Sri Lanka Institute of Information Technology

Bug Bounty Report 09

onemoresmile.aswatson.com

IE2062 – Web Security

Submitted by:

IT22227836 – ADHIKARI A.M.V.B.M

Date of submission

2024.10.31

Table of Contents

Report 09 – onemoresmile.aswatson.com 3
..... 3
Vulnerability detected 4
Vulnerability 6
1. Title - Out-of-date Version (PHP) 6
Description 6
Affected components 7
Impact Assessment 8
Steps to reproduce 9
Proof of concept (if applicable) 10
Vulnerability scanning using Netsparker 10
Vulnerability finding using namp 11
Vulnerability finding using Nikto 12
Proposed mitigation or fix 12
2. Tittle - Out-of-date Version (WordPress) 14
Description 14
Impact Assessment 16
Steps to reproduce 17
Proof of concept (if applicable) 18
Vulnerability scanning using Netsparker 18
Vulnerability finding using namp 19
Vulnerability finding using Nikto 21
Proposed mitigation or fix 22

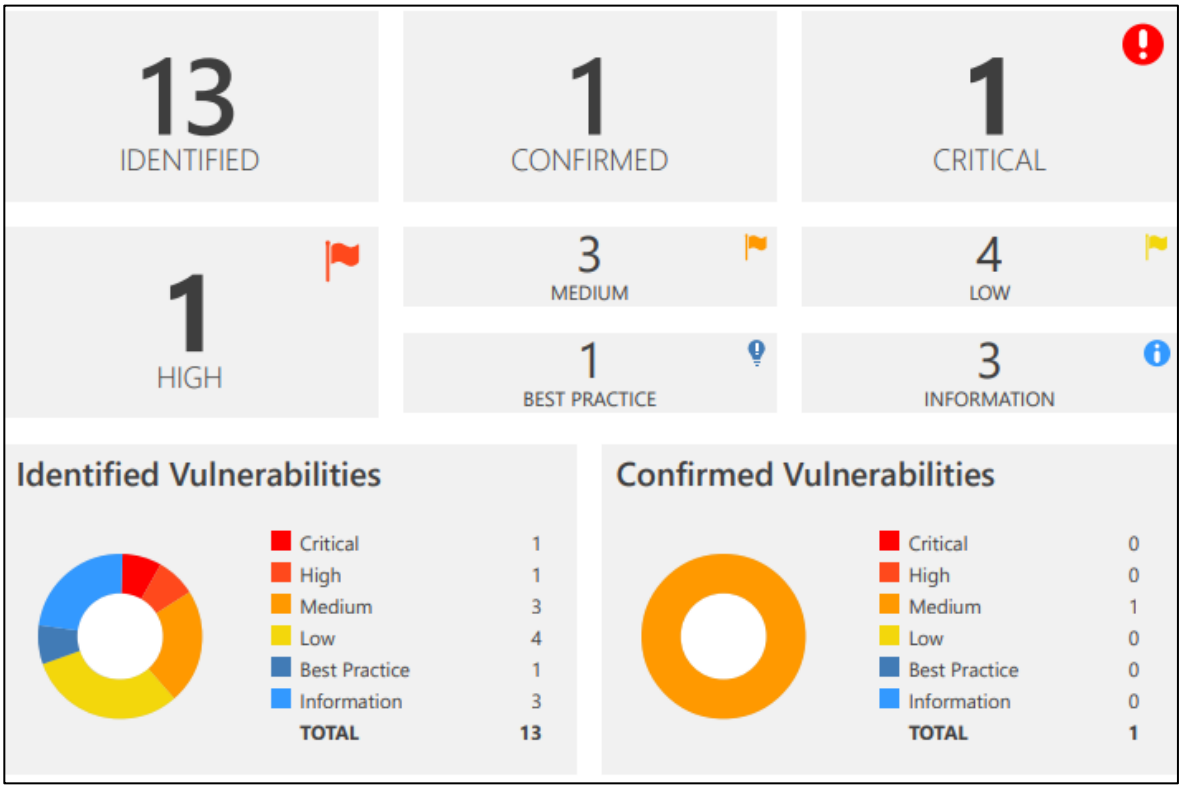
Report 09 – onemoresmile.aswatson.com

Main domain	https://www.aswatson.com/
Sub domain	onemoresmile.aswatson.com
IP address	20.239.60.58
platform	HackerOne

The A.S. Watson Group is the world's largest health and beauty retail group, with over 15,700 stores in 25 markets worldwide serving over 28 million customers per week, and over 3 billion customers and members.

A.S. Watson Group looks forward to working with the security community to discover vulnerabilities to keep our businesses and customers safe. As we operate in many different countries, we will be rolling out our bug bounty program in phases. Our focus within this rollout, is our retail websites (and soon, mobile apps on both Android and IOS)

