Sri Lanka Institute of Information Technology

BUG BOUNTY REPORT - 4

Web Security – IE2062

IT22362780

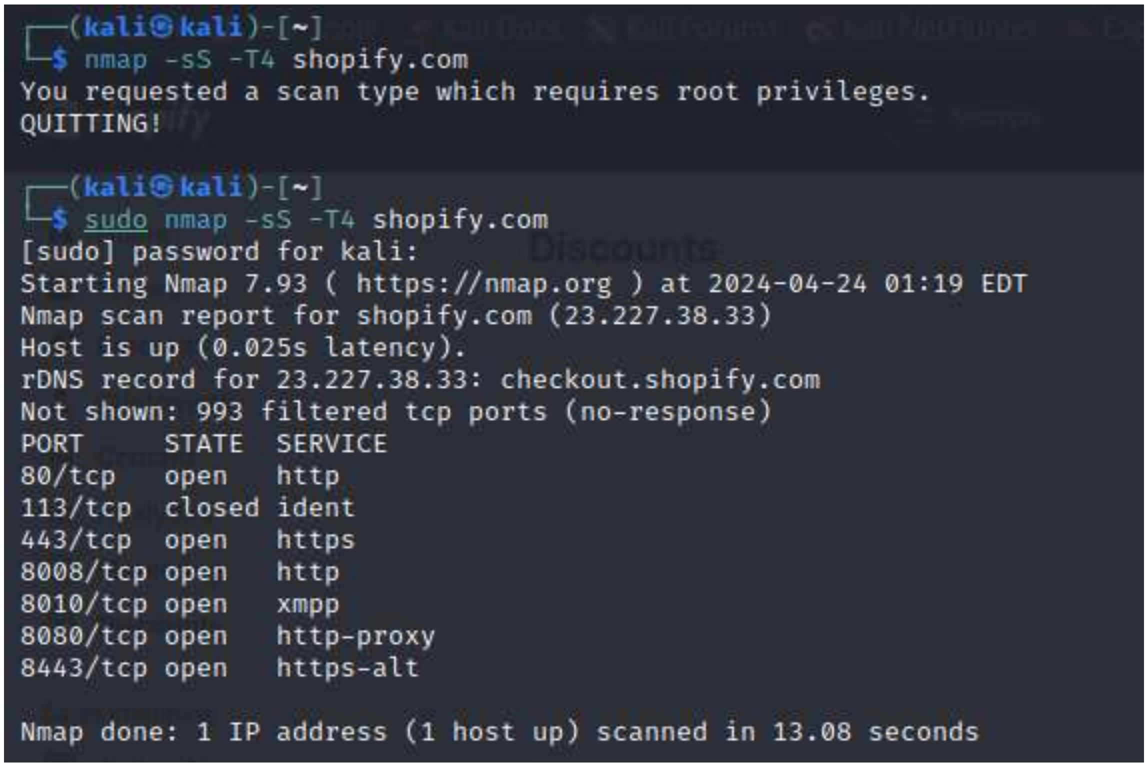
Jayaweera N.S

Report Details

Report #	- 04
Domain	- https://shopify.com
Platform	-hackerone.com
Scans performed -	Nmap scan
	Wafw00f scan
	Dotdotpwn scan
	Nikto scan
	Sqlmap scan
	Manual scanning using Wapplyzer
	Text injection testing
	File upload vulnerability testing
	Command injection
	XSS injection testing
	nslookup
	recon-ng scan
	CSRF scan
	Zap scan

Nmap scan

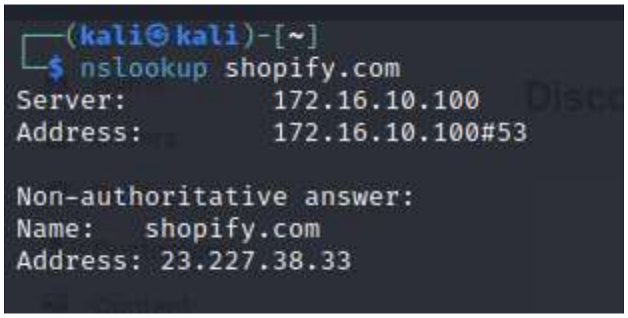
Used to find all the open ports



```
┌──(kali㉿kali)-[~]
└─$ nmap -sS -T4 shopify.com
You requested a scan type which requires root privileges.
QUITTING!

┌──(kali㉿kali)-[~]
└─$ sudo nmap -sS -T4 shopify.com
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2024-04-24 01:19 EDT
Nmap scan report for shopify.com (23.227.38.33)
Host is up (0.025s latency).
rDNS record for 23.227.38.33: checkout.shopify.com
Not shown: 993 filtered tcp ports (no-response)
PORT     STATE  SERVICE
80/tcp   open   http
113/tcp  closed ident
443/tcp  open   https
8008/tcp open   http
8010/tcp open   xmpp
8080/tcp open   http-proxy
8443/tcp open   https-alt

Nmap done: 1 IP address (1 host up) scanned in 13.08 seconds
```

No unusual ports found.

Nslookup

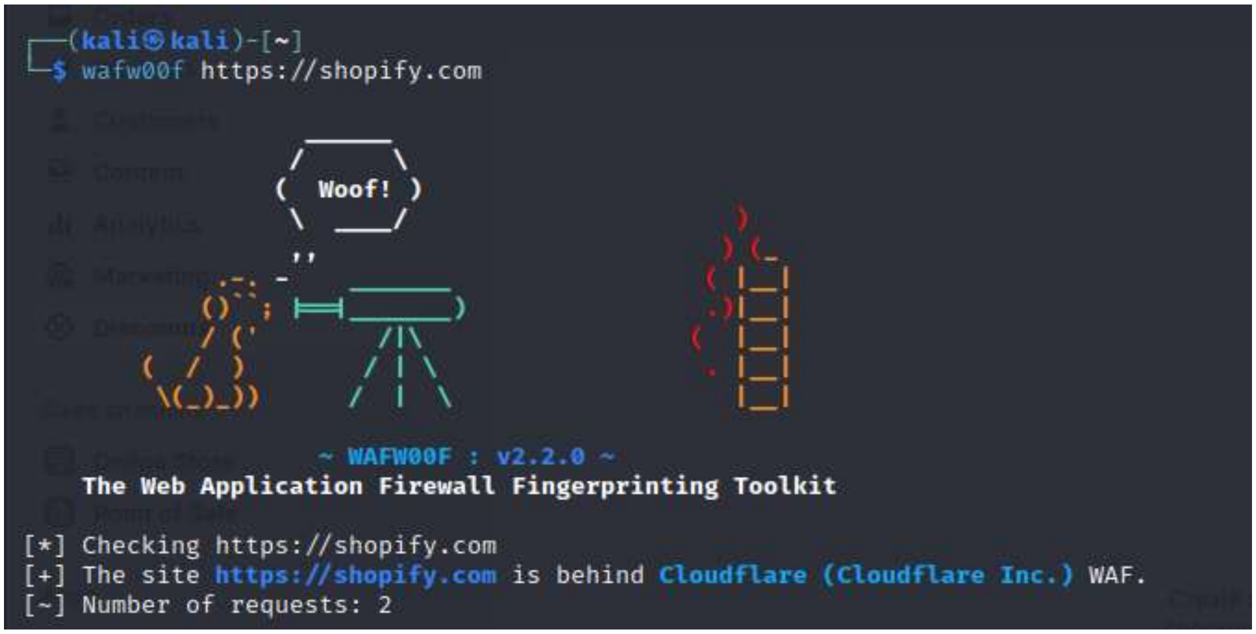


```
┌──(kali㉿kali)-[~]
└─$ nslookup shopify.com
Server:         172.16.10.100
Address:        172.16.10.100#53

Non-authoritative answer:
Name:   shopify.com
Address: 23.227.38.33
```

The ip addresses of shopify.com can be found.

<u>Wafw00f scan</u>

Used to identify the type of WAF that is used to protect the web application.

