| Photo displaying partial image of two pie charts on a canvas-textured page |
| --- |
| **Incident Response Report – Capstone Project**  ***Response to the Data Breach Scenario at Premium Light House Company*** |
| | **Titobiloluwa ILORI** | 9/11/23 | Cohort June 26, Lighthouse Labs | | --- | --- | --- | |

**Table of Content**

|  | **Sections** | | | **Page No.** | |
| --- | --- | --- | --- | --- | --- |
| **1** | **Executive Summary** | | | | **2** |
| **2** | **Incident Timeline** | | | | **3** |
| **3** | **Technical Analysis** | | | | **6** |
| **3.1** | | **Attack Origin and Impact** | | | **6** |
| **3.1.1** | | | **Attack Origin** | | **6** |
| **3.1.2** | | | **Attack Impact** | | **7** |
| **3.2** | | **Detailed Analysis of How Systems Were Accessed** | | | **8** |
| **3.3** | | **Outline of Weaknesses That Allowed for This Incident to Occur** | | | **18** |
| **4** | **Incident Response Playbook** | | | | **21** |
| **4.1** | | **Steps to Contain and Remediate the Incident** | | | **23** |
| **5** | **Post-Incident Recommendations** | | | | **25** |
| **5.1** | | **How Should the Company Protect Itself Against Such Attacks in The Future** | | | **25** |
| **5.2** | | **Recommended Potential Adjustments to Security Policy** | | | **26** |
|  | **Appendix A: References** | | | | **29** |

1. **Executive Summary**

In response to a ransomware attack on PHL, we have conducted a comprehensive incident analysis and developed an incident response playbook tailored to the unique circumstances of the breach. The attack targeted a public-facing web server and subsequently compromised systems within the Production VLAN, including a database and file server. Our technical analysis identified multiple weaknesses that allowed the attack to occur, including an initial web application vulnerability.

To address this incident effectively, we have structured the incident response playbook into key phases: Preparation, Detection, Analysis, Containment, Eradication, Recovery, Coordination, and Post-Incident Activity. Specific workflows and procedures have been customized to ensure an efficient and coordinated response.

We recommend the adoption of NIST 800-61 guidelines1 as a foundation for the incident response playbook, with an emphasis on communication, documentation, and continuous improvement. Following the playbook's guidance will help PHL mitigate the impact of future incidents and strengthen its overall security posture. Additionally, we advise updating security policies, recommend robust security following NIST 800-532 controls and enhance staff training to bolster resilience against ransomware attacks.

1. **Incident Timeline**

In this security incident at Premium House Light (PHL) Company, the attacker targeted vulnerabilities in publicly accessible PHL applications to initiate an unauthorized breach. The attacker employed a brute force strategy, attempting to access various URLs using a wordlist approach. They then utilized a web shell with a carefully constructed Python one-liner in a POST request to establish a reverse shell for remote command execution. To avoid detection, the payload was URL-encoded and disguised to appear legitimate.

After gaining execution access to the web server as "***www-data***," the attacker proceeded to explore the system. They employed the Nmap tool for network scanning, identifying open ports and services on the network, including SSH and Telnet services on different hosts.

Subsequently, the attacker attempted unauthorized Telnet access to a remote Ubuntu system, repeatedly trying default username/password combinations. Privilege escalation occurred when the attacker executed MySQL commands as the "root" user via the **sudo** command without needing a password. This granted them elevated privileges within the MySQL database, enabling them to access, copy, and transfer the "***phl***" database contents to a "***phl.db***" file using the SCP protocol to a remote server.

The attacker concluded the intrusion by exiting the Production subnet.

The attacker moved from the public-facing webserver hosting premiumhouselight.com and performed a lateral movement to the database with escalated privilege exfiltrated sensitive data about customers and securely copied to a remote server. The attack lasted for approximately 4 minutes and 32 seconds. The diagram below (Fig 1) gives a high-level summary of the incident timeline.

A screenshot of a computer

Description automatically generated

*Fig 2.1: Diagram showing the incident timeline and summary of how the attack happened.*

Below is a table that shows the timeline and location of the attack; and provides the stages and steps of the attack using the MITRE ATT&CK Framework.

