



## Official incident report

Event ID: 268

Rule Name: SOC292 - Possible PHP Injection Detected (CVE-2024-4577)

Made By

LinkedIn: Engineer.Ahmed Mansour

Link: <https://www.linkedin.com/in/ahmed-mansour-5631b5323/>

Github link: <https://github.com/AhmedMansour93>

# Table of contents

<b>Official incident report</b>	<b>1</b>
Event ID: 268	1
Rule Name: SOC292 - Possible PHP Injection Detected (CVE-2024-4577)	1
<b>Table of contents</b>	<b>2</b>
<b>Event Details</b>	<b>3</b>
<b>Network Information Details</b>	<b>4</b>
<b>Analysis</b>	<b>5</b>
Log management	5
<b>Security Email</b>	<b>11</b>
<b>Detection</b>	<b>12</b>
Threat intelligence	12
<b>Endpoint Security</b>	<b>14</b>
<b>Conclusion</b>	<b>15</b>

# Event Details

**Event ID:**

268

**Event Date and Time:**

Jun, 12, 2024, 08:26 AM

**Rule:**

SOC292 - Possible PHP Injection Detected (CVE-2024-4577)

**Level:**

Incident Responder

**Hostname:**

PHP-Server

**HTTP Request Method:**

POST

**Requested URL:**

POST /test.hello?%ADd+allow\_url\_include%3d1+%ADd+auto\_prepend\_file%3dphp://input  
HTTP/1.1

**Alert Trigger Reason:**

An HTTP request containing suspicious PHP directives (e.g., allow\_url\_include, auto\_prepend\_file) was detected, potentially exploiting CVE-2024-4577. This could indicate an attempt to inject and execute arbitrary PHP code on the server.

**L1 Note:**

I saw that the request in the log that triggered the alert received HTTP response code 400. However, I could not comment on whether the attack was successful based on the following logs.

# Network Information Details

## Destination IP Address:

172.16.20.59 internal

## Source IP Address:

185.177.124.100 external

### *Destination IP Address:*

- **172.16.20.59** (Internal)
- This is an internal IP address associated with the PHP-Server in your network. IP addresses in the range 172.16.0.0 to 172.31.255.255 are private and used within internal networks. These addresses are not routable on the public internet, meaning they belong to your organization's internal infrastructure.

### *Source IP Address:*

- **185.177.124.100** (External)
- This is a public IP address, meaning it comes from outside your organization's internal network. It indicates that the attack originated from an external entity, possibly over the internet.
- **The attack is external.**

# Analysis:

## Log Management

We'll proceed by entering the destination IP address and reviewing the results.

Please refer to the attached image for further details regarding the attack.

DATE ↑	TYPE	SRC ADDRESS	SRC PORT	DEST. ADDRESS	DEST. PORT	RAW
Columns	Operator	Value				
X Dest. Address	contains	172.16.20.59				
Jun, 12, 2024, 08:29 AM	Firewall	185.177.124.1...	11111	3.143.143.219	80	
Jun, 12, 2024, 08:29 AM	Firewall	185.177.124.1...	11101	3.143.143.219	80	
Jun, 12, 2024, 08:29 AM	Firewall	185.177.124.1...	60193	3.143.143.219	80	
Jun, 12, 2024, 08:29 AM	Firewall	185.177.124.1...	63417	3.143.143.219	80	
Jun, 12, 2024, 08:31 AM	Proxy	185.177.124.1...	26537	3.143.143.219	80	
Jun, 12, 2024, 08:31 AM	Firewall	185.177.124.1...	13164	3.143.143.219	80	
Jun, 12, 2024, 08:31 AM	Firewall	185.177.124.1...	44581	3.143.143.219	80	

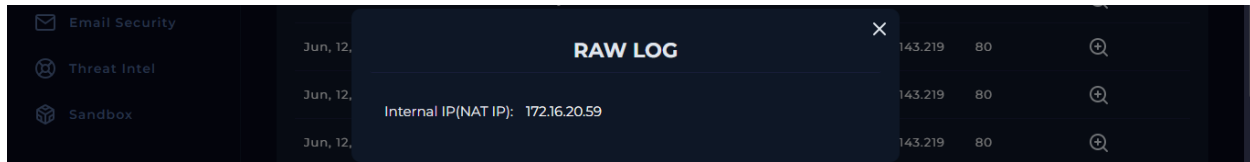
### 8 Logs records for the destination IP.