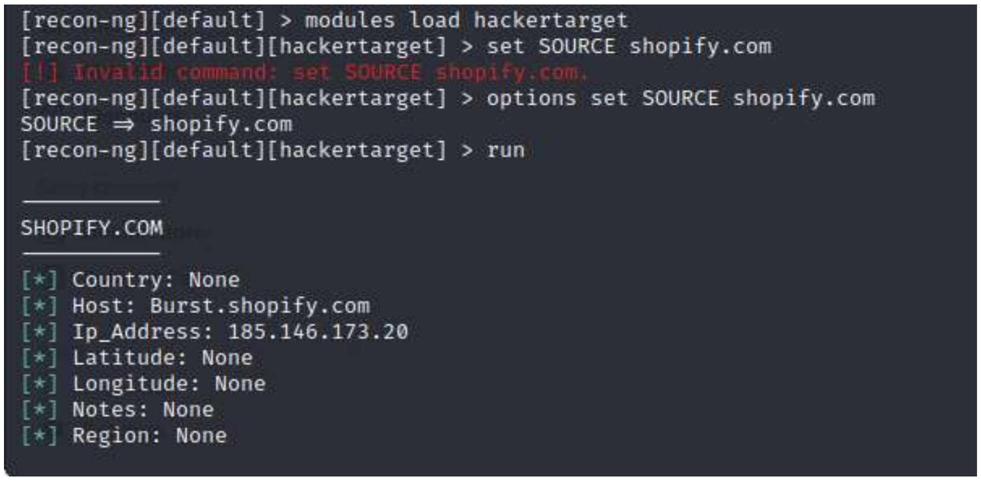


```
┌──(kali㉿kali)-[~]
└─$ wafw00f https://shopify.com

                ~ WAFW00F : v2.2.0 ~
    The Web Application Firewall Fingerprinting Toolkit

[*] Checking https://shopify.com
[+] The site https://shopify.com is behind Cloudflare (Cloudflare Inc.) WAF.
[~] Number of requests: 2
```

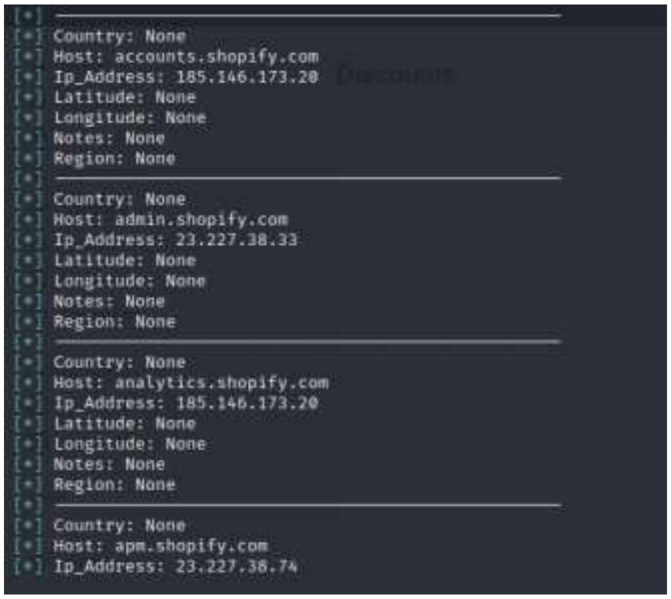
According to the test results, “Cloudflare (Cloudflare Inc.)” is used as the firewall of the web application.

Recon scan

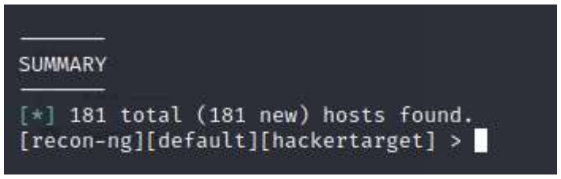
Recon-ng will be used to find all the sub domains in the target.



```
[recon-ng][default] > modules load hackertarget
[recon-ng][default][hackertarget] > set SOURCE shopify.com
[!] Invalid command: set SOURCE shopify.com.
[recon-ng][default][hackertarget] > options set SOURCE shopify.com
SOURCE ⇒ shopify.com
[recon-ng][default][hackertarget] > run

-----------
SHOPIFY.COM
-----------
[*] Country: None
[*] Host: Burst.shopify.com
[*] Ip_Address: 185.146.173.20
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
```



```
[*] ----------------------------------------
[*] Country: None
[*] Host: accounts.shopify.com
[*] Ip_Address: 185.146.173.20
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] ----------------------------------------
[*] Country: None
[*] Host: admin.shopify.com
[*] Ip_Address: 23.227.38.33
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] ----------------------------------------
[*] Country: None
[*] Host: analytics.shopify.com
[*] Ip_Address: 185.146.173.20
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] ----------------------------------------
[*] Country: None
[*] Host: apm.shopify.com
[*] Ip_Address: 23.227.38.74
```

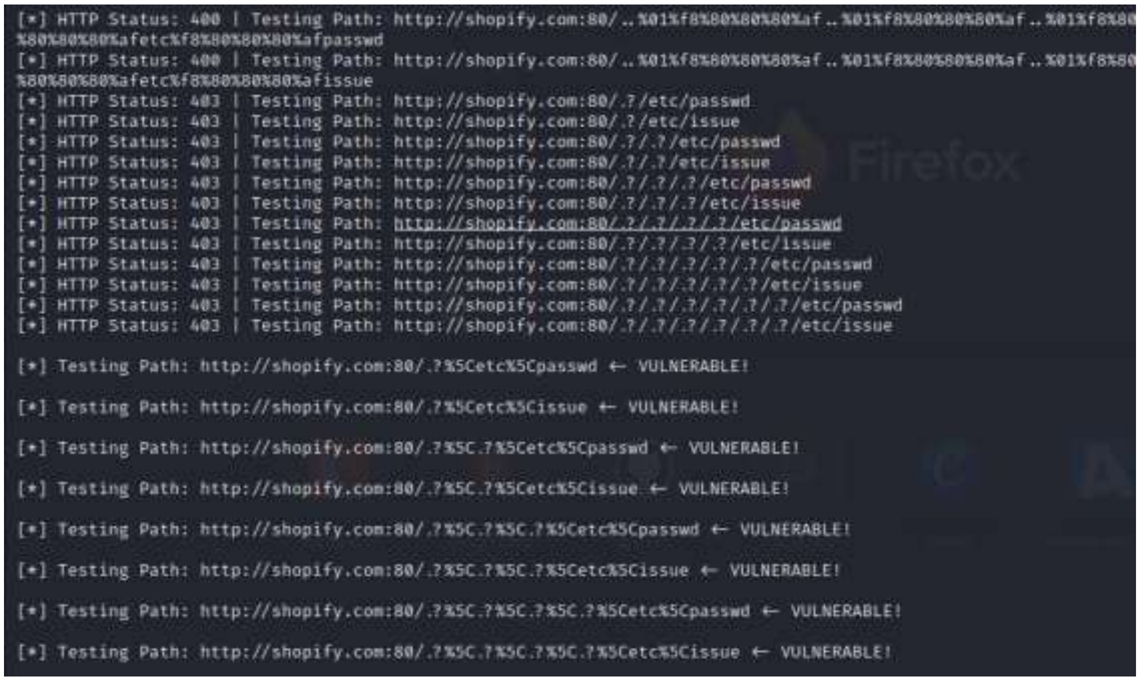


```
-------
SUMMARY
-------
[*] 181 total (181 new) hosts found.
[recon-ng][default][hackertarget] >
```

181 sub domains found.

