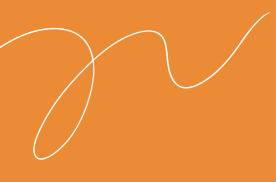




Burcu Yarar @brcyrr
Roadmap Suggestion of the Week

API Pentesting Roadmap





This roadmap includes sample answers to the following questions.

- 1 Which source should you read?
- Which cheat sheet should **you use?**
- Which vulnerable lab environment should you practice?



Which source should **you read?**Pentest Book

Resource Link







Which cheat sheet should you use?

Cloud Pentest Cheat Sheets

Resource Link

API Security Checklist necklist of the most important security countermeasures when designing, testing, and releasing your API. uthentication Don't reinvent the wheel in lauthentication, token generation, password storage. Use the standards. Use leax Retry; and jail features in Login. Use leax Retry; and jail features in Login. Use encryption on all sensitive data. WT (JSON Web Token) Use a random complicated key (JwT Secret) to make brute forcing the token very hard. Don't extract the algorithm from the header. Force the algorithm in the backend (18236 or 18256). Make token expiration (TTL, RTL) as short as possible. Don't store sensitive data in the JWT payload, it can be decoded easily. Avoid storing too much data. JWT is usually shared in headers and they have a size limit. CCESS Limit requests (Throttling) to avoid DDoS / brute-force attacks. Use HTTPS on server side with TLS 1.2+ and secure ciphers to avoid MITM (Man in the Middle Attack). Use ISTS header with SSL to avoid SSL Strip attacks. Turn off directory listings. For private APIs, allow access only from safelisted IPs/hosts. uthorization Auth Always validate redirect_ur1 server-side to allow only safelisted URLs. Always try to exchange for code and not tokens (don't allow response_type=token). Use ISTSTE parameter with a random hash to prevent CSRF on the OAuth authorization process. Define the default scope, and validate scope parameters for each application.





Which vulnerable lab environment should **you practice?**

Awesome Cloud Security / Vulnerable By Design

Resource Link

ulnerableApp4APISecurity

is repository was developed using .NET 7.0 API technology based on findings listed in the OWASP 2019 API bourly 1 fop 10. This project has been developed to be used as an example project while explaining API Security do easily generate attack scenarios while companies are trying the products they test to ensure API Security, dition, Mongo was used as the database in the project, and it was developed considering Clean Architecture a slid Principles as much as possible.

ne project will be examined under the following headings

- 1 Vulnerabilities
- 2 Run
- 2.2 Running from Container
- 2.3 Running from Docker-compose
- 2.4 Running from Rub
 3 Dummy Traffics
- 3 Dummy Traffics

 3.1 Available Side
- o 3.2 Updatable Side
- 4 API Details Documentation
 41 Postman Collection
- 4.2 Excel of API Details

- Vulnerabilities 🗈

e vulnerabilities on the developed application are as follows. While some of these vulnerabilities can be found ectly, some have been developed to find and exploit indirectly (Path/Arguments fuzzing, Brute-force/rate limit) simulate real-life scenarios.

- API 01:2019 Broken object level authorization
- API 02:2019 Broken authentication
- API 03:2019 Excessive data exposure
- API 04-2019 Lack of resources and rate limiting
 API 05:2019 Broken function level authorization
- API 06:2019 Mass assignment
- API 07:2019 Security
- API 08:2019 Injection
 - 21 10:2019 Insufficient logging and monitoring



