



# SLIIT

*Discover Your Future*

Sri Lanka Institute of Information Technology

## Bug Bounty Report 02

[mcms.app.aswatson.com](https://mcms.app.aswatson.com)

IE2062 – Web Security

Submitted by:

**IT22227836 – ADHIKARI A.M.V.B.M**

Date of submission

2024.10.31

## Table of Contents

Report 02 – mcms.app.aswatson.com .....	3
Vulnerability .....	5
1. Title - Weak Ciphers Enabled .....	5
Description .....	5
Affected components .....	6
Impact Assessment .....	7
Steps to reproduce .....	8
Proof of concept (if applicable) .....	9
Vulnerability scanning using Netsparker .....	9
Vulnerability finding using namp .....	10
Proposed mitigation or fix .....	10
2. Title - HTTP Strict Transport Security (HSTS) Policy Not Enabled .....	11
Description .....	12
Impact Assessment .....	13
Steps to reproduce .....	13
Proof of concept (if applicable) .....	14
Vulnerability scanning using netsparker .....	14
Vulnerability finding using namp .....	16
Vulnerability finding using curl .....	16
Proposed mitigation or fix .....	17

## Report 02 – mcms.app.aswatson.com

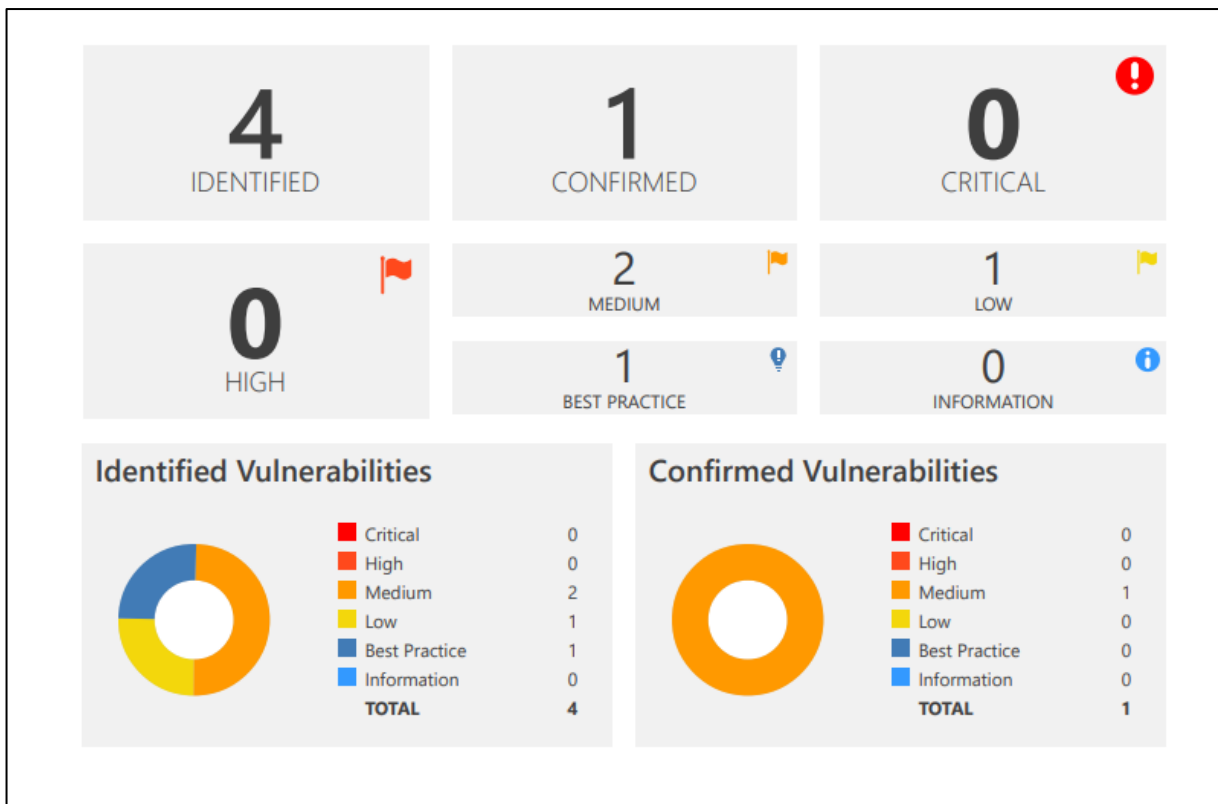
Main domain	<a href="https://www.aswatson.com/">https://www.aswatson.com/</a>
Sub domain	mcms.app.aswatson.com
IP address	23.215.7.8
platform	HackerOne