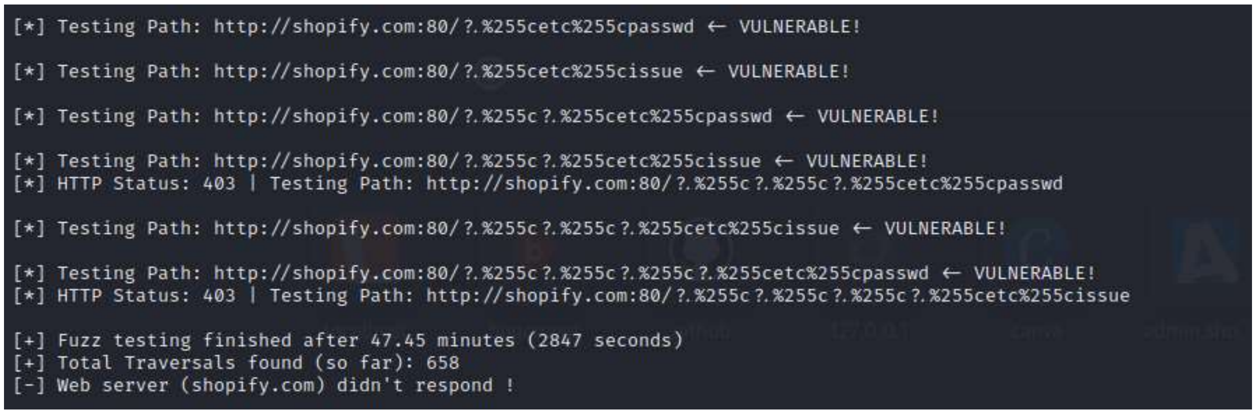
Dotdotpwn

Dotdotpwn is a directory traversal checker.



```
[*] HTTP Status: 400 | Testing Path: http://shopify.com:80/..%01%f8%80%80%80%af..%01%f8%80%80%80%80%af..%01%f8%80
%80%80%80%afetc%f8%80%80%80%afpasswd
[*] HTTP Status: 400 | Testing Path: http://shopify.com:80/..%01%f8%80%80%80%af..%01%f8%80%80%80%af..%01%f8%80
%80%80%80%afetc%f8%80%80%80%afissue
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/.?/etc/passwd
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/.?/etc/issue
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/.?/.?/etc/passwd
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/.?/.?/etc/issue
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/.?/.?/.?/etc/passwd
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/.?/.?/.?/etc/issue
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/.?/.?/.?/.?/etc/passwd
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/.?/.?/.?/.?/etc/issue
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/.?/.?/.?/.?/.?/etc/passwd
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/.?/.?/.?/.?/.?/etc/issue
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/.?/.?/.?/.?/.?/.?/etc/passwd
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/.?/.?/.?/.?/.?/.?/etc/issue

[*] Testing Path: http://shopify.com:80/.?%5Cetc%5Cpasswd ← VULNERABLE!

[*] Testing Path: http://shopify.com:80/.?%5Cetc%5Cissue ← VULNERABLE!

[*] Testing Path: http://shopify.com:80/.?%5C.?%5Cetc%5Cpasswd ← VULNERABLE!

[*] Testing Path: http://shopify.com:80/.?%5C.?%5Cetc%5Cissue ← VULNERABLE!

[*] Testing Path: http://shopify.com:80/.?%5C.?%5C.?%5Cetc%5Cpasswd ← VULNERABLE!

[*] Testing Path: http://shopify.com:80/.?%5C.?%5C.?%5Cetc%5Cissue ← VULNERABLE!

[*] Testing Path: http://shopify.com:80/.?%5C.?%5C.?%5C.?%5Cetc%5Cpasswd ← VULNERABLE!

[*] Testing Path: http://shopify.com:80/.?%5C.?%5C.?%5C.?%5Cetc%5Cissue ← VULNERABLE!
```

Severable vulnerable directory traversal paths were identified in port 80 of the website.

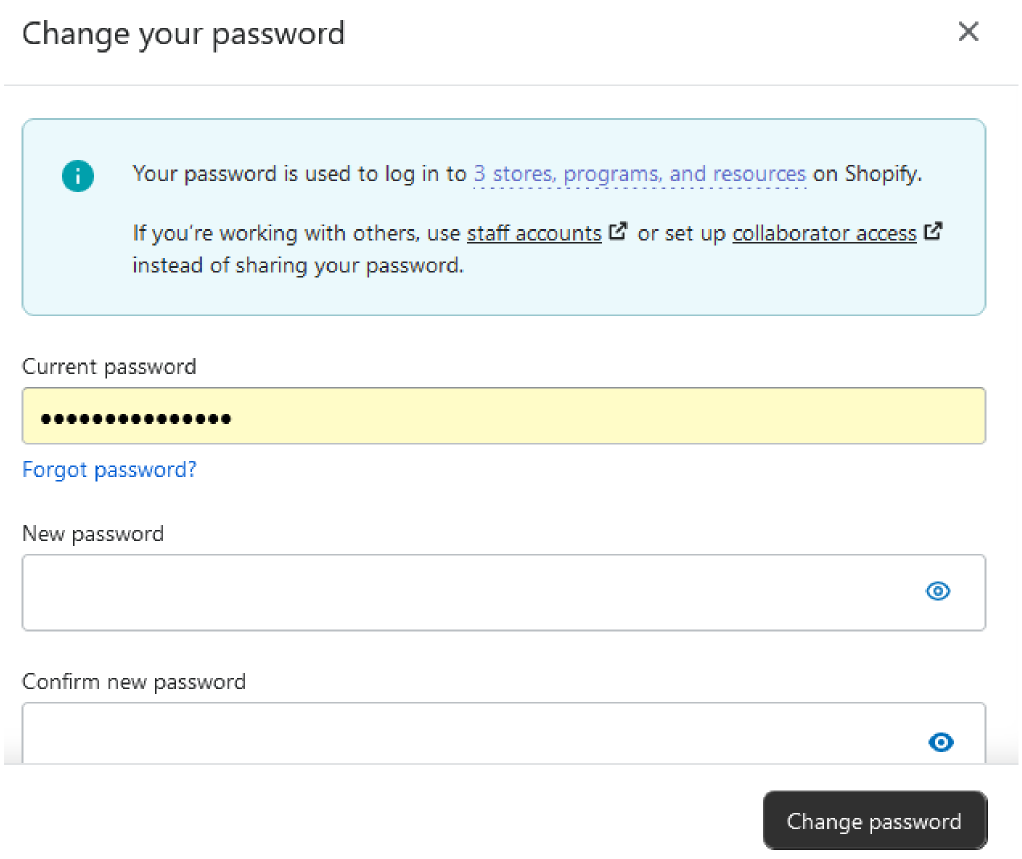


```
[*] Testing Path: http://shopify.com:80/?.%255cetc%255cpasswd ← VULNERABLE!

[*] Testing Path: http://shopify.com:80/?.%255cetc%255cissue ← VULNERABLE!

[*] Testing Path: http://shopify.com:80/?.%255c?.%255cetc%255cpasswd ← VULNERABLE!

[*] Testing Path: http://shopify.com:80/?.%255c?.%255cetc%255cissue ← VULNERABLE!
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/?.%255c?.%255c?.%255cetc%255cpasswd

[*] Testing Path: http://shopify.com:80/?.%255c?.%255c?.%255cetc%255cissue ← VULNERABLE!

[*] Testing Path: http://shopify.com:80/?.%255c?.%255c?.%255c?.%255cetc%255cpasswd ← VULNERABLE!
[*] HTTP Status: 403 | Testing Path: http://shopify.com:80/?.%255c?.%255c?.%255c?.%255cetc%255cissue

[+] Fuzz testing finished after 47.45 minutes (2847 seconds)
[+] Total Traversals found (so far): 658
[-] Web server (shopify.com) didn't respond !
```

658 possible traversals found.

Manual testing – CSRF testing

Checking for CSRF in change password feature.



Change your password

Your password is used to log in to 3 stores, programs, and resources on Shopify.

If you're working with others, use staff accounts or set up collaborator access instead of sharing your password.

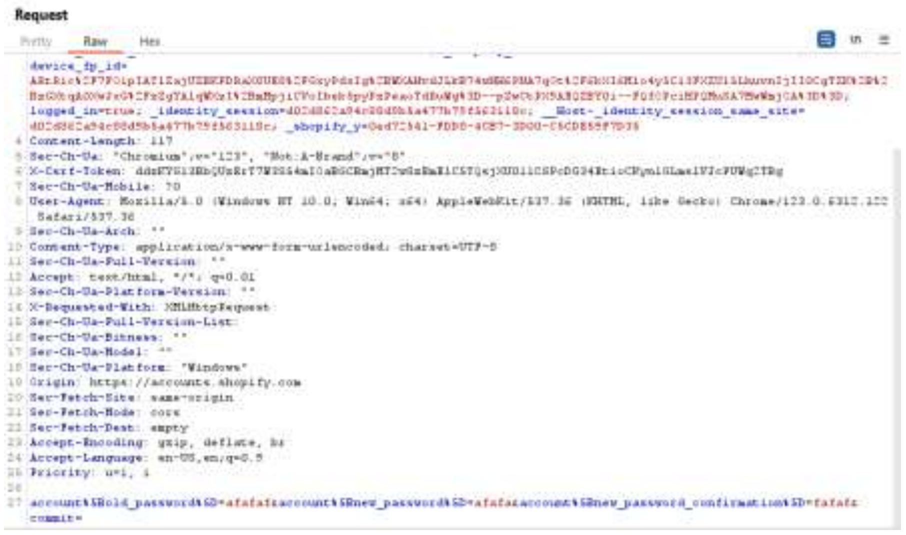
Current password

●●●●●●●●●●●●●●●

Forgot password?

