Sri Lanka Institute of Information Technology

Bug Bounty Report 06

AutoDiscover.aswatson.com

IE2062 – Web Security

Submitted by:

IT22227836 – ADHIKARI A.M.V.B.M

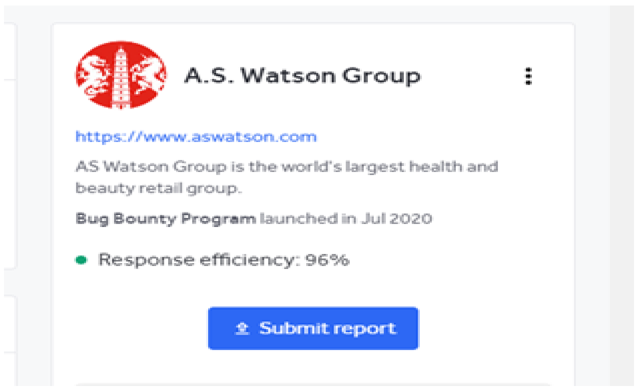
Date of submission

2024.10.31

Table of Contents

Report 06 – AutoDiscover.aswatson.com

Main domain	https://www.aswatson.com/
Sub domain	AutoDiscover.aswatson.com
IP address	52.98.123.248
platform	HackerOne


A.S. Watson Group

https://www.aswatson.com

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

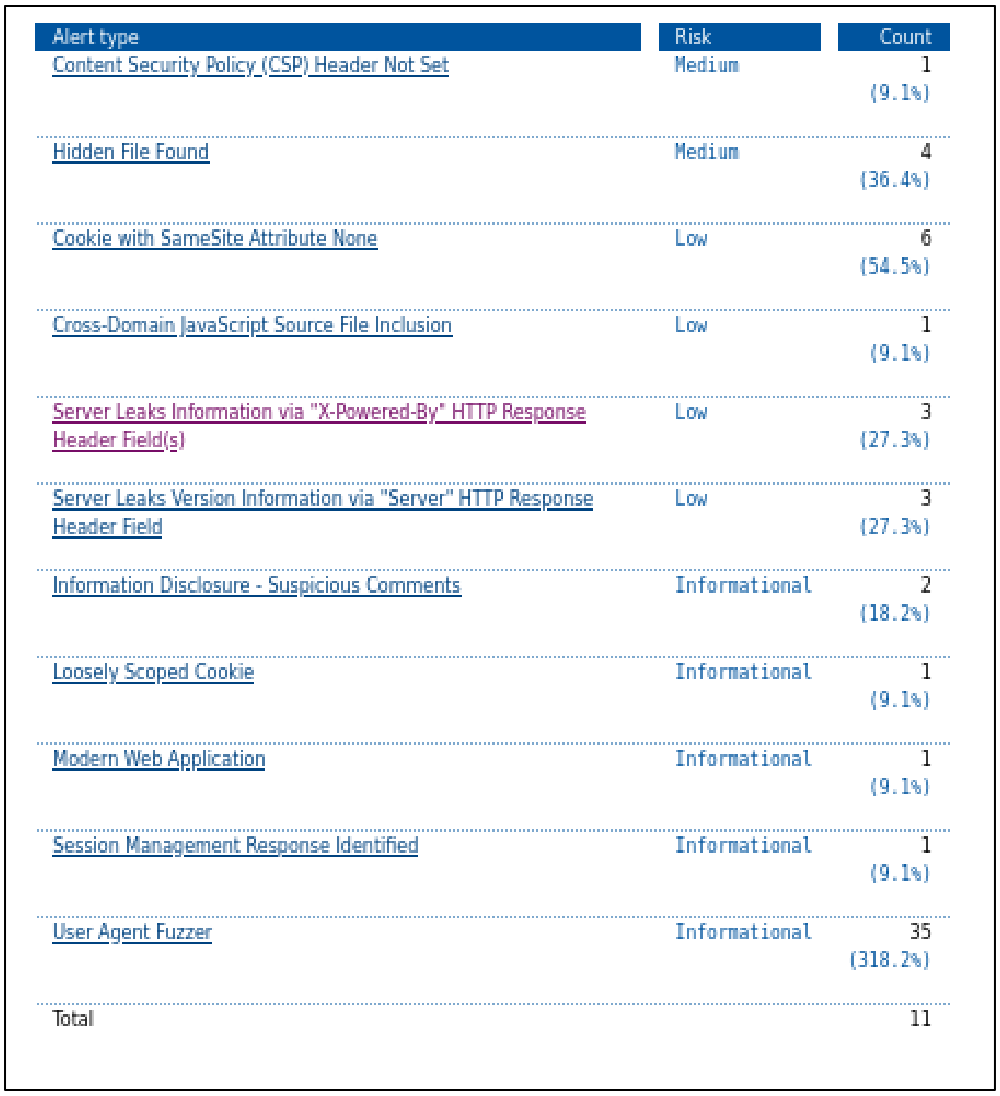
Bug Bounty Program launched in Jul 2020

● 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)

