

External - Vulnerability Assessment and Penetration & Testing Report

23rd February 2024

Prepared By:

IT Team,

Verifacts Services Pvt Ltd,

No 4007, 1st and 2nd floor, 100ft Road, Indira Nagar, Bangalore – 560038

Document Details

Document Version Control

|  |  |
| --- | --- |
| Document Title | Internal - Vulnerability Assessment and Penetration Testing Report |
| Document ID | VERIFACTS\_EXTERNAL\_VAPT\_v4.0 |
| Document Version | Version 2.0 |
| Prepared By | Arun Kumar |
| Approved By | Rajiv Sharma |
| Date | 22nd June 2019 |

Document Submission Details

|  |  |
| --- | --- |
| Document Version | Version 2.0 |
| Submitted by | Arun Kumar |
| Department | Information Technology |
| Address | No 4007, 1st and 2nd floor, 100FT Road, Indira Nagar, Bangalore – 560 038 |
| Date of Completion | 23rd February 2024 |

Table of Contents

[Executive Summary 4](#_Toc121225024)

[Objective 4](#_Toc121225025)

[Scope 4](#_Toc121225026)

[Methodology 4](#_Toc121225027)

[Planning and Analysis 5](#_Toc121225028)

[Attack & Penetration testing 5](#_Toc121225029)

[Analysis & Reports 6](#_Toc121225030)

[Severity Rating 6](#_Toc121225031)

[CVSS v3 Score Range 6](#_Toc121225032)

[Ease of Exploitation 7](#_Toc121225033)

[Report Collation and Report Preparation 7](#_Toc121225034)

[Toolkit 7](#_Toc121225035)

[Summary of Findings & Recommendations 7](#_Toc121225036)

[Summary of Findings 7](#_Toc121225037)

[Graphical Summary 8](#_Toc121225038)

[Vulnerability Report 8](#_Toc121225039)

[Conclusion 15](#_Toc121225040)

# Executive Summary

IT Team, Verifacts Services Pvt Ltd conducted Vulnerability Assessment and Penetration Testing on all servers, network devices and IPs of the Company as a part of the external network. Industry standard framework and tools were used to perform the testing. During this assignment **1** **CRITICAL** **1 HIGH, 7 MEDIUM and 10 LOW** vulnerability were found. The details of the finding are as stated under.

## Objective

The goal was to find out technology vulnerabilities in the Servers and all network devices of the Company. The tests were carried out assuming the identity of an attacker or a user with malicious intent. No special conditions like firewall exceptions or disabling of IDS was required during this assessment but in few tests firewalls and end points are brought down to obtain the accuracy of results.

## Scope

As a part of the on-going activities to assess the security infrastructure of Verifacts Vulnerability Assessment and Penetration Testing for servers and network devices were carried out externally. As a part of the mandate, the IT Team carried out the activity from **19th February 2024 to 23rd February 2024**. This document details the findings and recommendations of the activity so conducted.

Scope of the servers, network devices and IP’s testing are stated in below.

|  |  |  |
| --- | --- | --- |
| Server / Domain / Application | IP Address | Location |
| SonicWall TZ 500D | 123.136.128.226 | HQ |
| SonicWall TZ 300D | 183.82.117.88 | Hyderabad |
| SonicWall TZ 300D | 103.16.202.240 | Chennai |
| SonicWall TZ 300D | 14.97.177.2 | Kolkata |
| [https://dr-vibe.verifacts.co.in](https://dr-vibe.verifacts.co.in/) | 13.127.170.117 | AWS - Cloud |
| [https://verifacts.co.in](https://verifacts.co.in/) | x.x.x.x | AWS - Cloud |
| [https://vibe.verifacts.co.in](https://vibe.verifacts.co.in/) | 3.7.230.213 | AWS - Cloud |
| [https://vibe.verifacts.ph](https://vibe.verifacts.ph/) | 54.255.110.253 | AWS - Cloud |
| VIBE\_PH\_DB Server | 3.0.142.99 | AWS - Cloud |
| VIBE\_DB Server | 3.6.175.176 | AWS - Cloud |

# Methodology

Verifacts Services Pvt Ltd believes in Securing infrastructure by using industry best standards for conducting Vulnerability Assessment and Penetration Tests.

The Vulnerability Assessment & Penetration Testing was done to test the security posture of the Information Systems as an external attacker also referred to as ethical hacker. The primary purpose of the assessment is to simulate an external attack by a user with network access, attempt to gain unauthorized information and enumerate any vulnerability that may exist. The methodology is to perform attacks in a controlled environment with minimum disruption to the production environment.

