Sri Lanka Institute of Information Technology

Bug Bounty Report 05

aswi-op01t.aswatson.com

IE2062 – Web Security

Submitted by:

IT22227836 – ADHIKARI A.M.V.B.M

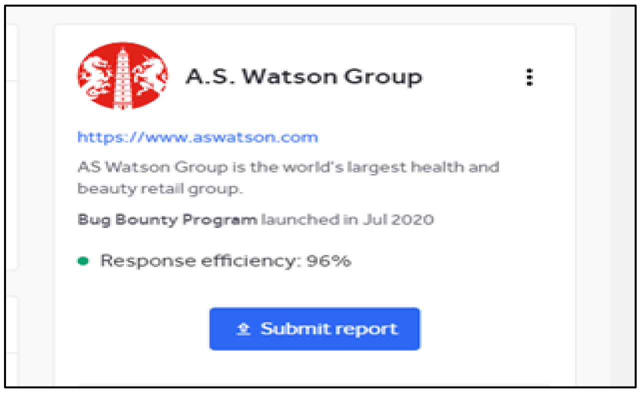
Date of submission

2024.10.31

Table of Contents

Report 05 – aswi-op01t.aswatson.com

Main domain	https://www.aswatson.com/
Sub domain	aswi-op01t.aswatson.com
IP address	140.238.64.202
platform	HackerOne


A.S. Watson Group

https://www.aswatson.com

AS Watson Group is the world's largest health and beauty retail group.

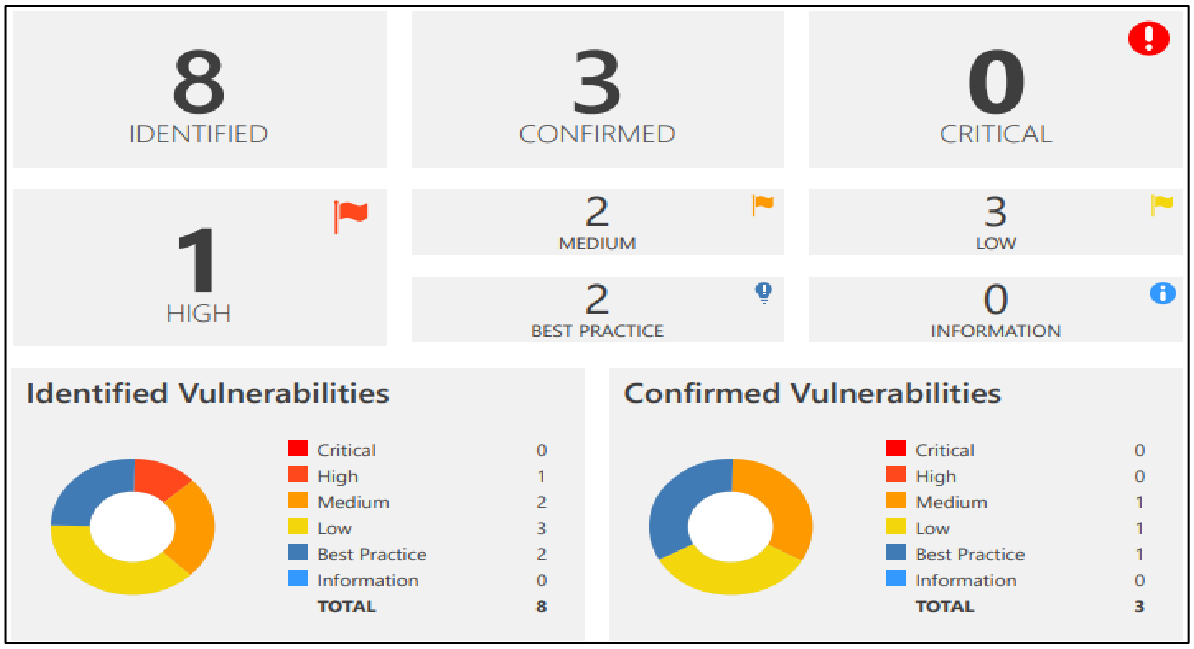
Bug Bounty Program launched in Jul 2020

● Response efficiency: 96%

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



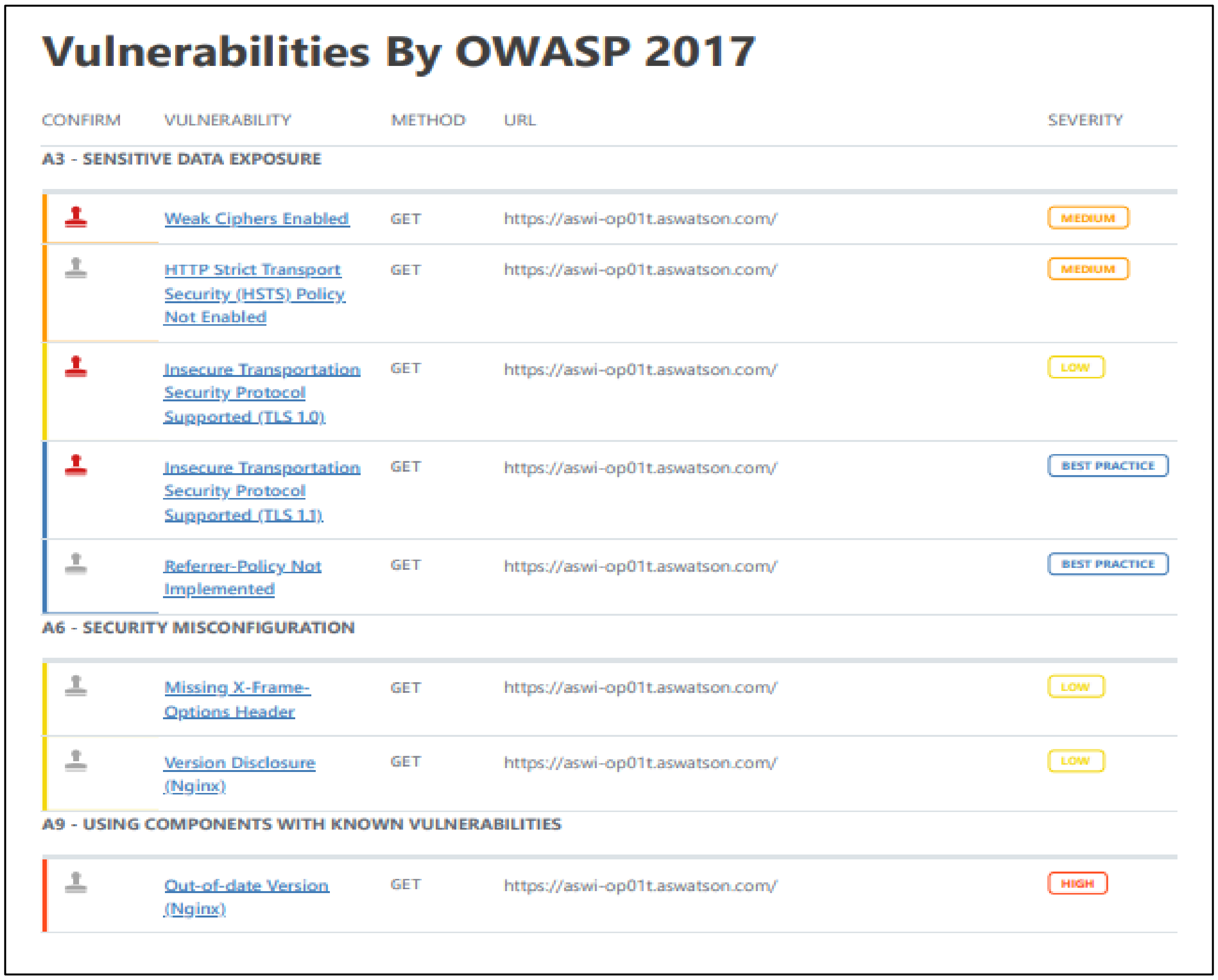
8 IDENTIFIED	3 CONFIRMED	0 CRITICAL
1 HIGH	2 MEDIUM	3 LOW
	2 BEST PRACTICE	0 INFORMATION

