HACKERONE

2022 Penetration Test Final Summary Report

February 28, 2022 - March 14, 2022

Lead Pentester

Joao Lucas Melo Brasio - @whhackersbr

Supporting Pentester

dkd - @dkd

Juho Myllys - @muon4

Managed By

Sasha Zivojinovic, Technical Engagement Manager

sasha@hackerone.com

Disclaimer

The scope and objectives of this review are summarized in the Findings Summary. The matters raised in this report are only those identified during the review and are not necessarily a comprehensive statement of all weaknesses that exist or all actions that might be taken. This work was performed under limitations of time and scope that may not be a limitation faced by a persistent actor. The review is based at a specific point in time, in an environment where both the systems and the threat profiles are dynamically evolving. It is therefore possible that vulnerabilities exist or will arise that were not identified during the review and there may or will have been events, developments, and changes in circumstances subsequent to its issue.

Table of Contents

1. Executive Summary

Hackerone engaged a team of three testers to perform a penetration test from February 28, 2022 to March 14, 2022. This report is a reflection of the state of security across systems tested during this period.

During this timeframe, one vulnerability was identified. Zero vulnerabilities were found that had a Common Vulnerability Scoring System (CVSS) rating of 7.0 or higher.

A gray box penetration test of HackerOne's Web Application, API and external infrastructure was conducted to assess its risk posture and identify security issues that could negatively affect the data, systems, or reputation of HackerOne.

The testers approached the engagement with a hybrid methodology, consisting of a largely manual set of tests with the help of some automated tooling to provide coverage. The main goal was a holistic assessment of the security of the application's functionality, business logic, and vulnerabilities such as those cataloged in the 2021 OWASP Top 10. The assessment also included a review of security controls and requirements listed in the OWASP Application Security Verification Standard (ASVS).

The in-scope assets "hackerone.com" and "api.hackerone.com" were found to be very resilient against the vulnerability types outlined in the methodology. No vulnerabilities were found in the Cross-Site Scripting, Cross-Site Request Forgery, Remote Code Execution, injection categories. The external infrastructure elements were well configured and did not present any services with vulnerabilities or misconfigurations that could pose a security risk.

Section 4. Methodology contains additional information related to the testing methodology used in this engagement.

1.1 Recommendations

Based on the results of this assessment, there are no recommendations.

2. Scope Summary

2.1 In-Scope Assets

The following assets were considered explicitly in-scope for testing:

Assets In Scope	Hostname/CIDR
HackerOne Web Application	hackerone.com
HackerOne API	api.hackerone.com
HackerOne External Infrastructure	66.232.20.0/23
HackerOne Application Infrastructure	34.213.196.80/28

2.2 Vulnerable Assets

There were no notable vulnerabilities found in the following assets:

- hackerone.com
- api.hackerone.com
- 66.232.20.0/23
- 34.213.196.80/28

3. Findings Summary

Findings are sorted by their severity and grouped by the asset and CWE classification. Each asset section will contain a summary. Table 1 in the executive summary contains the total number of identified security vulnerabilities per asset per risk indication. All findings were entered in the HackerOne platform, which is the authoritative source for the information on the vulnerabilities and can be referred to for details about each finding.

3.1 Vulnerability Classification & Severity

To categorize vulnerabilities according to a commonly understood vulnerability taxonomy, HackerOne uses the industry-standard Common Weakness Enumeration (CWE). CWE is a community-developed taxonomy of common software security weaknesses. It serves as a common language, a measuring stick for software security tools, and as a baseline for weakness identification, mitigation, and prevention efforts.

To rate the severity of vulnerabilities, HackerOne uses the industry standard Common Vulnerability Scoring System (CVSS) to calculate severity for each identified security vulnerability. CVSS provides a way to capture the principal characteristics of a vulnerability, and produce a numerical score reflecting its severity, as well as a textual representation of that score.

To help prioritize vulnerabilities and assist vulnerability management processes, HackerOne translates the numerical CVSS rating to a qualitative representation (such as low, medium, high and critical):

🟥	**Critical**	CVSS rating 9.0 - 10.0
🟪	**High**	CVSS rating 7.0 - 8.9
🟧	**Medium**	CVSS rating 4.0 - 6.9
🟨	**Low**	CVSS rating 0.1 - 3.9
🟩	**None**	CVSS rating 0.0

More information about CWE can be found on MITRE's website: https://cwe.mitre.org/.