Vulnerability detected

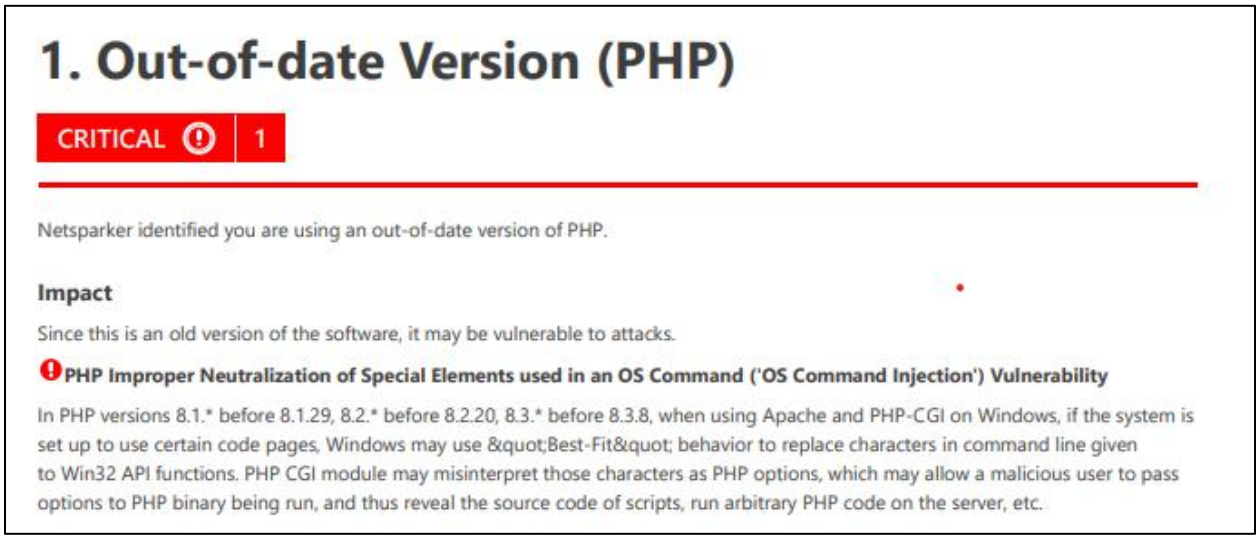


Vulnerabilities By OWASP 2017

CONFIRM	VULNERABILITY	METHOD	URL	SEVERITY
A3 - SENSITIVE DATA EXPOSURE				
	Weak Ciphers Enabled	GET	https://onemoresmile.aswatson.com/	MEDIUM
	HTTP Strict Transport Security (HSTS) Policy Not Enabled	GET	https://onemoresmile.aswatson.com/	MEDIUM
	Referrer-Policy Not Implemented	GET	https://onemoresmile.aswatson.com/cgi-sys/	BEST PRACTICE
A5 - BROKEN ACCESS CONTROL				
	[Possible] Cross-site Request Forgery	GET	https://onemoresmile.aswatson.com/hkcocktail/#gf_1	LOW
A6 - SECURITY MISCONFIGURATION				
	Missing X-Frame-Options Header	GET	https://onemoresmile.aswatson.com/cgi-sys/	LOW
	Version Disclosure (Nginx)	GET	https://onemoresmile.aswatson.com/	LOW
	Version Disclosure (PHP)	GET	https://onemoresmile.aswatson.com/	LOW
	.htaccess File Detected	GET	https://onemoresmile.aswatson.com/.htaccess	INFORMATION
A9 - USING COMPONENTS WITH KNOWN VULNERABILITIES				
	Out-of-date Version (PHP)	GET	https://onemoresmile.aswatson.com/	CRITICAL
	Out-of-date Version (WordPress)	GET	https://onemoresmile.aswatson.com/wp-includes/images/arrow-pointer-blue.png	HIGH
	Out-of-date Version (Nginx)	GET	https://onemoresmile.aswatson.com/	MEDIUM

Vulnerability

1. Title - Out-of-date Version (PHP)



- Risk – critical

Description

It uses the outdated version of PHP 8.0.30, which has been marked for multiple security vulnerabilities. Most importantly, it is vulnerable to CVE-2024-4577, relating to OS command injection due to insufficient input validation. This will let an attacker execute arbitrary commands on the server and totally compromise its integrity and security. CVE-2024-5458 also mentions that a weakness in the URL validation mechanism will finally let malicious payloads pass as valid. Such a vulnerability opens another exploitation avenue, giving remote access to data unauthorized individuals should never see.

This version no longer receives security updates since November 26, 2023, and thus presents a high risk to the application and sensitive data under it. Immediate action needs to be taken rather urgently to upgrade to a secure version of PHP to close these vulnerabilities and protect against such potential attacks.

Affected components

- **PHP Version:** The most significant contributor to this vulnerability is the very much obsolete version of PHP being used, 8.0.30. This is a critical version as it handles the server-side logic of the application and is in charge of processing user inputs, maintaining sessions, and interacting with the database.

- **Web Server:** The web server hosting the PHP application, like Apache or Nginx, will be another affected component. Poor configuration of server settings can further aggravate the vulnerabilities introduced by using an outdated version of PHP.

- **Application Code:** Any code interfacing with user inputs is vulnerable, especially those depending on the validation of URLs or execution of system commands. Application code vulnerabilities result in unauthorized server access and manipulation.

- **Database:** In case the application interacts with a database, any vulnerabilities in PHP code may further result in SQL injection attacks that can damage integrity of stored data.

- **Dependencies:** Consequently, any of these third-party libraries or frameworks depending on this vulnerable version of PHP will equally be subject to this same vulnerability.

In other words, the core application and its environment are directly hit by the vulnerable version of PHP, raising a high security risk due to many diverse components. Remediation needs to be urgently done in order to help protect those elements.

Impact Assessment

Surely, running an application on an outdated version of PHP8.0.30 poses serious security risks, with several potentially costly consequences for your application and your stakeholders. The impact can be categorized as follows:

- **Data Breach:** OS command injection vulnerabilities, such as CVE-2024-4577, and insufficient data validation, such as CVE-2024-5458, are the two most common means for gaining access to sensitive information. With such vulnerabilities, an intruder may extract personal information regarding payments or strategic business information, resulting in data leakage that may affect the users and the organization.

- **Service Disruption:** The exploitation of these kinds of issues may result in service disruptions due to denial-of-service conditions. This is because, when an attacker performs arbitrary commands on the server, normal application processing may be interfered with, thus making the system unavailable and unreliable.

- **Legal and Compliance Issues:** Vendors are therefore in a potentially sensitive position when vulnerabilities in their products disclose customer data. Breaches through disclosed vulnerabilities lead to violations of various data protection regulations, including GDPR, HIPAA, or PCI DSS. The aftermath may range from legal lawsuits against the organization to regulatory fines and loss of corporate reputation.

- **Reputation Damage:** Knowledge among the public about security breaches can extremely damage an organization's reputation. Due to this, users and clients cannot trust the security of an application, hence may lead to less user engagement, and further cause customer loss.

- **Financial Loss:** A security breach could result in a financial impact on an organization through remediation effort costs, legal fees, regulatory fines, and possible loss of business. This may also include costs related to similar activities aimed at restoring the organization's image through campaigns on public relations.

Running an out-of-date version of PHP in production will, therefore, continue to pose a higher risk to data integrity, application availability, and an organization's general trustworthiness.

Steps to reproduce

- **Identify the Target Application:**
  Access the application hosted at the URL: https://onemoresmile.aswatson.com/.

- **Check PHP Version:**
  Run a version check using a tool such as Netsparker, Nikto, or any web vulnerability scanner.

- **Validate Version Vulnerabilities:**
  Look up the version in the PHP release notes or security advisories to confirm version 8.0.30 is no longer a supported version and carries CVE-2024-4577 and CVE-2024-5458.

