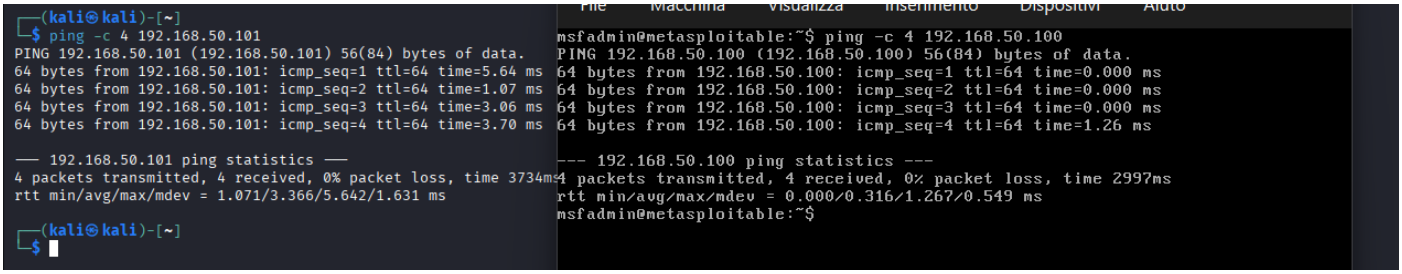
INDICE

INDICE……..1
INTRODUZIONE ED OBIETTIVO……2
METOLOGIA OPERATIVA……..3-6
CONCLUSIONE……7

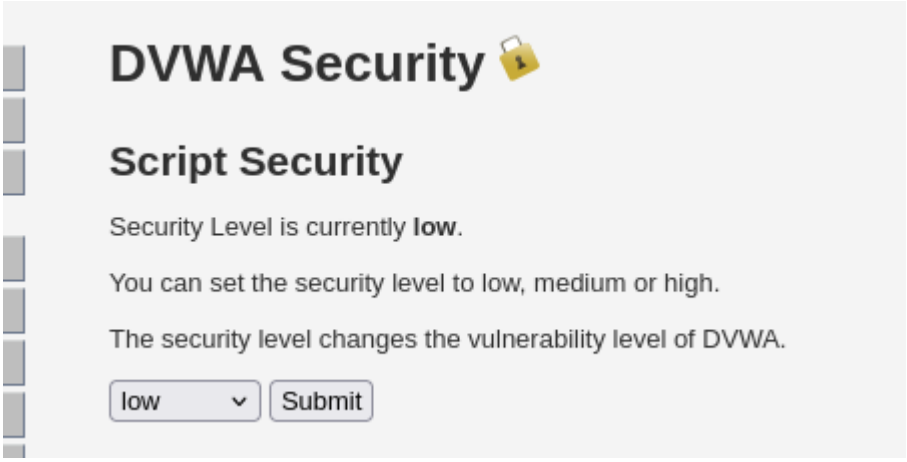
INTRODUZIONE ED OBIETTIVO

INTRODUZIONE: nel laboratorio si analizza una catena completa di compromissione delle credenziali partendo da una vulnerabilità applicativa e arrivando alla ricostruzione delle password in chiaro.
Attraverso l'uso controllato di una SQL Injection viene forzata l'applicazione a restituire i dati sensibili memorizzati nel database, evidenziando come la protezione basata su hashing MD5 non sia sufficiente a garantire la sicurezza delle credenziali quando il backend è esposto.

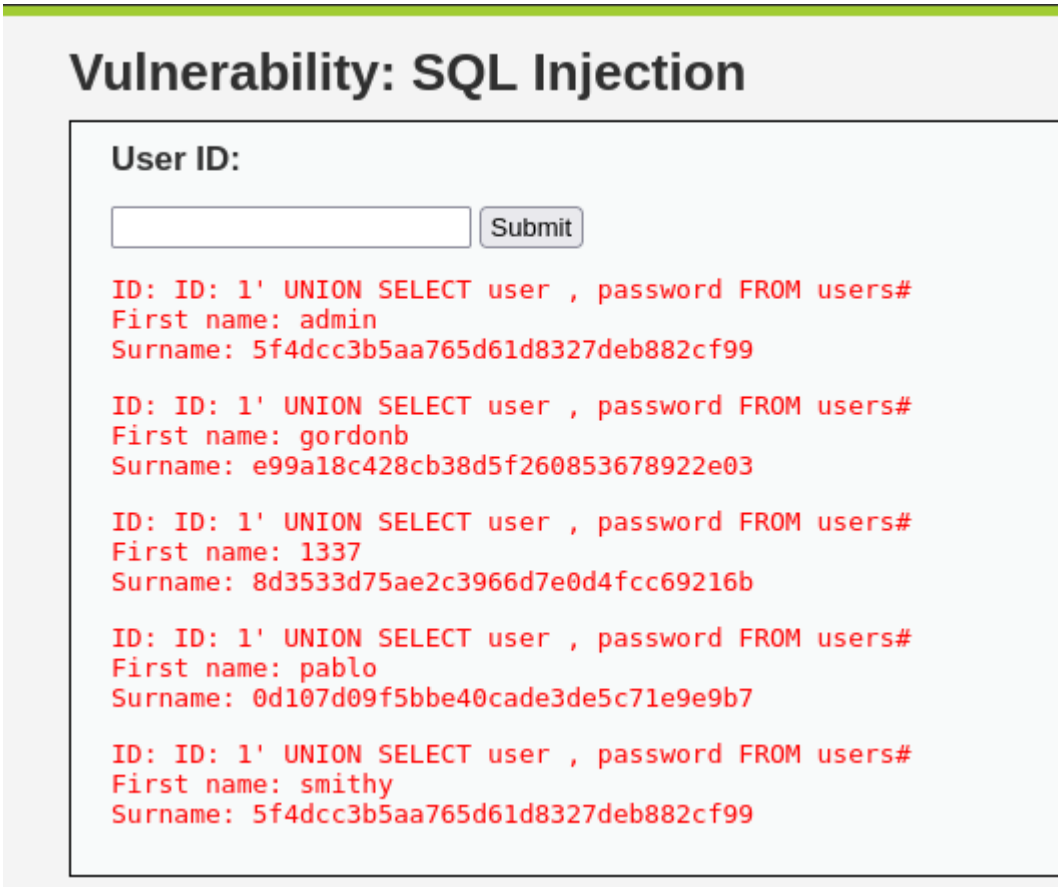
OBIETTIVO: ottenere gli hash delle password dal database tramite SQL Injection e recuperarne le corrispondenti password in chiaro mediante una sessione di cracking controllata.
Il risultato atteso è la ricostruzione completa di tutte le credenziali associate agli utenti estratti.