Our Security Assessment Flowchart is as follows,

Identification of Targets & Port Scanning

Report Collation and Preparation

Perform Re - Validation

Remediating Reported Vulnerabilities

Identification of Vulnerabilities and Potential Risk

# Planning and Analysis

Scope was defined based on the nature, timing and extent of the evaluation which was to be conducted. The nature and type of the tools to be used was determined by IT Team and approved by Chief Operating Officer. The machine(s) which was to be used for the conducting the tests was prepared accordingly. Testing was conducted in the normal business hours.

# Attack & Penetration testing

It was done presuming that the tester was an external attacker. Applicable tools were used for the purpose. POC for each of the vulnerability which was exploited during the test was compiled for further reporting along with the recommendation for closing them in a planned manner.

# Analysis & Reports

The analysis was performed based on the CVSS rating of the vulnerabilities reported by Nessus, NMAP, OWASP-ZAP and Kali Linux. The report was then created by articulating the analyses so performed along with recommendations for closure.

# Severity Rating

This rating depicts the severity of impact to the organization. Depending on the criticality of the device, it could represent business impact, financial impact or damage to customer, partner, or reputation. Remediation time frames for Critical & High Vulnerabilities are one month and Medium & Low Vulnerabilities are 2 Months or based on availability of resources.

**CRITICAL & HIGH**

It is imperative, that effort to be undertaken immediately to mitigates the vulnerabilities in this category.

**MEDIUM**

Medium threats need to be reviewed after which its treatment should be designed as a part of the security plan.

**LOW**

Low threats need to be reviewed.

# CVSS v3 Score Range

|  |  |
| --- | --- |
| CVSS V3 SCORE RANGE | SEVERITY IN ADVISORY |
| 0.1-3.9 | LOW |
| 4.0-6.9 | MEDIUM |
| 7.0-8.9 | HIGH |
| 9.0-10.0 | CRITICAL |

# 

# Ease of Exploitation

Ease of Exploitation has 3 values - Difficult, Moderate, Easy. This depicts the knowledge required for exploitation and gaining access to the concerned internal IPs. It essentially depends on the availability of the exploit and the technology required in carrying out the exploitation.

|  |  |  |  |
| --- | --- | --- | --- |
| **Severity Rating \ Ease of Exploitation** | **EASY** | **MODERATE** | **DIFFICULT** |
| **HIGH** | **HIGH** | **HIGH** | **MEDIUM** |
| **MEDIUM** | **HIGH** | **MEDIUM** | **LOW** |
| **LOW** | **MEDIUM** | **LOW** | **LOW** |

# Report Collation and Report Preparation

Results are then collated in tabular form for easy reference. Vulnerability will be accompanied by a short description and screenshots.

# Toolkit

* Nessus
* Nmap
* Kali Linux
* OWASP -ZAP
* Parrot OS

# Summary of Findings & Recommendations

## Summary of Findings

Below is the table that summarizes the list of findings discovered during vulnerability Assessment and Penetration Testing.

|  |  |  |
| --- | --- | --- |
| **Sr No** | **Severity Rating** | **Vulnerabilities** |
| 1 | **CRITICAL** | PHP Unsupported Version Detection |
| 2 | **HIGH** | Cloud Metadata Potentially Exposed |
| 3 | **MEDIUM** | SSL Certificate with Wrong Hostname |
| 4 | **MEDIUM** | HSTS Missing From HTTPS Server (RFC 6797) |
| 5 | **MEDIUM** | Hidden File Found |
| 6 | **MEDIUM** | CSP: Wildcard Directive |
| 7 | **MEDIUM** | Content Security Policy (CSP) Header Not Set |
| 8 | **MEDIUM** | Missing Anti-clickjacking Header |
| 9 | **MEDIUM** | Cross-Domain Misconfiguration |
| 10 | **LOW** | SSH Server CBC Mode Ciphers Enabled |
| 11 | **LOW** | SSH Weak Key Exchange Algorithms Enabled |
| 12 | **LOW** | Private IP Disclosure |
| 13 | **LOW** | Application Error Disclosure |
| 14 | **LOW** | Cookie No HttpOnly Flag |
| 15 | **LOW** | Cookie Without Secure Flag |
| 16 | **LOW** | Cookie without SameSite Attribute |
| 17 | **LOW** | Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s) |
| 18 | **LOW** | Server Leaks Version Information via "Server" HTTP Response Header Field |
| 19 | **LOW** | Timestamp Disclosure - Unix |

## Graphical Summary

The following pie chart summarizes the overall Vulnerability Severity determined across the samples.

