

# **Sri Lanka Institute of Information Technology**



## **Bug Bounty Report 04**

**WS Assignment**  
**IE2062 – Web Security**

**IT23159730**

**W.H.M.S.R.Bandara**

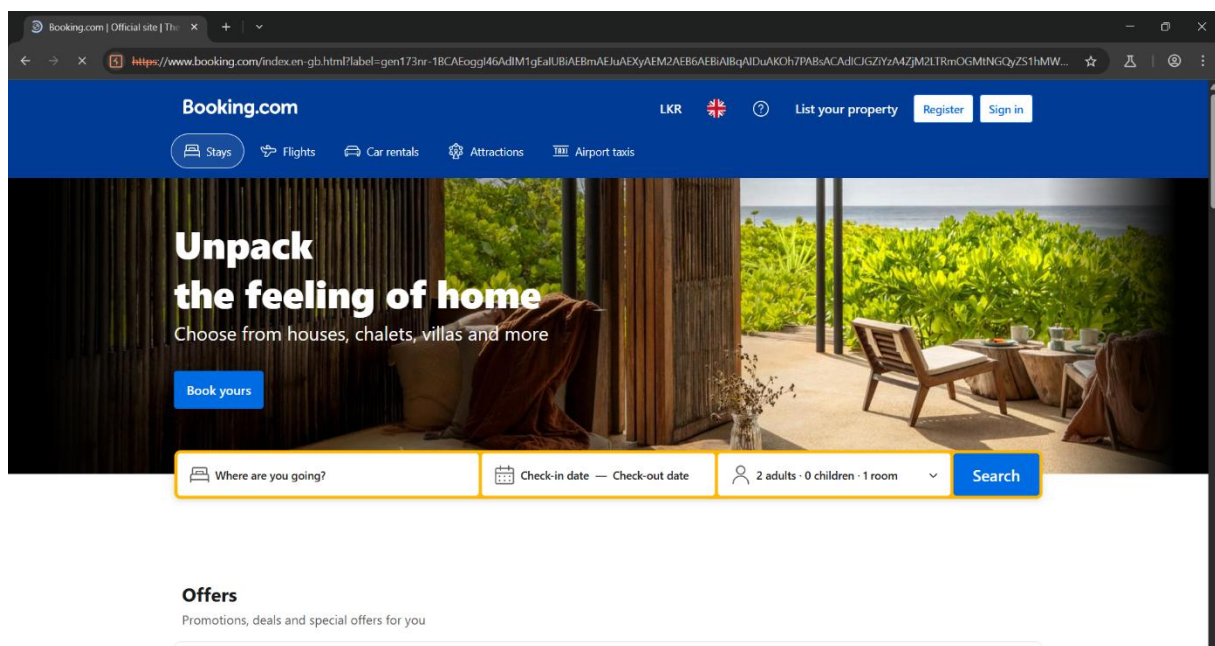
# Vulnerability Title

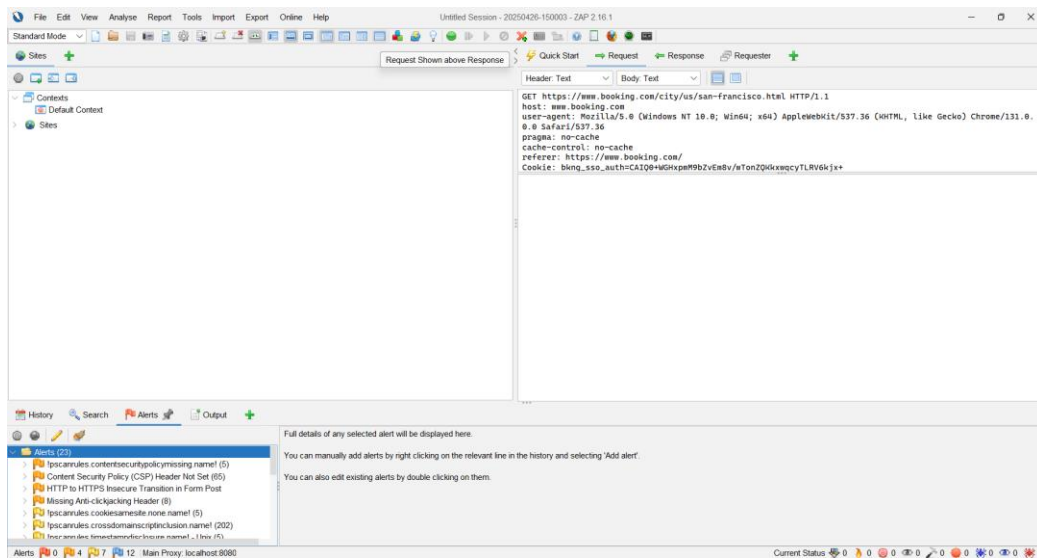
Path Traversal Testing on [www.booking.com](http://www.booking.com)

## Vulnerability Description

Path Traversal vulnerabilities occur when a program allows user-inputted information to read beyond directories and files that do not exist within the anticipated directory tree. Such an attack would result in unauthorized access to confidential files, server setup files, or system files.