Prima di avviare le attività operative dell'esercizio è stata effettuata una fase preliminare di verifica dell'ambiente, finalizzata ad assicurare la corretta comunicazione tra i sistemi coinvolti.
In questa fase è stato controllato che le macchine fossero in grado di scambiare traffico di rete in modo stabile e senza perdite, condizione necessaria per garantire l'accesso all'applicazione vulnerabile e l'esecuzione delle successive fasi di attacco.
L'esito dei test ha confermato la piena raggiungibilità reciproca dei sistemi, escludendo la presenza di problemi infrastrutturali o di connettività che avrebbero potuto compromettere l'affidabilità delle attività successive.



Prima di procedere con lo sfruttamento della vulnerabilità SQL Injection, è stata verificata la configurazione di sicurezza dell'applicazione.
Il livello di protezione risulta impostato su *low*, condizione che abilita un comportamento intenzionalmente vulnerabile del codice applicativo, consentendo l'esecuzione di query non filtrate e prive di meccanismi di sanitizzazione degli input.
Questa configurazione è necessaria per permettere la corretta esecuzione dell'esercizio, in quanto consente di osservare in modo diretto l'impatto della SQL Injection sull'accesso ai dati memorizzati nel database.



Una volta impostato il livello di sicurezza dell'applicazione su una configurazione vulnerabile, è stata avviata l'attività di sfruttamento della SQL Injection.

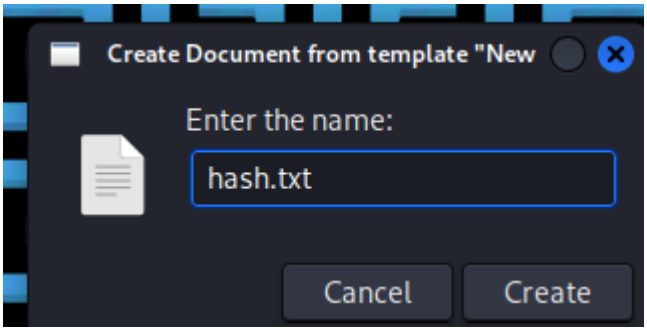
Attraverso l'inserimento di una query opportunamente costruita nel campo di input dell'applicazione, è stato possibile forzare l'esecuzione di una interrogazione aggiuntiva sul database, ottenendo l'accesso diretto alla tabella contenente le credenziali degli utenti.

L'output restituito mostra per ciascun record il nome dell'utente e il valore associato alla password, i dati recuperati non risultano in formato leggibile, ma sono rappresentati da stringhe esadecimali di lunghezza fissa, caratteristiche di un digest MD5.

Questo indica che le password non sono memorizzate in chiaro, ma vengono salvate sotto forma di hash crittografici.

L'analisi dei risultati evidenzia la presenza di cinque record distinti, di cui il primo e l'ultimo condividono lo stesso valore di hash, suggerendo l'utilizzo della stessa password da parte di due utenti diversi.

