Sri Lanka Institute of Information Technology

Web Application Security Audit

Assessment Report

IE2062 – Web Security

Submitted by:

Student Registration Number	Student Name
M. Thushitharan	IT19983370

Date of submission: 01/11/2021

Table of Contents

Video Link:
https://drive.google.com/file/d/1cfQF5scxL0_6LTa7BKm3mrbTEEwiTIad/view?usp=sharing

1. Declaration

I declare that this assignment Report has purely done by me and I mention that this report was not copied from anyone or any institute. And I have used internet to take reference notes, not for copying purpose.

2. Abstract

The project titled Web Application Security Audit is the most ideal approach to guarantee your application is secure before you discharge it and to forestall hacks, harm to notoriety and critical misfortunes to your main concern for application designers.

Find out a web application registered in Hackerone/Bugcrowd and analyzed their vulnerabilities by penetration testing tools. Security testing software' primary function is to perform functional testing of a web application under observance and find as many security issues as possible that could potentially lead to hacking. All of this is done without the need to access the source code. Here some security testing tools used to analyze the vulnerabilities such as Netsparker - web application security, Sublist3r - python tool designed to enumerate subdomains of websites using OSINT, OWASP ZAP - an open-source web application security scanner, Paros - Java based HTTP/HTTPS proxy for assessing web application vulnerability, Golismero - an open-source framework built in Python that can be used for security testing.

This project report will help to decrease the vulnerabilities and get out from the theft of sensitive data in this global technologized world. Many security tests have been carried on, and they are attached and discussed in this document.

3. Acknowledgement

The completion of this Project is not possible without the help and support of many people and entities. Their contributions are sincerely appreciated and gratefully acknowledged. However, the group would like to express their deep appreciation and indebtedness particularly to the following:

Our Lecturers Dr. Lakmal Rupasinghe, Ms. Chethana Liyanapathirana, Ms. Chathu Udagedara for their endless support and understanding spirit during the project development. Some professional bug bounty hunters Ben Sadeghipour (@nahamsec), STÖK (@stokfredrik), Vickie Li (@vickieli7) and Some YouTube Channel contents HackerSploit, John Hammond. I have got many useful ideas from these people and entities to finish this assignment.

4. Assessment Objectives

This web application security audit assignment for http://netflix.com is for second semester cyber security students. He objective of this assignment is measure the capabilities to understand the security measurements of a web application in real world and moreover to understand how to write a vulnerability assessment and penetration testing report.

5. Application Credentials and URL

I have selected Bugcrowd platform to find the domain for this assignment. This assignment is based on root domain, even though I did some scanning to following subdomains

- Netflix.com (**root domain**)
- Help.netflix.com
- Media.netflix.com
- candidate.netflix.com
- brand.netflix.com
- meechum.netflix.com
- openconnect.netflix.com
- devices.netflix.com
- dvd.netflix.com
- jobs.netflix.com
- partner.netflix.com

6. Assessment methodology

7. OWASP Top 10 Security Risks

OWASP is stands for **Open Web Application Security Project**. This is a non-profit organization which is launched to create a standard for web application security and improve it.

Risk	About it
Injections	These vulnerabilities allow hackers or attackers to inject malicious code in a or many systems. XPath Injection, SQL Injection, LDAP Injection, XML Injection, and Command Injection are some of the common injection vulnerabilities
Broken Authentication	This vulnerability is related to session management and authentication management. Using this vulnerability attackers can compromise passwords, session tokens and keys. Attackers can gain access through manual or automatic methods. This vulnerability allows to do credential stuffing and brute force attacks.
Sensitive Data Exposure	Customers and employees Passwords, Credit card numbers, tax IDs, medical information, and other personal information should be protected by the web application. If attackers gain the access of the web application, they can steal and compromise the data.
XML External Entity attacks (XXE)	If there is a vulnerable XML processor then Attackers can upload malicious xml or malicious content with XML file. Vulnerable code, Dependencies, integrations allows attackers to do XXE attacks.
Cross Site Scripting (XSS)	XSS attacks means injecting malicious client-side scripts into a web application and using the website as a propagation method. It allows the attacker to inject malicious script into a website and modify how it is displayed, forcing a end user browser to execute the code provided by the attacker while loading the page.
Security Misconfigurations	Unpatched flaws, Default configurations, Unused pages, Unprotected files and directories, and Unnecessary services leads to security misconfigurations
Broken Access Control	Access control implements strategy with the end goal that clients can't act outside of their planned authorizations. Disappointments normally lead to unapproved data divulgence, alteration or pulverization of all information, or playing out a business work outside of the restrictions of the client.

Insecure Deserialization	Controlled item is infused into the setting of the web application. In the event that the application is helpless, the item is deserialized and executed, which can bring about SQL Injection, Path Traversal, Application Denial of Service and Remote Code Execution
Using Components with known vulnerabilities	Parts, such as, libraries, structures, and other programming modules, quite often run with full benefits. In the event that a powerless part is misused, such an assault can encourage genuine information misfortune or worker takeover. Applications utilizing parts with realised weaknesses may subvert application safeguards furthermore, empower a scope of potential assaults and effects.
Insufficient logging and monitoring	Hackers depend on the absence of checking and opportune reaction to accomplish their objectives without being identified

Figure 1: *Summary of Each OWASP top 10 Risks*

8. Vulnerability Severity and Examples

Severity	Few Examples
High	• Server Security Misconfiguration • File Inclusion • SQL Injection • Command Injection • Authentication Bypass
Medium	• Insecure Direct Object Reference (IDOR) • Cross Site Scripting (XSS) • HTTP Response Manipulation • Content Spoofing
Low	• Clickjacking • CAPTCHA Bypass • Insecure SSL • Parameter Pollution • Session Expiration

9. Assessment Scope Details

9.1 Target In scope

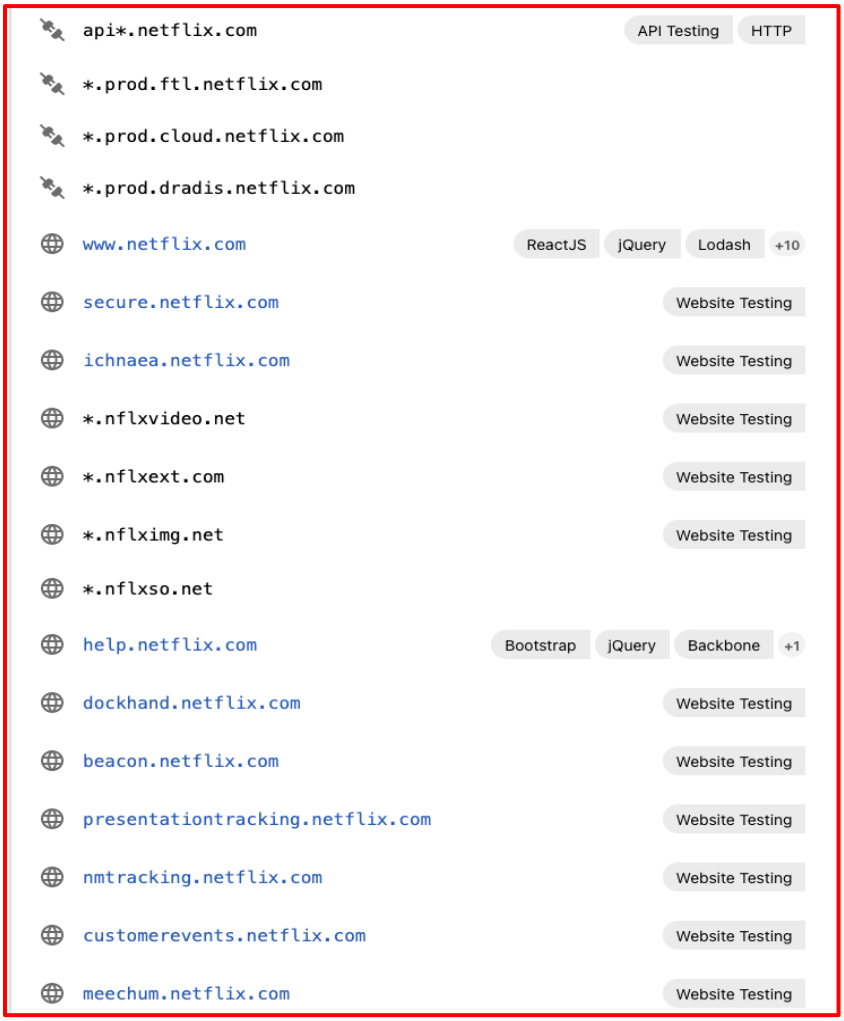


Figure 2: *In scope sections listed in Bugcrowd Netflix page*

9.2 Target Out of Scope

- Third Party websites hosted by non-Netflix domains
- Ir.netflix.com
- Netinvestor.com
- Netflix Client Application

10. Information Gathering (Recon) Phase

Gathering Information is an important part of penetration testing. It gives a huge amount of information as much as possible from the target web application. This phase also called as reconnaissance phase.

10.1 Finding Subdomains

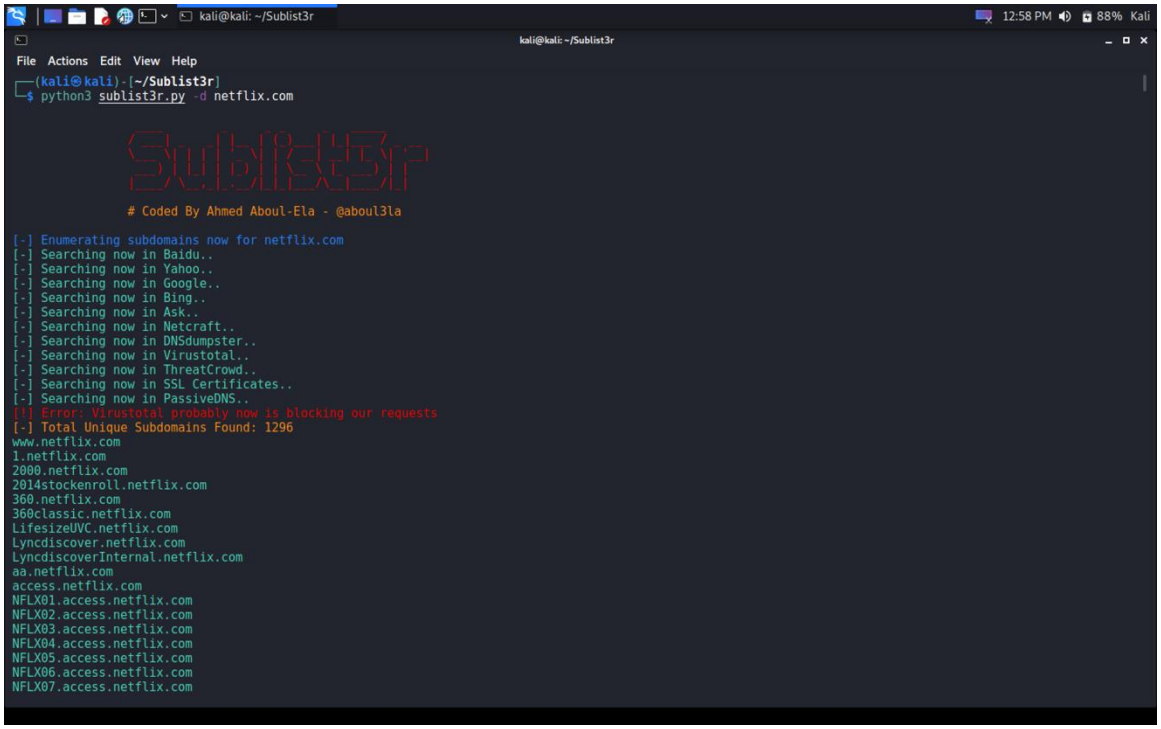
Sublist3r – Python Script

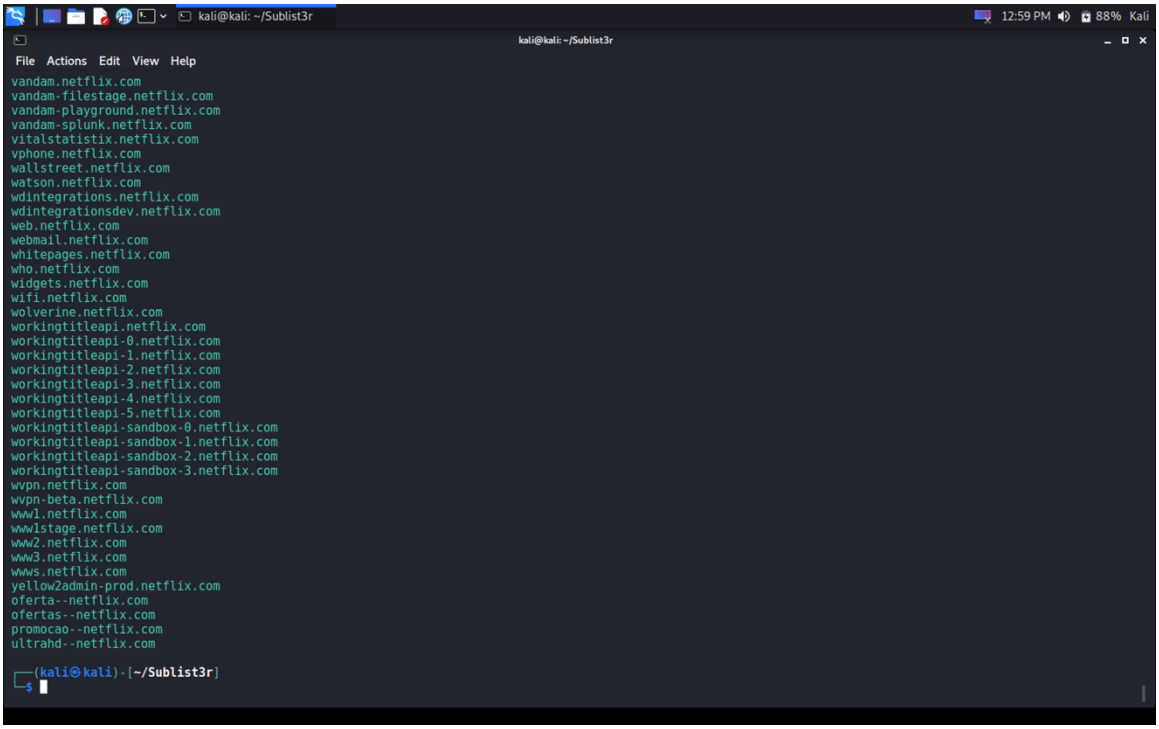
Github Repository Link: https://github.com/aboul3la/Sublist3r