# Vulnerability Report

PHP Unsupported Version Detection

**Severity of Vulnerability**

**CRITICAL**

**Affected IP**

Verifacts.co.in

**Description**

According to its version, the installation of PHP on the remote host is no longer supported.  
Lack of support implies that no new security patches for the product will be released by the vendor. As a result, it is likely to contain security vulnerabilities.

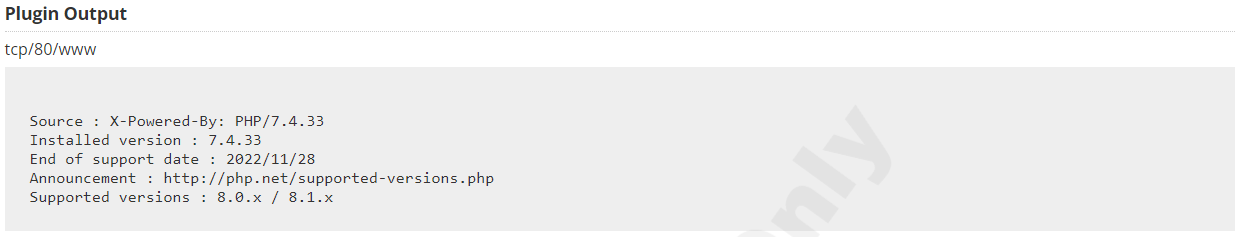
**Recommendation**

Upgrade to a version of PHP that is currently supported.

**Reference**

<https://www.php.net/eol.php>

**Output:**



Cloud Metadata Potentially Exposed

**Severity of Vulnerability**

**HIGH**

**Affected IP**

vibe.verifacts.co.in

vibe.verifacts.ph

**Description**

The Cloud Metadata Attack attempts to abuse a misconfigured NGINX server in order to access the instance metadata maintained by cloud service providers such as AWS, GCP and Azure. All of these providers provide metadata via an internal unroutable IP address ‘169.254.169.254’ - this can be exposed by incorrectly configured NGINX servers and accessed by using this IP address in the Host header field.

**Recommendation**

Do not trust any user data in NGINX configs. In this case it is probably the use of the $host variable which is set from the 'Host' header and can be controlled by an attacker.

**Reference**

<https://www.nginx.com/blog/trust-no-one-perils-of-trusting-user-input/>

SSL Certificate with Wrong Hostname

**Severity of Vulnerabilities**

**MEDIUM**

**Affected IP**

verifacts.co.in

**Description**

Hostname mismatch in the SSL certificate. This happens when the common name to which an SSL Even if a certificate is well-formed, signed, and follows the chain of trust, it may simply be a valid certificate for a different site than the site that the product is interacting with. If the certificate's host-specific data is not properly checked - such as the Common Name (CN) in the Subject or the Subject Alternative Name (SAN) extension of an X.509 certificate - it may be possible for a redirection or spoofing attack to allow a malicious host with a valid certificate to provide data, impersonating a trusted host. In order to ensure data integrity, the certificate must be valid and it must pertain to the site that is being accessed.

Even if the product attempts to check the hostname, it is still possible to incorrectly check the hostname. For example, attackers could create a certificate with a name that begins with a trusted name followed by a NUL byte, which could cause some string-based comparisons to only examine the portion that contains the trusted name.

This weakness can occur even when the product uses Certificate Pinning, if the product does not verify the hostname at the time a certificate is pinned.

