

Sri Lanka Institute of Information Technology



Cyber Security Assignment (2025) XSS Vulnerabilities and CAPTCHA Bypass

Bug Bounty Report- 02
IT23363366

TABLE OF CONTENT

1. Title	3
2. Scope and Objective	3
3. Enumeration and Reconnaissance	3
3.1. Tools Used	3
3.2. Steps Taken.....	4
4. Vulnerability Description	5
4.1 freshdesk.bhasha.lk – CAPTCHA Validation.....	5
4.2 care.bhasha.lk – Reflected XSS Testing	5
5. Affected Component.....	5
5.1 freshdesk.bhasha.lk – CAPTCHA Validation.....	5
5.2 care.bhasha.lk – Reflected XSS.....	5
6. Impact Assessment	6
6.1 Reflected XSS care.bhasha.lk	6
6.2 CAPTCHA Validation freshdesk.bhasha.lk.....	6
7. Proof of Concept	7
7.1 Subdomain: freshdesk.bhasha.lk	7
7.2 Subdomain: care.bhasha.lk	10
8. Proposed Mitigation.....	13
8.1 Reflected XSS care.bhasha.lk	13
8.2 CAPTCHA Validation freshdesk.bhasha.lk.....	13
9. Conclusion	13
10. References.....	13

1. Title

Report Title: XSS Vulnerabilities and CAPTCHA Bypass

Reported By: Raahim Mahmooth

Tested on: <https://care.bhasha.lk>

Platform: <https://bugzero.io>

2. Scope and Objective

The web application penetration test was conducted on **bhasha.lk** and its associated subdomains. The objective was to perform a series of **passive and manual security tests** to identify vulnerabilities within the website and subdomains, without exploiting them. Specifically, the tests focused on **input validation**, **authentication mechanisms**, **user enumeration**, **XSS vulnerabilities**, and **CAPTCHA bypass**.

- Identify potential vulnerabilities in the **bhasha.lk** website and its subdomains.
- Conduct security testing to assess the effectiveness of **CAPTCHA mechanisms**, input validation, and user enumeration protection.
- Test for common web application vulnerabilities, including **Cross-Site Scripting (XSS)** and **broken authentication**.
- Ensure no DoS or full exploitation attacks were attempted during the testing phase, in line with responsible disclosure practices.

3. Enumeration and Reconnaissance

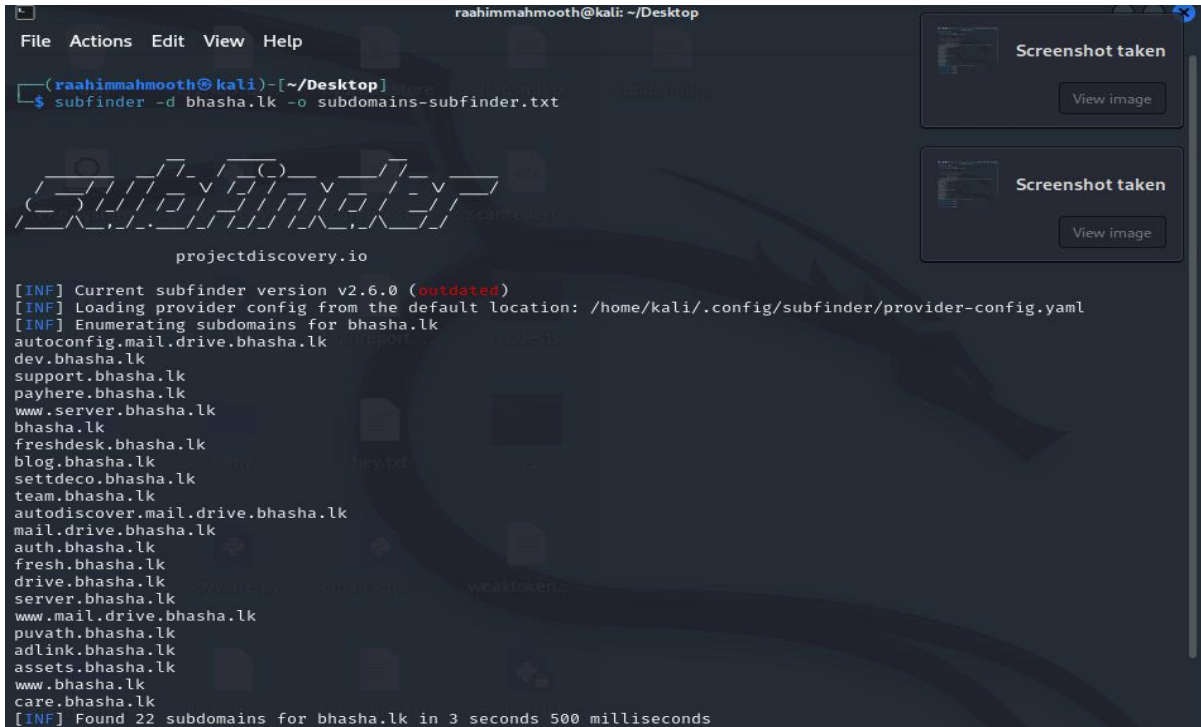
The initial phase of the penetration testing involved **subdomain enumeration** to identify potential attack surfaces within the **bhasha.lk** domain.