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
<b>A3 - SENSITIVE DATA EXPOSURE</b>				
🚩	<a href="#">Weak Ciphers Enabled</a>	GET	https://mcms.app.aswatson.com/	MEDIUM
🚩	<a href="#">HTTP Strict Transport Security (HSTS) Policy Not Enabled</a>	GET	https://mcms.app.aswatson.com/	MEDIUM
🚩	<a href="#">Referrer-Policy Not Implemented</a>	GET	https://mcms.app.aswatson.com/	BEST PRACTICE
<b>A6 - SECURITY MISCONFIGURATION</b>				
🚩	<a href="#">Missing X-Frame-Options Header</a>	GET	https://mcms.app.aswatson.com/	LOW

## Vulnerability

### 1. Title - Weak Ciphers Enabled

### 1. Weak Ciphers Enabled

MEDIUM  1

CONFIRMED  1

---

Netsparker detected that weak ciphers are enabled during secure communication (SSL).

You should allow only strong ciphers on your web server to protect secure communication with your visitors.

**Impact**

Attackers might decrypt SSL traffic between your server and your visitors.

**Vulnerabilities**

1.1. <https://mcms.app.aswatson.com/>

CONFIRMED

- Risk – Medium

## Description

Weak Ciphers Enabled: This typically indicates that a server supports insecure encryption algorithms of an older version that may potentially be exploited by an attacker to break data security. The weak ciphers include RC4, DES, and older versions of SSL/TLS such as SSLv2 and SSLv3. All these are vulnerable to known cryptographic attacks, which may allow attackers to decrypt sensitive data, intercept communications, or perform man-in-the-middle and downgrade attacks.

The ability to use such weak ciphers allows the attacker unauthorized access to sensitive information, such as login credentials or session tokens. Organizations should disable weak ciphers and allow only strong ones resistant to unauthorized decryption, such as TLS 1.2 or TLS 1.3. Execution of periodic vulnerability assessments, combined with periodic updates within server configuration, would minimize the risk of weak ciphers.

## Affected components

- **Web Servers:**

Servers that run Web applications and have weak or obsolete encryption algorithms enabled in their settings of SSL/TLS are those that are most vulnerable to attack. The attacker can use weak ciphers to intercept or decrypt sensitive data that is in transit.

- **Email Servers:**

Email servers, for instance, SMTP servers, that have weak ciphers turned on may expose email communications to eavesdropping and manipulation.

- **VPN Servers:**

The weak encryption ciphers that are employed by the VPN servers can reveal sensitive data. This is because the attackers may decrypt the traffic between the user and the VPN server.

- **Database Servers:**

If the connections to the database use weak ciphers to secure the communication of applications with the databases, then the tendency for data disclosure is great.

- **Load Balancers and Proxy Servers:**

These are components used in managing traffic between users and servers, and in cases where weak ciphers are supported, data passing through can easily be intercepted or manipulated.

- **Any SSL/TLS-Enabled Service:**

This means any service utilizing outdated ciphers-for instance, APIs, file transfer protocols of a custom application-will be exploited if these weak ciphers remain enabled in their configuration. protocols, such as TLS 1.2 or 1.3.

## Impact Assessment

context of the weak ciphers used clearly shows that the Impact Assessment of the Weak The Ciphers Enabled vulnerability could be big. Here's a breakdown of the potential impact:

- **Confidentiality Breach:**

Exposure of Data: Weak ciphers can allow attackers to decrypt sensitive data that may be transmitted across the network. Examples of this include login credentials, personal information, or financial details. These may lead to breaches of confidential information.

Man-in-the-Middle Attack: Third-party intruders can hijack and read communications between the client and server, which is an attack on data privacy.

- **Integrity Compromise:**

Data Tampering: Using weak encryption, an intruder could alter the data in transmission without being detected, such that malicious modifications to transactions, documents, or communications might be done.

Spoofing: Impostors would be able to pose as a legitimate user or service, creating the possibility for unauthorized application or system actions.

- **Availability Risk:**

Downgrade Attacks: An attacker uses weak ciphers to forcibly make a connection use even weaker or older encryption, like forcing it from TLS 1.3 down to SSLv3. This leaves the application more susceptible to other types of attacks. Sometimes, one might exploit weak ciphers as part of DoS attacks; the effectiveness relates to exhausting system resources by forcing it to perform great numbers of cryptographic operations.

- **Reputation Damage:**

Breach of Trust: Data breaches or manipulation through weak ciphers would destroy organizations' reputations and therefore lower customer trust, leading to lost business opportunities.

- **Compliance Violations:**

Regulatory Fines: Most industries have strict regulations about how data should be protected—for example, GDPR, HIPAA, and PCI DSS. The use of weak ciphers could amount to non-compliance, hence attracting legal actions or financial penalties.

