OWASP Top 10 – Lab Report

Submitted By:
Name: Arsalan Khan
Position/Role: Internee
Date: August 10, 2025

Platform:
TryHackMe

Objective:
To understand and explore the OWASP Top 10 web application security risks, focusing on identifying, exploiting, and mitigating common vulnerabilities. This report covers **7 practical tasks** from the room.

Tools Used:
• Web Browser
• SQLite3 (for database analysis)
• CrackStation (for hash cracking)
• TryHackMe AttackBox

Task 1
Introduction to OWASP Top 10

This TryHackMe room breaks down each OWASP Top 10 vulnerability, providing details on what the vulnerabilities are, how they occur, and how they can be exploited. The room includes practical challenges to put the theory into practice.

The OWASP Top 10 topics covered are:

- Broken Access Control
- Cryptographic Failures
- Injection
- Insecure Design
- Security Misconfiguration
- Vulnerable and Outdated Components
- Identification and Authentication Failures
- Software and Data Integrity Failures
- Security Logging & Monitoring Failures
- Server-Side Request Forgery (SSRF)

This room is designed for beginners and assumes no prior security knowledge.

Task 2
Accessing Machines

To complete practical tasks, you will need to interact with virtual machines on TryHackMe.

You can start by pressing the green **Start Machine** button to deploy a machine.

Task 3
Broken Access Control

Access control mechanisms protect certain pages or functionalities on websites, allowing only authorized users (e.g., admins) to access them.

Broken access control occurs when unauthorized users can bypass these restrictions, enabling them to:

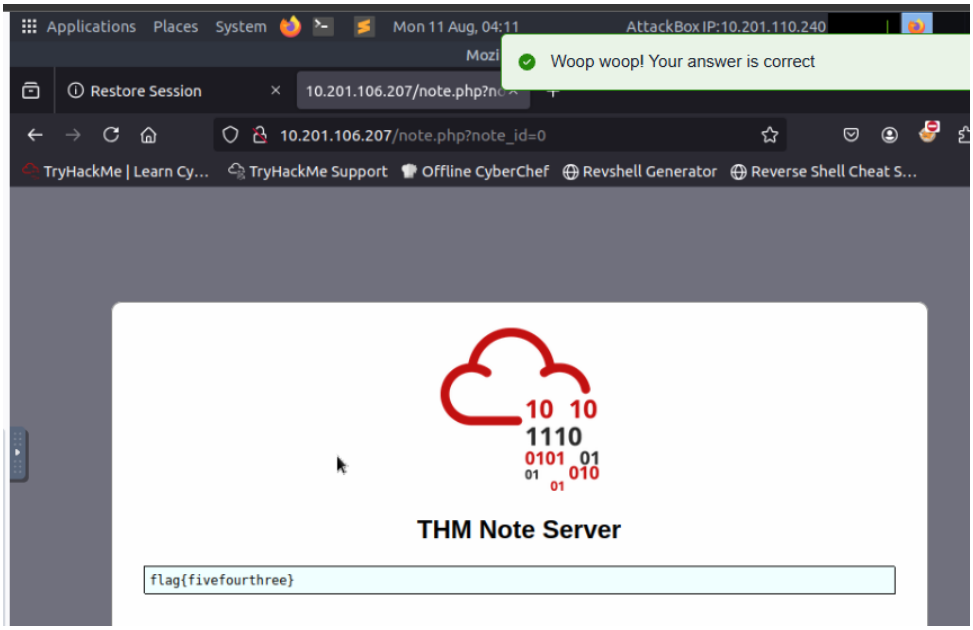
- View sensitive information of other users
- Access or perform unauthorized functions

A real-world example from 2019 involved a vulnerability in YouTube, where an attacker could retrieve frames from videos marked as private, effectively bypassing privacy settings and reconstructing the video. This clearly demonstrated broken access control.

Task 4
Broken Access Control (IDOR Challenge)

Insecure Direct Object Reference (IDOR) is a vulnerability where an application exposes a direct reference to an object (like a file, user account, or database entry) through identifiers such as URLs or parameters. If the application does not properly validate whether the logged-in user is authorized to access the referenced object, attackers can manipulate the reference to access data they shouldn't see.

Example:
A banking site URL like `https://bank.thm/account?id=111111` shows user-specific data. If an attacker changes the `id` to `222222` and the application fails to verify ownership, they might access another user's sensitive account information.



Task 5: Cryptographic Failures

Overview:
Cryptographic failures occur when cryptographic algorithms are misused or not used at all, leading to vulnerabilities in protecting sensitive information. Web applications rely on cryptography to ensure data confidentiality both during transmission (data in transit) and while stored on servers (data at rest).

For example, a secure email application encrypts communications between the user's browser and the server to prevent eavesdropping. Additionally, emails may be encrypted on the server to prevent even the provider from reading the contents.

Common consequences of cryptographic failures include the unintended disclosure of sensitive user data such as personal details or login credentials. Attackers can exploit these weaknesses through techniques like Man-in-the-Middle (MitM) attacks, capturing and decrypting weakly protected data.