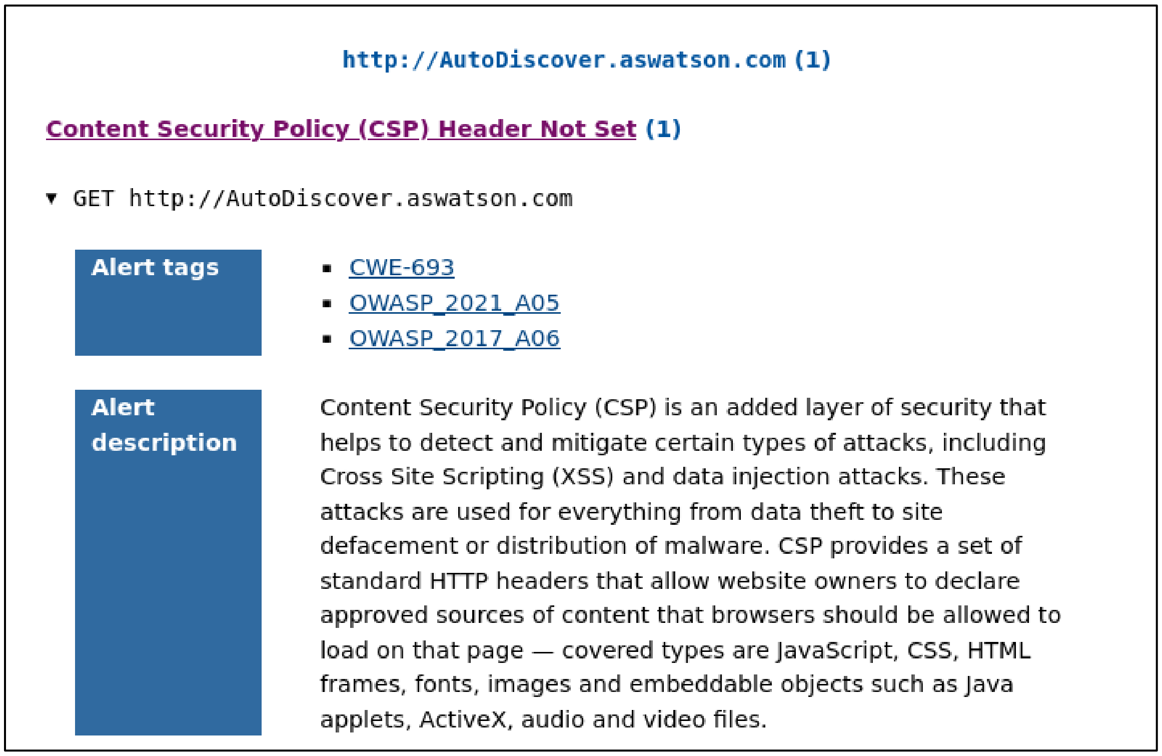
Vulnerability detected



Alert type	Risk	Count
Content Security Policy (CSP) Header Not Set	Medium	1 (9.1%)
Hidden File Found	Medium	4 (36.4%)
Cookie with SameSite Attribute None	Low	6 (54.5%)
Cross-Domain JavaScript Source File Inclusion	Low	1 (9.1%)
Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)	Low	3 (27.3%)
Server Leaks Version Information via "Server" HTTP Response Header Field	Low	3 (27.3%)
Information Disclosure - Suspicious Comments	Informational	2 (18.2%)
Loosely Scoped Cookie	Informational	1 (9.1%)
Modern Web Application	Informational	1 (9.1%)
Session Management Response Identified	Informational	1 (9.1%)
User Agent Fuzzer	Informational	35 (318.2%)
Total		11

Vulnerability

1. Title - Content Security Policy (CSP) Header Not Set



http://AutoDiscover.aswatson.com (1)

Content Security Policy (CSP) Header Not Set (1)

▼ GET http://AutoDiscover.aswatson.com

Alert tags	▪ CWE-693 ▪ OWASP_2021_A05 ▪ OWASP_2017_A06
Alert 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.

Description

Web applications are vulnerable to several attacks, including data injection and Cross-Site Scripting (XSS), when a Content Security Policy (CSP) header is missing. By defining which sources are permitted to load resources like scripts, styles, and media on a web page, CSP is intended to stop the execution of untrusted programs. Attackers may insert malicious scripts that could cause data theft, user session hijacking, or unauthorized access if this header is missing.

The web application is more vulnerable to client-side attacks since it does not have a suitable CSP header, which is a critical layer of defense. Strictly limiting the origins from which the website can load executable content improves the application's overall security and helps lower the chance of code insertion

Affected components

The following are the main elements that are impacted when a Content Security Policy (CSP) header is missing:

- **Client-Side Scripts**: Malicious JavaScript can be introduced into a browser without a CSP, resulting in XSS attacks that compromise user sessions, pilfer confidential information, or disseminate malware.
- **Loading External Resources**: The application's attack surface can be further increased by allowing the unrestricted loading of external resources from potentially malicious domains, such as images, fonts, and iframes.
- **Inline Execution:** Unrestricted execution of insecure techniques, such as event handlers and inline scripts, which are often stopped by a CSP, raises the possibility of exploitation.
- **Data Integrity:** In the event of a man-in-the-middle attack, when sensitive data is intercepted or altered, attackers may find it easier to influence the content that the browser processes in the absence of a CSP.

.

To shield these components from potential exploitation, the web server must be updated to implement the latest encryption protocols, such as TLS 1.2 or 1.3.

Impact Assessment

Particularly for contemporary web applications, improperly configured Content Security Policy (CSP) headers might have serious security consequences:

- **Increased Risk of XSS Attacks:**
  - Without a properly configured CSP, the web application is more vulnerable to Cross-Site Scripting (XSS) attacks. Attackers can inject malicious scripts into the application, potentially allowing them to steal session tokens, cookies, or sensitive user data.

- **Unrestricted External Resource Loading:**
  - The absence of a CSP header means there are no restrictions on loading external resources (such as scripts, stylesheets, or media). This could expose the application to untrusted sources, increasing the risk of malicious content being loaded, such as malware or harmful code.

- **Sensitive Data Exposure:**
  - If there is no CSP in place, an attacker may be able to take advantage of vulnerability and carry out unauthorized actions that would expose sensitive data, such as private information or user credentials.

- **Violated Application Security and User Trust:**
  - If an application is susceptible to such assaults, users are likely to lose faith in it because their data could be compromised, potentially jeopardizing the application's overall security.

Steps to reproduce

- **Access the Target Application**
  Launch a web browser (such as Firefox or Chrome) and go to the URL of the desired online application.

- **Open Developer Tools**
  To access the browser's Developer Tools, press F12 (or right-click on the page and choose "Inspect"). Go to the Network tab.

- **Reload the webpage**
  Press F5 or click the refresh button to capture network traffic once the Developer Tools are open.