More information about CVSS can be found on the Forum for Incident Response and Security Teams' (FIRST) website: https://www.first.org/cvss.

3.2 Total Findings

During the engagement, no vulnerabilities were identified.

3.3 Findings by Asset

The following section breaks down the state of security and findings for each individual asset that was tested during the engagement

3.3.1 Asset: hackerone.com

State of security: hackerone.com

No vulnerabilities were identified during the pentest.

Vulnerability summary: hackerone.com

No vulnerabilities were identified during the pentest.

3.3.2 Asset: api.hackerone.com

State of security: api.hackerone.com

No vulnerabilities were identified during the pentest.

Vulnerability summary: api.hackerone.com

During the engagement, no vulnerabilities were found in this asset.

3.3.3 Asset: 66.232.20.0/23

State of security: 66.232.20.0/23

No vulnerabilities were identified during the pentest.

Vulnerability summary: 66.232.20.0/23

During the engagement, no vulnerabilities were found in this asset.

3.3.4 Asset: 34.213.196.80/28

State of security: 34.213.196.80/28

No vulnerabilities were identified during the pentest.

Vulnerability summary: 34.213.196.80/28

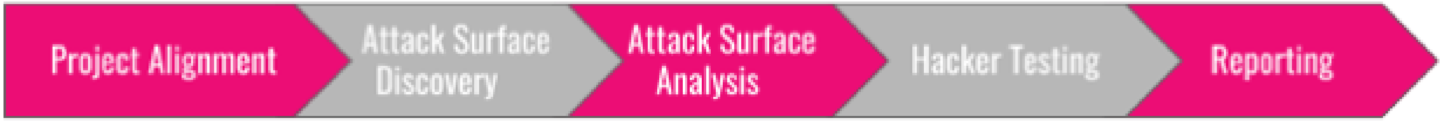
During the engagement, no vulnerabilities were found in this asset.

4. Methodology

4.1 Overview

A HackerOne pentest engagement follows a series of methodologies, checklists, and guidelines to ensure a balance between consistent customer experience, coverage of testing, and depth of testing. HackerOne develops these tools using industry best practices such as OSSTMM, OWASP, NIST, PTES, and ISSAF; as well as, proprietary knowledge gained through HackerOne's platform that services hundreds of on-going and/or timeboxed engagements and a community of over 1,000,000 hackers. Using this combination of best practices and proprietary experience HackerOne is confident that its penetration tests provide a thorough level of security assurance and an unbiased assessment of the state of security for its customers. This section covers the engagement experience and approach.

4.2 Engagement Phases



4.2.1 Project Alignment

HackerOne leverages experts from several internal teams to support customers and understand the goals and expectations for the pentest engagement. HackerOne then works with our community to select highly talented and qualified pentesters that best fit the individual customer's needs and technologies. HackerOne works with the customer to establish Rules of Engagement for the testing activities where applicable, and establishes lines of communication for all stakeholders to ensure that risks to the in-scope assets are minimized. The outcome of this phase is a fully aligned team from customer to hackers to ensure the testing engagement launches with the utmost chance of success.

4.2.2 Attack Surface Discovery

The selected pentesters for the engagement begin their testing efforts by conducting initial discovery efforts including tasks such as ensuring hosts are alive and stable, understanding all possible functionalities, and identifying access levels that exist on the in-scope assets. During this phase findings may certainly be discovered; however, the true intent of this phase is for pentesters to familiarize themselves with the environment and conduct initial research towards the customer's in-scope assets. The outcome of this phase is that the Pentest Team is familiar with the assets and environment that they are conducting testing against/within.

4.2.3 Attack Surface Analysis

During this phase, the Pentest Team will begin active testing activities to understand the state of perimeter defenses and identify the most likely attack vectors for the environment. The Pentest Team will also begin looking at core functionality to begin to identify initial weaknesses in the system. The outcome of this phase is to identify likely attack vectors, and gain a deeper understanding towards the state of security for the assets/environment being tested.

4.2.4 Hacker Testing

In this phase, HackerOne empowers the Pentest Team with both high level coverage requirements to ensure breadth of coverage, and detailed testing guides to ensure depth of coverage towards the assets in-scope for the engagement while also allowing for unique experience that each pentester brings to the engagement. During this phase, pentesters launch their most aggressive attacks towards the in-scope assets in an effort to ensure the most thorough level of security assurance for the customer. The outcome of this phase is to gain an appropriate level of understanding towards the security assurance for all assets engaged.