Identified Vulnerabilities

🟥	Critical	0
🟧	High	1
🟧	Medium	2
🟨	Low	3
🟦	Best Practice	2
🟦	Information	0
	TOTAL	**8**

Confirmed Vulnerabilities

🟥	Critical	0
🟧	High	0
🟧	Medium	1
🟨	Low	1
🟦	Best Practice	1
🟦	Information	0
	TOTAL	**3**

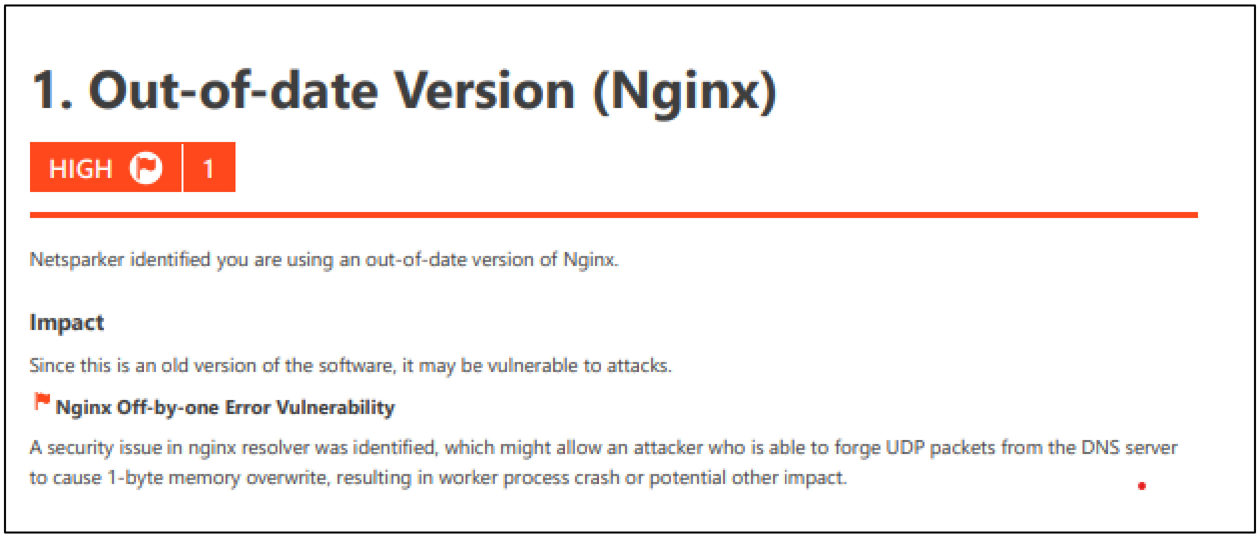


Vulnerabilities By OWASP 2017

CONFIRM	VULNERABILITY	METHOD	URL	SEVERITY
A3 - SENSITIVE DATA EXPOSURE				
✓	Weak Ciphers Enabled	GET	https://aswi-op01t.aswatson.com/	MEDIUM
	HTTP Strict Transport Security (HSTS) Policy Not Enabled	GET	https://aswi-op01t.aswatson.com/	MEDIUM
✓	Insecure Transportation Security Protocol Supported (TLS 1.0)	GET	https://aswi-op01t.aswatson.com/	LOW
✓	Insecure Transportation Security Protocol Supported (TLS 1.1)	GET	https://aswi-op01t.aswatson.com/	BEST PRACTICE
	Referrer-Policy Not Implemented	GET	https://aswi-op01t.aswatson.com/	BEST PRACTICE
A6 - SECURITY MISCONFIGURATION				
	Missing X-Frame-Options Header	GET	https://aswi-op01t.aswatson.com/	LOW
	Version Disclosure (Nginx)	GET	https://aswi-op01t.aswatson.com/	LOW
A9 - USING COMPONENTS WITH KNOWN VULNERABILITIES				
	Out-of-date Version (Nginx)	GET	https://aswi-op01t.aswatson.com/	HIGH

Vulnerability

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



1. Out-of-date Version (Nginx)

HIGH | 1

Netsparker identified you are using an out-of-date version of Nginx.

Impact

Since this is an old version of the software, it may be vulnerable to attacks.

Nginx Off-by-one Error Vulnerability

A security issue in nginx resolver was identified, which might allow an attacker who is able to forge UDP packets from the DNS server to cause 1-byte memory overwrite, resulting in worker process crash or potential other impact.

- Risk – High

Description

The web server at https://aswi-op01t.aswatson.com/, running Nginx version 1.18.0, has not been updated since April 20, 2021. This version is very outdated, leaving the server open to known vulnerabilities for exploitation by attackers. Of essence, the Nginx team has stopped updating this branch, increasing the possibility of security breaches.

Running obsolete software creates many opportunities for security-related incidents to occur, including DoS attacks intended to bring it down, unauthorized access, and other data breaches. This should be upgraded to the latest stable version if the security and integrity of the server and its data are to be maintained.

Affected components

The affected components by Out-of-date Version (Nginx) are:

- **Nginx Web Server:** the central one being the Nginx web server, responsible for processing HTTP requests and web content serving. The exact version in question, 1.18.0, has not received security updates since April 2021, hence it is exposed to a certain set of known vulnerabilities.

- **Web Applications:** This would apply to web applications running on an Nginx server. Such applications may be exposed, too. Running outdated software opens these applications to known attacks against the web server.

- **Configuration in Server:** Configuration files on the server may include security settings that are outdated or ineffective due to the unavailability of updates to a version of Nginx in use. That might result in wrong handling of requests or failure to respond correctly against an attack.

- **Operating System Underlying the Setup:** Although not directly an issue of Nginx per se, the operating system upon which the Nginx server is run may also be open to vulnerabilities, should it depend on outdated libraries or other dependencies in order to interface with Nginx.

- **Network Infrastructure:** Other network devices around the server, including firewalls or load balancers, are also indirectly vulnerable if their security policies depend on server configurations.

