

# Scan Results for flamman.se

Scan started: 2025-04-06 18:07:33

Initializing scan... Please wait while we analyze the target.

## Scanning in progress

The scan is now running. Results will appear here as they are processed.

## Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)

### Issue Explanation

The vulnerability alert details indicate that the web/application server is leaking information via the "X-Powered-By" HTTP response header field(s). This header is used to indicate the software or framework that is powering the web application. However, exposing this information can be a security risk.

### Impact Analysis

The potential risks and security impact if this vulnerability is exploited include:

- Information Disclosure: Attackers can identify the specific web server, application server, or framework that the application is using. This information can be used to research known vulnerabilities associated with those components.
- Targeted Attacks: Attackers can tailor their attacks to exploit known vulnerabilities specific to the identified components.
- Phishing: Attackers can craft phishing emails or malicious websites that appear legitimate, as they can mimic the look and feel of the application.

### Exploitation Details

An attacker might exploit this vulnerability by:

1. Identifying the "X-Powered-By" header in the HTTP response.
2. Using the information to research known vulnerabilities associated with the identified components.
3. Crafting targeted attacks or phishing attempts based on the information.

### Step-by-Step Remediation

To mitigate or resolve the issue, follow these steps:

1. Configure Server Headers: Ensure that the web server, application server, or load balancer is configured to suppress the "X-Powered-By" header. This can be done through server configuration files or using server-side scripting.
2. Use Custom Headers: If necessary, use custom headers to provide information about

- the application without exposing sensitive details.
3. Regular Updates: Keep all software components, including the web server and application server, up to date with the latest security patches.
  4. Security Testing: Regularly conduct security testing, including penetration testing and code reviews, to identify and remediate any potential vulnerabilities.
  5. Security Policies: Implement and enforce strong security policies, such as regular security audits and the principle of least privilege.

## References & Best Practices

- OWASP HTTP Security Headers: <https://owasp.org/www-project-owasp-top-ten/2017/A03-2017-Sensitive-Data-Exposure>
  - OWASP HTTP Security Headers: [https://www.owasp.org/index.php/OWASP\\_HTTP-Headers-Project](https://www.owasp.org/index.php/OWASP_HTTP-Headers-Project)
  - OWASP Security Headers Cheat Sheet: <https://www.owasp.org/index.php/Security-Headers>
  - OWASP Security Headers: <https://www.owasp.org/www-project-owasp-top-ten/2017/A03-2017-Sensitive-Data-Exposure>
- 

## Strict-Transport-Security Header Not Set

### Issue Explanation

The vulnerability alert details indicate that the web server is not enforcing HTTP Strict Transport Security (HSTS). HSTS is a security policy mechanism that ensures that all communications between a client and a server are over HTTPS, even if the initial connection is made over HTTP.

### Impact Analysis

The potential risks and security impact if this vulnerability is exploited include:

- Man-in-the-Middle (MitM) Attacks: Attackers can intercept and potentially modify the communication between the client and the server, leading to data interception or man-in-the-middle attacks.
- Downgrade Attacks: Attackers can force the client to downgrade to HTTP, bypassing the security provided by HTTPS.
- Phishing: Attackers can craft phishing emails or malicious websites that appear legitimate, as they can mimic the look and feel of the application.

### Exploitation Details

An attacker might exploit this vulnerability by:

1. Intercepting the initial HTTP connection to the server.
2. Redirecting the client to a malicious website or phishing page.
3. Performing man-in-the-middle attacks or downgrading the connection to HTTP.

### Step-by-Step Remediation

To mitigate or resolve the issue, follow these steps:

1. Configure HSTS: Ensure that the web server, application server, or load balancer is configured to enforce HSTS. This can be done through server configuration files or using server-side scripting.
2. Set HSTS Policy: Set the HSTS policy with a long duration and include the "includeSubDomains" and "preload" directives to enforce HSTS across all subdomains and to instruct browsers to preload the HSTS policy.
3. Regular Updates: Keep all software components, including the web server and application server, up to date with the latest security patches.
4. Security Testing: Regularly conduct security testing, including penetration testing and code reviews, to identify and remediate any potential vulnerabilities.
5. Security Policies: Implement and enforce strong security policies, such as regular security audits and the principle of least privilege.