| **Timeline UTC+1 and Location** | **Stage of Attack** | **Steps of the attack** | **ATT&CK** |
| --- | --- | --- | --- |
| **2:58:12.32 AM – 2:58:12.85 AM**  **(Web server)** | **Initial Access** | The attacker attempts to exploit vulnerabilities in applications accessible from the public internet to gain initial access to a system. | **T1190**: Exploit Public Facing Application3 |
| **2:58:22.15 AM – 2:58:55.90 AM (Web server)** | **Reconnaissance** | The attacker then exploited the vulnerability known with web server running with Apache 2.4.4116 and attempted to access various URLs (a wordlist, which is essentially a list of common or known directory or file names) on the PHL web server (e.g., "/uploads/shell.php") using a brute force or wordlist-based approach. | **T1595.003**: Active Scanning: Wordlist Scanning4 |
| **2:59:04.17 AM**  **(Web server)** | **Execution, Defense Evasion, C&C and Persistence** | The attacker attempts to execute arbitrary commands on the target system using a web shell with a POST request that contains a Python one-liner that establishes a reverse shell to enable remote command execution. The payload in the POST request has legitimate-looking and is also URL-encoded to obfuscate the command and make it harder to detect and establish a web shell on the server. | **T1059**: Command and Scripting Interpreter5  **T1036**: Masquerading6  **T1132**: Data Encoding7  **T1505.003**: Web Shell8 |
| **2:59:04.29 AM – 2:59:15.96 AM**  **(Production VLAN: Web server and Database server)** | **Execution** | The attacker (Port 4444 = Metasploit tool) first gained execution access to the web server, identified themselves as "www-data," spawned an interactive shell, listed files in the current directory to discover "shell.php," and explored the system for further actions. | **T1059**: Command and Scripting Interpreter5 |
| **Discovery** | The attacker used the nmap tool installed to perform a network scan of IP addresses in the 10.10.1.0/24 range to identify open ports and services on the network, discovering SSH and Telnet services on different hosts. | **T1046**: Network Service Discovery9 |
| **2:59:47.84 AM – 3:00:19.20 AM**  **(Database server)** | **Credential Access** | The attacker initiated a Telnet connection to the IP address 10.10.1.3 and attempted to gain access to the remote system (Ubuntu 20.04.3 LTS) by repeatedly trying different default usernames and passwords combinations. | **T1110**: Brute Force11 |
| **3:00:58.20 AM**  **(Database server)** | **Privilege Escalation** | The attacker using user "phl" is escalating their privileges by executing the sudo command to run MySQL commands as the "root" user without entering a password. This allowed them to perform administrative tasks within the MySQL database with elevated privileges. | **T1548**: Abuse Elevation Control Mechanism12 |
| **3:00:58.20 AM - 3:01:34.24 AM**  **(Database server)** | **Discovery** | The attacker listed the databases and tables to gather information about the database schema and structure. | **T1010**: Application Window Discovery13 |
| **3:01:45.58 AM - 3:02:44.74 AM (Database server and attacker’s remote server)** | **Exfiltration** | Attacker accessed the MySQL database with elevated privileges, dumped/copied the contents of the "phl" database into a "phl.db" file, used the SCP protocol to securely transfer the "phl.db" file to a remote server and exited the Production subnet. | **T1048**: Exfiltration Over Alternative Protocol14 |

*Table 1: Incident timeline with stages and steps of the attack using the MITRE ATT&CK Framework*

**3. Technical Analysis**

**3.1 Attack origin and impact**

**3.1.1 Attack origin**

While analyzing this incident, various artifacts from the web server and database server were utilized. These artifacts included an email from the threat actor, in which they demanded a ransom of 10 BTC to prevent the exposure of sensitive customer information on the internet (see Fig 3.1 below). Additionally, we examined access logs, network packet capture files, the network topology, and the database files of Premium House Lights Company.

A screenshot of a computer

Description automatically generated

*Fig 3.1: Threat actor’s extortion email*

During the review of these logs and packet capture files, the attacker's IP address was identified as **138.68.92.163** (highlighted in green). This determination was made based on the multiple attempts made by this IP address within a short timeframe, specifically within 43 seconds (as shown in Figure 3.2a). Notably, the attacker's actions involved persistent efforts to gain access and scanning for vulnerabilities on the web server.

Furthermore, it's important to note the presence of the web server's public IP address, highlighted in blue.

A screenshot of a computer

Description automatically generated

*Fig 3.2a: Web Server’s access log showing attacker’s IP address and Web public IP address.*

**3.1.2 Attack Impact**

The attack's impact on Premium House Lights Company can be significant. The attacker gained unauthorized access to the web server, compromising the confidentiality, integrity, and availability of sensitive data15. Their actions included exfiltrating customer data, as indicated by the threat actor's ransom demand to prevent data leakage.

The potential impact areas include:

**Data Breach:** The attacker's exfiltration of customer sensitive information poses a substantial risk of data exposure. Information, including personal and sensitive data, may be further compromised.

**Reputation Damage**: In the event of a successful data leakage, the company's reputation could be severely tarnished. Customers may lose trust in the organization's ability to protect their data.

**Financial Loss**: Addressing the breach, notifying affected customers, and implementing security measures can result in substantial financial costs.

**Operational Disruption**: The attacker's actions, including potential compromise of the web server, could lead to operational disruptions and downtime, affecting business continuity.

**Legal and Regulatory Consequences**: Non-compliance with data protection laws and regulations could lead to legal actions, fines, or penalties.

**Network and System Vulnerabilities**: The attacker's scanning for vulnerabilities may have exposed weaknesses in the network and system architecture, requiring immediate remediation.

Overall, the attack has the potential for severe consequences, necessitating a robust incident response plan, data breach notification, and efforts to mitigate the impact on both the organization and its customers.

**3.2 Detailed Analysis of How Systems Were Accessed**

**Initial Access**

The attacker with IP address 138.68.92.163 attempted unauthorized access to a web server (IP: 134.122.33.221) by executing SYN requests on multiple ports. The attacker's reconnaissance efforts led to a successful initial access, as Port 80 responded with a SYN, ACK packet. This breach signifies potential security risks. The attacker's activities included port scanning, which preceded further attacks. The technique used is to Exploit Public Facing Applications. See Fig 3.2b below.