4.2.5 Reporting

During this phase HackerOne collaborates with the Pentest Team and the customer to ensure that all testing efforts and details towards findings are accurately gathered and included in deliverables for the customer. HackerOne’s reports are an impartial reflection of the assessment conducted against the customers assets and while they may be customized, they can not be influenced by the customers directive. The goal of this phase is to capture the true state of security for the assets in-scope, from HackerOne’s perspective, in a media form that is transferable and reusable as needed.

5. About HackerOne

HackerOne is trusted by over 1,350 organizations worldwide to find and fix security vulnerabilities using the largest team of security researchers on the planet.

Our community of over 1,000,000 researchers has found over 120,000 valid vulnerabilities for organizations including Starbucks, Google, Lufthansa, Toyota, Hyatt, and Goldman Sachs, as well as for high-profile programs for the U.S. Department of Defense such as Hack the Pentagon, Hack the Army, Hack the Air Force, and Hack the Marines.

HackerOne customers worldwide depend on our penetration testing products and services to secure their applications, data, and people, and to make the internet a safer place for everyone.

6. Appendix

Appendix A. HackerOne Security Checklists

HackerOne Web Security Checklist

Check Name	Description
Unvalidated Redirects and Forwards	Unvalidated redirects and forwards are possible when a web application accepts untrusted input that could cause the web application to redirect the request to a URL contained within untrusted input. By modifying untrusted URL input to a malicious site, an attacker may successfully launch a phishing scam, steal user credentials, or bypass referrer checks to perform chained attacks.
Using Components with Known Vulnerabilities	Using components with known vulnerabilities occurs when application developers or deployers fail to keep third-party libraries/tools up to date. This can result in cases where known vulnerabilities and their exploits are able to apply outside of their original context.
Insecure Deserialization	Insecure deserialization vulnerabilities occur when attacker-controlled data is directly deserialized, often leading to remote code execution. This could affect Java deserialization via `ObjectInputStream`, PHP via `unserialize`, Python via `pickle` or `marshal`, and many others.

Cross-Site Request Forgery (CSRF)	Cross-site request forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. CSRF attacks specifically target state-changing requests, not theft of data, since the attacker has no way to see the response to the forged request. An attacker may be able to trick the users of a web application into executing actions of the attacker's choosing. If the victim is a normal user, a successful CSRF attack can force the user to perform state changing requests like transferring funds, changing their email address, and so forth. If the victim is an administrative account, CSRF can compromise the entire web application.
Cross-Site Scripting (XSS)	Cross-site scripting vulnerabilities occur when an application lacks proper output encoding when inserting data into HTML or JavaScript content. This leads to the ability to execute attacker-controlled JavaScript code in the context of a victim browser; this may be a user's browser or a headless browser on the server side.
Security Misconfiguration	Security misconfiguration occurs when application development or deployment does not follow security best practices. Security misconfiguration can happen at any level of an application stack, including the network services, platform, web server, application server, database, frameworks, custom code, and pre-installed virtual machines, containers, or storage.
Broken Access Control	Broken access control occurs when an application lacks thorough permission or role checks. These missing checks typically lead to unauthorized information disclosure, modification or destruction of data, or performing a business function outside of the typical limits of the user.
XML External Entities (XXE)	XML external entities vulnerabilities occur when an application is processing untrusted XML while allowing external entity declarations. This often leads to file access, SSRF attacks, and more.

Sensitive Data Exposure	Sensitive data exposure relates to issues regarding insecure or missing cryptographic protocols. This may affect data transmission, storage, or access.
Broken Authentication	Broken authentication occurs when an application lacks controls around the authentication process or session management. This may lead to an attacker being able to compromise an individual account or even a system-wide authentication bypass.
Injection	Injection flaws occur when unvalidated/unfiltered user-supplied data is used directly by an application. Injection vulnerabilities are often found in SQL queries, OS commands, XML parsers, SMTP headers, expression languages, and ORM queries.

Appendix B. Test Restrictions

No testing restrictions were encountered during the engagement.

Appendix C. Tools

The pentest team used the following tools:

- Burp Suite Pro
- Burp Suite Pro Extensions
- Metasploit
- Acunetix
- Nessus
- OpenVAS
- Nmap
- Ffuf

End of Report