A cura di Pierantonio Miglietta, Iris Canole, Rebecca Talone, Francesco Miolli, Tiziano Bramonti, Andrea Sottile, Alessandro Ricci

Web Application Exploit Sqli

Obiettivo

Sfruttare la vulnerabilità SQL injection presente sulla Web Application DVWA per recuperare in chiaro la password dell'utente Pablo Picasso (non dimenticarsi che una volta trovate le password, c'è bisogno di un ulteriore step per recuperare la password in chiaro) utilizzando le tecniche illustrate a lezione.
NB: non usare tool automatici come sqlmap. È ammesso l'uso di repeater burp suite.

Requisiti

- Livello di difficoltà DVWA: LOW
- IP Kali Linux: 192.168.13.100/24
- IP Metasploitable: 192.168.13.150/24

Introduzione al problema → SQL Injection

Cos'è →

- La SQL Injection rappresenta una delle vulnerabilità più critiche e diffuse nelle applicazioni web che interagiscono con un database SQL. Questo tipo di attacco si verifica nel momento in cui un'applicazione integra direttamente l'input non filtrato dell'utente all'interno di una query SQL.

Meccanismo di sfruttamento →

- Quando l'applicazione web costruisce una query dinamica concatenando del testo fisso e dei dati inseriti dall'utente (Ad esempio nome_utente o un ID) , un attaccante potrebbe iniettare del codice SQL malevolo.

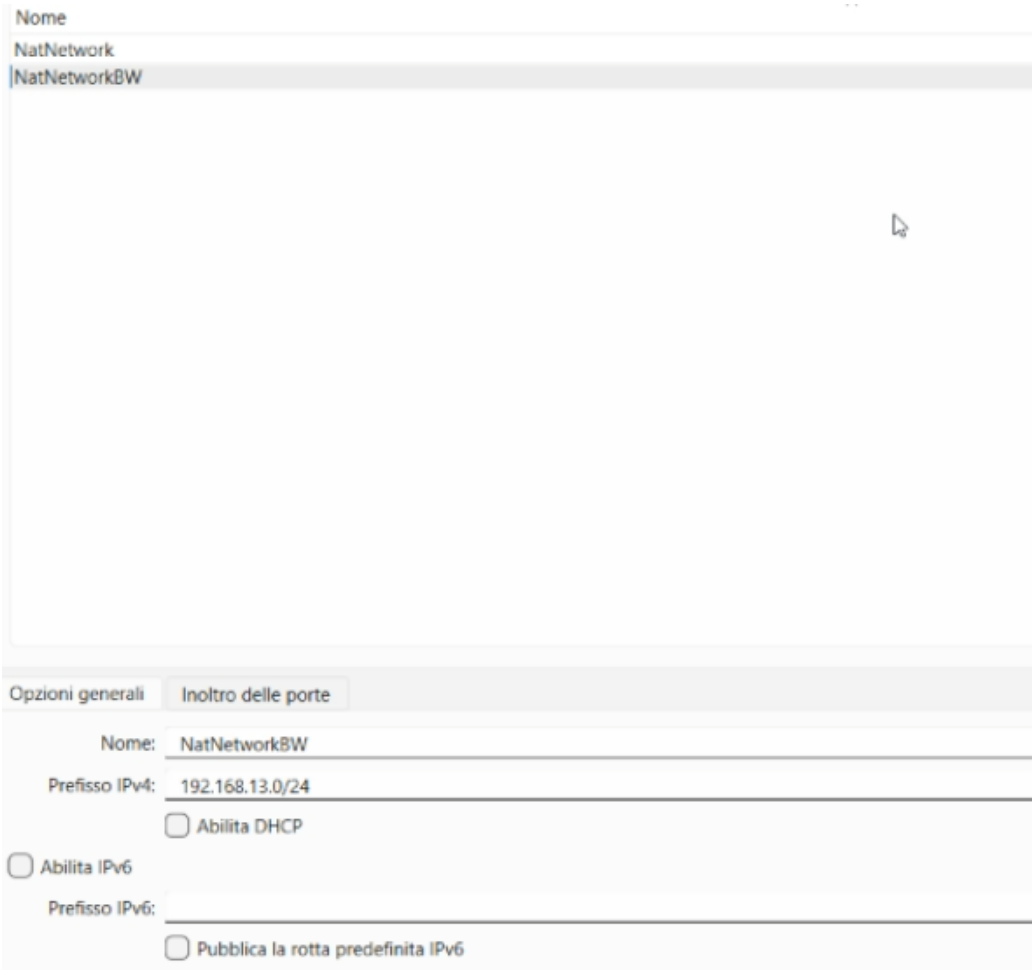
- Esempio → SELECT * FROM users WHERE username = "INPUT UTENTE" AND password = '..' → Qui un utente potrebbe manipolare la logica della query andando a chiudere l'apice e aggiungendo istruzioni differenti come ad esempio *UNION SELECT , OR 1=1* ecc)

Impatto →

- Un attacco SQLi può portare a gravi conseguenze, tra cui :
  1. Aggiramento autenticazione : Con logiche del tipo OR 1=1 per accedere al sistema senza credenziali.
  2. Estrazione di dati sensibili (come vedremo più avanti) : Lettura non prevista di intere tabelle sfruttando la clausola UNION SELECT
  3. Modifica o cancellazione di dati : Con istruzioni del tipo UPDATE o DELETE per alterare o distruggere il contenuto del DB.

