Sri Lanka Institute of Information Technology

VULNERABILITY ASSESSMENT - WEB AUDIT

https://www.grammarly.com/

Individual Assignment

IE2062 – Web Security

Submitted by:

Student Registration Number	Student Name
IT19976686	**Jayasinghe D.A.**

Date of submission: 31/05/2021

Table of Contents

Acknowledgement

I would like to thank Ms. Chethana Liyanapathirana, the lecturer in charge of the Web Security module, for her insightful feedback and advice, which was crucial to the start of this web audit.

I would like to express my gratitude to Dr. Lakmal Rupasinghe, Ms. Lanessha Rukgahakotuwa, and Ms. Chathu Udagedara for their assistance and advice during this web audit.

Assessment Opjectives

We were given the option of performing a web audit on any domain we like. The tests started on the 20th of April and ended on the 30th of May. This paper demonstrates how to perform a primary web audit that focuses on the topics of; information gathering, reconnaissance, discovering and scanning, and the vulnerability assessment**.**

Application Credentials and URL

The audit was done on the following URLs.

Target domain:

- ✓ https://www.grammarly.com/

Target sub-domains:

- ✓ https://account.grammarly.com/
- ✓ https://app.grammarly.com/

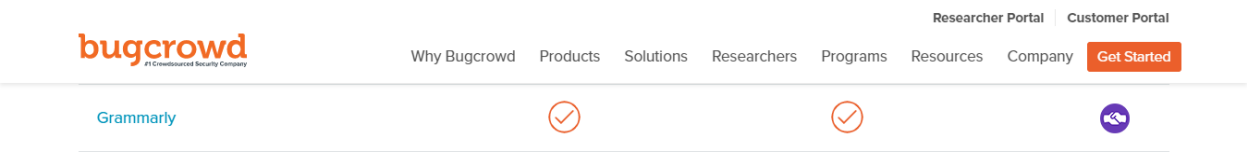
What is Website Audit?

A website audit is a process of assessing a web system which covers websites as well as web applications for vulnerabilities and loopholes. There are many explanations for a website audit, but the most important ones are usually SEO (Search Engine Optimization) and content marketing.

An SEO-based website audit identifies weak points in a website's SEO score and leads to a better understanding of its SEO status. The content audit is used to assess engagement and evaluate improvements to the content strategy in order to increase the website's performance. A systematic cyber protection audit examines all information infrastructure properties for safety protocols, security measures, and future risks. SQL Injections and Cross Site Scripting (XSS) are examples of cyber-attack tactics that pose a significant danger to data security and can result in the full compromise of computer networks. The biggest vulnerability risks associated with web-based applications are caused by web vulnerabilities. Web vulnerability audits involve any of a website's secret or undisclosed bugs, as well as the security infrastructure. Vulnerability scanners are the most popular method for evaluating a website's security.

Domain selection

Using "https://www.bugcrowd.com" website, I have selected this "Grammarly" for my web audit as shown in bellow.



Introduction

You have already heard of Grammarly if you do some sort of online writing. It is the most common spelling and grammar checker on the internet, with over 10 million regular active users. Grammarly analyzes and scans a piece of text you have composed for mistakes. The free version adds the fundamentals (grammar, punctuation, and spelling), while the premium version goes a little further by providing stylistic and best-writing-practice recommendations. Grammarly will help you polish your tweets, social media articles, journals, and formal documents, no matter what kind of writing you are doing – professional, informal, or hobby writing.

To that end, I believe the following people would learn the most from Grammarly:

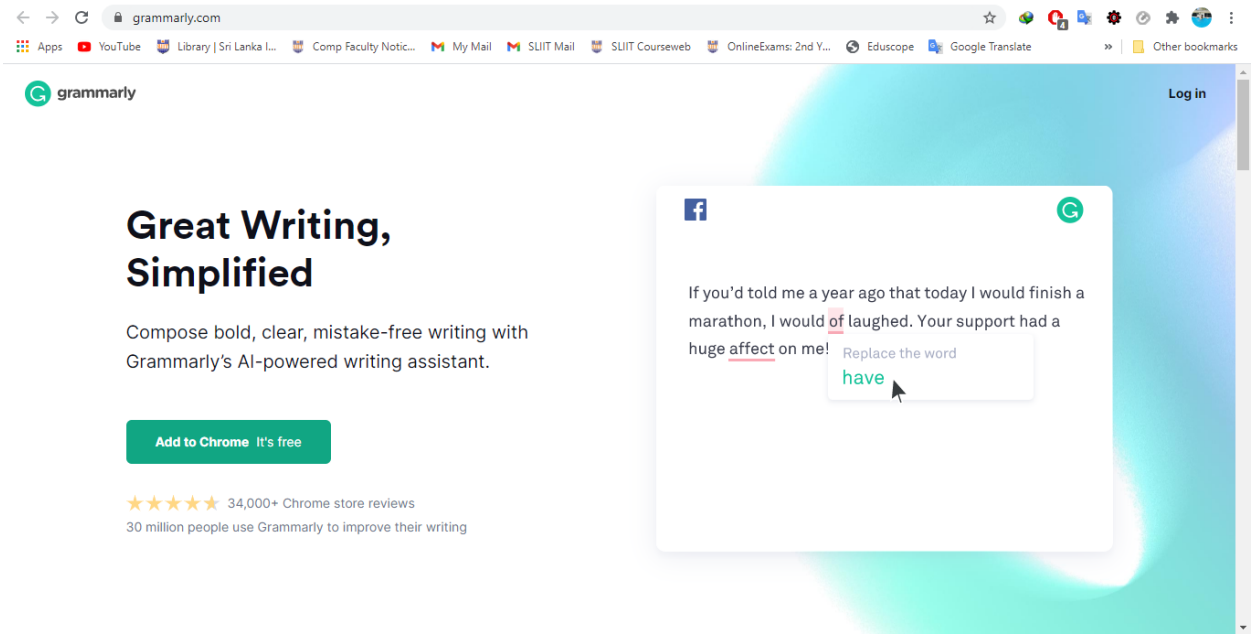
- Students
- Bloggers
- Copywriters and content marketers
- Professionals and business authors who need to create an accurate article, presentation, or email.

The different ways to use Grammarly:

Since Grammarly is an internet-based tool, in order to use it u need to be connected to the internet. Also, there are various ways to check your work through it.

1. **The Grammarly web tool**
2. **Desktop app**
3. **Microsoft Word**
4. **Browser extension**
5. **On your phone**

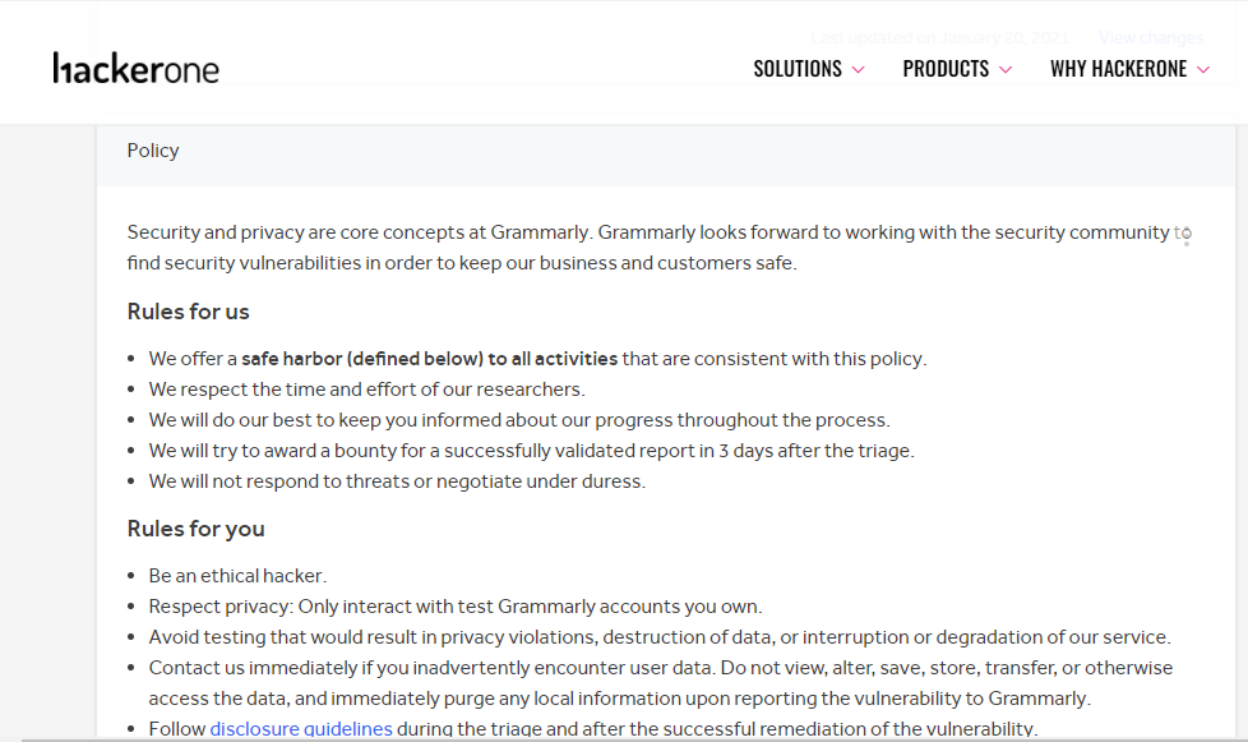
The below figure shows the interface of the “grammarly.com”.



1. Information Gathering

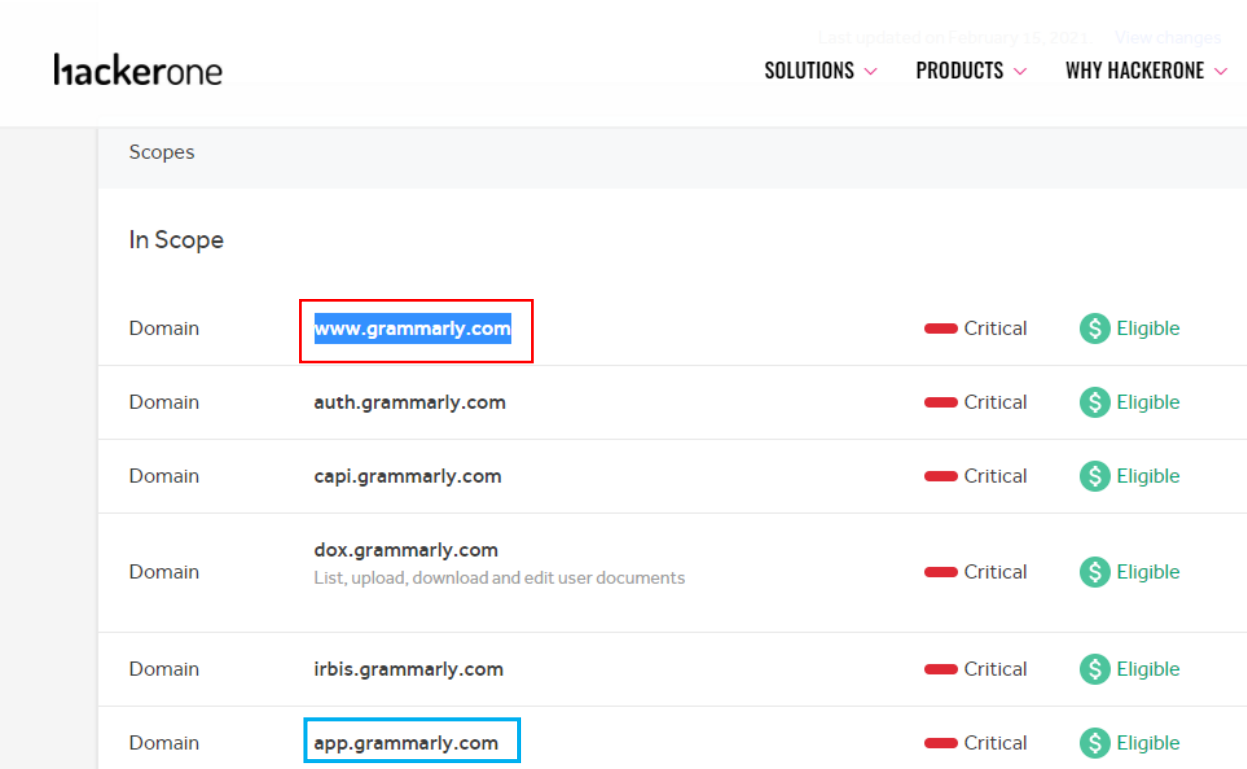
I. Policy

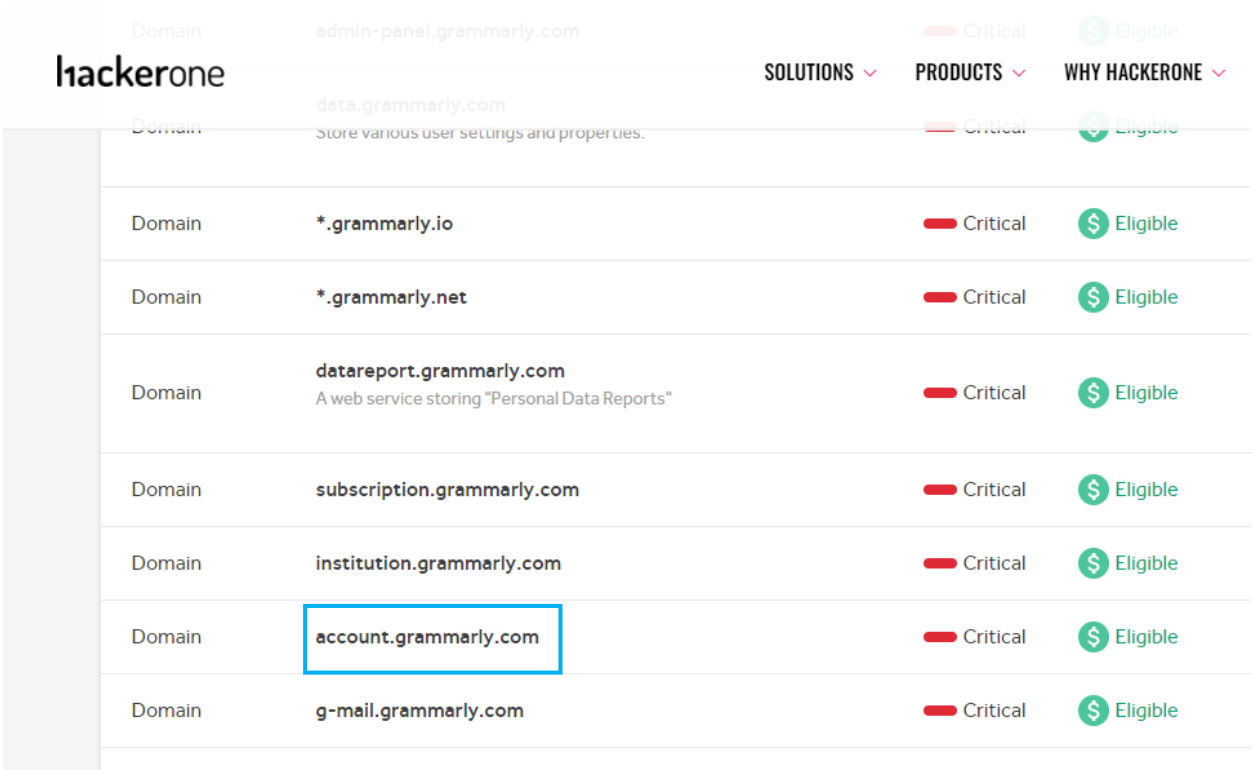
As shown in the figure, there are some policies that defines some rules and conditions which are regarding for web auditors and the organization.



II. Scope

The scope is listed with few sub domains and the main domain as shown below.




hackerone
SOLUTIONS
PRODUCTS
WHY HACKERONE
Domain
admin-panel.grammarly.com
Critical
Eligible
data.grammarly.com
Store various user settings and properties.
Domain
*.grammarly.io
Critical
Eligible
Domain
*.grammarly.net
Critical
Eligible
Domain
datareport.grammarly.com
A web service storing "Personal Data Reports"
Critical
Eligible
Domain
subscription.grammarly.com
Critical
Eligible
Domain
institution.grammarly.com
Critical
Eligible
Domain
account.grammarly.com
Critical
Eligible
Domain
g-mail.grammarly.com
Critical
Eligible

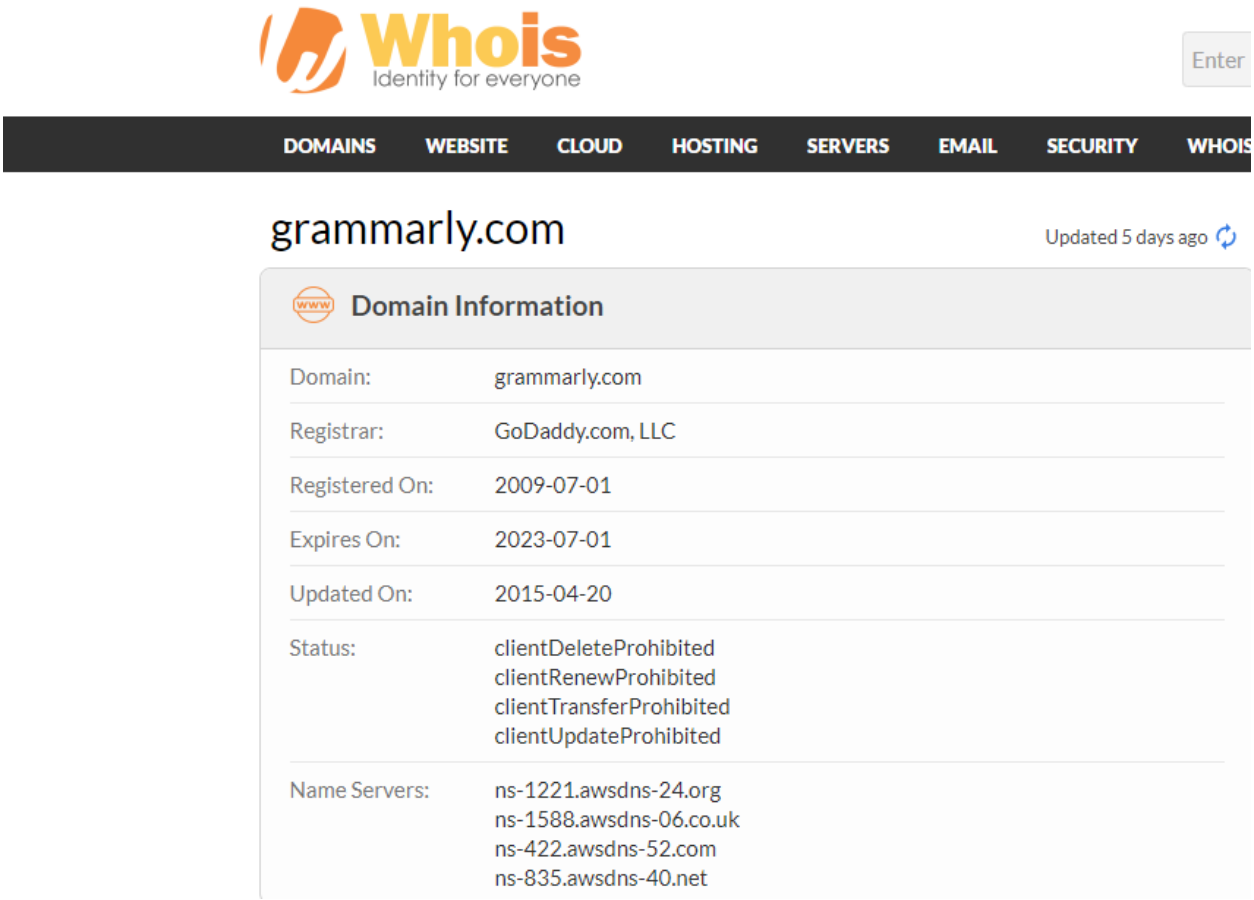
2. Reconnaissance

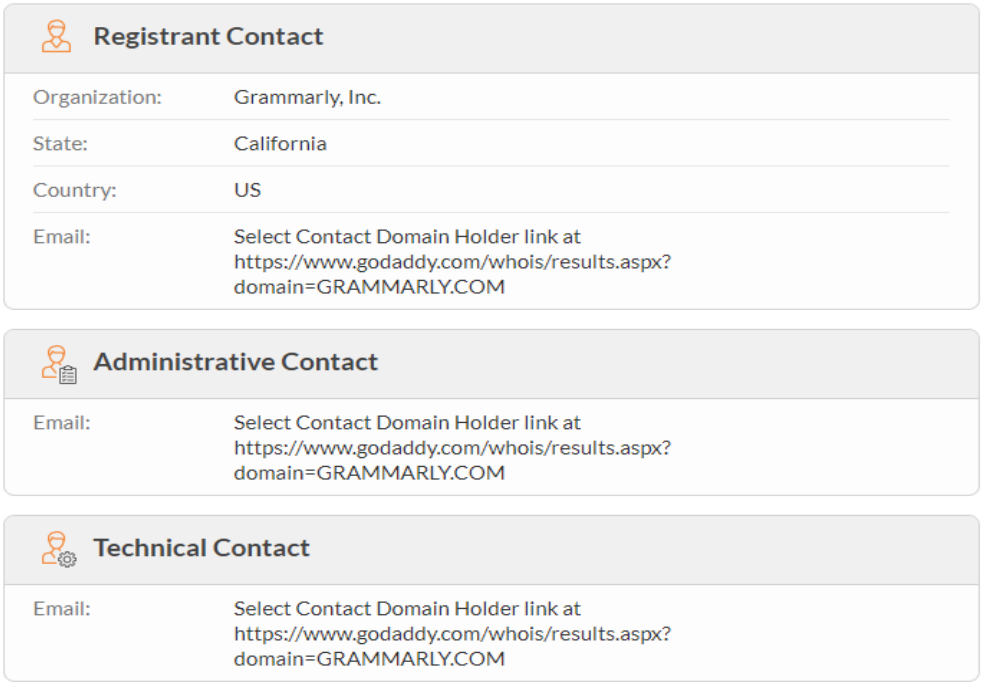
I. Target validation

Goal confirmation ensures that the target we are auditing is within reach and is the one provided by the customer. Several tools, such as Whois, nslookup, and dnsrecon, can be used to achieve this goal.

i. Whois

"Whois" is a popular Internet record listing that tells you who holds a domain and how to contact them. Domain name registration and possession was governed by the Internet Corporation for Assigned Names and Numbers (ICANN). "Whois" records have proved to be highly valuable and have evolved into an invaluable database for protecting the registration and integrity of domain names and websites.





