Sri Lanka Institute of Information Technology

A blue and orange logo

AI-generated content may be incorrect.

REPORT-TRYHACKME ROOM

Exploiting Web Authentication & Identification Flaws

IT23363366

Contents

[1. Room Overview 3](#_Toc196162041)

[2. Learning Objectives 4](#_Toc196162042)

[3. Room Structure 5](#_Toc196162043)

[Task 1: Introduction 5](#_Toc196162044)

[Task 2: The Enumeration – Authentication & Identification 5](#_Toc196162045)

[Task 3: Vulnerabilities in Authentication & Identification 5](#_Toc196162046)

[Task 4: Weak Credentials and Brute Force on Admin Login 6](#_Toc196162047)

[Task 5: Brute Force – Weak Authentication and Identification Vulnerabilities 7](#_Toc196162048)

[Task 6: Email Enumeration – Weak Authentication and Identification Vulnerabilities 8](#_Toc196162049)

[Task 7: Understanding Password Reset Vulnerabilities 9](#_Toc196162050)

[Task 8: Exploiting Predictable Tokens of the Password Reset Link – Weak Authentication and Identification Vulnerabilities 10](#_Toc196162051)

[4. Room Link 12](#_Toc196162052)

[5. Reflection 12](#_Toc196162053)

# 1. Room Overview

Topic Title: **Exploiting Web Authentication & Identification Flaws**

Room Name- **owaspwebshield**

**Summary:**-

This TryHackMe room is designed to guide learners through real-world authentication and identification vulnerabilities that are commonly discovered in web applications. Based on **personal bug bounty experience**, the room simulates a many vulnerable web applications that contains flaws in user enumeration, weak credentials, predictable password reset tokens, and insecure reset validation mechanisms.  
  
The goal of this room is to equip participants with a practical understanding of how attackers exploit these flaws and how developers and security professionals can prevent them. With allowing users to gain hands-on experience on following topics

* Key Concepts Covered:  
  - User Enumeration  
  - Weak and Default Credentials  
  - Brute Force Attacks  
  - Predictable Token Exploitation in Password Resets  
  - Insecure Password Reset Logic  
  - Real-world techniques using tools like Burp Suite Intruder

**Future Vision:-**

This room is part of a larger initiative to build **hands-on learning** environments for each of the OWASP Top 10 vulnerabilities. The "**owaspwebshield**" room currently focuses on identification and authentication flaws and serves as the first step in this ongoing series. Future updates will include new tasks addressing additional web security topics such as injection, security misconfiguration, broken access control, and more—providing a comprehensive and practical approach to mastering modern web application vulnerabilities.

The end goal is to build a complete, interactive learning path that not only simulates real-world attack scenarios but also equips participants with the skills to analyze, exploit, and secure web applications effectively. The name **"owaspwebshield"** reflects this broader vision.

# 2. Learning Objectives

* Understand the risks associated with poor authentication and password reset implementations.
* Gain insight into how flawed logic in login and password reset functions can lead to serious security issues.
* Identify user enumeration vulnerabilities in login and reset flows.
* Learn how to detect and exploit error-based information disclosure to enumerate valid usernames or emails.
* Exploit weak credentials using brute force attacks.
* Understand how attackers leverage tools like Burp Suite to perform credential stuffing and brute force attacks.
* Capture and manipulate password reset tokens.
* Practice intercepting and modifying password reset requests to gain unauthorized access.
* Analyze and bypass insecure token validation.
* Investigate scenarios where predictable or improperly validated tokens can be abused.
* Develop mitigation techniques to secure authentication and password recovery systems.
* Acquire best practices for hardening login and reset logic to prevent common attacks.
* Improve hands-on experience with Burp Suite for web security testing.
* Enhance practical skills in using Burp Suite to test web applications for authentication flaws.

# 3. Room Structure

### **Task 1: Introduction**

In this task, participants are introduced to a deliberately vulnerable **various web applications** that are **invidualluly hosted related to specific challenges**. I have developed this platform to simulate real-world scenarios involving common authentication and password reset vulnerabilities.

Users will explore identification and authentication weaknesses such as **weak credentials, brute force login attempts, user enumeration, and insecure password reset mechanisms**. The room provides a structured, hands-on learning experience with walkthrough-style challenges, helping participants understand how these vulnerabilities occur and how they can be **exploited in practice**.

### **Task 2: The Enumeration – Authentication & Identification**

In this task, I aimed to provide participants with theoretical knowledge on how authentication and identification enumeration works. The focus was on educating users about how attackers identify valid usernames and email addresses by analyzing system responses. This understanding is essential for tackling the practical challenges in the room.

