TPL - Pentesting report

Company header (Pentesting company)

Version header

Version	Author	Reviewer	Remarks
0.1 - DRAFT	Wesley	Wesley	• Remove table 3 • Insert OWASP refercen
0.2 - DRAFT	Wesley		

Goal of this document

This document serves to inform the client of any potential vulnerability that was found. Compliance with the OWASP top 10 will also be included plus all supporting test material. Every defect will be logged and an indication of impact will be given. We include remediation steps but these always need to be verified internally and can not always be directly implemented as is.

Who is this for

This document serves to inform security operations, management, and developers of the defects that exist so they can make an informed decision about securing their product. Take care not to share this document with unauthorized personnel before all defects have been resolved or accepted.

Executive summary

A total of x hours have been spent testing to uncover 4 critical issues, 10 high impact issues, 1 medium, and 30 low/informational. This indicates the health of the system is not as good as it should be and several of these issues can be very damaging. When it comes to the OWASP top 10, we have failed 4 out of the 10 categories.

Our recommendation is to at least get all the critical, high, and medium defects given the potential business impact.

Methodology used

We followed these 5 basic steps: Initial recon, enumeration, analysis, PoC creation, and reporting. We did this in accordance with the OWASP WSTG 4.1 and have been checking the OWASP top 10 on all endpoints.

We used burp suite pro’s automated scanner and OWASP ZAP’s automated vulnerability scanners in combination with manual testing of the most critical endpoints.

Issues found

Per issue, we note the following:

Severity – Title

Description

[[Talk in short about what happened in full sentences]]

Pre-requisites

[[Note down any requirements in a list]]

Steps to reproduce

[[Note down the steps, everything you have to do in great detail]]

Expected result

[[What should have happened]]

Actual result

[[What really happened]]

Impact

[[What is the impact in terms of this company]]

Remediation steps

[[What do you recommend to fix this]]

OWASP top 10 item

[[If applicable]]

OWASP compliance

OWASP item	Pass/Fail	Comment
API1:2019 Broken Object Level Authorization		
API2:2019 Broken User Authentication		
API3:2019 Excessive Data Exposure		
API4:2019 Lack of Resources & Rate Limiting		
API5:2019 Broken Function Level Authorization		
API6:2019 Mass Assignment		
API7:2019 Security Misconfiguration		
API8:2019 Injection		
API9:2019 Improper Assets Management		
API10:2019 Insufficient Logging & Monitoring		

Summary table

Issue title	description	severity
Missing httponly flag on cookie	The session cookie requires an httponly flag	Low
XSS in name field	On the invoice, we can trigger Stored XSS via the name field	High
SQLi on login	In the login form, the password field is vulnerable to time based SQLi	Critical

Metrics

Issues found

Severity	Count
Critical	4
High	10
Medium	1
Low/informational	30

Recommendations

- We recommend you add the httponly flag to the session cookie
- We recommend you use a stronger password policy
- We recommend implementing logging and monitoring on the most important flows so we can get alerted upon attack
- We recommend building 1 input sanitizer module and calling it everywhere we use user input. This way we can assure all sanitization happens uniform across the application.
- …

Conclusion

As of now, several critical security defects still exist that should be fixed before moving onto the production phans. We recommend applying proper root cause analysis on these and making sure the cause is fixed and not simply patched. It appears the JS files were all in plain text online, we also recommend applying obfuscation and chunking. A full architecture review would be recommended as well to see if other parts of the organization might be at risk.

The web application is currently not OWASP top 10 complient.

Appendix a.

```
from pydub import AudioSegment

audio_file = AudioSegment.from_mp3("example.mp3")
louder_audio_file = audio_file + 18
louder_audio_file.export("example_louder.mp3", format="mp3")
```

Appendix b.

...