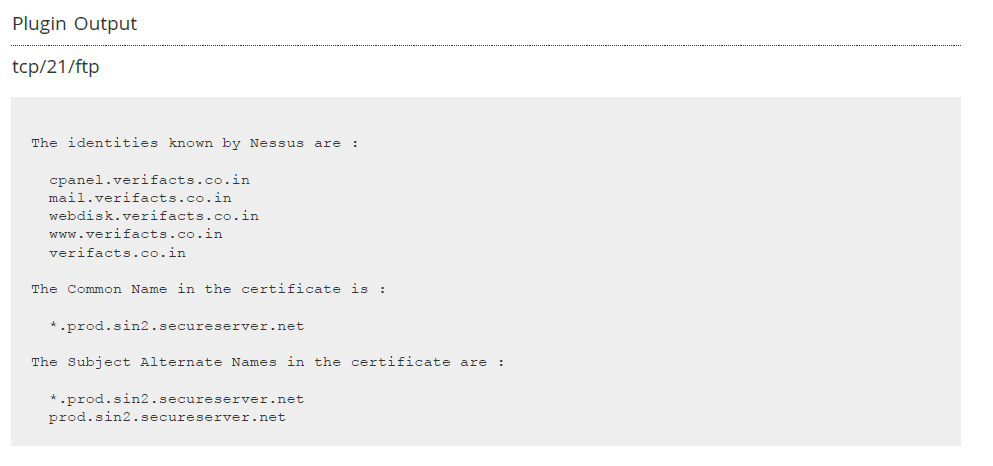
**Recommendations**

Purchase or generate a proper SSL certificate for this service.

**Reference**

|  |
| --- |
| https://cwe.mitre.org/data/definitions/297.html |

**Output**



HSTS Missing From HTTPS Server (RFC 6797)

**Severity of Vulnerabilities**

**MEDIUM**

**Affected IP**

verifacts.co.in

**Description**

The remote web server is not enforcing HSTS, as defined by RFC 6797. HSTS is an optional response header that can be configured on the server to instruct the browser to only communicate via HTTPS. The lack of HSTS allows downgrade attacks, SSL-stripping man-in-the-middle attacks, and weakens cookie-hijacking protections.

**Recommendations**

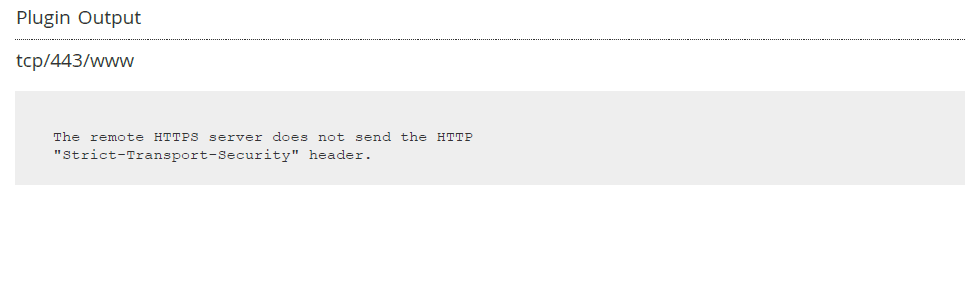
Configure the remote web server to use HSTS.