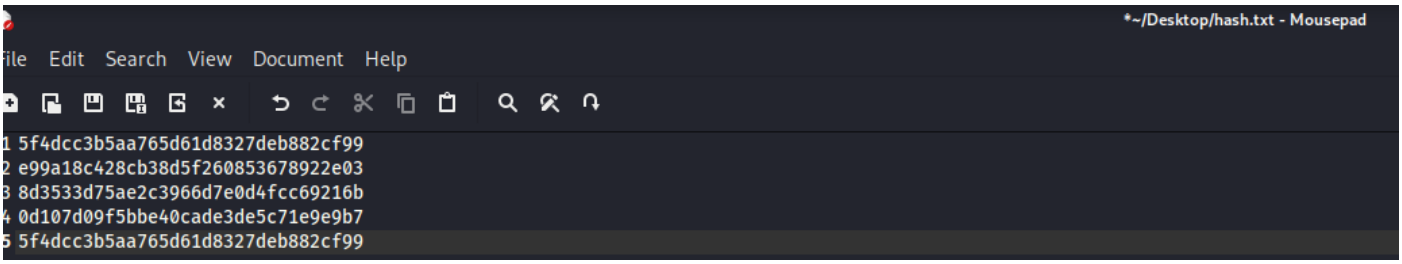
Gli hash estratti costituiscono l'insieme di dati che viene utilizzato nella fase successiva per il processo di cracking delle password.



Dopo l'estrazione delle credenziali dal database, i valori hash ottenuti vengono organizzati in un file dedicato, necessario per l'elaborazione da parte dello strumento di cracking.

In questa fase viene creato un file di testo denominato *hash.txt*, che ha la funzione di contenere esclusivamente gli hash delle password estratti in precedenza.

Questo file rappresenta il punto di ingresso dei dati nel processo di cracking e consente allo strumento di analisi di trattare ogni valore come un obiettivo indipendente da sottoporre a verifica.



All'interno del file *hash.txt* vengono inseriti gli hash MD5 precedentemente estratti dal database.

Ogni valore è riportato su una riga separata, senza informazioni aggiuntive, in modo da consentire allo strumento di cracking di interpretarli come obiettivi indipendenti.

La presenza di due hash identici nel file riflette la corrispondenza osservata in fase di estrazione, indicando che più utenti condividono la stessa password e la corretta formattazione di questo file è essenziale per garantire l'accuratezza della fase di elaborazione successiva.

```
┌──(kali㉿kali)-[~]
└─$ john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt /home/kali/Desktop/hash.txt
Using default input encoding: UTF-8
Loaded 4 password hashes with no different salts (Raw-MD5 [MD5 256/256 AVX2 8×3])
Warning: no OpenMP support for this hash type, consider --fork=2
Press 'q' or Ctrl-C to abort, almost any other key for status
password         (?)
abc123           (?)
letmein          (?)
charley          (?)
4g 0:00:00:00 DONE (2026-01-10 07:00) 200.0g/s 153600p/s 153600c/s 230400C/s my3kids..dangerous
Warning: passwords printed above might not be all those cracked
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

Una volta predisposto il file contenente gli hash, viene avviata la fase di cracking tramite uno strumento specializzato in analisi di password.

Il processo utilizza un dizionario di password comuni come base per generare candidati, che vengono trasformati in hash MD5 e confrontati con quelli presenti nel file *hash.txt*.

Durante l'esecuzione, lo strumento individua progressivamente le corrispondenze tra gli hash e le relative password in chiaro, producendo un output che conferma il successo dell'operazione.

I risultati intermedi mostrano l'individuazione delle password associate agli hash presenti, dimostrando che i digest memorizzati nel database possono essere risolti rapidamente quando vengono utilizzate funzioni di hashing deboli e prive di meccanismi di protezione aggiuntivi.

```
┌──(kali㉿kali)-[~]
└─$ john --show --format=Raw-MD5 /home/kali/Desktop/hash.txt
?:password
?:abc123
?:charley
?:letmein
?:password

5 password hashes cracked, 0 left
```

Al termine della fase di cracking viene eseguita una procedura di verifica che consente di visualizzare in forma strutturata le password associate agli hash presenti nel file *hash.txt*. L'output mostra per ciascun digest MD5 la relativa password in chiaro, confermando che tutti i valori estratti dal database sono stati risolti con successo.

Il risultato evidenzia che l'insieme completo delle credenziali è stato recuperato, senza alcun hash residuo non risolto.

La presenza di due password identiche corrisponde agli hash duplicati osservati nella fase di estrazione, dimostrando la coerenza dei dati lungo l'intero processo di analisi.

CONCLUSIONE

L’esercizio ha dimostrato come una vulnerabilità di tipo SQL Injection consenta di compromettere direttamente la riservatezza delle credenziali memorizzate in un sistema. L’accesso non autorizzato al database ha permesso l’estrazione degli hash delle password, evidenziando come la semplice protezione tramite MD5 non sia sufficiente a garantire un adeguato livello di sicurezza quando l’applicazione è esposta a input non filtrati.

La successiva fase di cracking ha confermato che gli hash ottenuti possono essere ricondotti rapidamente alle rispettive password in chiaro attraverso l’uso di dizionari di password comuni, rendendo evidente la fragilità di questo schema di memorizzazione.

Il recupero completo di tutte le credenziali conclude l’analisi, mostrando in modo concreto l’impatto che una combinazione di vulnerabilità applicative e meccanismi di hashing deboli può avere sulla sicurezza complessiva di un sistema.