New password

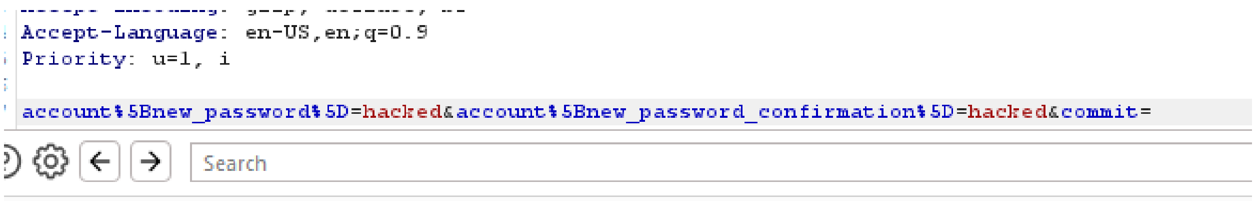
Confirm new password



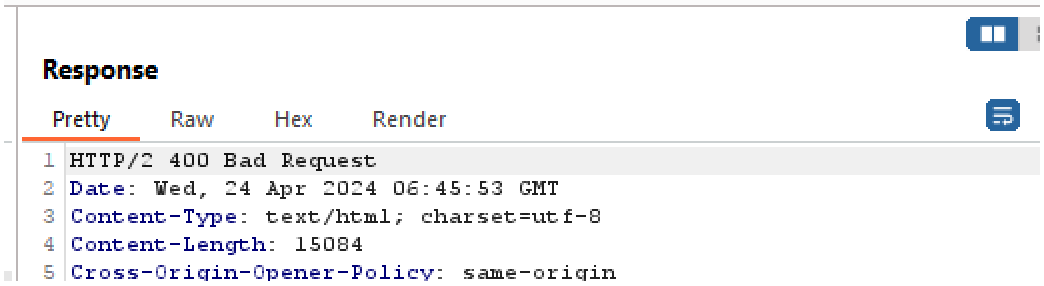
Request

```
device_fp_id=
[illegible]
[illegible]
logged_in=true; _identity_session=[illegible]; __Host-_identity_session_same_site=
[illegible]; _shopify_y=[illegible]
Content-Length: 117
Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
X-Csrf-Token: [illegible]
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122
 Safari/537.36
Sec-Ch-Ua-Arch: ""
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Sec-Ch-Ua-Full-Version: ""
Accept: text/html, */*; q=0.01
Sec-Ch-Ua-Platform-Version: ""
X-Requested-With: XMLHttpRequest
Sec-Ch-Ua-Full-Version-List:
Sec-Ch-Ua-Bitness: ""
Sec-Ch-Ua-Model: ""
Sec-Ch-Ua-Platform: "Windows"
Origin: https://accounts.shopify.com
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.5
Priority: u=1, i

account%5Bold_password%5D=afafafaaccount%5Bnew_password%5D=afafaaaccount%5Bnew_password_confirmation%5D=fafafa
commit=
```

Deleting the old password and check if it is being checked.



```
Accept-Language: en-US,en;q=0.9
Priority: u=1, i

account%5Bnew_password%5D=hacked&account%5Bnew_password_confirmation%5D=hacked&commit=
```

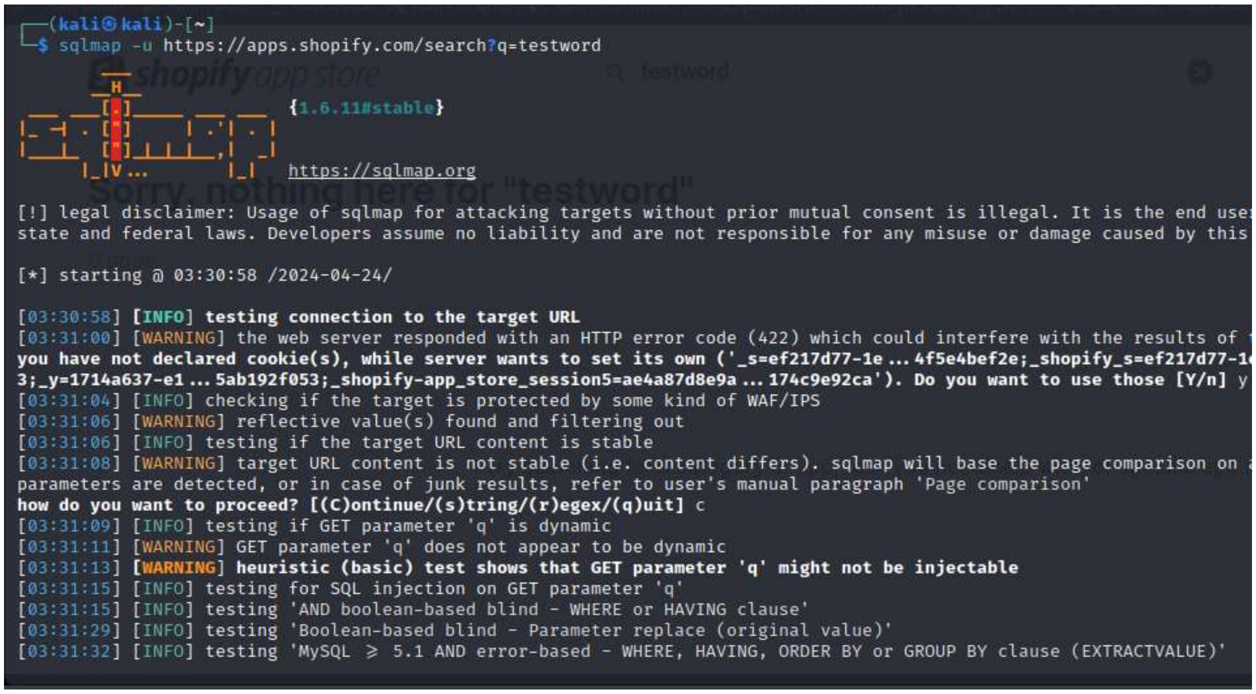


```
HTTP/2 400 Bad Request
Date: Wed, 24 Apr 2024 06:45:53 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 15084
Cross-Origin-Opener-Policy: same-origin
```

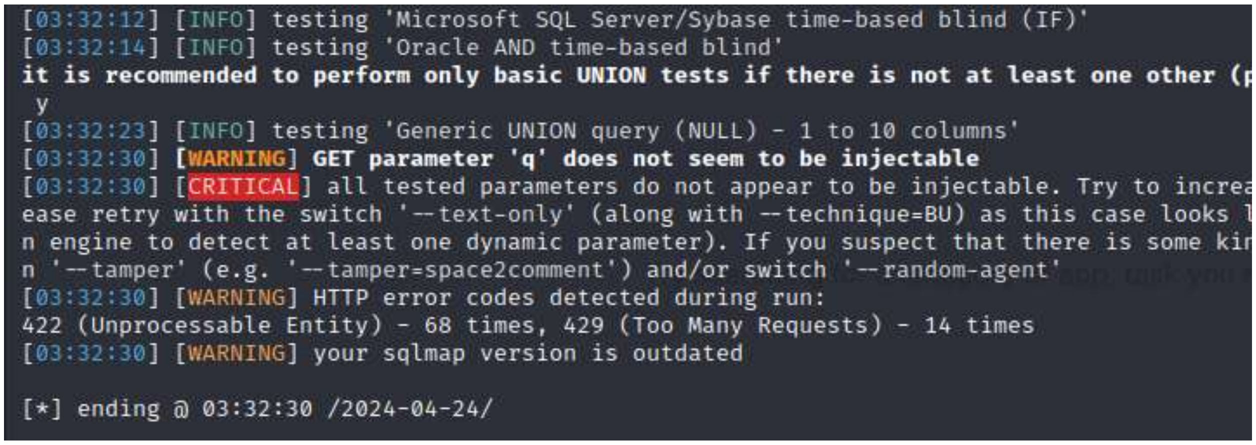
The old password is needed and being checked, so csrf on change password is not possible.

Sqlmap

It checks whether the web application is vulnerable over sql injection.



```
┌──(kali㉿kali)-[~]
└─$ sqlmap -u https://apps.shopify.com/search?q=testword
                                        {1.6.11#stable}


                                        https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end use
state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this

[*] starting @ 03:30:58 /2024-04-24/

[03:30:58] [INFO] testing connection to the target URL
[03:31:00] [WARNING] the web server responded with an HTTP error code (422) which could interfere with the results of
you have not declared cookie(s), while server wants to set its own ('_s=ef217d77-1e ... 4f5e4bef2e;_shopify_s=ef217d77-1
3;_y=1714a637-e1 ... 5ab192f053;_shopify-app_store_session5=ae4a87d8e9a ... 174c9e92ca'). Do you want to use those [Y/n] y
[03:31:04] [INFO] checking if the target is protected by some kind of WAF/IPS
[03:31:06] [WARNING] reflective value(s) found and filtering out
[03:31:06] [INFO] testing if the target URL content is stable
[03:31:08] [WARNING] target URL content is not stable (i.e. content differs). sqlmap will base the page comparison on
parameters are detected, or in case of junk results, refer to user's manual paragraph 'Page comparison'
how do you want to proceed? [(C)ontinue/(s)tring/(r)egex/(q)uit] c
[03:31:09] [INFO] testing if GET parameter 'q' is dynamic
[03:31:11] [WARNING] GET parameter 'q' does not appear to be dynamic
[03:31:13] [WARNING] heuristic (basic) test shows that GET parameter 'q' might not be injectable
[03:31:15] [INFO] testing for SQL injection on GET parameter 'q'
[03:31:15] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[03:31:29] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[03:31:32] [INFO] testing 'MySQL ≥ 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
```