A screenshot of a computer

Description automatically generated

Fig 3.2b:*Wireshark packet filtered to show attacker’s SYN requests and Web server port 80 SYN, ACK response.*

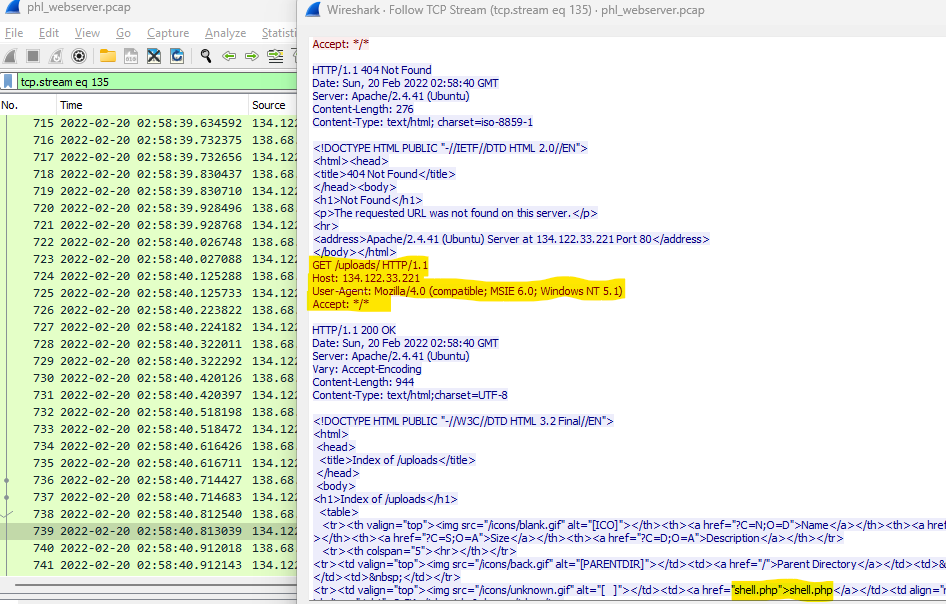
**Reconnaissance**

The attacker then took advantage of Apache 2.4.41 Ubuntu (on web server) known vulnerabilities as enumerated in CVE-2020-1927. *In Apache HTTP Server 2.4.0 to 2.4.41, redirects configured with* ***mod\_rewrite*** *that were intended to be self-referential might be fooled by* ***encoded newlines*** *and redirect instead to an unexpected URL within the request URL.*16 The NIST CVSS Base score metrics for this vulnerability is 6.5.

Attackers may use a wordlist, which is essentially a list of potential directories or file names, to scan for common or known files and directories on the web server and identify potential entry points and weaknesses in a target system during the initial access phase of an attack. In this scenario, the attackers are attempting to access various URLs on the server (e.g., "/upload.php") using a brute force or wordlist-based approach.

While reviewing the network packet showing HTTP requests and responses (Fig 3.3 below), it appears that an attempt was made to access various URLs on a web server (Public IP address: 134.122.33.221). The requests primarily use the HTTP GET method and target different paths on the server. However, most of the requests resulted in a "404 Not Found" response from the server.

There is a particular request that is notable (highlighted in yellow in Fig 3.3). This request is made to the /uploads/ path, and it returns a "200 OK" response, indicating that the /uploads/ directory on the server is accessible. Inside the /uploads/ directory, there are some files listed, including one named "shell.php."



*Fig 3.3: Wireshark packet filtered to show attacker’s GET requests and Web server response with 200 http code.*

This "shell.php" file in the /uploads/ directory is suspicious and could potentially be a web shell. *Web shells are malicious scripts that attackers upload to compromised web servers to gain unauthorized access and control over the server.10* They can be used to execute arbitrary commands, upload/download files, establish persistence, execute further malicious actions and perform other malicious activities.

**Execution, Defense Evasion, Command and Control, and Persistence (in the web server):**

At first, the attacker sent an HTTP curl request as shown in Fig 3.4 below. This HTTP request and response exchange between a client and a web server, like the previous one is a successful HTTP GET request to retrieve an index page listing the contents of the /uploads/ directory from the server. The server responds with a 200 OK status code and provides the HTML content for the index page in the response body. This request was made using the curl command-line tool with the specified ***user-agent***.

A screenshot of a computer

Description automatically generated

Fig 3.4: *Wireshark packet filtered to show attacker’s curl request and Web server response with 200 HTTP code.*

The response body containing an HTML document displays an "Index of /uploads" page, which lists the contents of the /uploads/ directory. The page includes a table with columns for file names, last modified dates, sizes, and descriptions. There are two entries listed in the directory:

* Parent Directory: This entry allows navigation to the parent directory.
* shell.php: This entry is a link to a file named shell.php with a last modified date of "2022-02-19 20:54 and a size of 2.5K.

Then, a packet showed an HTTP POST request and the server's response (see Fig 3.5 below). It appears to involve an attempt to **execute** a command via a web shell on a web server. The body of the POST request contains data in the form of URL-encoded parameters. Specifically, it includes a command to execute via Python. This command, when decoded, is an attempt to establish a reverse shell connection to the IP address 138.68.92.163 on port 4444 using Python's “socket” and “subprocess” libraries. It essentially aims to execute a reverse shell, allowing remote control of the server.

A screenshot of a computer

Description automatically generated

*Fig 3.5: Wireshark packet filtered to show attacker’s POST request and command execution and Web server response 200 HTTP code*

The attacker executed an arbitrary command on the target system using a web shell. The payload in the POST request contains a Python one-liner that establishes a reverse shell to enable remote command execution. It also used a legitimate-looking HTTP POST request with a content type of `application/x-www-form-urlencoded` and the user-agent string "curl/7.68.0" to make the malicious activity appear less suspicious.

Lastly, the attacker attempts to establish a web shell on the server by sending a POST request with a malicious command. The response from the server indicates that it provides an interface for executing further commands, potentially allowing persistence through remote access.