- **Review Response Headers**
  Click on the first request (typically the main page request) in the Network tab. Then, look for a Content-Security-Policy header in the Response Headers by swiping down in the Headers area.

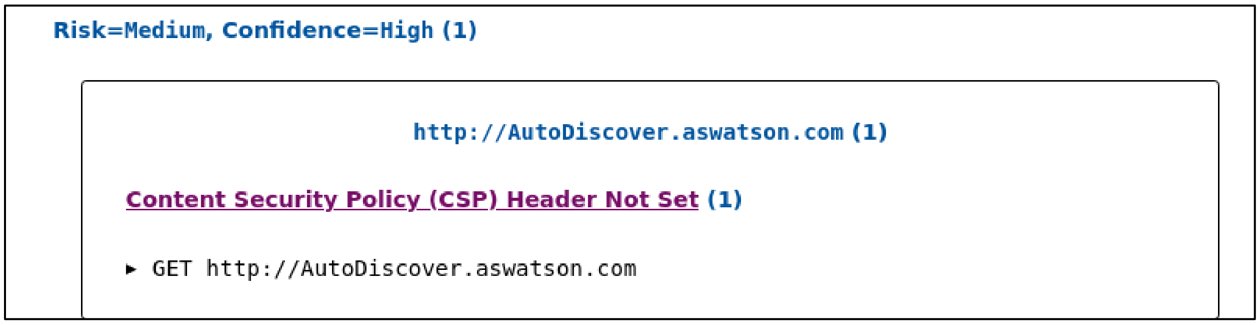
- **Identify the Absence of CSP**
  The vulnerability is indicated if the response's Content-Security-Policy header is missing. The program is vulnerable to some forms of attacks, such as Cross-Site Scripting (XSS), because it lacks CSP.

- **Inject Script Test (Optional)**
  Inject a basic XSS payload (such as **<script>alert(1); </script>)** into any user-input field as an additional validation step. The script's ability to run without error verifies the absence of a security policy that would typically forbid such behaviors

  By taking these steps, you can verify that the application is not vulnerable to any kind of client-side attack because the CSP header is not set.

Proof of concept (if applicable)

Vulnerability scanning using OWASP ZAP

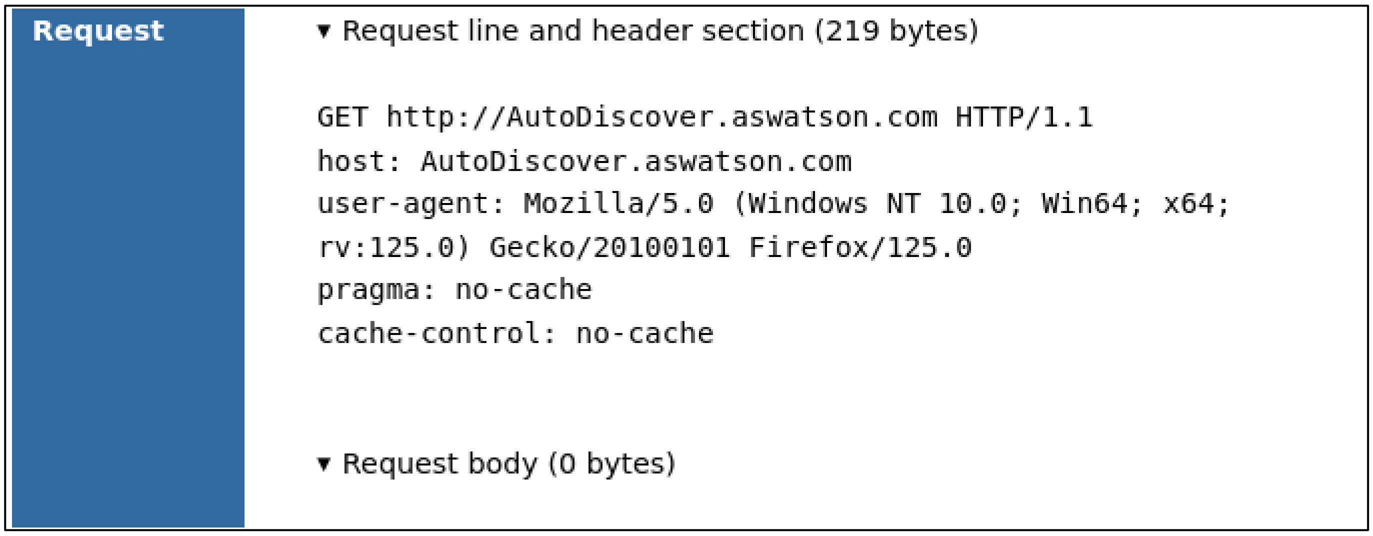

Risk=Medium, Confidence=High (1)

http://AutoDiscover.aswatson.com (1)

Content Security Policy (CSP) Header Not Set (1)

```
▸ GET http://AutoDiscover.aswatson.com
```