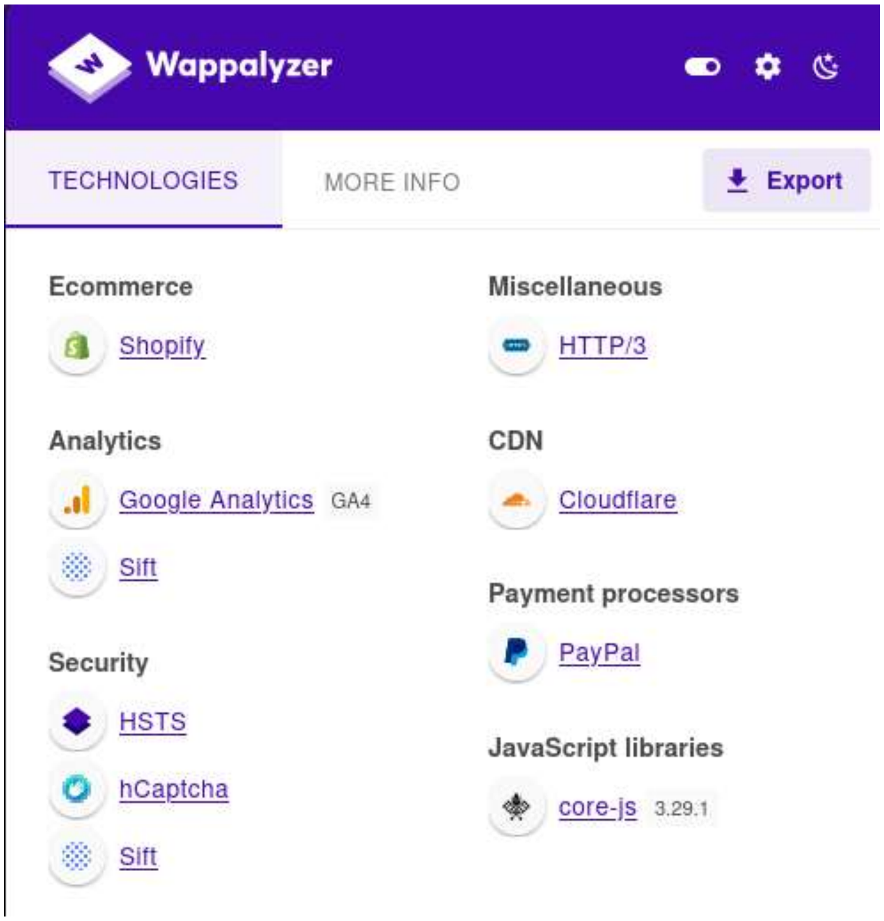


```
[03:32:12] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[03:32:14] [INFO] testing 'Oracle AND time-based blind'
it is recommended to perform only basic UNION tests if there is not at least one other (
 y
[03:32:23] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[03:32:30] [WARNING] GET parameter 'q' does not seem to be injectable
[03:32:30] [CRITICAL] all tested parameters do not appear to be injectable. Try to increa
ease retry with the switch '--text-only' (along with --technique=BU) as this case looks l
n engine to detect at least one dynamic parameter). If you suspect that there is some ki
n '--tamper' (e.g. '--tamper=space2comment') and/or switch '--random-agent'
[03:32:30] [WARNING] HTTP error codes detected during run:
422 (Unprocessable Entity) - 68 times, 429 (Too Many Requests) - 14 times
[03:32:30] [WARNING] your sqlmap version is outdated

[*] ending @ 03:32:30 /2024-04-24/
```

The web application does not seem to be injectable.

Wapplyzer

The Wapplyzer is used to identify the technologies used in the web application.


Wappalyzer

TECHNOLOGIES | MORE INFO | Export

Ecommerce
Shopify

Analytics
Google Analytics GA4
Sift

Security
HSTS
hCaptcha
Sift

Miscellaneous
HTTP/3

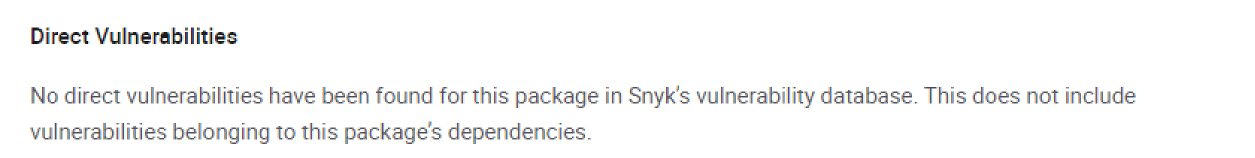
CDN
Cloudflare

Payment processors
PayPal

JavaScript libraries
core-js 3.29.1

These are the technologies used in the above web application

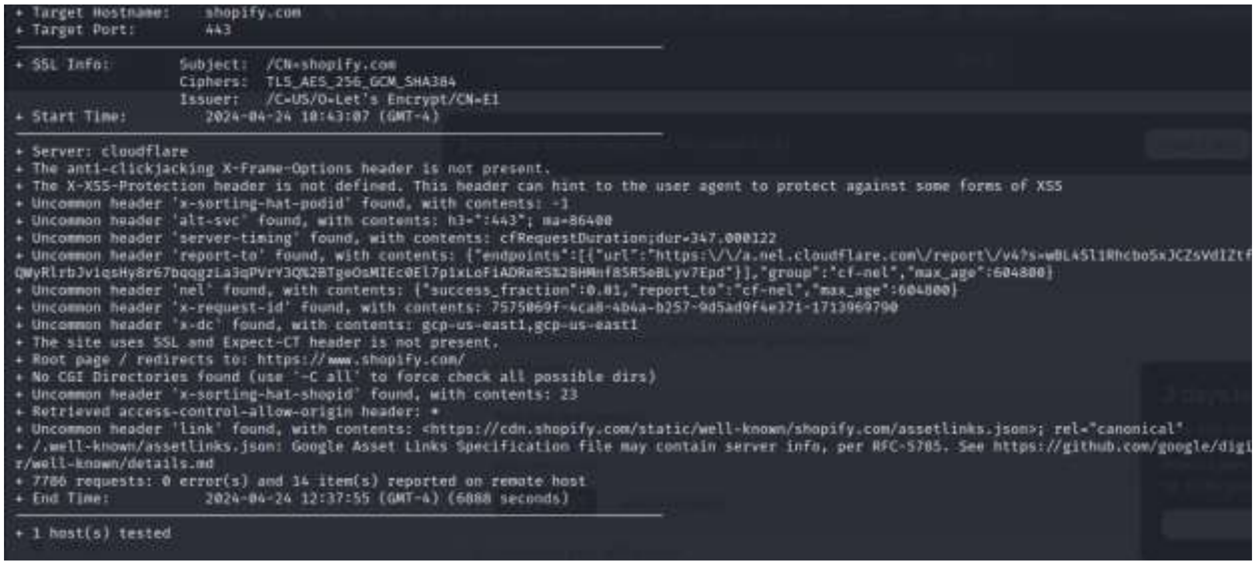
The core-js 3.29.1 is not a vulnerable version.


Direct Vulnerabilities

No direct vulnerabilities have been found for this package in Snyk's vulnerability database. This does not include vulnerabilities belonging to this package's dependencies.