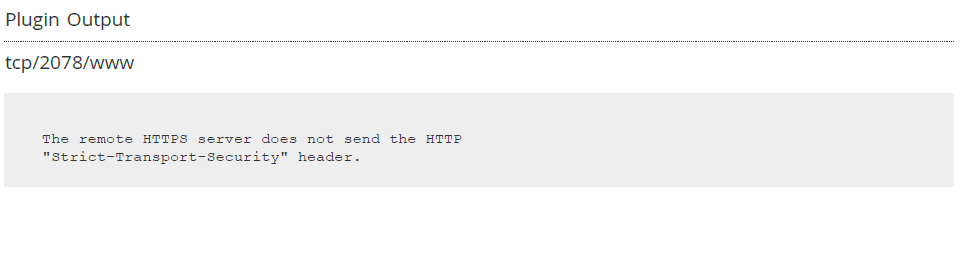
**Reference**

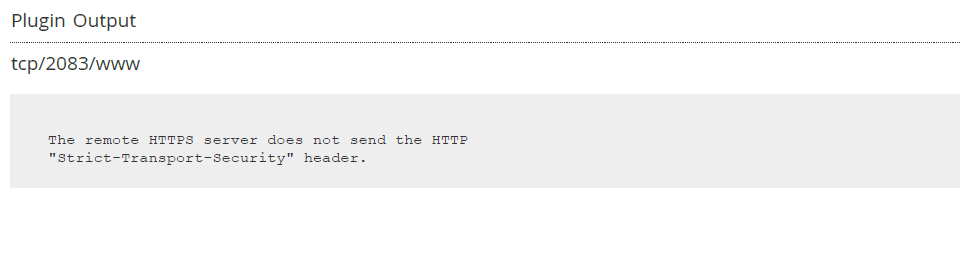
|  |
| --- |
|  |

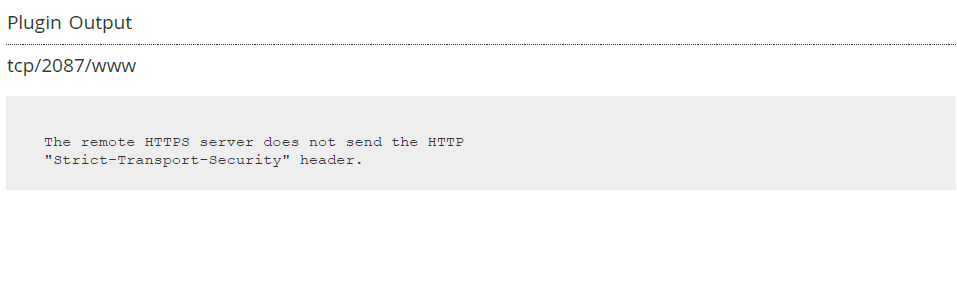
<https://datatracker.ietf.org/doc/html/rfc6797>

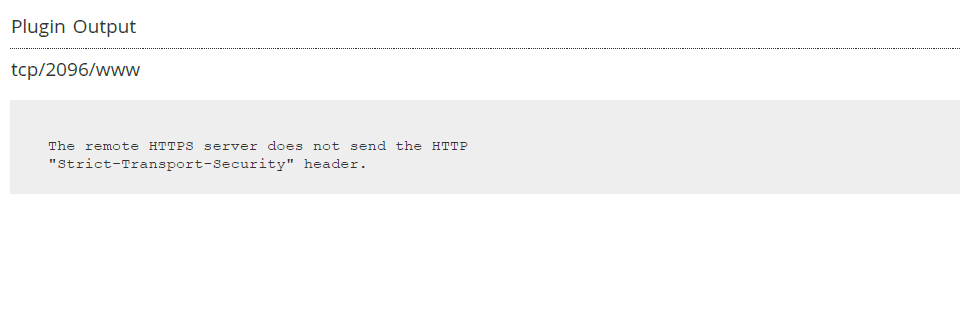
**Output**











Hidden File Found

**Severity of Vulnerabilities**

**MEDIUM**

**Affected IP**

vibe.verifacts.co.in

vibe.verifacts.ph

**Description**

The product places sensitive information into files or directories that are accessible to actors who are allowed to have access to the files, but not to the sensitive information.

**Recommendations**

Consider whether or not the component is actually required in production, if it isn't then disable it. If it is then ensure access to it requires appropriate authentication and authorization, or limit exposure to internal systems or specific source IPs, etc.

**Reference**

h[ttps://blog.hboeck.de/archives/892-Introducing-Snallygaster-a-Tool-to-Scan-for-Secrets-on-Web-Servers.html](https://blog.hboeck.de/archives/892-Introducing-Snallygaster-a-Tool-to-Scan-for-Secrets-on-Web-Servers.html)

CSP: Wildcard Directive

**Severity of Vulnerabilities**

**MEDIUM**

**Affected IP**

verifacts.co.in

vibe.verifacts.co.in

**Description**

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

**Recommendations**

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.

**Reference**

<https://www.w3.org/TR/CSP/>

Content Security Policy (CSP) Header Not Set

**Severity of Vulnerabilities**

**MEDIUM**

**Affected IP**

verifacts.co.in

vibe.verifacts.co.in

vibe.verifacts.ph

**Description**

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

**Recommendations**

Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.

**Reference**

<https://developer.mozilla.org/en-US/docs/Web/Security/CSP/Introducing_Content_Security_Policy>

Missing Anti-clickjacking Header

**Severity of Vulnerabilities**

**MEDIUM**

**Affected IP**

verifacts.co.in

vibe.verifacts.co.in

vibe.verifacts.ph

**Description**

The response does not include either Content-Security-Policy with ‘frame-ancestors’ directive or X-Frame-Options to protect against ‘ClickJacking’ attacks.

**Recommendations**

Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app. If you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. Alternatively consider implementing Content Security Policy's "frame-ancestors" directive.

**Reference**

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options>

Cross-Domain Misconfiguration

**Severity of Vulnerabilities**

**MEDIUM**

**Affected IP**

verifacts.co.in

vibe.verifacts.co.in

**Description**

Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server

**Recommendations**

Ensure that sensitive data is not available in an unauthenticated manner (using IP address white-listing, for instance). Configure the "Access-Control-Allow-Origin" HTTP header to a more restrictive set of domains, or remove all CORS headers entirely, to allow the web browser to enforce the Same Origin Policy (SOP) in a more restrictive manner.