II. Finding Subdomains

Finding subdomains for a one or more domains is known as the sub-domain enumeration which is a very essential part in reconnaissance stage. Finding vulnerabilities on web domain is a very complex task since web developers and organizations are very keen on their website. But with the help of sub-domain enumeration can reveal lot of vulnerabilities which are in the scope.

Therefore, I have selected some of sub-domain enumeration tool to find the sub-domains of the selected domain.

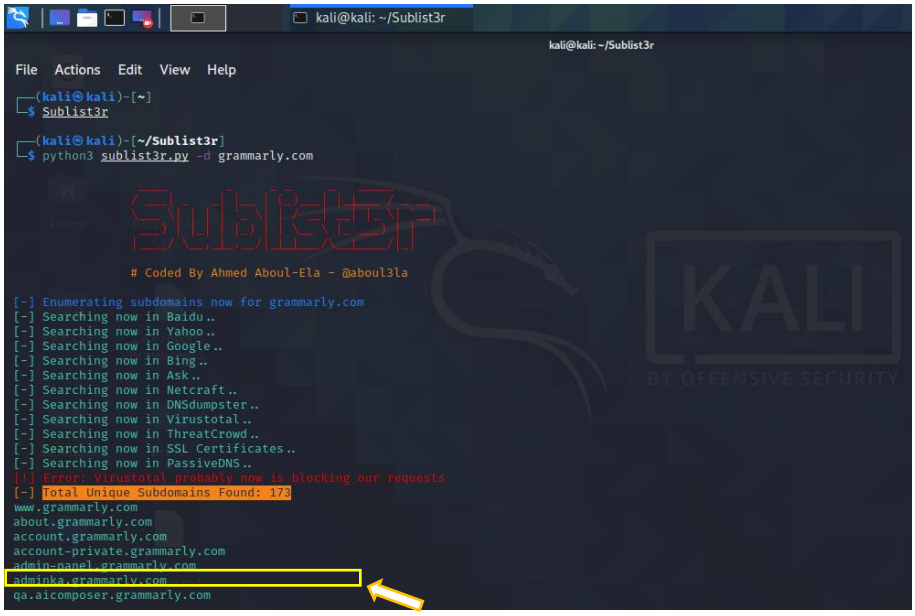
i. Sublist3r

Sublist3r is a python tool which uses OSINT to enumerate website subdomains. It assists penetration testers and bug hunters in gathering and collecting subdomains for the domain they are targeting. Google, Yahoo, Bing, Baidu, and Ask are variety of search engines that used by Sublist3r to find sub-domains.

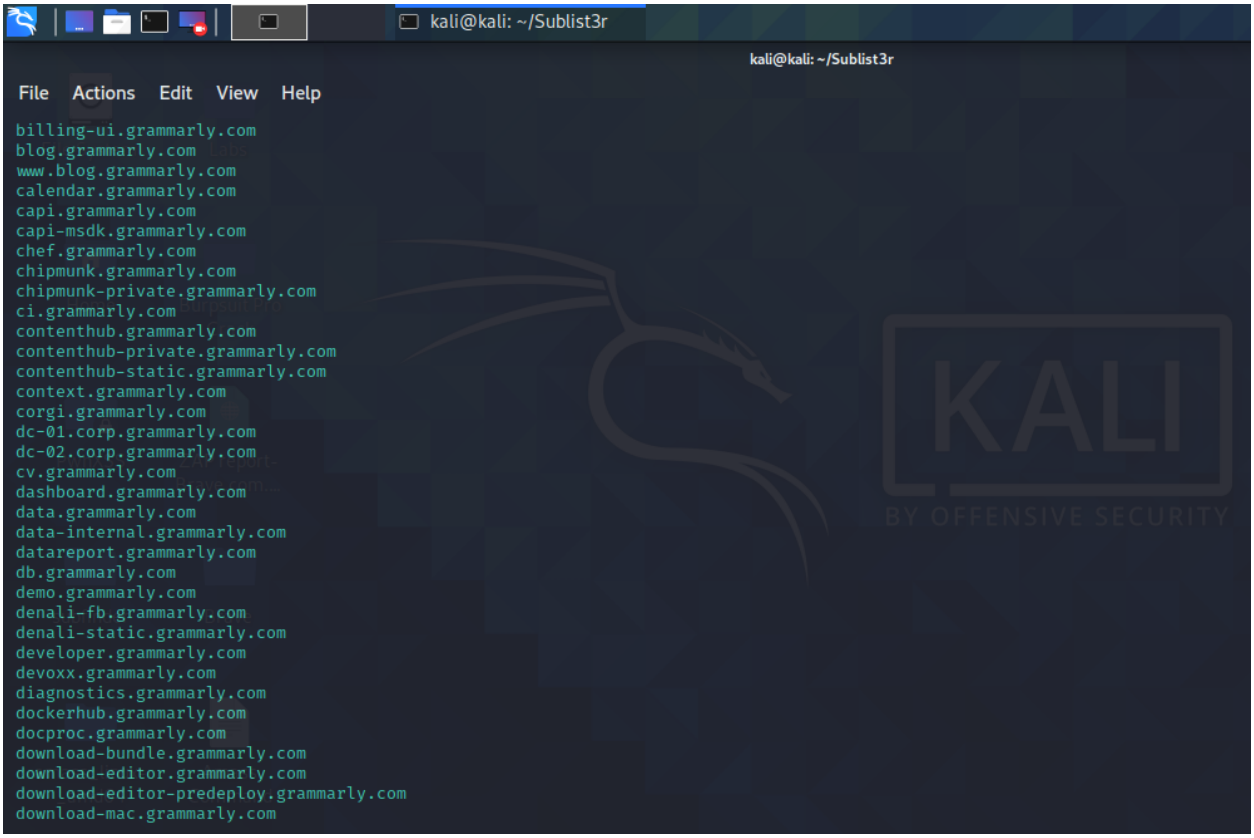
Sublist3r can install in Kali Linux and any other Linux based distributors by running "git clone https://github.com/aboul3la/Sublist3r.git" in the terminal. After

getting into the sublist3r directory we can find sub domain of the main domain by running the following command.

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



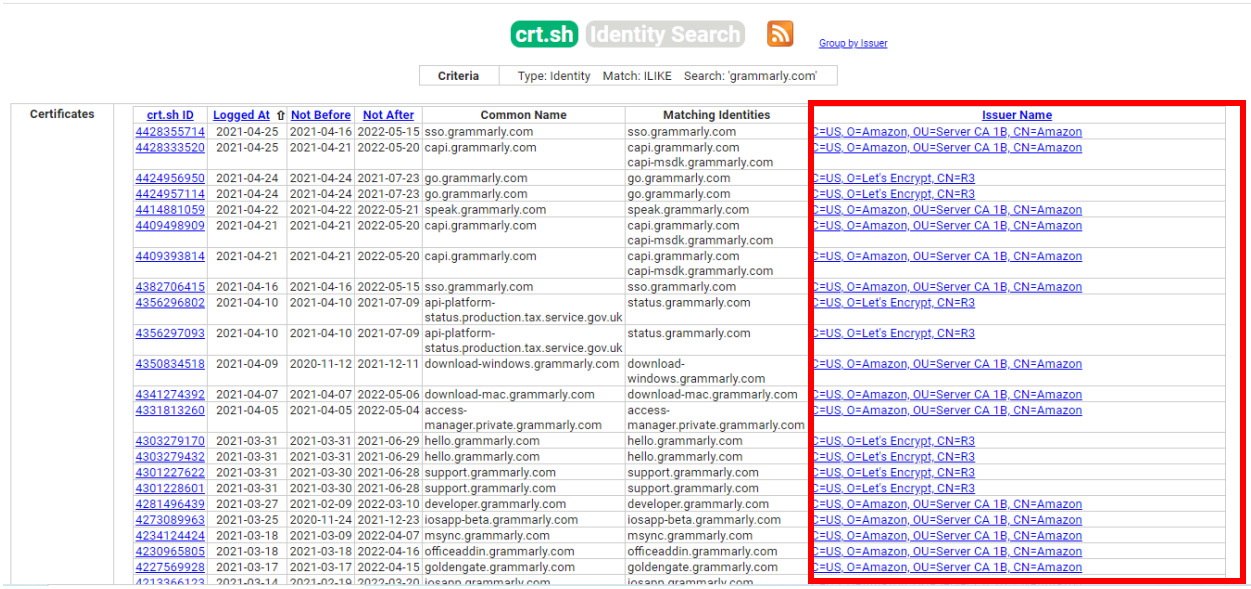
After running that command, I was able to be found **172 number of unique subdomains** in my target domain. Some of them are shown in the below figure.


kali@kali: ~/Sublist3r
kali@kali: ~/Sublist3r
File Actions Edit View Help
billing-ui.grammarly.com
blog.grammarly.com
www.blog.grammarly.com
calendar.grammarly.com
capi.grammarly.com
capi-msdk.grammarly.com
chef.grammarly.com
chipmunk.grammarly.com
chipmunk-private.grammarly.com
ci.grammarly.com
contenthub.grammarly.com
contenthub-private.grammarly.com
contenthub-static.grammarly.com
context.grammarly.com
corgi.grammarly.com
dc-01.corp.grammarly.com
dc-02.corp.grammarly.com
cv.grammarly.com
dashboard.grammarly.com
data.grammarly.com
data-internal.grammarly.com
datareport.grammarly.com
db.grammarly.com
demo.grammarly.com
denali-fb.grammarly.com
denali-static.grammarly.com
developer.grammarly.com
devoxx.grammarly.com
diagnostics.grammarly.com
dockerhub.grammarly.com
docproc.grammarly.com
download-bundle.grammarly.com
download-editor.grammarly.com
download-editor-predeploy.grammarly.com
download-mac.grammarly.com


spdmta3.send.grammarly.com
view.send.grammarly.com
sentinel.grammarly.com
sentinel-private.grammarly.com
snippets.grammarly.com
speak.grammarly.com
sso.grammarly.com
sso-institutions.grammarly.com
static.grammarly.com
stats-public.grammarly.com
status.grammarly.com
subscription.grammarly.com
support.grammarly.com
tech.grammarly.com
tokenism.grammarly.com
tokenism-private.grammarly.com
tokenism-relay.grammarly.com
tokenism-relay-private.grammarly.com
tr.grammarly.com
tracking-storage.grammarly.com
treatment.grammarly.com
trumpet.grammarly.com
ua-gec-dataset.grammarly.com
update.grammarly.com
update-officeaddin.grammarly.com
vox-static.grammarly.com
preprod.vox-static.grammarly.com
qa.vox-static.grammarly.com
vpn1.grammarly.com
wiki.grammarly.com
words.grammarly.com
wstest.grammarly.com
www2.grammarly.com
zoom-callbacks.grammarly.com
zoommarly.grammarly.com

ii. crt.sh

Simply, crt.sh can be defined as a web based tool to find subdomain, important things and also to do fingerprinting on a selected domain. There is specific column for cet.sh.ID and certificate issuer name and it will help to group the domains which are certified by a specific CA.



crt.sh Identity Search Group by Issuer

Criteria: Type: Identity Match: ILIKE Search: 'grammarly.com'

Certificates

crt.sh ID	Logged At	Not Before	Not After	Common Name	Matching Identities	Issuer Name
4428355714	2021-04-25	2021-04-16	2022-05-15	sso.grammarly.com	sso.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4428333520	2021-04-25	2021-04-21	2022-05-20	capi.grammarly.com	capi.grammarly.com capi-msdk.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4424956950	2021-04-24	2021-04-24	2021-07-23	go.grammarly.com	go.grammarly.com	C=US, O=Let's Encrypt, CN=R3
4424957114	2021-04-24	2021-04-24	2021-07-23	go.grammarly.com	go.grammarly.com	C=US, O=Let's Encrypt, CN=R3
4414881059	2021-04-22	2021-04-22	2022-05-21	speak.grammarly.com	speak.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4409498909	2021-04-21	2021-04-21	2022-05-20	capi.grammarly.com	capi.grammarly.com capi-msdk.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4409393814	2021-04-21	2021-04-21	2022-05-20	capi.grammarly.com	capi.grammarly.com capi-msdk.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4382706415	2021-04-16	2021-04-16	2022-05-15	sso.grammarly.com	sso.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4356296802	2021-04-10	2021-04-10	2021-07-09	api-platform-status.production.tax.service.gov.uk	status.grammarly.com	C=US, O=Let's Encrypt, CN=R3
4356297093	2021-04-10	2021-04-10	2021-07-09	api-platform-status.production.tax.service.gov.uk	status.grammarly.com	C=US, O=Let's Encrypt, CN=R3
4350834518	2021-04-09	2020-11-12	2021-12-11	download-windows.grammarly.com	download-windows.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4341274392	2021-04-07	2021-04-07	2022-05-06	download-mac.grammarly.com	download-mac.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4331813260	2021-04-05	2021-04-05	2022-05-04	access-manager.private.grammarly.com	access-manager.private.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4303279170	2021-03-31	2021-03-31	2021-06-29	hello.grammarly.com	hello.grammarly.com	C=US, O=Let's Encrypt, CN=R3
4303279432	2021-03-31	2021-03-31	2021-06-29	hello.grammarly.com	hello.grammarly.com	C=US, O=Let's Encrypt, CN=R3
4301227622	2021-03-31	2021-03-30	2021-06-28	support.grammarly.com	support.grammarly.com	C=US, O=Let's Encrypt, CN=R3
4301228601	2021-03-31	2021-03-30	2021-06-28	support.grammarly.com	support.grammarly.com	C=US, O=Let's Encrypt, CN=R3
4281496439	2021-03-27	2021-02-09	2022-03-10	developer.grammarly.com	developer.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4273089963	2021-03-25	2020-11-24	2021-12-23	iosapp-beta.grammarly.com	iosapp-beta.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4234124424	2021-03-18	2021-03-09	2022-04-07	msync.grammarly.com	msync.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4230965805	2021-03-18	2021-03-18	2022-04-16	officeaddin.grammarly.com	officeaddin.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4227569928	2021-03-17	2021-03-17	2022-04-15	goldengate.grammarly.com	goldengate.grammarly.com	C=US, O=Amazon, OU=Server CA 1B, CN=Amazon
4213366123	2021-03-14	2021-02-19	2022-03-20	iosapp.grammarly.com	iosapp.grammarly.com	

Above figure shows the results of my target domain using crt.sh.

iii. Subfinder

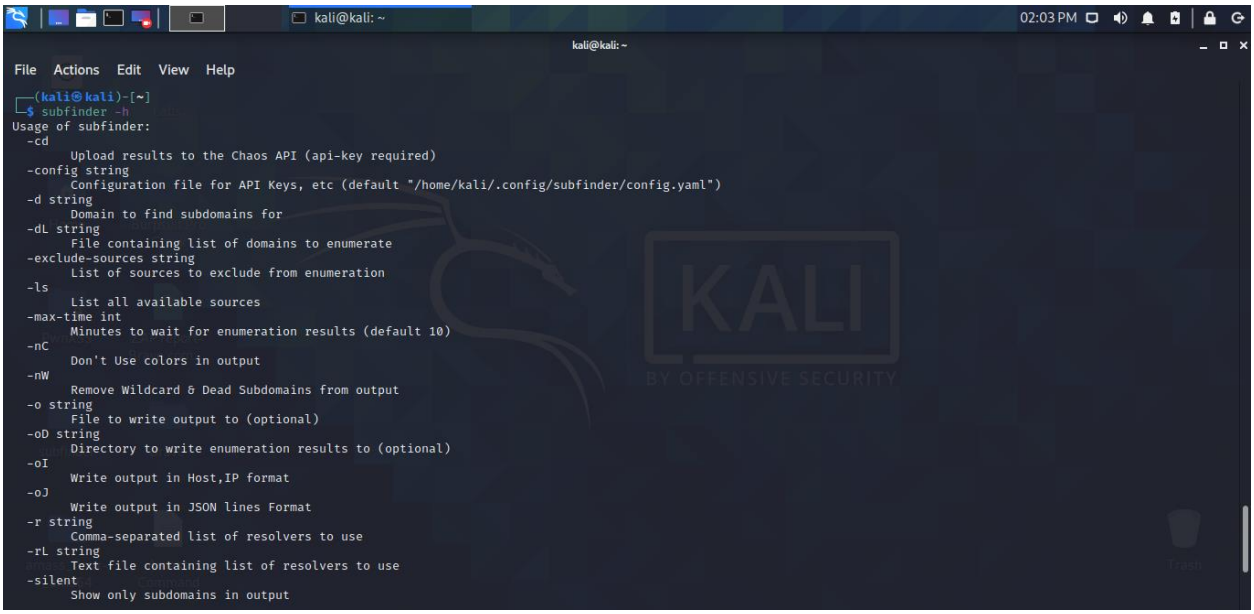
Subfinder is a subdomain exploration platform that uses passive online sources to locate relevant subdomains for websites. It has a straightforward modular design that is designed for speed. Subfinder was designed to do one thing and one thing well: passive subdomain enumeration. It is designed as a subfinder to adhere to all passive source permits and usage limitations while retaining a transparent passive model, making it useful for both penetration testers and bug bounty hunters.

There are different ways for the subfinder installation. Executing the following commands, we can simply install it from GitHub.

```
git clone https://github.com/projectdiscovery/subfinder.git
cd subfinder/v2/cmd/subfinder
go build .
mv subfinder /usr/local/bin/
```

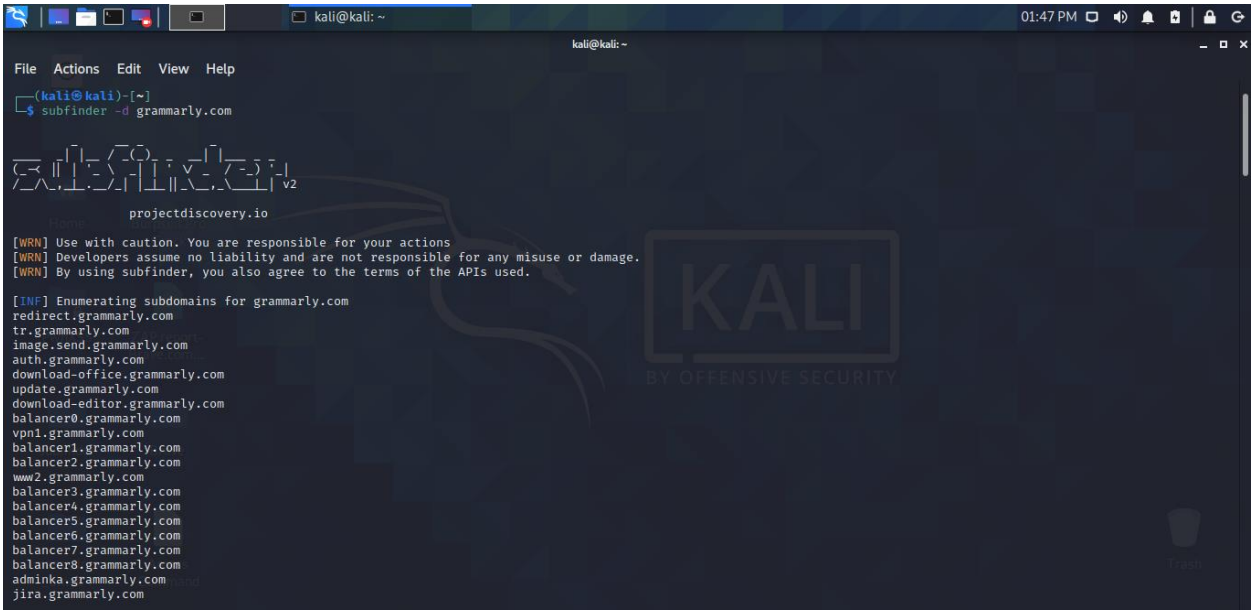
Using the following command, we can see the help for the tool, and it will clearly define what it does for each and every flag.