**Execution and Discovery (in the production VLAN):**

Subsequently, a sequence of commands was executed on the server by the attacker. Fig 3.6 below shows the first 6 sets of commands.

A screenshot of a computer

Description automatically generated

*Fig 3.6: Wireshark packet filtered to show attacker executing commands on the web server.*

1. **/bin/sh: 0: can't access tty; job control turned off:**

This message indicates that the shell (/bin/sh) cannot access the terminal and job control is turned off. It usually happens when you're in a non-interactive shell.

1. **$ whoami**:

The command returns "www-data," indicating that the user running these commands is www-data.

1. **$ python -c 'import pty; pty.spawn("/bin/bash")'**:

This command is used to spawn a new interactive bash shell from the Python process. It is commonly used to gain better control and interaction with the system.

1. **$ ls -l**:

This command lists the contents of the current directory (/var/www/html/uploads). It shows a file named "shell.php" with its permissions, owner, size, and modification date. This suggests that the attacker is inspecting the contents of the directory they are in.

1. **$ dpkg -l | grep nmap**:

This command lists installed packages and searches for the string "nmap." It reveals that the "nmap" package is installed on the system, along with its version.

1. **$ ifconfig:**

This command displays network interface information. It shows details for three interfaces: eth0, eth1, and lo. Notably, eth1 has the IP address 10.10.1.2, indicating that the system is on a local network.

In the next diagram below (Fig 3.7), this attacker continued to use the nmap tool, a common network scanning tool, to probe a range of IP addresses (10.10.1.0/24) on the local network. The remaining 3 sets of commands were used to discover the IPs and Ports available in the local network and connect to an open port (Telnet).

A screenshot of a computer

Description automatically generated

*Fig 3.7: Wireshark packet filtered to show attacker executing discovery commands on the webserver.*

1. **$ nmap 10.10.1.0/24 -sS:**

An attempt is made to run an Nmap scan on the IP range 10.10.1.0/24 using a SYN scan (-sS). However, it fails with a message indicating that root privileges are required.

1. **$ nmap 10.10.1.0/24**:

Another Nmap scan is performed on the same IP range without specifying a scan type. This scan successfully identifies two hosts (10.10.1.2 and 10.10.1.3 private IPs) on the local network. It lists open ports and services on these hosts, including SSH (port 22), HTTP (port 80), and Telnet (port 23).

1. **$ telnet 10.10.1.3**:

A telnet connection is established to IP address 10.10.1.3 (the private IP Address of the database server). It successfully connects to a remote system running Ubuntu 20.04.3 LTS which is the database server.

**Credential Access**

At this stage, the traffic packet seen in the Database Server packet capture shows that the attacker attempted to gain access to the remote system (Database server running Ubuntu 20.04.3 LTS) by repeatedly trying different usernames and passwords. Let's break down the analysis as shown in Fig 3.8 below.

A screenshot of a computer

Description automatically generated

*Fig 3.8: Wireshark packet filtered to show attacker attempting unauthorized access into the database server.*

1. **Authentication Attempts:**
   * The log starts with a series of login attempts (4 in total) where the attacker attempted to log in with various usernames and passwords.
   * The usernames used are "***admin***," "***administrator***," and "***phl***."
   * The passwords used are "***admin***," "***password***," and "***phl***."
   * In each case, the login attempt was unsuccessful, as indicated by the "Login incorrect" messages.
2. **Successful Login:**
   * At the fourth login attempt with username "***phl***" and password "***phl123***", there was a successful login.
   * The successful login results in access to the Ubuntu 20.04.3 LTS system.

Below is some information provided by the database server upon successful access.

1. **System Information:**
   * The log then proceeds to display system information about the Ubuntu server.
   * It indicates that the server is running Ubuntu 20.04.3 LTS with a specific Linux kernel version (5.4.0-97-generic) and architecture (x86\_64).
   * Additional information includes links to documentation, management, and support resources.
2. **System Status:**
   * The log provides a snapshot of the system's status as of the timestamp "Sat Feb 19 22:00:18 EST 2022."
   * This information includes system load, the percentage of disk usage on the root filesystem ("/"), IP addresses for network interfaces (eth0 and eth1), memory usage, swap usage, and the number of processes running.
3. **Available Updates:**
   * The log mentions that there are **14 updates** that can be applied immediately and suggests using the "apt list --upgradable" command to see the details of these updates.

**Privilege Escalation**

After gaining access to the target system, the attacker runs **netstat -atunp** to view active network connections and processes running on the system.

The attacker with user "***phl***" executed the command **sudo -l** to list the privileges granted to their user account. The output of the **sudo -l** command shows that the user "***phl***" has specific sudo privileges on the "***database***" system. These privileges include the ability to run the following commands as the "***root***" user without entering a password:

* **/usr/bin/mysql**
* **/usr/bin/mysqldump**

Finally, the attacker runs sudo mysql -u root -p and enters a password to access the MySQL server as the root user. After entering the password when prompted, the user (attacker) gained access to the MySQL monitor, where they can interact with the MySQL database server as the "***root***" user. See Fig 3.9 below for the evidence.

A screenshot of a computer

Description automatically generated

*Fig 3.9: Wireshark packet filtered to show attacker executing commands on the database server.*

**Discovery**

The attacker then executed the command **show databases;** to list the available databases on the MySQL server. The output shows a list of databases, including "***information\_schema***," "***mysql***," "***performance\_schema***," "***phl***," and "***sys***."