- **Request**

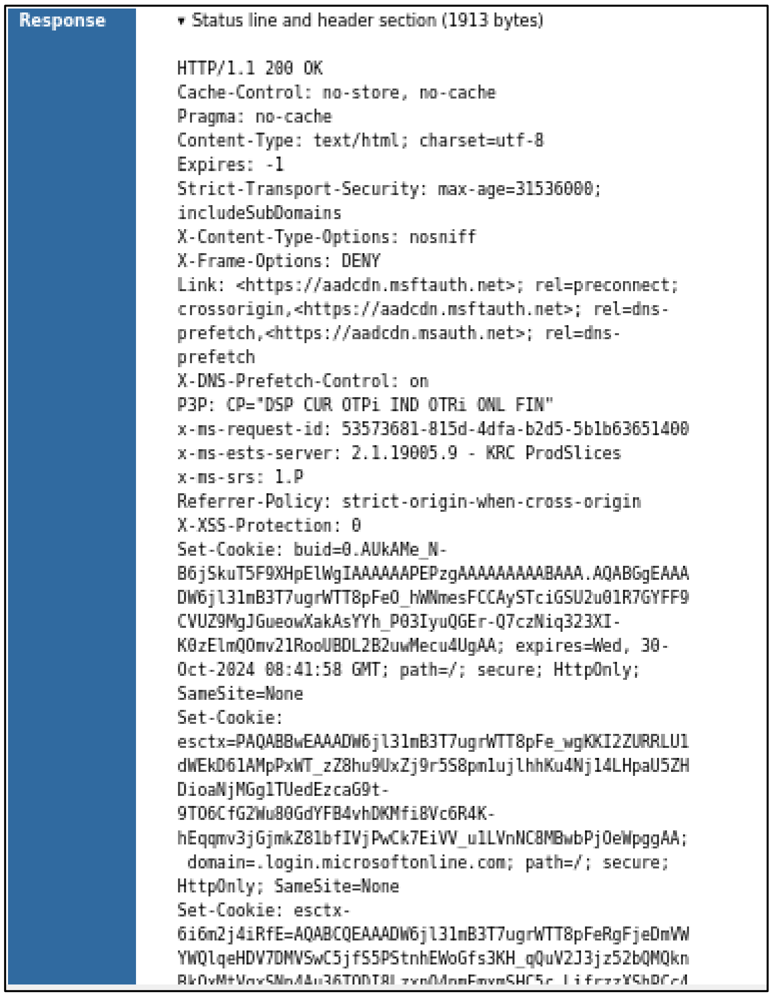

Request

▾ Request line and header section (219 bytes)

```
GET http://AutoDiscover.aswatson.com HTTP/1.1
host: AutoDiscover.aswatson.com
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
rv:125.0) Gecko/20100101 Firefox/125.0
pragma: no-cache
cache-control: no-cache
```

▾ Request body (0 bytes)

- **Response**



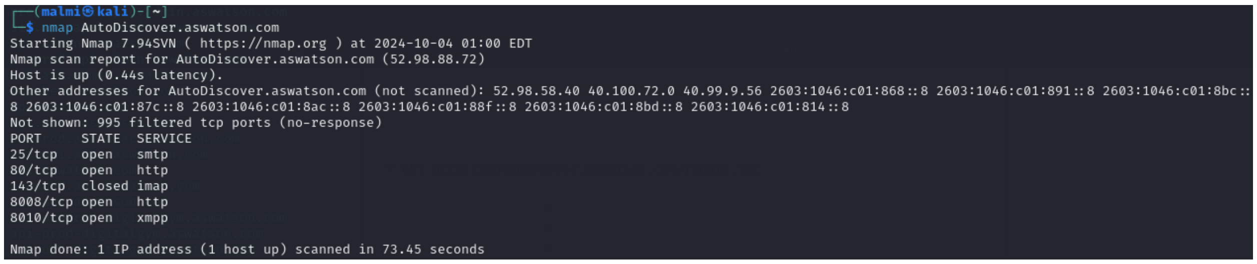
▾ Status line and header section (1913 bytes)

```
HTTP/1.1 200 OK
Cache-Control: no-store, no-cache
Pragma: no-cache
Content-Type: text/html; charset=utf-8
Expires: -1
Strict-Transport-Security: max-age=31536000;
includeSubDomains
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Link: <https://aadcdn.msftauth.net>; rel=preconnect;
crossorigin,<https://aadcdn.msftauth.net>; rel=dns-
prefetch,<https://aadcdn.msauth.net>; rel=dns-
prefetch
X-DNS-Prefetch-Control: on
P3P: CP="DSP CUR OTPi IND OTRi ONL FIN"
x-ms-request-id: 53573681-815d-4dfa-b2d5-5b1b63651400
x-ms-ests-server: 2.1.19005.9 - KRC ProdSlices
x-ms-srs: 1.P
Referrer-Policy: strict-origin-when-cross-origin
X-XSS-Protection: 0
Set-Cookie: buid=0.AUkAMe_N-
B6jSkuT5F9XHpElWgIAAAAAAPEPzgAAAAAAAAABAAA.AQABGgEAAA
DW6jl31mB3T7ugrWTT8pFeO_hWNmesFCCAySTciGSU2u01R7GYFF9
CVUZ9MgJGueowXakAsYYh_P03IyuQGEr-Q7czNiq323XI-
K0zElmQOmv21RooUBDL2B2uwMecu4UgAA; expires=Wed, 30-
Oct-2024 08:41:58 GMT; path=/; secure; HttpOnly;
SameSite=None
Set-Cookie:
esctx=PAQABBwEAAADW6jl31mB3T7ugrWTT8pFe_wgKKI2ZURRLU1
dWEkD61AMpPxWT_zZ8hu9UxZj9r5S8pmlujlhhKu4Nj14LHpaU5ZH
DioaNjMGg1TUedEzcaG9t-
9TO6CfG2Wu80GdYFB4vhDKMfi8Vc6R4K-
hEqqmv3jGjmkZ81bfIVjPwCk7EiVV_u1LVnNC8MBwbPjOeWpggAA;
 domain=.login.microsoftonline.com; path=/; secure;
HttpOnly; SameSite=None
Set-Cookie: esctx-
6i6m2j4iRfE=AQABCQEAAADW6jl31mB3T7ugrWTT8pFeRgFjeDmVW
YWQlqeHDV7DMVSwC5jfS5PStnhEWoGfs3KH_qQuV2J3jz52bQMQkn
[illegible]
```

Vulnerability finding using Namp

Used command

- nmap AutoDiscover.aswatson.com



```
┌──(malmi㉿kali)-[~]
└─$ nmap AutoDiscover.aswatson.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-04 01:00 EDT
Nmap scan report for AutoDiscover.aswatson.com (52.98.88.72)
Host is up (0.44s latency).
Other addresses for AutoDiscover.aswatson.com (not scanned): 52.98.58.40 40.100.72.0 40.99.9.56 2603:1046:c01:868::8 2603:1046:c01:891::8 2603:1046:c01:8bc::
8 2603:1046:c01:87c::8 2603:1046:c01:8ac::8 2603:1046:c01:88f::8 2603:1046:c01:8bd::8 2603:1046:c01:814::8
Not shown: 995 filtered tcp ports (no-response)
PORT     STATE  SERVICE
25/tcp   open   smtp
80/tcp   open   http
143/tcp  closed imap
8008/tcp open   http
8010/tcp open   xmpp

Nmap done: 1 IP address (1 host up) scanned in 73.45 seconds
```

Founded open port

- 25/tcp   open   smtp
- 80/tcp   open   http
- 143/tcp closed imap
- 8008/tcp open   http
- 8010/tcp open   xmpp

Proposed mitigation or fix

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

- **Implement a Basic CSP:** Set a Content Security Policy header that defines default sources for scripts, styles, images, and other resources.