3.1. Tools Used

- Subfinder: Used for passive subdomain enumeration.
- Crt.sh: Searched for SSL/TLS certificates to reveal historical subdomains.
- Burp Suite: For intercepting and analyzing requests
- XSSStrike: Automated tool for XSS injection
- Wfuzz: To load payload for XSS injection
- Manual browser testing

3.2. Steps Taken

Subdomain Enumeration: Using tools like Subfinder and crt.sh, I discovered over 22 subdomains related to the main domain “*bhasha.lk*”. These included domains such as *care.bhasha.lk*, *freshdesk.bhasha.lk*, *dev.bhasha.lk*, and *mail.drive.bhasha.lk*. Some of these led to login portals or services protected by CAPTCHA and authentication mechanisms.



```
raahimmahmooth@kali: ~/Desktop
File Actions Edit View Help
(raahimmahmooth@kali)-[~/Desktop]
$ subfinder -d bhasha.lk -o subdomains-subfinder.txt

Subfinder
projectdiscovery.io

[INF] Current subfinder version v2.6.0 (outdated)
[INF] Loading provider config from the default location: /home/kali/.config/subfinder/provider-config.yaml
[INF] Enumerating subdomains for bhasha.lk
autoconfig.mail.drive.bhasha.lk
dev.bhasha.lk
support.bhasha.lk
payhere.bhasha.lk
www.server.bhasha.lk
bhasha.lk
freshdesk.bhasha.lk
blog.bhasha.lk
settdeco.bhasha.lk
team.bhasha.lk
autodiscover.mail.drive.bhasha.lk
mail.drive.bhasha.lk
auth.bhasha.lk
fresh.bhasha.lk
drive.bhasha.lk
server.bhasha.lk
www.mail.drive.bhasha.lk
puvath.bhasha.lk
adlink.bhasha.lk
assets.bhasha.lk
www.bhasha.lk
care.bhasha.lk
[INF] Found 22 subdomains for bhasha.lk in 3 seconds 500 milliseconds
```

Also to test recently activate subdomains, explored Crt.sh: <https://crt.sh/?q=%25.bhasha.lk>

During manual analysis, most of the discovered subdomains returned server errors or were inaccessible. However, two subdomains *care.bhasha.lk* and *freshdesk.bhasha.lk* responded successfully and appeared functional. As a result, further testing was focused on these two targets

4. Vulnerability Description

4.1 freshdesk.bhasha.lk – CAPTCHA Validation

During testing of the login portal at *freshdesk.bhasha.lk*, the CAPTCHA mechanism was evaluated for potential weaknesses. CAPTCHA is intended to protect against automated login attempts and brute-force attacks. The focus was on identifying whether the CAPTCHA could be bypassed or reused.