Figure from https://security.snyk.io/package/npm/core-js

Nikto scan



```
+ Target Hostname:    shopify.com
+ Target Port:        443
---------------------------------------------------------------------------
+ SSL Info:        Subject:  /CN=shopify.com
                   Ciphers:  TLS_AES_256_GCM_SHA384
                   Issuer:   /C=US/O=Let's Encrypt/CN=E1
+ Start Time:         2024-04-24 10:43:07 (GMT-4)
---------------------------------------------------------------------------
+ Server: cloudflare
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ Uncommon header 'x-sorting-hat-podid' found, with contents: -1
+ Uncommon header 'alt-svc' found, with contents: h3=":443"; ma=86400
+ Uncommon header 'server-timing' found, with contents: cfRequestDuration;dur=347.000122
+ Uncommon header 'report-to' found, with contents: {"endpoints":[{"url":"https:\/\/a.nel.cloudflare.com\/report\/v4?s=wBL4S11Rhcbo5xJCZsVdIZtf
QWyRlrbJviqsHy8r67bqqgzLa3qPVrY3Q%2BTgeOsMIEc0El7p1xLoFiADReRS%2BHMnf8SRSeBLyv7Epd"}],"group":"cf-nel","max_age":604800}
+ Uncommon header 'nel' found, with contents: {"success_fraction":0.01,"report_to":"cf-nel","max_age":604800}
+ Uncommon header 'x-request-id' found, with contents: 7575069f-4ca8-4b4a-b257-9d5ad9f4e371-1713969790
+ Uncommon header 'x-dc' found, with contents: gcp-us-east1,gcp-us-east1
+ The site uses SSL and Expect-CT header is not present.
+ Root page / redirects to: https://www.shopify.com/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Uncommon header 'x-sorting-hat-shopid' found, with contents: 23
+ Retrieved access-control-allow-origin header: *
+ Uncommon header 'link' found, with contents: <https://cdn.shopify.com/static/well-known/shopify.com/assetlinks.json>; rel="canonical"
+ /.well-known/assetlinks.json: Google Asset Links Specification file may contain server info, per RFC-5785. See https://github.com/google/digi
r/well-known/details.md
+ 7786 requests: 0 error(s) and 14 item(s) reported on remote host
+ End Time:           2024-04-24 12:37:55 (GMT-4) (6888 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

The scan results are as follows:

- The anti-clickjacking X-Frame-Options header which is used for preventing clickjacking is not found.

- The X-XXS-Protection header that is useful in preventing some forms of XSS attacks is not found.

- This site uses SSL and Expect-CT header is not found.

The "expect-CT" header is a security feature that helps websites, and their users avoid the risks associated with incorrectly issued SSL certificates.

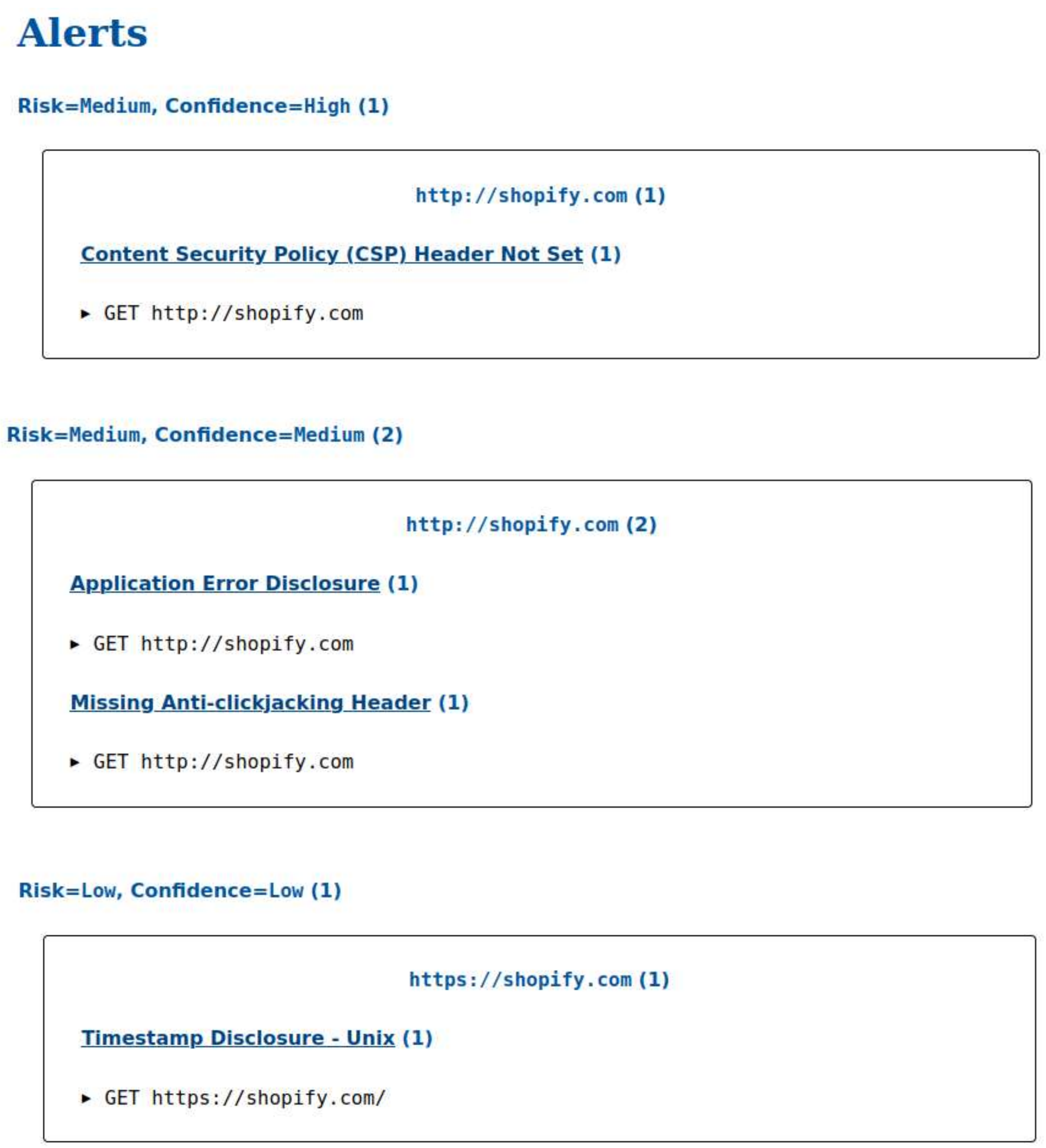
It supports transparency and accountability when issuing SSL certificates, which improves overall web security.

There are some issues/disadvantages occurred when the “expect-CT” header is absent:

- The protection against the mis issuing of SSL certificates will be low.
- Mismanagement of SSL certificates.
- No trust and security

But the absence of “expected-CT” header is not a huge vulnerability or a security issue in a website.

Zap scan



Alerts

Risk=Medium, Confidence=High (1)

http://shopify.com (1)

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

```
► GET http://shopify.com
```

Risk=Medium, Confidence=Medium (2)

http://shopify.com (2)

Application Error Disclosure (1)

```
► GET http://shopify.com
```

Missing Anti-clickjacking Header (1)

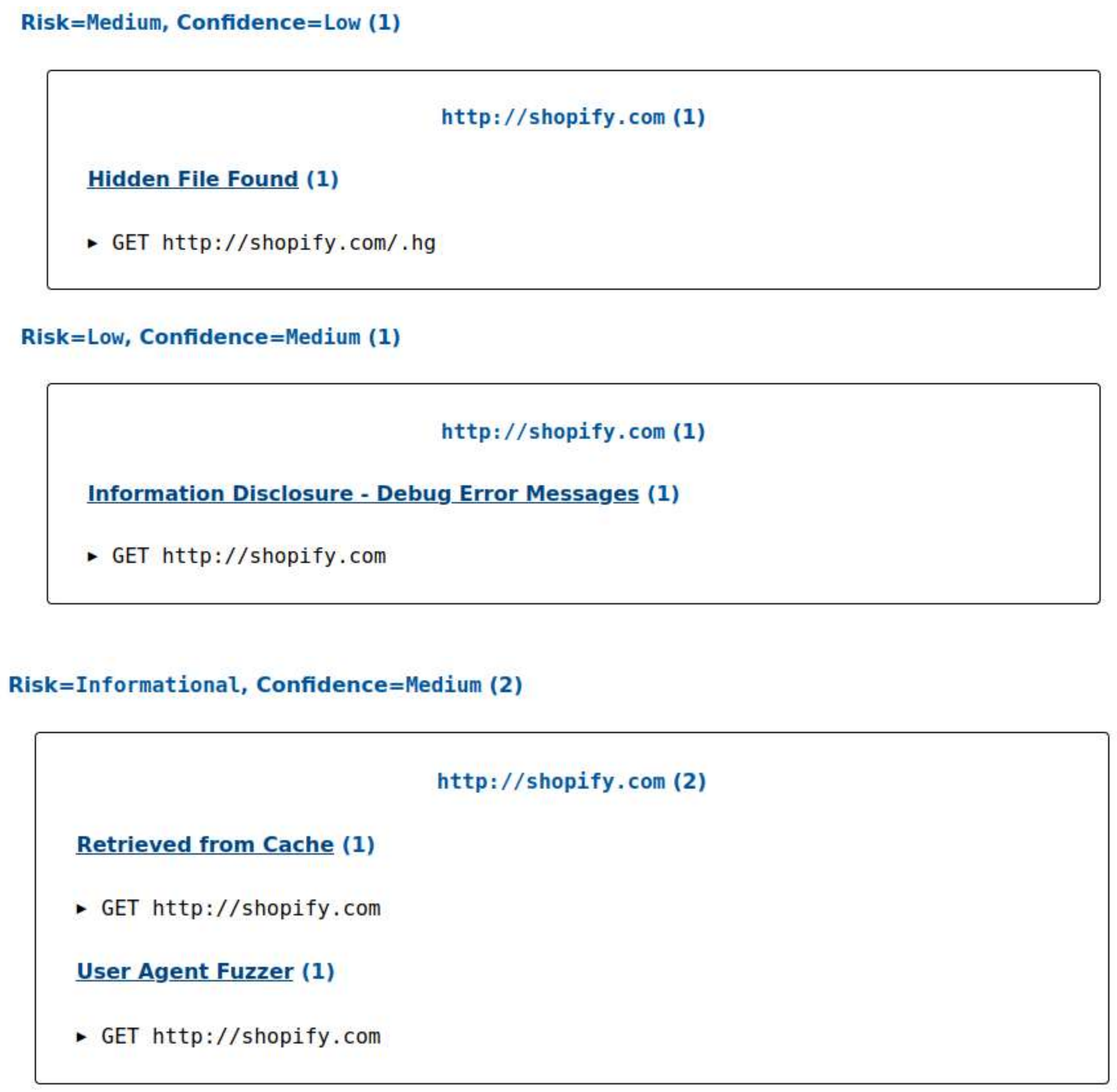
```
► GET http://shopify.com
```

Risk=Low, Confidence=Low (1)

https://shopify.com (1)

Timestamp Disclosure - Unix (1)