Usage:

```
python3 sublist3r.py -d netflix.com
```

Scan Results: *This tool showed that 1296 sub domains found from this website.*




kali@kali: ~/Sublist3r
12:59 PM
88%
Kali
kali@kali: ~/Sublist3r
File Actions Edit View Help
vandam.netflix.com
vandam-filestage.netflix.com
vandam-playground.netflix.com
vandam-splunk.netflix.com
vitalstatistix.netflix.com
vphone.netflix.com
wallstreet.netflix.com
watson.netflix.com
wdintegrations.netflix.com
wdintegrationsdev.netflix.com
web.netflix.com
webmail.netflix.com
whitepages.netflix.com
who.netflix.com
widgets.netflix.com
wifi.netflix.com
wolverine.netflix.com
workingtitleapi.netflix.com
workingtitleapi-0.netflix.com
workingtitleapi-1.netflix.com
workingtitleapi-2.netflix.com
workingtitleapi-3.netflix.com
workingtitleapi-4.netflix.com
workingtitleapi-5.netflix.com
workingtitleapi-sandbox-0.netflix.com
workingtitleapi-sandbox-1.netflix.com
workingtitleapi-sandbox-2.netflix.com
workingtitleapi-sandbox-3.netflix.com
wvpn.netflix.com
wvpn-beta.netflix.com
www1.netflix.com
www1stage.netflix.com
www2.netflix.com
www3.netflix.com
www5.netflix.com
yellow2admin-prod.netflix.com
oferta--netflix.com
ofertas--netflix.com
promocao--netflix.com
ultrahd--netflix.com
(kali㉿kali)-[~/Sublist3r]
$

Knock – Python Script

Github Repository Link: https://github.com/guelfoweb/knock

Usage:

```
python3 knockpy.py netflix.com
```

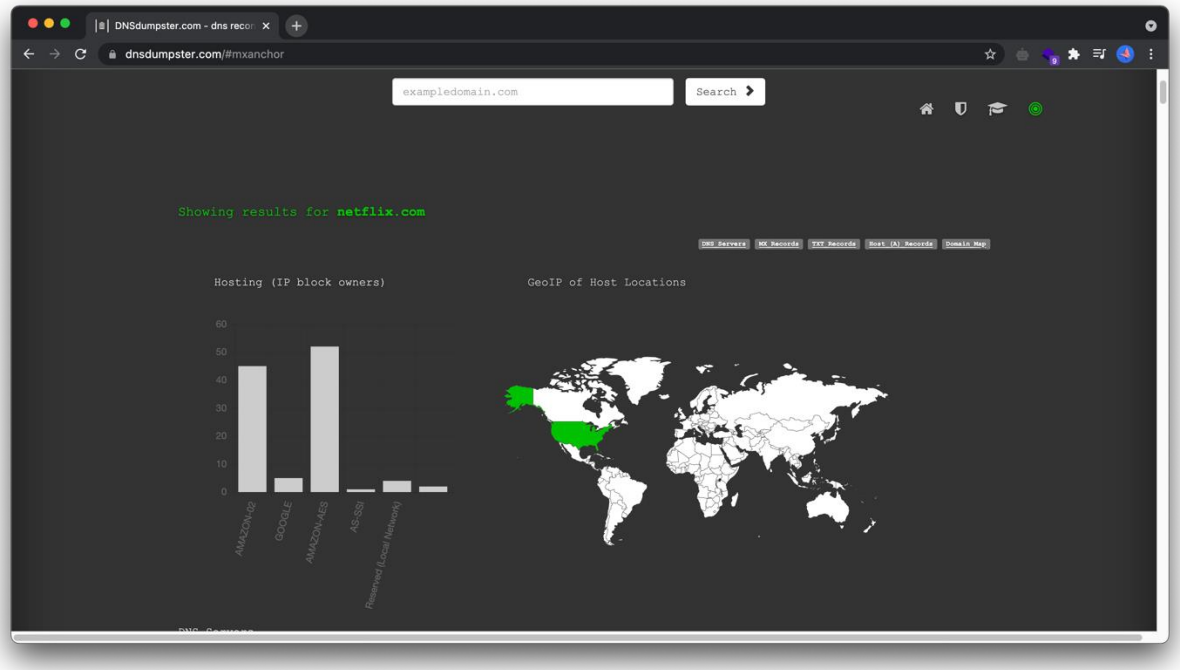
Scan Results: *This subdomain finder shows active subdomains and its IP address, status code, hosted server, and real hostname of its.*

```
kali@kali: ~
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$ knockpy netflix.com

 _ __
| |/ /              | |   v5.1.0
| ' / _ __   ___   _| | ___ __  _ _
|  < | '_ \ / _ \ / _ |/ / '_ \| | |
| . \| | | | (_) | (_| |  <| |_) | |_| |
|_|\_\_| |_|\___/ \__,_|_|\_\ .__/ \__, |
                             | |     __/ |
                             |_|    |___/

local: 2019 | google: 2 | duckduckgo: 5 | virustotal: 0

Wordlist: 2026 | Target: netflix.com | Ip: 3.251.50.149

05:37:57

Ip address      Code Subdomain                Server                      Real hostname
--------------- ---- ------------------------ --------------------------- -------------------------
34.205.97.36    200  about.netflix.com        Apache                      pubpresssite-2140919448.us-east-1.elb.amazonaws.com
52.38.7.83      200  account.netflix.com      nq_website_nonmember-prod-r dualstack.apiproxy-website-nlb-prod-1-bcf28d21f4bbcf2c.elb.us-west-2.amazonaws.
com
52.42.41.159    404  api.netflix.com          api-prod i-06e79382098793b5 api.us-west-2.origin.prodaa.netflix.com
52.38.7.83      200  app.netflix.com          nq_website_nonmember-prod-r dualstack.apiproxy-website-nlb-prod-1-bcf28d21f4bbcf2c.elb.us-west-2.amazonaws.
com
52.31.48.193         athena.netflix.com                                   detour.prod.netflix.net
207.45.73.144        b2b.netflix.com
52.31.48.193    404  blog.netflix.com         Apache                      detour.prod.netflix.net
44.236.157.89   200  brand.netflix.com        nginx                       brand-prod-external-1837959155.us-west-2.elb.amazonaws.com
104.18.119.155  200  cache.netflix.com        cloudflare                  netflix-sites.cloudcannon.com
172.217.194.121 200  calendar.netflix.com                                 ghs.googlehosted.com
52.39.128.98         catalog.netflix.com                                  dualstack.apiproxy-catalog-vpc0-1424560578.us-west-2.elb.amazonaws.com
104.75.84.11    404  cdn.netflix.com          AkamaiNetStorage            a1386.g.akamai.net
204.236.236.127 200  contact.netflix.com      poe-secure i-092f8055e87dce
44.234.232.238  404  css.netflix.com          api-prod i-093f9c45dd744989 dualstack.apiproxy-website-nlb-prod-2-e98cb8cf33ff3581.elb.us-west-2.amazonaws.
com
207.210.238.73       delta.netflix.com
34.252.74.1     200  developer.netflix.com    nq_website_nonmember-prod-r detour.prod.netflix.net
142.251.10.121  200  drive.netflix.com                                    ghs.googlehosted.com
207.45.72.201   200  dvd.netflix.com
74.125.200.121  200  email.netflix.com                                    ghs.googlehosted.com
142.251.10.121  200  employees.netflix.com    GSE                         ghs.googlehosted.com

18.232.2.164    200  g.netflix.com            GSE                         go2-meechum.prod.netflix.net
10.45.64.1           gateway.netflix.com
18.232.2.164    200  go.netflix.com           GSE                         go2-meechum.prod.netflix.net
142.251.10.121  200  groups.netflix.com       GSE                         ghs.googlehosted.com
44.226.110.19   200  help.netflix.com         poe-secure i-049eca6259ff23 dualstack.padme--uswest-vpc0-735221756.us-west-2.elb.amazonaws.com
104.75.84.10    404  image.netflix.com        AkamaiNetStorage            a743.g.akamai.net
104.75.84.9     404  images.netflix.com       AkamaiNetStorage            a743.g.akamai.net
44.240.158.19   404  info.netflix.com         api-prod i-07bc7762eb0fc710 dualstack.apiproxy-website-nlb-prod-1-bcf28d21f4bbcf2c.elb.us-west-2.amazonaws.
com
46.137.171.215  200  ir.netflix.com           cloudflare                  detour.prod.netflix.net
100.81.15.115        jira.netflix.com                                     corpjiraprod.mgmt.netflix.net
54.149.198.125  200  jobs.netflix.com         atsjobsiteui 448e70f1-e013- apiproxy-island-atsjobsiteui-v2-392483922.us-west-2.elb.amazonaws.com
100.127.235.114      mail.netflix.com                                     internal-vpc0-mailrelay-268773750.us-west-2.elb.amazonaws.com
172.26.1.8           manage.netflix.com
54.156.127.29   200  media.netflix.com        Apache                      pubmediacentersite-643055187.us-east-1.elb.amazonaws.com
46.137.171.215  200  movies.netflix.com       nq_website_nonmember-prod-r detour.prod.netflix.net
52.3.77.81      200  partner.netflix.com
172.26.2.250         pix.netflix.com
46.137.171.215  200  pr.netflix.com           Apache                      detour.prod.netflix.net
52.207.189.14   200  research.netflix.com     researchml 1e64ea7a-b4f6-40 apiproxy-research-767688609.us-east-1.elb.amazonaws.com
45.57.90.1      403  secure.netflix.com       nginx                       sec-oc.netflix.com
91.235.133.103  400  secured.netflix.com      Apache                      h-netflix.online-metrix.net
44.240.158.19   404  service.netflix.com      api-prod i-0b6bb59b77e253f1 dualstack.apiproxy-website-nlb-prod-1-bcf28d21f4bbcf2c.elb.us-west-2.amazonaws.
com
34.252.74.1     200  shop.netflix.com         cloudflare                  detour.prod.netflix.net
52.41.95.74     200  signup.netflix.com       nq_website_nonmember-prod-r dualstack.ecweb-prod-vpc0-2058625167.us-west-2.elb.amazonaws.com
69.53.236.55         static.netflix.com
52.31.48.193    200  tv.netflix.com           nq_website_nonmember-prod-r detour.prod.netflix.net
104.18.118.155  409  updates.netflix.com      cloudflare                  netflix-sites.cloudcannon.com
216.35.131.141       vpn.netflix.com
44.242.13.161   200  www.netflix.com          nq_website_nonmember-prod-r dualstack.apiproxy-website-nlb-prod-1-bcf28d21f4bbcf2c.elb.us-west-2.amazonaws.
com
34.194.141.126       www1.netflix.com                                     dualstack.wwwservice--frontend-san-vpc0-376358133.us-east-1.elb.amazonaws.com
34.213.69.2          www2.netflix.com                                     dualstack.wwwservice--frontend-san-vpc0-1072607415.us-west-2.elb.amazonaws.com
34.216.233.138       www3.netflix.com                                     dualstack.wwwservice--frontend-san-vpc0-1072607415.us-west-2.elb.amazonaws.com
100.82.14.115        yellow.netflix.com                                   internal-marqueeui-prod-frontend-fame-1534222568.us-east-1.elb.amazonaws.com

05:52:12

Ip address: 106 | Subdomain: 53 | elapsed time: 00:14:15

┌──(kali㉿kali)-[~]
```

DNSdumpster – Online Tool

Link: https://dnsdumpster.com/

Scan Results: *This tool categorized each subdomain in to MX Records, Host (A) Records and DNS Servers. Moreover, it generated a domain map which is showing interconnected subdomains and dns servers.*