4.2 care.bhasha.lk – Reflected XSS Testing

The search functionality on *care.bhasha.lk* was tested for reflected Cross-Site Scripting (XSS) vulnerabilities. The *term* parameter in the search query is user-controlled and was tested using various manual and automated payloads.

5. Affected Component

5.1 freshdesk.bhasha.lk – CAPTCHA Validation

- Endpoint: <https://freshdesk.bhasha.lk/api/v2/login>
- Parameter: captcha

5.2 care.bhasha.lk – Reflected XSS

- Endpoint: <https://care.bhasha.lk/support/search?term=>
- Parameter: term

6. Impact Assessment

6.1 Reflected XSS *care.bhasha.lk*

The search functionality on *care.bhasha.lk* was tested for reflected XSS vulnerabilities by targeting the `term` parameter. If this input were improperly handled, it could allow attackers to inject malicious scripts, potentially leading to session hijacking, phishing, or unauthorized access. However, no payloads were reflected or executed during testing, suggesting that the application either sanitizes the input or does not allow user-controlled input to be processed in a way that would make XSS feasible.

6.2 CAPTCHA Validation *freshdesk.bhasha.lk*

The CAPTCHA mechanism in place on *freshdesk.bhasha.lk* was tested to ensure that it could not be bypassed or replayed. If the CAPTCHA validation were not properly enforced, automated login attempts could occur, risking brute-force attacks or account enumeration. During testing, attempts to manipulate or reuse CAPTCHA tokens were rejected by the server, confirming that server-side validation is in place, and replay attacks are mitigated effectively.

7. Proof of Concept

7.1 Subdomain: freshdesk.bhasha.lk

7.1.1 CAPTCHA Validation Testing

Overview

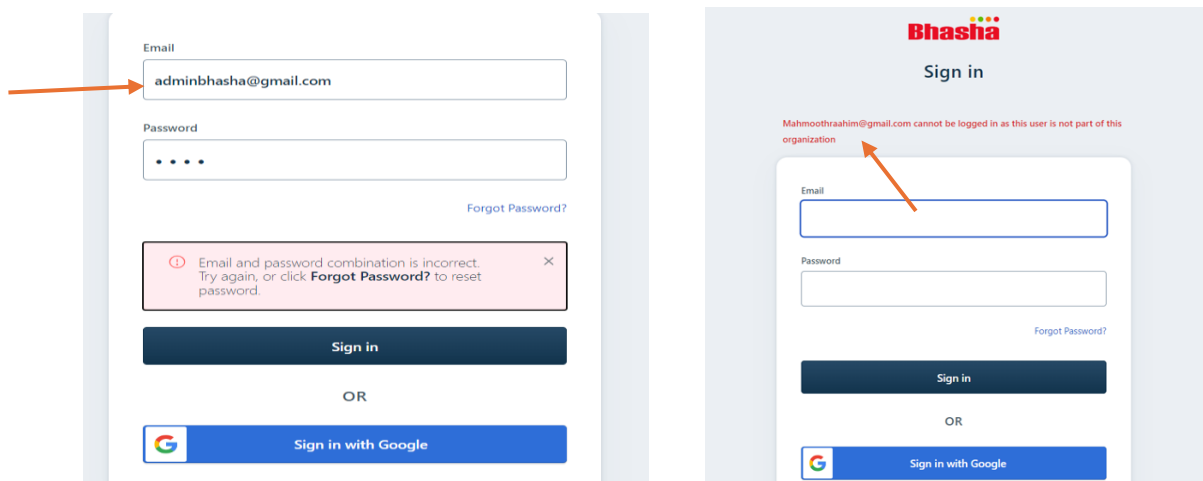
The subdomain `freshdesk.bhasha.lk` serves as a login portal intended for internal use by employees. The authentication form includes fields for an email address, password, and a Google reCAPTCHA challenge. This CAPTCHA mechanism is implemented to prevent automated abuse, such as brute-force login attempts or credential stuffing.