Therefore, addressing the outdated version of Nginx is highly critical in ensuring the integrity, availability, and confidentiality of web applications or services hosted on this server.

Impact Assessment

Increased Exposure to Vulnerability: Running an older version of Nginx exposes a server to known vulnerabilities that can be leveraged by an attacker. This includes critical issues like CVE-2021-23017, which can result in memory overwrites.

- **Denial of Service:** Vulnerabilities in older versions may provide a means for Denial-of-Service attacks, such as CVE-2023-44487, whereby the attacker may forcefully reset HTTP/2 streams, possibly leading to resource starvation of the server and denial of service.

- **Security Breaches:** Out-of-date software lacks critical security patches, since it opens the door wide for Man-in-the-Middle attacks, like CVE-2021-3618, against integrity and authentication.

- **Non-compliance Issues:** Running non-supported software will result in compliance issues such as with standards like PCI DSS, HIPAA, and ISO27001, resulting in fines and damage to reputation.

- **Loss of Trust:** Users may be very skeptical about using applications running on old servers, given the fear of data breaches; this could place a dent in building trust among customers.

- **Operational Risks:** There is a chance of unexpected downtime and also more remediation costs with continued use of outdated versions. Hence, timely upgrade is very important to ensure safety and stability.

In general, using an older version of Nginx makes quite a difference in terms of security and compliance, thereby impacting user confidence. Therefore, upgrading becomes critical.

Steps to reproduce

- **Access Target URL:**

  Open the web browser and type target application URL, for example, https://aswi-op01t.aswatson.com/, to check whether it is accessible.

- **Identify the Server Software:**

  Employ network scanning tools, like Nmap, to identify the software version of the server. In the scan results, identify the version of Nginx.

- **Check the Status of the Version:**

  Match the actual detected version, 1.18.0, against the official documentation or changelog for Nginx, to validate whether it is a truly outdated version that no longer receives updates.

- **Vulnerability Research:**

  Based on your knowledge, research known vulnerabilities specific to the version of Nginx using CVE databases or vulnerability management platforms.

- **Document Findings:**

  List version information, accompanied by any related vulnerabilities and associated application risks.

- **Gather Evidence:**

  Capture screenshots of version details and vulnerability information supporting this report.

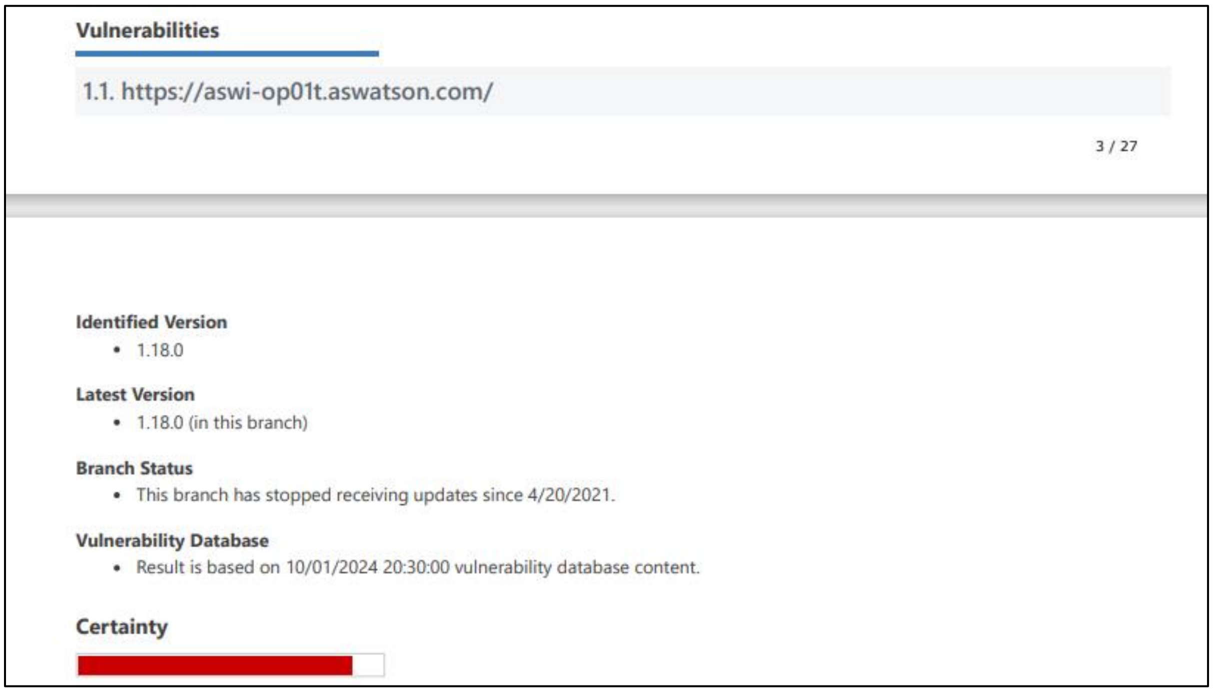
- **Recommendations:**

  Recommend upgrading Nginx to the most recent stable version to eliminate some risks.

  This simple procedure will lend a hand in depicting that the obsolete version of Nginx is present alongside

Proof of concept (if applicable)

Vulnerability scanning using Netsparker



Vulnerabilities

1.1. https://aswi-op01t.aswatson.com/

3 / 27

Identified Version

- 1.18.0

Latest Version

- 1.18.0 (in this branch)

Branch Status

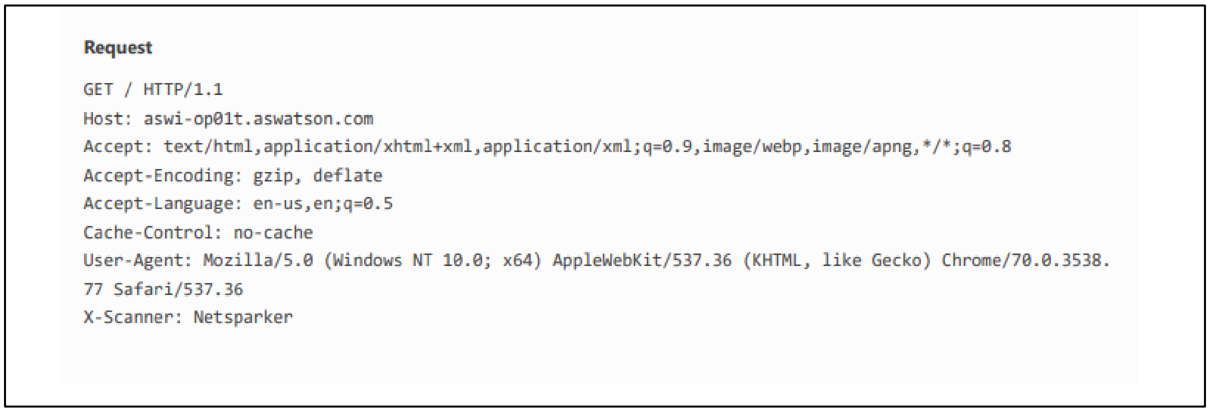
- This branch has stopped receiving updates since 4/20/2021.