**Concepts Covered:-**

* Types of Enumeration in Cybersecurity
* Focus Area: Authentication & Identification Enumeration
* Categories & Techniques

### **Task 3: Vulnerabilities in Authentication & Identification**

**Description:-**  
This task provides foundational theory and real-world context before users attempt hands-on exploitation. It focuses on common weaknesses in authentication and identification systems, helping users understand the underlying causes and effects of poor implementation. **By exploring various CVEs and misconfigurations**, users gain insight into what makes a system vulnerable and **how these flaws can be exploited.**

**Topics Covered:**

* Difference between Authentication and Identification.
* Real-world vulnerabilities (e.g., **CVE-287, CVE-297, CVE-384**).
* Categories of common weaknesses (e.g., session fixation, improper certificate validation, improper authentication).
* Example attack scenarios and step-by-step logic of how an attacker would exploit these vulnerabilities.
* Common causes of weak authentication:
  + Predictable reset tokens
  + Poor session handling
  + Lack of multi-factor authentication
  + Weak or hardcoded credentials
* Preventive techniques and best practices:
  + Strong session management
  + Secure token generation
  + Implementing MFA
  + Proper error handling and rate limiting

**Purpose:-**  
This is a **theory and walkthrough-based task** intended to **educate** users before they proceed to the practical challenges in the next tasks. It strengthens their understanding of vulnerabilities, which enhances their effectiveness in identifying and exploiting those flaws during hands-on tasks.

**Expected Outcome:**

* Understand the **difference between authentication and identification.**
* Recognize **real-world vulnerabilities and how they apply to web apps**.
* Learn why **secure implementation of authentication** logic is critical.
* Gain insight into **attacker perspectives and how to defend against them**.

### **Task 4: Weak Credentials and Brute Force on Admin Login**

**Scenario:-**  
Participants are given a login page built using a Flask application. The login page is styled with HTML and CSS. By analyzing responses to invalid inputs, participants must enumerate valid usernames and brute-force weak credentials for an admin account. Note that the username is not explicitly 'admin'—this requires enumeration to identify.

**Expected Outcome:**

* Identify a valid admin-like username through user enumeration.
* Successfully brute-force the password for the identified username.
* Capture the first flag post-login.

**Technologies Used:**

* **Flask Application (backend)**: For setting up the login page and handling requests.
* **HTML/ Taiwan CSS**: For the front-end interface of the login page.
* **Link: -** <https://web-production-e6d4.up.railway.app/> of the login page.

**NOTE**: In this task Users was provided with screenshots, attacking instructions and relevant wordlists in order to make the task beginner friendly.

### **Task 5: Brute Force – Weak Authentication and Identification Vulnerabilities**

**Scenario:-**  
Participants are presented with the same login page used in Task 4, hosted on a Flask application. However, this challenge shifts the focus from basic brute-force techniques to a more subtle approach involving weak password hygiene and light social engineering.

The account *raahim* is known to exist within the system. The participant's job is to guess the password — but instead of relying solely on wordlists, they are encouraged to think from the perspective of an attacker who might use personal or behavioral clues to craft guesses.

The challenge is designed to make participants think like a real-world attacker by blending technical knowledge with psychology and open-source intelligence (OSINT) techniques.

**Expected Outcome:-**  
• Recognize that raahim is a valid user account.  
• Use clues (either from hints, the room’s description, or online presence) to craft and guess a weak password that could plausibly be used by the user.  
• Successfully log in and retrieve the second flag.

**Technologies Used:-**  
• **Flask Application (backend):** Manages authentication logic.  
• **HTML / Taiwan CSS:** Provides the login page UI.  
• **Same Login Page Link as Task 4:** <https://web-production-e6d4.up.railway.app/>

**Note:** Participants are nudged to use contextual reasoning (like guessing a password based on common patterns or personal habits), rather than relying purely on brute-force automation. Hints and light guidance are provided to make the challenge engaging and achievable without prior OSINT experience.

### **Task 6: Email Enumeration – Weak Authentication and Identification Vulnerabilities**

**Scenario:-**  
In this task, participants are introduced to a login system vulnerable to email enumeration via verbose error messages. When testing different login credentials, the system reveals whether an email exists based on the nature of its response, enabling attackers to identify valid users without any prior knowledge.

After discovering a valid email address, participants are instructed to brute-force the corresponding password using a commonly used wordlist. Once logged in, they can capture a flag from the dashboard. A second flag can be retrieved by triggering the password reset process using the identified email address.

The challenge simulates a real-world scenario where seemingly harmless information disclosure can lead to significant account-level exploitation.

**Expected Outcome:-**  
• Identify a valid email address by analyzing the login response behavior.  
• Use *rockyou.txt* to brute-force the password for the discovered email.  
• Log in successfully and capture the first flag.  
• Trigger the password reset flow and retrieve a second flag.

**Technologies Used:-**  
• **Flask Application (backend):** Powers the login and reset functionalities.  
• **HTML / CSS:** Basic UI for login and password reset.  
**• Challenge URL:** [*https://web-production-59589.up.railway.app/*](https://web-production-59589.up.railway.app/)

**Note:-**  
Participants are encouraged to use **Burp Suite Intruder** or a **custom Python script** provided in the task to speed up the enumeration process. For email discovery, the recommended wordlist is:

* **Email Wordlist:**  
  [*usernames\_gmail.com.txt*](https://github.com/nyxgeek/username-lists/blob/master/usernames-top100/usernames_gmail.com.txt)

For brute-forcing the password:

* **Password Wordlist:** *rockyou.txt*

The main learning focus here is on **identification vulnerabilities**, not brute-force mechanics — hints are included to keep the task efficient and beginner-friendly.