## References & Best Practices

- OWASP HTTP Strict Transport Security: [https://www.owasp.org/www-project-owasp-top-ten/2017/A03\\_2017-Sensitive\\_Data\\_Exposure](https://www.owasp.org/www-project-owasp-top-ten/2017/A03_2017-Sensitive_Data_Exposure)
  - OWASP HTTP Strict Transport Security: [https://www.owasp.org/index.php/OWASP\\_HTTP\\_Strict\\_Transport\\_Security\\_Cheat\\_Sheet](https://www.owasp.org/index.php/OWASP_HTTP_Strict_Transport_Security_Cheat_Sheet)
  - OWASP HTTP Strict Transport Security: [https://www.owasp.org/www-project-owasp-top-ten/2017/A03\\_2017-Sensitive\\_Data\\_Exposure](https://www.owasp.org/www-project-owasp-top-ten/2017/A03_2017-Sensitive_Data_Exposure)
- 

## User Agent Fuzzer

### Issue Explanation

The vulnerability alert details indicate that the application is vulnerable to a User Agent Fuzzer. A User Agent Fuzzer is a tool that sends different User-Agent strings to a web server and checks for differences in the response.

### Impact Analysis

The potential risks and security impact if this vulnerability is exploited include:

- Information Disclosure: Attackers can identify different versions of the application or different features based on the responses.
- Phishing: Attackers can craft phishing emails or malicious websites that appear legitimate, as they can mimic the look and feel of the application.
- Targeted Attacks: Attackers can tailor their attacks to exploit known vulnerabilities specific to the identified versions or features.

### Exploitation Details

An attacker might exploit this vulnerability by:

1. Using a User Agent Fuzzer to send different User-Agent strings to the application.
2. Analyzing the responses to identify different versions or features of the application.
3. Crafting targeted attacks or phishing attempts based on the identified information.

### Step-by-Step Remediation

To mitigate or resolve the issue, follow these steps:

1. User-Agent String Validation: Implement strict validation of the User-Agent string to ensure that only expected values are accepted.
2. Feature Detection: Use feature detection techniques instead of relying on the User-Agent string to determine the capabilities of the client.
3. Regular Updates: Keep all software components, including the web server and application server, up to date with the latest security patches.
4. Security Testing: Regularly conduct security testing, including penetration testing and code reviews, to identify and remediate any potential vulnerabilities.
5. Security Policies: Implement and enforce strong security policies, such as regular security audits and the principle of least privilege.

## References & Best Practices

- OWASP User-Agent String: <https://www.owasp.org/www-project-owasp-top-ten/2017/A03-2017-Sensitive-Data-Exposure>
  - OWASP User-Agent String: [https://www.owasp.org/index.php/OWASP\\_Testing\\_Guide\\_Appendix\\_C#Testing\\_for\\_User-Agent\\_Spoofing](https://www.owasp.org/index.php/OWASP_Testing_Guide_Appendix_C#Testing_for_User-Agent_Spoofing)
  - OWASP User-Agent String: <https://www.owasp.org/www-project-owasp-top-ten/2017/A03-2017-Sensitive-Data-Exposure>
- 

# Nmap Scan Results Overview

## Hosts

- 35.228.57.67: The host at this IP address is up and running. It has 1 open port: 22 (ssh).

## Ports

- Port 22 (ssh): This port is open on 1 host: 35.228.57.67.

## Services

- SSH service is running on 1 host: 35.228.57.67.

## Vulnerabilities

- No vulnerabilities were found during the scan.

## Recommendations

- Further analysis is required to determine the security posture of the host and service.
- It is recommended to perform a more comprehensive scan with additional options to identify potential vulnerabilities and misconfigurations.

This overview provides a high-level summary of the Nmap scan results, including the host,

port, service, and any identified vulnerabilities. It is important to note that this is a simplified overview and further analysis is needed to fully assess the security posture of the network and its components.

## Scan Complete

Scan completed at: 2025-04-06 18:12:17