Vulnerability Database

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

Certainty

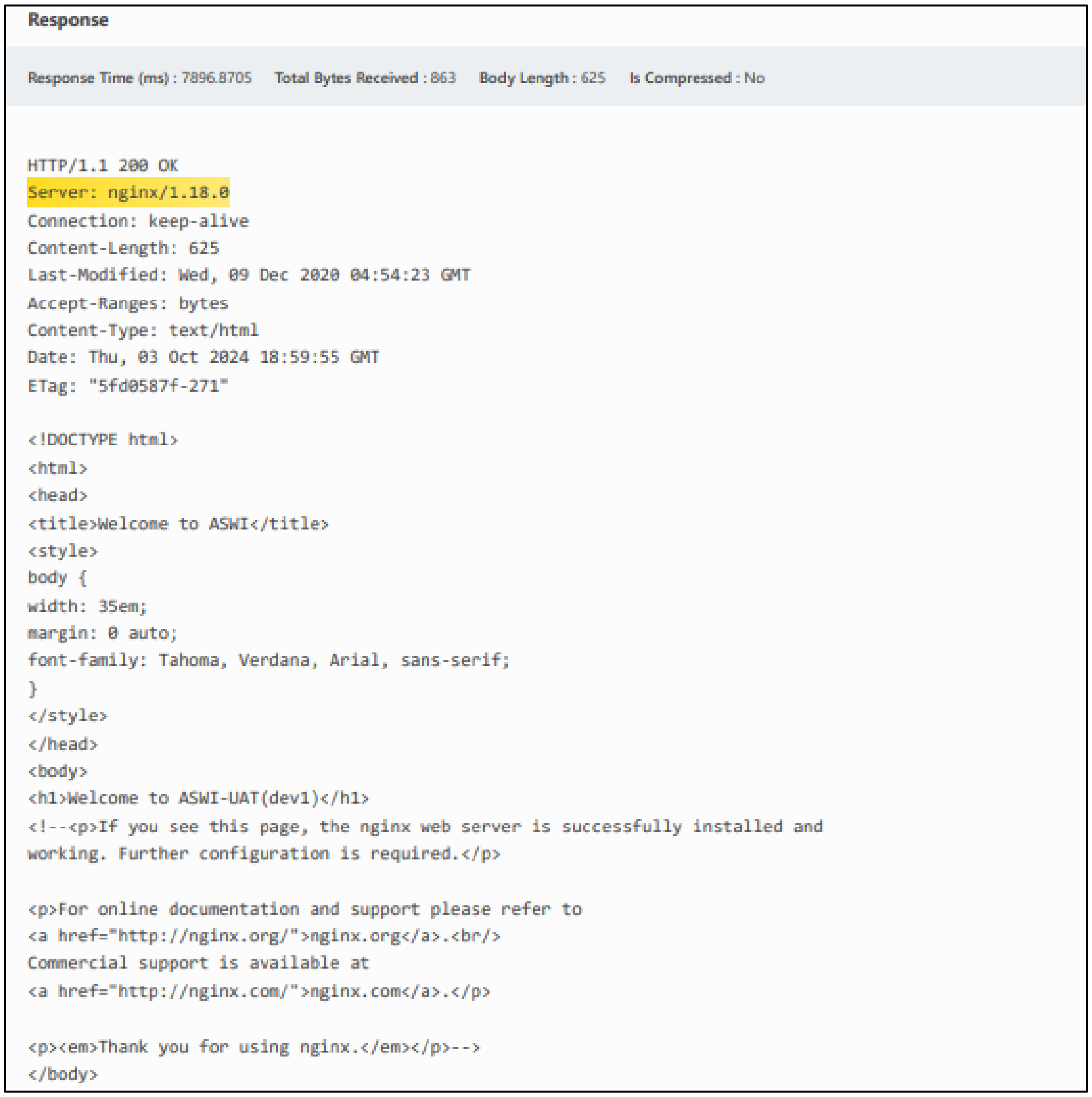
- **Request**



Request

```
GET / HTTP/1.1
Host: aswi-op01t.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**



Response

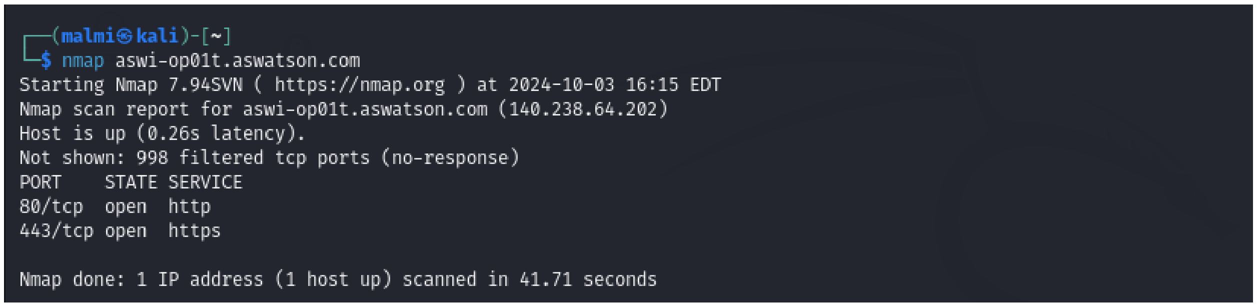
Response Time (ms) : 7896.8705 Total Bytes Received : 863 Body Length : 625 Is Compressed : No

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Connection: keep-alive
Content-Length: 625
Last-Modified: Wed, 09 Dec 2020 04:54:23 GMT
Accept-Ranges: bytes
Content-Type: text/html
Date: Thu, 03 Oct 2024 18:59:55 GMT
ETag: "5fd0587f-271"

<!DOCTYPE html>
<html>
<head>
<title>Welcome to ASWI</title>
<style>
body {
width: 35em;
margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif;
}
</style>
</head>
<body>
<h1>Welcome to ASWI-UAT(dev1)</h1>
<!--<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>-->
</body>
```

Vulnerability finding using namp

Used command

- nmap aswi-op01t.aswatson.com

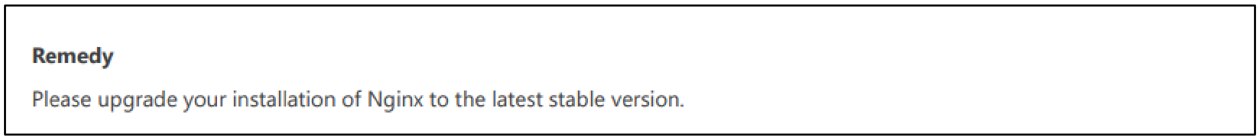


```
┌──(malmi㉿kali)-[~]
└─$ nmap aswi-op01t.aswatson.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-03 16:15 EDT
Nmap scan report for aswi-op01t.aswatson.com (140.238.64.202)
Host is up (0.26s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 41.71 seconds
```

Founded open port

- 80/tcp open http
- 443/tcp open https

Proposed mitigation or fix


Remedy

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

- **Upgrade Nginx:**
  Keep Nginx updated with the latest stable version available. It is recommended to keep Nginx updated since older versions may have security vulnerabilities that still remain unpatched. For example, it will find version 1.18.0 and suggest updating it to the most recent one available in October 2024 to avoid security vulnerabilities like DoS via the ngx_http_mp4_module module or problems with the validation of SSL certificates.