- **Restrict Resource Loading**: Limit where scripts, styles, and other content can be loaded from, using only trusted and secure sources.

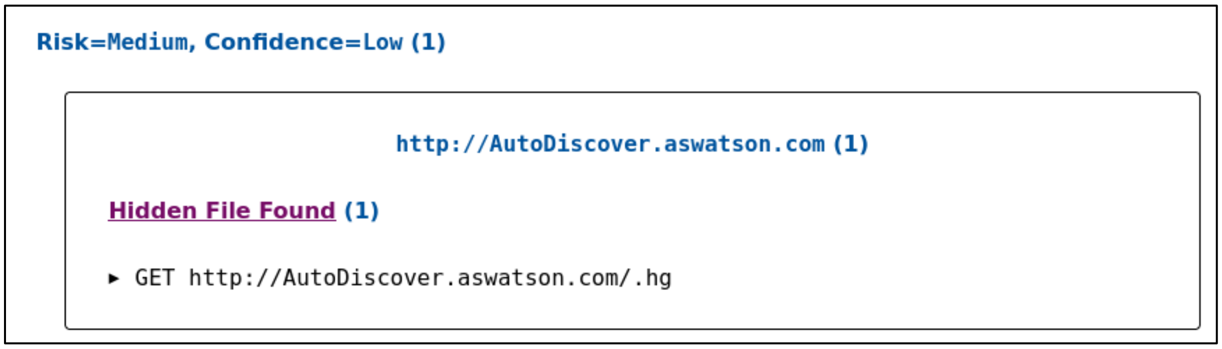
- **Disable Unsafe Inline Content**: Prevent the use of inline scripts and styles by disabling unsafe directives like 'unsafe-inline'.

- **Utilize Nonce or Hash**: For any inline scripts that must be used, implement nonce or hash-based mechanisms to ensure only trusted code executes.

- **Block Mixed Content**: Prevent the loading of insecure HTTP resources over HTTPS connections.
- **Monitor Using Report-Only Mode**: Enable CSP in report-only mode to monitor policy violations before enforcing the rules, ensuring that the policy does not break the site.
- **Set Up Violation Reporting**: Establish a reporting mechanism that logs any CSP violations to a designated endpoint for further review and improvement.
- **Regular Review and Update**: Continuously review and update the CSP to ensure it remains effective as the web application evolves.

These steps ensure that the application has a robust defense against content-based attacks.

2. Tittle – Hidden File Found


Risk=Medium, Confidence=Low (1)

http://AutoDiscover.aswatson.com (1)

Hidden File Found (1)

▸ GET http://AutoDiscover.aswatson.com/.hg

Description

It disclosed the presence of a hidden. hg directory at AutoDiscover.aswatson.com, which was related to the Mercurial Version Control System. Further, this may contain sensitive information such as source code, configurations, credentials, and so on, which can be utilized by an attacker. If it gets exposed, that may allow an attacker to get deeper insight into the system and increase the chance of further exploitation.

However, this may be mitigated either by cleanup of the unwanted version control directories from a production environment or by locking down access via appropriate authentication. This way, sensitive information will not be disclosed to unauthorized users.

Affected components

- **Sensitive Information Disclosure:** This could lead to attackers having access to sensitive project information, like code history, internal changes, and even credentials.

- **Publicly Disclosable Source Code:** Publicly available hg. directories can allow for complete access to source code disclosure; attackers can then find and exploit security vulnerabilities.

- **Poor Authorization Controls:** The so-called unauthorized access to version control files is because of poor authorization controls in place, where sensitive resources are made available to anyone knowing a particular URL.

- **Poor Security:** The existence of a. hg folder indicates poor concern for security best practices; such files must be banned from exposure to the outside world

Impact Assessment

- **Exposed Sensitive Information:** Publicly accessible version control files like the .hg directory have publicly exposed some details of the project-such as source code, configurations, internal documentation-exposing them to attackers, which can be very useful to find vulnerabilities in the system.

- **Increased Risk of Attacks**: More attack vectors could allow an attacker to realize code analysis using version control system access, which might expose security vulnerabilities, configuration flaws, or logical errors that can be used against the system. The risk related to code injections, privilege escalation, and remote code execution would increase manifold.

- **Compliance Exposure:** It might also result in the exposure of sensitive configuration files, which directly violates different security standards related to industries, including PCI DSS, OWASP, and HIPAA. This may involve some legal or financial consequences too.

- **Social Engineering:** The leaked administrative or credential information can be used in social engineering attacks. An attacker can manipulate employees or users into offering further unauthorized access to attackers.

Steps to reproduce

Identify the Target URL:
Use a web browser or command line utility to identify the target URL. This will be http://AutoDiscover.aswatson.com.