**Reference**

<https://vulncat.fortify.com/en/detail?id=desc.config.dotnet.html5_overly_permissive_cors_policy>

SSH Server CBC Mode Ciphers Enabled

**Severity of Vulnerabilities**

**LOW**

**Affected IP**

verifacts.co.in

**Description**

The SSH server is configured to support Cipher Block Chaining (CBC) encryption. This may allow an attacker to recover the plain text message from the ciphertext. Note that this plugin only checks for the options of the SSH server and does not check for vulnerable software versions.

**Recommendations**

Contact the vendor or consult product documentation to disable CBC mode cipher encryption, and enable CTR or GCM cipher mode encryption.

**Reference**

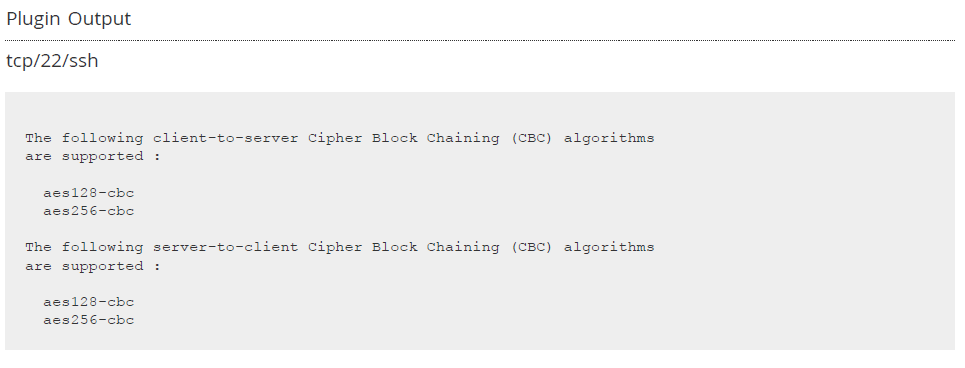
**CVE:**[CVE-2008-5161](https://www.tenable.com/cve/CVE-2008-5161)

**BID:**[32319](http://www.securityfocus.com/bid/32319)

**CWE:**[200](http://cwe.mitre.org/data/definitions/200)

**CERT:**[958563](http://www.kb.cert.org/vuls/id/958563)

**Output**

****

SSH Weak Key Exchange Algorithms Enabled

**Severity of Vulnerabilities**

**LOW**

**Affected IP**

verifacts.co.in

**Description**

The remote SSH server is configured to allow key exchange algorithms which are considered weak.  
  
This is based on the IETF draft document Key Exchange (KEX) Method Updates and Recommendations for Secure Shell (SSH) draft-ietf-curdle-ssh-kex-sha2-20. Section 4 lists guidance on key exchange algorithms that SHOULD NOT and MUST NOT be enabled. This includes:  
  
diffie-hellman-group-exchange-sha1  
  
diffie-hellman-group1-sha1  
  
gss-gex-sha1-\*  
  
gss-group1-sha1-\*  
  
gss-group14-sha1-\*  
  
rsa1024-sha1  
  
Note that this plugin only checks for the options of the SSH server, and it does not check for vulnerable software versions.

**Recommendations**

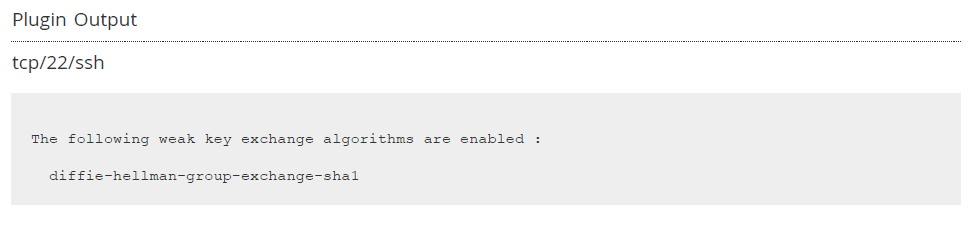
Contact the vendor or consult product documentation to disable the weak algorithms.

**Reference**

<https://wiki.mozilla.org/Security/Guidelines/OpenSSH>

<https://www.rfc-editor.org/rfc/rfc8732.html#name-deprecated-algorithms>

**Output**

****

Private IP Disclosure

**Severity of Vulnerabilities**

**LOW**

**Affected IP**

vibe.verifacts.co.in

vibe.verifacts.ph

**Description**

A private IP (such as 10.x.x.x, 172.x.x.x, 192.168.x.x) or an Amazon EC2 private hostname (for example, ip-10-0-56-78) has been found in the HTTP response body. This information might be helpful for further attacks targeting internal systems.