Configurazione ambienti di lavoro

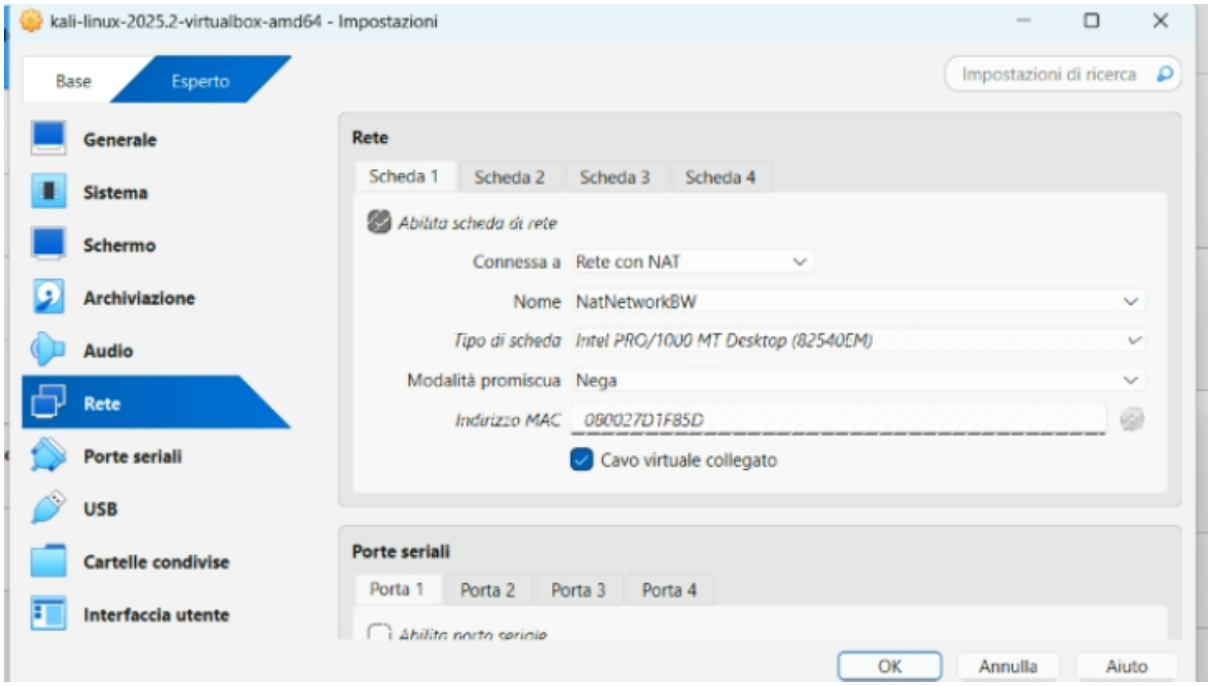
1. Iniziamo con la configurazione di rete del nostro laboratorio virtuale andando a configurare una nuova rete con NAT → NatNetworkBW :



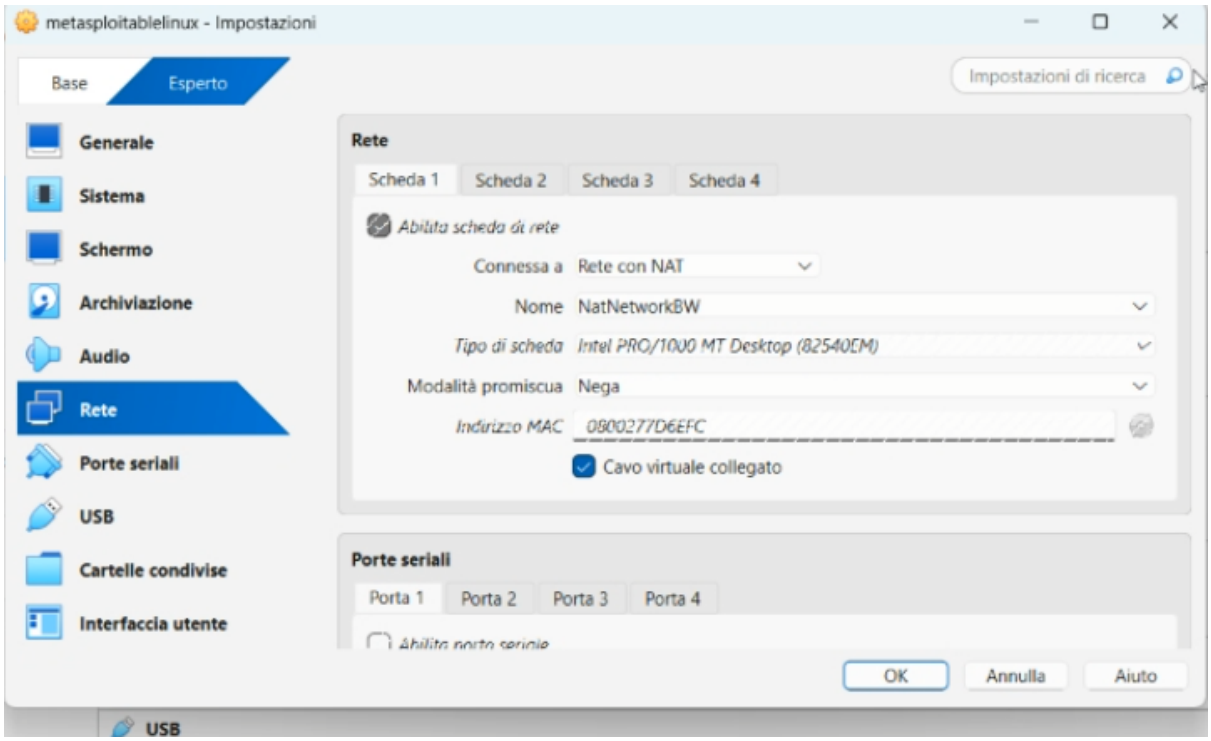
2. Procediamo con la configurazione di Kali e Metasploitable andando a metterle entrambe all'interno della nostra nuova rete NatNetwork :