Please refer to the attached image for further details regarding the attack.

We will explain all of them step by step

## Log Analysis

- **Log1:**



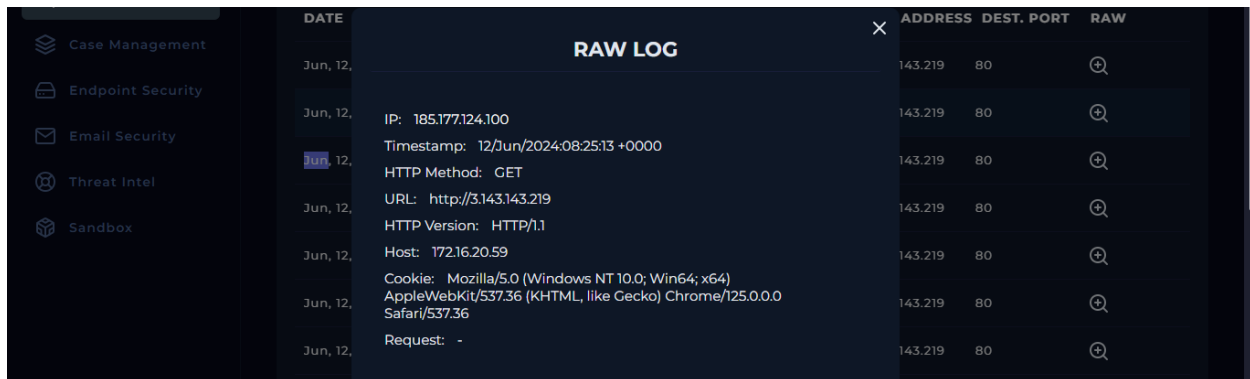
	DATE	RAW LOG	ADDRESS	DEST. PORT	RAW
	Jun, 12,	Internal IP(NAT IP): 172.16.20.59	143.219	80	+
	Jun, 12,		143.219	80	+
	Jun, 12,		143.219	80	+

- **Internal IP (NAT IP): 172.16.20.59**

### *What Happened:*

This log simply identifies the internal IP of the target, indicating that the system with IP 172.16.20.59 is under attack.

- **Log2:**



	DATE	RAW LOG	ADDRESS	DEST. PORT	RAW
	Jun, 12,		143.219	80	+
	Jun, 12,	IP: 185.177.124.100	143.219	80	+
	Jun, 12,	Timestamp: 12Jun/2024:08:25:13 +0000	143.219	80	+
	Jun, 12,	HTTP Method: GET	143.219	80	+
	Jun, 12,	URL: http://3.143.143.219	143.219	80	+
	Jun, 12,	HTTP Version: HTTP/1.1	143.219	80	+
	Jun, 12,	Host: 172.16.20.59	143.219	80	+
	Jun, 12,	Cookie: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36	143.219	80	+
	Jun, 12,	Request: -	143.219	80	+

### *What Happened:*

The attacker initially sends an HTTP `GET` request to the server. This appears to be a reconnaissance step, trying to gather basic information about the server's response to a simple GET request. The URL in the request does not contain any malicious payload yet.

- **Log3:**

	DATE	RAW LOG	ADDRESS	DEST. PORT	RAW
	Jun, 12,		143.219	80	+
	Jun, 12,	IP: 185.177.124.100	143.219	80	+
	Jun, 12,	Timestamp: 12Jun/2024:08:25:14 +0000	143.219	80	+
	Jun, 12,	HTTP Method: GET	143.219	80	+
	Jun, 12,	URL: http://3.143.143.219	143.219	80	+
	Jun, 12,	HTTP Version: HTTP/1.1	143.219	80	+
	Jun, 12,	HTTP Response Code: 404	143.219	80	+
	Jun, 12,	Host: 172.16.20.59	143.219	80	+
	Jun, 12,	Cookie: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36	143.219	80	+
	Jun, 12,	Request: /favicon.ico	143.219	80	+

*What Happened:*

This log shows the attacker following up with another GET request for `/favicon.ico`, a common file in web servers. The 404 (Not Found) response indicates the file doesn't exist on the server. The attacker is likely testing for available files and directories.