Testing Methodology

- **Initial Manual Tests**

Test credentials were submitted to observe error handling and to determine if user enumeration was possible:

- `mahmoothraahim@gmail.com`: Returned an error indicating that the user is not part of the organization.
- `adminbasha@gmail.com`: Returned a generic error message stating that the user and password do not match.



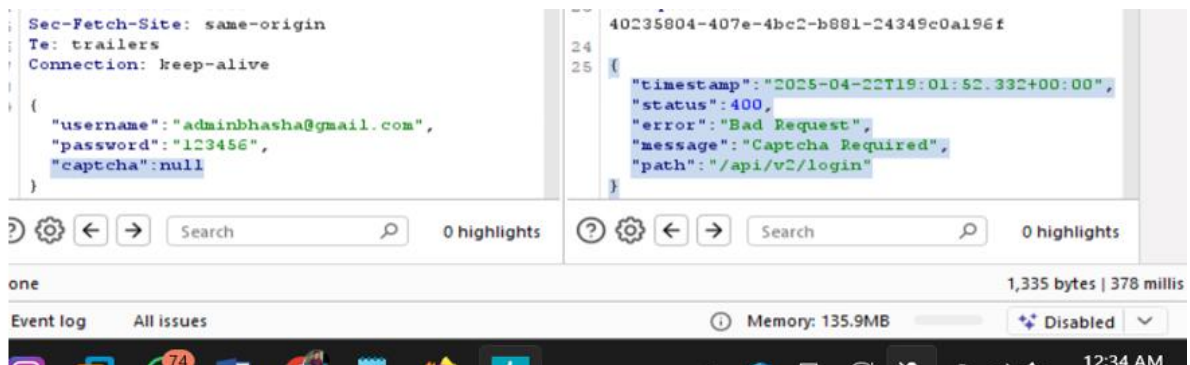
- **Intercepted Request Analysis**

Login requests were intercepted and modified using Burp Suite. The `captcha` field was altered with various test values such as:

```
{
  "username": "adminbhasha@gmail.com",
  "password": "123456",
  "captcha": "null"
}
```

The response consistently returned:

```
{
  "status": 400,
  "error": "Bad Request",
  "message": "Invalid Recaptcha"
}
```



- **CAPTCHA Bypass Attempts**

The following values were tested in place of the CAPTCHA token:

- `"captcha": "test"`
- **With no captcha**



All resulted in failed login attempts with HTTP status code 400, confirming proper validation.

- **Replay Attack Testing**

A valid CAPTCHA token was captured via the browser and replayed in a subsequent login attempt using:

The first screenshot shows a successful POST request to `/recaptcha/api2/userverify` with a valid token. The response is a 200 OK status with JSON data. The second screenshot shows a subsequent login attempt where the same token is reused. The response is a 400 status with an error message: "Bad Request", "Invalid Recaptcha".

```

1 POST /recaptcha/api2/userverify?k=6Lf0obwUAAAAAGWFjvijs2RS_i_yn-BuSHjZNMCF HTTP/2
2 Host: www.recaptcha.net
3 Cookie: _GBCAPTCHA=
4 OSlicevGvYLDJ00RTHcPUDq8qHq8Vp7_POTSDlWp6Kf5SHHc100cp4TYGcPjYcTzPToW8K3hLPdyY
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:13.0) Gecko/20100101 Firefox/137.0
6 Accept: */*
7 Accept-Language: en-US,en;q=0.5
8 Accept-Encoding: gzip, deflate, br
9 Content-Type: application/x-www-form-urlencoded;charset=utf-8
10 Content-Length: 8144
11 Origin: https://www.recaptcha.net
12 Referer: https://www.recaptcha.net/recaptcha/api2/hframe?hl=en&v=IcIkQ1GB1JDHuTkCh1T3zHpBak=6Lf0obwUAAAAAGWFjvijs2RS_i_yn-BuSHjZNMCF
13 Sec-Fetch-Dest: empty
14 Sec-Fetch-Mode: cors
15
16 HTTP/2 200 OK
17 Content-Type: application/json; charset=utf-8
18 Vary: Sec-Fetch-Dest, Sec-Fetch-Mode, Sec-Fetch-Site
19 Cache-Control: no-cache, no-store, max-age=0, must-revalidate
20 Pragma: no-cache
21 Expires: Mon, 01 Jan 1990 00:00:00 GMT
22 Date: Tue, 22 Apr 2025 19:15:39 GMT
23 X-Content-Type-Options: nosniff
24 Cross-Origin-Resource-Policy: same-site
25 Report-To:
26 {
27   "group": "coop_38fac5d5b82543fc4729580d18ff2d3d", "max_age": 2592000, "endpoints": [{"url": "https://csp.v
28   ithgoogle.com/csp/report-to/38fac5d5b82543fc4729580d18ff2d3d"}]}
29
30 Server: ESF
31 X-Xss-Protection: 0
32
33 {
34   "captcha": "captcha_token_here"
35 }
36
37 Content-Length: 137
38 Origin: https://freshdesk.bhasha.lk
39 Sec-Fetch-Dest: empty
40 Sec-Fetch-Mode: cors
41 Sec-Fetch-Site: same-origin
42 Te: trailers
43
44 {
45   "username": "adminbhasha@gmail.com",
46   "password": "123456",
47   "captcha": "IcIkQ1GB1JDHuTkCh1T3zHpBak=6Lf0obwUAAAAAGWFjvijs2RS_i_yn-BuSHjZNMCF"
48 }
49
50 X-Request-Id: 7a7aa0c6-e45a-4ec5-a21a-90bd4318b9f0
51
52 {
53   "timestamp": "2025-04-22T19:36:33.266+00:00",
54   "status": 400,
55   "error": "Bad Request",
56   "message": "Invalid Recaptcha",
57   "path": "/api/v2/login"
58 }

```

The reused token was rejected, indicating that tokens are either time-sensitive or tied to specific sessions.

Conclusion

the CAPTCHA system in place on `freshdesk.bhasha.lk` is implemented correctly and validated server-side. Bypass and replay attempts were unsuccessful. As a result, the CAPTCHA serves as an effective mitigation against automated brute-force attacks and user enumeration.

Refer to appended screenshots for request/response examples.

7.1.2 Reflected Cross-Site Scripting (XSS) Testing – redirect-Uri Parameter

Overview

The login endpoint on `freshdesk.bhasha.lk` includes a query parameter **redirect-Uri**, which may be used for redirection after unsuccessful login. The parameter was tested for possible reflected XSS vulnerabilities.

Testing Methodology

- The following URLs were manually accessed to test the behavior of the redirection logic:
 - Normal redirect:

`https://freshdesk.bhasha.lk/login?redirect_uri=https%3A%2F%2Ffreshdesk.bhasha.lk%2F`

- XSS payload attempt:

`https://freshdesk.bhasha.lk/login?redirect_uri=<script>alert('XSS')</script>`

Results

- No alert box or JavaScript execution was triggered.
- The XSS payload was not rendered in the HTML response.
- The application either sanitized the input or prevented execution through strict redirect validation.
- The page redirected back to the login screen without processing or reflecting the script.

Conclusion

The **redirect-Uri** parameter is handled securely. The application does not reflect or execute arbitrary input passed through this parameter, confirming that reflected XSS is not present in this context.