### **Task 7: Understanding Password Reset Vulnerabilities**

**Scenario:-**  
Before jumping into the next hands-on challenge, participants are given a theoretical overview of common weaknesses in password reset mechanisms. This task focuses on educating users about how reset logic typically works and where vulnerabilities tend to appear in real-world applications.

Password reset functionality is often overlooked, but if poorly implemented, it can be a critical entry point for attackers. In this task, users will explore how tokens are generated, how reset links are constructed, and the common pitfalls like predictable tokens, insecure transport, and improper validation.

**Key Concepts Covered:-**  
• How email, SMS, and security question-based reset flows operate.  
• The step-by-step logic of a standard password reset request.  
• Introduction to **Password Reset Poisoning** via manipulated Host headers.  
• Common vulnerabilities, such as:

* Predictable or low-entropy tokens
* Missing or long expiration windows
* Verbose error messages (info disclosure)
* Weak security questions
* Use of HTTP instead of HTTPS

**Expected Outcome:-**  
• Gain a solid conceptual understanding of how password reset mechanisms can be attacked.  
• Be able to identify which flaws are most dangerous in various scenarios.  
• Understand the theory behind the exploit you’ll perform in **Task 8**.

**Purpose:-**  
This is a **theory and walkthrough-based task** intended to **educate** users before they proceed to the practical challenges in the next tasks. It strengthens their understanding of vulnerabilities, which enhances their effectiveness in identifying and exploiting those flaws during hands-on tasks.

**Assessment:-**  
Multiple-choice questions are provided to reinforce the content. These include identifying risks in different reset flows and understanding how token-based systems can be compromised.

**Note:-**  
This task is intentionally non-technical to give users the necessary mental model before diving into the exploit in the next challenge. Treat this as the **"recon and planning"** stage before executing an actual attack.

### **Task 8: Exploiting Predictable Tokens of the Password Reset Link – Weak Authentication and Identification Vulnerabilities**

#### **Scenario:**

In this task, participants explore a password reset mechanism that uses **predictable numeric tokens**. Unlike secure implementations that rely on high-entropy values, this system generates a simple 3-digit token for each reset request. These tokens are not only **guessable**, but also **remain valid for an extended period**, allowing attackers to brute-force their way into user accounts.

Participants are given a known email address and must initiate a password reset. They will then **brute-force the token** using a limited numeric range. A correct token reveals a flag and the victim’s new password. By logging in with those credentials, participants retrieve a second flag — simulating a full account takeover.

This challenge reflects real-world incidents where developers fail to implement proper randomness and expiration in reset flows, resulting in devastating authentication bypasses.

#### **Expected Outcome:**

* Trigger a password reset for a known email address.
* Brute-force the predictable numeric token (between 100–200).
* Retrieve the reset flag and new password.
* Log into the account using the new password and capture the second flag.

#### **Technologies Used:**

* **Flask Application (backend):** Handles reset token generation and validation.
* **HTML / CSS:** Interface for password reset and login screens.
* **Challenge URL:** [*https://web-production-05652.up.railway.app/login*](https://web-production-05652.up.railway.app/login)

#### **Note:**

* The reset token is always a **3-digit number between 100–200**.
* Email delivery is simulated; the format of the reset URL is:

*https://webproduction05652.up.railway.app/reset\_password?toke=xyz*

* Participants must capture the token parameter in Burp Suite and use **Intruder** to brute-force it.
* A successful brute-force attempt will reveal a **flag** and a **new password** for the user *admin@admin.com.*

#### **Recommended Tools & Wordlists:**

* **Burp Suite Intruder** (for brute-forcing token)
* **Crunch or seq** (to generate numeric payloads)

*Seq 100 200 > token.txt*

#### **Hints for Participants:**

* No need to enumerate emails; a valid one is already provided (for practice purpose they can enumerate:*admin@admin.com*)
* Watch the response length or content in Burp Intruder to identify a successful token.
* Once the token is found, log in using the retrieved password to claim the second flag.

# 4. Room Link

**TryHackMe Room:-**  
[*https://tryhackme.com/jr/owaspwebshield*](https://tryhackme.com/jr/owaspwebshield)

# 5. Reflection

**Personal Insights:-**  
Designing this room was an enriching process. It helped reinforce my understanding of real-world web authentication flaws and how subtle implementation mistakes can lead to major security risks. By breaking down my bug bounty experiences into progressive hands-on tasks, I also learned to think like an educator—how to guide a participant without giving away too much.

**Contribution to the Learning Community:-**  
This room aims to bridge the gap between theory and practice. Rather than just describing flaws, it provides an interactive space to exploit and understand them firsthand. It helps beginners and intermediates develop a deeper understanding of secure authentication mechanisms, especially relevant for both developers and cybersecurity learners.