- **Log4:**

	DATE	RAW LOG	ADDRESS	DEST. PORT	RAW
	Jun, 12,		143.219	80	+
	Jun, 12,	IP: 185.177.124.100	143.219	80	+
	Jun, 12,	Timestamp: 12Jun/2024:08:26:11 +0000	143.219	80	+
	Jun, 12,	HTTP Method: POST	143.219	80	+
	Jun, 12,	URL: http://3.143.143.219	143.219	80	+
	Jun, 12,	HTTP Version: HTTP/1.1	143.219	80	+
	Jun, 12,	HTTP Response Code: 400	143.219	80	+
	Jun, 12,	Host: 172.16.20.59	143.219	80	+
	Jun, 12,	Cookie: curl/8.3.0	143.219	80	+
	Jun, 12,	Request: /test.hello?	143.219	80	+
	Jun, 12,	%ADd+allow_url_include%3d1+%ADd+auto_prepend_file%3dphp://input	143.219	80	+

*What Happened:*

Here, the attacker tries a **Remote Code Execution (RCE)** attack by sending a POST request with a suspicious query string, attempting to manipulate PHP configurations (`allow_url_include=1` and `auto_prepend_file=php://input`). This indicates that the attacker is trying to execute code by including malicious input directly into the server's PHP interpreter.

The server responds with a 400 (Bad Request), meaning the attack attempt may have been malformed or rejected, but this is an indicator of a more advanced attack attempt.

- **Log5:**

DATE	RAW LOG	ADDRESS	DEST. PORT	RAW
Jun, 12,		143.219	80	🔍
Jun, 12,	IP: 185.177.124.100	143.219	80	🔍
Jun, 12,	Timestamp: 12/Jun/2024:08:27:59 +0000	143.219	80	🔍
Jun, 12,	HTTP Method: POST	143.219	80	🔍
Jun, 12,	URL: http://3.143.143.219	143.219	80	🔍
Jun, 12,	HTTP Version: HTTP/1.1	143.219	80	🔍
Jun, 12,	HTTP Response Code: 200	143.219	80	🔍
Jun, 12,	Host: 172.16.20.59	143.219	80	🔍
Jun, 12,	Cookie: python-requests/2.28.1	143.219	80	🔍
Jun, 12,	Request: /index.php?%ADd+allow_url_include%3D1+-d+auto_prepend_file%3Dphp://input	143.219	80	🔍

*What Happened:*

In this request, the attacker tries again to execute malicious PHP code by injecting a payload into `/index.php`. This time, the response is 200 (OK), indicating the server has processed the request. The attacker may have successfully executed arbitrary code on the server using this method, meaning the server is likely compromised.

