HACKTHEBOX

TRILOCOR
CUTTING EDGE ROBOTICS

Private Web Application
Bug Bounty Program Assessment
Report of Findings

Trilocor Robotics

Month Day, Year

Version 1.0

Hack The Box Confidential
No part of this document may be disclosed to outside sources without the explicit written authorization of Hack The Box.

HACKTHEBOX

Tables of Contents

Statement of Confidentiality

The contents of this document have been developed by Hack The Box. Hack The Box considers the contents of this document to be proprietary and business confidential information. This information is to be used only in the performance of its intended use. This document may not be released to another vendor, business partner or contractor without prior written consent from Hack The Box. Additionally, no portion of this document may be communicated, reproduced, copied or distributed without the prior consent of Hack The Box.

The contents of this document do not constitute legal advice. Hack The Box's offer of services that relate to compliance, litigation or other legal interests are not intended as legal counsel and should not be taken as such. The assessment detailed herein is against a fictional company for training and examination purposes, and the vulnerabilities in no way affect Hack The Box external or internal infrastructure.

Engagement Contacts

Trilocor Contacts

Primary Contact	Title	Primary Contact Email
Yelon Husk	Chief Executive Officer	yelon@trilocor.local
Secondary Contact	**Title**	**Secondary Contact Email**
Zeyad AlMadani	Chief Technical Officer	zeyad@trilocor.local

Assessor Contact

Assessor Name	Title	Assessor Contact Email
<Candidate Name>	Security Consultant	<Candidate Email>

Executive Summary

Trilocor Robotics Ltd. (“Trilocor” herein) invited <ASSESSOR NAME> to a private bug bounty program to perform a targeted Web Application Penetration Test of Trilocor’s externally facing web applications to identify high-risk security weaknesses, determine the impact to Trilocor, document all findings in a clear and repeatable manner, and provide remediation recommendations. The following types of findings were in-scope for this private bug bounty program:

- Sensitive or personally identifiable information disclosure
- Cross-Site Scripting (XSS)
- Server-side or remote code execution (RCE)
- Arbitrary file upload
- Authentication or authorization flaws, such as insecure direct object references (IDOR), and authentication bypasses
- All forms of injection vulnerabilities
- Directory traversal
- Local file read
- Significant security misconfigurations and business logic flaws
- Exposed credentials that could be leveraged to gain further access

The following types of activities were considered out-of-scope for this bug bounty program:

- Scanning and assessing any other IP in the Entry Point's network
- Physical attacks against Trilocor properties
- Unverified scanner output
- Man-in-the-Middle attacks
- Any vulnerabilities identified through DDoS or spam attacks
- Self-XSS
- Login/logout CSRF
- Issues with SSL certificates, open ports, TLS versions, or missing HTTP response headers
- Vulnerabilities in third party libraries unless they can be leveraged to significantly impact the target
- Any theoretical attacks or attacks that require significant user interaction or low risk

<ASSESSOR NAME> performed testing under a “black box” approach from <START DATE> to <END DATE> without credentials or any advance knowledge of Trilocor’s web applications with the goal of identifying unknown weaknesses. Testing was performed from a non-evasive standpoint with the goal of uncovering as many misconfigurations and vulnerabilities as possible. Testing was performed remotely. Each weakness identified was documented and manually investigated to determine exploitation possibilities and escalation potential. <ASSESSOR NAME> sought to demonstrate the full impact of every vulnerability, up to and including internal network access.

Scope

The scope of this assessment was as follows **.trilocor.local* and any and all open web server ports discovered on the target IP address provided at the start of the assessment

In-Scope Assets

Host/URL/IP Address	Description
www.trilocor.local	Main Trilocor website/unauthenticated
<exam IP address>	PR website/unauthenticated
<exam IP address>	Jobs Portal/unauthenticated
<exam IP address>	HR website/unauthenticated
<exam IP address>	Trilocor online store/unauthenticated

Assessment Overview and Recommendations

<Summary of findings and recommendations>

During the course of testing <ASSESSOR NAME> identified…

HACKTHEBOX

Web Application Security Assessment Summary

<ASSESSOR NAME> began all testing activities from the perspective of an unauthenticated user on the internet. Trilocor provided the tester with a single URL and IP address but did not provide additional information such as operating system or configuration information.