## Steps to reproduce

- **Identify the Target Server:**  
Identify the specific web server, mail server, or VPN server that must be scanned for supporting weak ciphers.
- **Run a Security Scanning Tool:**  
One security scanning tool needs to be selected that is used to scan SSL/TLS configuration apart from other things. It may include but is not limited to SSL Labs, Nmap, and Qualys SSL Test.  
These utilities will automatically identify and scan the SSL/TLS configuration of the server and report back with active encryption ciphers of the server while highlighting weak or vulnerable ones.
- **Run the Scan:**  
Run a scan against the target server. Such tools will test the supported SSL/TLS ciphers to determine if any of the weak ones, like RC4, DES, and SSLv3, are enabled.
- **Analyze Results:**  
Look for weak or deprecated ciphers from the scan report. If the scan flags weak ciphers, this is your proof of vulnerability.
- **Proof of Concept:**  
Once the weak ciphers have been identified, document them by making a note of the weak encryption algorithms or very outdated SSL/TLS versions supported by the server.



## Proof of concept (if applicable)

### Vulnerability scanning using Netsparker

#### 1. Weak Ciphers Enabled

**MEDIUM**

1

**CONFIRMED**

1

Netsparker detected that weak ciphers are enabled during secure communication (SSL).

You should allow only strong ciphers on your web server to protect secure communication with your visitors.

##### Impact

Attackers might decrypt SSL traffic between your server and your visitors.

##### Vulnerabilities

1.1. <https://mcms.app.aswatson.com/>

**CONFIRMED**

- Request

##### Request

[NETSPARKER] SSL Connection

- Response

##### Response

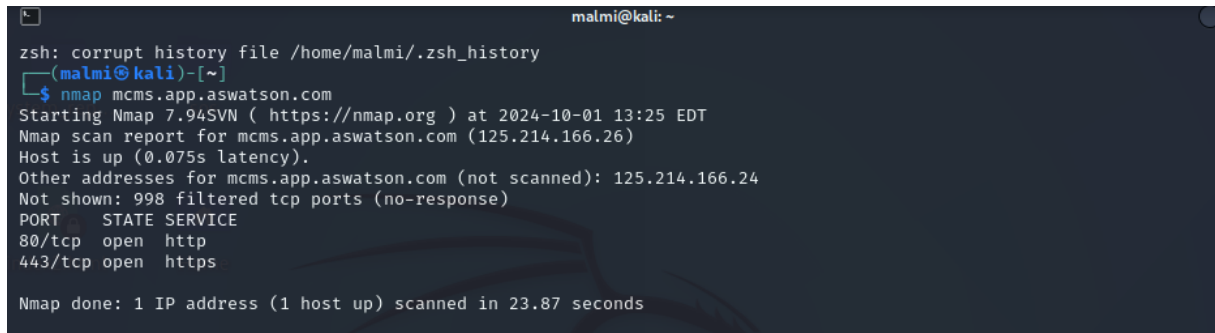
Response Time (ms) : 1    Total Bytes Received : 27    Body Length : 0    Is Compressed : No

[NETSPARKER] SSL Connection

## Vulnerability finding using nmap

### Used command

- `nmap mcms.app.aswatson.com`



```
malmi@kali: ~  
zsh: corrupt history file /home/malmi/.zsh_history  
(malmi@kali)-[~]  
$ nmap mcms.app.aswatson.com  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-01 13:25 EDT  
Nmap scan report for mcms.app.aswatson.com (125.214.166.26)  
Host is up (0.075s latency).  
Other addresses for mcms.app.aswatson.com (not scanned): 125.214.166.24  
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 23.87 seconds
```

### Founded open port

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

## Proposed mitigation or fix

- **Weak Ciphers:**  
Disable the usage of weak ciphers like RC4, DES, 3DES, and all ciphers associated with either SSLv2 or SSLv3, and ensure that only modern and strong ciphers like AES or ChaCha20 are used.
- **Strong Protocols**  
Deactivate the older versions of SSL/TLS, which have various vulnerabilities. It is about disabling SSLv2, SSLv3, and TLS 1.0/1.1, leaving only TLS 1.2 and TLS 1.3.
- **Enable Forward Secrecy:**  
Ensure that the cipher suites supporting Perfect Forward Secrecy/PFS, such as ECDHE or DHE-based ones, are prioritized. That will ensure the session keys cannot be compromised when a server's private key has been compromised.
- **Keep Regularly Updating SSL/TLS Configuration**  
Periodically review and update the server's SSL/TLS configuration to stay in step with evolving security standards and to disable any new weak ciphers as they are discovered.