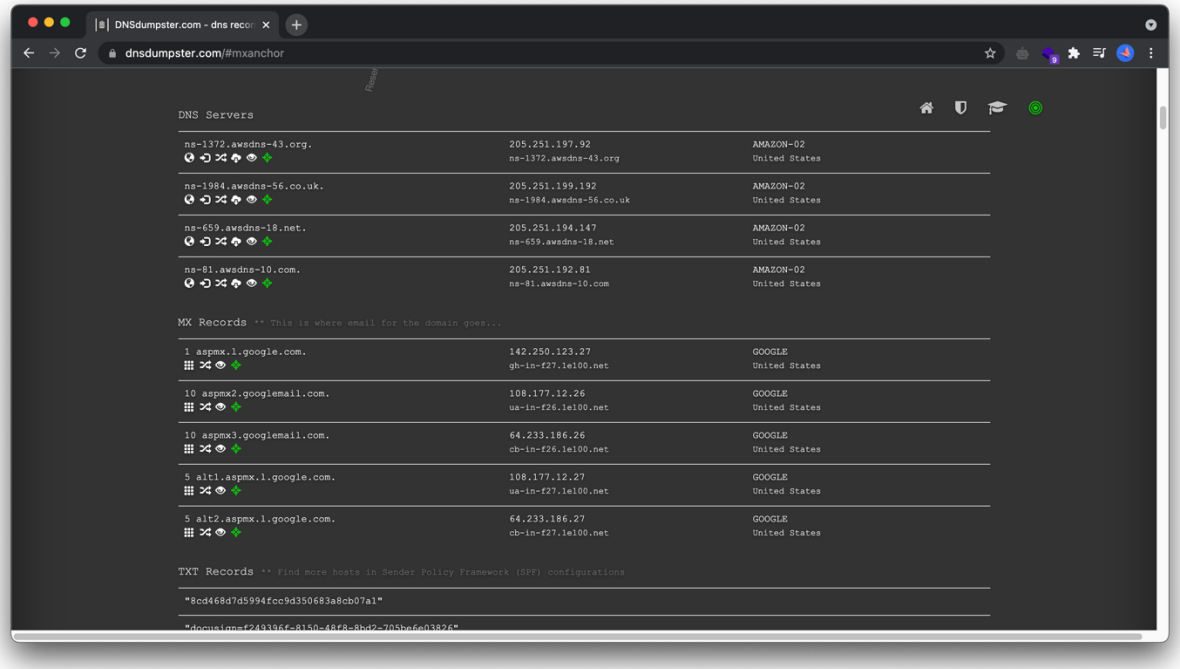
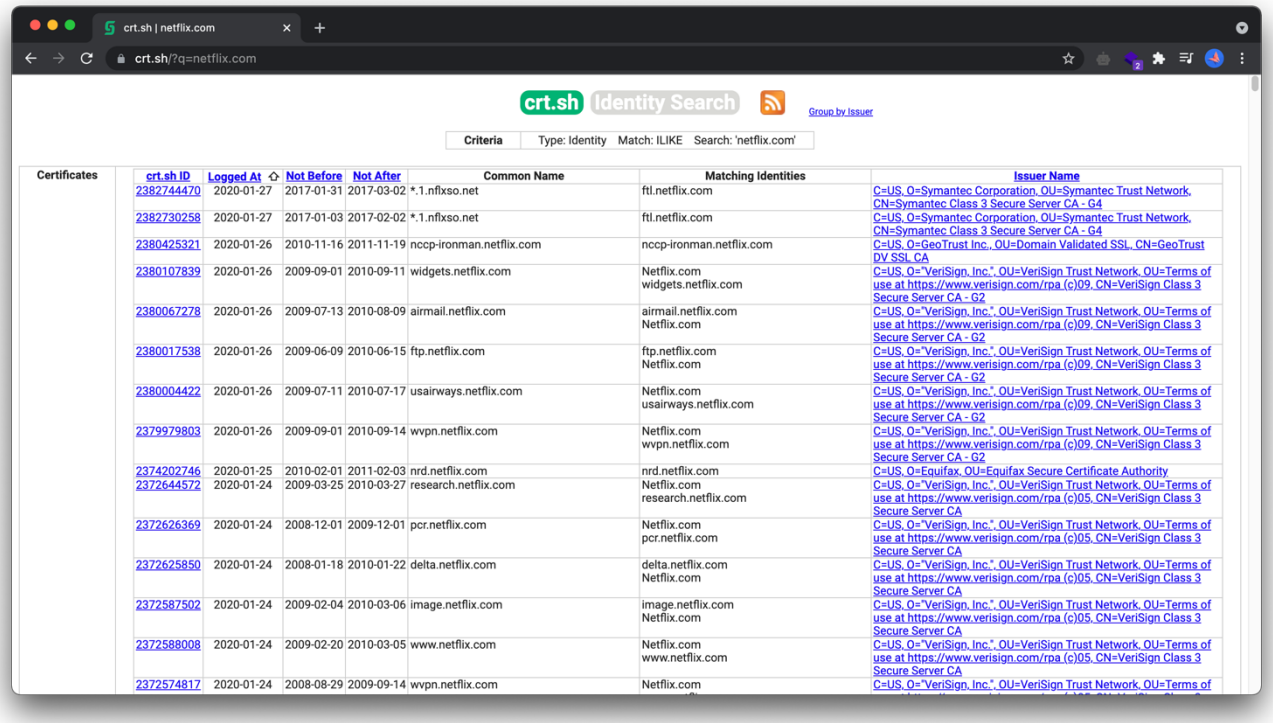


Figure 2: *In scope sections listed in Bugcrowd Netflix page*

Crt.sh – Online Tool

Link: https://crt.sh/

Scan Results:

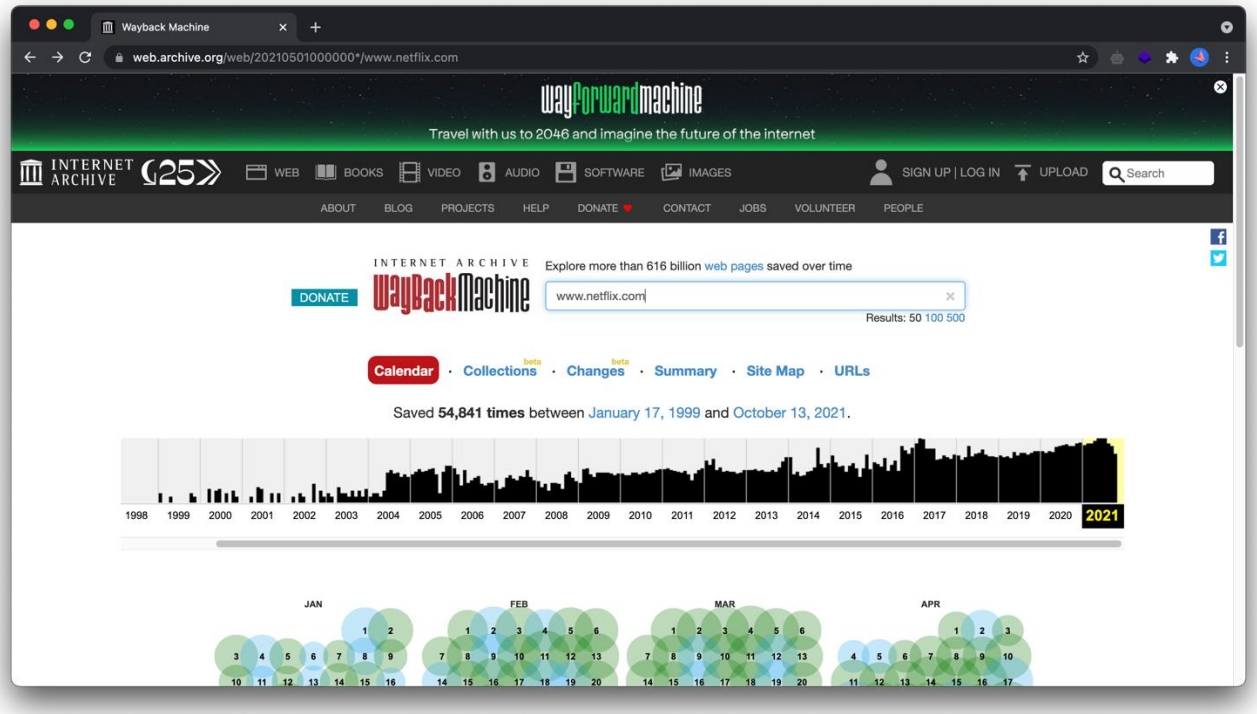


10.2 Website Archive Information

Wayback Machine

Link: https://archive.org/web/

Search Result: *This tool is used to view the website archive information from the beginning state. Here the Netflix.com website archive data information has been displayed from 1999 to 2021.*



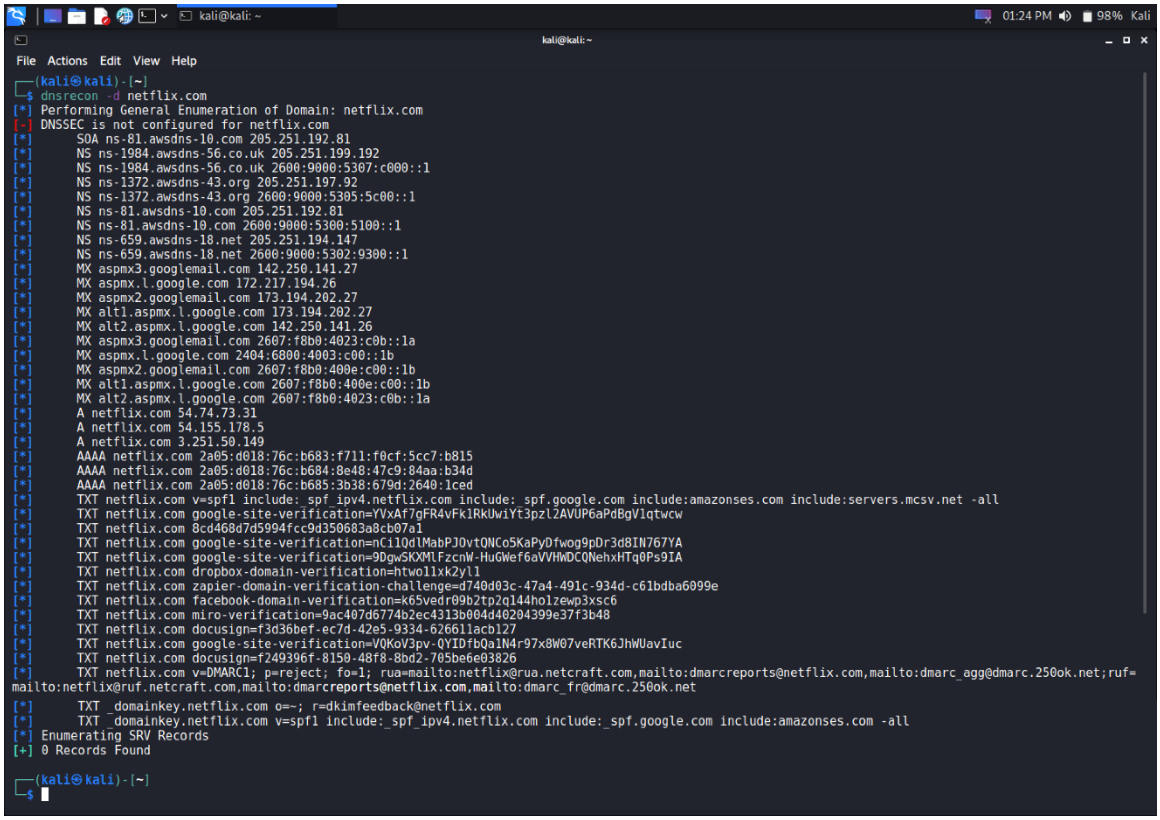
10.3 DNS Enumeration

dnsrecon – Kali Linux inbuilt Tool

Usage:

```
dnsrecon -d netflix.com
```

Scan Results: *Enumerate all DNS records of a web application.*



10.4 File structure Enumeration

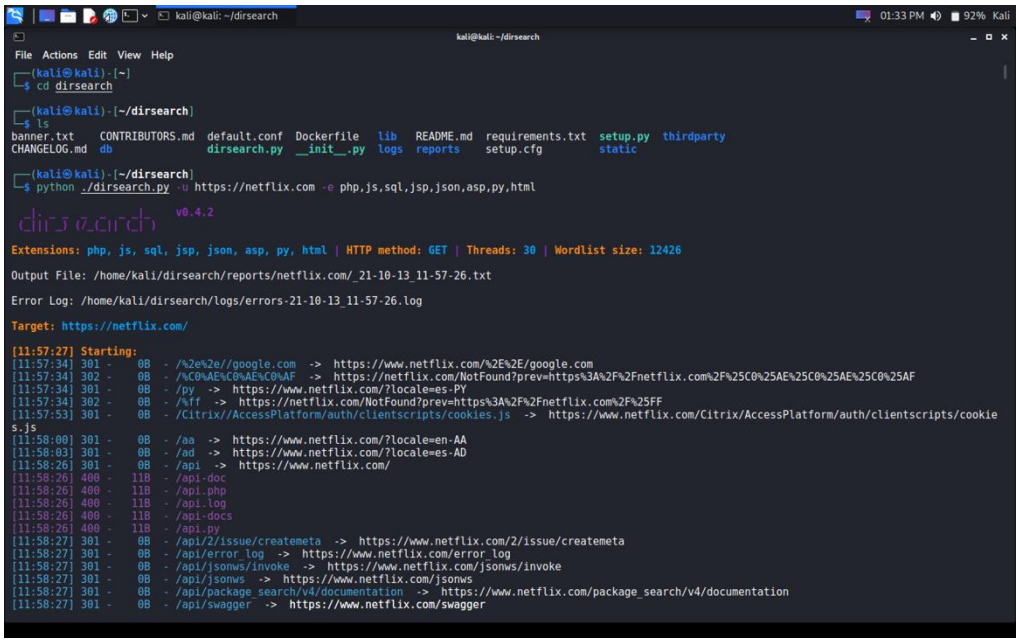
Dirsearch - Python Script

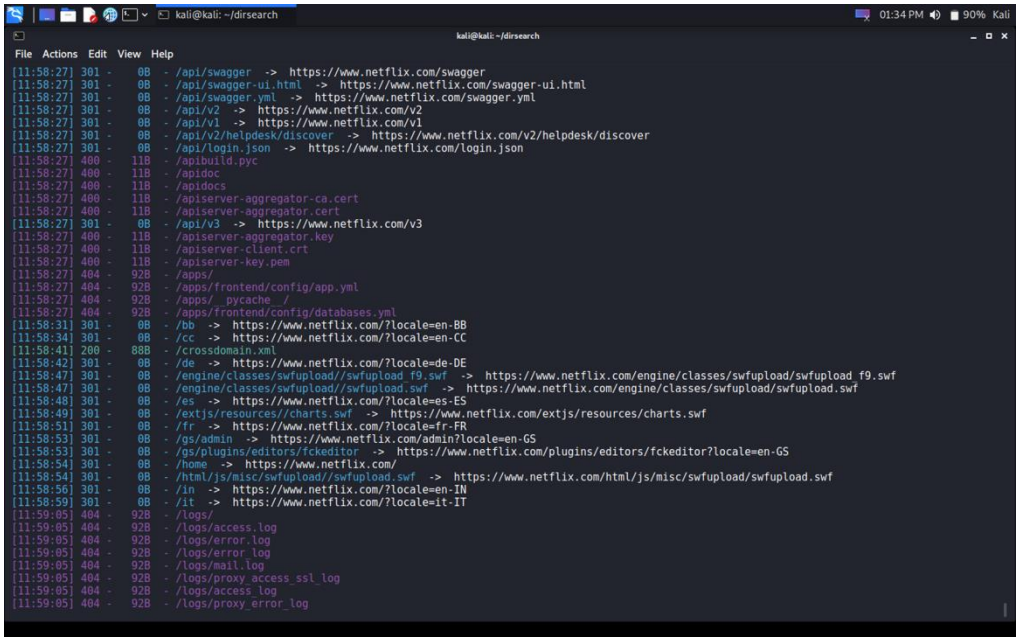
Github Repository Link: https://github.com/maurosoria/dirsearch

