Name : Patel Dhruv Nirmalkumar
Email : dhruvpatel.00700@gmail.com
Phone : 9426176903
Task : 13

API Security Testing Report

---

1. Understanding REST API and HTTP Methods

- The tested application is a **RESTful API**, where:
  - **GET** is used to retrieve data from the server
  - **POST** is used to create or update data
  - **PUT** is used to modify existing resources
  - **DELETE** is used to remove resources
- Each API endpoint represents a resource identified using a **URI** (e.g., /api/data/{id}).
- Communication between client and server occurs over **HTTP**, using request headers, request body, and response status codes.

2. API Configuration in Postman

- The API was configured in **Postman** using the provided documentation.
- Configuration included:
  - **Endpoint URL** (e.g., /api/data/{id}, /api/update, /api/ping)
  - **HTTP method** selection (GET, POST)
  - **Headers**, such as:
    - Authorization: Bearer <token>
    - Content-Type: application/json
  - **Request body** in JSON format where required (for POST requests)

3. Testing API Authentication

- Authentication testing was performed by:
    - Sending requests with **valid authentication tokens**
    - Sending requests with **invalid or expired tokens**
- Observations:
    - Some endpoints correctly validated authentication
    - Other endpoints returned data even when authentication was missing or invalid
- This behavior indicates **inconsistent authentication enforcement**.

4. Testing Unauthenticated Access

- Authentication headers were **removed** and requests were resent.
- Finding:
    - The endpoint /api/data/3 returned sensitive data **without any authentication**.
- Impact:
    - Any external user can access protected data.
- OWASP Mapping:
    - **API2:2023 – Broken Authentication**

5. Testing Broken Object Level Authorization (BOLA)

- The resource identifier (id) in the request URL was manually modified.
- Example:
    - Authenticated user user1 accessed /api/data/1 instead of /api/data/2
- Result:
    - Sensitive admin data was successfully retrieved.
- Impact:
    - Users can access data that does not belong to them.
- OWASP Mapping:
    - **API1:2023 – Broken Object Level Authorization**

6. Testing Input Validation

- Malformed and incomplete inputs were sent to the API.
- Example:
  - POST request to /api/update without the required id field.
- Result:
  - Server responded with:
    - 500 Internal Server Error
    - Message revealing internal logic: *Missing ID*
- Risk:
  - Verbose error messages can help attackers understand backend behavior.
- OWASP Mapping:
  - **API7:2023 – Security Misconfiguration**

7. Testing Rate Limiting

- Multiple rapid requests were sent to the /api/ping endpoint.
- Observation:
  - The API processed all requests without delay or blocking.
- Impact:
  - Vulnerable to brute-force attacks and Denial of Service (DoS).
- OWASP Mapping:
  - **API4:2023 – Lack of Resources & Rate Limiting**

8. Reviewing HTTP Response Codes and Error Messages

- HTTP response codes observed:
  - 200 OK for unauthorized and authenticated access
  - 500 Internal Server Error for client-side input issues
- Issues identified:
  - Incorrect use of status codes
  - Exposure of internal error messages
- Security Risk:
  - Helps attackers perform reconnaissance and exploit weaknesses.

Summary of Identified Vulnerabilities

Vulnerability ID	Issue Identified	OWASP API Risk	Severity
VULN-001	Broken Object Level Authorization	API1:2023	High
VULN-002	Broken Authentication	API2:2023	High
VULN-003	Improper Inventory Management	API9:2023	Medium
VULN-004	No Rate Limiting	API4:2023	Medium
VULN-005	Improper Error Handling	API7:2023	Low

Conclusion

- The API contains **critical security flaws**, mainly in:
    - Authorization
    - Authentication
    - Rate limiting
- The most severe risks are **BOLA** and **Broken Authentication**, which directly expose sensitive data.
- Immediate remediation is required, along with:
    - Centralized authentication enforcement
    - Object-level authorization checks
    - Proper error handling
    - Rate limiting controls
- Regular security testing aligned with **OWASP API Security Top 10** is strongly recommended.