- **Use Strong Key Sizes:**  
Configure your server for ample key lengths of encryption, such as a minimum of 2048-bit RSA in public-key encryption and minimum of 256-bit AES in symmetric encryption.
- **Run Security Audits on a Regular Schedule:**  
Regularly scan your server using SSL scanning tools to make sure weak ciphers do not come back again, and the configuration of SSL/TLS remains secure

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

### 2. 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

2.1. <https://mcms.app.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

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

MEDIUM  1

- **Request**

**Request**

```
GET / HTTP/1.1
Host: mcms.app.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) : 1209.6695    Total Bytes Received : 587    Body Length : 379    Is Compressed : No

HTTP/1.1 403 Forbidden  
Server: AkamaiGHost  
Content-Length: 379  
Expires: Tue, 01 Oct 2024 09:04:01 GMT  
Connection: close  
Mime-Version: 1.0  
Content-Type: text/html  
Date: Tue, 01 Oct 2024 09:04:01 GMT

```
<HTML><HEAD>  
<TITLE>Access Denied</TITLE>  
</HEAD><BODY>  
<H1>Access Denied</H1>
```

You don't have permission to access "http&#58;&#47;&#47;mcms&#46;app&#46;aswatson&#46;com&#47;" on this server.<P>

Reference&#32;&#35;18&#46;16a6d67d&#46;1727773440&#46;14232cb0

<P>https&#58;&#47;&#47;errors&#46;edgesuite&#46;net&#47;18&#46;16a6d67d&#46;1727773440&#46;14232cb0</P>

</BODY>

</HTML>

## Vulnerability finding using nmap

### ▪ Used command

- nmap mcms.app.aswatson.com

```
malmi@kali: ~  
zsh: corrupt history file /home/malmi/.zsh_history  
malmi@kali:~$  
$ nmap mcms.app.aswatson.com  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-01 13:25 EDT  
Nmap scan report for mcms.app.aswatson.com (125.214.166.26)  
Host is up (0.075s latency).  
Other addresses for mcms.app.aswatson.com (not scanned): 125.214.166.24  
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 23.87 seconds
```

### ▪ Founded open port

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

## Vulnerability finding using curl

```
malmi@kali:~$  
$ curl -v https://mcms.app.aswatson.com  
* Host mcms.app.aswatson.com:443 was resolved.  
* IPv6: (none)  
* IPv4: 23.59.168.145, 23.52.171.234  
* Trying 23.59.168.145:443...  
* Connected to mcms.app.aswatson.com (23.59.168.145) port 443  
* ALPN: curl offers h2,http/1.1  
* TLSv1.3 (OUT), TLS handshake, Client hello (1):  
* CAfile: /etc/ssl/certs/ca-certificates.crt  
* CApath: /etc/ssl/certs  
* TLSv1.3 (IN), TLS handshake, Server hello (2):  
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):  
* TLSv1.3 (IN), TLS handshake, Certificate (11):  
* TLSv1.3 (IN), TLS handshake, CERT verify (15):  
* TLSv1.3 (IN), TLS handshake, Finished (20):  
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):  
* TLSv1.3 (OUT), TLS handshake, finished (24):  
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384 / X25519 / id-ecPublicKey  
* ALPN: server accepted h2  
* Server certificate:  
* subject: CN=, L=New Territories; O=A. S. Watson Retail (HK) Limited; CN=www2.aswatson.com  
* start date: Dec  3 00:00:00 2023 GMT  
* expire date: Dec  3 23:59:59 2024 GMT  
* subjectAltName: host "mcms.app.aswatson.com" matched cert's "mcms.app.aswatson.com"  
* issuer: C=US; O=DigiCert Inc; CN=DigiCert TLS RSA SHA256 2020 CA1  
* SSL certificate verify ok.  
* Certificate level 0: Public key type EC/prime256v1 (256/128 Bits/secbits), signed using sha256withRSAEncryption  
* Certificate level 1: Public key type RSA (2048/112 Bits/secbits), signed using sha256withRSAEncryption  
* Certificate level 2: Public key type RSA (2048/112 Bits/secbits), signed using sha256withRSAEncryption  
* using HTTP/2  
* [HTTP/2] [1] OPENED stream for https://mcms.app.aswatson.com/  
* [HTTP/2] [1] [method: GET]  
* [HTTP/2] [1] [scheme: https]  
* [HTTP/2] [1] [authority: mcms.app.aswatson.com]  
* [HTTP/2] [1] [path: /]  
* [HTTP/2] [1] [user-agent: curl/8.5.0]  
* [HTTP/2] [1] [accept: */*]  
> GET / HTTP/2  
> Host: mcms.app.aswatson.com  
> User-Agent: curl/8.5.0  
> Accept: */*  
>  
* TLSv1.3 (IN), TLS handshake, NewSession Ticket (4):  
* TLSv1.3 (IN), TLS handshake, NewSession Ticket (4):  
* old SSL session ID is stale, removing  
< HTTP/2 200  
< content-type: text/html  
< content-length: 1558  
< content-md5: Lxc9mqYmL9mq5lgt64AQa==
```



- **Used command**

- `curl -v https://mcms.app.aswatson.com`

- **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

- **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.