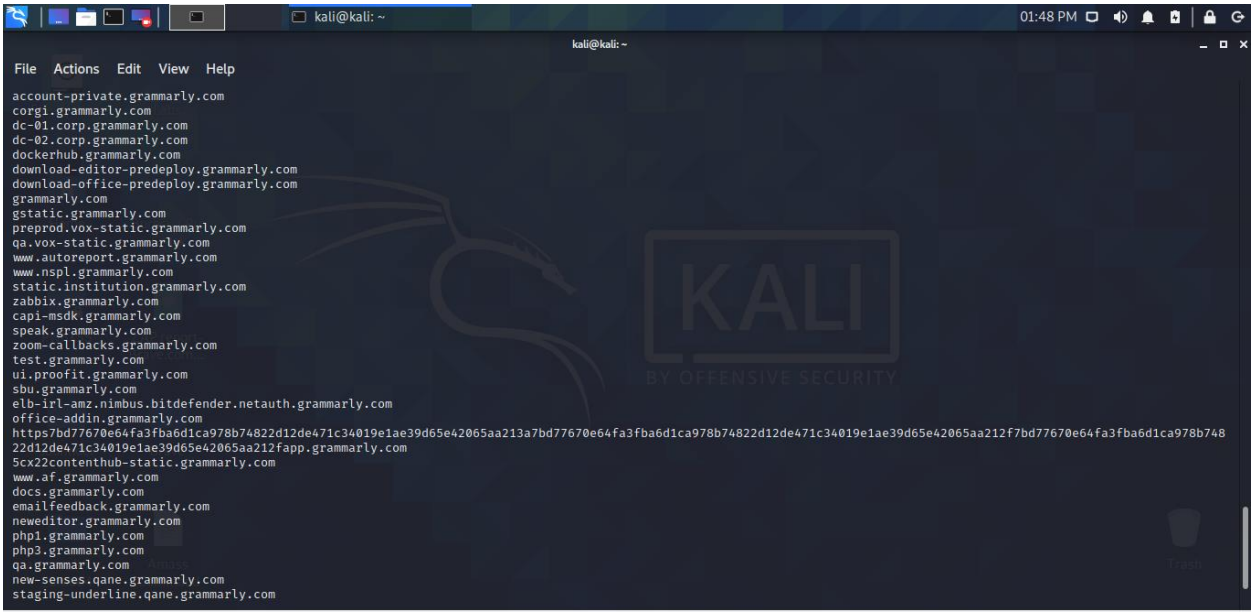
subfinder -h



As we already done in Sublitr3r we can find sub domain of the main domain by running the following command.

```
subfinder -d grammarly.com
```





As shown in the above figure, after running that command I was able to find several numbers of unique subdomains in my target domain.

iv. Amass

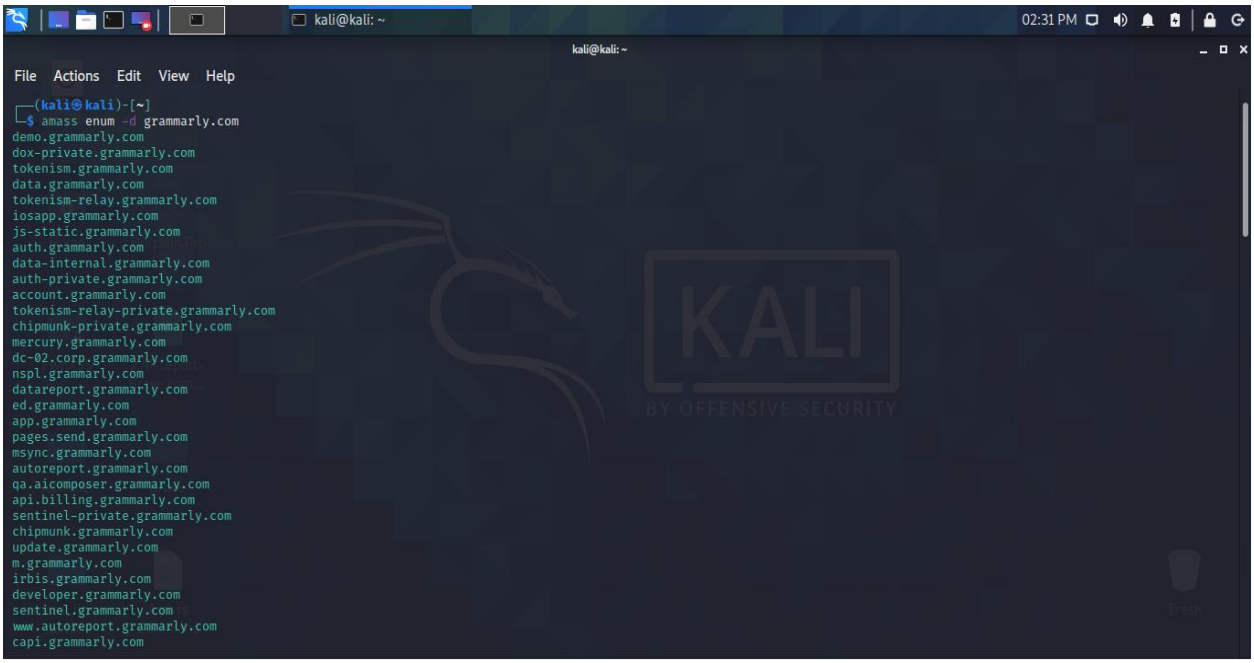
Amass is a great tool for gathering information on the attack surface of targets in multiple dimensions and it is based on OWASP. This tool really helps for penetration testers by mapping the organization's attack surface and with the use of open source information gathering and active reconnaissance techniques it allows to perform external asset discovery.

We can simply install it from GitHub using git clone command as shown below.

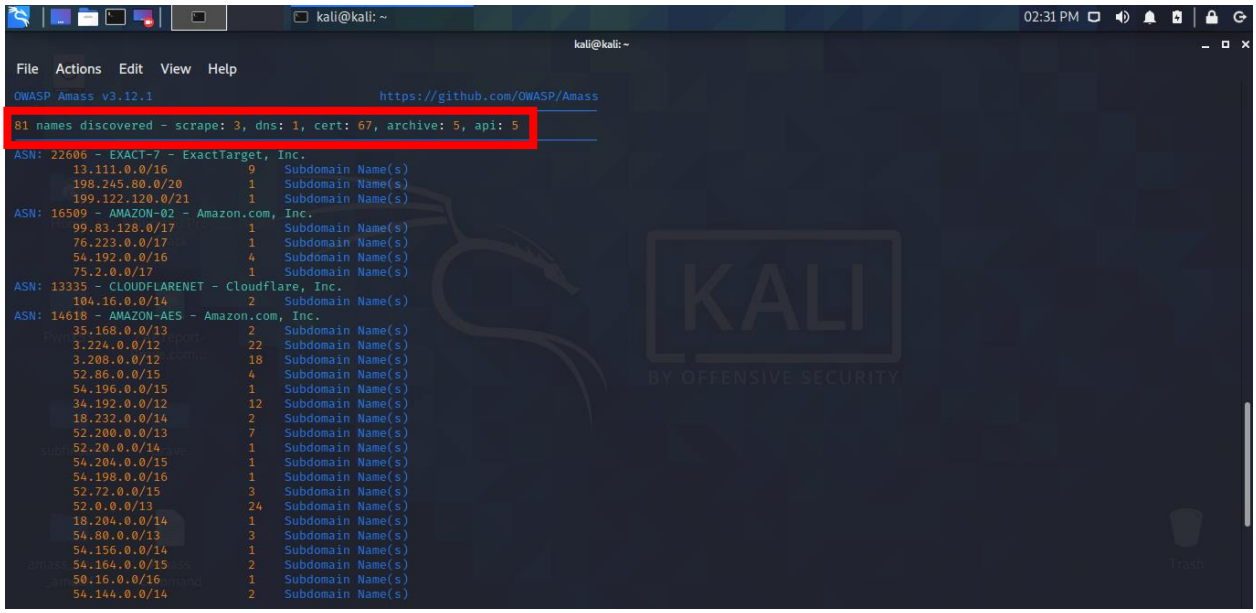
git clone https://github.com/OWASP/Amass.git

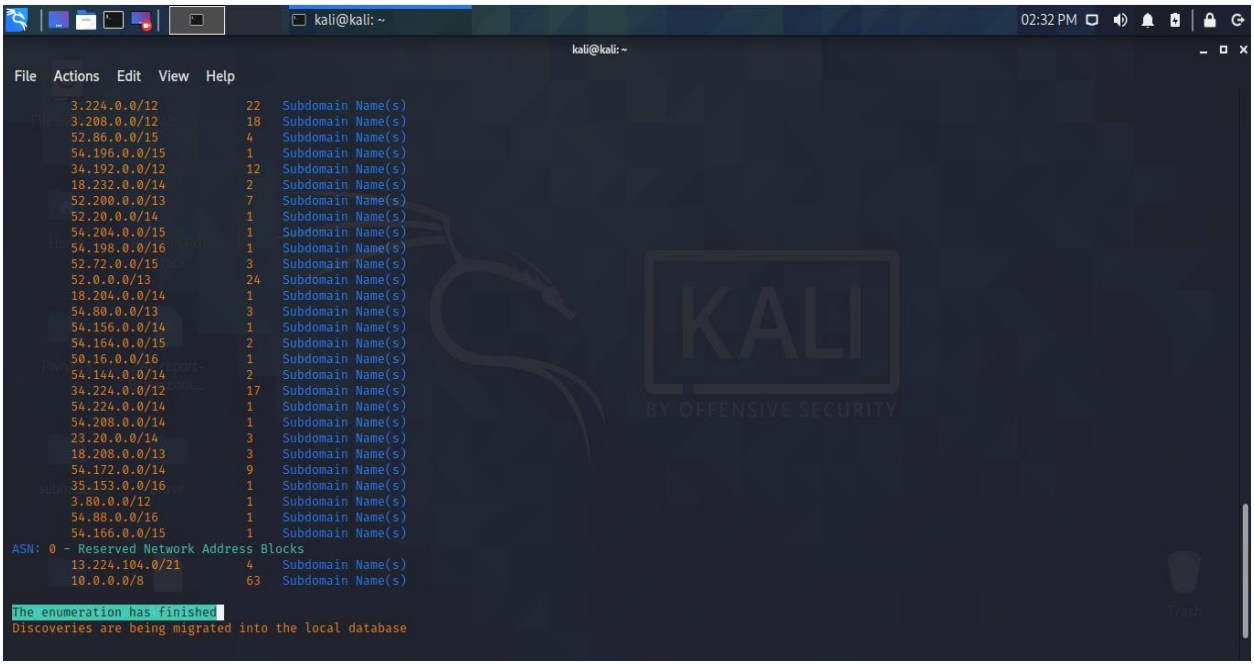
Here I used Amass tool in order to gather more and relevant information regarding my sub-domains of my target domain of grammarly.com.

```
amass enum -d grammarly.com
```



As shown in below, at the end of subdomain scan, it generates a report which contains all web server details, number of names discovered, IP address ranges of subdomains and their certificate issuers. Even it takes more time than the other sub domain scanners, at the end of it provide this categorized report and it is really helping to get a clear understand of the target domain and its sub domains.



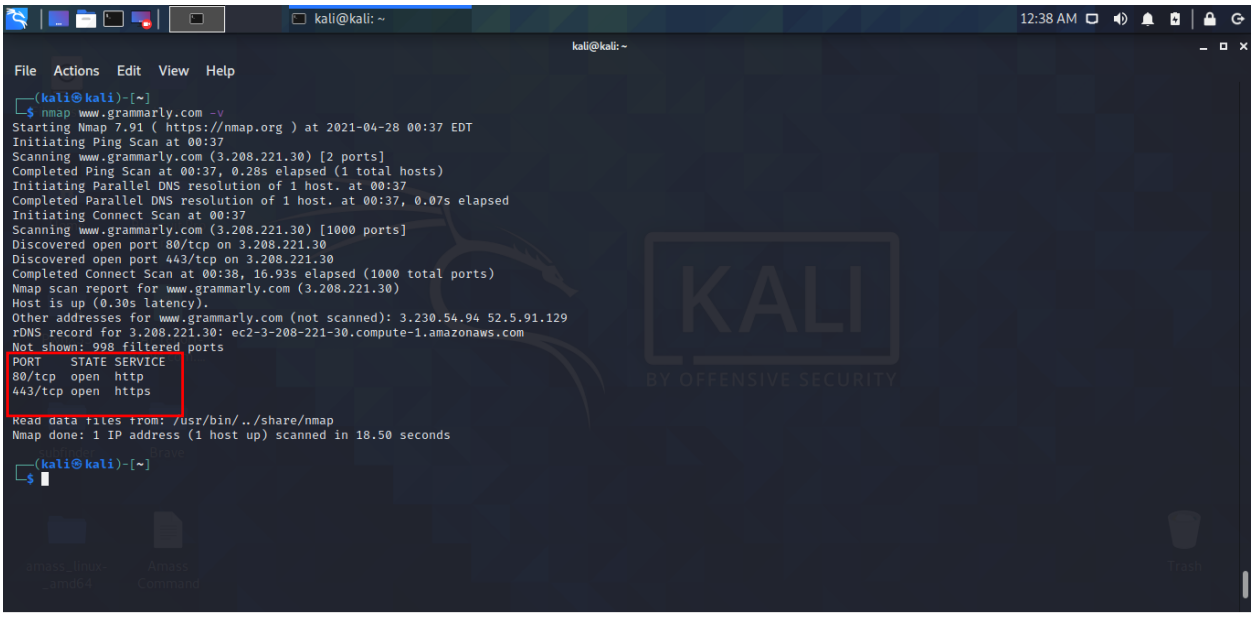


3. Discovery and Scanning

I. Nmap

Nmap (Network Mapper) is a network transparency and security examination tool that is free and open source (permit). Nmap makes use of raw IP packets in creative ways to find out what hosts are available on the network, what administrations (application name and form) those hosts are providing, what working systems (and OS renditions) they are using, what kind of parcel channels/firewalls they are using, and a variety of other things. It was designed to easily check large entities, but it still works well for single hosts. The Nmap suite includes a severe GUI and results watcher (Zenmap), an adaptable information move, redirection, and investigation equipment (Ncat), a utility for looking at search results (Ndiff), and a package age and reaction analysis unit, aside from the well-known order-line Nmap executable (Nping).

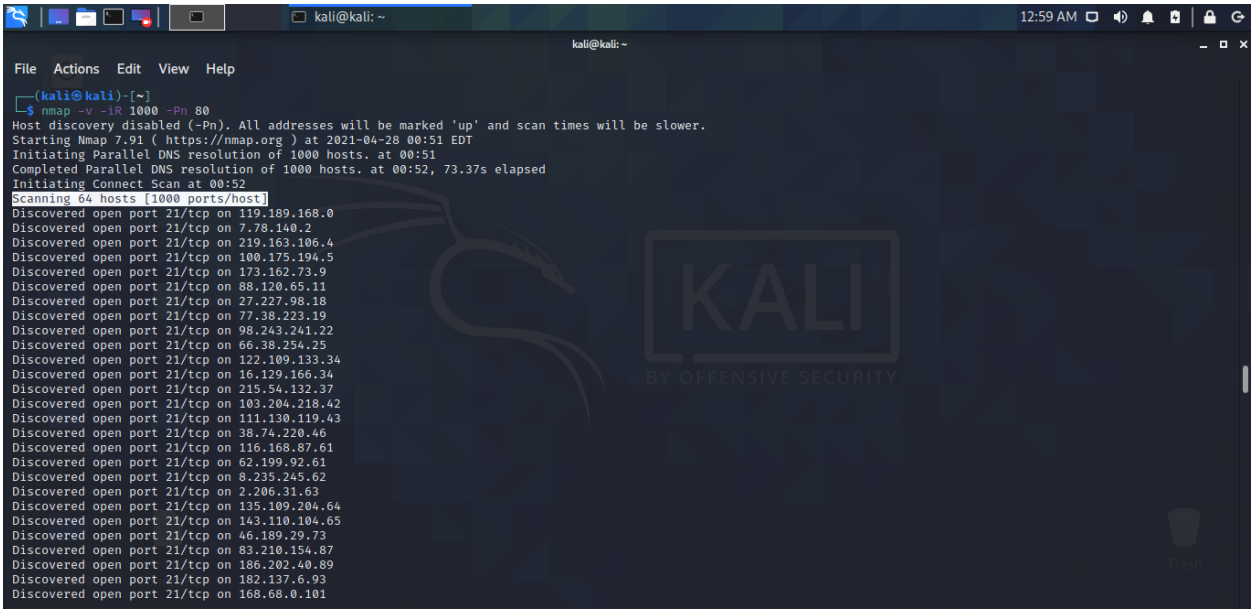
First, by running the command of nmap www.grammarly.com –v we were able to find two open ports which is common but informative.

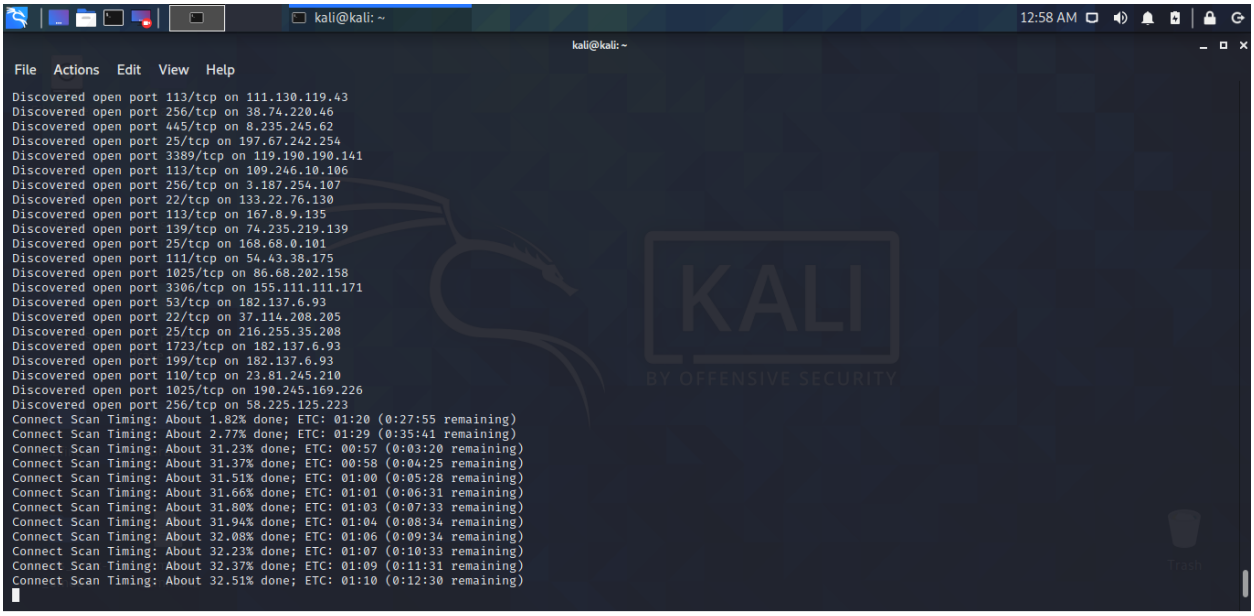


Using above shown scan we can find the IP address of our target domain as **3.208.221.30** and it can be used for further scans regarding my target domain. Furthermore, we can see there are two open ports which are currently open with specific service.

PORT	STATE	SERVICE
80/tcp	open	http
443/tcp	open	https

When we scanned port 80 specifically using the command nmap -v -iR 1000 -Pn 80 as shown in bellow it takes more time than the others and it gives more information about discovered open port on tcp with the specific IP address individually.