Usage:

```
python dirsearch.py -u https://netflix.com -e php,js,sql,jsp,json,asp,py,html
```

Scan Result: *This is used to brute force directories and files in webserver.*





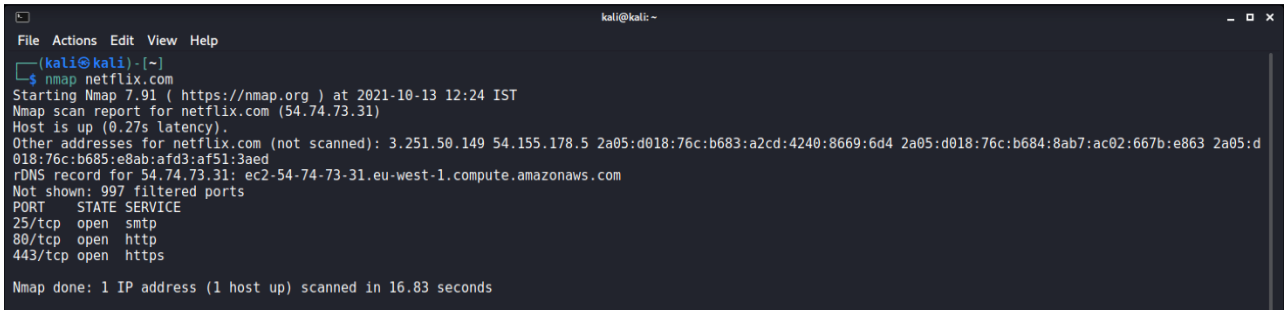
10.5 Gather ports and running services information

Nmap

Usage:

```
nmap netflix.com
```

Scan Result: *This scan returned port and running services information of the website.*



10.6 Gather Firewall protection Information

Wafwoof

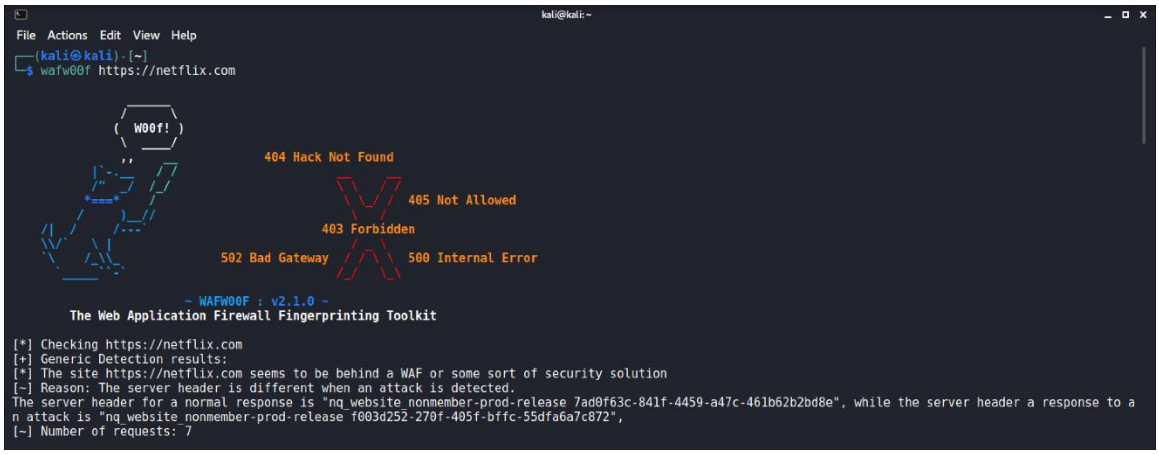
Github Repository Link: https://github.com/EnableSecurity/wafw00f

Usage:

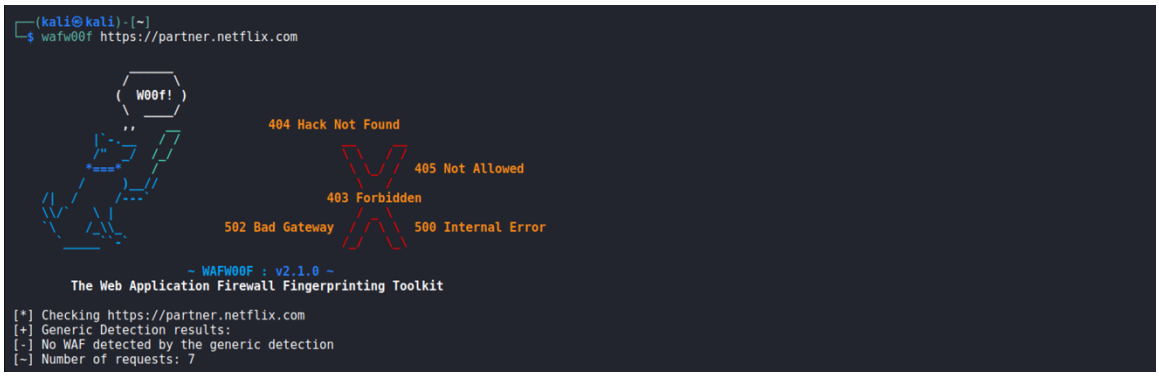
```
wafw00f https://netflix.com
```

Scan Results: *This tool returned firewall protection information of the domain. So, I did scan for the selected subdomains.*

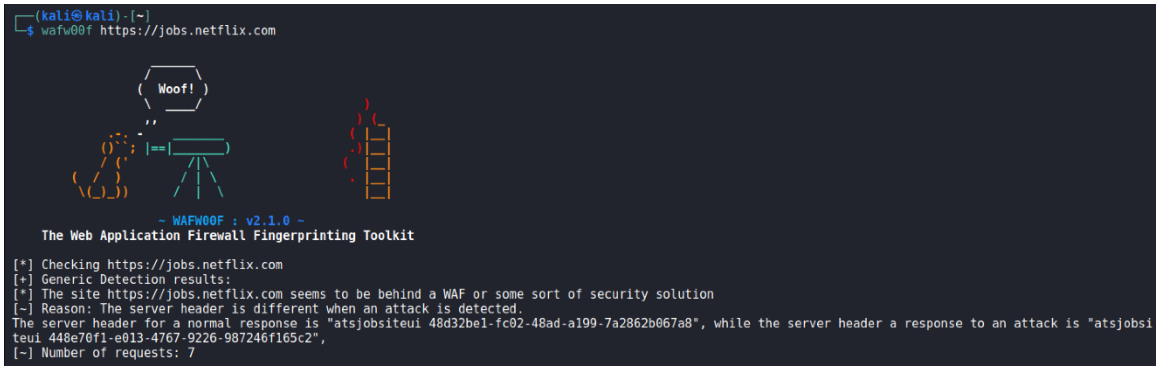
Netflix.com firewall protection details



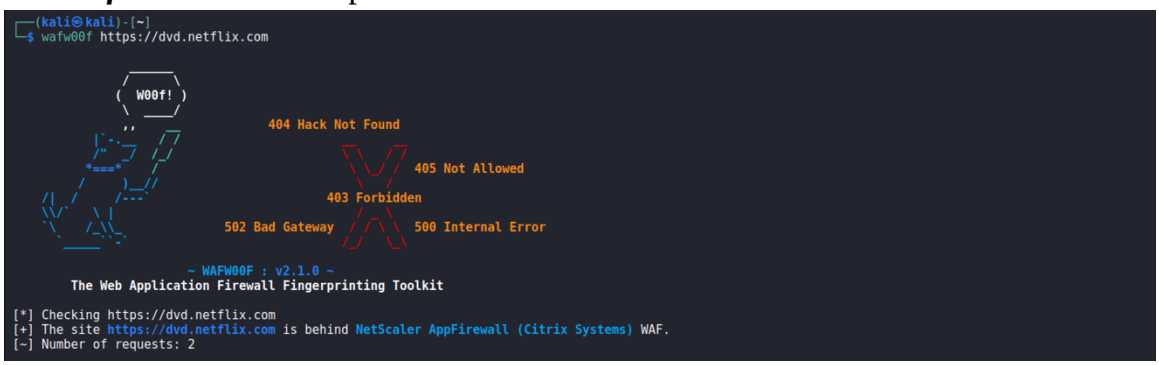
Partnet.netflix.com firewall protection details



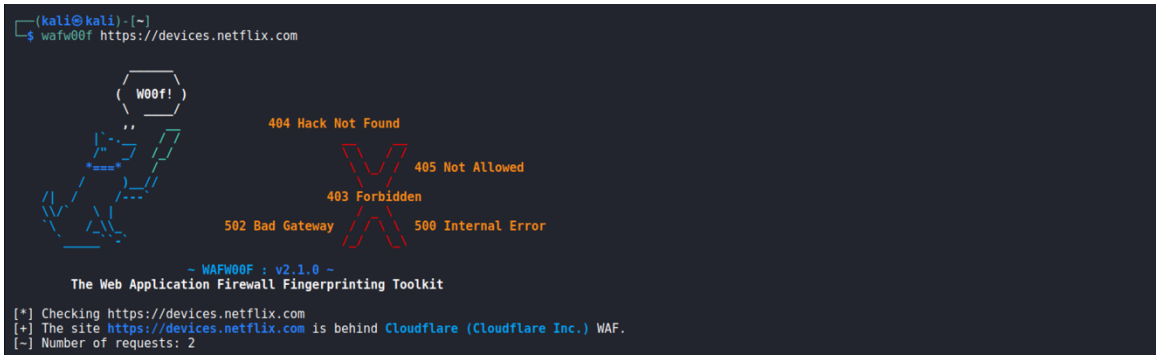
Jobs.netflix.com firewall protection details



dvd.netflix.com firewall protection details

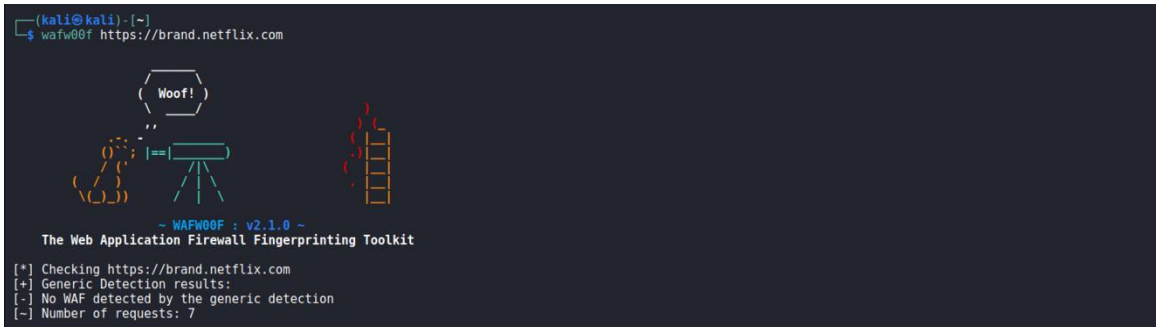


Devices.netflix.com firewall protection details



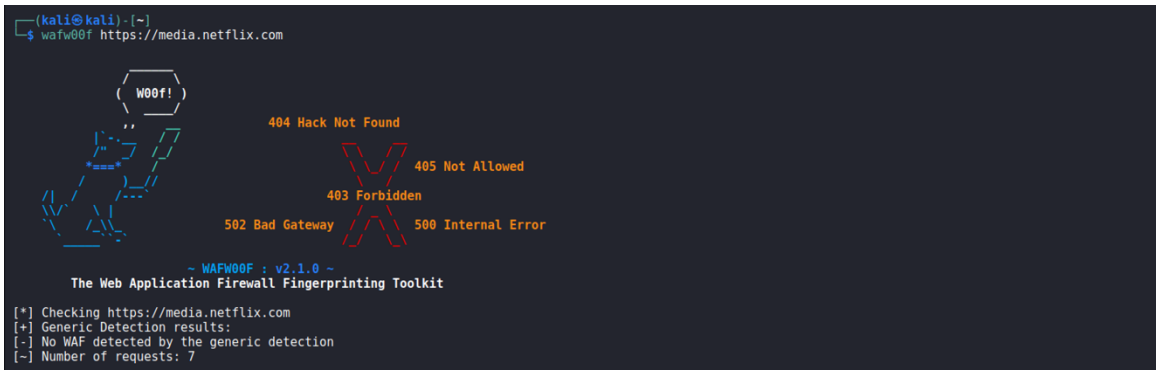
Meechum.netflix.com firewall protection details