7.2 Subdomain: care.bhasha.lk

7.2.1 Reflected XSS Testing – Search Input Field

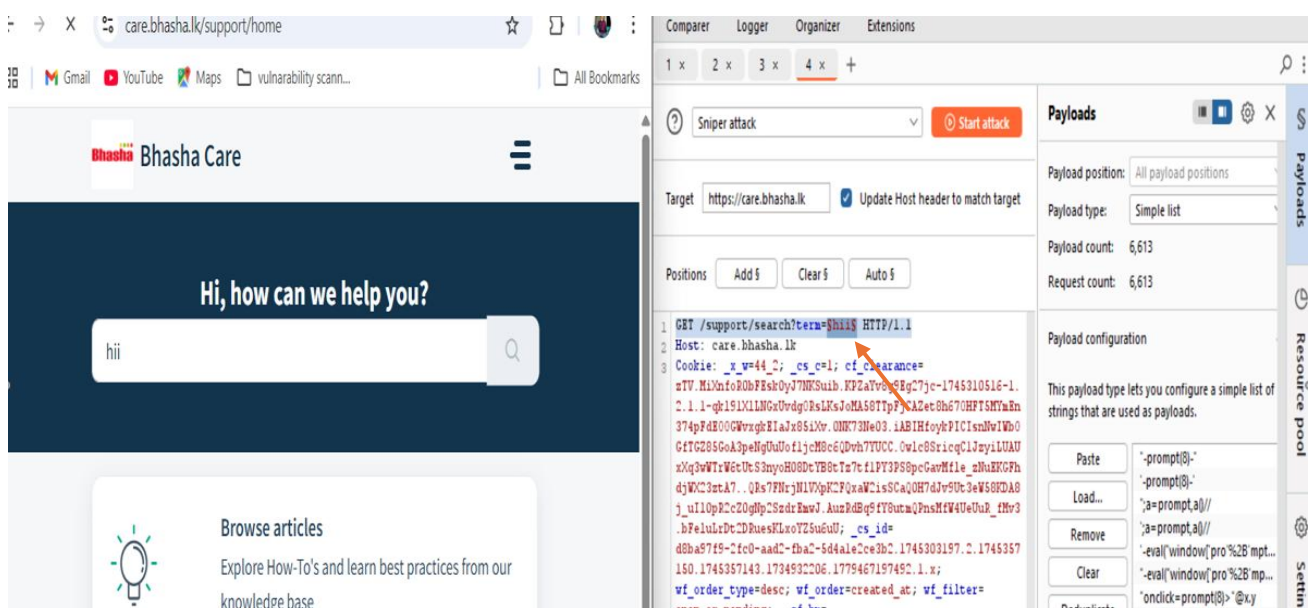
Overview

The subdomain `care.bhasha.lk` includes a public-facing search functionality, allowing users to query products or services via a `term` parameter. This input field became a candidate for reflected Cross-Site Scripting (XSS) testing due to its user-controlled input.

Testing Methodology

- **Step 1: Manual Inspection via Burp Suite**

The search request was captured using Burp Suite and analyzed:



```
GET /support/search?term=hii HTTP/1.1
Host: care.bhasha.lk
```

The `term` parameter was identified as potentially injectable, and its value was targeted for XSS payloads.

- **Step 2: Burp Suite Intruder Attack**

The request was sent to Intruder, and the `term` parameter was marked for fuzzing. A list of 6000 XSS payloads was loaded and executed.

- Result: The application responded slowly and inconsistently.
- No payloads were reflected or executed.
- Status codes remained normal or returned errors, indicating that either input was being filtered or the backend was discarding unexpected input.



- **Step 3: Automated XSS Payload Testing with XSSStrike**

The page was tested using XSSStrike:

```
python3 xssstrike.py -u https://care.bhasha.lk/support/search?term=XSS
```

- Result: No reflected or stored XSS vulnerabilities were identified by the tool.
- Input values were not echoed back into the response, nor was any script execution observed.

```
(raahimmahmooth@kali) - [~/XSSStrike]
$ python3 xssstrike.py -u "https://care.bhasha.lk/support/search?term=XSS"

XSSStrike v3.1.5

[~] Checking for DOM vulnerabilities
[+] WAF Status: Offline
[!] Testing parameter: term
[!] Reflections found: 5
[~] Analysing reflections
[~] Generating payloads
[!] Payloads generated: 4608
^Z Progress: 64/460808C^[[C^[[C^[[C^[[C^[[C~] Progress: 28/4608
zsh: suspended python3 xssstrike.py -u "https://care.bhasha.lk/support/search?term=XSS"
```