```
► GET https://shopify.com/
```



Risk=Medium, Confidence=Low (1)

http://shopify.com (1)

Hidden File Found (1)

▸ GET http://shopify.com/.hg

Risk=Low, Confidence=Medium (1)

http://shopify.com (1)

Information Disclosure - Debug Error Messages (1)

▸ GET http://shopify.com

Risk=Informational, Confidence=Medium (2)

http://shopify.com (2)

Retrieved from Cache (1)

▸ GET http://shopify.com

User Agent Fuzzer (1)

▸ GET http://shopify.com

Summary of the zap scan:

- Content security policy (CSP) header is not found. Due to the absence of this header, there can be XSS injections, data injections and clickjacking.

- There is an application error disclosure where the sensitive information could be exposed with a simple error/warning message.

- The Anti-clickjacking header which is used to prevent XSS attacks and enhance security posture is not present.

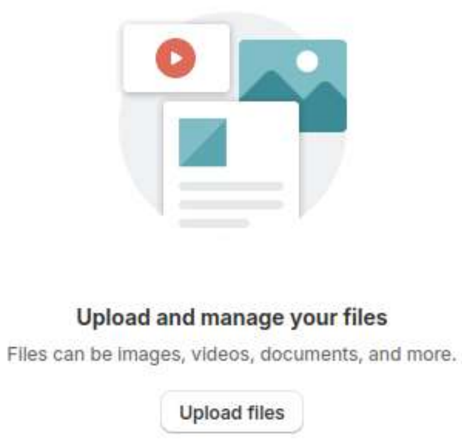
- There is an intentional disclosure of sensitive time stamp information where it could possibly lead to user tracking, session prediction, reconnaissance attacks, and temporal correlation attacks.

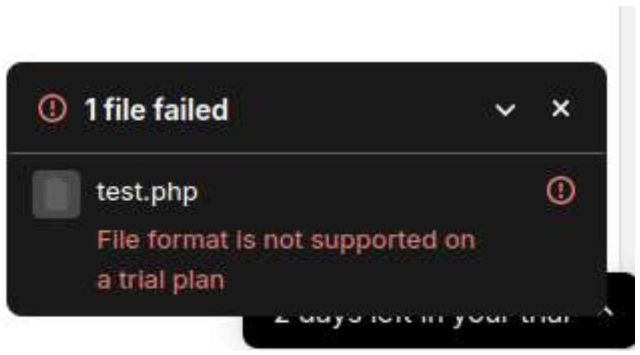
- Hidden file found.

- Informational disclosure-debug message error -inside the server response there is a message(error) that contains sensitive information about server which can be used as

File upload vulnerability

If a .php file can be uploaded from the file uploading facility, there is a possibility to upload and execute a reverse shell php code.


Upload and manage your files
Files can be images, videos, documents, and more.
Upload files



1 file failed

test.php

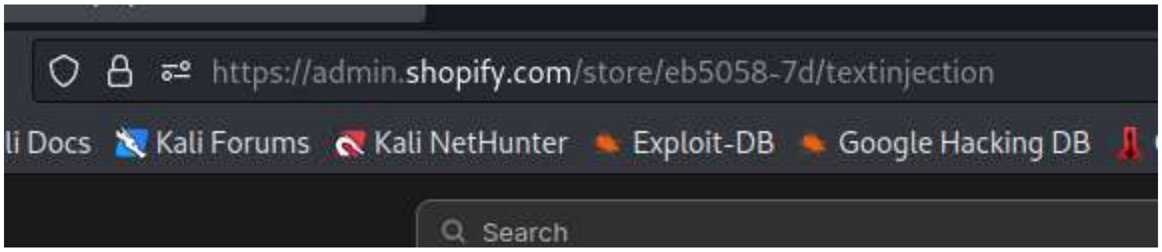
File format is not supported on a trial plan

2 days left in your trial

No vulnerability found.

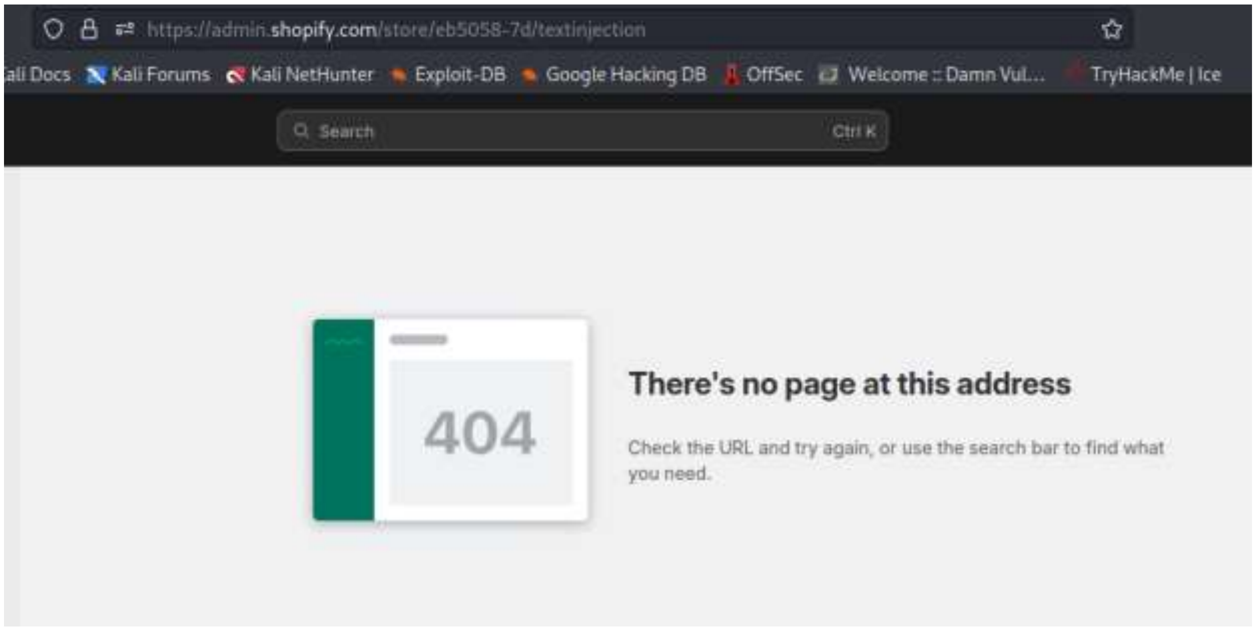
Text injection

An arbitrary string value is appended to the URL to see whether the web application is vulnerable towards a text injection.


https://admin.shopify.com/store/eb5058-7d/textinjection

