Cyber Security & Ethical Hacking Build Week – Progetto

**RELAZIONE SULL’ ESERCIZIO :**

SQL Injection su DVWA per Recuperare la Password di Pablo Picasso

**OBIETTIVO**

L’obiettivo dell’esercizio è sfruttare la vulnerabilità SQL injection (SQLi) presente nella Web Application Damn Vulnerable Web Application (DVWA), impostata al livello di difficoltà LOW, per recuperare la password associata all’utente Pablo Picasso. Successivamente, si procederà a decifrare la password qualora fosse memorizzata in forma criptata.

**REQUISITI LABORATORIO :**

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

**PASSAGGI OPERATIVI**

**1. Preparazione dell'Ambiente di Lavoro**

Configurare la rete virtuale assicurandosi che la macchina attaccante (Kali Linux) e quella vittima (Metasploitable) siano connesse sulla stessa rete (192.168.13.0/24).

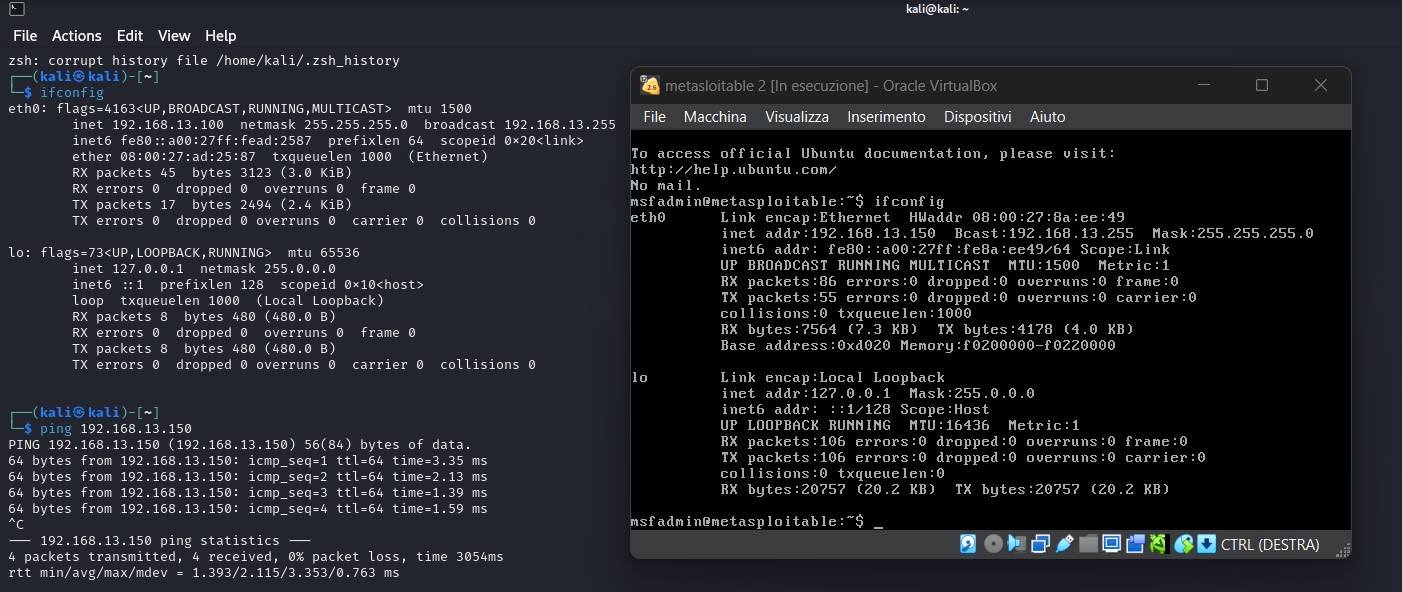
Accedere a DVWA dalla macchina attaccante tramite browser, usando l’URL: http://192.168.13.150/dvwa.

Effettuare il login su DVWA con le credenziali predefinite:

Username: admin

Password: password.

Impostare il livello di difficoltà su LOW.