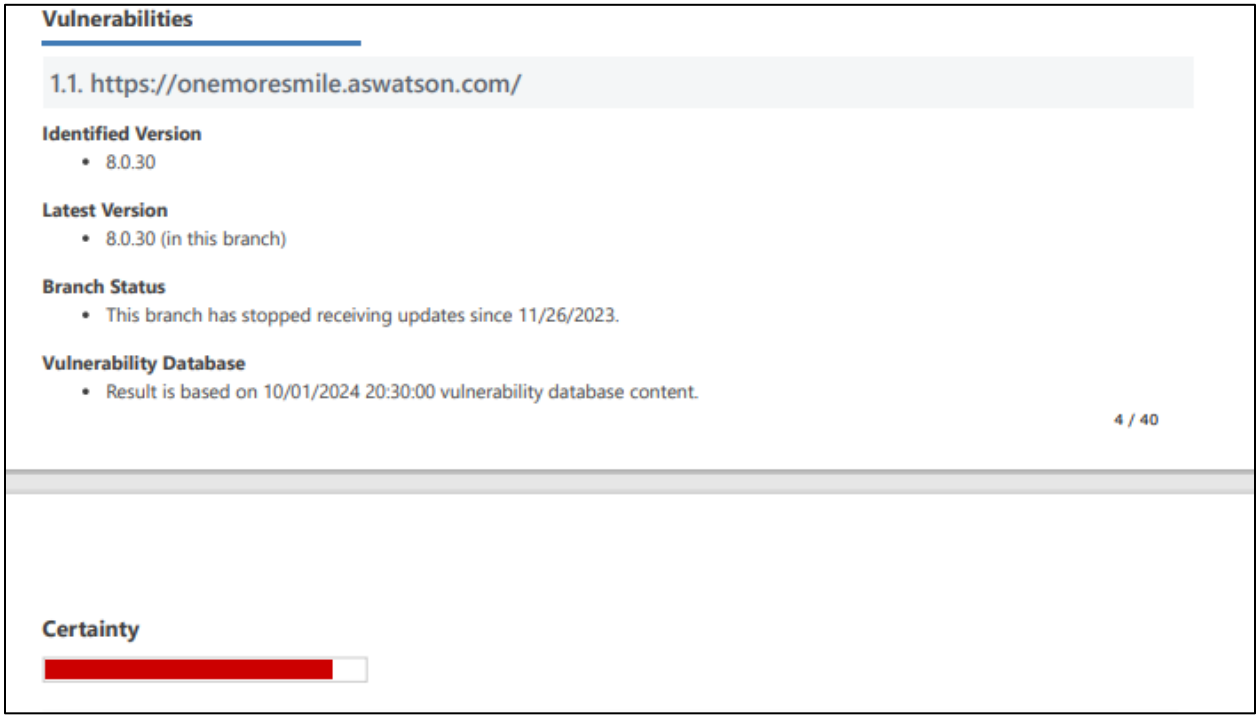
- **Testing for Specific Vulnerabilities:**
  Following the confirmation that the above vulnerabilities are from the identified version, proceed with further testing and exploit those identified:
  - For OS Command Injection,
  Try to bypass the application forms or URL parameters with crafted inputs to pass malicious commands to the server.
  - Lack of Input Validation:
  Pass URLs with malicious payloads in input fields to observe whether they get accepted and processed by the application.

- **Application Behavior:**
  Observe how the application reacts to injected commands or manipulated inputs.
  If successful, at this point, you must be able to confirm whether the application is vulnerable to command injection or not validating inputs properly.

- **Document findings:**
  Capture screenshots, logs, and other evidence that represent the vulnerabilities encountered; this includes very specific responses from the application that show security vulnerabilities. It would be easy to show, in this way, the presence of vulnerabilities related to the version of PHP being used, and to perform an effective assessment of the security posture of the application.

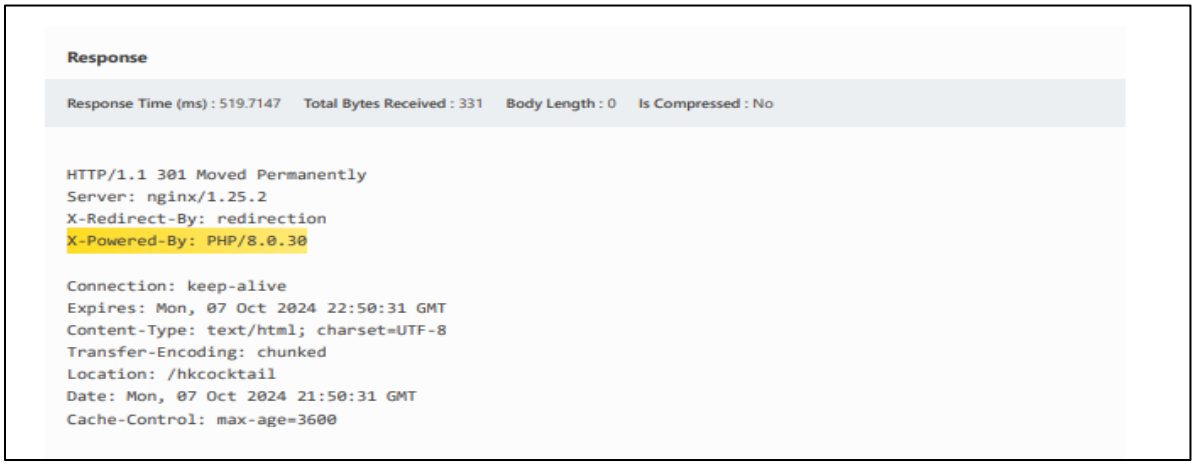
Proof of concept (if applicable)

Vulnerability scanning using Netsparker



- **Request**

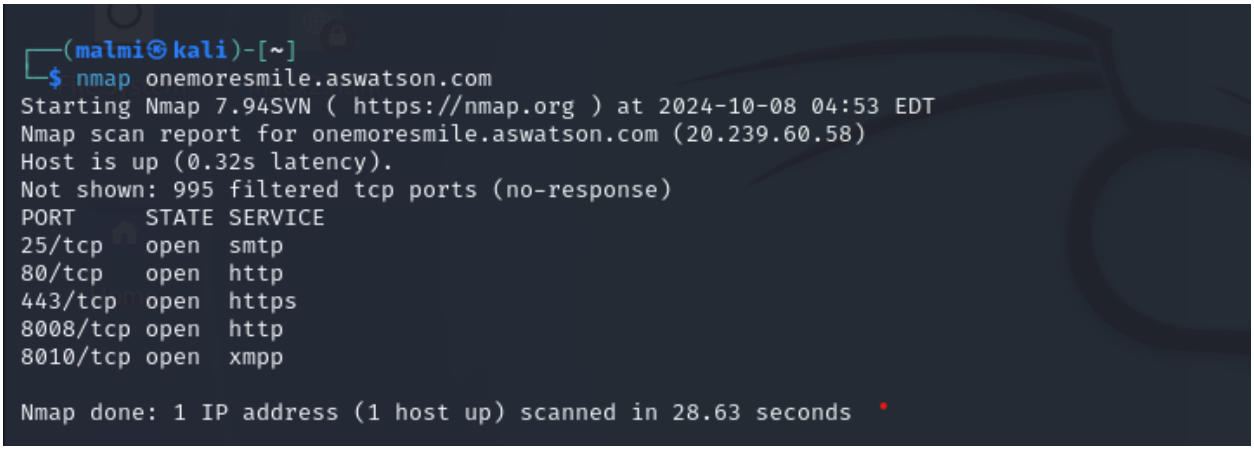
- **Response**



Vulnerability finding using namp

Used command

- nmap onemoresmile.aswatson.com



Founded open port

- 25/tcp open smtp
- 80/tcp open http
- 443/tcp open https
- 8008/tcp open http
- 8010/tcp open xmpp

Vulnerability finding using Nikto

Used command

- nikto -h https://onemoresmile.aswatson.com

Result

- The scan already shows that the server is running an outdated PHP version (8.0.30), which is vulnerable to specific known issues, such as:

    - **CVE-2024-4577**: Improper Neutralization of OS Command Injection.
    - **CVE-2024-5458**: Insufficient Verification of Data Authenticity in `filter_var`.