Impostazioni Kali →

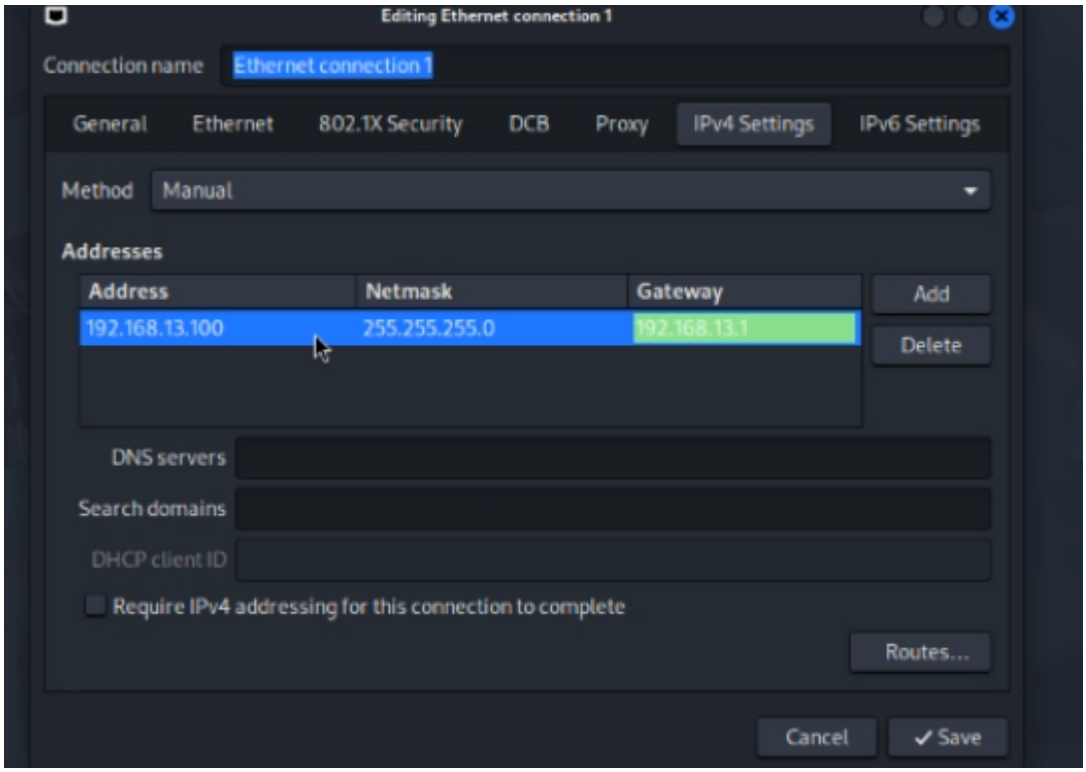
A cura di Pierantonio Miglietta, Iris Canole, Rebecca Talone, Francesco Miolli, Tiziano Bramonti, Andrea Sottile, Alessandro Ricci