- **Log6:**

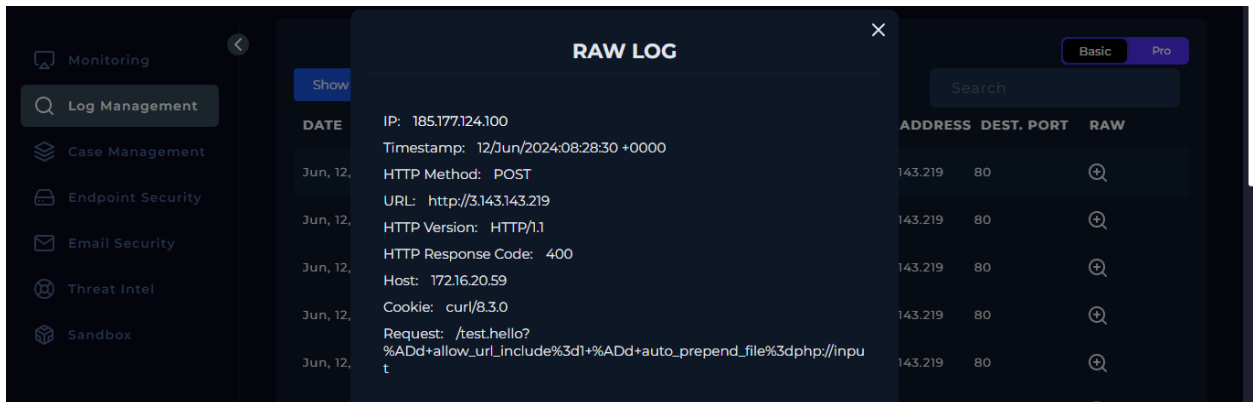
DATE	RAW LOG	ADDRESS	DEST. PORT	RAW
Jun, 12,		143.219	80	🔍
Jun, 12,		143.219	80	🔍
Jun, 12,	IP: 185.177.124.100	143.219	80	🔍
Jun, 12,	Timestamp: 12/Jun/2024:08:28:24 +0000	143.219	80	🔍
Jun, 12,	HTTP Method: POST	143.219	80	🔍
Jun, 12,	URL: http://3.143.143.219	143.219	80	🔍
Jun, 12,	HTTP Version: HTTP/1.1	143.219	80	🔍
Jun, 12,	HTTP Response Code: 200	143.219	80	🔍
Jun, 12,	Host: 172.16.20.59	143.219	80	🔍
Jun, 12,	Cookie: python-requests/2.28.1	143.219	80	🔍
Jun, 12,	Request: /index.php?%ADd+allow_url_include%3D1+-d+auto_prepend_file%3Dphp://input	143.219	80	🔍

*What Happened:*

The attacker sends a similar request as in Log 5, targeting `/index.php` again with the same PHP configuration injection method. Another 200 (OK) response indicates that the attacker is repeatedly attempting to execute commands on the server, possibly to establish a backdoor or upload malicious scripts.



- **Log7:**



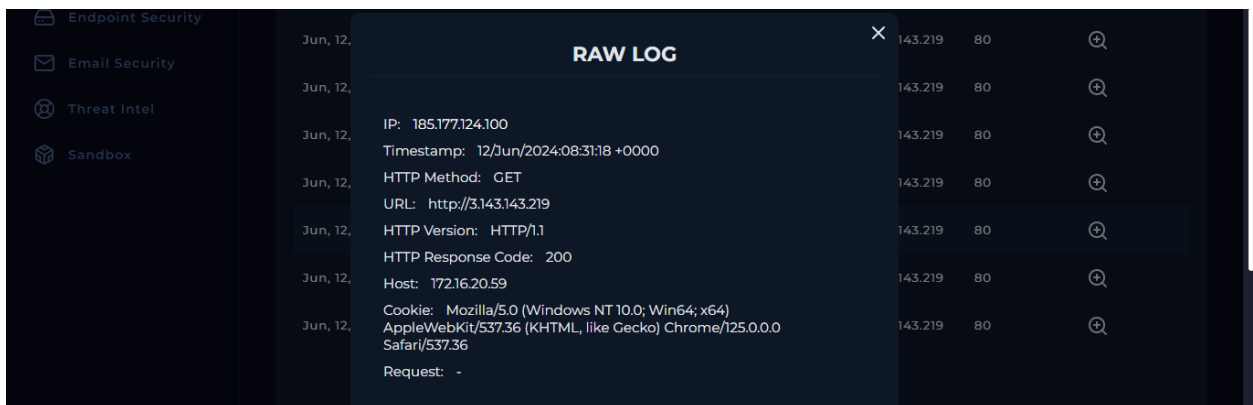
The screenshot shows a 'RAW LOG' window with a search bar and a table of log entries. The left sidebar contains navigation options: Monitoring, Log Management (selected), Case Management, Endpoint Security, Email Security, Threat Intel, and Sandbox. The table has columns for DATE, ADDRESS, DEST. PORT, and RAW. The log entry for Log 7 shows a POST request from IP 185.177.124.100 to 143.219.80, resulting in a 400 status code.

