

**Don Bosco Institute of Technology**  
**Department of Computer Engineering**

**Academic year – 2021-22**

**Class:** B.E. COMPUTER ENGINEERING

**Subject:** Computational Lab -I

**Course Code:** CSL704

**Experiment Title:** EXPERIMENT 4

**Student Name:** Hayden Cordeiro

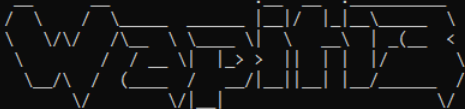
**Roll No.:** 05

**Batch:** D

**Date of Performance:** 31-08-2021

**Date of Submission:** 31-08-2021

```
hayden@DESKTOP-JVKS0LL:~$ wapiti -u "http://vbcsa.ca/login/login.asp"
```



```
Wapiti-3.0.5 (wapiti.sourceforge.io)
[*] Saving scan state, please wait...

Note
=====
This scan has been saved in the file /home/hayden/.wapiti/scans/vbcsa.ca_folder_51d85bb6.db
[*] Wapiti found 1 URLs and forms during the scan
[*] Loading modules:
    backup, brute_login_form, buster, cookieflags, crlf, csp, csrf, drupal_enum, exec, file, htaccess, http_headers
, methods, nikto, permanentxss, redirect, shellshock, sql, ssrf, timesql, wapp, wp_enum, xss, xxe

[*] Launching module csp
CSP is not set

[*] Launching module http_headers
Checking X-Frame-Options :
X-Frame-Options is not set
Checking X-XSS-Protection :
X-XSS-Protection is not set
Checking X-Content-Type-Options :
```

```
hayden@DESKTOP-JVKS0LL: ~
Checking X-Content-Type-Options :
X-Content-Type-Options is not set
Checking Strict-Transport-Security :
Strict-Transport-Security is not set

[*] Launching module cookieflags
Checking cookie : ASPSESSIONIDSADCQTCS
HttpOnly flag is not set in the cookie : ASPSESSIONIDSADCQTCS
Secure flag is not set in the cookie : ASPSESSIONIDSADCQTCS

[*] Launching module exec

[*] Launching module file

[*] Launching module sql

[*] Launching module xss

[*] Launching module ssrf
[*] Asking endpoint URL https://wapiti3.ovh/get_ssrf.php?session_id=m6djb0 for results, please wait...

[*] Launching module redirect

[*] Launching module permanentxss

Report
-----
A report has been generated in the file /home/hayden/.wapiti/generated_report
Open /home/hayden/.wapiti/generated_report/vbsca.ca_09022021_0405.html with a browser to see this report.
hayden@DESKTOP-JVKS0LL:~$
```

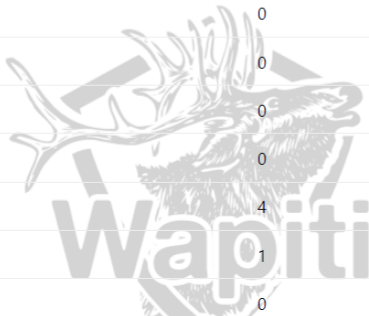
## Wapiti vulnerability report

Target: <http://vbsca.ca/login/login.asp>

Date of the scan: Thu, 02 Sep 2021 04:05:04 +0000. Scope of the scan: folder

### Summary

Category	Number of vulnerabilities found
Backup file	0
Weak credentials	0
CRLF Injection	0
<a href="#">Content Security Policy Configuration</a>	1
Cross Site Request Forgery	0
Potentially dangerous file	0
Command execution	0
Path Traversal	0
Fingerprint web application framework	0
Fingerprint web server	0
Htaccess Bypass	0
<a href="#">HTTP Secure Headers</a>	4
<a href="#">HttpOnly Flag.cookie</a>	1
Open Redirect	0



<a href="#">Secure Flag cookie</a>	1
SQL Injection	0
Server Side Request Forgery	0
Blind SQL Injection	0
Cross Site Scripting	0
XML External Entity	0
Internal Server Error	0
Resource consumption	0
Fingerprint web technology	0

## Content Security Policy Configuration

### 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.

### Vulnerability found in /login/login.asp

Description	HTTP Request	cURL command line
CSP is not set		

### Solutions

Configuring Content Security Policy involves adding the Content-Security-Policy HTTP header to a web page and giving it values to control what resources the user agent is allowed to load for that page.

### Vulnerability found in /login/login.asp

Description	HTTP Request	cURL command line
X-XSS-Protection is not set		

### Vulnerability found in /login/login.asp

Description	HTTP Request	cURL command line
X-Content-Type-Options is not set		

### Vulnerability found in /login/login.asp

Description	HTTP Request	cURL command line
Strict-Transport-Security is not set		

### Solutions

Use the recommendations for hardening your HTTP Security Headers.

### References

- [Netsparker: HTTP Security Headers: An Easy Way to Harden Your Web Applications](#)
- [KeyCDN: Hardening Your HTTP Security Headers](#)
- [OWASP: HTTP SECURITY HEADERS \(Protection For Browsers\) \(PDF\)](#)

## Solutions

Configuring Content Security Policy involves adding the Content-Security-Policy HTTP header to a web page and giving it values to control what resources the user agent is allowed to load for that page.

## References

- [Mozilla: Content Security Policy \(CSP\)](#)
- [OWASP: Content Security Policy Cheat Sheet](#)
- [OWASP: How to do Content Security Policy \(PDF\)](#)

---

## HTTP Secure Headers

### Description

HTTP security headers tell the browser how to behave when handling the website's content.

### Vulnerability found in /login/login.asp

Description	HTTP Request	cURL command line
X-Frame-Options is not set		

### Vulnerability found in /login/login.asp

Description	HTTP Request	cURL command line
X-XSS-Protection is not set		

---

### HttpOnly Flag cookie

#### Description

HttpOnly is an additional flag included in a Set-Cookie HTTP response header. Using the HttpOnly flag when generating a cookie helps mitigate the risk of client side script accessing the protected cookie (if the browser supports it).

### Vulnerability found in /login/login.asp

Description	HTTP Request	cURL command line
HttpOnly flag is not set in the cookie : ASPSESSIONIDSADCQTCS		

## Solutions

While creation of the cookie, make sure to set the HttpOnly Flag to True.

## References

- [OWASP: Testing for Cookies Attributes](#)
- [OWASP: HttpOnly](#)

---

### Secure Flag cookie

#### Description

The secure flag is an option that can be set by the application server when sending a new cookie to the user within an HTTP Response. The purpose of the secure flag is to prevent cookies from being observed by unauthorized parties due to the transmission of a the cookie in clear text.

### Vulnerability found in /login/login.asp

Description	HTTP Request	cURL command line
Secure flag is not set in the cookie : ASPSESSIONIDSADCQTCS		

## Solutions

When generating the cookie, make sure to set the Secure Flag to True.

## References

- [OWASP: Testing for Cookies Attributes](#)
- [OWASP: Secure Cookie Attribute](#)

Conclusion : Hence Web Vulnerabilities were scanned in the websites using wapti and desired results were achieved.