**Recommendations**

Remove the private IP address from the HTTP response body. For comments, use JSP/ASP/PHP comment instead of HTML/JavaScript comment which can be seen by client browsers.

**Reference**

<https://tools.ietf.org/html/rfc1918>

Application Error Disclosure

**Severity of Vulnerabilities**

**LOW**

**Affected IP**

vibe.verifacts.co.in

**Description**

This page contains an error/warning message that may disclose sensitive information like the location of the file that produced the unhandled exception. This information can be used to launch further attacks against the web application. The alert could be a false positive if the error message is found inside a documentation page.

**Recommendations**

Review the source code of this page. Implement custom error pages. Consider implementing a mechanism to provide a unique error reference/identifier to the client (browser) while logging the details on the server side and not exposing them to the user.

**Reference**

https://cwe.mitre.org/data/definitions/200.html

Cookie No HttpOnly Flag

**Severity of Vulnerabilities**

**LOW**

**Affected IP**

vibe.verifacts.co.in

**Description**

The HttpOnly flag directs compatible browsers to prevent client-side script from accessing cookies. Including the HttpOnly flag in the Set-Cookie HTTP response header helps mitigate the risk associated with Cross-Site Scripting (XSS) where an attacker's script code might attempt to read the contents of a cookie and exfiltrate information obtained. When set, browsers that support the flag will not reveal the contents of the cookie to a third party via client-side script executed via XSS.

**Recommendations**

Ensure that the HttpOnly flag is set for all cookies.

**Reference**

<https://owasp.org/www-community/HttpOnly>

Cookie Without Secure Flag

**Severity of Vulnerabilities**

**LOW**

**Affected IP**

vibe.verifacts.co.in

**Description**

The Secure attribute for sensitive cookies in HTTPS sessions is not set, which could cause the user agent to send those cookies in plaintext over an HTTP session.

**Recommendations**

Whenever a cookie contains sensitive information or is a session token, then it should always be passed using an encrypted channel. Ensure that the secure flag is set for cookies containing such sensitive information.

**Reference**

<https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/06-Session_Management_Testing/02-Testing_for_Cookies_Attributes.html>

Cookie without SameSite Attribute

**Severity of Vulnerabilities**

**LOW**

**Affected IP**

vibe.verifacts.co.in

**Description**

The SameSite attribute controls how cookies are sent for cross-domain requests. This attribute may have three values: 'Lax', 'Strict', or 'None'. If the 'None' value is used, a website may create a cross-domain POST HTTP request to another website, and the browser automatically adds cookies to this request. This may lead to Cross-Site-Request-Forgery (CSRF) attacks if there are no additional protections in place (such as Anti-CSRF tokens).

**Recommendations**

Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.

**Reference**

<https://tools.ietf.org/html/draft-ietf-httpbis-cookie-same-site>

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

**Severity of Vulnerabilities**

**LOW**

**Affected IP**

vibe.verifacts.co.in

**Description**

There are many different kinds of mistakes that introduce information exposures. The severity of the error can range widely, depending on the context in which the product operates, the type of sensitive information that is revealed, and the benefits it may provide to an attacker. Some kinds of sensitive information include:

* private, personal information, such as personal messages, financial data, health records, geographic location, or contact details
* system status and environment, such as the operating system and installed packages
* business secrets and intellectual property
* network status and configuration
* the product's own code or internal state
* metadata, e.g. logging of connections or message headers
* indirect information, such as a discrepancy between two internal operations that can be observed by an outsider

Information might be sensitive to different parties, each of which may have their own expectations for whether the information should be protected. These parties include:

* the product's own users
* people or organizations whose information is created or used by the product, even if they are not direct product users
* the product's administrators, including the admins of the system(s) and/or networks on which the product operates
* the developer

Information exposures can occur in different ways:

* the code **explicitly inserts**sensitive information into resources or messages that are intentionally made accessible to unauthorized actors, but should not contain the information - i.e., the information should have been "scrubbed" or "sanitized"
* a different weakness or mistake **indirectly inserts**the sensitive information into resources, such as a web script error revealing the full system path of the program.
* the code manages resources that intentionally contain sensitive information, but the resources are **unintentionally made accessible**to unauthorized actors. In this case, the information exposure is resultant - i.e., a different weakness enabled the access to the information in the first place.