Proposed mitigation or fix

Remedy

Please upgrade your installation of PHP to the latest stable version.

- **PHP Version: Update**

Immediately upgrade to the latest stable version of PHP. The version should be fully supported, meaning security updates are active and regular. For the time being, go with versions beyond 8.0.30, such as PHP 8.1 or above.

- **Security Updates:**
  Create a schedule for regular security updates to PHP and other installed components, perform the upgrade. Install notification of new releases and publicity of vulnerabilities.

- **Code Review and Refactoring:**
  Perform a complete code review of the application and, to an even greater degree, scrutinize the area where the processing of user input is being handled. Ensure that all inputs are validating and sanitizing to prevent injection.
  Prepare your database queries using prepared statements. This inherently guards against SQL injection attacks.

- **Set Security Headers:**
  Configure the Web server to include security headers such as Content-Security-Policy, X-Content-Type-Options, and X-XSS-Protection.

- **Security Testing:**
  The application should implement regular vulnerability assessment and/or penetration testing to identify possible security weaknesses and address them.
  Security testing can, therefore, be automated with tools such as Netsparker, Burp Suite, or OWASP ZAP to ensure that vulnerabilities become caught before they could be exploited.

- **Educate Development Team:**
  Provide training to developers on secure coding practices and keeping software components up to date to minimize the risk of new vulnerabilities in future releases.

- **Planning for backup and recovery:**
  Develop robust backup and recovery plans that would restore the services immediately in case of a security incident or data loss. These mitigation strategies are rather simple, hence assisting an organization in drastically reducing the risk of vulnerabilities from outdated versions of PHP and improving the general security posture concerning web applications.

2. Title - Out-of-date Version (WordPress)

Description

The type of vulnerability identified in WordPress installed onhttps://onemoresmile.aswatson.com is Out-of-date Version. The version in issue, 6.3.1, is flagged because it includes security bugs vulnerable to attacks. Specifically, it has the following multiple vulnerabilities:

- **Cross-site Scripting:** This vulnerability, besides data theft, also allows an attacker to inject malicious scripts into the web pages viewed by clients. Such a success would have enabled him to hijack user sessions, deface websites, or redirect users to malware hosting sites.

- **Information Disclosure:** The old one may accidentally disclose information belonging to users, such as e-mail addresses and personally identifiable information, by default through insecurely secured APIs or when access controls are weak. Denial of Service: By using some of the vulnerabilities, the attackers can provoke conditions for denial of service, which will make the site unusable for its users.

- **Input Validation:** Weak input validation of user-supplied data may facilitate different types of injection-type attacks, which have the potential to severely disrupt the integrity of the site and its data.

Affected components

- **Core of WordPress:**
  - The most obvious is the core itself, specifically WordPress 6.3.1. A set of known vulnerabilities exists in this version; for example, cross-site scripting (XSS), poor input validation, and sensitive information exposure are various issues all arising from poor coding/architecture of WordPress itself. Plugins
  - While the core contains the main vulnerability, any installed plugins not updated may contain the indirect vulnerability as well. Plugins leveraging the WordPress core could inherit vulnerabilities or fail to mitigate the risks of an outdated CMS.

- **Themes:**
  Similarly, themes installed along with WordPress can also lead to a site's vulnerability. If the theme is not developed securely or if it invokes older functions of the core, then that can make your website more vulnerable
  .
- **Server Configuration:**
  Second, it may be the server environment in which the WordPress website runs that dictates the presentation of the vulnerability, such as Apache or Nginx. Poor settings within these servers can let security risks caused by out-of-date software worsen.

- **Database:**
  The database-which will most likely be MySQL or MariaDB-is where all your WordPress content resides and will also be at risk when an attacker uses certain vulnerabilities to inject malicious content or extract sensitive information.

Impact Assessment

The impact resulting from the use of an outdated version of WordPress 6.3.1 will be high for the following reasons.

- Security Vulnerabilities:
  The detected version includes several known security vulnerabilities such as Cross-Site Scripting, enabling attackers to perform malicious scripts in users' browsers. It could be used to perform unauthorized actions on behalf of users, steal certain data, or hijack sessions.

- **Data Exposure:**
  Through the sensitive information exposure vulnerabilities, the attackers would have unauthorized access to private user data that might include even email addresses and personal information of users, which can easily be used to commit identity theft or targeted phishing.

- **Reputation Damage:**
  A successful exploitation can result in loss of reputation for an organization by disengaging users' trust and possibly even losing any customers. Living in the digital age, incidents of security quickly scale to public relations crises. Financial Loss: These vulnerabilities contain potential financial losses due to downtime, recovery costs, possible legal liabilities, and the cost of notification to users affected.

- **Compliance Risk:**
  The usage of the very old version of the software may lead to its non-compliance with the set regulatory requirements such as PCI DSS and GDPR, consequently resulting in potential legal fines and loss of revenue. The identified risks of the outdated version of WordPress will impact the operational integrity, user trust, financial stability, or periodic updates of an organization**.**

Steps to reproduce

Before exploiting the vulnerability associated with the older version of WordPress, 6.3.1, first do the following:

- **Access Target URL:**
  Open the browser and reach the concerned website: https://onemoresmile.aswatson.com.

- **Check WordPress Version:**
  - Add /wp-includes/version.php to the base URL: https://onemoresmile.aswatson.com/wp-includes/version.php.
  - Look for it in the source code or use utility such as curl or wget to fetch this file.
  - Look for the variable called wp_version, which stores the WordPress version currently in use.

- **Use Vulnerability Scanning Tools:**
  - Run automated tools in a recon directory like Netsparker, WPScan, or Nikto against the target URL to scan for known vulnerabilities.
  -
- **Find Vulnerabilities:**
  - Check the Scan Report for any of the vulnerability reporting issues identified in the version of WordPress being deployed, 6.3.1, such as cross-site scripting (XSS), information exposure, or any other known vulnerabilities that relate to that version.