Brand.netflix.com firewall protection details

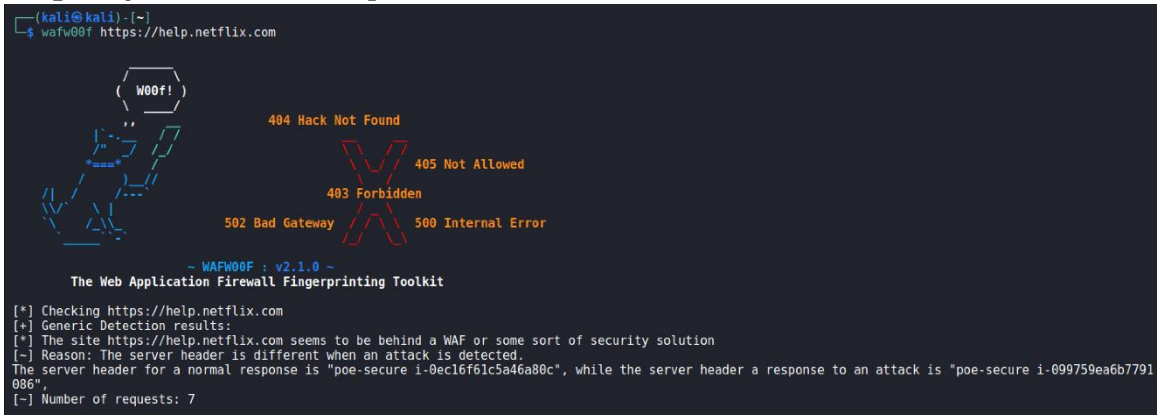


Candidate.netflix.com firewall protection details

Media.netflix.com firewall protection details



Help.netflix.com firewall protection details

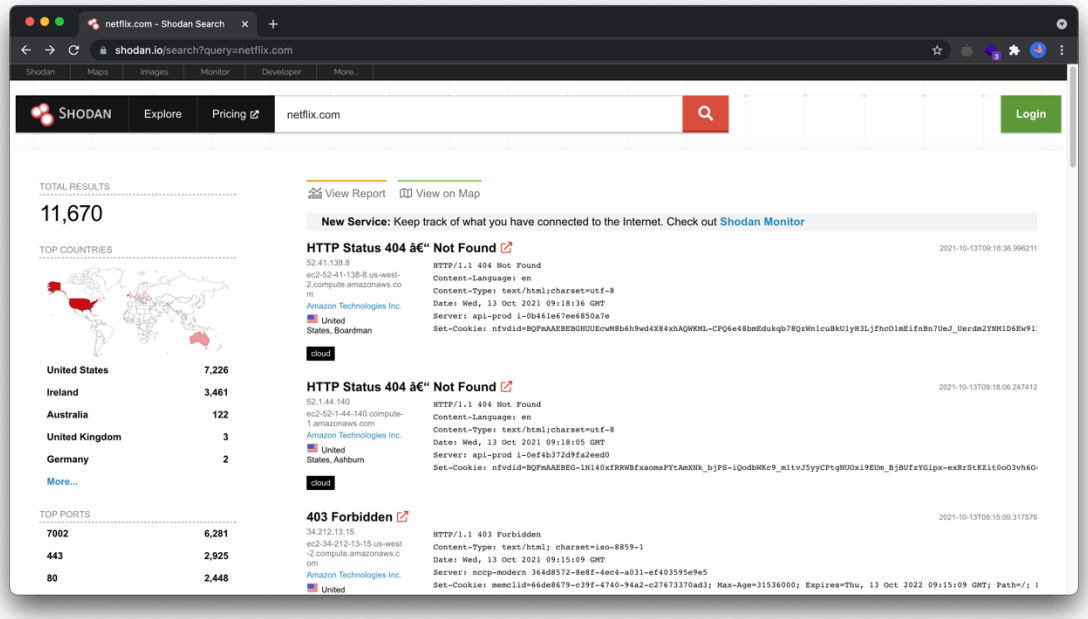


10.7 Publicly Connected Service Details Enumeration

Shodon.io – Online Tool

Link: https://www.shodan.io/

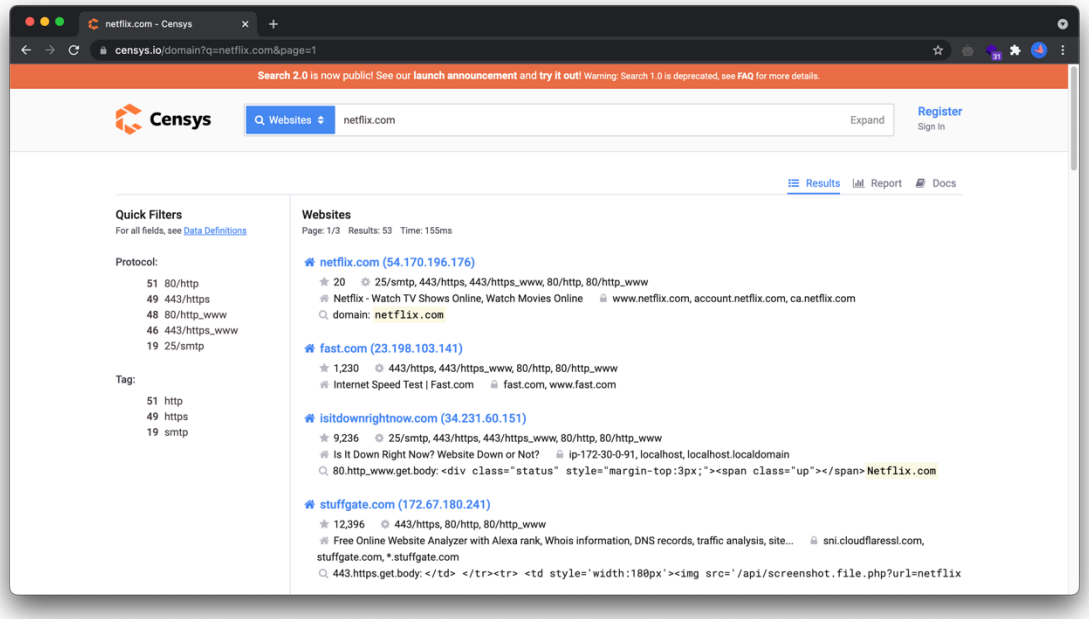
Scan Results:



Censys – Online Tool

Link: https://censys.io/domain

Scan Results:



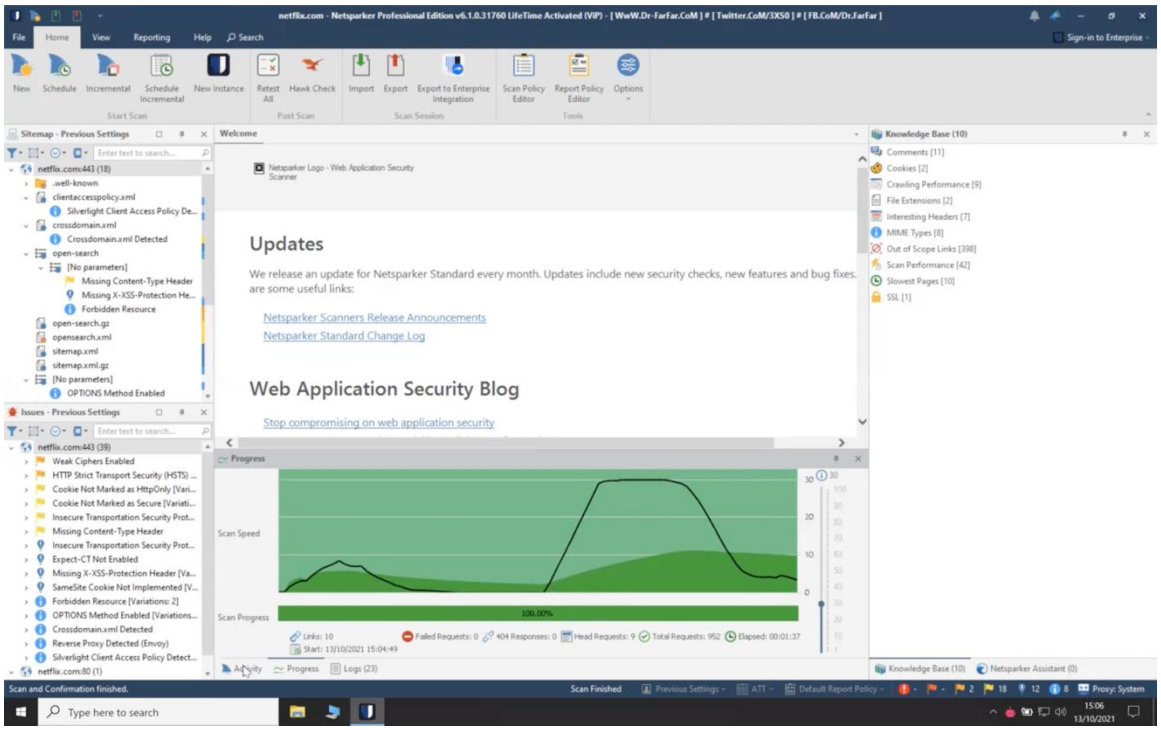
11. Vulnerability Analyzing and Reporting Phase

Tools I used for Vulnerability scanning

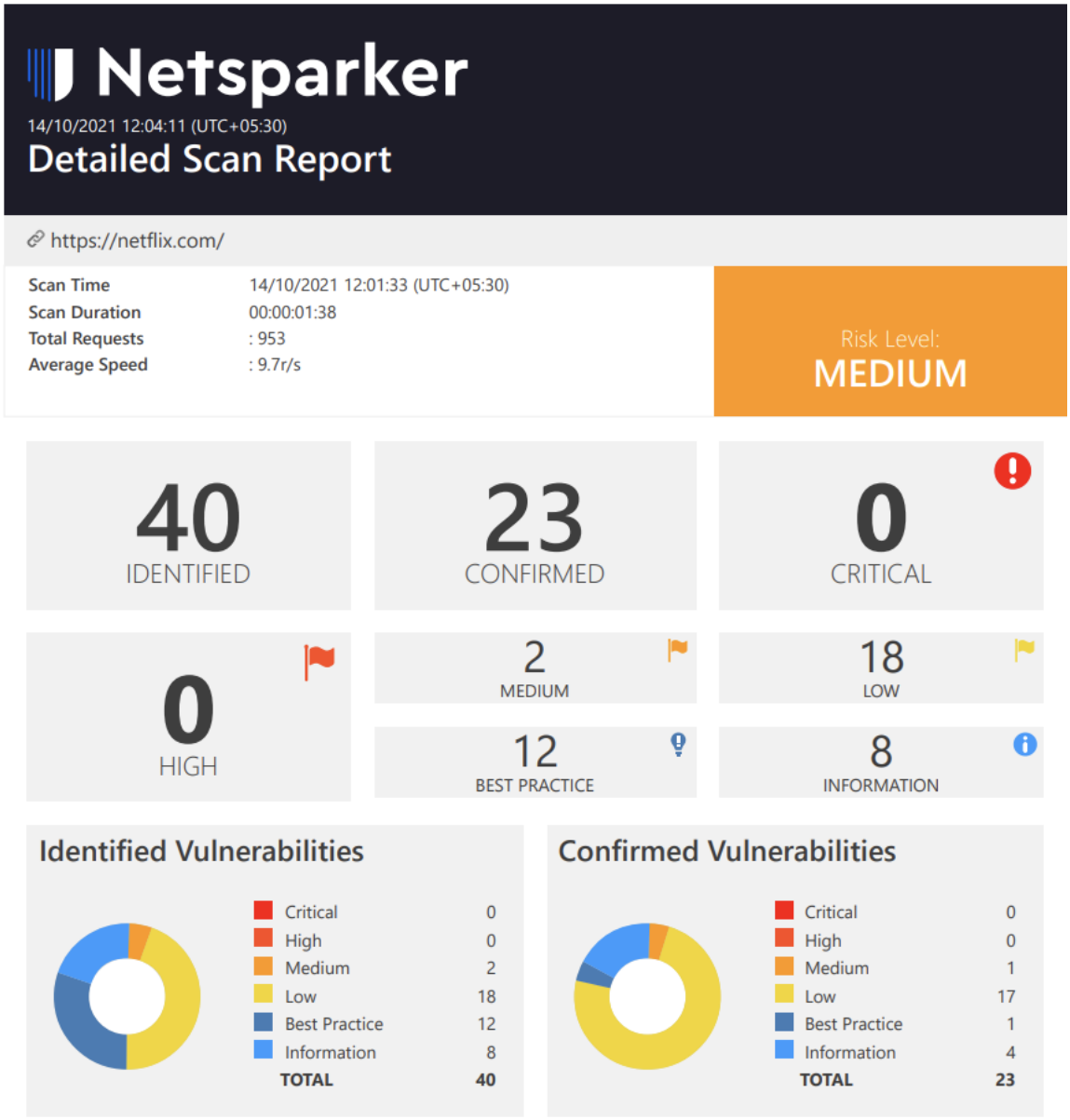
- Netsparker
- Nikto
- OWASP Zap Proxy

11.1 Netsparker – Automated Vulnerability Scanner

Netsparker is an automated vulnerability scanner which is used to find and analyze common vulnerabilities in web applications. Also, this scanner gives us the suggestion of remedy those found vulnerabilities. I did the scan process for root domain of Netflix.