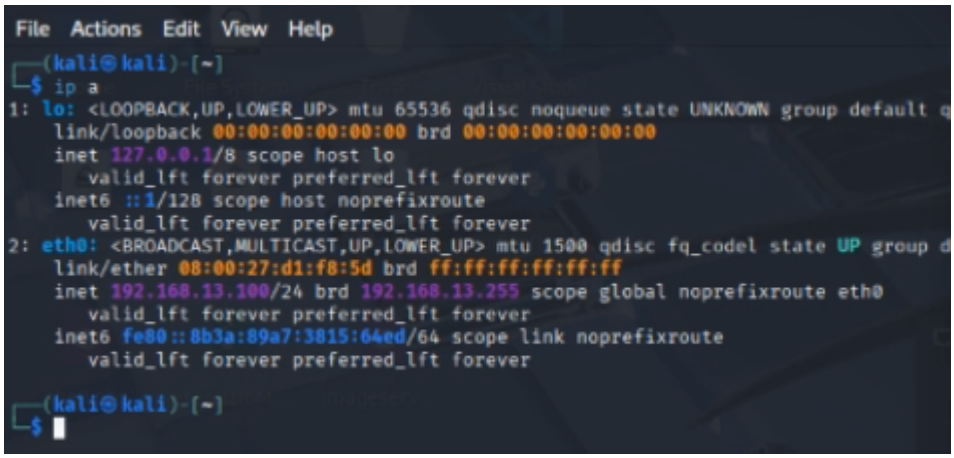
Impostazioni Metasploitable →



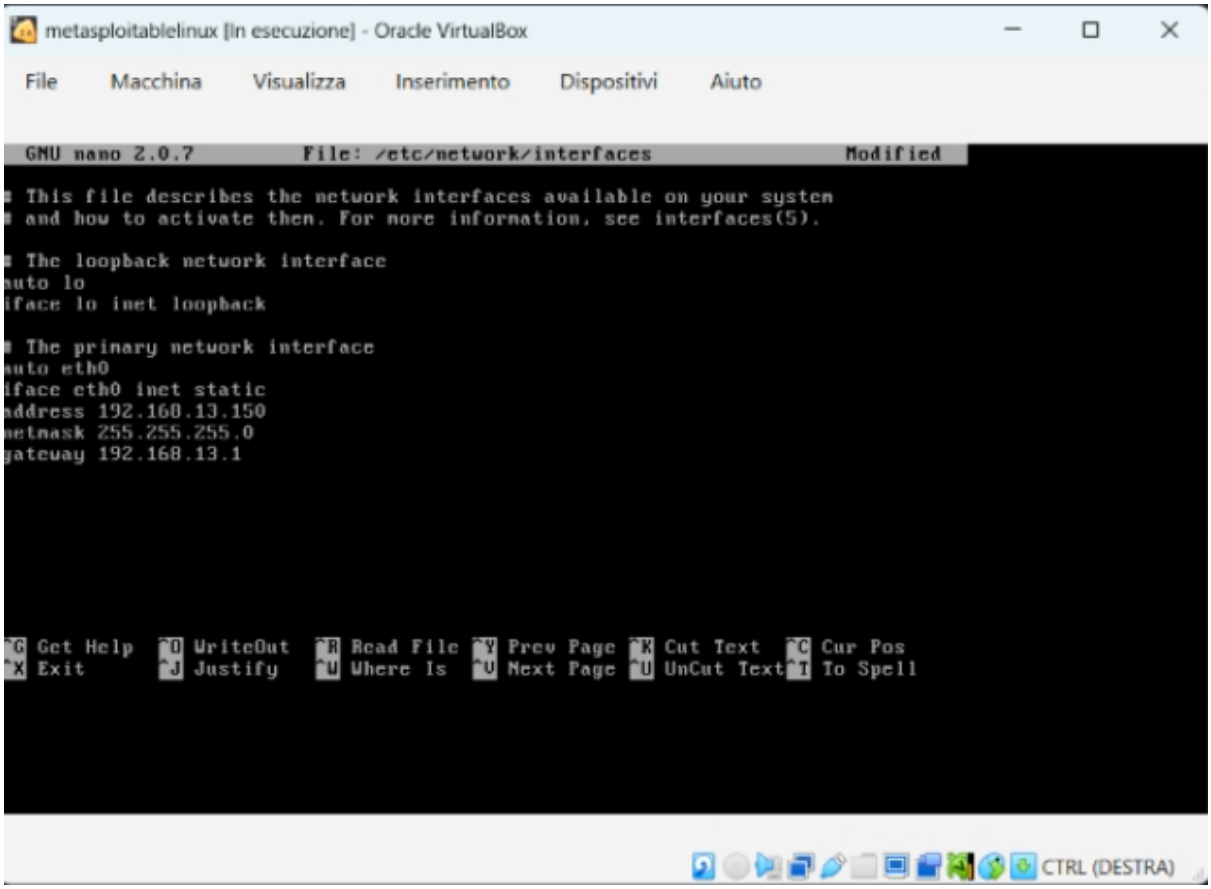
3. Ora sulla VM Kali apportiamo modifiche all'interfaccia di NetworkManager (quindi "Edit Connections") e inseriamo l'IP richiesto per lo scope di questa simulazione →



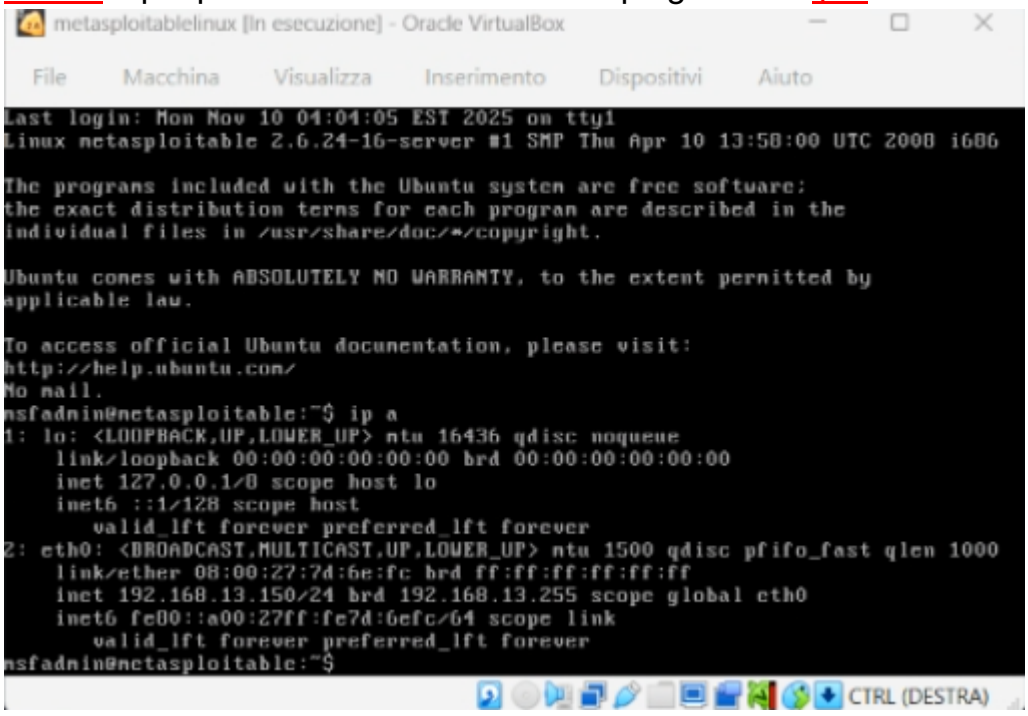
4. Tramite il comando ip a andiamo a verificare l'avvenuta modifica del nostro IP Kali →



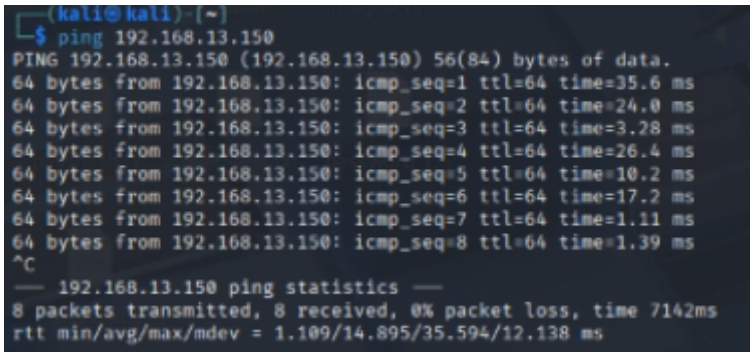
5. Ora procediamo con il cambio dell'IP della VM Metasploitable sul terminale di Metasploitable con il comando sudo nano /etc/network/interfaces:



Effettuiamo un riavvio dell'interfaccia di rete tramite sudo /etc/init.d/networking restart e poi procediamo al controllo del ping tramite ip a



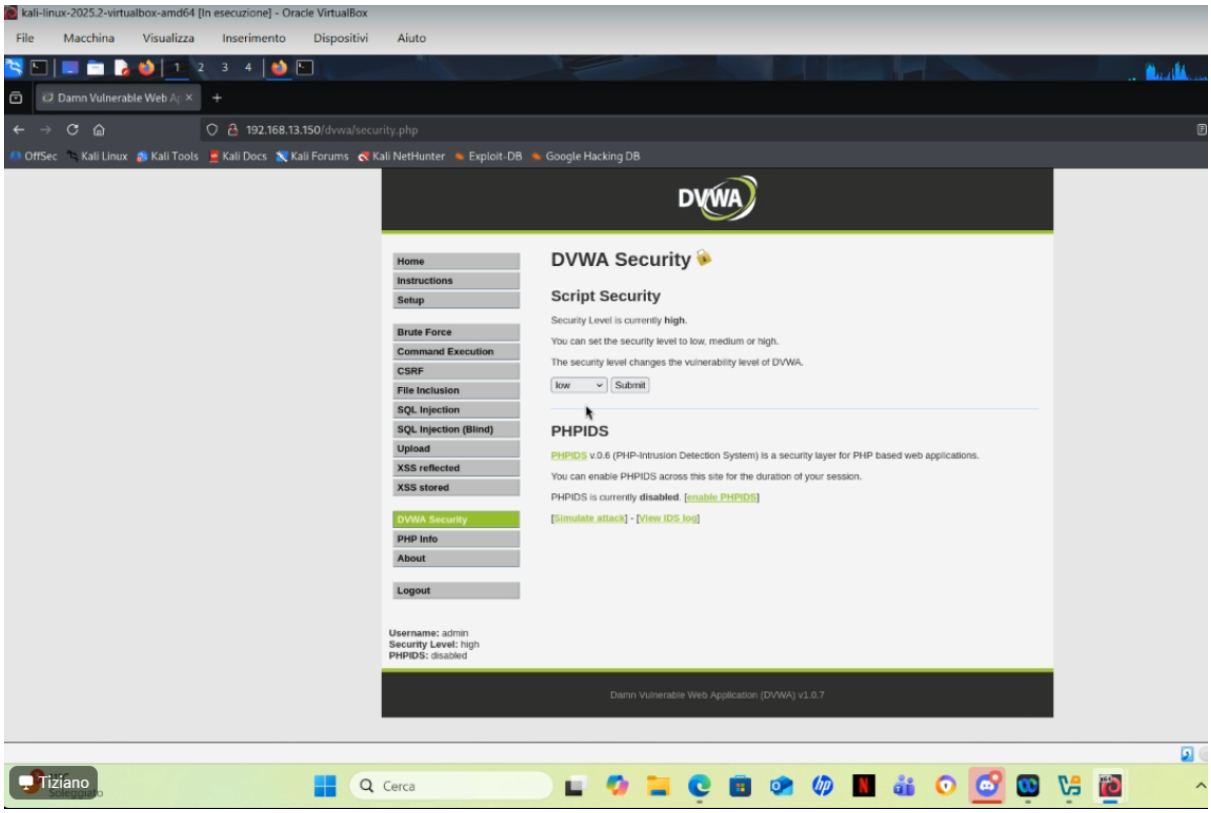
6. Ora effettuiamo un test di connettività all'interno del nostro ambiente tramite l'invio di pacchetti ICMP (ping) tramite il comando ping 192.168.13.150:



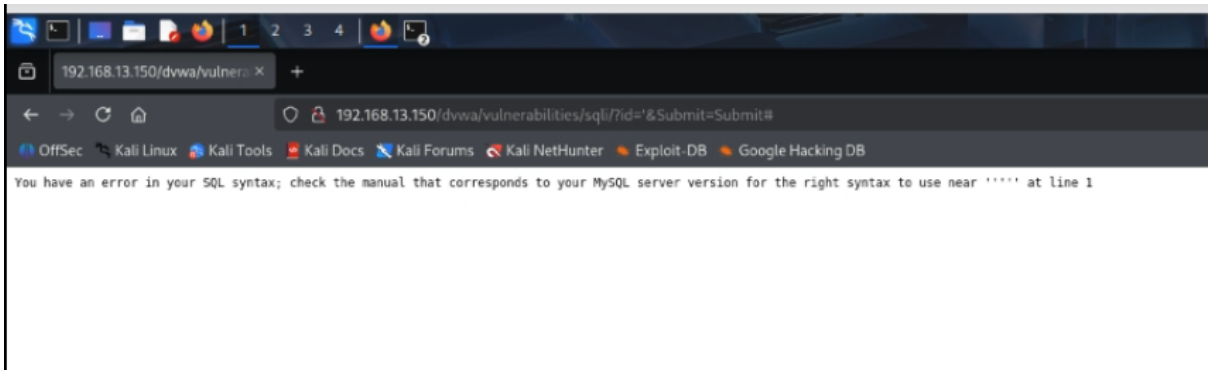
Exploit con SQLi (Securtiy LOW)

FASE 1 :