DATE	ADDRESS	DEST. PORT	RAW
Jun, 12,	143.219	80	IP: 185.177.124.100 Timestamp: 12/Jun/2024:08:28:30 +0000 HTTP Method: POST URL: http://3.143.143.219 HTTP Version: HTTP/1.1 HTTP Response Code: 400 Host: 172.16.20.59 Cookie: curl/8.3.0 Request: /test.hello? %ADd+allow_url_include%3d1+%ADd+auto_prepend_file%3dphp://inpu t

*What Happened:*

The attacker tries the same RCE attempt as in Log 4, but with a similar result: the server responds with a 400 (Bad Request), meaning this attempt likely failed.

- **Log8:**



The screenshot shows a 'RAW LOG' window with a search bar and a table of log entries. The left sidebar contains navigation options: Endpoint Security, Email Security, Threat Intel, and Sandbox. The table has columns for DATE, ADDRESS, DEST. PORT, and RAW. The log entry for Log 8 shows a GET request from IP 185.177.124.100 to 143.219.80, resulting in a 200 status code.

DATE	ADDRESS	DEST. PORT	RAW
Jun, 12,	143.219	80	IP: 185.177.124.100 Timestamp: 12/Jun/2024:08:31:18 +0000 HTTP Method: GET URL: http://3.143.143.219 HTTP Version: HTTP/1.1 HTTP Response Code: 200 Host: 172.16.20.59 Cookie: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36 Request: -

*What Happened:*

The final log shows the attacker sending a GET request, and receiving a 200 (OK) response. This could be the attacker verifying if the server is still responsive and checking the state after the RCE attempts.

## What Happened?

The attacker, originating from IP 185.177.124.100, attempted several actions:

1. **Initial Reconnaissance:** The attacker probed the server using basic GET requests to see if it would respond, checking for files like `/favicon.ico`.
2. **Remote Code Execution Attempts:** The attacker used POST requests with parameters such as `allow_url_include` and `auto_prepend_file` to try and inject malicious code into the server's PHP interpreter.
3. **Successful Exploitation:** Logs 5 and 6 indicate that the attacker successfully executed commands or uploaded malicious files by exploiting the server's vulnerability to include and execute code.
4. **Verification:** After the successful exploitation, the attacker sent further requests to check the status of the server, confirming that it remained responsive and accessible after the attack.