The user then executed the command **use** **mysql**; to switch to the "***mysql***" database. After switching to the "***mysql***" database, the user executed the command **show tables;** to list the tables within the "***mysql***" database (see Fig 3.10 below).

A screenshot of a computer

Description automatically generated

*Fig 3.10: Wireshark packet filtered to show attacker executing MySQL commands on the database server.*

The attacker continued to explore the MySQL database server until it used **mysql> use phl;** command, which instructed MySQL to switch to the "***phl***" database. It executed **mysql> show tables;** command to list the tables within the "***phl***" database. The output displays a single table named "***customers***" within the "***phl***" database. After identifying the "***customers***" table, the command ***mysql> SELECT \* FROM customers;*** was used to retrieve all rows and columns from the "***customers***" table.

The "***customers***" table appears to store information related to customers, such as their names, contact details, phone numbers, addresses, and financial data. The table contains several columns, including "customerNumber," "customerName," "customerId," "contactLastName," "contactFirstName," "phone," "addressLine1," "addressLine2," "city," "state," "postalCode," "country," and "amount\_spent." (See Fig 3.11 below)

A screenshot of a computer

Description automatically generated

Fig 3.11: *Wireshark packet filtered to show attacker identifying sensitive information on the database server.*

**Exfiltration**

The attacker then executed a command (see Fig 3.12) to perform a MySQL database dump into a file named "***phl.db***," specifying the username **-u root** and the database name ***phl***. Then execute the **file phl.db** command to identify the type of the "***phl.db***" file. The output indicates that it's a UTF-8 Unicode text file with very long lines.

A screenshot of a computer

Description automatically generated

Fig 3.12: Database shell text file and *Wireshark packet filtered to show the attacker dumping the data into a file and securely copying to a remote server.*

The user used the **SCP** command to securely copy (SCP) the "phl.db" file to a remote server with the IP address "178.62.228.28," placing it in the ***"/tmp***" directory on the remote server. They specified the remote username as "***fierce***" and were prompted to enter the password for that remote user.

After successfully transferring the file, the user executed **rm phl.db** to remove the local "***phl.db***" file. Finally, the user exited the session.

**Summary**

The attack vector used by the threat actor involved exploiting vulnerabilities in PHL's web server. The attacker initially gained access to the server by exploiting a known vulnerability in a web application. This allowed them to execute arbitrary code on the server with the privileges of the "www-data" user. The attacker then leveraged this initial access to perform reconnaissance, escalate privileges, and exfiltrate sensitive data.

**3.3. Outline of Weaknesses That Allowed for This Incident to Occur.**

1. **Inadequate Authentication Controls:** Weak or improperly configured authentication controls allow the attacker to gain unauthorized access to the server. Example: Lack of strong authentication mechanisms, such as multi-factor authentication (MFA), left the system vulnerable to password-based attacks.
2. **Weak Passwords**: Weak and easily guessable passwords, such as "admin," "password," and "phl123," were used, indicating poor password policies and enforcement. Example: Password complexity requirements and regular password changes were not enforced.
3. **Lack of Account Lockout Mechanism**: The absence of an account lockout mechanism made it easier for the attacker to conduct brute-force attacks without any restrictions. Example: After multiple failed login attempts, user accounts should have been temporarily locked.
4. **Insufficient or no Intrusion Detection**: Inadequate intrusion detection capabilities failed to identify and respond to unauthorized login attempts promptly, allowing the attacker to continue their activities unchecked. Example: Intrusion detection systems should have triggered alerts for multiple failed login attempts.
5. **Unpatched Software**: Pending updates and the need for a system restart indicated a failure to regularly update and patch the server's software, leaving it vulnerable to known security vulnerabilities. Example: Security patches for critical software components were not applied in a timely manner. In **Apache HTTP Server versions 2.4.0 to 2.4.41**, a security vulnerability exists within the mod\_rewrite module. Self-referential redirects may be manipulated by encoded newlines, causing unintended redirection to unexpected URLs within the original request URL. This poses potential security risks and warrants attention from users of these versions. It is advisable to apply updates or patches to mitigate this issue.16
6. **Ineffective Network Segmentation**: Inadequate network segmentation allowed the attacker to move laterally from the compromised web server to another server on the local network (10.10.1.3), increasing the attack surface. Example: Critical systems should have been isolated from less secure ones using proper network segmentation.
7. **Insufficient Monitoring**: The lack of immediate detection and response to the intrusion suggested inadequate monitoring or alerting systems to identify and respond to suspicious activities. Example: Real-time monitoring of system logs and network traffic was not robust.
8. **Default or Weak Credentials**: The attacker escalated privileges by accessing the MySQL database, possibly due to the use of default or weak credentials. Example: Default usernames and passwords for services should be changed, and strong credentials enforced.
9. **Inadequate Database Security**: The database containing customer information lacked strong access controls and encryption, exposing sensitive data to unauthorized access. Example: Access controls, encryption, and data masking should have been implemented to protect customer data.
10. **Lack of Logging and Auditing**: Inadequate logging and auditing practices hindered the ability to track and investigate security incidents effectively. Example: Detailed logs of user activities, authentication events, and system changes were not maintained.
11. **No Multi-Factor Authentication (MFA):** The absence of MFA mechanisms left authentication reliant solely on usernames and passwords, making it vulnerable to password-based attacks. Example: Implementing MFA would have added an additional layer of security.
12. **Failure to Disable Unused Services**: Services like Telnet, known for security vulnerabilities, were active and accessible, providing potential entry points for attackers. Example: Unused or insecure services should be disabled or replaced with more secure alternatives.
13. **Only Port 80 (HTTP) is Opened:** The fact that only Port 80 (HTTP) was open on the web server implies that other ports and services might not have been properly secured or were not adequately monitored. In a secure configuration, only necessary ports and services should be exposed to the public internet, and each should be carefully configured and monitored to prevent unauthorized access.
14. **Requests Library is Installed**: While the presence of the Requests library itself may not be a vulnerability, it highlights the importance of monitoring installed software and libraries for potential security risks. If the Requests library is not needed for the web application's functionality, it should be removed to reduce the attack surface and potential risks associated with its use.
15. **Absence of Access Controls:** The absence of access controls or firewall rules may have allowed the attacker's IP address to reach the server. Proper access controls should have been in place to restrict access to trusted sources only.
16. **Data Encryption:** Customer-sensitive data in the database should have been encrypted to protect it from unauthorized access even in the event of a breach.
    * 1. **Incident Response Playbook**