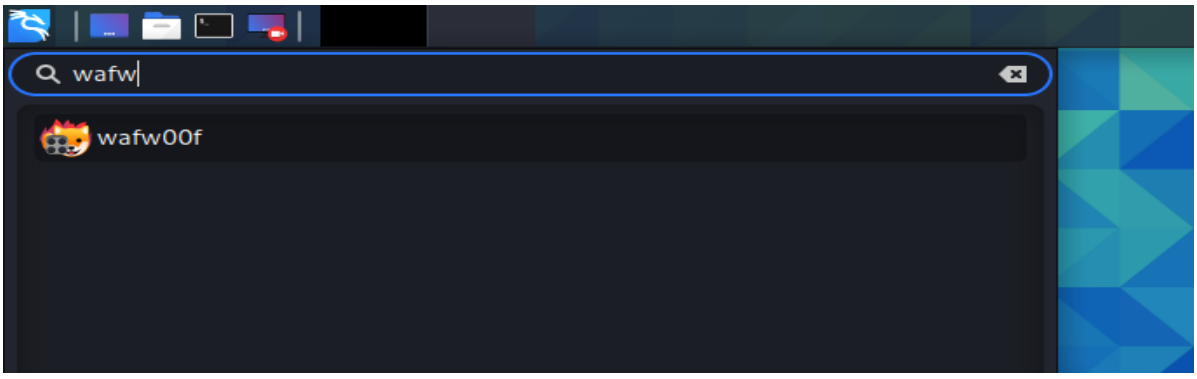




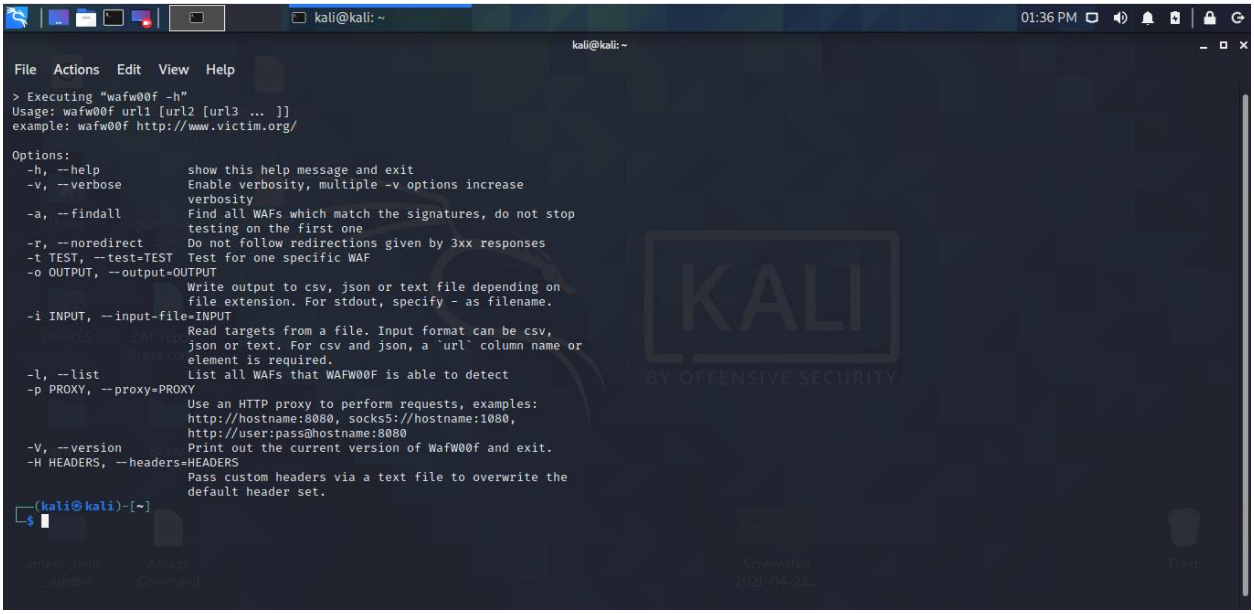
II. Wafw00f

Wafw00f is a well-known Python application that automates the process of fingerprinting a website's firewall. Wafw00f will evaluate the underlying firewall used by a provider it probes based on the answers to a set of precisely designed web requests. Wafw00f is pre-installed in Kali Linux, but it can also be installed on any Python-enabled device. Wafw00f consistently provided more complete and reliable results during testing.

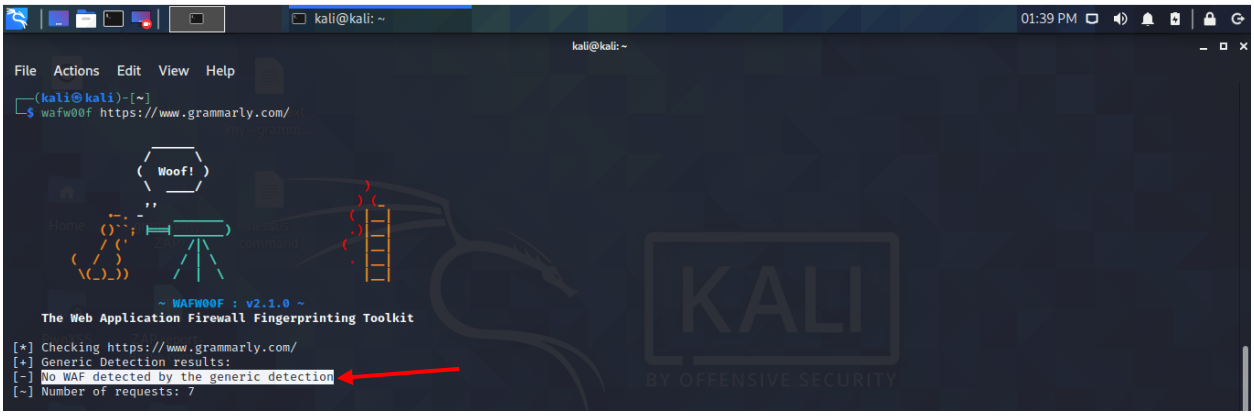
Since Wafw00f is used in the Kali Linux Penetration Testing OS by default, it can be accessed by simply searching and selecting the Wafw00f in the search bar in Kali Linux as shown below.



Then it will execute the command of "wafw00f -h" by default and it will display the available options as shown below.



In order to scan our target, I have executed the command with the target URL.
wafw00f https://www.grammarly.com/



Scan was completed by displaying that the target domain has not a Web Application Firewall (WAF). This can be considered as a critical vulnerability because we can reduce intruder attacks by 20% with the help of WAF.

i. Solution:

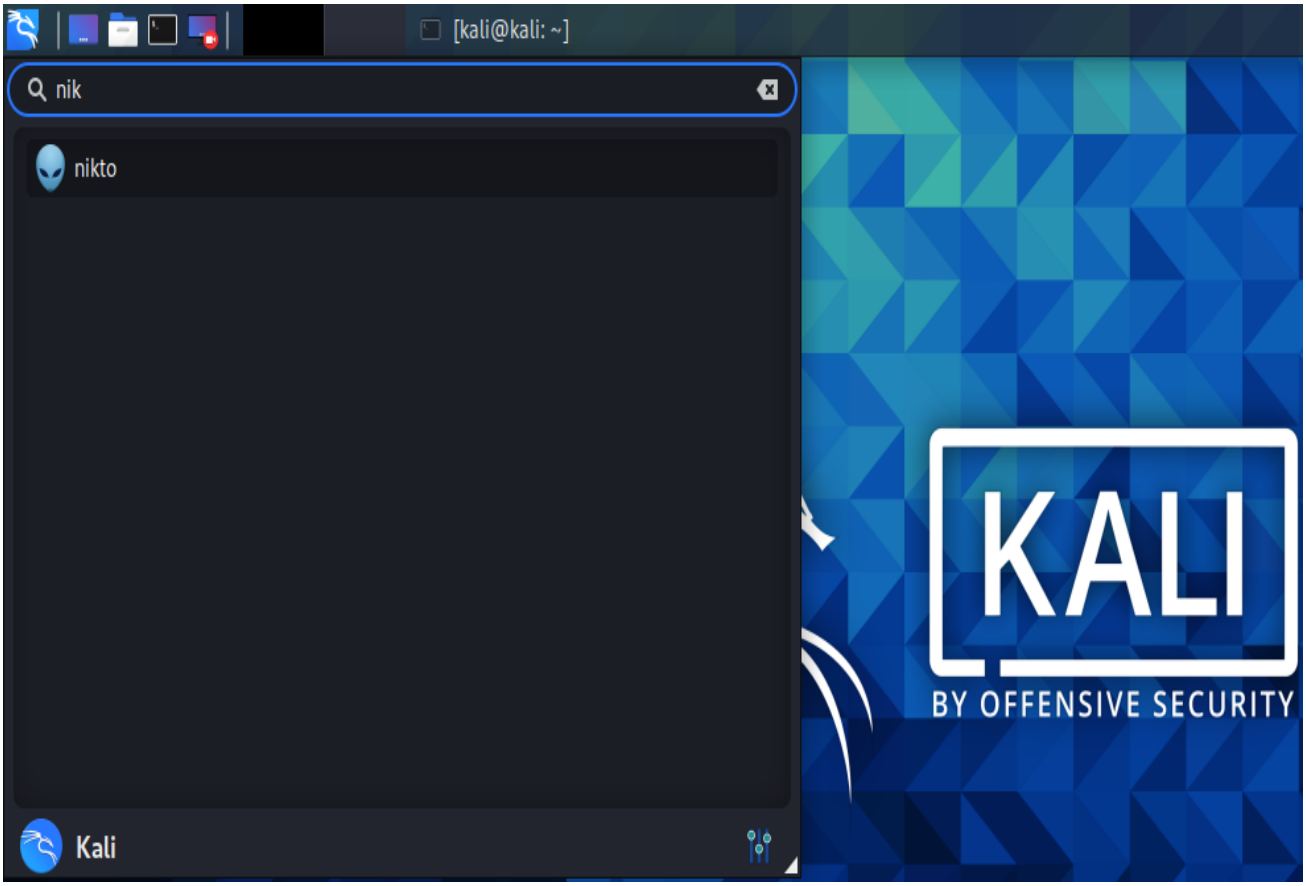
Through filtering and tracking HTTP traffic between a web application and the Internet, a WAF, or Web Application Firewall, aids in the protection of web apps. It usually defends web applications from threats like cross-site forgery, cross-site scripting (XSS), file inclusion, and SQL injection. A WAF (in the OSI model) is a protocol layer 7 security that is not intended to protect against all forms of attacks. This type of attack prevention is typically part of a larger set of techniques that work together to provide a comprehensive response against a variety of threats. A WAF acts as a barrier between a web application and the Internet when it is deployed in front of it.

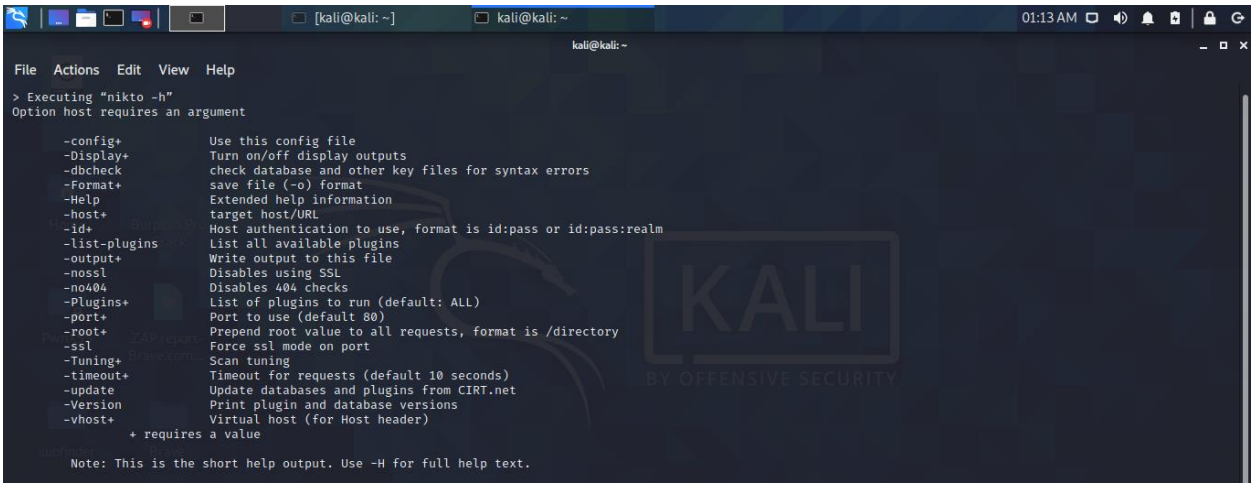
4. Vulnerability Assessment

I. Nikto

Nikto is an Open Source (GPL) web worker scanner that runs comprehensive tests against web workers for a variety of items, including over 6700 potentially dangerous documents/programs, scans for outdated adaptations on over 1250 servers, and type explicit problems on over 270 servers. It also looks for server architecture elements like the inclusion of multiple record documentation and HTTP server options, as well as attempting to understand installed web servers and programming. Nikto can detect a wide variety of problems as well as look at configuration issues. This tool is also pre-installed in Kali.

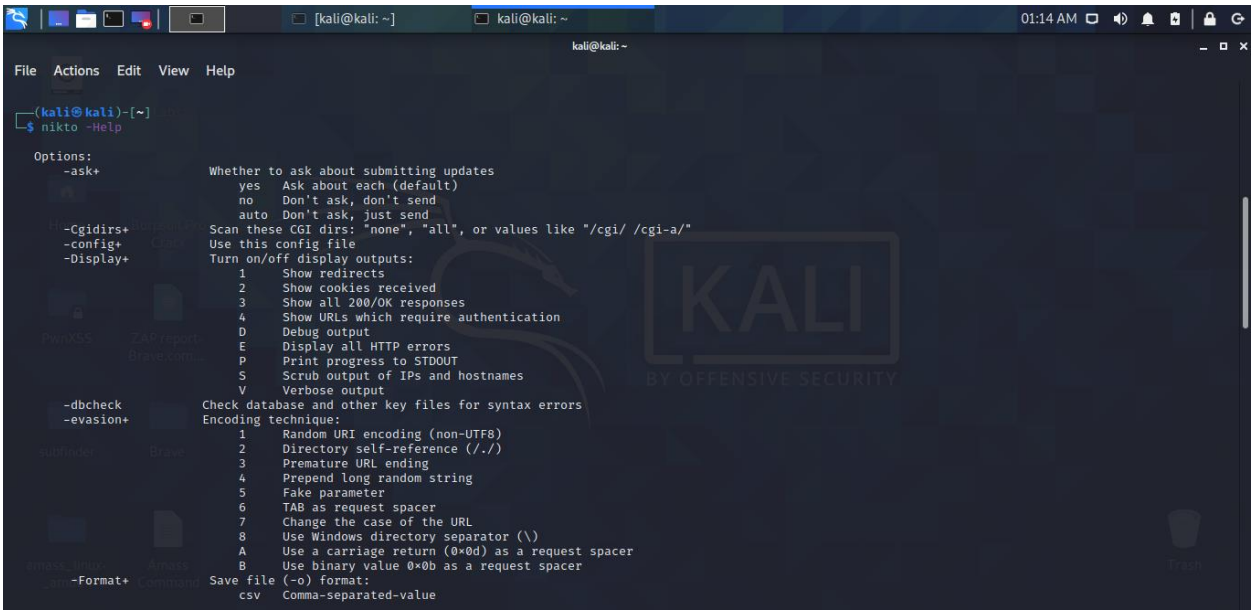
We can run it by simply searching in Kali environment as shown below.

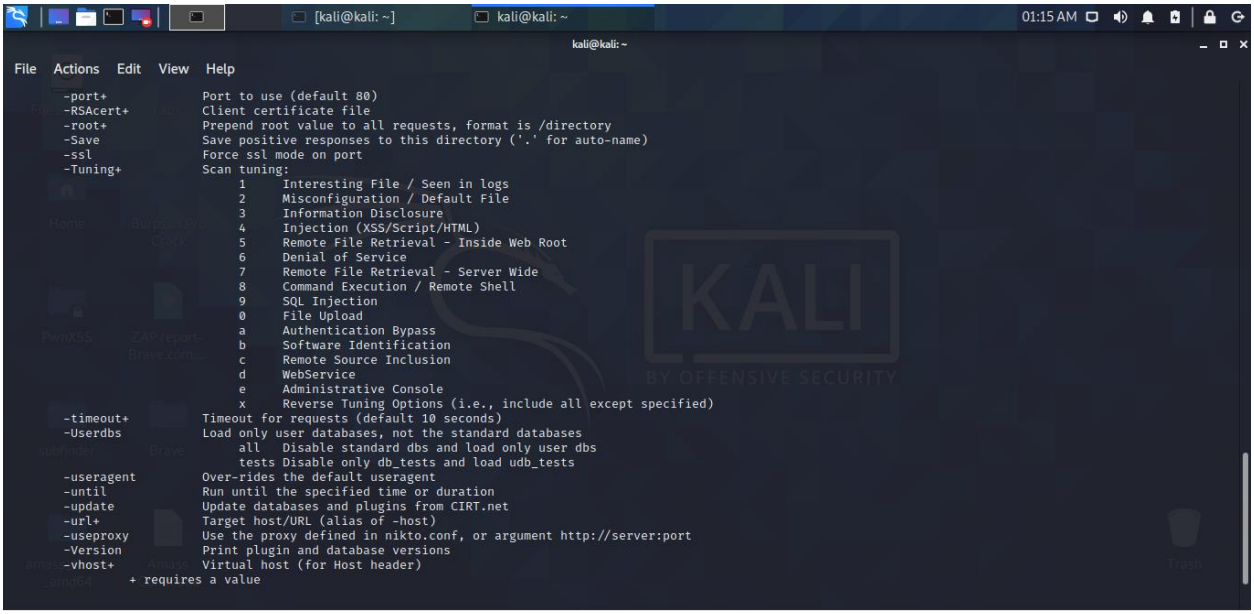




We can find all the options which are available in Nikto tool by running "nikto -Help" command in open window as shown below.

nikto -Help





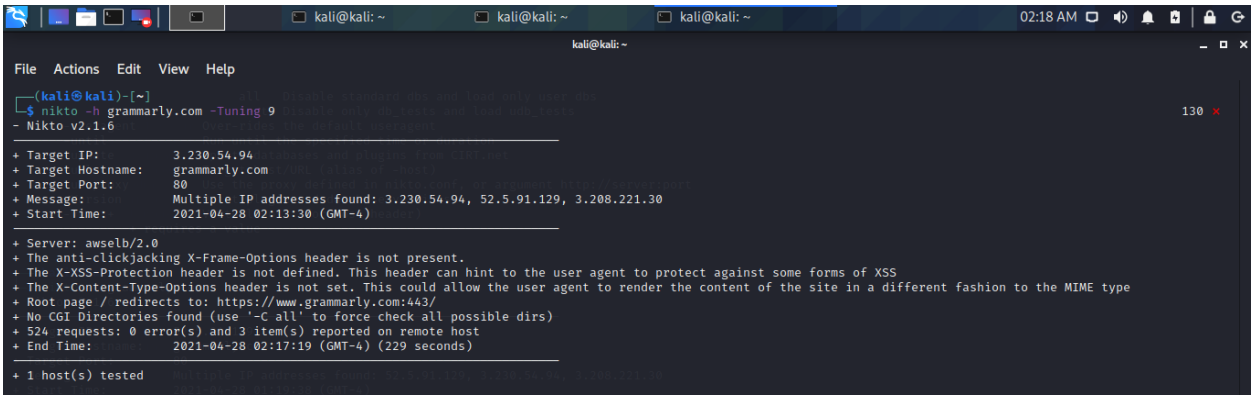
Using the "nikto -h grammarly.com" we can find that the X-XSS-Protection header is not defined, and server name can be found as shown below.

In order to complete this scan, it takes more than 40 minutes, but it cannot find any error requests within all 7786 requests.

Then I scanned my target domain with the number of 1 option for the Interesting File / Seen in logs command as shown below, and it does not give any vulnerability and end up with 0 errors within all of 2139 total requests.

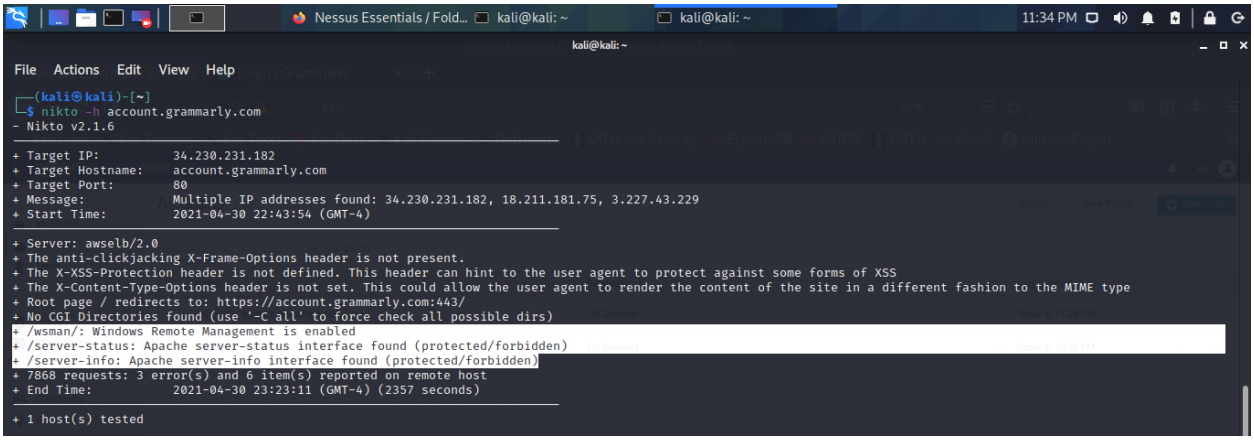
Then I scanned my target domain with the number of 4 option for the Injection (XSS/Script/HTML) command as shown below, and it does not give any vulnerability and end up with 0 errors.

Next, I scanned my target domain with the number of 9 option for the SQL Injection command as shown below, and it does not give any vulnerability and end up with 0 errors as previous.