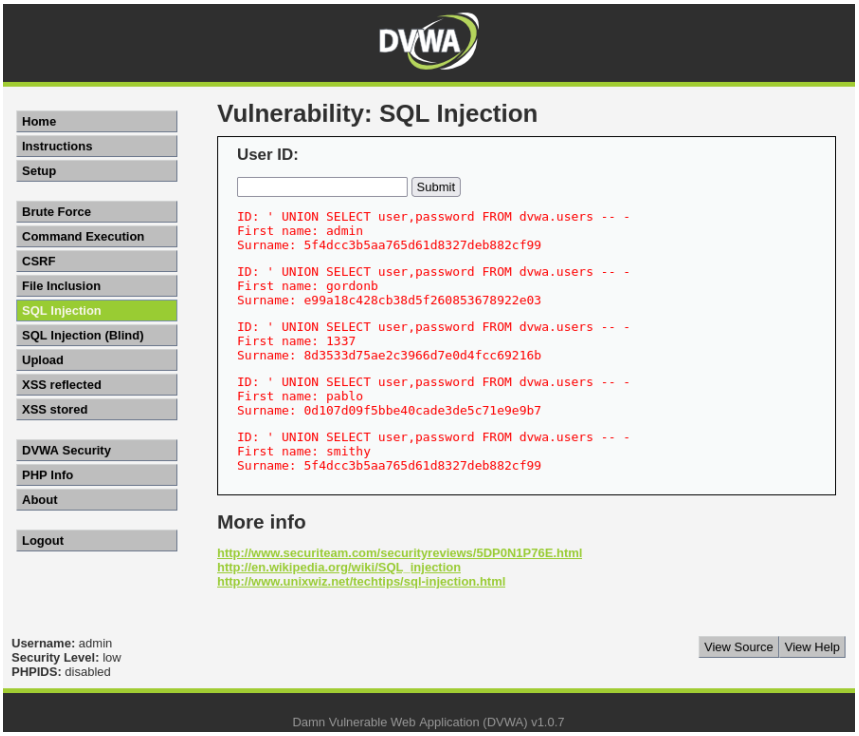
Avendo completato le operazioni di configurazione di rete possiamo ora procedere ad effettuare l'accesso alla pagina DVWA sulla VM kali andando ad impostare dapprima il DVWA Security a LOW:



1. Dal seguente screenshot si può evincere come ci siano errori di sintassi e si possa inserire codice malevolo:



2. Andando ad iniettare nel payload possiamo dimostrare la rottura della logica della query e l'estrazione di dati.
    - Payload iniettato : UNION SELECT user, password FROM dvwa.users -- -
    - Come funziona?
        1. L'utente chiude la query originale.
        2. L'istruzione UNION SELECT combina il risultato della query originale con i dati estratti dalla tabella dvwa.users.
        3. Troviamo le colonne user e password (memorizzate come hash) ignorando la normale logica di visualizzazione.

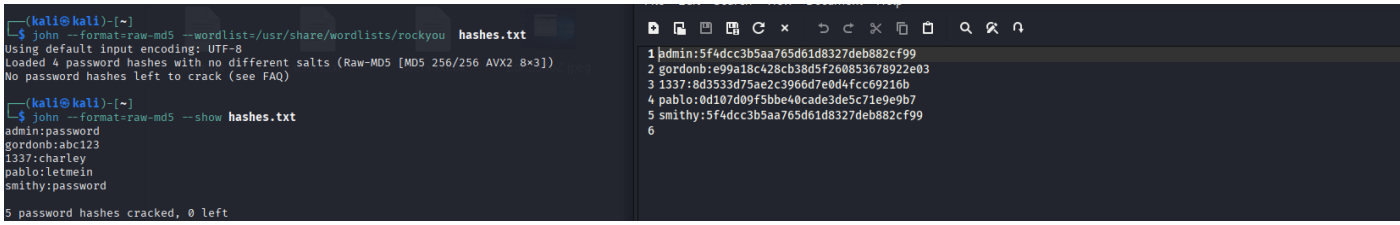


3. Compromissione dei dati → L'attacco ha dimostrato come la mancata separazione tra **dati** e **ordini SQL** consenta a un malintenzionato di

leggere informazioni dal database. L'accesso agli **hash delle password** è classificato come **grave compromissione**, in quanto questi hash possono essere utilizzati per attacchi offline mirati.

FASE 2 : Cracking delle Password

1. Abbiamo ora ricavato degli hash delle password dal database procediamo all'attacco offline per recuperare le password in chiaro.
2. Tramite l'utilizzo del tool John The Ripper →
    - Cos'è Jonh The Ripper? **John The Ripper (JTR)** è uno strumento open-source ampiamente riconosciuto nel campo della sicurezza informatica, specializzato nel recupero di password da hash crittografici.
    - Come funziona?
        - JTR prende gli hash target e, per ogni **password candidata** che genera (secondo il metodo d'attacco scelto), calcola l'hash utilizzando lo stesso algoritmo (ad esempio, MD5, come tipico in DVWA Low). Se l'hash generato coincide con l'hash target, la password in chiaro è stata trovata.
3. Andiamo quindi a utilizzare il comando john –format=raw-md5 –wordlist=/usr/share/wordlist/rockyoi hashes.txt e una volta terminato andiamo a controllare il risultato tramite un john –format=raw-md5 –show hashes.txt



Abbiamo dunque ricavato in chiaro la password di Pablo Picasso → password : "letmein".

Exploit con SQLi (Securtiy MEDIUM)

1. Cosa succede di diverso in MEDIUM?

Medium SQL Injection Source

```php
<?php

if (isset($_GET['Submit'])) {

    // Retrieve data

    $id = $_GET['id'];
    $id = mysql_real_escape_string($id);

    $getid = "SELECT first_name, last_name FROM users WHERE user_id = $id";

    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>' );

    $num = mysql_numrows($result);

    $i=0;

    while ($i < $num) {

        $first = mysql_result($result,$i,"first_name");
        $last = mysql_result($result,$i,"last_name");

        echo '<pre>';
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
        echo '</pre>';

        $i++;
    }
}
?>
```

Abbiamo $id = mysql_real_escape_string($id) →

- (mysql_real_escape_string) : Questa è la misura di sicurezza appena introdotta a livello MEDIUM.
- Lo scopo della funzione è quello di neutralizzare i caratteri speciali che potrebbero essere usati per un attacco SQLi(come l'apice singolo). Aggiunge un \ prima di ogni apice o carattere pericoloso.
-

Perchè questo codice è ancora vulnerabile?

