# **Penetration Test Report for Hack The Box’s Cap**

## 

by

**Lydia Gracia (18222035)**

# **Disclaimer**

*Note that this assessment may not disclose all vulnerabilities that are present on the systems within the scope of the engagement. This report is a summary of the findings from a “point-in-time” assessment made on* Cap*’s environment. Any changes made to the environment during the period of testing may affect the results of the assessment.*

# Findings

During the assessment, we uncovered a total of two (2) findings that pose a material risk to Cap’s information systems. we also identified one informational finding that, if addressed, could further strengthen Cap’s overall security posture. Informational findings are observations for areas of improvement by the organization. The below table provides a summary of the findings by security level.

| **Finding Severity** | | |
| --- | --- | --- |
| **High** | **Medium** | **Low** |
| 2 | 0 | 0 |

Below is a high-level overview of each finding identified during testing.

| **Finding #** | **Severity Level** | **Finding Name** |
| --- | --- | --- |
| 1. | High | Insecure Direct Object Reference (IDOR) |
| 2. | High | Linux Privilege Escalation |
| 3. | Info | Discourage Password Reuse |

## Insecure Object Reference (IDOR)

We found an Insecure Direct Object Reference (IDOR) vulnerability in the web application. This issue is raised by an improper access control mechanism, allowing unauthorized users to access data belonging to other users by manipulating the URL parameters. Specifically, the application’s URL structure for retrieving packet captures that followed a pattern of /data/<id> where <id?> is an incrementing integer. By incrementing the ID parameter, we were able to access packet capture files of other users. This vulnerability further leads to revealing sensitive information such as plaintext credentials.

While there’s no direct impact on data integrity and availability, as there is no sign of data altering or being inaccessible for the authorized user, this vulnerability compromises confidentiality.

## Linux Privilege Escalation

The machine also has a vulnerability related to privilege escalation on Linux due to the incorrectly set file capabilities. Examination proves Python has the capability to CAP\_SETUID and CAP\_NET\_BIND\_SERVICE, which enable a non-privileged user to modify their user ID (UID) and obtain root access.

An attacker might exploit these capabilities once they gain non-privileged credentials to the system and increase their privileges to take complete control of the system. As a result, the system is vulnerable to availability, confidentiality, and integrity threats.

## Discouraging Password Reuse

To enhance overall security, it is advisable to not reuse the same password across multiple services. One of the reasons to not reuse your password is because an attacker who gains credentials from one service can exploit them to gain entry into other applications if the same password is used. The system will significantly be prone to unauthorized access.

# 

# Remediation

# 

As a result of this assessment there are several opportunities for Cap to strengthen its internal network security. Remediation efforts are prioritized below starting with those that will likely take the least amount of time and effort to complete. Cap should ensure that all remediation steps and mitigating controls are carefully planned and tested to prevent any service disruptions or loss of data.

**Short Term**

* [Finding 1] Implement proper authorization
* [Finding 2] Remove unnecessary capabilities from binaries.
* [Finding 3] Educate users about password reuse.

**Long Term**

* [Finding 3] Implement Multi-Factor Authentication (MFA).
* [Finding 2] Execute elevated capabilities in a controlled environment.

## Implement Proper Authorization

Never rely solely on user input to determine which resources to access. Ensure you validate user access to each object based on their permissions. Implement this using the recommended approach for your web framework. Additionally, avoid using incremental identifiers and switch to more complex identifiers.

## Remove Unnecessary Capabilities from Binaries

Determine if the packet capture feature is absolutely necessary for the web app and consider implementing it with less privilege. As a short term solution, Cap can disable Python’s elevated privilege using setcap. This solution will probably result in loss of functionality to capture packets, but it will significantly reduce system vulnerability.

## Implement Multi-Factor Authentication (MFA)

Multi Factor Authentication enhances security by requiring an additional form of verification every time a user login to a service. Cap can set it up for SSH on a Linux server by installing and configuring a PAM module for MFA. MFA will prevent attackers from leveraging stolen or reused passwords.

## Execute Elevated Capabilities in a Controlled Environment

Rather than running the elevated privilege on the main machine, Cap can consider isolating it in a container or virtual machine with strict security configuration. This approach might increase the risk of packet loss, but it will mitigate the security vulnerabilities on the main machine.

# 

# Testing Methodology

# 

The testing methodology was split into three phases: Reconnaissance, Target Assessment, and Execution of Vulnerabilities. During reconnaissance, we gathered information about Cap’s network systems using port scanning and other enumeration methods to refine target information. Next, we conducted our targeted assessment. We simulated an attacker exploiting vulnerabilities in Cap’s network. We gathered a walkthrough and the evidence of vulnerabilities during this phase that can be accessed on[11\_deliverables](https://docs.google.com/document/d/1UGjOvPxkQjDpYrNIcPC7oeWf9nZ0U_w3hRo9-9kn0Uo/edit?usp=sharing).

# 

# Appendices

## Appendix A - Finding Severities

Each finding has been assigned a severity rating of high, medium, or low. The rating is based off of an assessment of the priority with which each finding should be viewed and the potential impact each has on the confidentiality, integrity, and availability of Inlanefreight’s data.

| **Rating** | **Severity Rating Definition** |
| --- | --- |
| High | Exploitation of the technical or procedural vulnerability will cause substantial harm. Significant political, financial, and/or legal damage is likely to result. The threat exposure is high, thereby increasing the likelihood of occurrence. Security controls are not effectively implemented to reduce the severity of impact if the vulnerability were exploited. |
| Medium | Exploitation of the technical or procedural vulnerability will significantly impact the confidentiality, integrity, and/or availability of the system, application, or data. Exploitation of the vulnerability may cause moderate financial loss or public embarrassment. The threat exposure is moderate-to-high, thereby increasing the likelihood of occurrence. Security controls are in place to contain the severity of impact if the vulnerability were exploited, such that further political, financial, or legal damage will not occur. - OR - The vulnerability is such that it would otherwise be considered High Risk, but the threat exposure is so limited that the likelihood of occurrence is minimal. |
| Low | Exploitation of the technical or procedural vulnerability will cause minimal impact to operations. The Confidentiality, Integrity and Availability (CIA) of sensitive information are not at risk of compromise. Exploitation of the vulnerability may cause slight financial loss or public embarrassment. The threat exposure is moderate-to-low. Security controls are in place to contain the severity of impact if the vulnerability were exploited, such that further political, financial, or legal damage will not occur. - OR - The vulnerability is such that it would otherwise be considered Medium Risk, but the threat exposure is so limited that the likelihood of occurrence is minimal. |