The flowchart below illustrates the recommended incident response playbook with a customized workflow.

A diagram of a diagram

Description automatically generated

A diagram of a system

Description automatically generated

*Fig 4.1: Recommended incident response playbook with customized workflow*

By customizing this incident response playbook to follow NIST Standard procedures1 to the specifics of the ransomware attack and the organization's environment, PHL can better prepare for, detect, analyze, contain, eradicate, recover from, coordinate during, and learn from incidents of this nature. The next section will elaborate more on steps to take, to contain and remediate the incident.

**4.1 Steps to Contain and Remediate the Incident.**

Here are steps to contain and remediate the ransomware incident based on the information provided in this chat and the ongoing chat:

**4.1.1 Containment Steps**

**Triage the Event**: Begin by confirming the incident's nature and scope. Review emails, logs, and indicators of compromise (IOCs) to understand the extent of the attack.

**Isolate Affected Systems**: Immediately disconnect compromised systems, including the web server, from the network to prevent further lateral movement and data exfiltration. Ensure the systems remain isolated until they are deemed safe.

**Disable Compromised Accounts**: Lock or disable the user accounts that were compromised to prevent unauthorized access.

**Identify and Block Command and Control (C2) Servers**: Determine the C2 servers used by the attacker and block network traffic to and from these servers at the perimeter firewall.

**Change All Passwords**: Force password resets for all users, emphasizing the creation of strong, unique passwords. Enable multi-factor authentication (MFA) for all accounts.

**4.1.2 Remediation Steps**

**Investigation and Analysis**: Conduct a detailed forensic analysis of affected systems to determine the initial attack vector, extent of data compromise, and the ransomware variant used.

**Data Recovery**: Restore data from secure backups, ensuring that the backups themselves are free of malware. Prioritize critical systems and data.

**Patch and Update Systems**: Apply security patches to all systems on the web server and database server to address vulnerabilities exploited by the attacker. Implement regular patch management practices.

**Security Policy Review**: Review and update security policies, including password policies, email security, and access controls, to prevent similar incidents in the future.

**User Training and Awareness**: Provide training to employees on recognizing phishing attempts and practising safe email and internet usage.

**Network Segmentation**: Implement network segmentation to isolate critical servers, such as databases, from public-facing systems like web servers.

**Incident Response Improvement**: Evaluate the incident response process and make necessary improvements, including refining the incident response plan, clarifying roles and responsibilities, and establishing communication protocols.

**Legal and Regulatory Compliance**: Ensure compliance with data breach notification laws and regulations by reporting the incident to appropriate authorities and affected parties.

**Lessons Learned**: Conduct a post-incident review to identify lessons learned and update the incident response playbook and security policies accordingly.

**Security Audits**: Perform security audits and penetration testing to identify and remediate any remaining vulnerabilities.

**Monitor for Reoccurrence**: Continuously monitor the network for signs of reoccurrence or new threats and promptly respond to any suspicious activities.

**4.1.3 Others**

**Coordination and Communication**

* Incident Response Team Activation
* Engage external cybersecurity experts and legal counsel if necessary
* Develop a clear communication plan for informing internal and external stakeholders
* Notification to Law Enforcement

**Post-Incident Activity**

* Review and update security policies, procedures, and incident response plans
* Conduct cybersecurity awareness training for employees
* Vendor and Third-Party Assessment

By following these containment and remediation steps, the organization can effectively respond to the ransomware incident, minimize damage, and strengthen its security posture to prevent future attacks. It is to be established that ***NO ransom should paid*** according to the bad actor’s request. This is not industry standard. This holistic approach to incident response encompasses preparation, detection, analysis, containment, eradication, recovery, coordination, and post-incident activities. It aims to minimize the impact of the ransomware incident, enhance cybersecurity defenses, and ensure that the organization is better equipped to handle similar threats in the future.