If the entered text is reflected on the error response of the web page, there is a possibility to inject malicious content.

If not, the web application is safe.


https://admin.shopify.com/store/eb5058-7d/textinjection

404

There's no page at this address

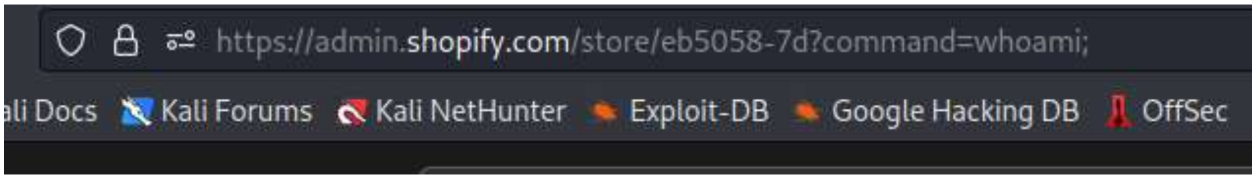
Check the URL and try again, or use the search bar to find what you need.

No text injection vulnerability can be found.

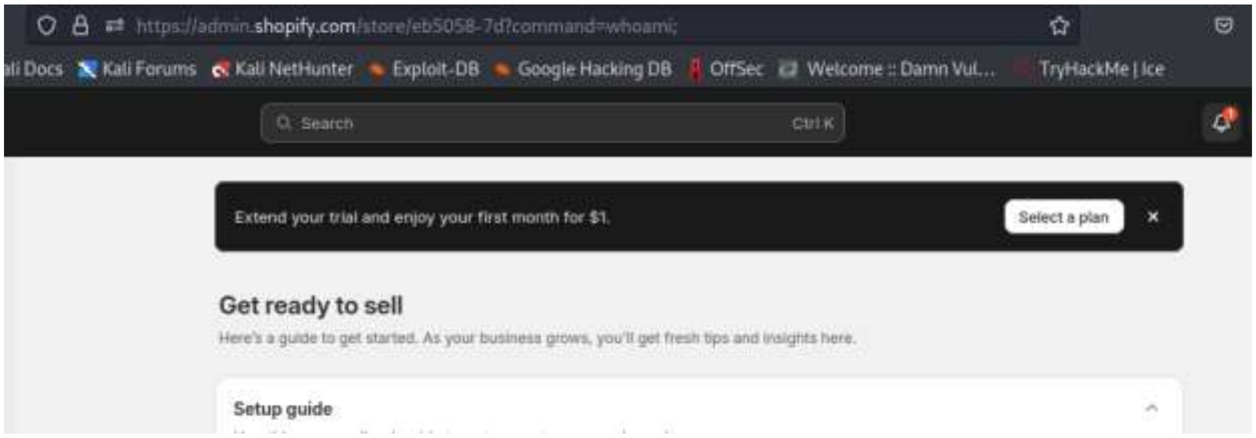
command injection

The query that is used for searching is used against this vulnerability.

The “whoami” command is appended to the url.



https://admin.shopify.com/store/eb5058-7d?command=whoami;



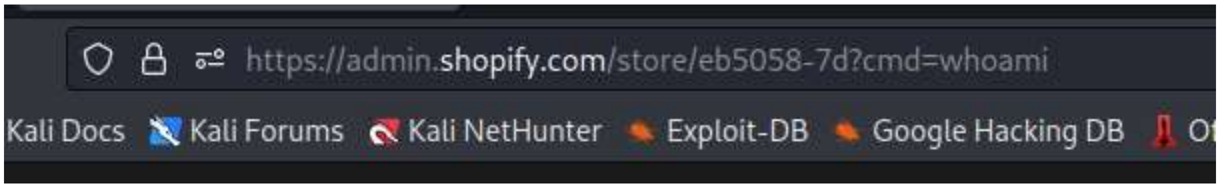
https://admin.shopify.com/store/eb5058-7d?command=whoami;

Extend your trial and enjoy your first month for $1.

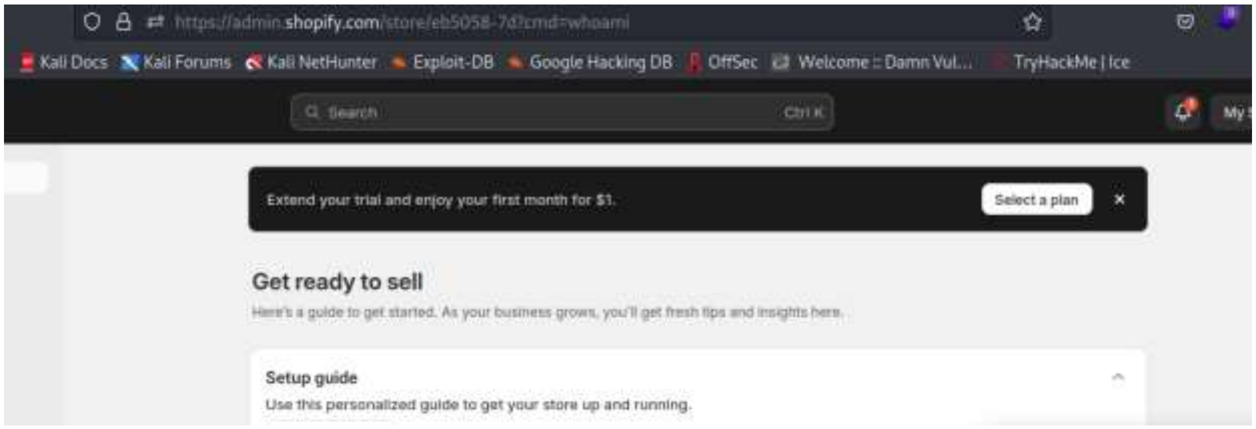
Get ready to sell

Here's a guide to get started. As your business grows, you'll get fresh tips and insights here.

Setup guide



https://admin.shopify.com/store/eb5058-7d?cmd=whoami



https://admin.shopify.com/store/eb5058-7d?cmd=whoami

Extend your trial and enjoy your first month for $1.

Get ready to sell

Here's a guide to get started. As your business grows, you'll get fresh tips and insights here.

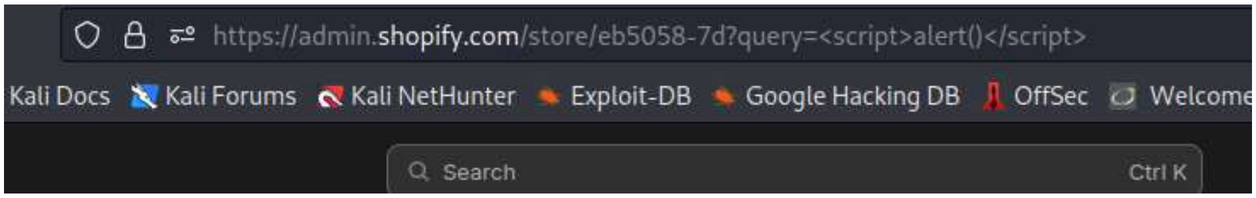
Setup guide

Use this personalized guide to get your store up and running.

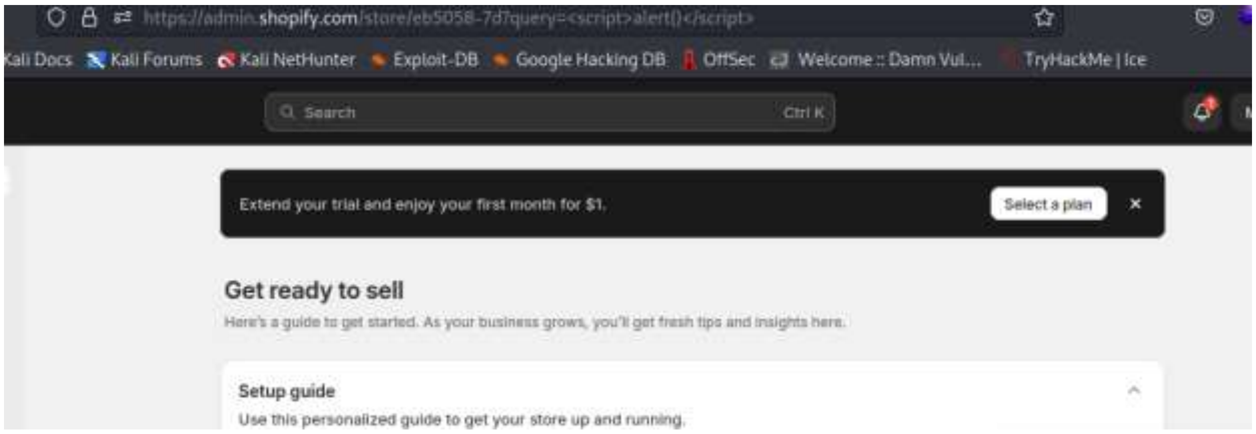
No command injection vulnerability can be found.

Xss injection

A payload is apended to the url to test against xss injection.



```
https://admin.shopify.com/store/eb5058-7d?query=<script>alert()</script>
```



```
https://admin.shopify.com/store/eb5058-7d?query=<script>alert()</script>
```

Extend your trial and enjoy your first month for $1.

Get ready to sell

Here's a guide to get started. As your business grows, you'll get fresh tips and insights here.

Setup guide

Use this personalized guide to get your store up and running.

Input sanitization is there as the script is considered as text So no xss vulnerability is present.