Summary of Findings

During the course of testing <ASSESSOR NAME> uncovered a total of <number> of findings that pose a material risk to Trilocor's information systems. The below table provides a summary of the findings by severity level.

Finding Severity			
High	**Medium**	**Low**	**Total**
0	0	0	0

Below is a high-level overview of each finding identified during the course of testing. These findings are covered in depth in the Technical Findings Details section of this report.

Finding #	Severity Level	Finding Name
1.	*High*	*Command Injection*
2.	*Medium*	*Username Enumeration*
3.	*Low*	*Cookie Missing Secure Flag*

Technical Findings Details

1. SQL Injection - High

CWE	
CVSS 3.1 Score	
Description (Incl. Root Cause)	The application does not properly sanitize input data, allowing an unauthenticated attacker to inject SQL code into database queries. **EXAMPLE FINDING**
Security Impact	A successful SQL injection attack can result in access to sensitive data from the database, modifications to database data (Insert/Update/Delete), execution of administration operations on the database (such as shutting down the DBMS), recovering the contents of a given file present on the DBMS file system and in some cases issuing commands on the underlying operating system.
Affected Host(s)	• mytestsite.com ○ Id parameter
Remediation	Where possible, use parameterized queries to ensure that database interactions cannot be contaminated. Also, escape all user supplied input/utilize a whitelist of approved characters to validate all input that is passed to the database.
External References	https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet

Finding Evidence:

Note to candidate: Finding evidence should include detailed reproduction steps, showing how you discovered the vulnerability, exploitation steps, and a screenshot showing the flag obtained using the vulnerability (if it resulted in discovery of a flag). It should be possible to easily recreate each finding from the evidence & steps you provide. If you are having trouble with reporting or would like to see a sample of the type of report we expect for a passing grade, check out the Documentation and Reporting module on HTB Academy.

```
$ sqlmap -u 'http://mytestsite.com/page.php?id=5'

GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N] n
sqlmap identified the following injection point(s) with a total of 53 HTTP(s) requests:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=1 AND 9561=9561

    Type: AND/OR time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind
    Payload: id=1 AND SLEEP(5)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: id=-6630 UNION ALL SELECT
NULL,CONCAT(0x7178786271,0x79434e597a45536f5a4c695273427857546c76554854574c4f5a534f587368725142615a54456256,0
x716b767a71),NULL-- mIJj
---
[12:56:52] [INFO] the back-end DBMS is MySQL
web application technology: Nginx, PHP 5.3.10
back-end DBMS: MySQL >= 5.0.12
[12:56:52] [INFO] fetched data logged to text files under '/home/elliot/.sqlmap/output/mytestsite'

[*] shutting down at 12:56:52
```

<Insert screenshots as appropriate>

2. Username Enumeration - Medium

CWE	<Fill in>
CVSS 3.1 Score	<Fill in>
Description (Incl. Root Cause)	<Fill in>
Security Impact	<Fill in>
Affected Host(s)	• <Fill in>
Remediation	<Fill in>
External References	<Fill in>

Finding Evidence:

<Add command output as appropriate>

<Insert screenshots as appropriate>

3. Cookie Missing Secure Flag - Low

CWE	<Fill in>
CVSS 3.1 Score	<Fill in>
Description (Incl. Root Cause)	<Fill in>
Security Impact	<Fill in>
Affected Host(s)	• <Fill in>
Remediation	<Fill in>
External References	<Fill in>

Finding Evidence:

<Add command output as appropriate>

<Insert screenshots as appropriate>

Appendices

Appendix A – Flags Discovered

Flag #	Application	Flag Value	Flag Location	Method Used
1.	*Main*	*HTB{<random value>}*	*Web root*	*Command Injection (example)*
2.				
3.				
4.				
5.				
6.				
7.				
8.				
9.				
10.				