**2. Identificazione della Vulnerabilità**

La fase di identificazione della vulnerabilità è essenziale per capire se un'applicazione web è suscettibile ad un attacco SQL injection. Ecco come funziona nel dettaglio:

Navigare alla sezione SQL Injection di DVWA.

Nel campo di input della pagina vulnerabile (ad esempio, un form per cercare utenti), inseriamo una stringa progettata per manipolare la query SQL sottostante. Un esempio classico è:

1' OR '1'='1

Dopo aver inviato il valore nel campo vulnerabile:

* Se l'applicazione restituisce più informazioni del previsto (ad esempio, tutti i dati degli utenti invece di un singolo utente), è confermato che il campo è vulnerabile a SQL injection.
* Se la query non viene elaborata correttamente (ad esempio, mostra un errore SQL), il sistema è vulnerabile ma ha un comportamento diverso, che può comunque essere sfruttato.

La fase di identificazione della vulnerabilità ci permette di:

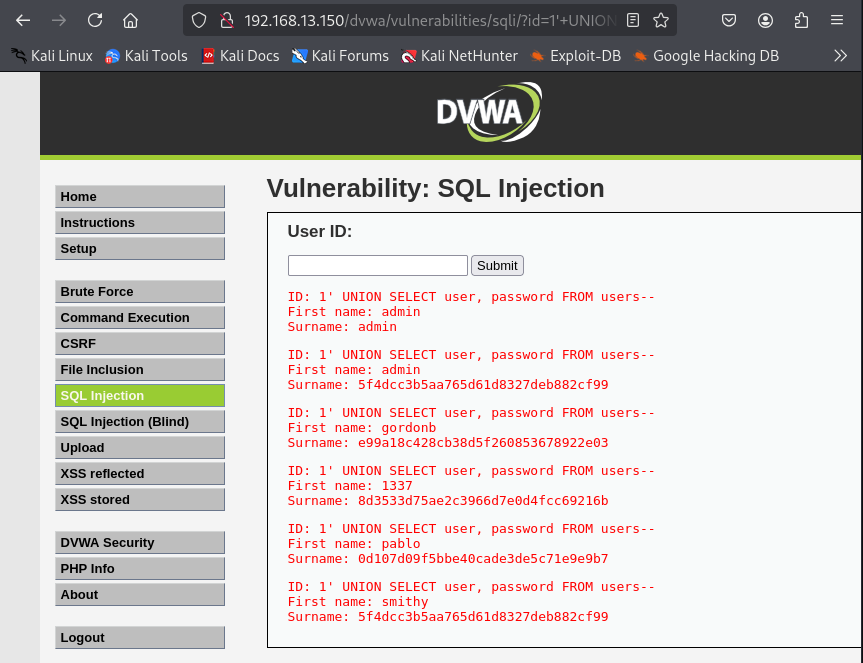
1. Confermare che il campo è suscettibile a SQL injection.
2. Comprendere la struttura della query SQL sottostante, essenziale per costruire iniezioni più complesse.
3. **Esecuzione dell'Attacco SQL Injection**

Questo passaggio consiste nell'uso di un'iniezione SQL più avanzata per estrarre dati sensibili dal database, come username e password degli utenti, incluso quello di Pablo Picasso. Analizziamolo nel dettaglio.

' UNION SELECT null, password FROM users--

E’ un esempio di attacco SQL injection progettato per estrarre informazioni sensibili dal database.

Analizzare la risposta della web application per identificare i dati degli utenti registrati, incluso Pablo Picasso e la sua password criptata.



1. **Recupero della Password in Chiaro**

In questa fase, dopo aver ottenuto le password (o hash delle password) tramite SQL injection, dobbiamo decifrare gli hash per ottenere le password in chiaro. Questo processo si chiama hash cracking.

Cos'è un Hash?

Un hash è una rappresentazione criptata di una password. È generato da un algoritmo di hashing (es. MD5, SHA1, bcrypt) e serve per memorizzare le password in modo sicuro. Gli hash sono progettati per essere unidirezionali, cioè non possono essere "decrittati" direttamente, ma devono essere "craccati" confrontandoli con un dizionario di parole o provando tutte le possibili combinazioni (attacco brute force).