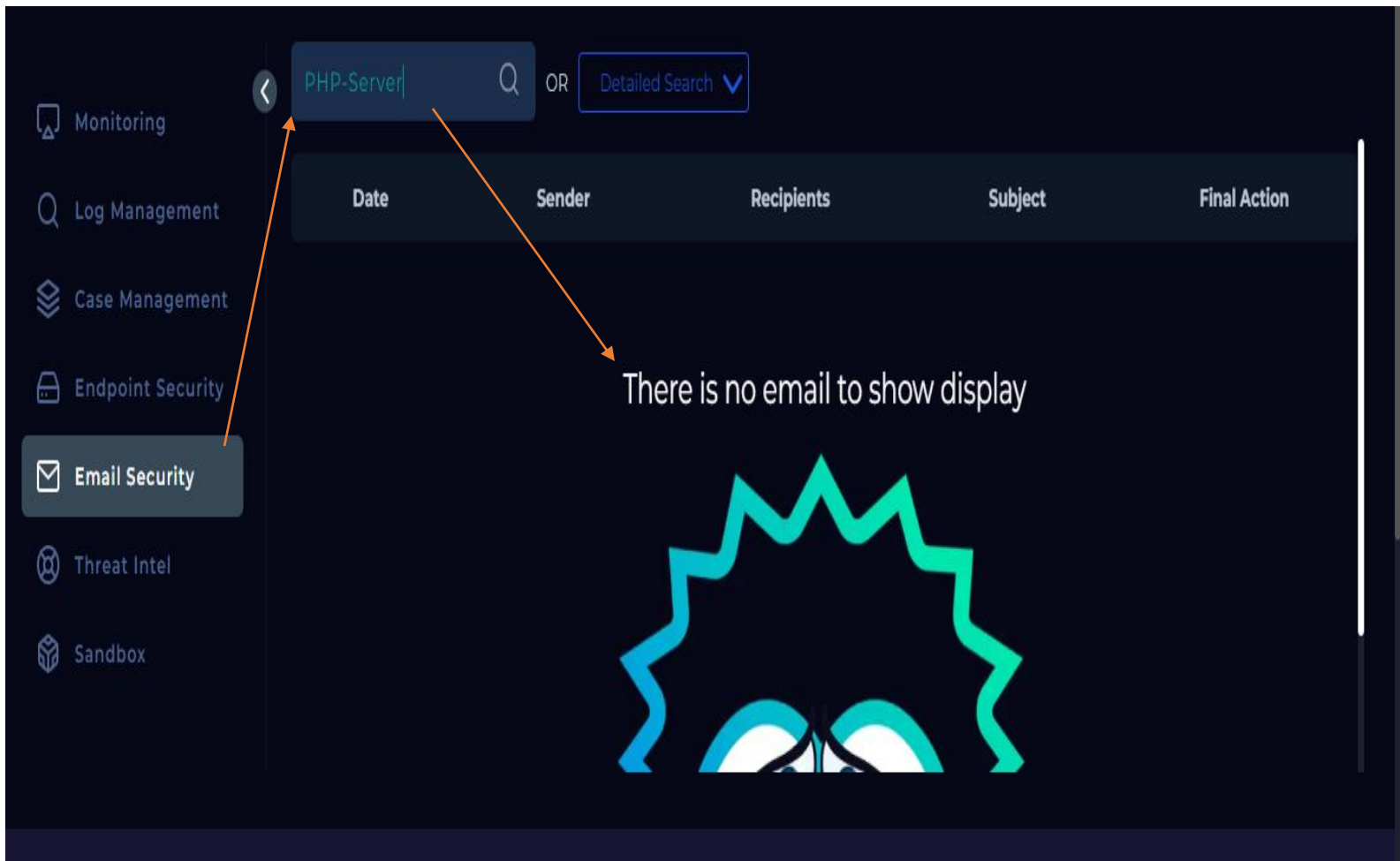
### • The Attack Were Successful.

- The attack type is **Command Injection**, and here are the detailed reasons why it's not any of the other attack types (IDOR, LFI & RFI, SQL injection, XML injection, or XSS):

## Why It's Command Injection:

1. **Use of Command-Line Switches (`-d, %ADd`):** The attacker in the logs is passing command-line switches, like `-d`, which is commonly used in PHP's command-line interface to modify runtime configuration. This indicates an attempt to inject commands directly into the system.
- This is clearly a **Command Injection** attack, where the attacker is manipulating how PHP runs by injecting command-line switches into the request. The other attack types from the list (IDOR, LFI & RFI, SQL injection, XML injection, XSS) don't fit the observed behavior, as none of them involve command-line execution or direct manipulation of the server's runtime environment in the way that Command Injection does.