The deployed machine contains a vulnerability related to cryptographic failures. Further tasks provide material and practical challenges to understand and exploit this issue.

Task 6: Cryptographic Failures (Supporting Material 1)

Overview:
This task introduces flat-file databases, specifically SQLite databases, commonly used in smaller web applications due to their simplicity and portability. Unlike full database servers, SQLite databases are stored as single files on disk.

A key security concern arises if such a database is stored within the website's root directory, making it accessible for download by users. This can lead to **Sensitive Data Exposure**, as attackers can retrieve and query the database independently.

Task 7: Cryptographic Failures (Supporting Material 2)

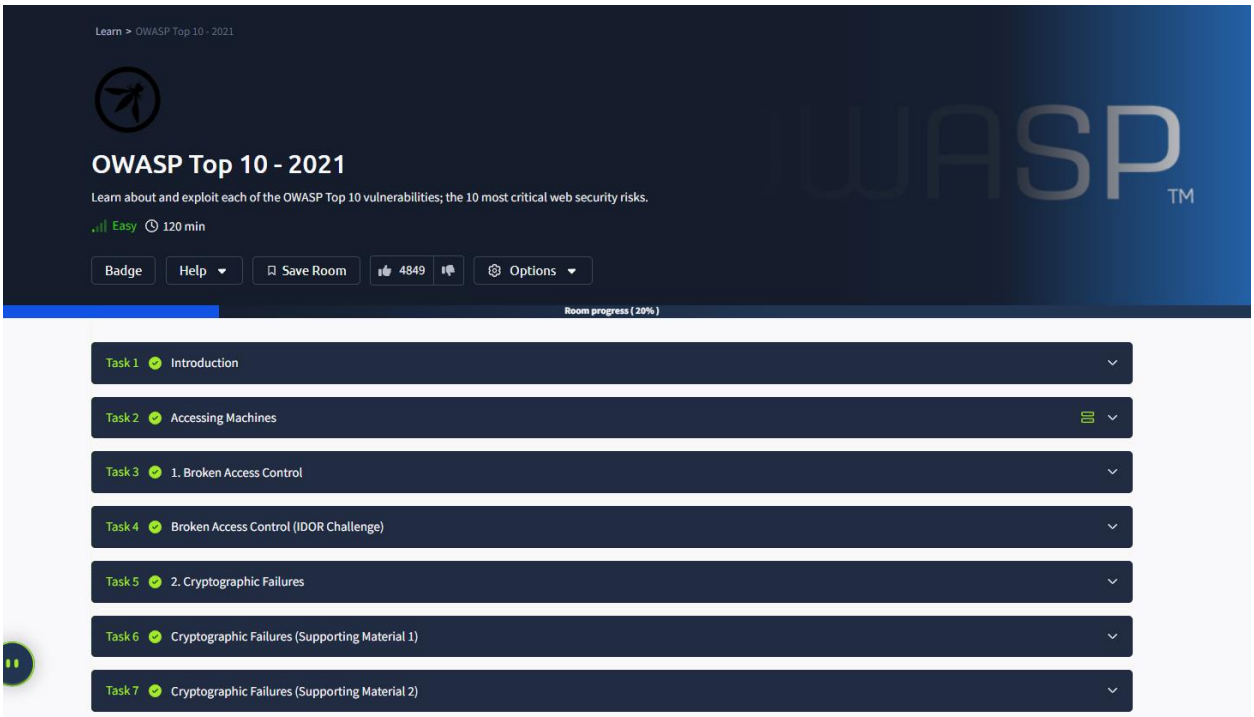
Overview:
In the previous task, we extracted password hashes from the SQLite database. Now, we focus on cracking these hashes to reveal the actual passwords.

While advanced hash-cracking tools are available on platforms like Kali Linux, this task uses the online tool **Crackstation**, which is effective for cracking weak hashes such as MD5.

For example, the hash `5f4dcc3b5aa765d61d8327deb882cf99` (found for user Joy Paulson) was pasted into Crackstation. After completing the Captcha and clicking "Crack Hashes," the password revealed was `"password"`—a very weak password.

Key Points:

- Crackstation uses a massive wordlist to crack hashes.
- If a password is not in the wordlist, Crackstation will fail to crack the hash.
- The hashes in this challenge are weak MD5 hashes, suitable for Crackstation.
- Uncrackable hashes in this challenge are intentionally designed.


Learn > OWASP Top 10 - 2021
OWASP Top 10 - 2021
Learn about and exploit each of the OWASP Top 10 vulnerabilities; the 10 most critical web security risks.
Easy
120 min
Badge
Help
Save Room
4849
Options
Room progress (20%)
Task 1 Introduction
Task 2 Accessing Machines
Task 3 1. Broken Access Control
Task 4 Broken Access Control (IDOR Challenge)
Task 5 2. Cryptographic Failures
Task 6 Cryptographic Failures (Supporting Material 1)
Task 7 Cryptographic Failures (Supporting Material 2)