Send an HTTP GET Request:
Using a command line utility, like curl or wet, make a GET request for the backdoor
curl -v http://AutoDiscover.aswatson.com/.hg

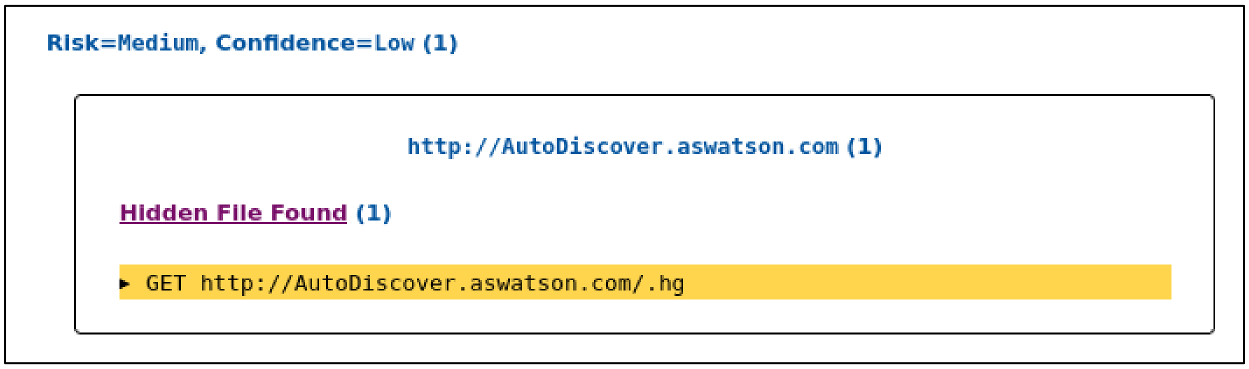
Analyze the Response:
Observe the response of the server. A response indicating a redirect, HTTP 301, or an error status, for instance 404, does not necessarily indicate a vulnerability. If you receive a 200 OK or any other successful response, know that the (hidden) file is accessible.

Existence Check of File:
If it comes in response that the file exists or is redirected to other locations, then this may be because of the possible vulnerability in exposing sensitive files.

Document findings:
by screenshotting the request and response, to include in your vulnerability report, as this will be your proof that the sensitive file was disclosed. These enable you to double-check the existence of potentially sensitive files, which really should not be available to unauthorized users.

Proof of concept (if applicable)

Vulnerability scanning using OWASP ZAP

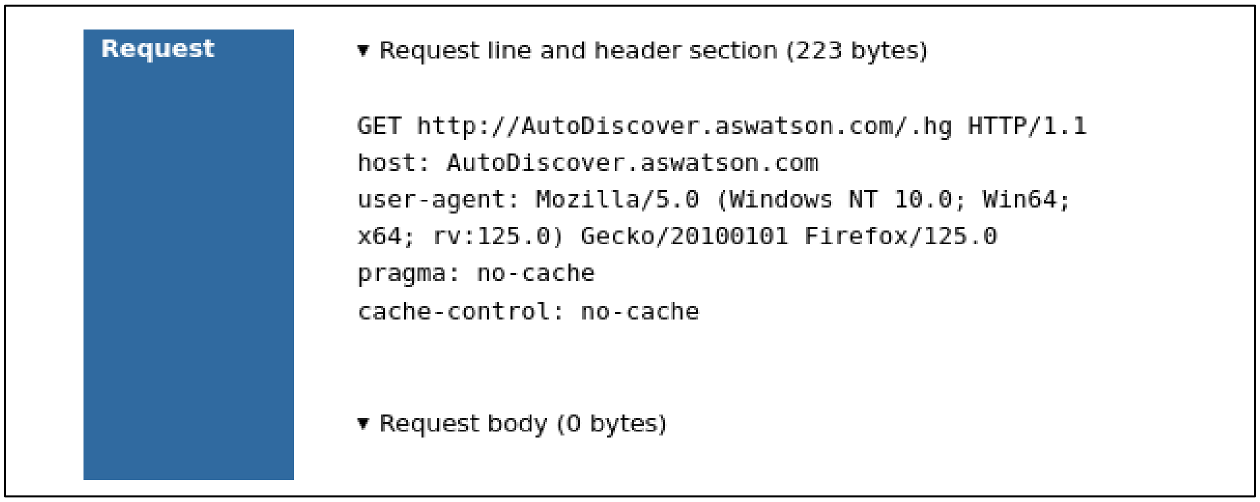

Risk=Medium, Confidence=Low (1)

http://AutoDiscover.aswatson.com (1)

Hidden File Found (1)