## Email Security:

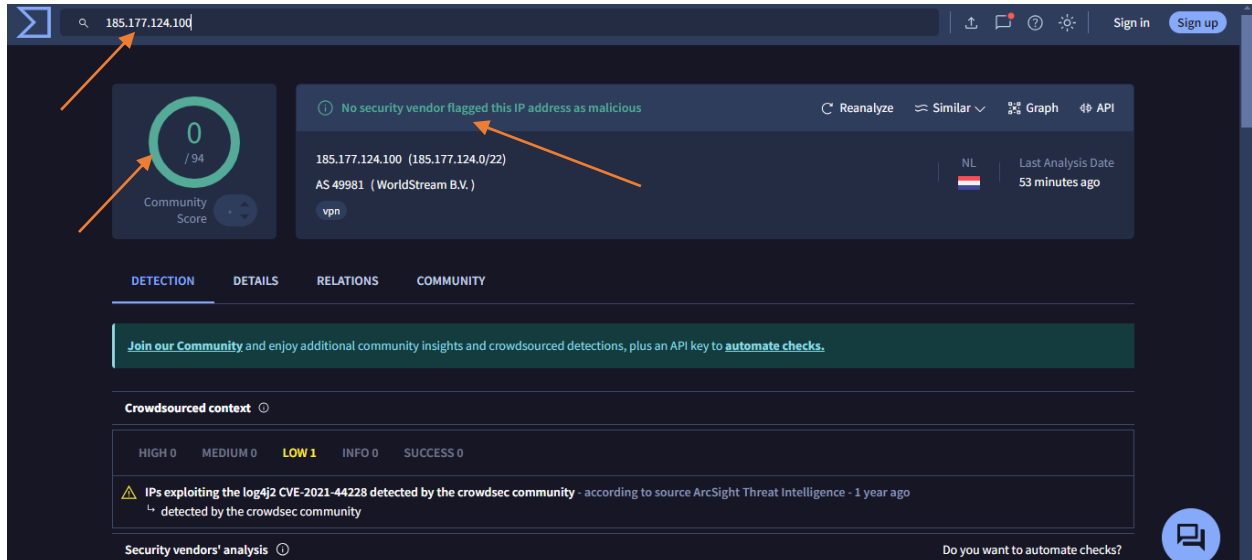


- **Despite entering the source host name in the email security section, no emails have been sent, indicating that the attack was not executed.**

# Detection:

## Threat Intelligence Results

We will conduct a comprehensive scan of the source IP address using VirusTotal to assess its reputation and determine if it has been associated with any known malicious activities or threats.



- The results is clean.
- [Reference result.](#)

- We will conduct a comprehensive scan of the source IP address using abuseipdb to assess its reputation and determine if it has been associated with any known malicious activities or threats.

Check an IP Address, Domain Name, or Subnet  
e.g. 156.197.31.248, microsoft.com, or 5.188.10.0/24

156.197.31.248

CHECK

**185.177.124.100 was found in our database!**

This IP was reported **39** times. Confidence of Abuse is **21%**.

21%

ISP: WorldStream B.V.

Usage Type: Data Center/Web Hosting/Transit

Hostname(s): 185-177-124-100.hosted-by-worldstream.net

Domain Name: worldstream.nl

Country: Netherlands

City: Naaldwijk, Zuid-Holland

IP Info including ISP, Usage Type, and Location provided by IP2Location. Updated monthly.

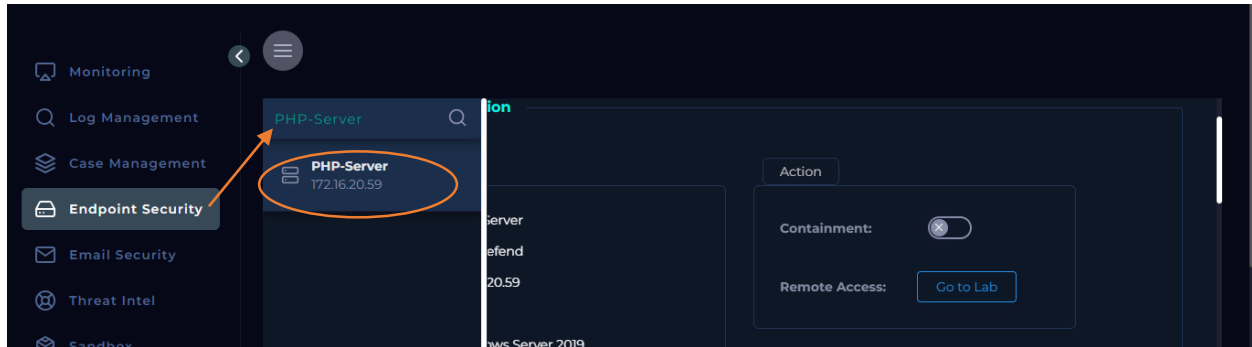
REPORT 185.177.124.100 WHOIS 185.177.124.100

- **IP Address:** 185.177.124.100
- **Reports:** This IP has been reported 39 times in the AbuseIPDB database.
- **Confidence of Abuse:** 21% (indicating moderate likelihood of malicious activity).
- **ISP:** WorldStream B.V.
- **Usage Type:** Data Center/Web Hosting/Transit
- **Hostname:** 185-177-124-100.hosted-by-worldstream.net
- **Domain:** worldstream.nl
- **Country:** Netherlands
- **City:** Naaldwijk, Zuid-Holland