OWASP ZAP was run in automated mode initially against <http://www.booking.com> and didn't display any high-priority vulnerabilities. To ensure comprehensive coverage, manual penetration testing was subsequently conducted using Burp Suite with an emphasis on potential path traversal vectors. Manual exploitation of directory traversal with specially crafted payloads was unsuccessful. The server correctly validated and processed the user input, preventing unauthorised access to files.

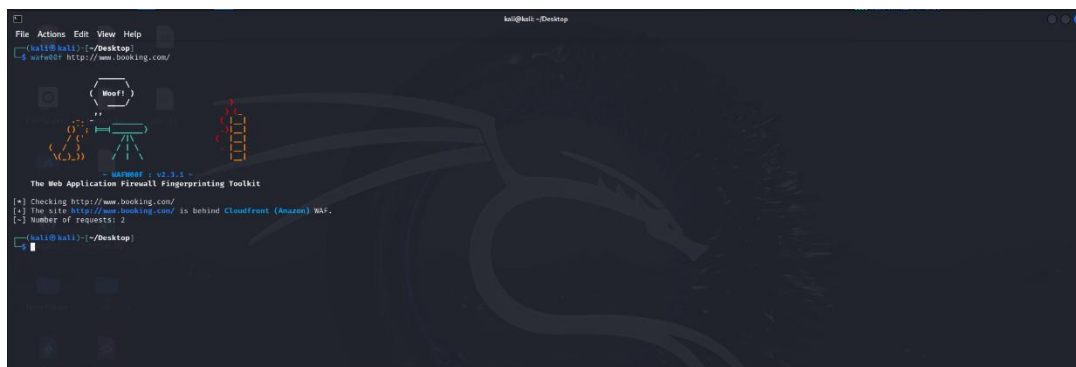




## Other scans

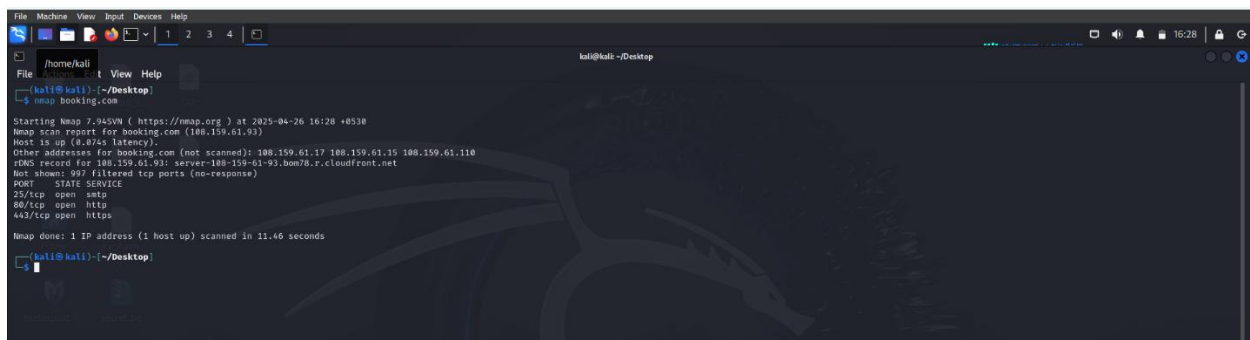
### Firewall Detection.

Before the automated scan with OWASP ZAP (2.16.0), Identify the Firewall by using Wafw00f Tool.



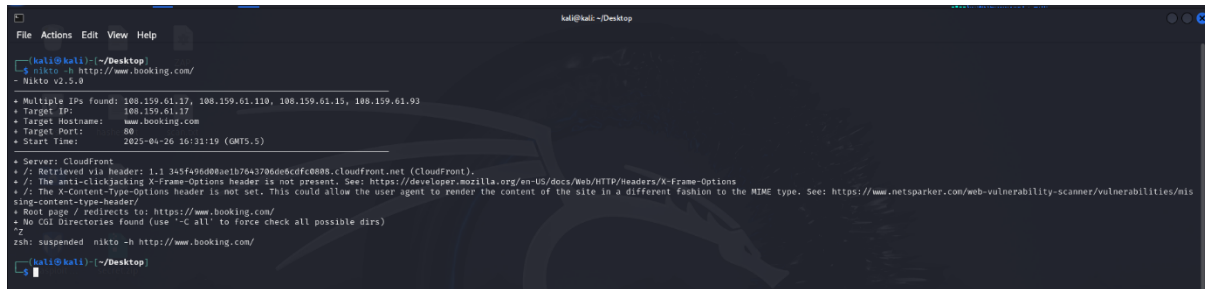
## Open Ports Detection

Done the nmap scan for detect open ports.



