harm to the system and its users.