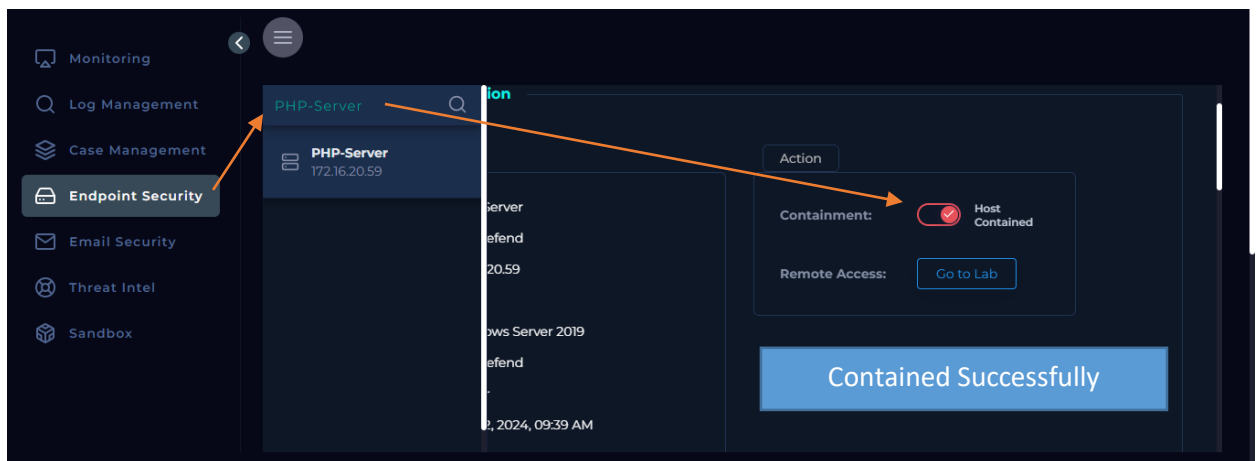
This information indicates that the IP address is associated with a data center in the Netherlands, and while it has been flagged for potential abuse, the confidence level is relatively low at 21%. However, due to the number of reports, further investigation is warranted.

- **The Traffic is Malicious**

## Endpoint Security:



- Based on our analysis, the reconnaissance activity identified is focused on gathering victim and company user's identity information's.
- The device must be Contain. And we contained Successfully.



## Conclusion:

The incident involving the source IP address **185.177.124.100** targeted our internal PHP-Server (IP: **172.16.20.59**) through a sophisticated **Command Injection** attack, exploiting known PHP vulnerabilities (CVE-2024-4577). The attacker initiated with reconnaissance techniques, using HTTP GET requests to probe the server, followed by multiple POST requests containing command-line switches (`-d, %ADd`), aiming to alter PHP configurations and execute arbitrary code. Logs indicate that several attempts resulted in **200 OK** responses, confirming successful code execution on the server.

The attack likely compromised the PHP-Server by executing commands via PHP configuration manipulation. However, some requests were met with **400 Bad Request** responses, indicating partial failure of the exploit. Despite this, the attacker was able to execute commands and potentially establish persistence, verifying the server's responsiveness post-exploitation.

A comprehensive reputation scan of the source IP address revealed it had been reported 39 times in AbuseIPDB, with a **21% confidence of abuse**, signaling moderate likelihood of malicious activity. The IP, associated with a data center in the Netherlands (WorldStream B.V.), confirms the external origin of the attack.

**Endpoint containment measures** were swiftly and successfully implemented, mitigating further damage. Given the severity of the attack, the situation warrants further investigation and additional security hardening to prevent future exploitation.

In conclusion, this incident highlights a critical vulnerability in our PHP server that was actively exploited, resulting in a successful command injection. Proactive response efforts and continuous monitoring have ensured the containment of the attack, but further scrutiny is necessary to ensure comprehensive protection.