|  |  |
| --- | --- |
| **Observation No. 1** | |
| **Vulnerability** | SQL injection |
| **Status** | Close |
| **Severity** | **High** |
| **Description** | SQL injection vulnerabilities arise when user-controllable data is incorporated into database SQL queries in an unsafe manner. An attacker can supply crafted input to break out of the data context in which their input appears and interfere with the structure of the surrounding query. A wide range of damaging attacks can often be delivered via SQL injection, including reading or modifying critical application data, interfering with application logic, escalating privileges within the database and taking control of the database server |
| **Remediation** | The most effective way to prevent SQL injection attacks is to use parameterized queries (also known as prepared statements) for all database access. This method uses two steps to incorporate potentially tainted data into SQL queries: first, the application specifies the structure of the query, leaving placeholders for each item of user input; second, the application specifies the contents of each placeholder |
| **Affected URLs** | * http://localhost:3000/observation/create/64f5cfc774d05ba3ae39cef7 * http://localhost:3000/observation/create/64f5cfc774d05ba3ae39cef7 |
| **References** | Web Security Academy: SQL injection Using Burp to Test for Injection Flaws Web Security Academy: SQL Injection Cheat Sheet |
| **Proof Of Concept** | Screenshot shared below. |
| http://localhost:3000/observation/create/64f5cfc774d05ba3ae39cef7 | |
| http://localhost:3000/observation/create/64f5cfc774d05ba3ae39cef7 | |
| http://localhost:3000/observation/create/64f5cfc774d05ba3ae39cef7 | |
| http://localhost:3000/observation/create/64f5cfc774d05ba3ae39cef7 | |