▸ GET http://AutoDiscover.aswatson.com/.hg

- **Request**

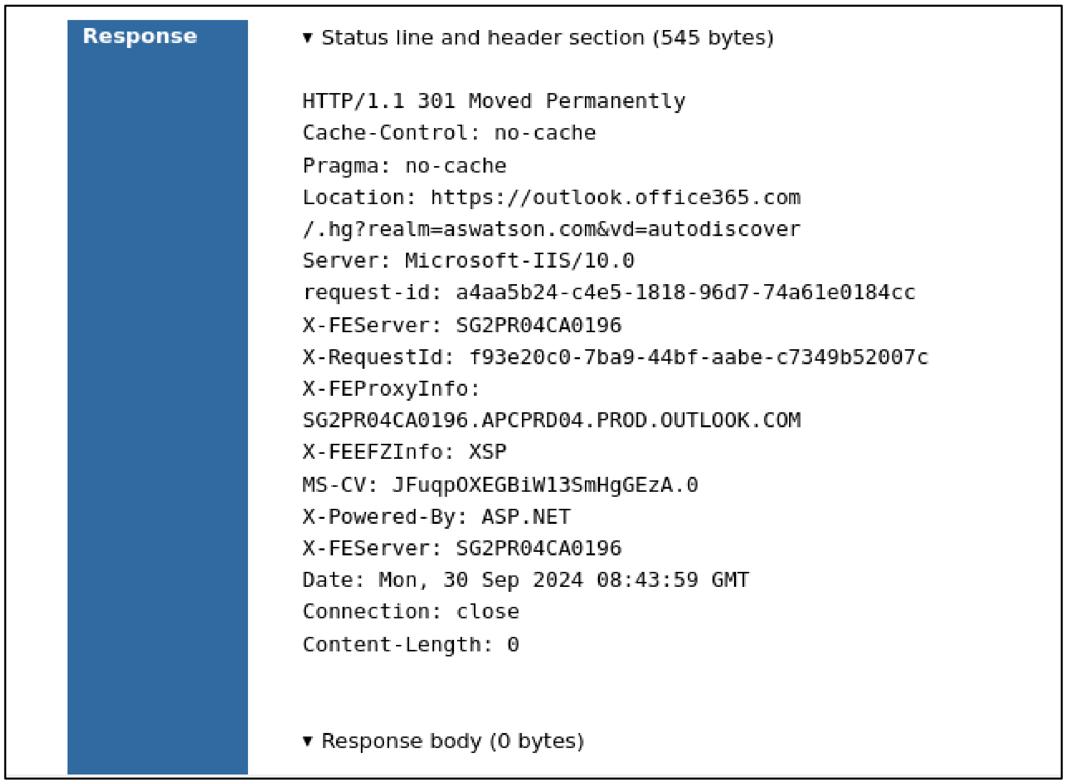

Request

▾ Request line and header section (223 bytes)

```
GET http://AutoDiscover.aswatson.com/.hg HTTP/1.1
host: AutoDiscover.aswatson.com
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64; rv:125.0) Gecko/20100101 Firefox/125.0
pragma: no-cache
cache-control: no-cache
```

▾ Request body (0 bytes)

- **Response**

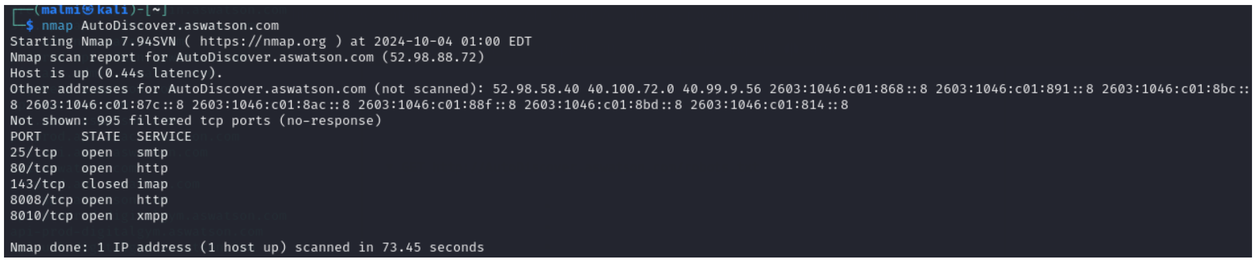


```
HTTP/1.1 301 Moved Permanently
Cache-Control: no-cache
Pragma: no-cache
Location: https://outlook.office365.com
/.hg?realm=aswatson.com&vd=autodiscover
Server: Microsoft-IIS/10.0
request-id: a4aa5b24-c4e5-1818-96d7-74a61e0184cc
X-FEServer: SG2PR04CA0196
X-RequestId: f93e20c0-7ba9-44bf-aabe-c7349b52007c
X-FEProxyInfo:
SG2PR04CA0196.APCPRD04.PROD.OUTLOOK.COM
X-FEEFZInfo: XSP
MS-CV: JFuqpOXEGBiW13SmHgGEzA.0
X-Powered-By: ASP.NET
X-FEServer: SG2PR04CA0196
Date: Mon, 30 Sep 2024 08:43:59 GMT
Connection: close
Content-Length: 0
```

Vulnerability finding using namp

Used command

- nmap AutoDiscover.aswatson.com



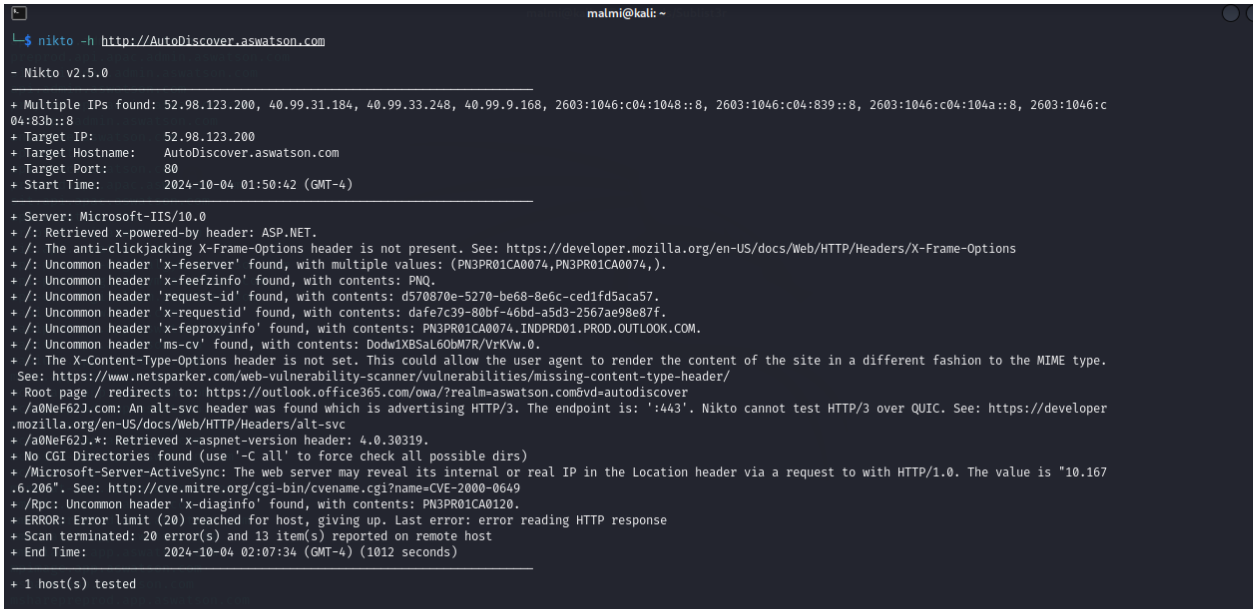
```
┌──(malmi㉿kali)-[~]
└─$ nmap AutoDiscover.aswatson.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-04 01:00 EDT
Nmap scan report for AutoDiscover.aswatson.com (52.98.88.72)
Host is up (0.44s latency).
Other addresses for AutoDiscover.aswatson.com (not scanned): 52.98.58.40 40.100.72.0 40.99.9.56 2603:1046:c01:868::8 2603:1046:c01:891::8 2603:1046:c01:8bc::
8 2603:1046:c01:87c::8 2603:1046:c01:8ac::8 2603:1046:c01:88f::8 2603:1046:c01:8bd::8 2603:1046:c01:814::8
Not shown: 995 filtered tcp ports (no-response)
PORT     STATE  SERVICE
25/tcp   open   smtp
80/tcp   open   http
143/tcp  closed imap
8008/tcp open   http
8010/tcp open   xmpp

Nmap done: 1 IP address (1 host up) scanned in 73.45 seconds
```