**5. Post-Incident Recommendations**

**5.1 How Should the Company Protect Itself Against Such Attacks in The Future**

Premium House Light (PHL) should enhance its security posture and protect itself against future attacks by implementing the following NIST 800-53 controls2 and associated examples:

* + 1. **Access Control (AC):**
* **Implement AC-2 - Account Management**: Enforce robust password policies requiring minimum length, complexity, and periodic changes. For example, passwords should include a mix of upper and lower-case characters and special symbols.
* **Implement AC-3 - Access Enforcement**: Deploy account lockout mechanisms to prevent multiple failed login attempts. After a specified number of unsuccessful tries, lock user accounts for a defined time, discouraging brute-force attacks.
* **Enforce AC-7 - Unsuccessful Logon Attempts:** Monitor and record unsuccessful login attempts. Configure automated alerts to promptly identify and respond to repeated failures, indicative of potential security threats.
  + 1. **System and Communications Protection (SC):**
* **Apply SC-7 - Boundary Protection**: Segment the network to isolate critical systems. Employ firewalls and access controls to restrict unauthorized access between network segments.
* **Enforce SC-8 - Transmission Confidentiality and Integrity**: Use encryption, such as TLS/SSL, to protect sensitive data during transmission.
  + 1. **Security Assessment and Authorization (CA):**
* **Adopt CA-2 - Security Assessments**: Regularly conduct vulnerability scanning and penetration testing to identify and remediate weaknesses. For instance, quarterly vulnerability assessments can help detect and address vulnerabilities proactively.
* **Adopt CA-6 - Security Authorization:** Establish a formal process for authorizing systems before they are deployed. Ensure that security controls are thoroughly assessed and approved before systems go live.
* **Implement CA-7 - Continuous Monitoring**: Deploy continuous monitoring tools to detect and respond to security incidents in real-time, enhancing incident response capabilities.
* **Adopt CA-8 - Security Assessment Report**: Following security assessments, produce comprehensive assessment reports detailing identified vulnerabilities, their severity, and recommended remediation actions. Use these reports as actionable roadmaps for improving security.
  + 1. **Audit and Accountability (AU):**
* **Enhance AU-3 - Content of Audit Records**: Ensure that audit logs comprehensively record security events, including failed login attempts and access control changes. Maintain detailed audit records for forensic analysis.
* **Implement AU-4 - Auditable Events:** Ensure that systems generate auditable events for critical security events, including unauthorized access attempts and changes to security configurations.
* **Leverage AU-6 - Audit Review, Analysis, and Reporting**: Regularly review and analyze audit logs. Utilize automated log analysis tools to promptly identify and respond to suspicious activities.
* **Leverage AU-12 - Audit Generation**: Utilize automated tools for generating and maintaining audit logs, ensuring they capture relevant events and remain tamper-resistant.
  + 1. **Configuration Management (CM):**
* **Apply CM-2 - Baseline Configuration**: Establish and maintain secure baseline configurations for all systems. Consistently apply patches and updates to address known vulnerabilities.
* **Implement CM-3 - Configuration Change Control**: Enforce robust change control processes to assess and document the security impact of system configuration changes.
* **Implement CM-6 - Configuration Settings:** Configure systems to disable or remove unnecessary and insecure services. For example, disable Telnet and other insecure services in favor of secure alternatives like SSH.
  + 1. **Incident Response (IR):**
* **Develop IR-4 - Incident Handling**: Establish an incident response plan that encompasses detection, reporting, and response procedures. Conduct periodic incident response exercises to ensure preparedness.
* **Enhance IR-5 - Incident Monitoring:** Establish continuous incident monitoring capabilities to detect and respond to security incidents promptly. Implement Security Information and Event Management (SIEM) solutions to correlate and analyze security events.
  + 1. **Security Training and Awareness (AT):**
* **Promote AT-2 - Security Awareness and Training**: Provide comprehensive security awareness and training programs to educate employees on security risks and best practices. Emphasize the importance of strong passwords, secure configurations, and incident reporting.
* **Leverage AT-3 - Role-Based Security Training:** Provide role-specific security training to employees based on their responsibilities. For instance, administrators should receive specialized training on secure system configurations and access controls.

By implementing these NIST 800-53 controls and corresponding examples, PHL can significantly enhance its security posture, mitigate vulnerabilities, and fortify its defences against future cyberattacks.

**5.2 Recommended Potential Adjustments to Security Policy**

Based on the NIST 800-53 controls and recommendations2, here are 20 potential adjustments to PHL's security policy:

1. **Data Classification and Handling:**

* Develop a data classification policy to categorize data based on sensitivity and define appropriate handling, storage, and access controls for each category.
* Educate employees on data classification and handling guidelines.

1. **Password Policy Enhancement:**

* Adjust the current password policy to align with NIST 800-53 guidelines, including requirements for password complexity, length, and regular changes.
* Require multi-factor authentication (MFA) for privileged accounts to add an additional layer of security.

1. **Access Control and Account Lockout:**

* Formalize an account lockout policy, specifying the number of failed login attempts allowed and the duration of lockout periods.
* Implement a role-based access control (RBAC) model to ensure that users have the minimum level of access required for their responsibilities.

1. **Network Access Controls:**

* Strengthen network access controls by implementing intrusion detection and prevention systems (IDS/IPS) to detect and block malicious traffic.
* Enforce strict access controls at the network perimeter to limit unauthorized access.

1. **Continuous Monitoring:**

* Establish a continuous monitoring program to track security events, network traffic, and system changes in real-time.
* Implement Security Information and Event Management (SIEM) tools to correlate and analyze logs for early threat detection.

1. **Vulnerability Management:**

* Introduce a systematic vulnerability management program that includes regular vulnerability scanning and penetration testing.
* Prioritize and remediate vulnerabilities based on severity and potential impact.

1. **Security Awareness Training:**

* Develop and implement role-specific security awareness training for employees to enhance their understanding of security risks and best practices.
* Conduct regular phishing awareness exercises to test and improve employee resilience to social engineering attacks.