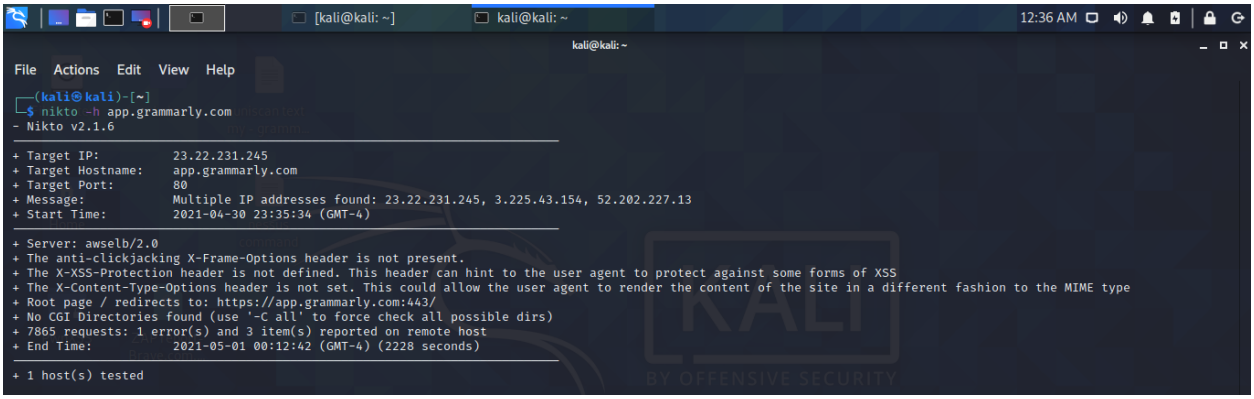
Furthermore, I have scanned my sub-domains in order to find IP addresses and errors using Nikto tool as shown below. Those sub-domains were already fallen under the scope of bug-bounty rules and regulations.

As the first sub-domain I have scanned account.grammarly.com using the command of "nikto -h account.grammarly.com" as shown below.



As the above figure shows, I found 3 errors within 7868 requests and those errors can be only consider as the info vulnerabilities since all of them were protected and forbidden under the states of the vulnerability.

As the second sub-domain I have scanned app.grammarly.com using the command of "nikto -h app.grammarly.com" as shown below.



That scan also finished as the previous sub-domain scan with one error within all 7865 requests and it did not give any vulnerabilities apart from the previous scan results.

All the scan I have done using Nikto, we can find three vulnerabilities listed as **“** The **X-Content-Type-Options** header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type**”** and **“**The **X-XSS-Protection** header is not defined. This header can hint to the user agent to protect against some forms of XSS**”** and **“**Anti-clickjacking **X-Frame-options** header is not present**”**

i. X-Content-Type-Options

In order to reduces exposure to drive-by-download attacks and sites serving uploaded content they (organization) should set this header correctly.

a. Solution

Set the X-Content-Type-Options: nosniff header

ii. X-XSS-Protection

Entering data to a web application across an unstructured source will end up with Cross-Site-Scripting (XSS). This data which included in a dynamic content is sent to a web client apart from any filtering or validation of malicious code

a. Solution

Set the X-XSS-Protection "1; mode=block"

iii. X-Frame-Options

This X-Frame-Options header can be used to indicate whether or not a browser should be allowed to render a page in a <frame>, <iframe>, <embed> or <object>. This will be used by websites to prevent click-jacking attacks by ensuring that their content is not hidden in other websites.

a. Solution

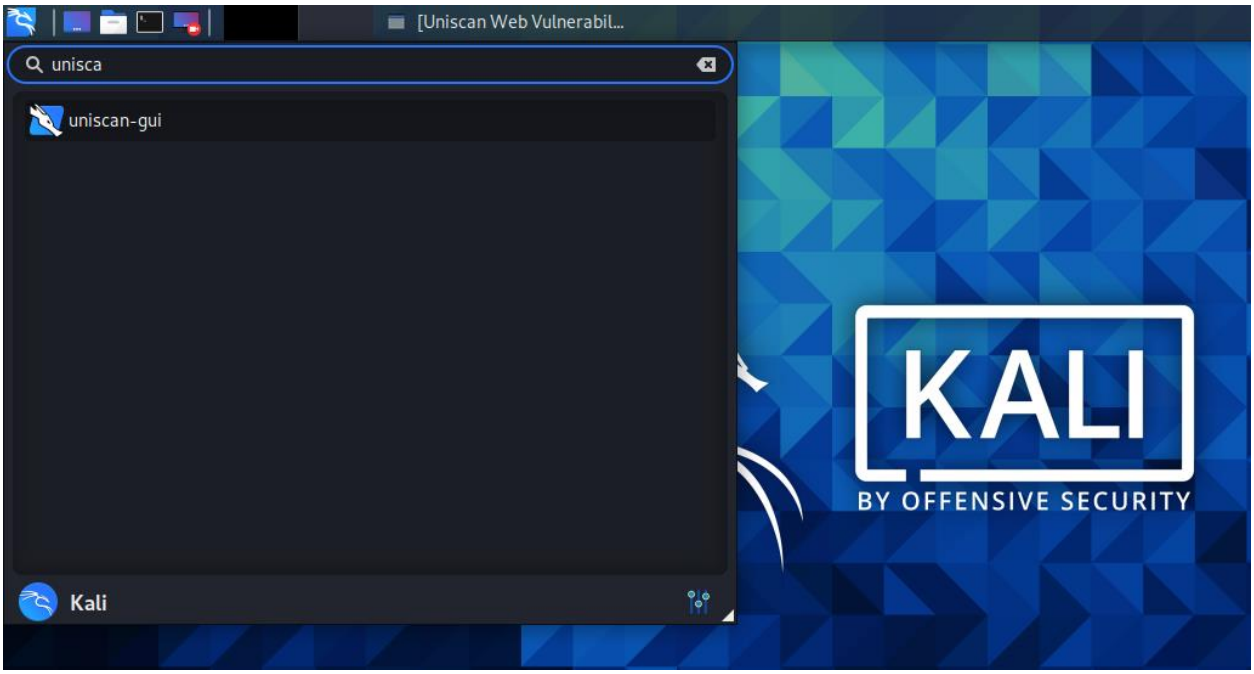
Set the X-Frame-Options: deny

II. Uniscan

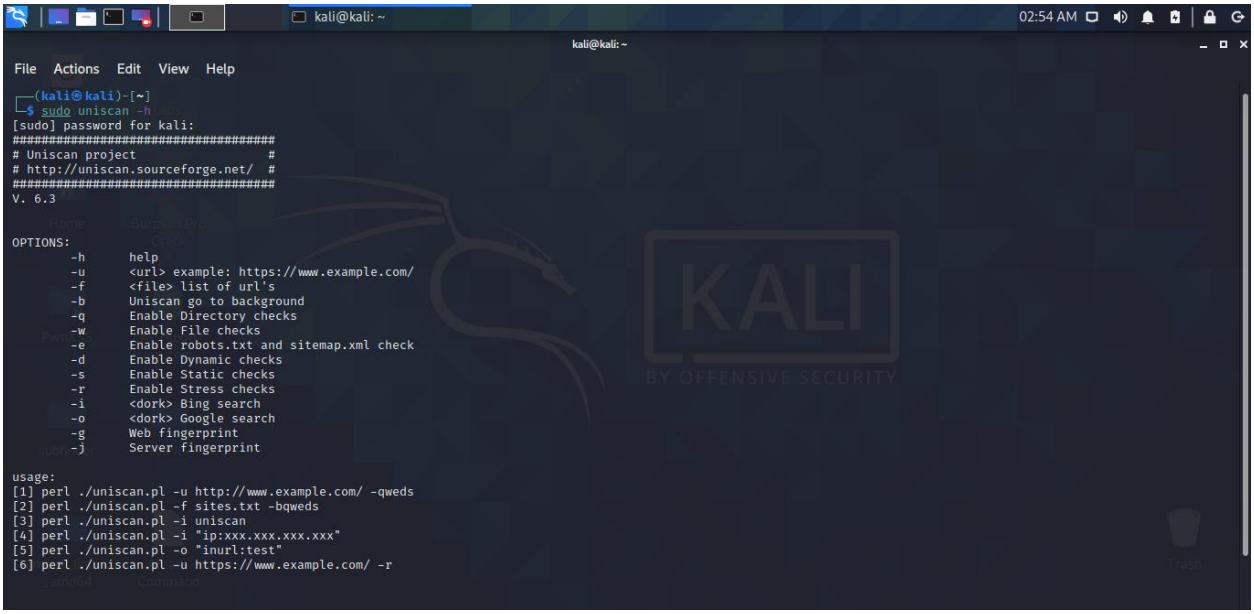
Uniscan is a simple web vulnerability scanner that searches for common flaws like neighboring document include, far off order execution, and far off record include flaws. It can also be used to label and list web administrations, interesting records and registries, and employee information. This instrument is written in Perl and can be used as a natural order line computer or as a graphical user interface. When the GUI and the command line are compared, the command line is quicker.

There is no need to install anything if you are using Kali Linux as your pen testing station because Uniscan is well-known for it.

We can run it by simply searching in Kali environment as shown below.

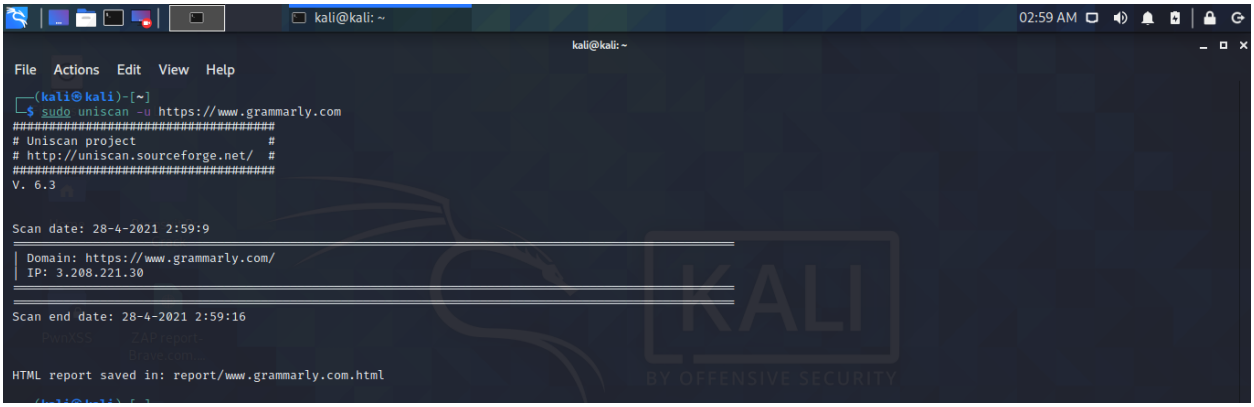


Using “uniscan -h” command we can show the options available with Uniscan tool.



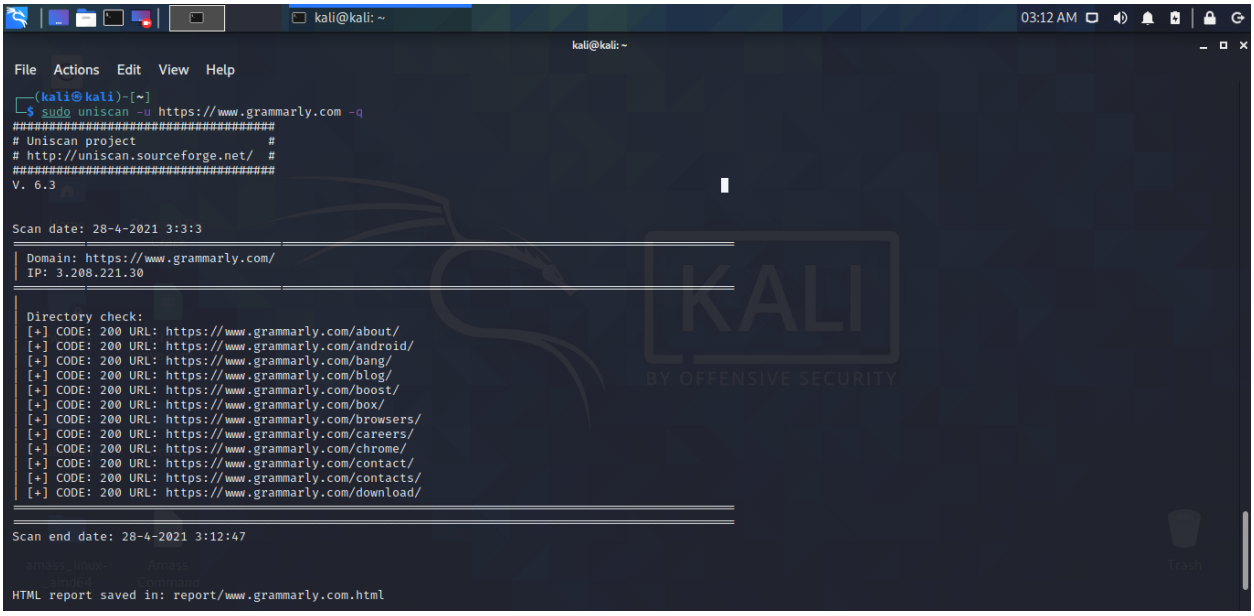
In order to scan the main domain, we need to set the "-u" flag followed by the target domain as shown below and it will end up with giving some brief information about the server and the IP address of that target domain.

sudo uniscan -u https://www.grammarly.com



By using the "-q" flag, we can search for directories on the target. It appears that it discovered some directories including what appears to be documentation and PHP configuration data.

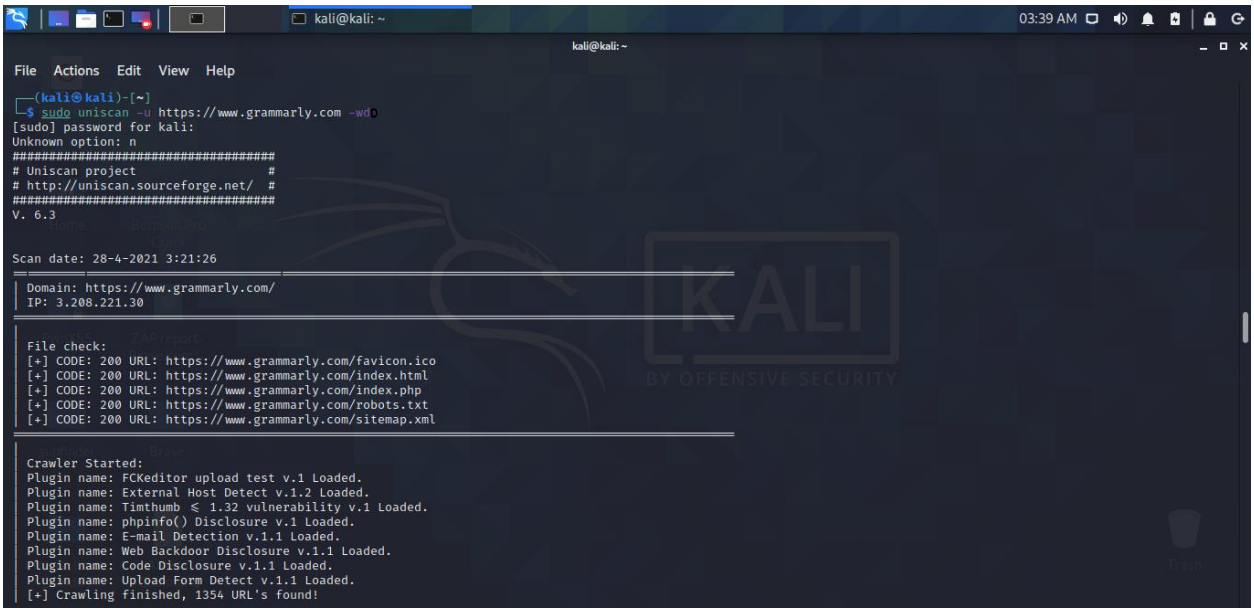
sudo uniscan -u https://www.grammarly.com -q

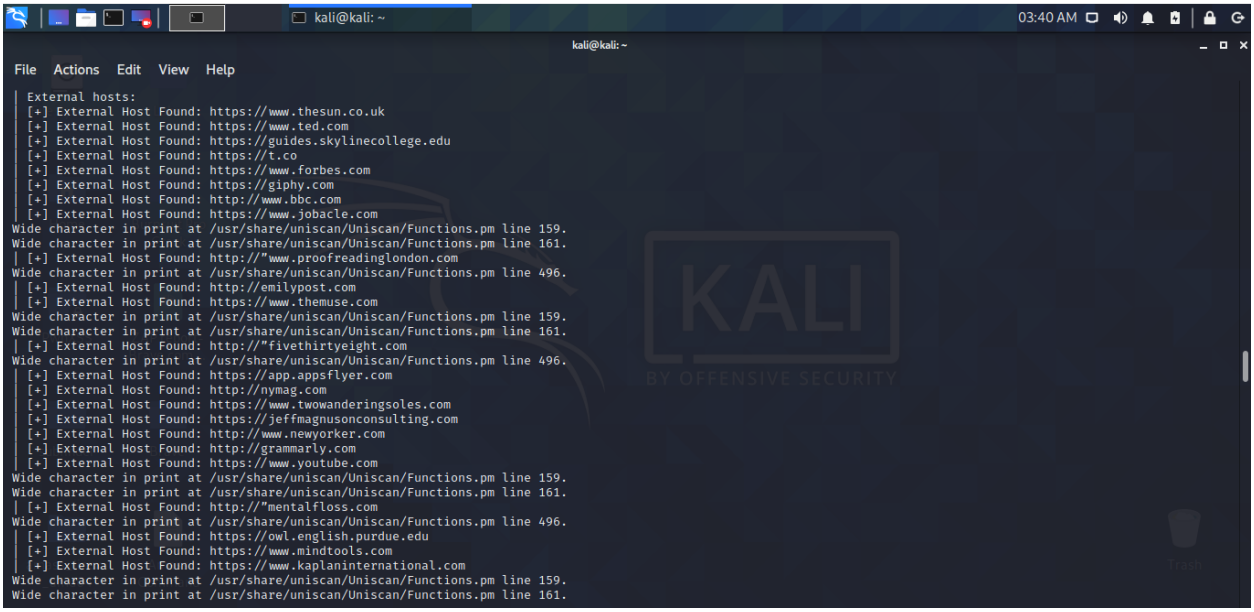


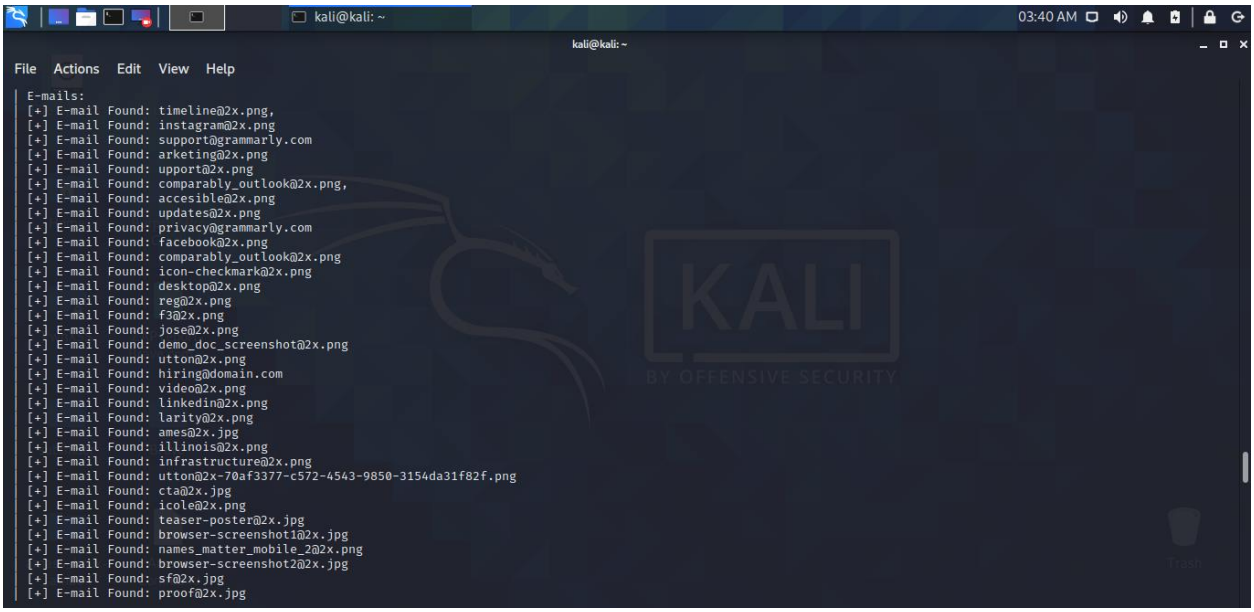
All the directories are in secured manner with code of 200 as shown above and we cannot find any vulnerable directory regarding the target domain.