- **Step 4: Manual Fuzzing Using wfuzz**

A manual test using wfuzz was conducted with a custom wordlist of payloads:

```
Wfuzz -u "https://care.bhasha.lk/support/search?term=FUZZ" -w payload.txt
```

- Result: The server returned 404 not found for the vast majority of payloads.
- This suggests that invalid input is either not routed correctly or filtered before processing.

```
(raahimmahmooth@kali)-[~/XSStrike]
$ wfuzz -u "https://care.bhasha.lk/support/search?term=FUZZ" -w payload.txt
*****
* Wfuzz 3.1.0 - The Web Fuzzer
*****

Target: https://care.bhasha.lk/support/search?term=FUZZ
Total requests: 6613

ID      Response  Lines  Word    Chars    Payload
-----
000000001: 403        93 L    349 W    4517 Ch  ""-prompt(8)-"
000000025: 403        93 L    349 W    4517 Ch  "<x onclick=alert(1)>click this!"
000000023: 403        93 L    349 W    4517 Ch  "</script><svg onload=alert(1)>"
000000015: 403        93 L    349 W    4517 Ch  "</scrip</script>t><img src =q onerror=prompt(8)>"
000000026: 403        93 L    349 W    4517 Ch  "<x oncopy=alert(1)>copy this!"
000000024: 403        93 L    349 W    4517 Ch  "<x contenteditable onblur=alert(1)>lose focus!"
000000007: 403        93 L    349 W    4517 Ch  ""onclick=prompt(8)>"@x.y"
000000027: 403        93 L    349 W    4517 Ch  "<x oncontextmenu=alert(1)>right click this!"
000000028: 403        93 L    349 W    4517 Ch  "<x oncut=alert(1)>copy this!"
000000003: 403        93 L    349 W    4517 Ch  "";a=prompt,a()//
000000022: 403        93 L    349 W    4517 Ch  "'-alert(1)//"
000000021: 403        93 L    349 W    4517 Ch  "'-alert(1)//"
000000020: 403        93 L    349 W    4517 Ch  "'-alert(1)'"
```

Conclusion

Reflected XSS was not exploitable through the term parameter on *care.bhasha.lk*. Both manual and automated tests failed to produce any script execution or input reflection. It is likely that the application either sanitizes input properly or does not process unrecognized input through the vulnerable context.

This test aligns with **OWASP Top 10 - A03:2021 – Injection**, focusing specifically on input validation and improper output encoding. Based on the results, the subdomain does not appear to be vulnerable to reflected XSS through its search functionality.

8. Proposed Mitigation

8.1 Reflected XSS `care.bhasha.lk`

Although no reflected XSS vulnerabilities were found, it is essential to maintain proper input validation and output encoding practices to ensure continued protection against injection attacks. The application should continue to sanitize user input and ensure that no user-controlled data is reflected back into the page without proper handling, as this will help prevent future vulnerabilities.

8.2 CAPTCHA Validation `freshdesk.bhasha.lk`

The CAPTCHA system is functioning as intended, effectively preventing automated login attempts and brute-force attacks. No further action is required, but it is recommended to keep CAPTCHA tokens session-bound and short-lived to prevent replay attacks. Regular monitoring and review of CAPTCHA implementation will help maintain its effectiveness as a security control.

9. Conclusion

The security testing conducted on the subdomains `freshdesk.bhasha.lk` and `care.bhasha.lk` focused on identifying weaknesses in CAPTCHA validation and input handling mechanisms. While the CAPTCHA on the login page at `freshdesk.bhasha.lk` was found to be well-implemented and resistant to common bypass techniques, attempts to exploit potential reflected XSS vulnerabilities on both subdomains were unsuccessful. The application correctly filtered or rejected malicious input, and no user input was reflected back in a vulnerable context. Overall, no critical vulnerabilities were discovered during the engagement, indicating that these subdomains currently implement effective basic security controls against automated attacks and input-based threats.

10. References

- OWASP Top 10 – A03:2021 – Injection: https://owasp.org/Top10/A03_2021-Injection/