1. **Incident Response Plan Enhancement:**

* Update the incident response plan to align with NIST 800-53 controls and guidelines.
* Define clear procedures for incident detection, reporting, containment, eradication, and recovery.
* Maintain an up-to-date repository of security documentation.

1. **Configuration Management:**

* Implement strict change control processes to assess and document the security impact of system configuration changes.
* Disable or remove unnecessary and insecure services to minimize attack surfaces.

1. **Security Testing and Evaluation:**

* Integrate security testing and evaluation as a standard practice in the policy. This includes vulnerability scanning, penetration testing, and security assessments during system development and maintenance phases.

1. **Security Assessment and Authorization:**

* Formalize a security authorization process for all systems before deployment.
* Use assessment reports to prioritize security improvements and ensure that security controls are implemented effectively.

1. **Network Segmentation:**

* Review and enhance network segmentation to isolate critical systems and sensitive data from less secure areas.
* Ensure that firewalls and access controls are properly configured to enforce network segmentation.

1. **Regular Security Audits:**

* Incorporate regular security audits and assessments into the security policy to identify and remediate vulnerabilities proactively.
* Define the frequency and scope of security audits and the responsible parties.

1. **Logging and Monitoring:**

* Expand the logging and monitoring policy to include comprehensive auditing of security-relevant events.
* Establish procedures for reviewing and analyzing audit logs and responding to incidents promptly.

1. **Patch Management:**

* Formalize a patch management policy to ensure the timely application of security updates and patches.
* Define patching schedules and responsible personnel for patch management.

1. **Default Credentials and Unused Services:**

* Prohibit the use of default credentials for any system or service. Ensure that all default passwords are changed during system deployment.
* Establish a policy to disable or remove unnecessary and insecure services, such as Telnet, SSH, or FTP, from production systems.

1. **Data Encryption:**

* Enforce data encryption policies to protect sensitive data at rest and during transmission.
* Implement encryption mechanisms for databases containing customer-sensitive information.

1. **Compliance and Regulatory Alignment:**

* Align the security policy with relevant industry regulations and compliance standards, such as GDPR, PIPEDA, HIPAA, or PCI-DSS if applicable to PHL's operations.

1. **Security Metrics and Key Performance Indicators (KPIs):**

* Define measurable security metrics and KPIs to assess the effectiveness of security controls and incident response procedures.
* Regularly report on these metrics to management for visibility into the organization's security posture.

1. **Regular Policy Review:**

* Establish a schedule for regular policy reviews and updates to ensure that security practices remain aligned with evolving threats and industry standards.
* Involve key stakeholders in policy reviews to gather insights and improve policy effectiveness.

These adjustments to the security policy align with NIST 800-53 controls and best practices, helping PHL enhance its security posture, minimize vulnerabilities, and respond effectively to cybersecurity threats. Regular updates and ongoing training are essential to maintain the effectiveness of these policies.

**Appendix: References**

1. Computer Security Incident Handling Guide - Special Publication 800-61 Revision 2 | NIST.gov (August 2012) - <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf>

2. Security and Privacy Controls for Information Systems and Organizations | NIST Joint Task Force Working Group (September 2020) - <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r5.pdf>

3. Exploit Public-Facing Application | MITRE ATT&CK.org; Praetorian; Yossi Weizman, Azure Defender Research Team (April 14, 2023) - <https://attack.mitre.org/techniques/T1190/>

4. Active Scanning: Wordlist Scanning | MITRE ATT&CK; Elvis Veliz, Jan Petrov, Richard Julian, Citi; (April 15, 2022) - <https://attack.mitre.org/techniques/T1595/003/>

5. Command and Scripting Interpreter | MITRE ATT&CK (March 27, 2023) - <https://attack.mitre.org/techniques/T1059/>

6. Masquerading | MITRE ATT&CK (April 7, 2023) - <https://attack.mitre.org/techniques/T1036/>

7. Data Encoding | MITRE ATT&CK (April 21, 2023) - <https://attack.mitre.org/techniques/T1132/>

8. Server Software Component: Web Shell | MITRE ATT&CK (March 30, 2023) - <https://attack.mitre.org/techniques/T1505/003/>

9. Network Service Discovery | MITRE ATT&CK (March 30, 2023) - <https://attack.mitre.org/techniques/T1046/>

10. Compromised Web Servers and Web Shells - Threat Awareness and Guidance | CISA.gov (August 09, 2017) - <https://www.cisa.gov/news-events/alerts/2015/11/10/compromised-web-servers-and-web-shells-threat-awareness-and-guidance>

11. Brute Force | MITRE ATT&CK (April 14, 2023) - <https://attack.mitre.org/techniques/T1110/>

12. Abuse Elevation Control Mechanism | MITRE ATT&CK (April 21, 2023) - <https://attack.mitre.org/techniques/T1548/>

13. Application Window Discovery | MITRE ATT&CK (April 14, 2023) - <https://attack.mitre.org/techniques/T1010/>

14. Exfiltration Over Alternative Protocol | MITRE ATT&CK (April 15, 2023) - <https://attack.mitre.org/techniques/T1048/>

15. How Will A Ransomware Attack Affect Your Business? | cm-alliance.com (May 19, 2022) - <https://www.cm-alliance.com/cybersecurity-blog/how-will-a-ransomware-attack-affect-your-business>

16. CVE-2020-1927 Detail | NIST NVD (July, 25, 2022) - [NVD - CVE-2020-1927 (nist.gov)](https://nvd.nist.gov/vuln/detail/CVE-2020-1927)