- **Validate exploits:**
  For any identified vulnerabilities, try to exploit them - when legally and explicitly authorized to do so.
  XSS via injecting a script in a comment or user input that may potentially view unsensitized outputs. Validate any information leakage using REST API endpoints, by trying to access sensitive information of the users.

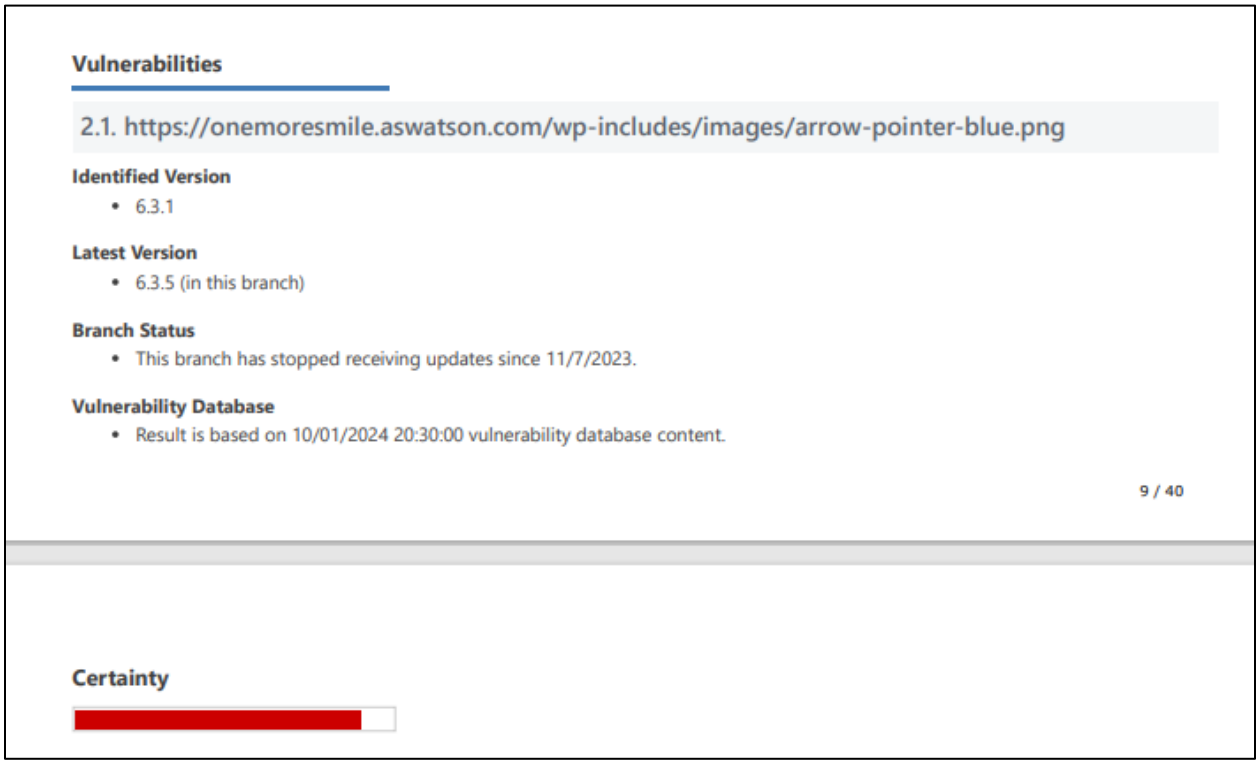
- **Review Results:**
  Review the results from your testing. Document in detail what was found if successful exploits or vulnerabilities were identified.

- **Compile Evidence:**
  You need to provide evidence that includes screenshots, logs, or the results of running the scanning tools showing the vulnerabilities existing on the old version.

These steps will provide evidence of the presence of vulnerabilities due to the very old version of WordPress that is in place, along with reproducible evidence of the findings

Proof of concept (if applicable)

Vulnerability scanning using Netsparker



Vulnerabilities

2.1. https://onemoresmile.aswatson.com/wp-includes/images/arrow-pointer-blue.png

Identified Version

- 6.3.1

Latest Version

- 6.3.5 (in this branch)

Branch Status

- This branch has stopped receiving updates since 11/7/2023.

Vulnerability Database

- Result is based on 10/01/2024 20:30:00 vulnerability database content.

9 / 40

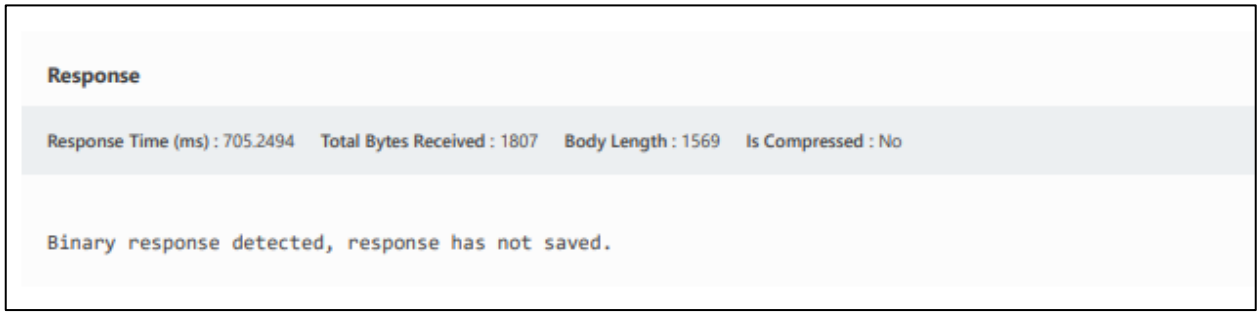
Certainty

- **Request**

Request

```
GET /wp-includes/images/arrow-pointer-blue.png HTTP/1.1
Host: onemoresmile.aswatson.com
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-us,en;q=0.5
Cache-Control: no-cache
User-Agent: Mozilla/5.0 (Windows NT 10.0; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.
77 Safari/537.36
X-Scanner: Netsparker
```