Visto che la password è hashata (es. in formato MD5 o SHA1), utilizzare uno strumento di cracking come John the Ripper .

Salviamo l’hash in un file di nome “pablo.txt”

Eseguiremo il cracking con il comando:

john --wordlist=/usr/share/wordlists/rockyou.txt hash.txt

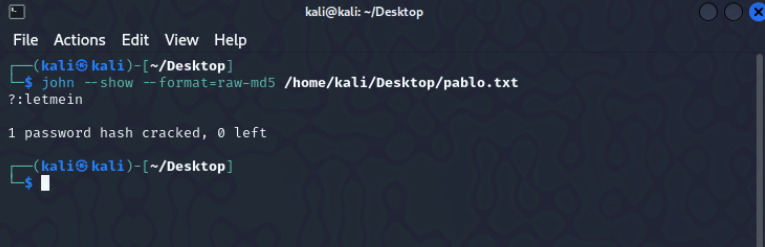
**Risultati**

Dati estratti tramite SQL Injection:

Username: Pablo Picasso

Password hash: 0d107d09f5bbe40cade3de5c71e9e9b7

Password in chiaro: letmein



**SQL INJECTION**

Un attacco SQL Injection è una tecnica utilizzata dagli attaccanti per manipolare le query SQL inviate a un database tramite un'applicazione web vulnerabile. Questo attacco sfrutta l'assenza di validazione nell'input dell'utente per inserire codice SQL malevolo, alterando il comportamento della query originale.

Ad esempio, in una query che cerca un utente per ID, un attaccante può iniettare un codice come `' OR '1'='1`, forzando la query a restituire tutti i record. Gli scopi possono includere il furto di dati, come credenziali o informazioni personali, la modifica di dati, la cancellazione di tabelle o persino l'esecuzione di comandi amministrativi.

Questo tipo di attacco è possibile quando l'applicazione non utilizza misure di sicurezza come prepared statements, parametri bindati o una corretta validazione e sanificazione dell'input. SQL Injection è una delle vulnerabilità più pericolose e frequenti in ambito web, classificata tra le prime nel report OWASP Top 10.

**JOHN THE RIPPER**

John the Ripper è uno strumento open-source progettato per il cracking delle password.

È ampiamente utilizzato dagli esperti di sicurezza informatica per testare la robustezza delle password in ambienti controllati. Funziona confrontando hash di password con un dizionario di parole (wordlist) o generando combinazioni tramite attacchi brute-force.

Supporta molti algoritmi di hashing, tra cui MD5, SHA1, bcrypt e altri, ed è compatibile con diversi sistemi operativi come Linux, macOS e Windows. John è personalizzabile e può essere ottimizzato per utilizzare la GPU o configurato per attacchi avanzati, come regole personalizzate per generare varianti di password.

Un tipico utilizzo consiste nel fornire un file contenente gli hash delle password e una wordlist (es. rockyou.txt ). John analizza ogni parola, la trasforma in hash e la confronta con quelli forniti, rivelando la password in chiaro se trova una corrispondenza.

John è uno strumento essenziale per l’ethical hacking, ma il suo uso improprio può avere implicazioni legali ed etiche.

**CONSIDERAZIONI**

L'esercizio di sfruttamento della vulnerabilità SQL Injection su DVWA ha evidenziato l'importanza di comprendere e testare le debolezze comuni nelle applicazioni web. Il livello di difficoltà *LOW* di DVWA ha permesso di eseguire l'attacco senza protezioni avanzate, rendendo evidente come l'assenza di validazione dell'input possa esporre il database a manipolazioni pericolose.

Abbiamo utilizzato un'iniezione SQL per estrarre username e password hashate dalla tabella *users*. Successivamente, il recupero delle password in chiaro tramite *John the Ripper* ha dimostrato la vulnerabilità di algoritmi deboli come MD5 e l'importanza di utilizzare tecniche di hashing sicure (es. bcrypt).