Founded open port

- 25/tcp open smtp
- 80/tcp open http
- 143/tcp closed imap
- 8008/tcp open http
- 8010/tcp open xmpp

Vulnerability finding using Nikto



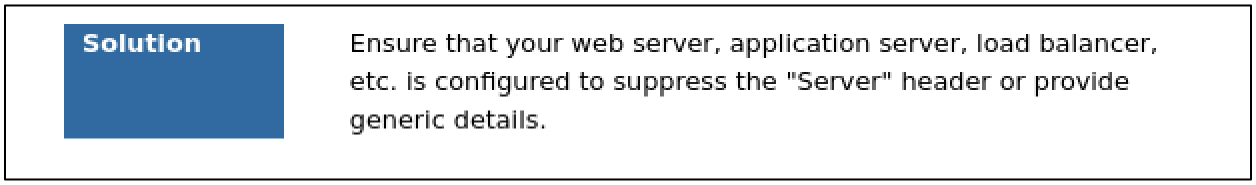
```
malmi@kali: ~
└─$ nikto -h http://AutoDiscover.aswatson.com

- Nikto v2.5.0
---------------------------------------------------------------------------
+ Multiple IPs found: 52.98.123.200, 40.99.31.184, 40.99.33.248, 40.99.9.168, 2603:1046:c04:1048::8, 2603:1046:c04:839::8, 2603:1046:c04:104a::8, 2603:1046:c
04:83b::8
+ Target IP:          52.98.123.200
+ Target Hostname:    AutoDiscover.aswatson.com
+ Target Port:        80
+ Start Time:         2024-10-04 01:50:42 (GMT-4)
---------------------------------------------------------------------------
+ Server: Microsoft-IIS/10.0
+ /: Retrieved x-powered-by header: ASP.NET.
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: Uncommon header 'x-feserver' found, with multiple values: (PN3PR01CA0074,PN3PR01CA0074,).
+ /: Uncommon header 'x-feefzinfo' found, with contents: PNQ.
+ /: Uncommon header 'request-id' found, with contents: d570870e-5270-be68-8e6c-ced1fd5aca57.
+ /: Uncommon header 'x-requestid' found, with contents: dafe7c39-80bf-46bd-a5d3-2567ae98e87f.
+ /: Uncommon header 'x-feproxyinfo' found, with contents: PN3PR01CA0074.INDPRD01.PROD.OUTLOOK.COM.
+ /: Uncommon header 'ms-cv' found, with contents: Dodw1XBSaL6ObM7R/VrKVw.0.
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type.
 See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ Root page / redirects to: https://outlook.office365.com/owa/?realm=aswatson.com&vd=autodiscover
+ /a0NeF62J.com: An alt-svc header was found which is advertising HTTP/3. The endpoint is: ':443'. Nikto cannot test HTTP/3 over QUIC. See: https://developer
.mozilla.org/en-US/docs/Web/HTTP/Headers/alt-svc
+ /a0NeF62J.*: Retrieved x-aspnet-version header: 4.0.30319.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ /Microsoft-Server-ActiveSync: The web server may reveal its internal or real IP in the Location header via a request to with HTTP/1.0. The value is "10.167
.6.206". See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0649
+ /Rpc: Uncommon header 'x-diaginfo' found, with contents: PN3PR01CA0120.
+ ERROR: Error limit (20) reached for host, giving up. Last error: error reading HTTP response
+ Scan terminated: 20 error(s) and 13 item(s) reported on remote host
+ End Time:           2024-10-04 02:07:34 (GMT-4) (1012 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

- **Used command**
  - nikto -h http://AutoDiscover.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.
  - A sensitive file .hg was found

Proposed mitigation or fix



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

Following is some of the proposed mitigations needed based on some of the identified vulnerabilities and security issues found from the Nikto scan result for http://AutoDiscover.aswatson.com:

- **Access Control to Sensitive Files**
  Review configuration to make sure that sensitive files, such as the. hg, are not open to individuals who should not have access. Access should only be provided to internal users or to specific IP addresses. Among the more important configuration directives for file security, as noted on the OWASP Website,

- **Web Security Headers:**
  X-Frame-Options, Content-Security-Policy and Strict-Transport-Security response headers Set. To prevent clickjacking, XSS and man-in-the-middle attacks respectively. See on how to setup on MDN.

- **Update Software:**
  Out of band: Regular server software should in general be updated to the latest stable versions of software to plug known security vulnerabilities. Stick with best practices as given in the CIS Benchmarks.

- **Routine Security Assessments:**
  Consider running periodic regime of security testing and scanning for vulnerabilities using automated detection and mitigation capabilities for security risks and security threat vectors, which can be facilitated by utilizing OWASP ZAP or Burp Suite. More information can be found on the OWASP ZAP webpage.

- **Redirects:**
  The Server should also be checked with respect to any set redirection. The same should be appropriate and must never point to any disclosure of sensitive information or insecure sites. For best practices on HTTP status code and redirect refer to RFC 7231. The above implementation gives the server a better security posture by mitigating the identified vulnerabilities.