Through this scanning I could able to generate Full-fledged vulnerability report. From this report I got 2 Medium risk level bugs, and 18 Low level Bugs, and some informative bugs.



This summary is cover page of the report which is graphically explains the whole report in a single view.

Following chart shows the name, risk level, method and URL of every vulnerability.

CONFIRM	VULNERABILITY	METHOD	URL	PARAMETER
	HTTP Strict Transport Security (HSTS) Errors and Warnings	GET	https://netflix.com/	
	Weak Ciphers Enabled	GET	https://netflix.com/	
	Missing Content-Type Header	HEAD	https://netflix.com/opensearch	
	Cookie Not Marked as HttpOnly	GET	https://netflix.com/	
	Cookie Not Marked as HttpOnly	GET	https://netflix.com/.well-known/	
	Cookie Not Marked as HttpOnly	GET	https://netflix.com/clientaccesspolicy.xml	
	Cookie Not Marked as HttpOnly	GET	https://netflix.com/crossdomain.xml	
	Cookie Not Marked as HttpOnly	GET	https://netflix.com/open-search.gz	
	Cookie Not Marked as HttpOnly	GET	https://netflix.com/opensearch.xml	
	Cookie Not Marked as HttpOnly	GET	https://netflix.com/sitemap.xml	
	Cookie Not Marked as HttpOnly	GET	https://netflix.com/sitemap.xml.gz	
	Cookie Not Marked as Secure	GET	https://netflix.com/	
	Cookie Not Marked as Secure	GET	https://netflix.com/.well-known/	
	Cookie Not Marked as Secure	GET	https://netflix.com/clientaccesspolicy.xml	
	Cookie Not Marked as Secure	GET	https://netflix.com/crossdomain.xml	
	Cookie Not Marked as Secure	GET	https://netflix.com/open-search.gz	

Cookie Not Marked as Secure	GET	https://netflix.com/opensearch.xml
Cookie Not Marked as Secure	GET	https://netflix.com/sitemap.xml
Cookie Not Marked as Secure	GET	https://netflix.com/sitemap.xml.gz
Insecure Transportation Security Protocol Supported (TLS 1.0)	GET	https://netflix.com/
Expect-CT Not Enabled	GET	https://netflix.com/
Missing X-XSS-Protection Header	HEAD	https://netflix.com/opensearch
Missing X-XSS-Protection Header	GET	https://netflix.com/swagger.json
SameSite Cookie Not Implemented	GET	https://netflix.com/
SameSite Cookie Not Implemented	GET	https://netflix.com/.well-known/
SameSite Cookie Not Implemented	GET	https://netflix.com/clientaccesspolicy.xml
SameSite Cookie Not Implemented	GET	https://netflix.com/crossdomain.xml
SameSite Cookie Not Implemented	GET	https://netflix.com/open-search.gz
SameSite Cookie Not Implemented	GET	https://netflix.com/opensearch.xml
SameSite Cookie Not Implemented	GET	https://netflix.com/sitemap.xml
SameSite Cookie Not Implemented	GET	https://netflix.com/sitemap.xml.gz
Insecure Transportation Security Protocol Supported (TLS 1.1)	GET	https://netflix.com/
Crossdomain.xml Detected	GET	https://netflix.com/crossdomain.xml

HTTP Strict Transport Security (HSTS) via HTTP	GET	http://netflix.com/
Reverse Proxy Detected (Envoy)	GET	https://netflix.com/
Silverlight Client Access Policy Detected	GET	https://netflix.com/clientaccesspolicy.xml
Forbidden Resource	HEAD	https://netflix.com/opensearch
Forbidden Resource	GET	https://netflix.com/swagger.json
OPTIONS Method Enabled	OPTIONS	https://netflix.com/
OPTIONS Method Enabled	OPTIONS	https://netflix.com/.well-known/

Let’s Discuss about some important vulnerabilities from this scan result report.

1. HTTP Strict Transport Security (HSTS) Errors and Warnings

MEDIUM | 1

Netsparker detected errors during parsing of Strict-Transport-Security header.

Impact

The HSTS Warning and Error may allow attackers to bypass HSTS, effectively allowing them to read and modify your communication with the website.

Vulnerabilities

1.1. https://netflix.com/

Error	Resolution
preload directive not present	Submit domain for inclusion in browsers' HTTP Strict Transport Security (HSTS) preload list.

Solution

Ideally, after fixing the errors and warnings, you should consider adding your domain to the HSTS preload list. This will ensure that browsers automatically connect your website by using HTTPS, actively preventing users from visiting your site using HTTP. Since this list is hardcoded in users' browsers, it will enable HSTS even before they visit your page for the first time, eliminating the need for Trust on First Use (TOFU) with its associated risks and disadvantages. Unless you fix the errors and warnings your website won't meet the conditions required to enter the browser's preload list.

Browser vendors declared:

- Serve a valid certificate.
- If you are listening on port 80, redirect all domains from HTTP to HTTPS on the same host. Serve all subdomains over HTTPS:
  - In particular, you must support HTTPS for the www subdomain if a DNS record for that subdomain exists
- Serve an HSTS header on the base domain for HTTPS requests:
  - The max-age must be at least 31536000 seconds (1 year)
  - ***includeSubDomains*** directive must be specified
  - The preload directive must be specified
  - If you are serving an additional redirect from your HTTPS site, that redirect must have the HSTS header (rather than the page it redirects to)

2. Weak Ciphers Enabled

MEDIUM 1 CONFIRMED 1

Netsparker detected that weak ciphers are enabled during secure communication (SSL).

You should allow only strong ciphers on your web server to protect secure communication with your visitors.

Impact

Attackers might decrypt SSL traffic between your server and your visitors.

Vulnerabilities

2.1. https://netflix.com/

CONFIRMED

List of Supported Weak Ciphers

- TLS_RSA_WITH_3DES_EDE_CBC_SHA (0x000A)
- TLS_RSA_WITH_AES_128_CBC_SHA (0x002F)
- TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xC013)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xC014)

Action to Take:

1. For Apache, you should modify the SSLCipherSuite directive in the `httpd.conf`.

```
SSLCipherSuite HIGH:MEDIUM:!MD5:!RC4
```

2. Lighttpd:

```
ssl.honor-cipher-order = "enable"
ssl.cipher-list = "EECDH+AESGCM:EDH+AESGCM"
```

3. For Microsoft IIS, you should make some changes to the system registry. **Incorrectly editing the registry may severely damage your system. Before making changes to the registry, you should back up any valued data on your computer.**

   a.Click Start, click Run, type `regedt32`or type `regedit`, and then click OK.
   b.In Registry Editor, locate the following registry key: `HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders`
   c.Set "Enabled" DWORD to "0x0" for the following registry keys:

```
SCHANNEL\Ciphers\DES 56/56
SCHANNEL\Ciphers\RC4 64/128
SCHANNEL\Ciphers\RC4 40/128
SCHANNEL\Ciphers\RC2 56/128
SCHANNEL\Ciphers\RC2 40/128
SCHANNEL\Ciphers\NULL
SCHANNEL\Hashes\MD5
```

Solution

Configure your web server to disallow using weak ciphers.

3. Cookie Not Marked as HttpOnly

LOW 8 | CONFIRMED 8

Netsparker identified a cookie not marked as HTTPOnly.

HTTPOnly cookies cannot be read by client-side scripts, therefore marking a cookie as HTTPOnly can provide an additional layer of protection against cross-site scripting attacks.

Impact

During a cross-site scripting attack, an attacker might easily access cookies and hijack the victim's session.

Vulnerabilities

3.1. https://netflix.com/

CONFIRMED

Identified Cookie(s)

- nfvdid
- memclid

Cookie Source

- HTTP Header

3.2. https://netflix.com/.well-known/

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

3.3. https://netflix.com/clientaccesspolicy.xml

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

3.4. https://netflix.com/crossdomain.xml

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

3.5. https://netflix.com/open-search.gz

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

3.6. https://netflix.com/opensearch.xml

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

3.7. https://netflix.com/sitemap.xml

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

3.8. https://netflix.com/sitemap.xml.gz

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

Solution

Mark the cookie as HTTPOnly. This will be an extra layer of defense against XSS. However, this is not a silver bullet and will not protect the system against cross-site scripting attacks. An attacker can use a tool such as XSS Tunnel to bypass HTTPOnly protection.

4. Cookie Not Marked as Secure

LOW 8 CONFIRMED 8

Netsparker identified a cookie not marked as secure, and transmitted over HTTPS.

This means the cookie could potentially be stolen by an attacker who can successfully intercept and decrypt the traffic, or following a successful man-in-the-middle attack.

Impact

This cookie will be transmitted over a HTTP connection, therefore if this cookie is important (*such as a session cookie*), an attacker might intercept it and hijack a victim's session. If the attacker can carry out a man-in-the-middle attack, he/she can force the victim to make an HTTP request to steal the cookie.

Vulnerabilities

4.1. https://netflix.com/

CONFIRMED

Identified Cookie(s)

- nfvdid
- memclid

Cookie Source

- HTTP Header

4.2. https://netflix.com/.well-known/

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

4.3. https://netflix.com/clientaccesspolicy.xml

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

4.4. https://netflix.com/crossdomain.xml

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

4.5. https://netflix.com/open-search.gz

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

4.6. https://netflix.com/opensearch.xml

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

4.7. https://netflix.com/sitemap.xml

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

4.8. https://netflix.com/sitemap.xml.gz

CONFIRMED

Identified Cookie(s)

- memclid

Cookie Source

- HTTP Header

Solution

Mark all cookies used within the application as secure.

5. Insecure Transportation Security Protocol Supported (TLS 1.0)

LOW 1 CONFIRMED 1

Netsparker detected that insecure transportation security protocol (TLS 1.0) is supported by your web server.

TLS 1.0 has several flaws. An attacker can cause connection failures and they can trigger the use of TLS 1.0 to exploit vulnerabilities like BEAST (Browser Exploit Against SSL/TLS).

Websites using TLS 1.0 are considered non-compliant by PCI since 30 June 2018.

Impact

Attackers can perform man-in-the-middle attacks and observe the encryption traffic between your website and its visitors.

Vulnerabilities

5.1. https://netflix.com/

CONFIRMED

Actions to Take

We recommended to disable TLS 1.0 and replace it with TLS 1.2 or higher. See Remedy section for more details.

Remedy

Configure your web server to disallow using weak ciphers. You need to restart the web server to enable changes.

- For Apache, adjust the SSLProtocol directive provided by the mod_ssl module. This directive can be set either at the server level or in a virtual host configuration.

```
SSLProtocol +TLSv1.2
```

- For Nginx, locate any use of the directive ssl_protocols in the `nginx.conf`file and remove `TLSv1`.

```
ssl_protocols TLSv1.2;
```

- For Microsoft IIS, you should make some changes on the system registry. **Incorrectly editing the registry may severely damage your system. Before making changes to the registry, you should back up any valued data on your computer.**
    1. Click on Start and then Run, type `regedt32`or `regedit`, and then click OK.
    2. In Registry Editor, locate the following registry key or create if it does not exist:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocol
s\TLS 1.0\
```

    3. Locate a key named `Server`or create if it doesn't exist.
    4. Under the `Server`key, locate a DWORD value named `Enabled`or create if it doesn't exist and set its value to "0".
- For lighttpd, put the following lines in your configuration file:

```
ssl.use-sslv2 = "disable"
ssl.use-sslv3 = "disable"
ssl.openssl.ssl-conf-cmd = ("Protocol" => "-TLSv1.1, -TLSv1, -SSLv3") # v1.4.48 or up
ssl.ec-curve = "secp384r1"
```

6. Missing Content-Type Header

LOW | 1

Netsparker detected a missing `Content-Type` header which means that this website could be at risk of a MIME-sniffing attacks.

Impact

MIME type sniffing is a standard functionality in browsers to find an appropriate way to render data where the HTTP headers sent by the server are either inconclusive or missing.

This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the intended content type.

The problem arises once a website allows users to upload content which is then published on the web server. If an attacker can carry out XSS (Cross-site Scripting) attack by manipulating the content in a way to be accepted by the web application and rendered as HTML by the browser, it is possible to inject code in e.g. an image file and make the victim execute it by viewing the image.

Vulnerabilities

6.1. https://netflix.com/opensearch

Remedy

1. When serving resources, make sure you send the content-type header to appropriately match the type of the resource being served. For example, if you are serving an HTML page, you should send the HTTP header:

```
Content-Type: text/html
```

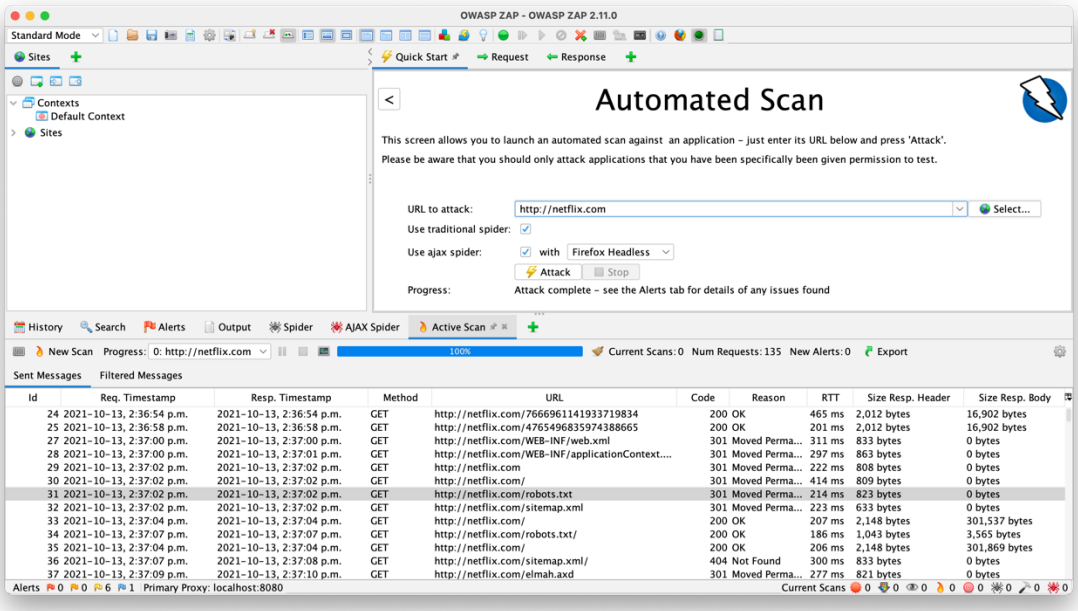
2. Add the X-Content-Type-Options header with a value of "nosniff" to inform the browser to trust what the site has sent is the appropriate content-type, and to not attempt "sniffing" the real content-type.