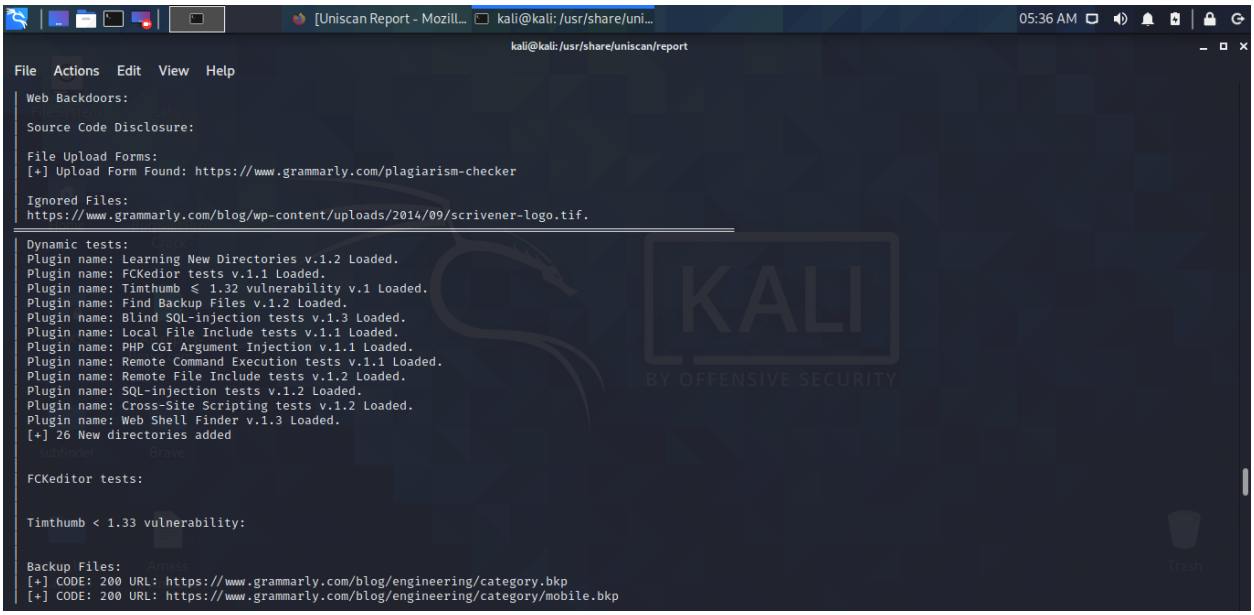
Then I scanned domain by setting the flag with the options of "-wd" in order to find vulnerable information. Using -w flag allow to file check and gives some URLs, and which could yield to vulnerable and -d flag allow to conduct dynamic tests on the aim, such as email recognition, backdoor detection, and SQL and other injection point discovery. To end up this scan takes more time and in order to verify, I went through each of results, but they were not in vulnerable manner and can consider as informative result.

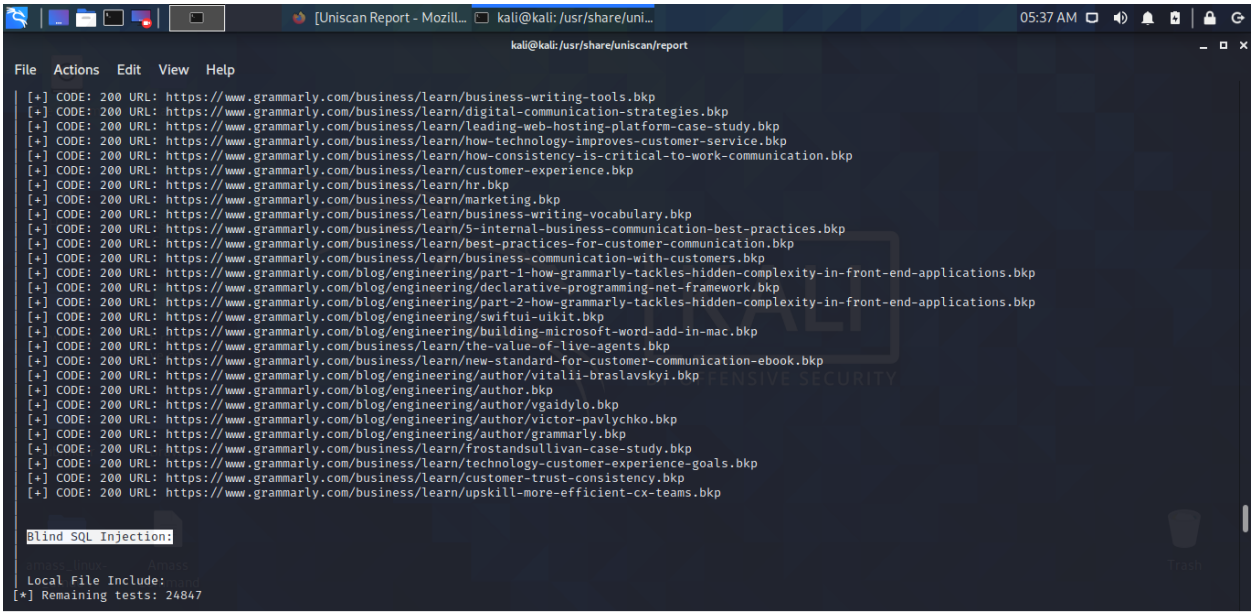
sudo uniscan -u https://www.grammarly.com -wd


kali@kali: ~
03:39 AM
kali@kali: ~
File Actions Edit View Help
┌──(kali㉿kali)-[~]
└─$ sudo uniscan -u https://www.grammarly.com -wd
[sudo] password for kali:
Unknown option: n
###################################
Uniscan project
http://uniscan.sourceforge.net/
###################################
V. 6.3
Scan date: 28-4-2021 3:21:26
| Domain: https://www.grammarly.com/
| IP: 3.208.221.30
| File check:
| [+] CODE: 200 URL: https://www.grammarly.com/favicon.ico
| [+] CODE: 200 URL: https://www.grammarly.com/index.html
| [+] CODE: 200 URL: https://www.grammarly.com/index.php
| [+] CODE: 200 URL: https://www.grammarly.com/robots.txt
| [+] CODE: 200 URL: https://www.grammarly.com/sitemap.xml
| Crawler Started:
| Plugin name: FCKeditor upload test v.1 Loaded.
| Plugin name: External Host Detect v.1.2 Loaded.
| Plugin name: Timthumb ⩽ 1.32 vulnerability v.1 Loaded.
| Plugin name: phpinfo() Disclosure v.1 Loaded.
| Plugin name: E-mail Detection v.1.1 Loaded.
| Plugin name: Web Backdoor Disclosure v.1.1 Loaded.
| Plugin name: Code Disclosure v.1.1 Loaded.
| Plugin name: Upload Form Detect v.1.1 Loaded.
| [+] Crawling finished, 1354 URL's found!

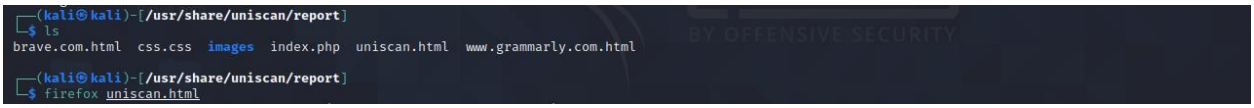

kali@kali: ~
03:40 AM
kali@kali: ~
File Actions Edit View Help
| External hosts:
| [+] External Host Found: https://www.thesun.co.uk
| [+] External Host Found: https://www.ted.com
| [+] External Host Found: https://guides.skylinecollege.edu
| [+] External Host Found: https://t.co
| [+] External Host Found: https://www.forbes.com
| [+] External Host Found: https://giphy.com
| [+] External Host Found: http://www.bbc.com
| [+] External Host Found: https://www.jobacle.com
Wide character in print at /usr/share/uniscan/Uniscan/Functions.pm line 159.
Wide character in print at /usr/share/uniscan/Uniscan/Functions.pm line 161.
| [+] External Host Found: http://"www.proofreadinglondon.com
Wide character in print at /usr/share/uniscan/Uniscan/Functions.pm line 496.
| [+] External Host Found: http://emilypost.com
| [+] External Host Found: https://www.themuse.com
Wide character in print at /usr/share/uniscan/Uniscan/Functions.pm line 159.
Wide character in print at /usr/share/uniscan/Uniscan/Functions.pm line 161.
| [+] External Host Found: http://"fivethirtyeight.com
Wide character in print at /usr/share/uniscan/Uniscan/Functions.pm line 496.
| [+] External Host Found: https://app.appsflyer.com
| [+] External Host Found: http://nymag.com
| [+] External Host Found: https://www.twowanderingsoles.com
| [+] External Host Found: https://jeffmagnusonconsulting.com
| [+] External Host Found: http://www.newyorker.com
| [+] External Host Found: http://grammarly.com
| [+] External Host Found: https://www.youtube.com
Wide character in print at /usr/share/uniscan/Uniscan/Functions.pm line 159.
Wide character in print at /usr/share/uniscan/Uniscan/Functions.pm line 161.
| [+] External Host Found: http://"mentalfloss.com
Wide character in print at /usr/share/uniscan/Uniscan/Functions.pm line 496.
| [+] External Host Found: https://owl.english.purdue.edu
| [+] External Host Found: https://www.mindtools.com
| [+] External Host Found: https://www.kaplaninternational.com
Wide character in print at /usr/share/uniscan/Uniscan/Functions.pm line 159.
Wide character in print at /usr/share/uniscan/Uniscan/Functions.pm line 161.

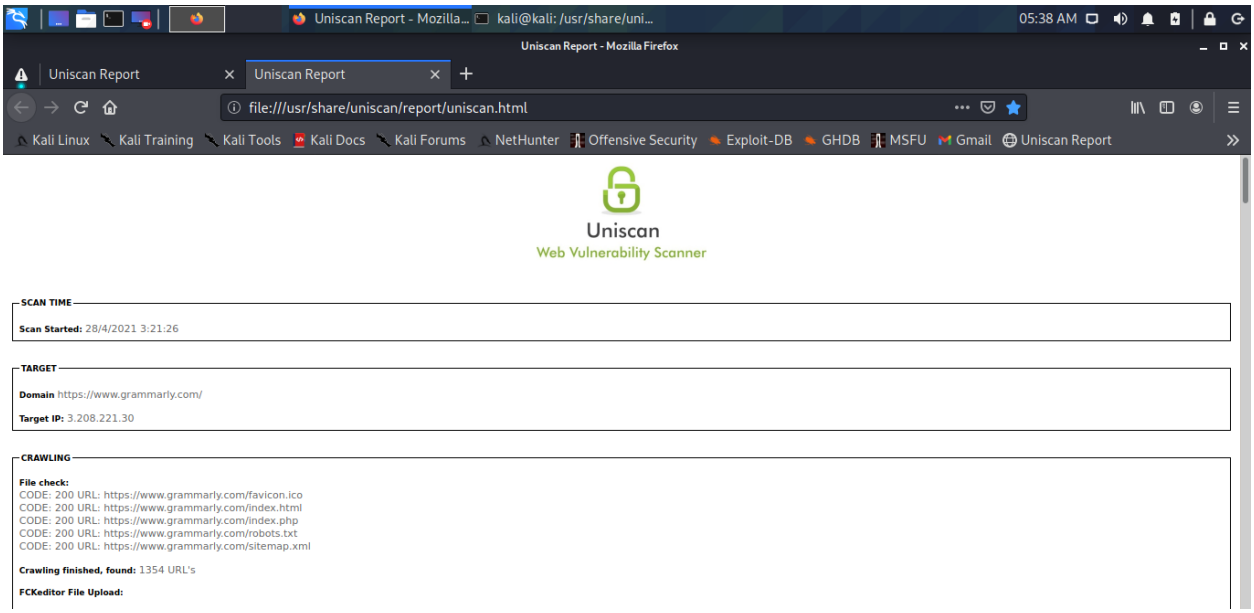

kali@kali: ~
03:40 AM
kali@kali: ~
File Actions Edit View Help
E-mails:
[+] E-mail Found: timeline@2x.png,
[+] E-mail Found: instagram@2x.png
[+] E-mail Found: support@grammarly.com
[+] E-mail Found: arketing@2x.png
[+] E-mail Found: upport@2x.png
[+] E-mail Found: comparably_outlook@2x.png,
[+] E-mail Found: accesible@2x.png
[+] E-mail Found: updates@2x.png
[+] E-mail Found: privacy@grammarly.com
[+] E-mail Found: facebook@2x.png
[+] E-mail Found: comparably_outlook@2x.png
[+] E-mail Found: icon-checkmark@2x.png
[+] E-mail Found: desktop@2x.png
[+] E-mail Found: reg@2x.png
[+] E-mail Found: f3@2x.png
[+] E-mail Found: jose@2x.png
[+] E-mail Found: demo_doc_screenshot@2x.png
[+] E-mail Found: utton@2x.png
[+] E-mail Found: hiring@domain.com
[+] E-mail Found: video@2x.png
[+] E-mail Found: linkedin@2x.png
[+] E-mail Found: larity@2x.png
[+] E-mail Found: ames@2x.jpg
[+] E-mail Found: illinois@2x.png
[+] E-mail Found: infrastructure@2x.png
[+] E-mail Found: utton@2x-70af3377-c572-4543-9850-3154da31f82f.png
[+] E-mail Found: cta@2x.jpg
[+] E-mail Found: icole@2x.png
[+] E-mail Found: teaser-poster@2x.jpg
[+] E-mail Found: browser-screenshot1@2x.jpg
[+] E-mail Found: names_matter_mobile_2@2x.png
[+] E-mail Found: browser-screenshot2@2x.jpg
[+] E-mail Found: sf@2x.jpg
[+] E-mail Found: proof@2x.jpg

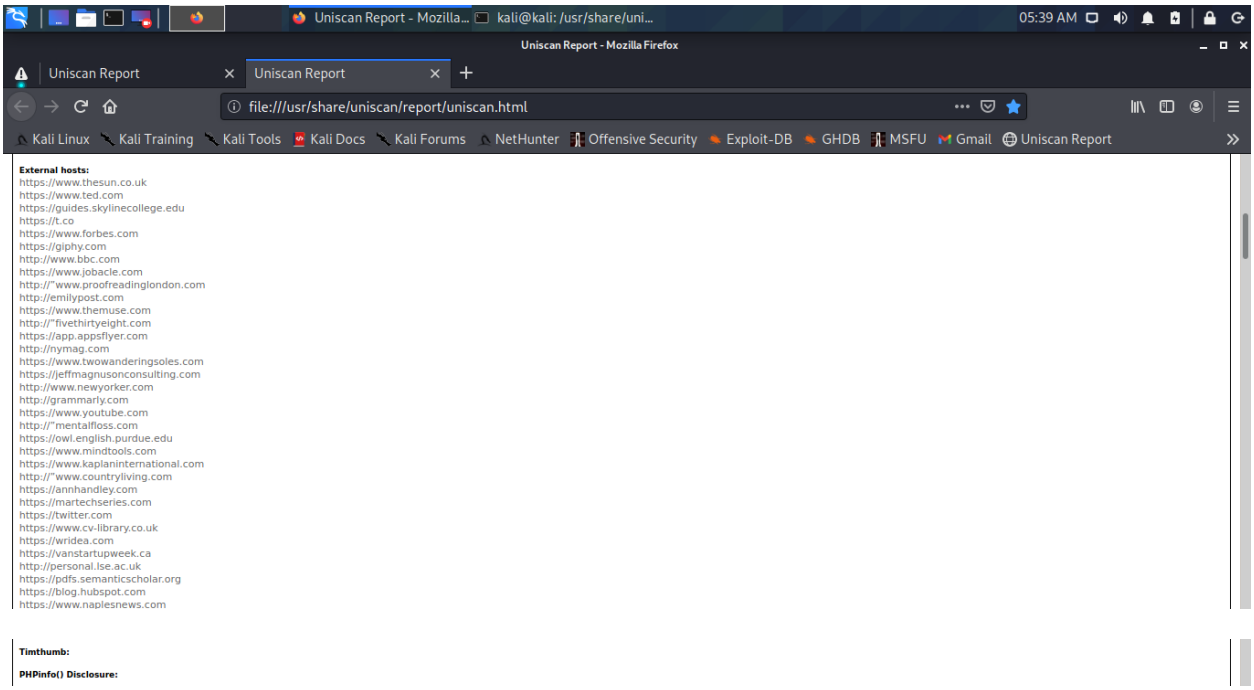

[Uniscan Report - Mozill...
kali@kali: /usr/share/uni...
05:36 AM
kali@kali: /usr/share/uniscan/report
File Actions Edit View Help
Web Backdoors:
Source Code Disclosure:
File Upload Forms:
[+] Upload Form Found: https://www.grammarly.com/plagiarism-checker
Ignored Files:
https://www.grammarly.com/blog/wp-content/uploads/2014/09/scrivener-logo.tif.
Dynamic tests:
Plugin name: Learning New Directories v.1.2 Loaded.
Plugin name: FCKedior tests v.1.1 Loaded.
Plugin name: Timthumb ⩽ 1.32 vulnerability v.1 Loaded.
Plugin name: Find Backup Files v.1.2 Loaded.
Plugin name: Blind SQL-injection tests v.1.3 Loaded.
Plugin name: Local File Include tests v.1.1 Loaded.
Plugin name: PHP CGI Argument Injection v.1.1 Loaded.
Plugin name: Remote Command Execution tests v.1.1 Loaded.
Plugin name: Remote File Include tests v.1.2 Loaded.
Plugin name: SQL-injection tests v.1.2 Loaded.
Plugin name: Cross-Site Scripting tests v.1.2 Loaded.
Plugin name: Web Shell Finder v.1.3 Loaded.
[+] 26 New directories added
FCKeditor tests:
Timthumb < 1.33 vulnerability:
Backup Files:
[+] CODE: 200 URL: https://www.grammarly.com/blog/engineering/category.bkp
[+] CODE: 200 URL: https://www.grammarly.com/blog/engineering/category/mobile.bkp

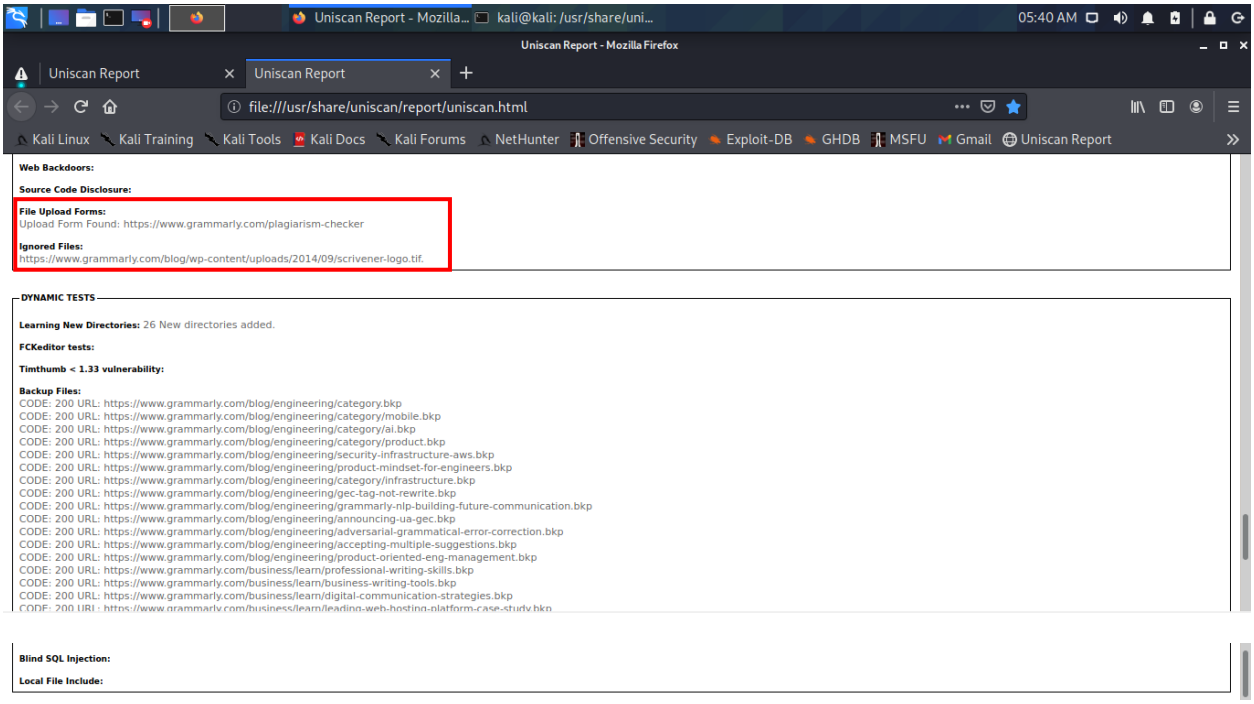


Uniscan also saves each scan as an HTML file under **/usr/share/uniscan/report/** directory and we can simply examine the results at a later time in graphical manner.