# Scanning for common Issues

These scans done by with Nikto.



```
kali@kali: ~/Desktop
$ nikto -h http://www.booking.com/
- Nikto v2.5.0

+ Multiple IPs found: 108.159.61.17, 108.159.61.110, 108.159.61.15, 108.159.61.93
+ Target IP: 108.159.61.17
+ Target Hostname: www.booking.com
+ Target Port: 80
+ Start Time: 2025-04-26 16:31:19 (GMT+5.5)

+ Server: CloudFront
+ /: Retrieved via Header: 1.1 3a5fa0d08a1b76a278d8dc0f0888.cloudfront.net (CloudFront).
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: 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/mis
xing-content-type-header/
+ Root page / redirects to: https://www.booking.com/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ 2
zsh: suspended nikto -h http://www.booking.com/
kali@kali: ~/Desktop
```

There is no vulnerabilities found on these scans. So I manually test the Path Traversal Attack using Burp Suite Community Edition.

## Affected Components

- **Domain:** [www.booking.com](https://www.booking.com)
- **Tested Endpoints:** Various GET and POST parameters, URL paths
- **Service:** Web Application
- **Environment:** Public Production

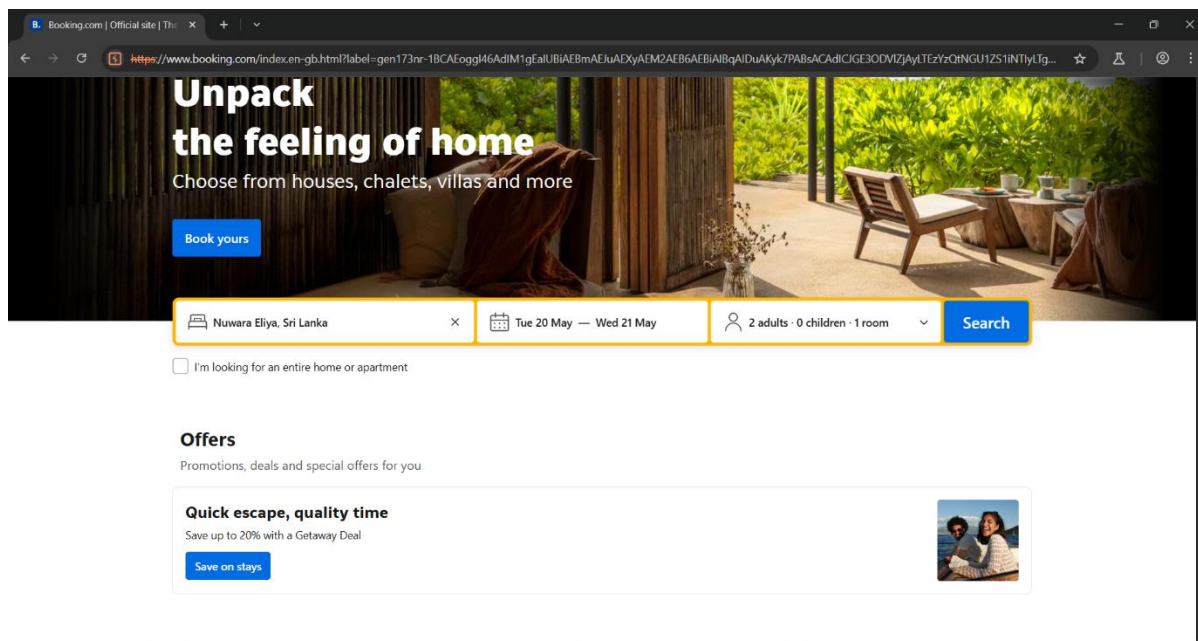
## Impact Assessment

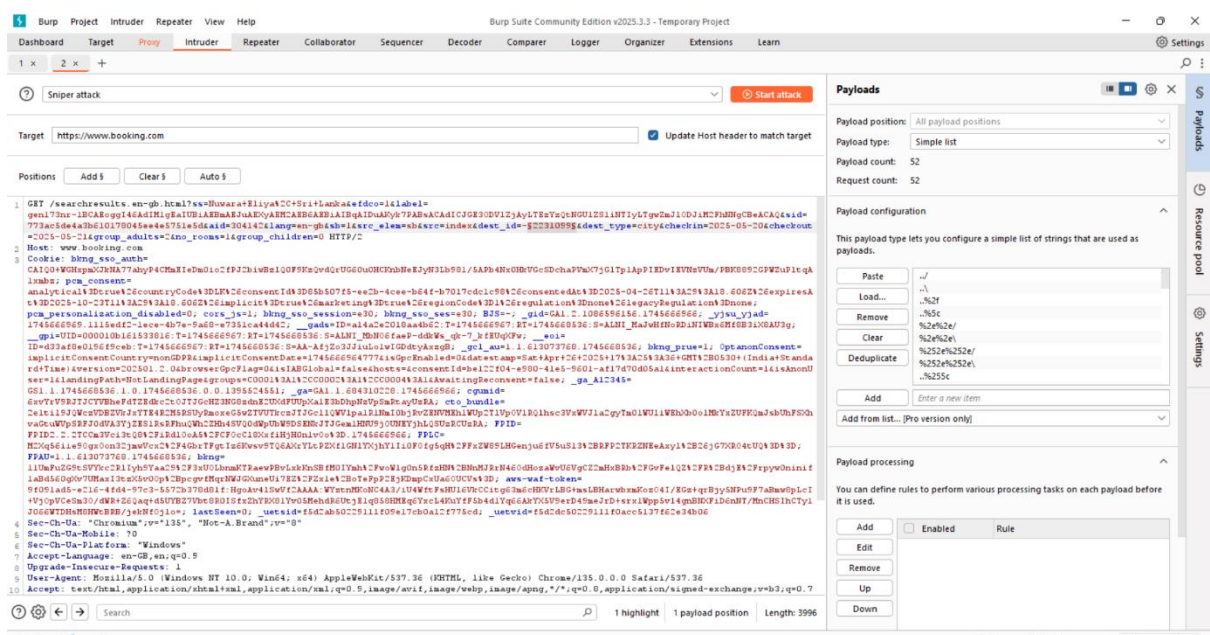
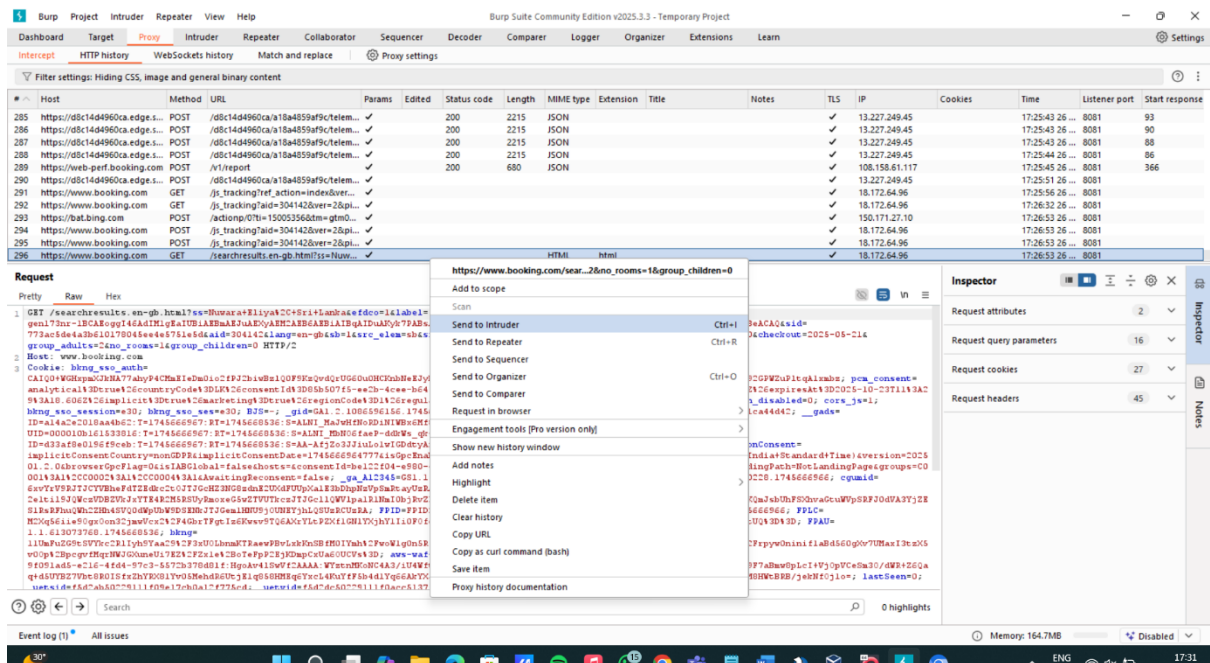
No path traversal vulnerabilities having been discovered, there is no direct impact. In the case such a vulnerability does exist or is introduced in the future, the potential threats would be:

- **Unauthorized File Access:** Attackers may access sensitive files (like /etc/passwd, readable configuration files, or the application source code) that can lead to information disclosure.
- **System Compromise:** If an attacker has read access to the system files, they could launch further attacks, like privilege escalation or remote code execution.
- **Data Exposure:** Exposure of sensitive data (e.g., database credentials in configuration files) could lead to broader data breaches.
- **Business Impact:** Business Impact By being a travel booking platform with personal data and payment information, it would lose money, face lawsuits, and lose customers who had their information compromised or stolen.
- **Reputation Risk:** If exploited, negative publicity could be generated, and the user will lose trust in the platform.

## Steps to Reproduce

1. Perform an Automated Scan with OWASP ZAP
  - Target: <http://www.booking.com>
  - Result: No High-risk vulnerabilities detected.
2. Set Up Burp Suite for Manual Testing.
  - Configure Burp Suite to intercept requests to <http://www.booking.com/>.
  - Identify a form or input field (e.g. search form, filtering option) to test for Path traversal.
  -
3. Use Burp Suite Intruder to Inject Payloads.
  - Capture a request to the target form in Burp Suite.
  - Send the request to the Intruder tool and select the parameter to test.





#### 4. Payload Configuration.

- Use 50 Path Traversal Payloads in the “Payloads” tab.

#### 5. Attack Results.

- Path traversal payloads were accepted (HTTP 200 OK).
- No immediate WAF protection blocked the traversal attempts.
- No sensitive files (like /etc/passwd) were successfully accessed.
- The server shows a potential weakness but no confirmed file disclosure yet.
- Further exploration needed (e.g., finding endpoints that return file content).



## Proof of Concept Screenshot

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
0		200	397			1500746	
1	./	200	397			1503890	
2	../	200	395			1503886	
3	../%2f	200	387			1503892	
4	../%5c	200	398			1503966	
5	../%2e%2e/	200	384			1503886	
6	../%2e%2e%2e/	200	402			1503886	
7	../%252e%252e/	200	396			1504099	
8	../%252e%252e%252e/	200	378			1504111	
9	../%255c	200	412			1503946	
10	../%c0%af	200	431			1504002	
11	../%c1%9c	200	429			1503998	
12	../%00	200	430			1504273	
13	../%00/	200	397			1504138	
14	../%00/	200	398			1504114	
15	../%00/	200	397			1504338	
16	../%00/	200	441			1504468	
17	../%00/	200	400			1500958	
18	../%2e%2e%2f	200	445			1503884	
19	../%2e%2e%5c	200	423			1503966	
20	../%2e%2e%252f	200	454			1504023	
21	../%2e%2e%255c	200	438			1503943	
22	../%2e%2e%5c	200	448			1503898	
23	../%2e%2e%2f	200	431			1503892	
24	../%252e%252e%252e%255c	200	408			1504068	
25	../%252e%252e%252e%252f	200	386			1504060	
26	../%255c%252e%252f	200	400			1504349	
27	../%2f%2f%2f	200	412			1504186	
28	../%00	200	385			1504050	
29	../%00/	200	464			1503972	
30	../%00/	200	438			1503976	
31	../%00/	200	409			1504468	
32	../%00/	200	414			1504792	

## Proposed Mitigation or Fix

While there were no path traversal vulnerabilities discovered, the following countermeasures are suggested to ensure ongoing protection from possible path traversal attacks and preserve a solid security stance:

- Sanitize and Normalize File Paths.
- Implement Strict Input Validation.
- Periodic Security Testing.
- Web Application Firewall (WAF).

## Conclusion.

The testing with [www.booking.com](https://www.booking.com) did not yield any path traversal vulnerabilities, suggesting that the application likely possesses strong security measures like path normalization, input validation, or access controls. Manual testing using Burp Suite validated the lack of exploitable vulnerabilities since all attempts to retrieve forbidden files returned HTTP 200 responses with large response lengths, most likely reflecting the return of a default webpage or error page. Because the application deals with sensitive user data, it's imperative that an additional layer of defense is put in place to prevent path traversal and other web vulnerabilities. The recommendations that follow in this report are intended to offer further preventative measures to strengthen the application's defenses. Constant monitoring, secure coding, and regular audits are what will ultimately ensure the site's security is sound against future incoming attacks.