```
X-Content-Type-Options: nosniff
```

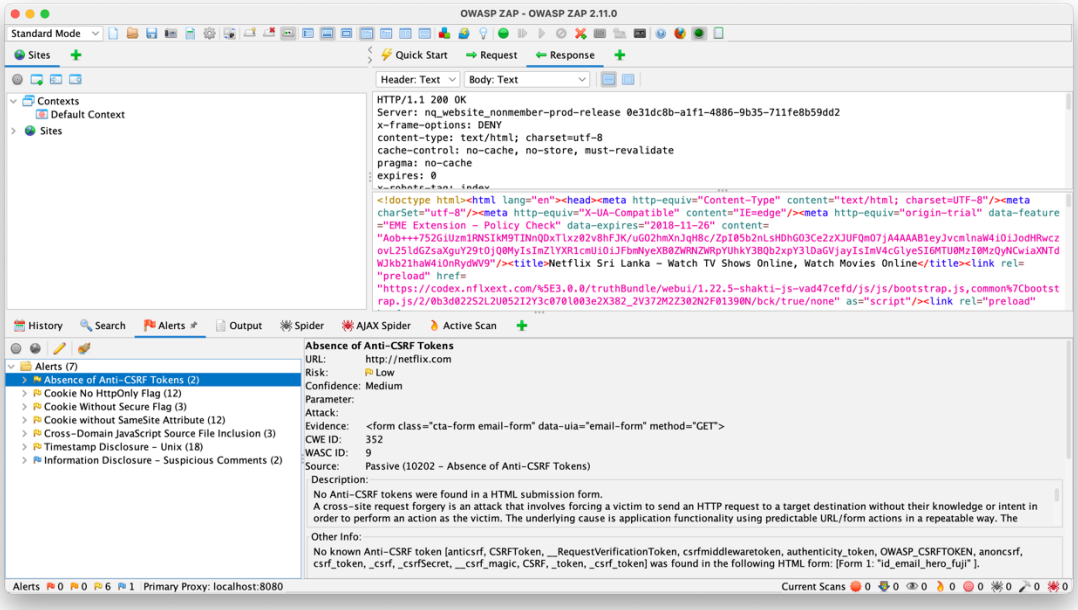
11.2 OWASP ZAP (Zed Attack Proxy) – OWASP Top 10 Vulnerabilities scanner

This is a learner friendly and pentester friendly free software which is developed and maintained by international volunteers to find web application vulnerabilities. This software works in Windows, Mac OS, Linux OS platforms. This software can generate easily understandable scanning report.

Active scanning



Found Vulnerabilities from ZAP



Zap proxy scan result report for **Netflix.com**

Summary of Alerts

Generated on Thu, 14 Oct 2021 22:41:06

Risk Level	Number of Alerts
High	0
Medium	0
Low	7
Informational	3

Alerts

Name	Risk Level	Number of Instances
Absence of Anti-CSRF Tokens	Low	2
Cookie No HttpOnly Flag	Low	11
Cookie Without SameSite Attribute	Low	11
Cookie Without Secure Flag	Low	3
Cross-Domain JavaScript Source File Inclusion	Low	3
Information Disclosure - Suspicious Comments	Informational	1
Timestamp Disclosure - Unix	Informational	17

Low (Medium)	Absence of Anti-CSRF Tokens
Description	No Anti-CSRF tokens were found in a HTML submission form. A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf. CSRF attacks are effective in a number of situations, including: * The victim has an active session on the target site. * The victim is authenticated via HTTP auth on the target site. * The victim is on the same local network as the target site. CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.
URL	http://netflix.com
Method	GET
Evidence	<form class="cta-form email-form" data-uia="email-form" method="GET">
URL	http://netflix.com
Method	GET
Evidence	<form class="cta-form email-form" data-uia="email-form" method="GET">
Instances	2
Solution	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. For example, use anti-CSRF packages such as the OWASP CSRFGuard. Phase: Implementation Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script. Phase: Architecture and Design

	Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330). Note that this can be bypassed using XSS. Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation. Note that this can be bypassed using XSS. Use the ESAPI Session Management control. This control includes a component for CSRF. Do not use the GET method for any request that triggers a state change. Phase: Implementation Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.
Other information	No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anoncsrf, csrf_token, _csrf, _csrfSecret, __csrf_magic, CSRF] was found in the following HTML form: [Form 2: "id_email_faq"].

Reference	http://projects.webappsec.org/Cross-Site-Request-Forgery http://cwe.mitre.org/data/definitions/352.html
CWE Id	352
WASC Id	9
Source ID	3

Low (Medium)	Cookie No HttpOnly Flag
Description	A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.
URL	https://netflix.com/sitemap.xml
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	https://netflix.com/
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	https://netflix.com/robots.txt
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
Instances	3
Solution	Ensure that the HttpOnly flag is set for all cookies.
Reference	https://owasp.org/www-community/HttpOnly
CWE Id	16
WASC Id	13
Source ID	3

Low (Medium)	Cookie No HttpOnly Flag
Description	A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.
URL	http://netflix.com/
Method	GET
Parameter	nfvdid
Evidence	Set-Cookie: nfvdid
URL	http://netflix.com
Method	GET
Parameter	clSharedContext
Evidence	Set-Cookie: clSharedContext
URL	http://netflix.com/
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	http://netflix.com/sitemap.xml
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	http://netflix.com/robots.txt
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	http://netflix.com
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	http://netflix.com/robots.txt
Method	GET
Parameter	nfvdid
Evidence	Set-Cookie: nfvdid
URL	http://netflix.com
Method	GET
Parameter	nfvdid
Evidence	Set-Cookie: nfvdid
Instances	8
Solution	Ensure that the HttpOnly flag is set for all cookies.
Reference	https://owasp.org/www-community/HttpOnly
CWE Id	16
WASC Id	13
Source ID	3

Low (Medium)	Cookie No HttpOnly Flag
Description	A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.
URL	https://netflix.com/sitemap.xml
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	https://netflix.com/
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	https://netflix.com/robots.txt
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
Instances	3
Solution	Ensure that the HttpOnly flag is set for all cookies.
Reference	https://owasp.org/www-community/HttpOnly
CWE Id	16
WASC Id	13
Source ID	3

Low (Medium)	Cookie No HttpOnly Flag
Description	A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.
URL	http://netflix.com/
Method	GET
Parameter	nfvdid
Evidence	Set-Cookie: nfvdid
URL	http://netflix.com
Method	GET
Parameter	clSharedContext
Evidence	Set-Cookie: clSharedContext
URL	http://netflix.com/
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	http://netflix.com/sitemap.xml
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	http://netflix.com/robots.txt
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	http://netflix.com
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	http://netflix.com/robots.txt
Method	GET
Parameter	nfvdid
Evidence	Set-Cookie: nfvdid
URL	http://netflix.com
Method	GET
Parameter	nfvdid
Evidence	Set-Cookie: nfvdid
Instances	8
Solution	Ensure that the HttpOnly flag is set for all cookies.
Reference	https://owasp.org/www-community/HttpOnly
CWE Id	16
WASC Id	13
Source ID	3

Low (Medium)	Cookie Without SameSite Attribute
Description	A cookie has been set without the SameSite attribute, which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.
URL	https://netflix.com/
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	https://netflix.com/sitemap.xml
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	https://netflix.com/robots.txt
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
Instances	3
Solution	Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.
Reference	https://tools.ietf.org/html/draft-ietf-httpbis-cookie-same-site
CWE Id	16
WASC Id	13
Source ID	3

Low (Medium)	Cookie Without SameSite Attribute
Description	A cookie has been set without the SameSite attribute, which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.
URL	http://netflix.com
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	http://netflix.com
Method	GET
Parameter	nfvdid
Evidence	Set-Cookie: nfvdid
URL	http://netflix.com/robots.txt
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	http://netflix.com/
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	http://netflix.com/sitemap.xml
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	http://netflix.com/robots.txt
Method	GET
Parameter	nfvdid
Evidence	Set-Cookie: nfvdid
URL	http://netflix.com/
Method	GET
Parameter	nfvdid
Evidence	Set-Cookie: nfvdid
URL	http://netflix.com
Method	GET
Parameter	clSharedContext
Evidence	Set-Cookie: clSharedContext
Instances	8
Solution	Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.
Reference	https://tools.ietf.org/html/draft-ietf-httpbis-cookie-same-site
CWE Id	16
WASC Id	13
Source ID	3

Low (Medium)	Cookie Without Secure Flag
Description	A cookie has been set without the secure flag, which means that the cookie can be accessed via unencrypted connections.
URL	https://netflix.com/robots.txt
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	https://netflix.com/sitemap.xml
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
URL	https://netflix.com/
Method	GET
Parameter	memclid
Evidence	Set-Cookie: memclid
Instances	3
Solution	Whenever a cookie contains sensitive information or is a session token, then it should always be passed using an encrypted channel. Ensure that the secure flag is set for cookies containing such sensitive information.
Reference	https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/06-Session_Management_Testing/02-Testing_for_Cookies_Attributes.html
CWE Id	614
WASC Id	13
Source ID	3

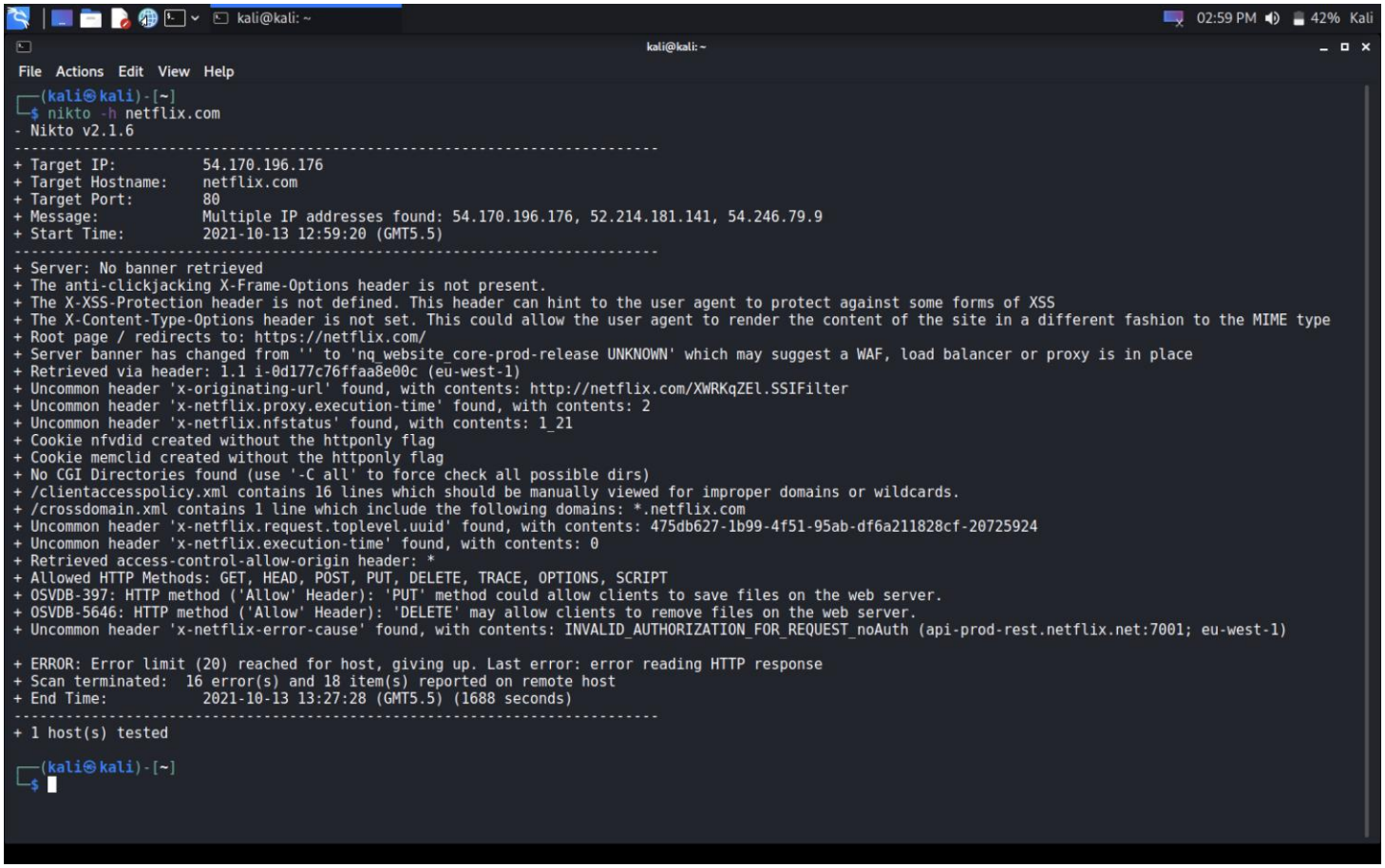
Low (Medium)	Cross-Domain JavaScript Source File Inclusion
Description	The page includes one or more script files from a third-party domain.
URL	http://netflix.com
Method	GET
Parameter	https://codex.nflxext.com/%5E3.0.0/truthBundle/webui/1.22.5-shakti-js-vad47cefd/js/js/bootstrap.js,common%7Cbootstrap.js/2/0b3d022S2L2U052I2Y3c070I003e2X382_2V372M2Z302N2F01390N/bck/true/none
Evidence	<script src="https://codex.nflxext.com/%5E3.0.0/truthBundle/webui/1.22.5-shakti-js-vad47cefd/js/js/bootstrap.js,common%7Cbootstrap.js/2/0b3d022S2L2U052I2Y3c070I003e2X382_2V372M2Z302N2F01390N/bck/true/none"></script>
URL	http://netflix.com
Method	GET
Parameter	https://codex.nflxext.com/%5E3.0.0/truthBundle/webui/1.22.5-shakti-js-vad47cefd/js/js/signup%7Cnmhp%7CnmhpFrameworkClient.js/2/0b3d022S2L2U052I2Y3c070I003e2X382_2V372M2Z302N2F01390N/l/true/none
Evidence	<script src="https://codex.nflxext.com/%5E3.0.0/truthBundle/webui/1.22.5-shakti-js-vad47cefd/js/js/signup%7Cnmhp%7CnmhpFrameworkClient.js/2/0b3d022S2L2U052I2Y3c070I003e2X382_2V372M2Z302N2F01390N/l/true/none"></script>
URL	http://netflix.com
Method	GET
Parameter	https://cdn.cookielaw.org/scripttemplates/otSDKStub.js
Evidence	<script type="text/javascript" charSet="UTF-8" data-domain-script="87b6a5c0-0104-4e96-a291-092c11350111" src="https://cdn.cookielaw.org/scripttemplates/otSDKStub.js"></script>
Instances	3
Solution	Ensure JavaScript source files are loaded from only trusted sources, and the sources can't be controlled by end users of the application.
Reference	
CWE Id	829
WASC Id	15
Source ID	3

11.3 Nikto Scanning

Usage:

```
nikto -h netflix.com
```

Scan Result: *Nikto is a kali Linux inbuilt tool which is used to scan webservers for malicious files or dangerous files, expired server data information.*

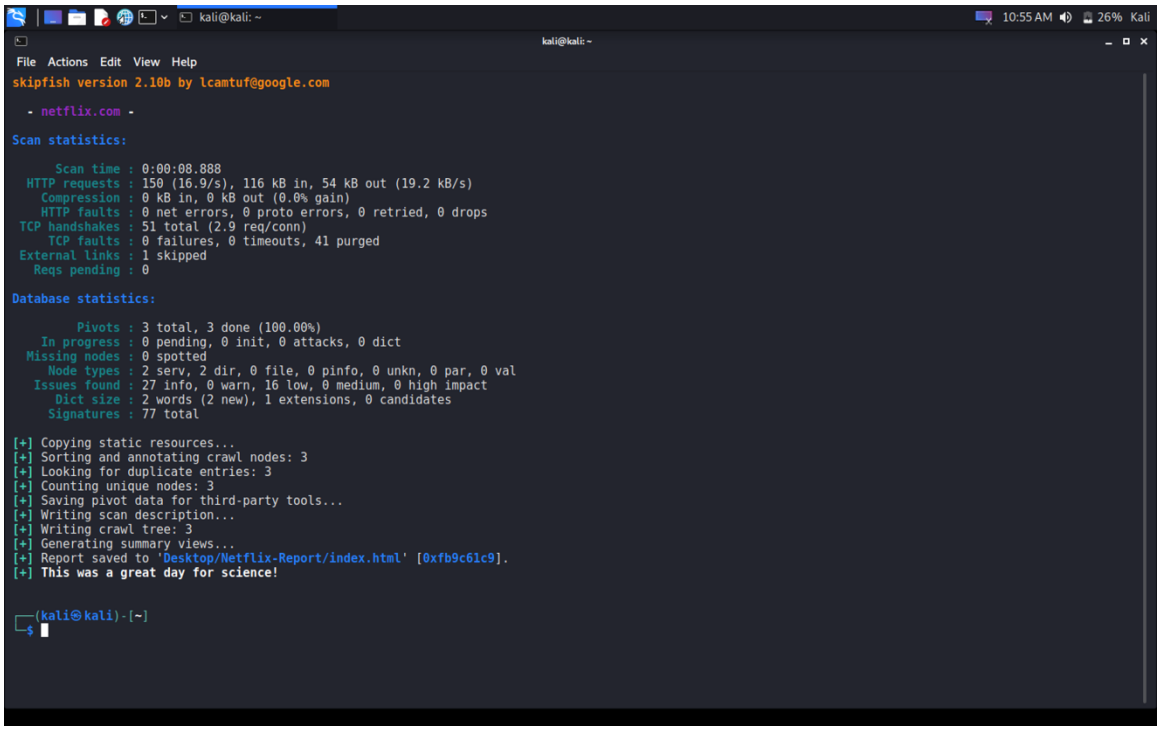


11.4 Skipfish Scanning

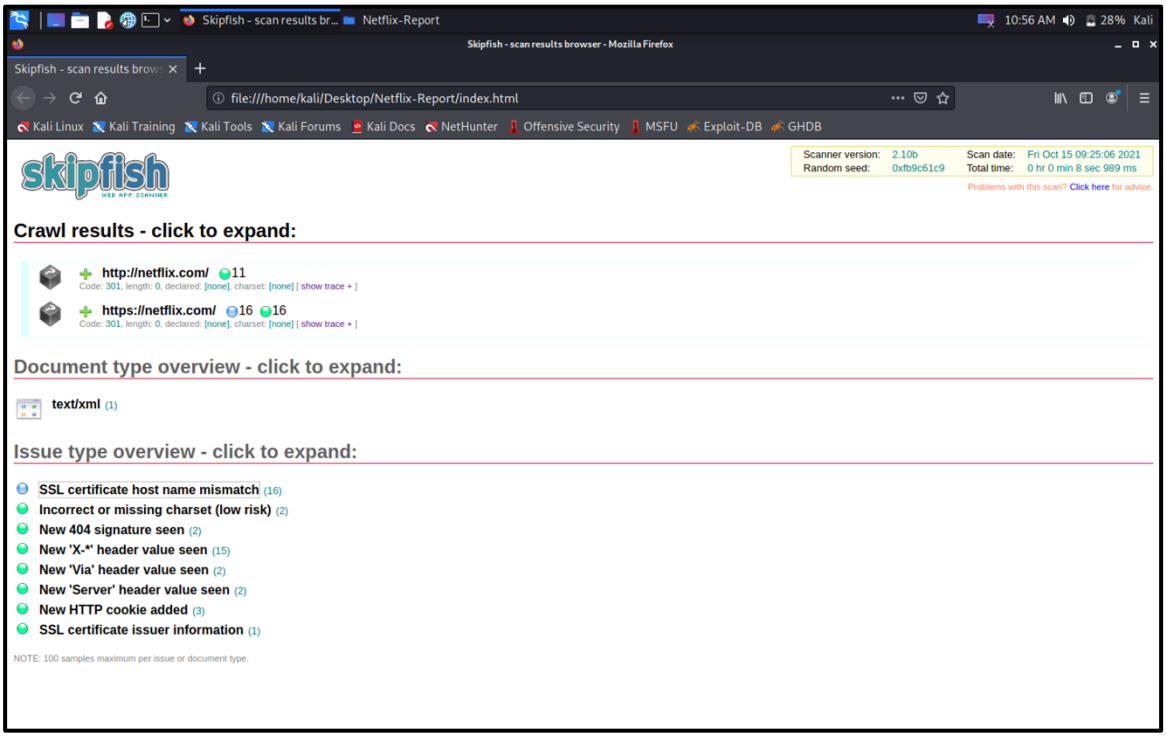
Usage:

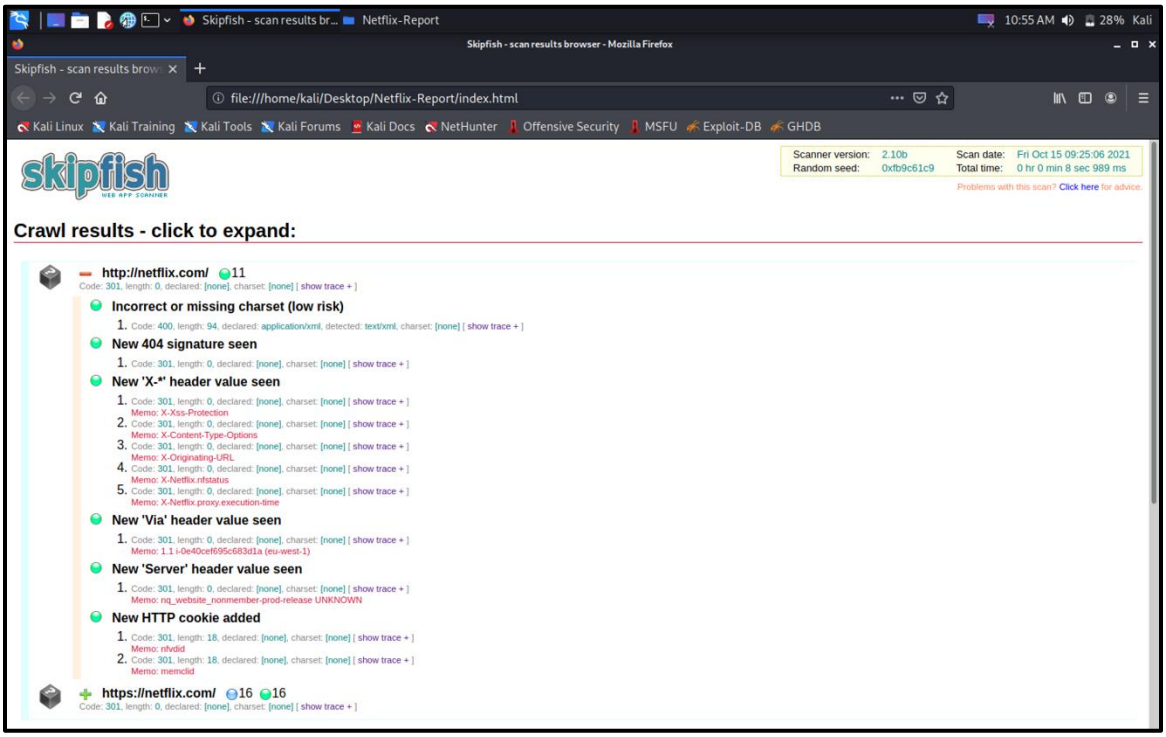
```
skipfish -o Desktop/Netflix-Report http://netflix.com
```

Scan Result:



Generated Report:





12. Conclusion and Recommendations

This report clearly explained about why information gathering is important in web application penetration testing and how the gathered informations are helpful in vulnerability assessment. Vulnerabilities are categorized by their severity as high, medium, low and informational. Informational bugs do not affect the business, but Critical, High, medium level bugs are major part of the security and they should have cleared.

From this assessment I learnt how to use tools efficiently and how to write a proper audit report. This assignment will be helpful to me in future when I am working and doing researches. Web security is one of the most important part of information security. If we don't secure applications, it will not only affect the website's confidentiality, availability, integrity. Also, it will affect the business related through the web applications. As a Cyber Security student, it is our responsibility to spread awareness about security threads in digital world. Be Secured.

13. References

[1] " OWASP Top Ten". Available: https://owasp.org/www-project-top-ten/. [Accessed: 15- October- 2021].

[2] " Netflix's Bug Bounty Program". Available: https://bugcrowd.com/netflix. [Accessed: 15- October- 2021].

[3] " How to Write a Better Vulnerability Report". Available: https://medium.com/swlh/how-to-write-a-better-vulnerability-report-20163ab913fb. [Accessed: 15- October- 2021].https://medium.com/swlh/how-to-write-a-better-vulnerability-report-20163ab913fb

[4] " Sublist3r". https://github.com/aboul3la/Sublist3r

[5] " Knockpy". https://github.com/guelfoweb/knock

[6] " Dirsearch". https://github.com/maurosoria/dirsearch

[7] " Netsparker Reporting Overview video walkthroughs". https://www.youtube.com/watch?v=wpFo6uPAgJY

[8] " ZAP Deep Dive: Report Generation". https://www.youtube.com/watch?v=kD540gUWJ3I

Thank You