- **Security Patches:**
  Apply all security patches for Nginx. One of the best ways to get notified of new patches and vulnerabilities is by subscribing to Nginx Security Advisories.

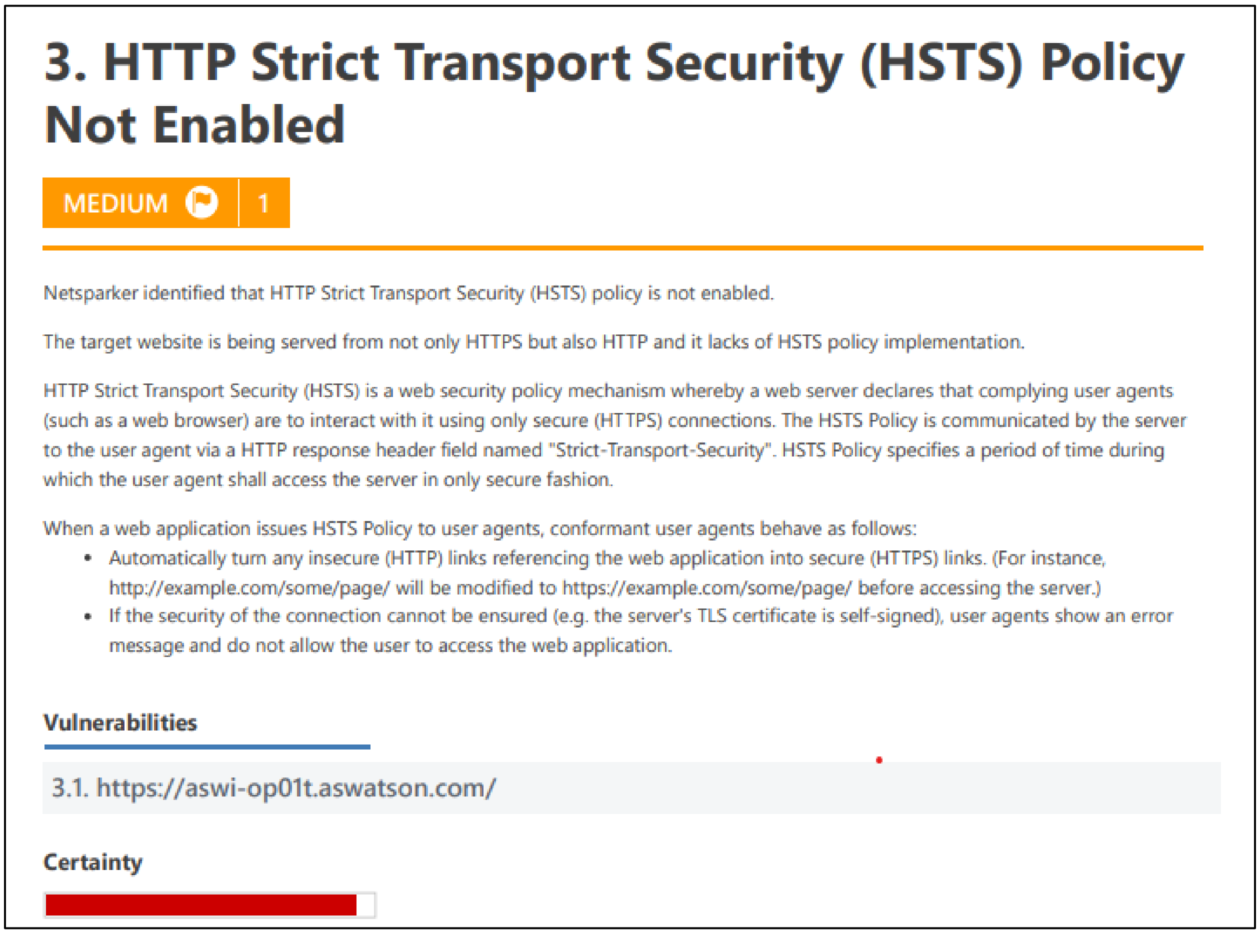
- **Testing of the Update:**
  After the update, thorough testing of the server functionality should be done to make sure that the upgrade will not cause disruptions to any of your applications.

- **practice to deploy a Web Application Firewall (WAF):**
  Employ a Web Application Firewall to protect against possible exploits targeting Nginx vulnerabilities. A WAF will be able to provide some level of detection and blocking, even against zero-day attacks.

- **Implement Hardened Security Configuration:**
  Ensure Nginx uses secure configuration: for instance, disabling modules not in use, setting resource limits, and using security headers to help prevent attacks.

This upgrade and configuration will go a long way toward mitigating the risks identified by using a very outdated version of Nginx.

2. Tittle - HTTP Strict Transport Security (HSTS) Policy Not Enabled



3. HTTP Strict Transport Security (HSTS) Policy Not Enabled

MEDIUM 1

Netsparker identified that HTTP Strict Transport Security (HSTS) policy is not enabled.

The target website is being served from not only HTTPS but also HTTP and it lacks of HSTS policy implementation.

HTTP Strict Transport Security (HSTS) is a web security policy mechanism whereby a web server declares that complying user agents (such as a web browser) are to interact with it using only secure (HTTPS) connections. The HSTS Policy is communicated by the server to the user agent via a HTTP response header field named "Strict-Transport-Security". HSTS Policy specifies a period of time during which the user agent shall access the server in only secure fashion.

When a web application issues HSTS Policy to user agents, conformant user agents behave as follows:

- Automatically turn any insecure (HTTP) links referencing the web application into secure (HTTPS) links. (For instance, http://example.com/some/page/ will be modified to https://example.com/some/page/ before accessing the server.)
- If the security of the connection cannot be ensured (e.g. the server's TLS certificate is self-signed), user agents show an error message and do not allow the user to access the web application.

Vulnerabilities

3.1. https://aswi-op01t.aswatson.com/

Certainty

Description

The vulnerability in HTTP Strict Transport Security Policy Not Enabled refers to the missing feature in a website that would ensure web browsers can connect only over a secure, encrypted HTTPS connection. HSTS is a mechanism of web security policy that helps protect websites from man-in-the-middle attacks such as protocol downgrade attack, or cookie hijacking.

Users may inadvertently connect to a website over an insecure, unencrypted HTTP connection, especially during the normal course of typing a URL without specifying HTTPS-which happens not to have HSTS enabled. An active attacker can intercept such communications and steal sensitive data or inject malicious content into the communication. In this regard, with the website failing to force HTTPS via HSTS, they remain exposed to such attacks and compromise not only their security but also that of their users.