Nonostante l'uso di mysql_real_escape_string(), questo codice è ancora considerato vulnerabile (livello Medium) per due motivi principali:

1. **Funzioni Obsolete**: Le funzioni PHP con il prefisso mysql_ sono deprecated e sono state rimosse nelle versioni recenti di PHP, poiché non supportano funzionalità di sicurezza moderne (come le Prepared Statements).
2. Bypass del Multi-byte Character Set: Può essere aggirata in ambienti datati sfruttando i caratteri multi-byte (come la sequenza – -), che "consumano" il backslash di protezione, liberando l'apice iniettato per l'attacco.

FASE 1

1. Andiamo dapprima a vedere le tabelle che possiamo sfruttare nel nostro attacco → tramite 1 UNION SELECT null, schema_name FROM information_schema.schemata → Come funziona?
    - 1 sta per l'ID che l'attaccante vuole che la query originale continui a cercare.
    - In pratica, 1 è solo il dato d'ingresso che precede la parte di iniezione UNION.

- UNION SELECT → La clausola UNION combina il set di risultati della query originale con il set di risultati di questa nuova query iniettata. Affinchè UNION funzioni , entrambe le query devono avere lo stesso numero di colonne e i tipi di dato delle colonne corrispondenti devono essere compatibili.

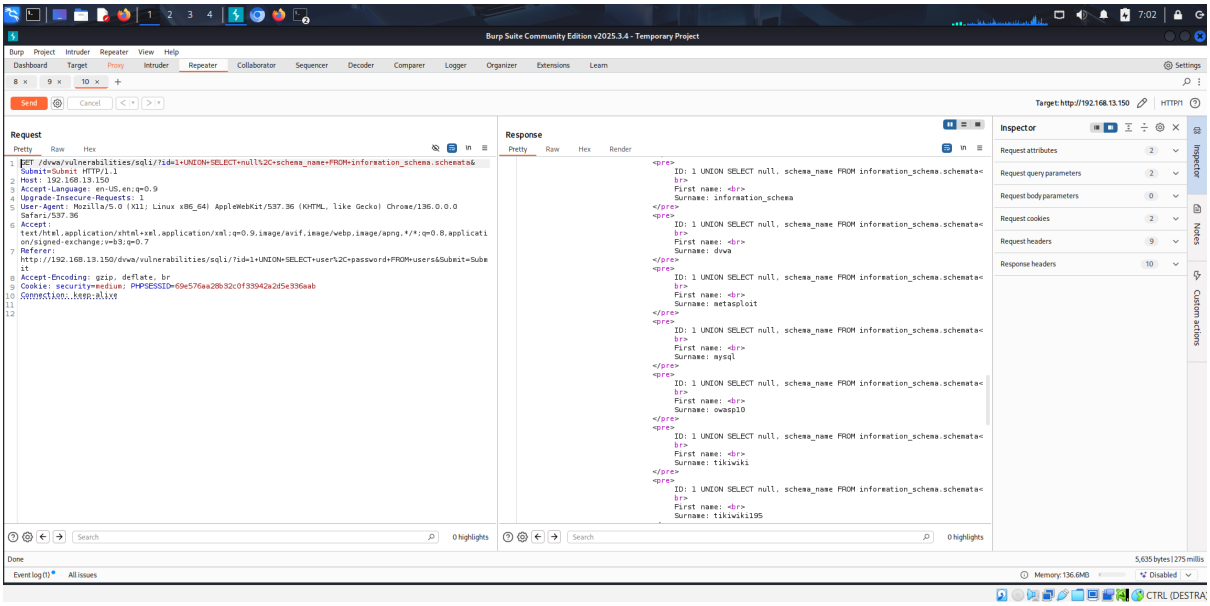
2. null, schema_name (Le Due Colonne)

   Questo è l'elenco delle due colonne che la query iniettata seleziona, per abbinare le due colonne della query originale.

   - **null (Colonna 1):** È usato come segnaposto per la prima colonna della query originale. Il NULL è compatibile con qualsiasi tipo di dato, risolvendo potenziali conflitti tra il tipo di dato della prima colonna della query originale (che potrebbe essere un ID numerico) e il tipo di dato nella query iniettata.
   - **schema_name (Colonna 2):** Questa è la parte cruciale. Specifica che il valore da estrarre e visualizzare è il nome di uno schema (database) dal server MySQL. **Questo è il dato che l'attaccante vuole rubare.**
3. FROM information_schema.schemata

- **information_schema:** È un database speciale presente in MySQL/MariaDB che contiene i **metadati** del database (informazioni sul database stesso). Gli utenti con i permessi corretti possono accedervi.
- **schemata:** È una tabella all'interno di information_schema che elenca tutti i database presenti sul server.

