# Report PENETRATION TESTING 2024

Simone Cappabianca - Mat: 5423306
simone.cappabianca@edu.unifi.it

Febbraio 8, 2025

# Contents

# 1 Executive Summary

Questo rapporto descrive i risultati di un penetration test condotto su un **container Docker**, identificato con l'indirizzo IP 172.19.0.3. Il test, effettuato in data 8 Febbraio 2025, aveva lo scopo di valutare la sicurezza del sistema e identificare potenziali vulnerabilità sfruttabili. Le principali scoperte includono la presenza di una **vulnerabilità di SQL Injection** nella procedura di login e una **vulnerabilità di Cross-Site Scripting (XSS)** nella procedura di recupero password.

# 2 Methodology Used

- **Definizione dello Scopo del Test:** Il test è stato eseguito su un singolo container Docker, identificato dall'indirizzo IP 172.19.0.3. Non è stata necessaria alcuna fase di raccolta di informazioni preliminare poichè l'ambiente era preconfigurato.
- **Scansione della Rete:** La scansione di rete è stata eseguita utilizzando `nmap` per identificare i servizi attivi. È stata condotta una scansione ICMP/Ping di livello 3 e una scansione TCP SYN di livello 4.
- **Enumerazione:** Dopo la scansione, è stato effettuato il banner grabbing tramite `nmap` e `telnet` per identificare il sistema operativo e il web server. È stata eseguita una verifica per l'enumerazione UserDir, risultata non attiva.
- **Valutazione delle Vulnerabilità:** La valutazione delle vulnerabilità è stata condotta identificando manualmente la SQL Injection nella procedura di login e la XSS nella procedura di password recovery.
- **Sfruttamento (Exploitation):** Sono state create delle Proof of Concept (PoC) per le vulnerabilità identificate, dimostrando la possibilità di sfruttarle.
- **Post-Sfruttamento:** Sono stati identificati script e query per mantenere l'accesso e raccogliere dati, simulando le azioni di un attaccante in una fase di post-exploitation.

- **Strumenti Utilizzati:** `nmap`, `telnet` e `Metasploit` (per l'enumerazione UserDir, anche se il modulo non ha fornito risultati).

# 3 Findings

- **Sistema Operativo:** Linux 4.15 - 5.8.
- **Web Server:** Apache/2.4.38 (Debian).
- **PHP:** PHP/7.2.34.
- **UserDir:** Il modulo `mod_userdir` non è attivo sul web server.
- **Vulnerabilità di SQL Injection:**
  - **Descrizione:** È presente una vulnerabilità di SQL Injection nella procedura di login che permette di manipolare la query eseguita per il controllo delle credenziali. Inserendo un apice singolo (') nel campo username, è possibile iniettare comandi SQL arbitrari.
  - **Livello di Rischio:** Alto. La SQL Injection permette di bypassare l'autenticazione e di recuperare informazioni dal database.
  - **Proof of Concept (PoC):**
    1. **Username:** `' OR null is null limit 1 #`
       **Paswword:** Qualsiasi valore
    2. **Username:** `' UNION SELECT null, null, user() #`
       **Password:** Qualsiasi valore

    Questi PoC dimostrano che è possibile manipolare la query SQL, ottenendo informazioni sul database o bypassando l'autenticazione.
  - **Evidenze:** Messaggio di errore SQL: `Notice: Invalid query: You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'c9c35cf409344312146fa7546a94d1a6' at line 1 in /var/www/html/login.php on line 63` viene mostrato quando viene inserito un apice singolo nel campo username.
- **Vulnerabilità' di XSS:**

- **Descrizione:** È presente una vulnerabilità di Cross-Site Scripting (XSS) nella procedura di password recovery. Inserendo codice JavaScript nel campo **Email address** è possibile eseguire codice arbitrario nel browser dell'utente.
- **Livello di Rischio:** Medio. Permette l'esecuzione di codice arbitrario nel browser dell'utente e potenzialmente il furto di cookie e sessioni.
- **Proof of Concept (PoC):**
  1. **Email address:** `<SCRIPT>alert('Hello')</SCRIPT>`
  2. **Email address:** `<IMG SRC="pic/1.jpg">`

  Questi PoC dimostrano la possibilità di iniettare codice JavaScript nel campo email.
- **Evidenze:** L'esecuzione di codice JavaScript tramite il campo dell'email e i payload specificati dimostrano la vulnerabilità XSS.

# 4 Remediation Plan

- **SQL Injection:**
  - **Patch:** Utilizzare query parametrizzate (prepared statements) per evitare che l'input dell'utente venga interpretato come codice SQL.
  - **Mitigazione:** Implementare la validazione degli input per impedire l'inserimento di caratteri speciali come l'apice singolo nelle query SQL.
- **XSS:**
  - **Patch:** Implementare la sanificazione dell'input per rimuovere o codificare i tag HTML che possono eseguire script.
  - **Mitigazione:** Utilizzare Content Security Policy (CSP) per limitare le fonti da cui il browser può caricare risorse.
- **Misure aggiuntive:**

- Aggiornare regolarmente il sistema operativo, il server web e tutte le applicazioni in uso per applicare le patch di sicurezza più recenti.
- Disabilitare o proteggere l'accesso a servizi non necessari.
- Implementare un firewall per limitare gli accessi non autorizzati.

# 5 Post-Exploitation (simulato)

Sono stati identificati i seguenti script/query come esempio delle attività di un attaccante in una fase di post-exploitation:

- Script per visuaizzare un file del server:
  ```
  <SCRIPT>function test(){fetch('TEST.txt').
  then(response => response.text()).then(data => alert(data))}test();
  </SCRIPT>
  ```
- SQLi per recuperare l'elenco degli users/customers:
  ```
  '  or null is null INTO OUTFILE '/var/www/html/USERS.txt' #
  ```
- SQLi per recuperare l'elenco delle tabelle del db:
  ```
  '  or null is not null UNION SELECT null, TABLE_NAME, TABLE_SCHEMA
  FROM information_schema.TABLES WHERE TABLE_SCHEMA not like
  'information_schema' AND TABLE_SCHEMA not like 'mysql' AND TABLE_SCHEMA
  not like 'performance_schema' INTO OUTFILE '/var/www/html/TABLES.txt'
  #
  ```

Questi script dimostrano come un attaccante può accedere a file e informazioni dal database, utilizzando le vulnerabilità SQLi e XSS.

# 6 Conclusioni

Questo rapporto evidenzia la presenza di vulnerabilità critiche che richiedono immediata attenzione. Implementando le patch e le mitigazioni suggerite, la sicurezza del sistema può essere notevolmente migliorata.