SQL Injection Vulnerability

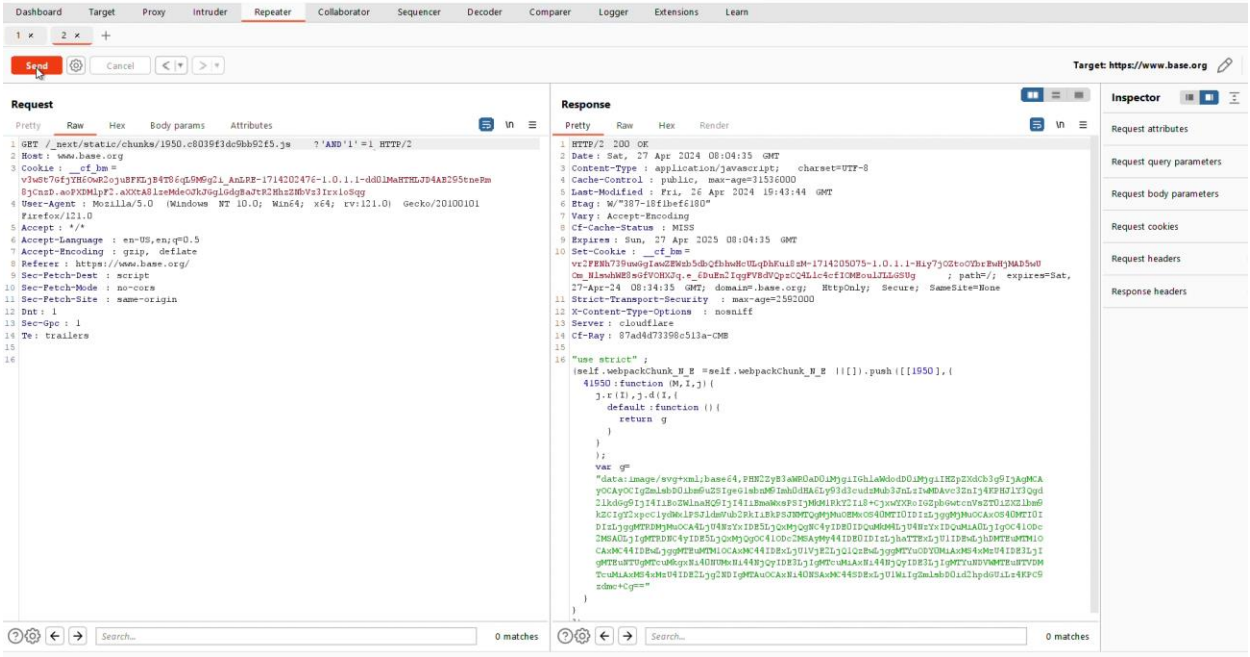
Vulnerability Description:

The application's URL parameter handling is susceptible to Blind SQL injection attacks. An attacker can exploit this vulnerability by injecting crafted SQL code into the URL parameters. The payload ' and 1='1 I was tested. The respond is 200.

Affected Components:

URL parameter handling mechanism

Proof of Concept:



Proposed Mitigation or Fix:

1. Input Validation: Implement strict input validation mechanisms to sanitize user-supplied data before incorporating it into SQL queries.

2. Parameterized Queries: Utilize parameterized queries (prepared statements) to separate SQL query structure from user inputs, preventing injection attacks.

3. Least Privilege Principle: Limit database user permissions to mitigate the impact of successful injection attacks.

4. Regular Security Audits: Conduct regular security audits and penetration testing to identify and remediate vulnerabilities proactively.

Impact:

1. Unauthorized Data Access: Attackers can retrieve sensitive information stored in the application's database.

2. Data Manipulation: Malicious actors may modify or delete critical data, disrupting application functionality.

3. Privilege Escalation: Successful exploitation can lead to unauthorized access and privilege escalation within the database.

4. Application Compromise: SQL injection can compromise the integrity and confidentiality of the entire application.

Conclusion:

Addressing SQL injection vulnerabilities requires a comprehensive approach involving input validation, secure coding practices, and regular security assessments. By implementing robust security measures, organizations can safeguard their applications against SQL injection attacks and protect sensitive data from unauthorized access and manipulation.