A cura di Pierantonio Miglietta, Iris Canole, Rebecca Talone, Francesco Miolli, Tiziano Bramonti, Andrea Sottile, Alessandro Ricci

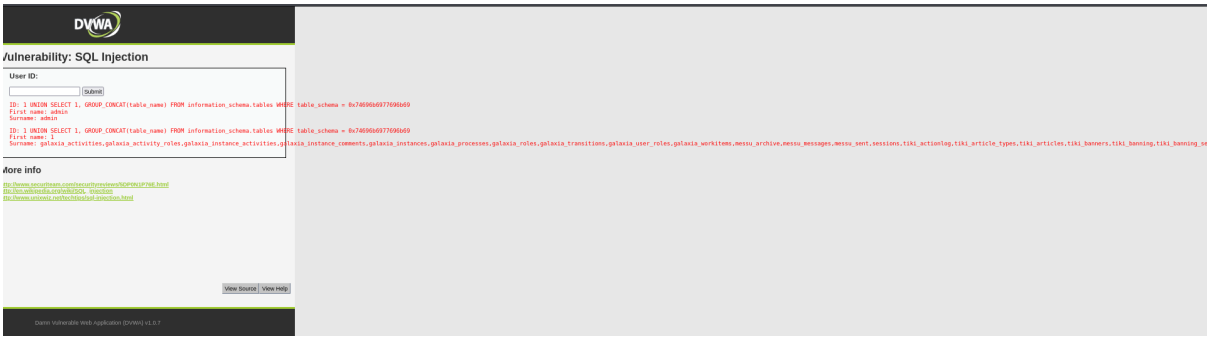


FASE 2

Molte applicazioni web hanno **filtri rudimentali** che bloccano stringhe contenenti caratteri speciali o nomi di tabelle comuni (users, password, admin, information_schema).

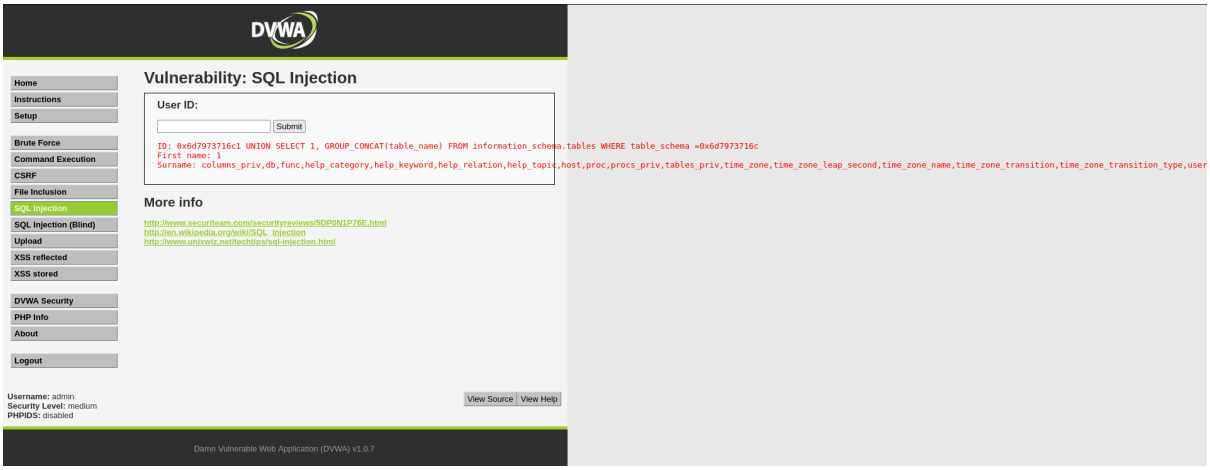
- **Codifica Nascosta:** Convertendo un nome (ad esempio, la stringa "users") nel suo equivalente esadecimale (ad esempio, 0x7573657273), l'attaccante camuffa l'attacco. La query iniettata manipola il database per **decodificare l'esadecimale a runtime**.

1. Tabelle database tikiwiki



2. Tabelle database owasp10 →

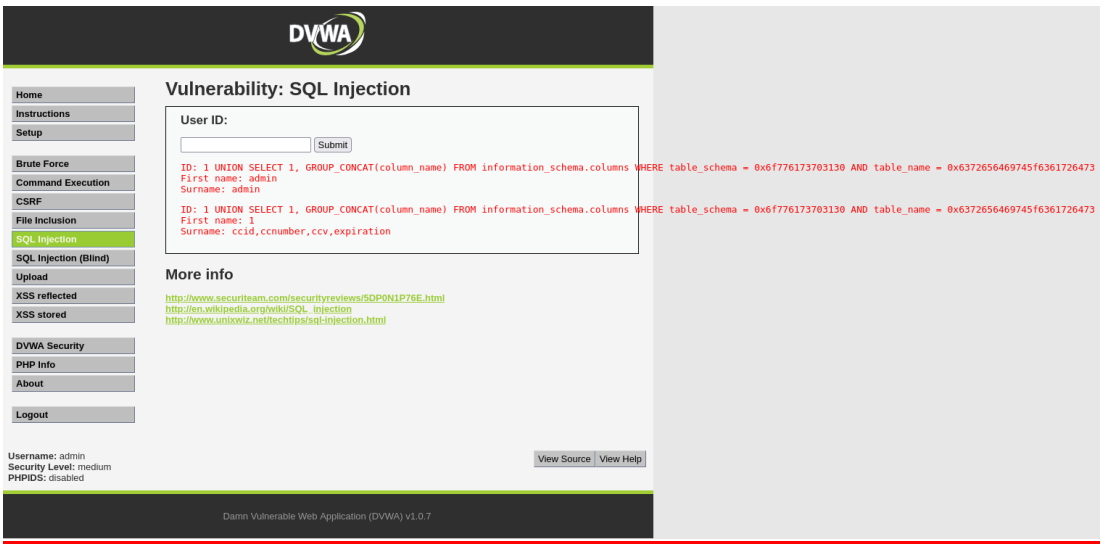
3. Tabelle database mysql →



Dalla nostra scansione abbiamo ricavato delle informazioni sensibili tramite il database Owasp10 contente informazioni sulle carte di credito presenti nel database →

- Query utilizzata : 1 UNION SELECT 1, GROUP_CONCAT(column_name) FROM information_schema.columns WHERE table_schema = 0x6f776173703130 AND table_name =

0x6372656469745f6361726473



- Query utilizzata : 1 UNION SELECT ccv, ccnumber FROM owasp10.credit_cards -- - Tramite cui abbiamo ricavato i numeri di carta e i codici segreti.