Affected components

The Affected Components of the HTTP Strict Transport Security (HSTS) Policy Not Enabled vulnerability usually include:

- **Web Servers:**
  The web server, such as Apache, Nginx, IIS, which is hosting the application that does not have the HSTS policy enabled. The web server, in this case, would have the capability to support secure connections and implement the HSTS headers.

- **Web Applications:**
  All the web applications that are hosted on the server, especially which handle any kind of sensitive data or authentication, will be affected. These applications in the absence of HSTS will be open to man-in-the-middle attacks.

- **Web Browsers:**
  The users' web browsers are affected because they can connect, by default, to an insecure HTTP connection when visiting a site; therefore, users become prone to various attacks.

- **User Sessions:**
  User session cookies and login information might get exposed if the users connect over insecure HTTP instead of HTTPS; this may lead to session hijacking.

- **Mobile Applications:**
  Such mobile applications interacting with the web server are also vulnerable because they usually exchange data through web services. In the absence of HSTS, interception of mobile apps is possible in case of connectivity over insecure HTTP.

Impact Assessment

- **Man-in-the-Middle Attacks:** MITM attacks are possible simply because an attacker, without HSTS in place, can intercept and modify traffic between the user and server to steal sensitive information, such as login credentials or financial data. Users could inadvertently connect over HTTP, which would make the attack very easy to conduct.

- **Protocol Downgrade Attack:** An attacker can force a user's connection to fall back from HTTPS to HTTP, thus making it susceptible to interception. This would expose, to eavesdropping and tampering, the usually encrypted communication.

- **Session Hijacking:** Without enforced HTTPS, session cookies may be transmitted over insecure channels, where attackers may hijack user sessions, posing as the users.

- **SSL Stripping:** Attackers will forcibly shift traffic from HTTPS to HTTP in SSL stripping, whereby users believe they are on a secure connection while they are not.

Steps to reproduce

- **Target a Website to Attack:**
  Select any website that you want to check if it has implemented HSTS.

- **Check HTTP Response Headers:**
  Using a web browser, visit the target website over HTTP-for instance, by typing in the URL, starting with http://)
  Open the developer tools, usually via F12 or right-click and select "Inspect," then head to the "Network" tab.
  Reload the page, select the website's entry in the network panel, and find the Response Headers section.

- **Find the HSTS Header:**
  Look for the Strict-Transport-Security header coming from the response. If that's not there, then HSTS is not enabled on the website.

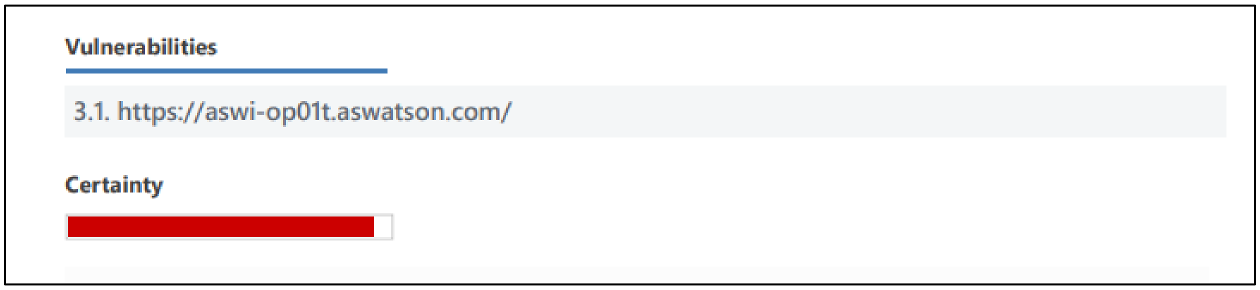
- **HTTPS Connection Test:**
  Now, open the website with HTTPS-for example, https:// in the URL.
  Again, developer tools will look for the Strict-Transport-Security header in the response headers.

- **Observe the Behavior:**
  This means the site is vulnerable as the HSTS header is not set over both HTTP and HTTPS responses.
  You should also observe whether access to the site over HTTP automatically redirects to HTTPS or remains insecure.

- **Document Findings:**
  Note that the HSTS header wasn't present and any observation you have made while you were testing.

Proof of concept (if applicable)