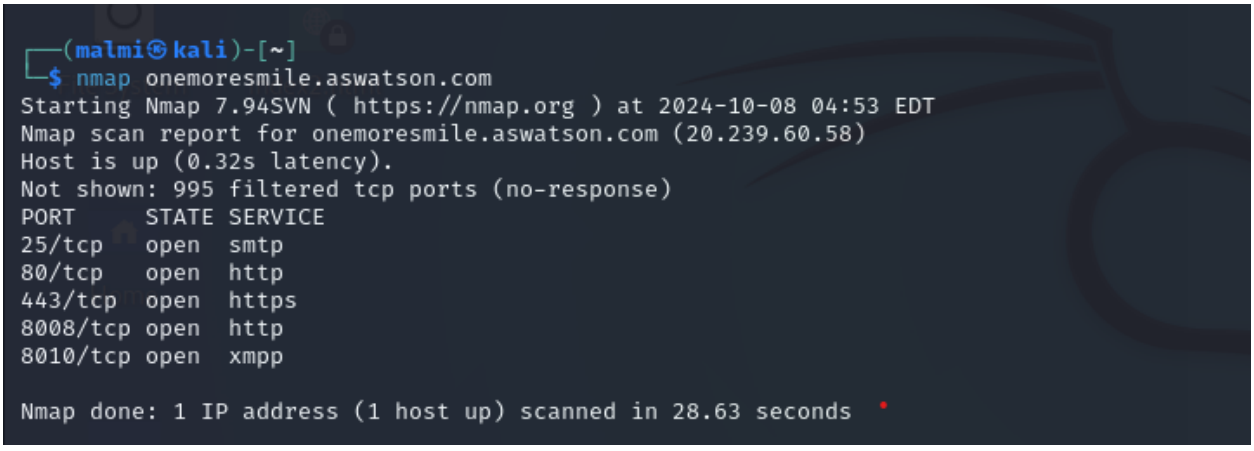
- **Response**

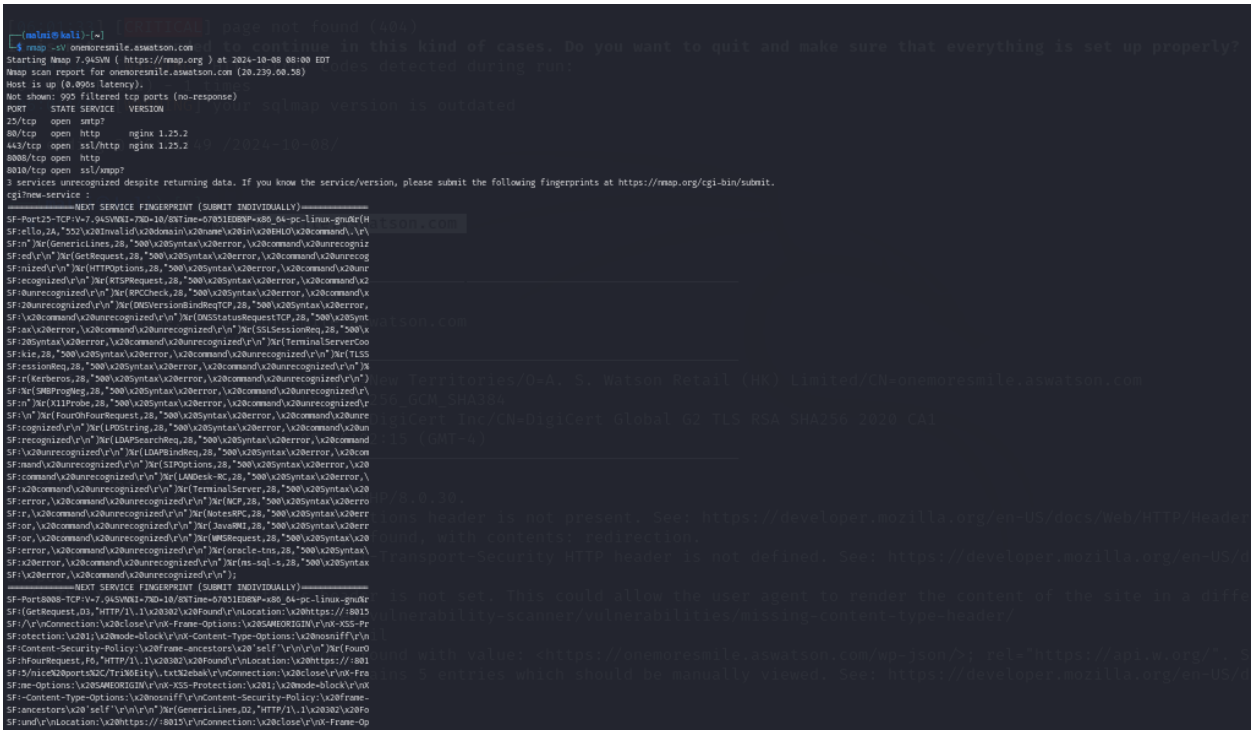


Vulnerability finding using namp

Used command

- nmap onemoresmile.aswatson.com
- nmap -sV onemoresmile.aswatson.com





```
┌──(malni㉿kali)-[~]
└─$ nmap -sV onemoresmile.aswatson.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-08 08:00 EDT
Nmap scan report for onemoresmile.aswatson.com (20.239.60.58)
Host is up (0.096s latency).
Not shown: 995 filtered tcp ports (no-response)
PORT     STATE SERVICE   VERSION
25/tcp   open  smtp?
80/tcp   open  http      nginx 1.25.2
443/tcp  open  ssl/http  nginx 1.25.2
8008/tcp open  http
8010/tcp open  ssl/xmpp?
3 services unrecognized despite returning data. If you know the service/version, please submit the following fingerprints at https://nmap.org/cgi-bin/submit.
cgi?new-service :
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port25-TCP:V=7.94SVN%I=7%D=10/8%Time=67051EDB%P=x86_64-pc-linux-gnu%r(H
SF:ello,2A,"552\x20Invalid\x20domain\x20name\x20in\x20EHLO\x20command\.\r\
SF:n")%r(GenericLines,28,"500\x20Syntax\x20error,\x20command\x20unrecogniz
SF:ed\r\n")%r(GetRequest,28,"500\x20Syntax\x20error,\x20command\x20unrecog
SF:nized\r\n")%r(HTTPOptions,28,"500\x20Syntax\x20error,\x20command\x20unr
SF:ecognized\r\n")%r(RTSPRequest,28,"500\x20Syntax\x20error,\x20command\x2
SF:0unrecognized\r\n")%r(RPCCheck,28,"500\x20Syntax\x20error,\x20command\x
SF:20unrecognized\r\n")%r(DNSVersionBindReqTCP,28,"500\x20Syntax\x20error,
SF:\x20command\x20unrecognized\r\n")%r(DNSStatusRequestTCP,28,"500\x20Synt
SF:ax\x20error,\x20command\x20unrecognized\r\n")%r(SSLSessionReq,28,"500\x
SF:20Syntax\x20error,\x20command\x20unrecognized\r\n")%r(TerminalServerCoo
SF:kie,28,"500\x20Syntax\x20error,\x20command\x20unrecognized\r\n")%r(TLS
SF:essionReq,28,"500\x20Syntax\x20error,\x20command\x20unrecognized\r\n")%
SF:r(Kerberos,28,"500\x20Syntax\x20error,\x20command\x20unrecognized\r\n")
SF:%r(SMBProgNeg,28,"500\x20Syntax\x20error,\x20command\x20unrecognized\r\