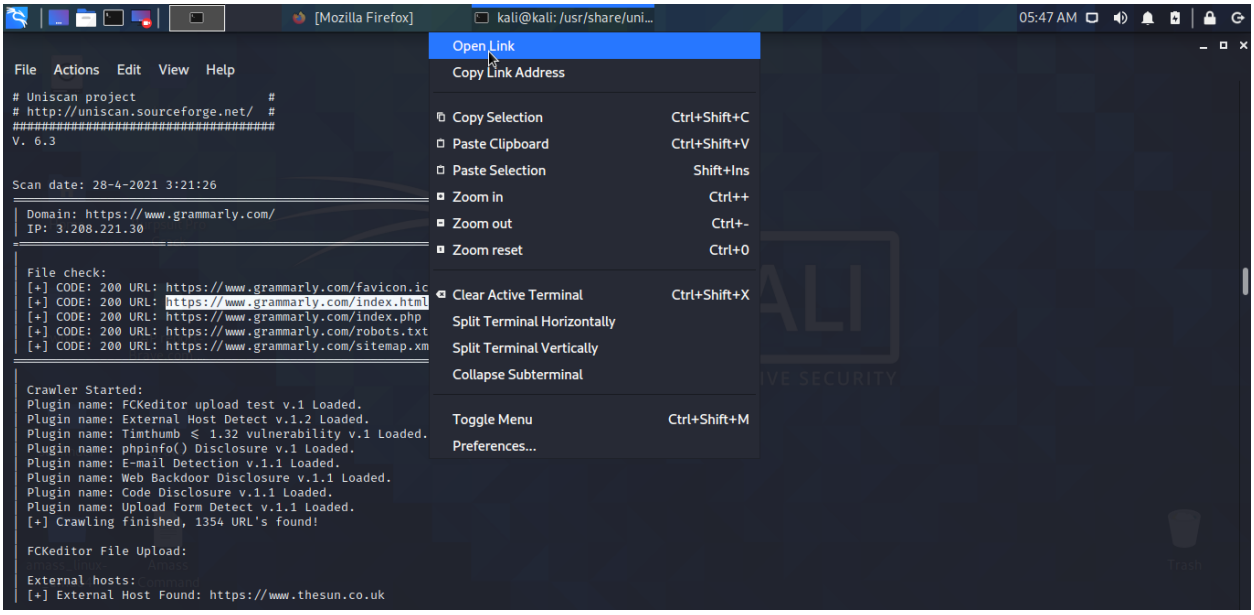


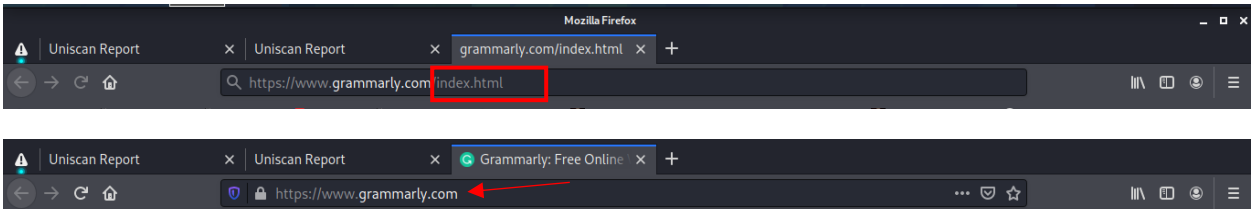




As shown above figures, we did not find any Web Backdoors, Source file disclosure, PHP info disclosure details and Blind SQL injection vulnerabilities. With the help of this tool even we found some external host addresses, e-mail addresses and backup files; all of them are informative. Under the crawling, it scans for files and all of those files are in secured manner with the code of 200 as shown in the figure.

Furthermore, I have checked some outputted URL under the file check category as shown below, and even it shows the path of the link as “index.html”, it simply redirected to the main domain of grammarly.com when it opens trough the browser. Steps are shown below.

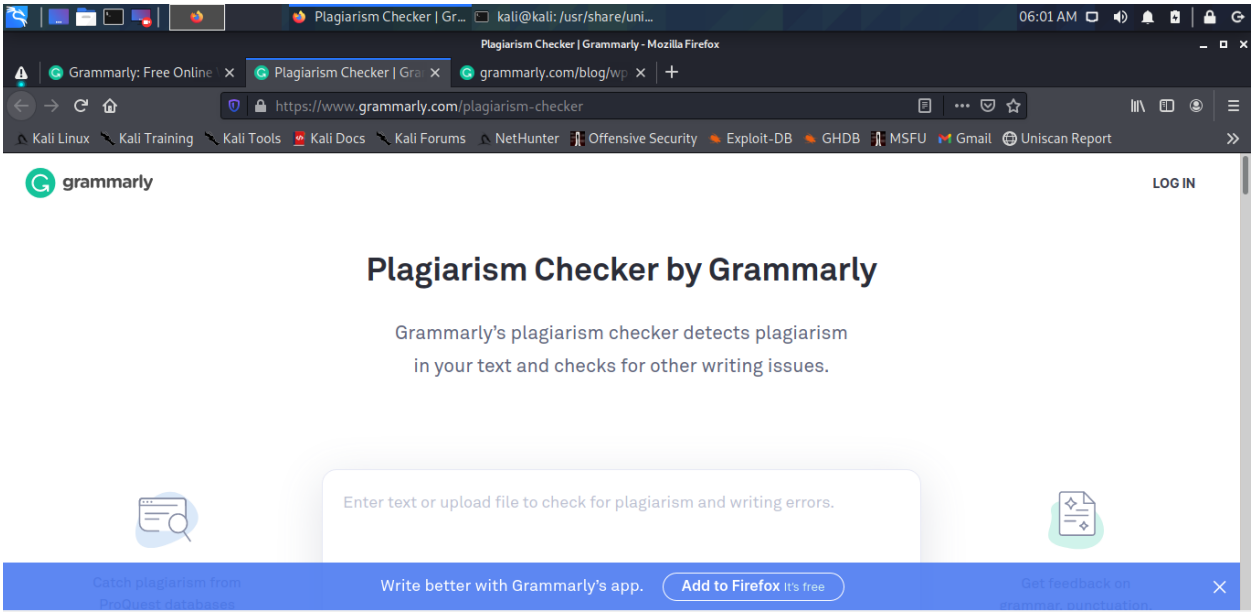




Using that tool, I was able to find a file uploading form under the File Upload Forms checking category and it is a simple plagiarism checking form which belongs to the same domain and that also can be considered as an informational result.

| File Upload Forms:

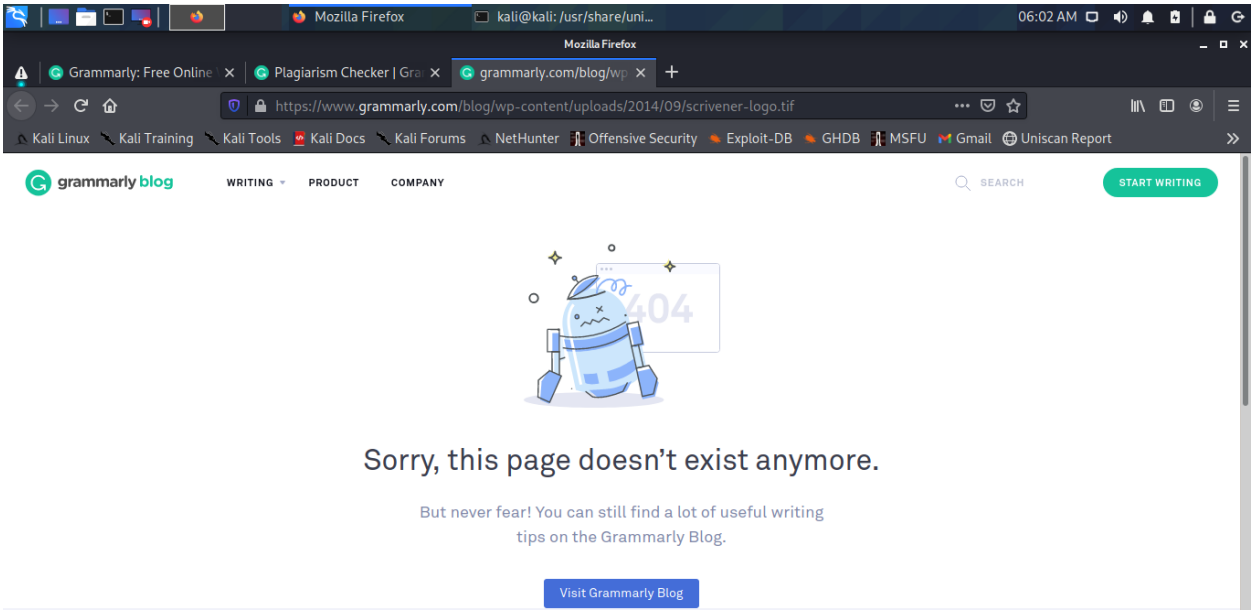
| [+] Upload Form Found: https://www.grammarly.com/plagiarism-checker



Also, there was an ignored URL under the Ignored File category and as it says that page is not exist now. There was only a direction to the main domain as previous.

| Ignored Files:

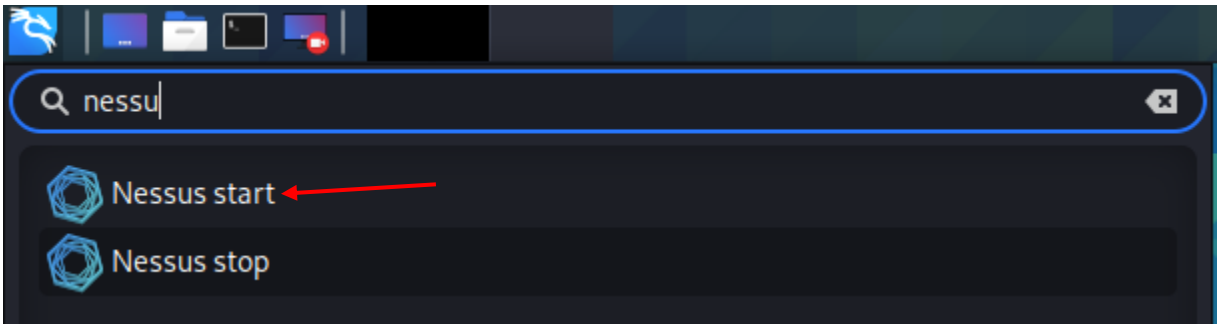
| https://www.grammarly.com/blog/wp-content/uploads/2014/09/scrivener-logo.tif.



III. Nessus

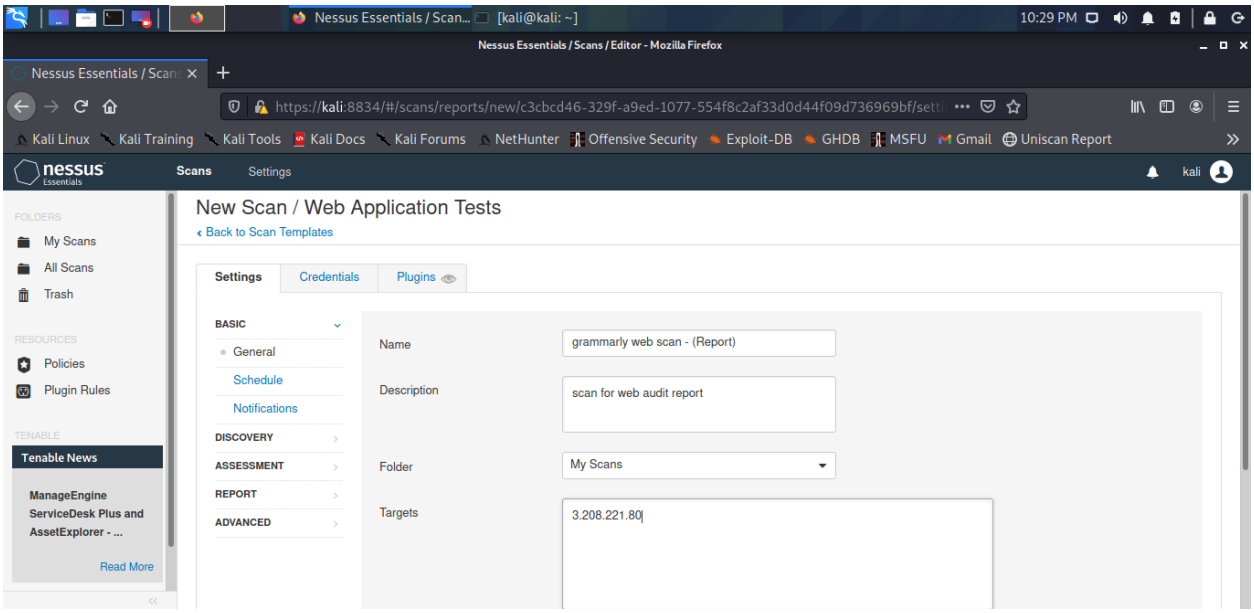
Nessus is one of the security scanners used in vulnerability tests and penetration testing, as well as malware attacks. Plug-ins are used by the Nessus scanning engine to find new vulnerabilities. Within 24 hours of a flaw being public, Tenable distributes plug-ins with the most up-to-date information to customers' systems. Customers get regular plug-in feeds to keep current and new bugs occur virtually every day. While Nessus does not have penetration testing capability, administrators can use the results of Nessus scans in conjunction with common penetration testing tools like Metasploit, Core IMPACT, and Immunity CANVAS to gain insight into risk.

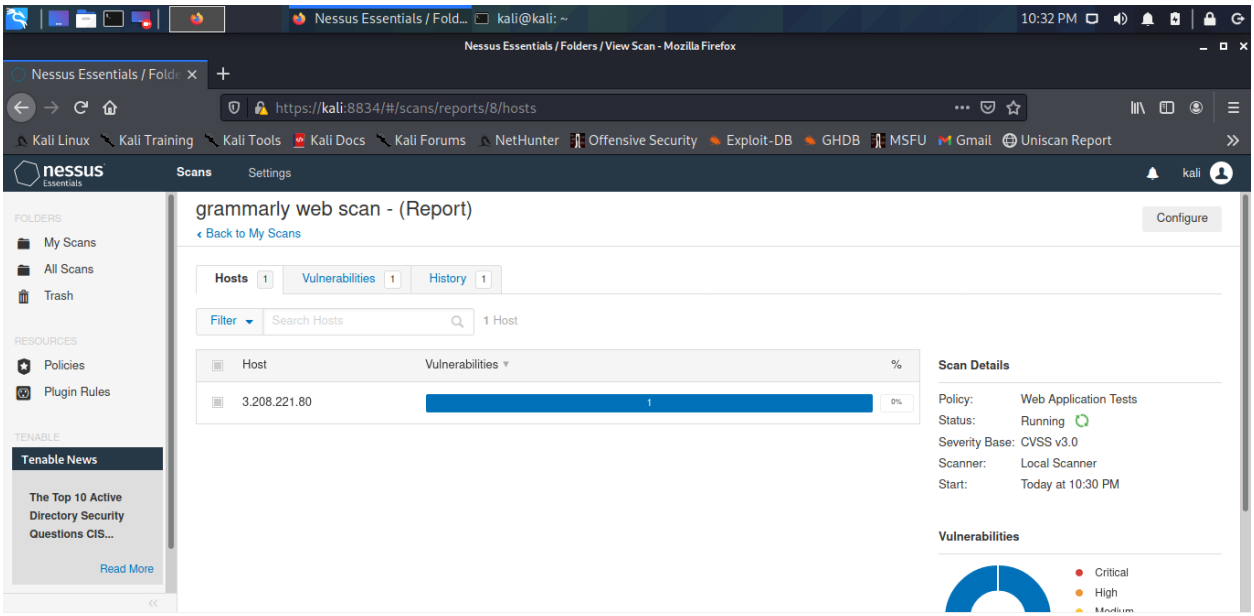
The downloadable installer can be found for Linux-based systems in the Nessus official webpage, and we need to install it using terminal. After the install we need to start Nessus daemon by terminal or simply searching in Kali environment as shown below.

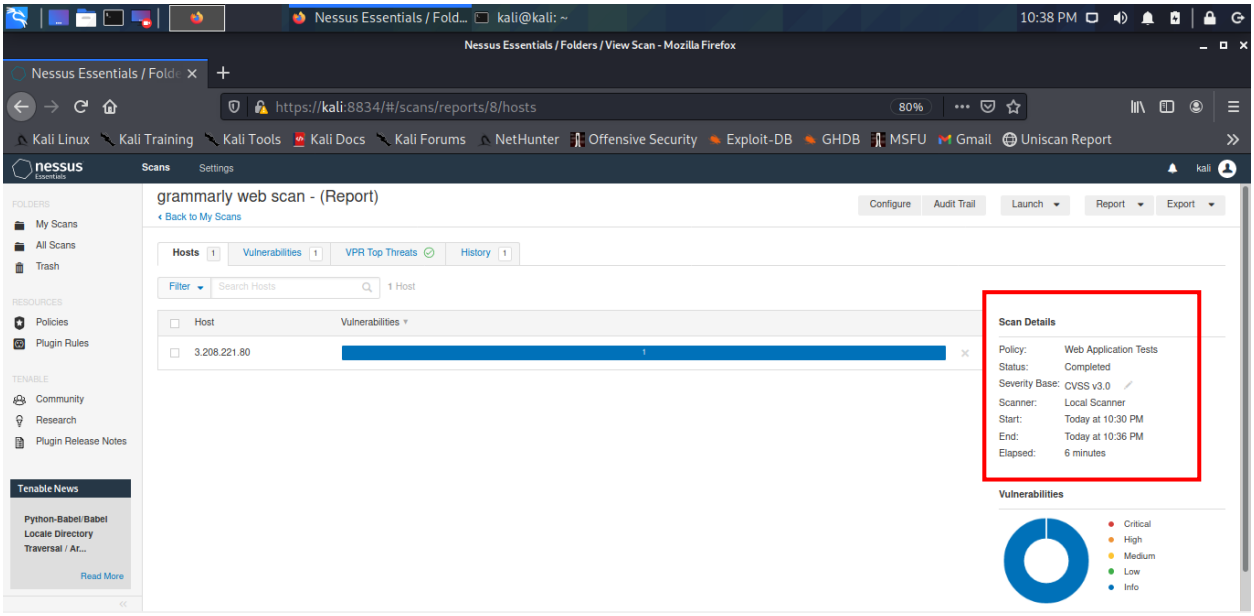


After login to the Nessus with valid credentials, we can start our scans as we prefer. Then I was done Web Application Tests scan to my target domain by filling the relevant information as shown below. We need to enter our target domain IP separately as the target or we can enter several IPs regarding to our target domain.

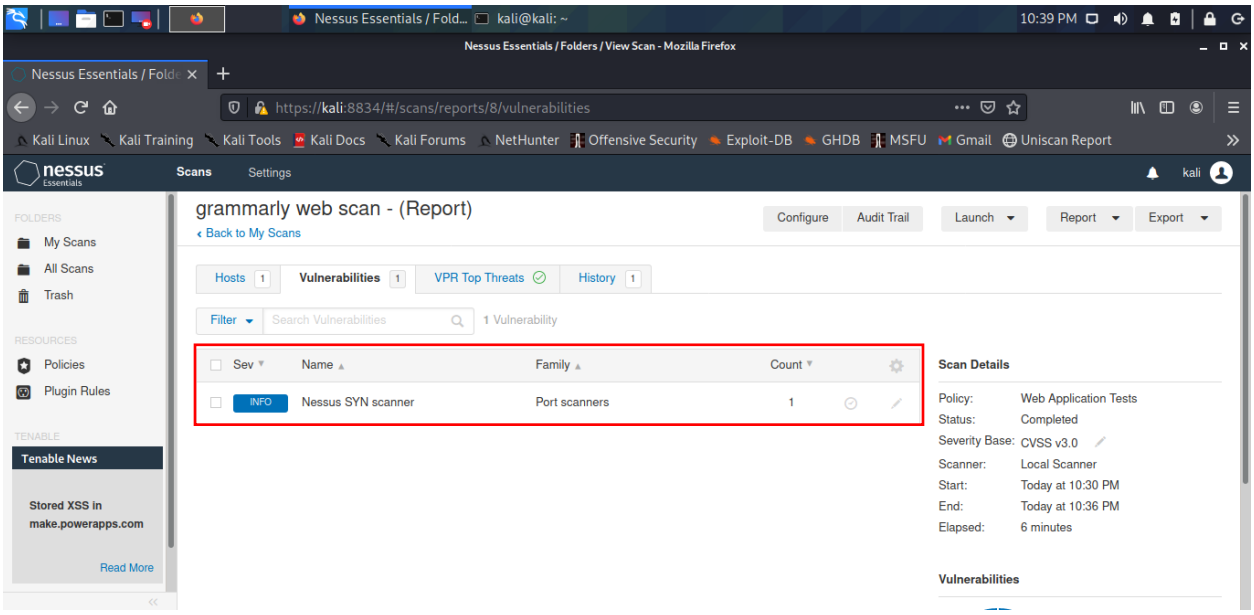
First, I have done Web Application Test separate scan regarding my target domain as shown below.