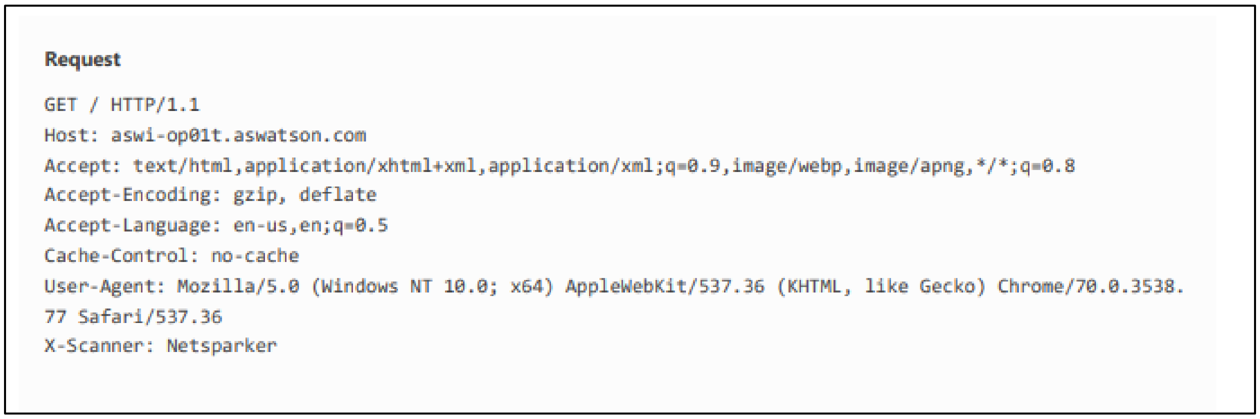
Vulnerability scanning using netsparker


Vulnerabilities

3.1. https://aswi-op01t.aswatson.com/

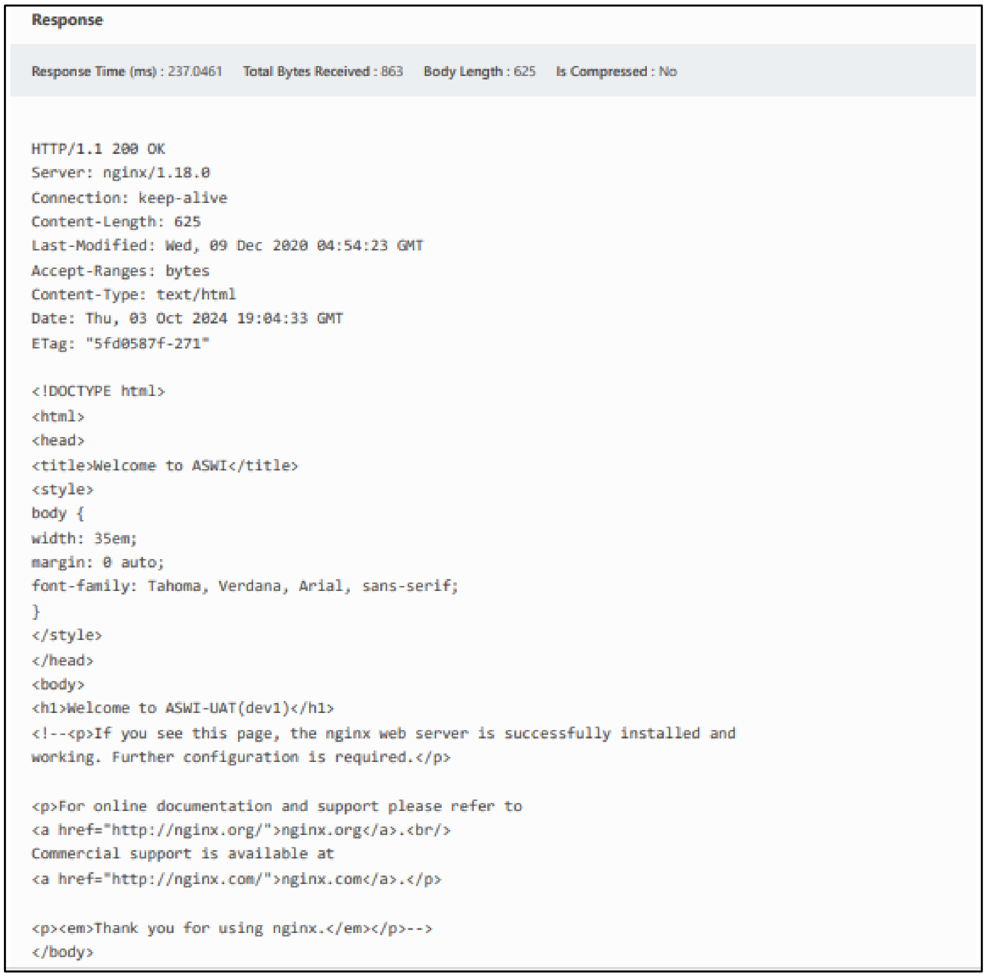
Certainty

- **Request**


Request

```
GET / HTTP/1.1
Host: aswi-op01t.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**



Response

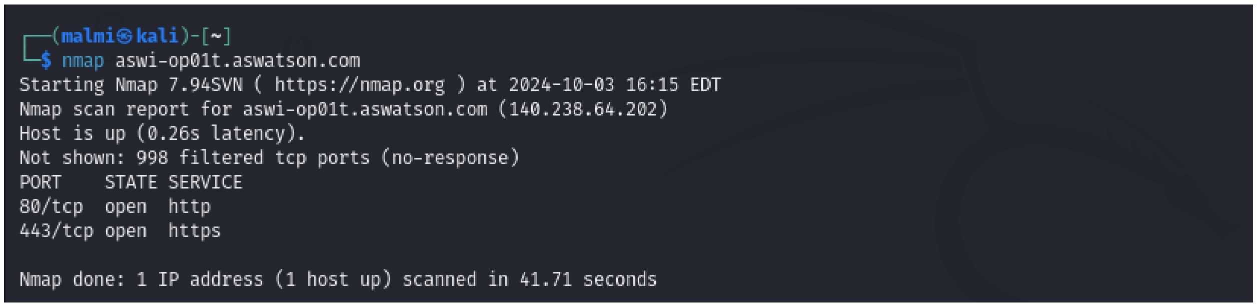
Response Time (ms) : 237.0461 Total Bytes Received : 863 Body Length : 625 Is Compressed : No

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Connection: keep-alive
Content-Length: 625
Last-Modified: Wed, 09 Dec 2020 04:54:23 GMT
Accept-Ranges: bytes
Content-Type: text/html
Date: Thu, 03 Oct 2024 19:04:33 GMT
ETag: "5fd0587f-271"

<!DOCTYPE html>
<html>
<head>
<title>Welcome to ASWI</title>
<style>
body {
width: 35em;
margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif;
}
</style>
</head>
<body>
<h1>Welcome to ASWI-UAT(dev1)</h1>
<!--<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>-->
</body>
```

Vulnerability finding using namp

Used command

- nmap aswi-op01t.aswatson.com



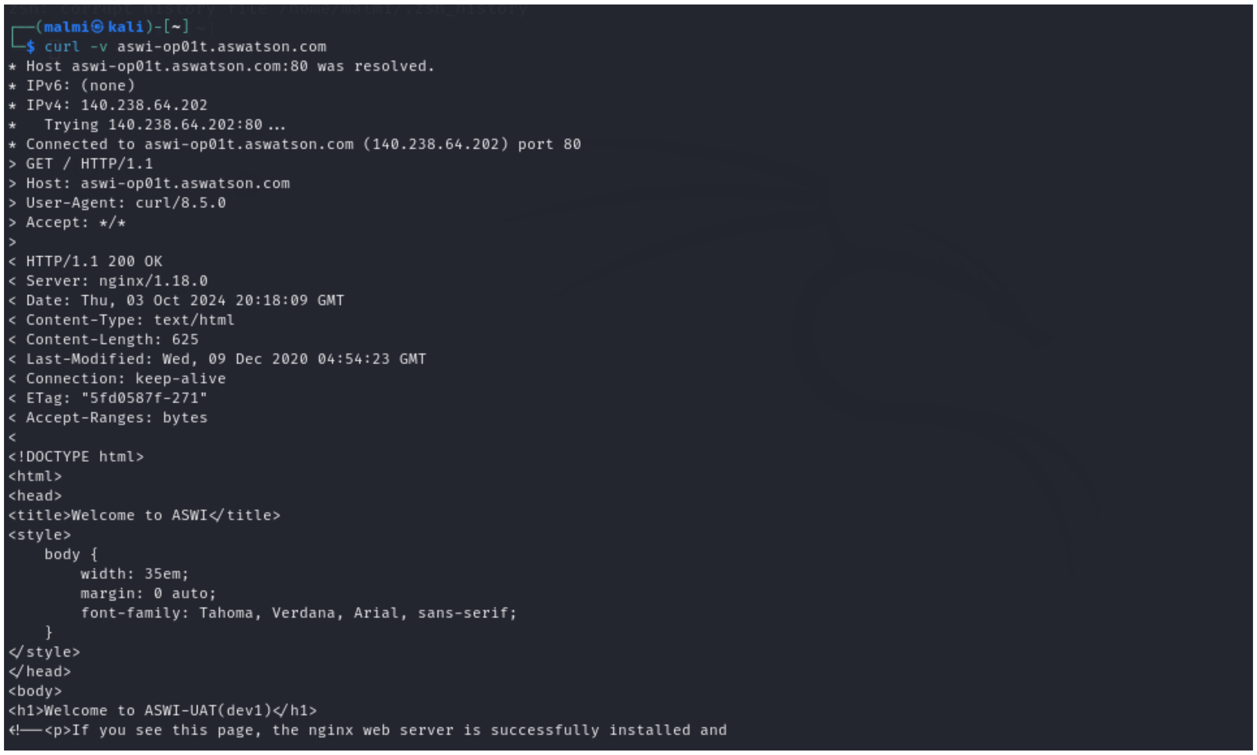
```
┌──(malmi㉿kali)-[~]
└─$ nmap aswi-op01t.aswatson.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-03 16:15 EDT
Nmap scan report for aswi-op01t.aswatson.com (140.238.64.202)
Host is up (0.26s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 41.71 seconds
```