SF:n")%r(X11Probe,28,"500\x20Syntax\x20error,\x20command\x20unrecognized\r
SF:\n")%r(FourOhFourRequest,28,"500\x20Syntax\x20error,\x20command\x20unre
SF:cognized\r\n")%r(LPDString,28,"500\x20Syntax\x20error,\x20command\x20un
SF:recognized\r\n")%r(LDAPSearchReq,28,"500\x20Syntax\x20error,\x20command
SF:\x20unrecognized\r\n")%r(LDAPBindReq,28,"500\x20Syntax\x20error,\x20com
SF:mand\x20unrecognized\r\n")%r(SIPOptions,28,"500\x20Syntax\x20error,\x20
SF:command\x20unrecognized\r\n")%r(LANDesk-RC,28,"500\x20Syntax\x20error,\
SF:x20command\x20unrecognized\r\n")%r(TerminalServer,28,"500\x20Syntax\x20
SF:error,\x20command\x20unrecognized\r\n")%r(NCP,28,"500\x20Syntax\x20erro
SF:r,\x20command\x20unrecognized\r\n")%r(NotesRPC,28,"500\x20Syntax\x20err
SF:or,\x20command\x20unrecognized\r\n")%r(JavaRMI,28,"500\x20Syntax\x20err
SF:or,\x20command\x20unrecognized\r\n")%r(WMSRequest,28,"500\x20Syntax\x20
SF:error,\x20command\x20unrecognized\r\n")%r(oracle-tns,28,"500\x20Syntax\
SF:x20error,\x20command\x20unrecognized\r\n")%r(ms-sql-s,28,"500\x20Syntax
SF:\x20error,\x20command\x20unrecognized\r\n");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port8008-TCP:V=7.94SVN%I=7%D=10/8%Time=67051EDB%P=x86_64-pc-linux-gnu%r
SF:(GetRequest,83,"HTTP/1\.1\x20302\x20Found\r\nLocation:\x20https://:8015
SF:/\r\nConnection:\x20close\r\nX-Frame-Options:\x20SAMEORIGIN\r\nX-XSS-Pr
SF:otection:\x201;\x20mode=block\r\nX-Content-Type-Options:\x20nosniff\r\n
SF:Content-Security-Policy:\x20frame-ancestors\x20'self'\r\n\r\n")%r(Four0
SF:hFourRequest,F6,"HTTP/1\.1\x20302\x20Found\r\nLocation:\x20https://:801
SF:5/nice%20ports%2C/Tri%6Eity\.txt%2ebak\r\nConnection:\x20close\r\nX-Fra
SF:me-Options:\x20SAMEORIGIN\r\nX-XSS-Protection:\x201;\x20mode=block\r\nX
SF:-Content-Type-Options:\x20nosniff\r\nContent-Security-Policy:\x20frame-
SF:ancestors\x20'self'\r\n\r\n")%r(GenericLines,D2,"HTTP/1\.1\x20302\x20Fo
SF:und\r\nLocation:\x20https://:8015\r\nConnection:\x20close\r\nX-Frame-Op
```

