PENETRATION TESTING

easydoc.co.il

Site	easydoc.co.il	**Date:**	15.02.2021
Security Assurance:	Nitzan Levi		
Total Vulnerabilities:	14		

TABLE OF CONTENTS

Contents

1 EXECUTIVE SUMMARY

1.1 Purpose

During the penetration test to easydoc.co.il application, common cyber attacks scenarios are reproduced to evaluate the efficiency of the application ability to face cyberattacks. Discovering vulnerabilities will allow to establish a remediation plan which once correctly implemented, will lower the chance toe xploit vulnerabilities in the application. These Each discovered vulnerability is rated according to the cyber industry best practices risk rating methodology. In addition, We take into consideration the CVE system, MITRE and uses the best-of-breed tooling to determine the severity. The table on the right displays a risking methodology that takes into consideration both the technical impact overall Risk severity and likelihood of the vulnerability to be exploited. However, it is important to note, that this table lacks the business impact perspective, which is specific and customized for each organization. As we cannot take into consideration the specific business model of each customer, it is recommended to consider the business impact to get a better threat level assessment likelihood for each specific vulnerability in the tested target.

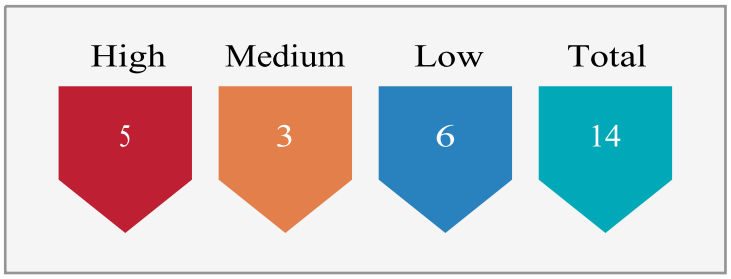
1.2 Tests Results

Test Category	Status
Application Based DOS	Pass
API Data Exposure	Pass
Authentication Bypass	Pass
Blind Time-Based OS Command Injection	Pass
Broken JWT Authentication	Pass
Broken SAML Authentication	Pass
Brute Force Login	Pass
DOM Based Cross-site scripting (DomXSS)	Pass
HTTP Request Smuggling	Pass
LDAP Injection	Pass
Local File Inclusion	Pass
NoSQL Injection	Pass
NoSQL Injection: Blind Time Based	Pass
Open Bucket	Pass
Open DataBase	Pass
OS Command Injection	Pass
Persistent Cross-site scripting (pXSS)	Pass

Prototype Pollution	Pass
Reflective Cross-site scripting (rXSS)	Pass
Remote File Inclusion	Pass
Server-Side Request Forgery	Pass
SQL Injection	Pass
SQL Injection: Blind Boolean Based	Pass
SQL Injection: Blind Time Based	Pass
SSTI - Server-Side Template Injection	Pass
Unrestricted File Upload	Pass
XML External Entity (XXE)	Pass
XPATH Injection	Pass
Authorized Cross-Site Request Forgery (CSRF)	Pass
Backup Location	Pass
Broken Data Structure	Pass
Business Constraint Bypass	Pass
Database connection crashed	Pass
Directory Listing	Pass
Full Path Disclosure	Pass
HTML Injection	Pass
ID Enumeration	Pass
ID Manipulation	Pass
Improper Assets Management	Pass
JavaScript Component with Known Vulnerabilities	Pass
Non-HTML Based XSS	Pass
Secret Tokens Leak	Pass
Session Fixation: Cookie	Pass
Unvalidated Date Range	Pass
Unvalidated Redirect	Pass
Version Control Systems data leak	Pass
WordPress Component with Known Vulnerabilities	Pass
Connection String	Pass
Cookie Reuse	Pass
Default Login Location	Pass
Exposed Common File	Pass
Insecure HTTP Method	Pass
Insecure HTTP Method	Pass
Missing X-Frame-Options Header	Pass
NoSQL Error Message in Response	Pass
Sensitive Cookie in HTTPS Session Without Secure Attribute	Pass

Sensitive Cookie Without HttpOnly Flag	Pass
Source Code Disclosure	Pass
SQL DB Error Message in Response	Pass
Unauthorized Cross-Site Request Forgery (CSRF)	Pass
Unrestricted File Upload	Pass
X-XSS-Protection Header is Missing or Misconfigured	Pass

1.3 Vulnerability Summary



#	Vulnerability	Resolved	Severity
1	JavaScript Component with Known Vulnerabilities		H
2	Brute Force Login		H
3	ID Manipulation		H
4	API Data Exposure		H
5	Unrestricted File Upload		H
6	CVE-2020-11022		M