Founded open port

- 80/tcp open http
- 443/tcp open https

Vulnerability finding using curl

- **Used command**
  - curl -v aswi-op01t.aswatson.com

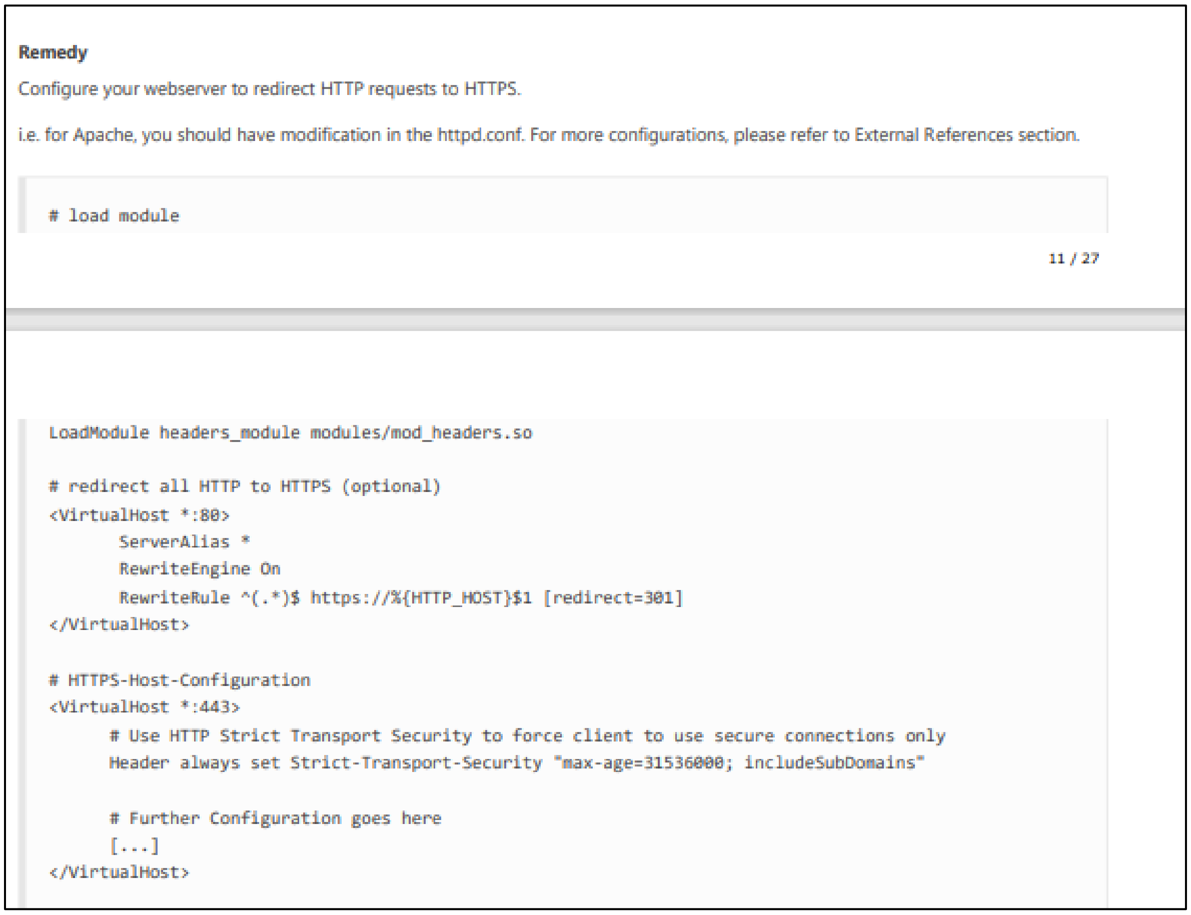


```
┌──(malmi㉿kali)-[~]
└─$ curl -v aswi-op01t.aswatson.com
* Host aswi-op01t.aswatson.com:80 was resolved.
* IPv6: (none)
* IPv4: 140.238.64.202
*   Trying 140.238.64.202:80 ...
* Connected to aswi-op01t.aswatson.com (140.238.64.202) port 80
> GET / HTTP/1.1
> Host: aswi-op01t.aswatson.com
> User-Agent: curl/8.5.0
> Accept: */*
>
< HTTP/1.1 200 OK
< Server: nginx/1.18.0
< Date: Thu, 03 Oct 2024 20:18:09 GMT
< Content-Type: text/html
< Content-Length: 625
< Last-Modified: Wed, 09 Dec 2020 04:54:23 GMT
< Connection: keep-alive
< ETag: "5fd0587f-271"
< Accept-Ranges: bytes
<
<!DOCTYPE html>
<html>
<head>
<title>Welcome to ASWI</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to ASWI-UAT(dev1)</h1>
<!--<p>If you see this page, the nginx web server is successfully installed and
```

- **Results**

  The absence of the Strict-Transport-Security header in the response proves that HSTS is not enabled on the server, leaving it vulnerable to various SSL/TLS-related attacks.

Proposed mitigation or fix



Remedy

Configure your webserver to redirect HTTP requests to HTTPS.

i.e. for Apache, you should have modification in the httpd.conf. For more configurations, please refer to External References section.

```
# load module
```

11 / 27

```
LoadModule headers_module modules/mod_headers.so

# redirect all HTTP to HTTPS (optional)
<VirtualHost *:80>
       ServerAlias *
       RewriteEngine On
       RewriteRule ^(.*)$ https://%{HTTP_HOST}$1 [redirect=301]
</VirtualHost>

# HTTPS-Host-Configuration
<VirtualHost *:443>
      # Use HTTP Strict Transport Security to force client to use secure connections only
      Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"

      # Further Configuration goes here
      [...]
</VirtualHost>
```

- **Enable HSTS:**
  Configure your web server to include the HSTS header in the HTTP responses. This will tell the browser to access the site only over HTTPS.

- **Set Header Parameters:**
  The HSTS header should set appropriate parameters for the period of time the browser should enforce HTTPS connections and whether the policy applies to subdomains.

- **Redirect All HTTP Traffic to HTTPS:**
  Make sure any request made to your website over HTTP is automatically forwarded to HTTPS. This could prevent an attacker from allowing a user to access the website over an insecure connection.

- **Regular Security Audits:**
  Regularly verify that HSTS is set up properly and that at no instance is the site communicating with any of its users via HTTPS.

- **Inform Users:**
  Educate users about the implications of using your site with insecure connections. Suggest that they should bookmark the HTTPS version of the site.
  The good news is that with these steps, you avoid the risks of not having HSTS enabled, thereby securing your web application even more.