It is common practice to describe any loss of confidentiality as an "information exposure," but this can lead to overuse of [CWE-200](https://cwe.mitre.org/data/definitions/200.html) in CWE mapping. From the CWE perspective, loss of confidentiality is a technical impact that can arise from dozens of different weaknesses, such as insecure file permissions or out-of-bounds read. [CWE-200](https://cwe.mitre.org/data/definitions/200.html) and its lower-level descendants are intended to cover the mistakes that occur in behaviors that explicitly manage, store, transfer, or cleanse sensitive information.

**Recommendations**

Ensure that your web server, application server, load balancer, etc. is configured to suppress "X-Powered-By" headers.

**Reference**

https://httpd.apache.org/docs/current/mod/core.html#servertokens

https://learn.microsoft.com/en-us/previous-versions/msp-n-p/ff648552(v=pandp.10)

https://www.troyhunt.com/shhh-dont-let-your-response-headers/

Server Leaks Version Information via "Server" HTTP Response Header Field

**Severity of Vulnerabilities**

**LOW**

**Affected IP**

vibe.verifacts.co.in

**Description**

There are many different kinds of mistakes that introduce information exposures. The severity of the error can range widely, depending on the context in which the product operates, the type of sensitive information that is revealed, and the benefits it may provide to an attacker. Some kinds of sensitive information include:

* private, personal information, such as personal messages, financial data, health records, geographic location, or contact details
* system status and environment, such as the operating system and installed packages
* business secrets and intellectual property
* network status and configuration
* the product's own code or internal state
* metadata, e.g. logging of connections or message headers
* indirect information, such as a discrepancy between two internal operations that can be observed by an outsider

Information might be sensitive to different parties, each of which may have their own expectations for whether the information should be protected. These parties include:

* the product's own users
* people or organizations whose information is created or used by the product, even if they are not direct product users
* the product's administrators, including the admins of the system(s) and/or networks on which the product operates
* the developer

Information exposures can occur in different ways:

* the code **explicitly inserts**sensitive information into resources or messages that are intentionally made accessible to unauthorized actors, but should not contain the information - i.e., the information should have been "scrubbed" or "sanitized"
* a different weakness or mistake **indirectly inserts**the sensitive information into resources, such as a web script error revealing the full system path of the program.
* the code manages resources that intentionally contain sensitive information, but the resources are **unintentionally made accessible**to unauthorized actors. In this case, the information exposure is resultant - i.e., a different weakness enabled the access to the information in the first place.

It is common practice to describe any loss of confidentiality as an "information exposure," but this can lead to overuse of [CWE-200](https://cwe.mitre.org/data/definitions/200.html) in CWE mapping. From the CWE perspective, loss of confidentiality is a technical impact that can arise from dozens of different weaknesses, such as insecure file permissions or out-of-bounds read. [CWE-200](https://cwe.mitre.org/data/definitions/200.html) and its lower-level descendants are intended to cover the mistakes that occur in behaviors that explicitly manage, store, transfer, or cleanse sensitive information.

**Recommendations**

Ensure that your web server, application server, load balancer, etc. is configured to suppress the "Server" header or provide generic details.

**Reference**

<https://httpd.apache.org/docs/current/mod/core.html#servertokens>

[https://learn.microsoft.com/en-us/previous-versions/msp-n-p/ff648552(v=pandp.10)](https://learn.microsoft.com/en-us/previous-versions/msp-n-p/ff648552%28v=pandp.10%29)

<https://www.troyhunt.com/shhh-dont-let-your-response-headers/>

Timestamp Disclosure - Unix

**Severity of Vulnerabilities**

**LOW**

**Affected IP**

verifacts.co.in

vibe.verifacts.co.in

vibe.verifacts.ph

**Description**

A timestamp was disclosed by the application/web server – Unix

**Recommendations**

Manually confirm that the timestamp data is not sensitive, and that the data cannot be aggregated to disclose exploitable patterns.

**Reference**

<https://cwe.mitre.org/data/definitions/200.html>

# Conclusion

Based on the External Vulnerability Assessment and Penetration Testing conducted, it was observed that vulnerabilities do exist within the scope. Reported vulnerabilities to be remediated within the timelines. CRITICAL & HIGH shall be closed in 30 days and MEDIUM & LOW to be closed in 60 days.