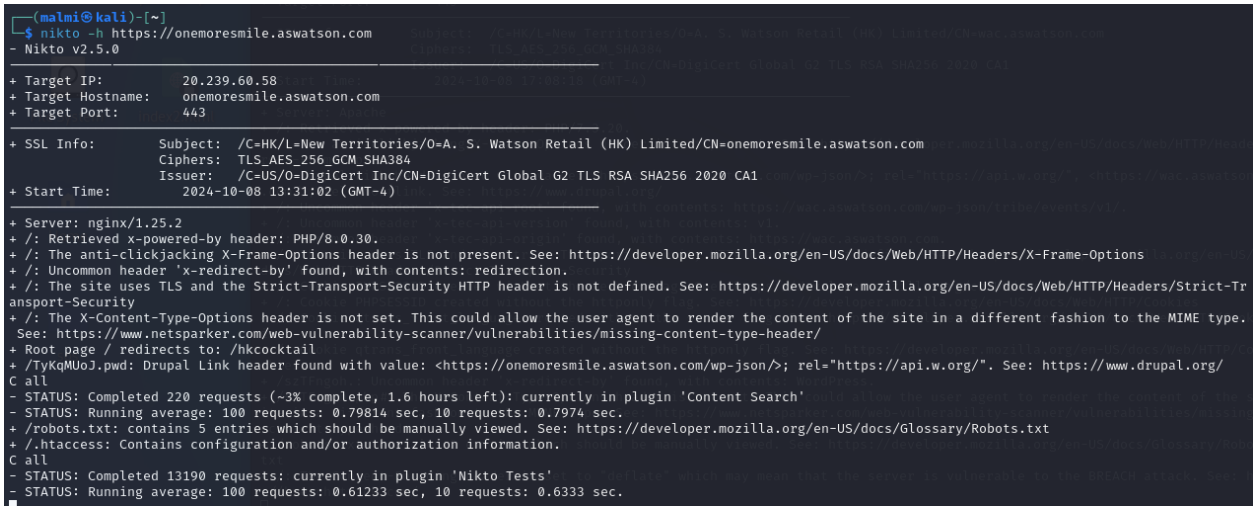
```
SF:-Content-Type-Options:\x20nosniff\r\nContent-Security-Policy:\x20frame-
SF:ancestors\x20'self'\r\n\r\n")%r(GenericLines,D2,"HTTP/1\.1\x20302\x20Fo
SF:und\r\nLocation:\x20https://:8015\r\nConnection:\x20close\r\nX-Frame-Op
SF:tions:\x20SAMEORIGIN\r\nX-XSS-Protection:\x201;\x20mode-block\r\nX-Cont
SF:ent-Type-Options:\x20nosniff\r\nContent-Security-Policy:\x20frame-ances
SF:tors\x20'self'\r\n\r\n")%r(HTTPOptions,D2,"HTTP/1\.1\x20302\x20Found\r\
SF:nLocation:\x20https://:8015\r\nConnection:\x20close\r\nX-Frame-Options:
SF:\x20SAMEORIGIN\r\nX-XSS-Protection:\x201;\x20mode-block\r\nX-Content-Ty
SF:pe-Options:\x20nosniff\r\nContent-Security-Policy:\x20frame-ancestors\x
SF:20'self'\r\n\r\n")%r(RTSPRequest,D2,"HTTP/1\.1\x20302\x20Found\r\nLocat
SF:ion:\x20https://:8015\r\nConnection:\x20close\r\nX-Frame-Options:\x20SA
SF:MEORIGIN\r\nX-XSS-Protection:\x201;\x20mode-block\r\nX-Content-Type-Opt
SF:ions:\x20nosniff\r\nContent-Security-Policy:\x20frame-ancestors\x20'sel
SF:f'\r\n\r\n")%r(SIPOptions,D2,"HTTP/1\.1\x20302\x20Found\r\nLocation:\x2
SF:0https://:8015\r\nConnection:\x20close\r\nX-Frame-Options:\x20SAMEORIGI
SF:N\r\nX-XSS-Protection:\x201;\x20mode-block\r\nX-Content-Type-Options:\x
SF:20nosniff\r\nContent-Security-Policy:\x20frame-ancestors\x20'self'\r\n\
SF:r\n");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port8010-TCP:V=7.94SVN%T=SSL%I=7%D=10/8%Time=67051EEC%P=x86_64-pc-linux
SF:-gnu%r(GenericLines,BBD,"HTTP/1\.1\x20200\x20OK\r\nContent-Length:\x202
SF:736\r\nConnection:\x20close\r\nCache-Control:\x20no-cache\r\nContent-Ty
SF:pe:\x20text/html;\x20charset=utf-8\r\nX-Frame-Options:\x20SAMEORIGIN\r\
SF:nX-XSS-Protection:\x201;\x20mode-block\r\nX-Content-Type-Options:\x20no
SF:sniff\r\nContent-Security-Policy:\x20frame-ancestors\x20'self'\r\n\r\n<
SF:!DOCTYPE\x20html><html\x20lang=\"en\">\x20<head>\x20<meta\x20charset=\"
SF:UTF-8\">\x20<meta\x20http-equiv=\"X-UA-Compatible\"\x20content=\"IE=8;\
SF:x20IE=EDGE\">\x20<meta\x20name=\"viewport\"\x20content=\"width=device-w
SF:idth,\x20initial-scale=1\">\x20<style\x20type=\"text/css\">\x20body\x20
SF:{\x20height:\x20100%;\x20font-family:\x20Helvetica,\x20Arial,\x20sans-s
SF:erif;\x20color:\x20#6a6a6a;\x20margin:\x200;\x20display:\x20flex;\x20al
SF:ign-items:\x20center;\x20justify-content:\x20center;\x20}\x20input\[typ
SF:e=date\],\x20input\[type=email\],\x20input\[type=number\],\x20input\[ty
SF:pe=password\],\x20input\[type=search\],\x20input\[type=tel\],\x20input\
SF:[type=text\],\x20input\[type=time\],\x20input\[type=url\],\x20select,\x
SF:20textarea\x20{\x20color:\x20#262626;\x20vertical-align:\x20baseline;\x
SF:20margin:\x20\.2em;\x20border-style:\x20solid}")%r(GetRequest,BBD,"HTTP/1
SF:\.1\x20200\x20OK\r\nContent-Length:\x202736\r\nConnection:\x20close\r\n
SF:Cache-Control:\x20no-cache\r\nContent-Type:\x20text/html;\x20charset=ut
SF:f-8\r\nX-Frame-Options:\x20SAMEORIGIN\r\nX-XSS-Protection:\x201;\x20mod
SF:e=block\r\nX-Content-Type-Options:\x20nosniff\r\nContent-Security-Polic
SF:y:\x20frame-ancestors\x20'self'\r\n\r\n<!DOCTYPE\x20html><html\x20lang=
SF:\"en\">\x20<head>\x20<meta\x20charset=\"UTF-8\">\x20<meta\x20http-equiv
SF:=\"X-UA-Compatible\"\x20content=\"IE=8;\x20IE=EDGE\">\x20<meta\x20name=
SF:\"viewport\"\x20content=\"width=device-width,\x20initial-scale=1\">\x20
SF:<style\x20type=\"text/css\">\x20body\x20{\x20height:\x20100%;\x20font-f
SF:amily:\x20Helvetica,\x20Arial,\x20sans-serif;\x20color:\x20#6a6a6a;\x20
SF:margin:\x200;\x20display:\x20flex;\x20align-items:\x20center;\x20justif
SF:y-content:\x20center;\x20}\x20input\[type=date\],\x20input\[type=email\
SF:],\x20input\[type=number\],\x20input\[type=password\],\x20input\[type=s
SF:earch\],\x20input\[type=tel\],\x20input\[type=text\],\x20input\[type=ti
SF:me\],\x20input\[type=url\],\x20select,\x20textarea\x20{\x20color:\x20#2
SF:62626;\x20vertical-align:\x20baseline;\x20margin:\x20\.2em;\x20border-s
SF:tyle:\x20solid");

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 192.63 seconds
```

Founded open port

- 25/tcp open smtp
- 80/tcp open http
- 443/tcp open https
- 8008/tcp open http
- 8010/tcp open xmpp

Vulnerability finding using Nikto

- **Used command**
  - nikto -h onemoresmile.aswatson.com



- **Results**
  - **Missing Security Headers**: The anti-clickjacking X-Frame-Options header is not present.
  - **X-Content-Type-Options Header**: The X-Content-Type-Options header is not set.

Proposed mitigation or fix

Remedy

Please upgrade your installation of WordPress to the latest stable version.

The following mitigations may be applied to help reduce this vulnerability with the version of WordPress used, 6.3.1, and further enhance the security posture of your website:

- **Upgrade WordPress to the Latest Stable Version:**
  Immediately upgrade WordPress to the latest stable version, which is currently 6.3.5, so it can patch known vulnerabilities and thus ensure that the website automatically benefits from security fixes and enhancements provided by the WordPress development team.

- **Perform Regular Updates:**
  Establish a schedule for frequent updating of WordPress, along with themes and plugins. It prevents exploits resulting from the existence of any vulnerabilities.

- **Security Plugins:**
  Deploy security plugins that are known to be top-rated, such as Wordfence or Sucuri. These offers added layers of security through firewall protection, malware scanning, and intrusion detection.

- **Monitor and Update Plugins and Themes:**
  Keep all installed plugins and themes updated to the latest versions. Remove any plugins not in use or those that are deprecated, as doing so reduces the attack surface.

- **Backup Process:**
  Regular backup of your website; this would allow you to revert your site instantly at any time should there be a compromise.

- **Limit User Privileges:**
  Verify the user roles and permissions of those who can have administrative access to the WordPress dashboard.

- **Set up security headers:**
  Content Security Policy, X-Frame-Options, and X-Content-Type-Options will help mitigate common web vulnerabilities, such as XSS and clickjacking.

- **Perform Security Audits:**
  Regular security audits and active vulnerability scanning should be run to identify and fix emerging threats. Professional security services are highly recommended to check these risks professionally.

Based on these steps, the website would minimize its exploitability vulnerability, caused by running software that is out of date, and enhance security posture at the website.