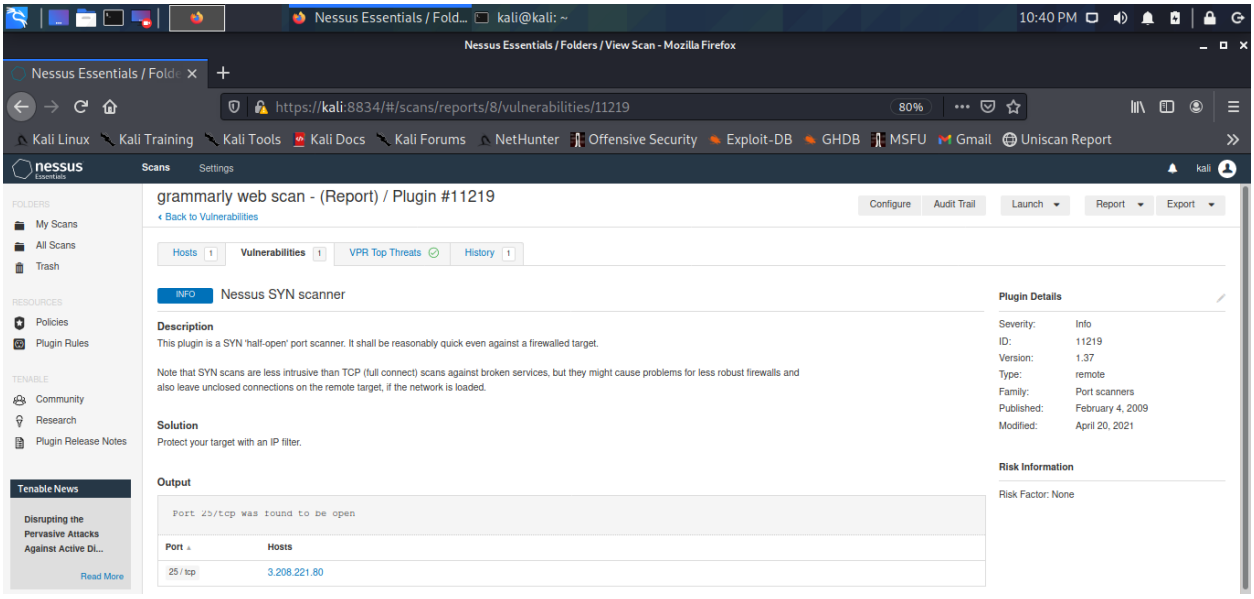


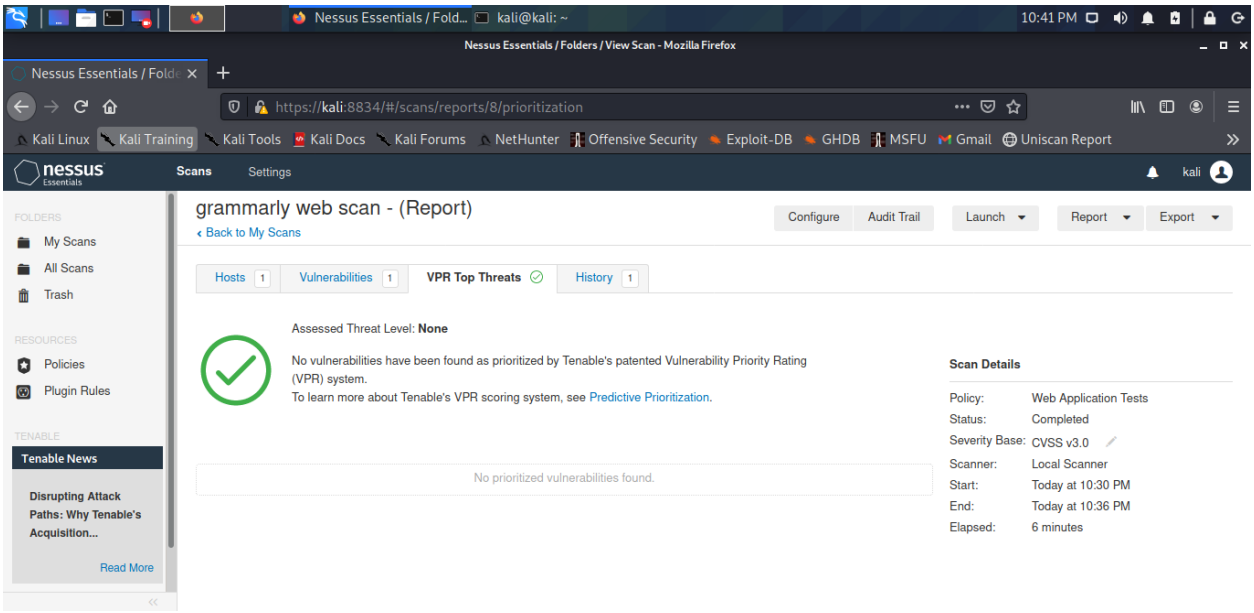




It was completed within 6 minutes and it did not give any low, medium, high, and critical vulnerabilities. It only gave one vulnerability which was falls under the info category as shown below.

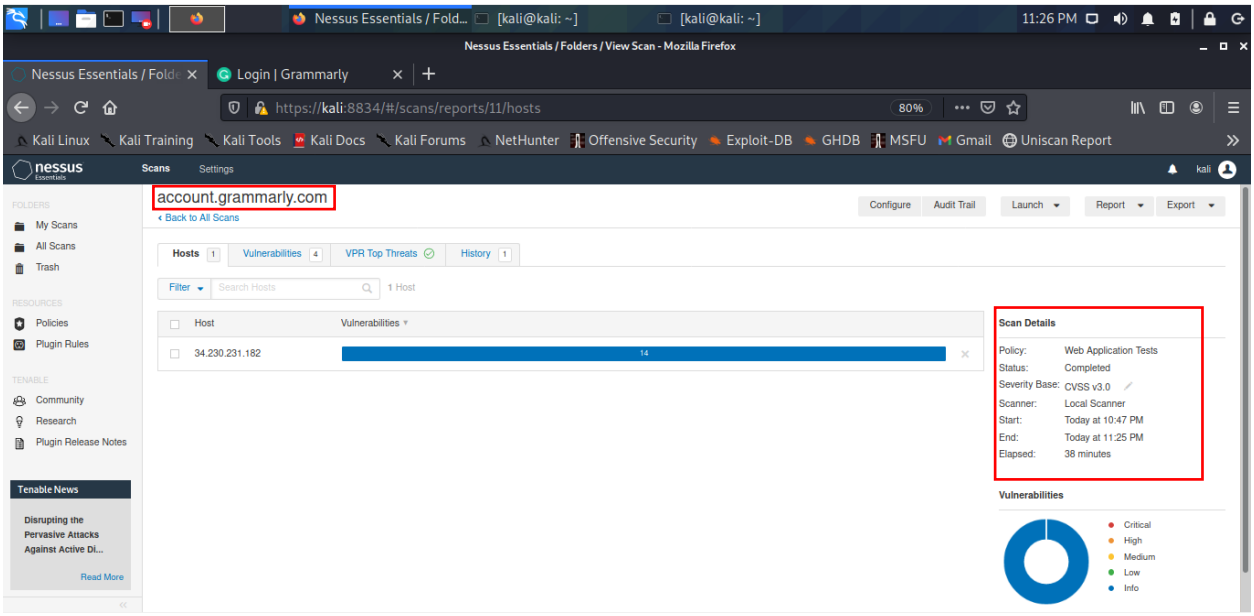


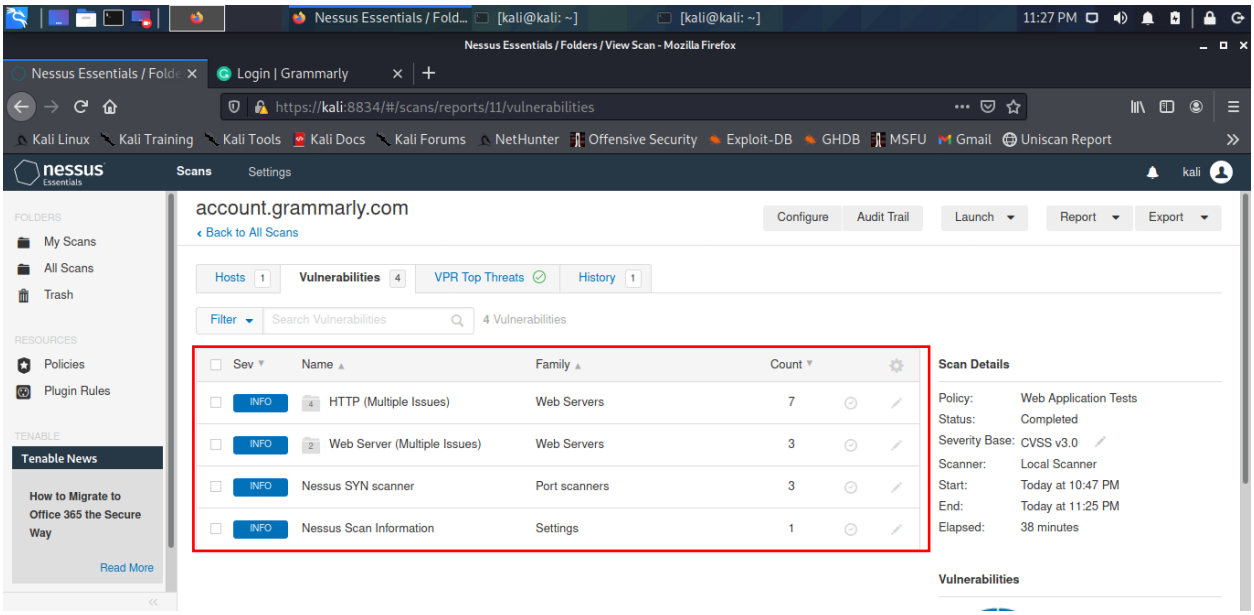




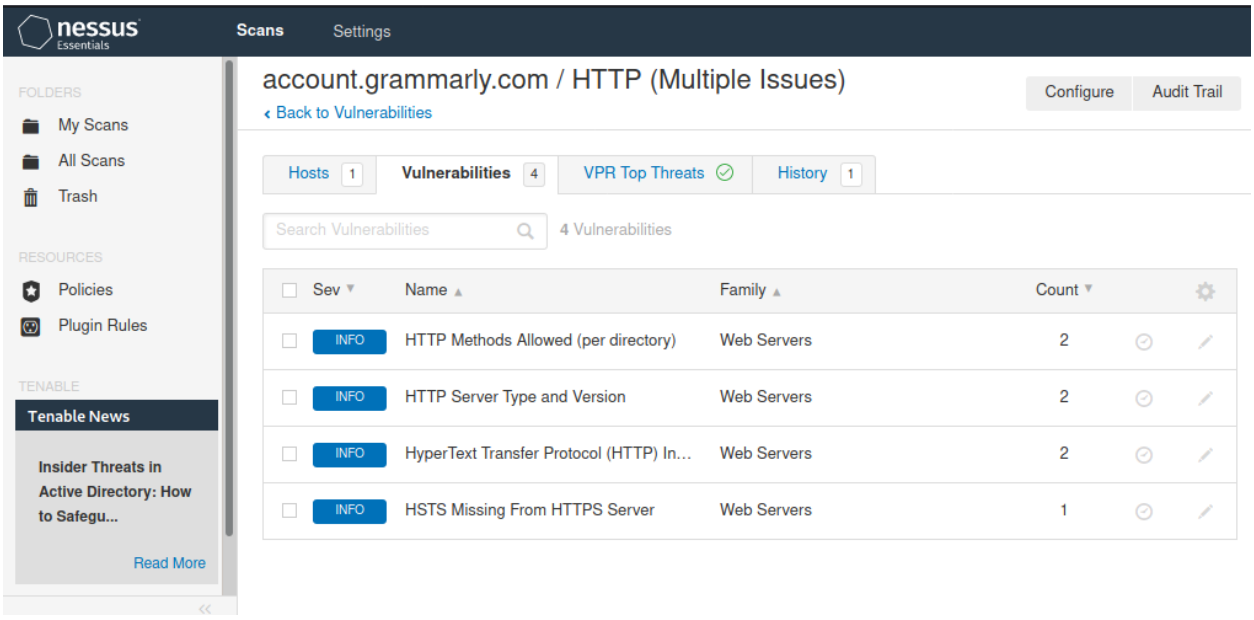
As shown above figure, no vulnerabilities have been found as prioritized by Tenable's patented Vulnerability Priority Rating (VPR) system and Assessed Treat Level was none.

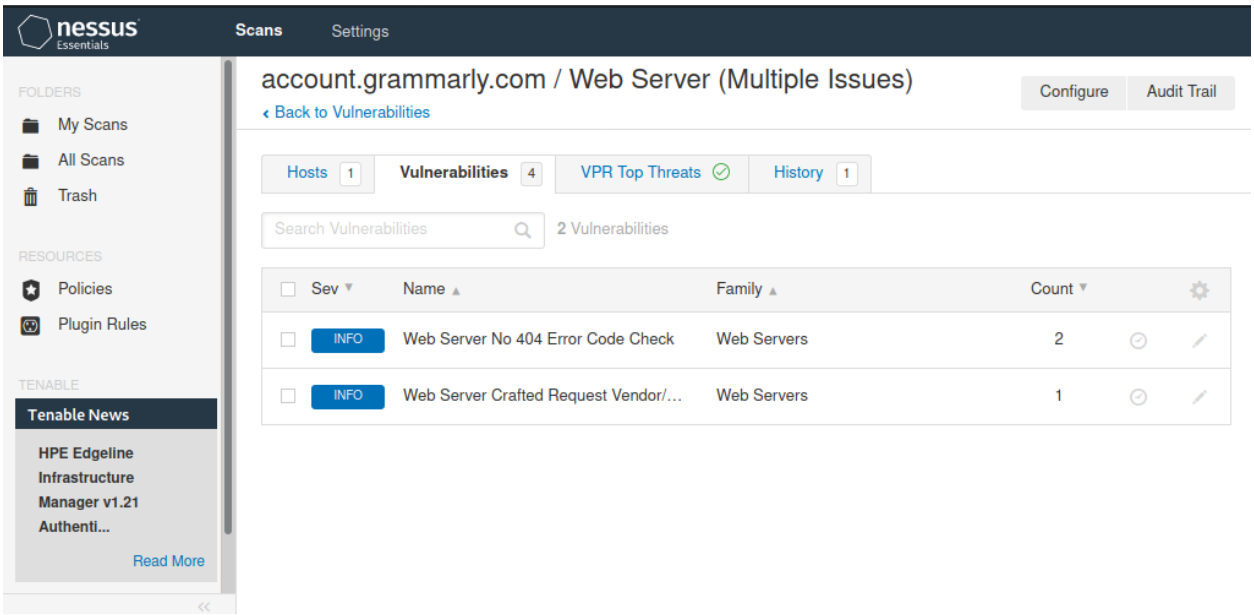
Then I separately scanned one of sub-domain under my target domain, but it took more time to finished than the previous scan of main-domain. My target sub-domain was, account.grammarly.com.

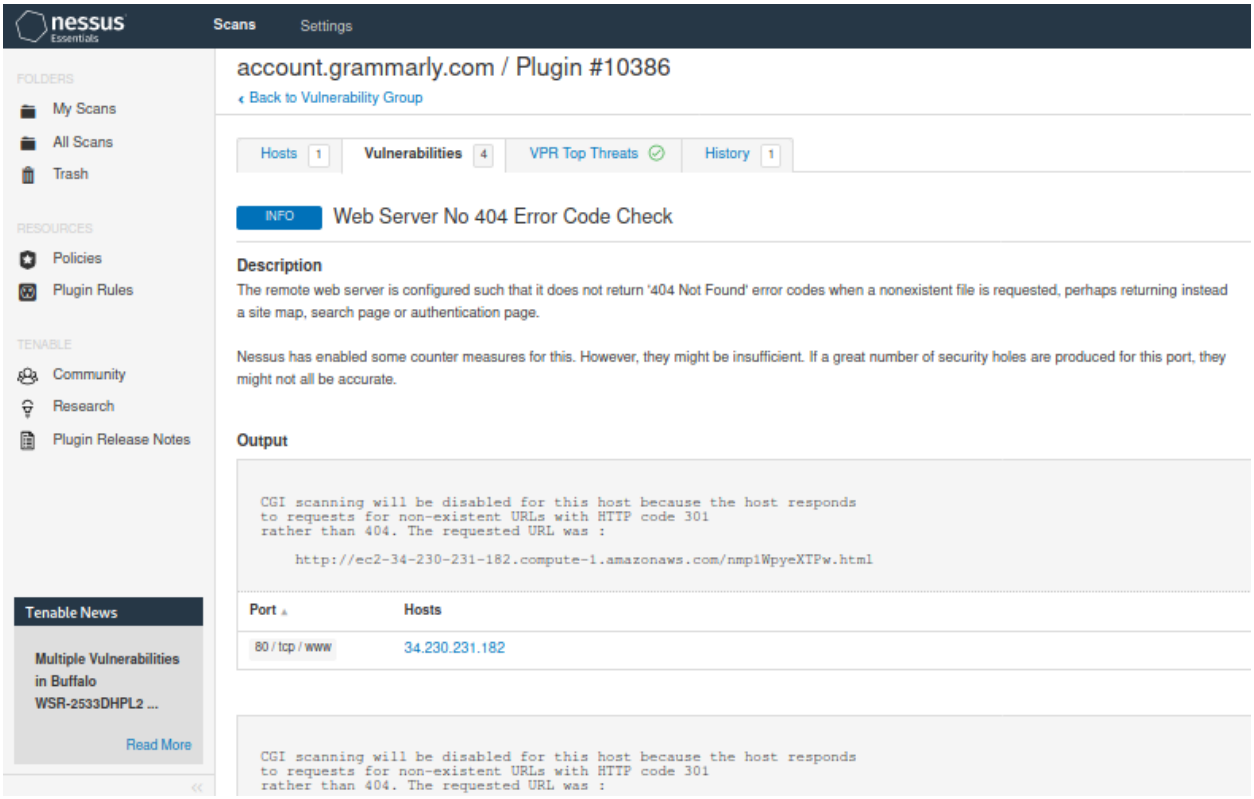


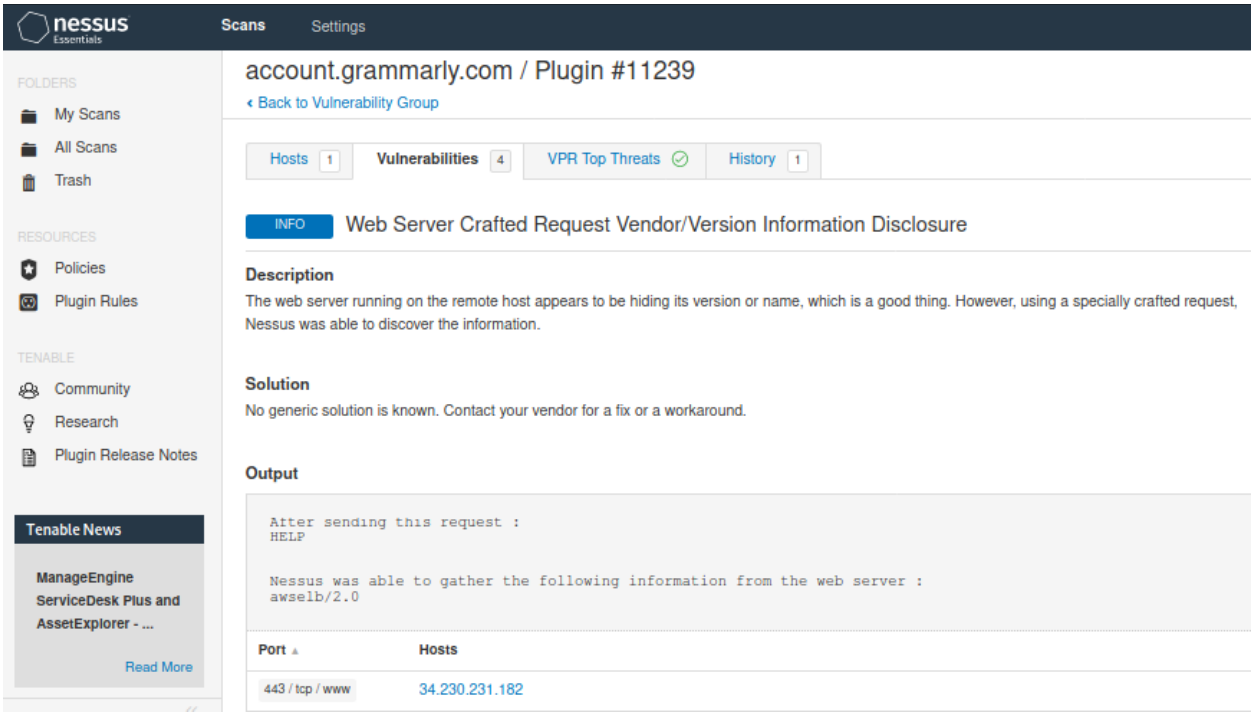


Even that scan end up with giving several vulnerabilities, all of them were under the info category and we could not find any low, medium, high, or critical vulnerabilities as the previous scan.


nessus Essentials
Scans
Settings
FOLDERS
My Scans
All Scans
Trash
RESOURCES
Policies
Plugin Rules
TENABLE
Tenable News
Insider Threats in Active Directory: How to Safegu...
Read More
account.grammarly.com / HTTP (Multiple Issues)
Back to Vulnerabilities
Configure
Audit Trail
Hosts 1
Vulnerabilities 4
VPR Top Threats
History 1
Search Vulnerabilities
4 Vulnerabilities
Sev
Name
Family
Count
INFO
HTTP Methods Allowed (per directory)
Web Servers
2
INFO
HTTP Server Type and Version
Web Servers
2
INFO
HyperText Transfer Protocol (HTTP) In...
Web Servers
2
INFO
HSTS Missing From HTTPS Server
Web Servers
1

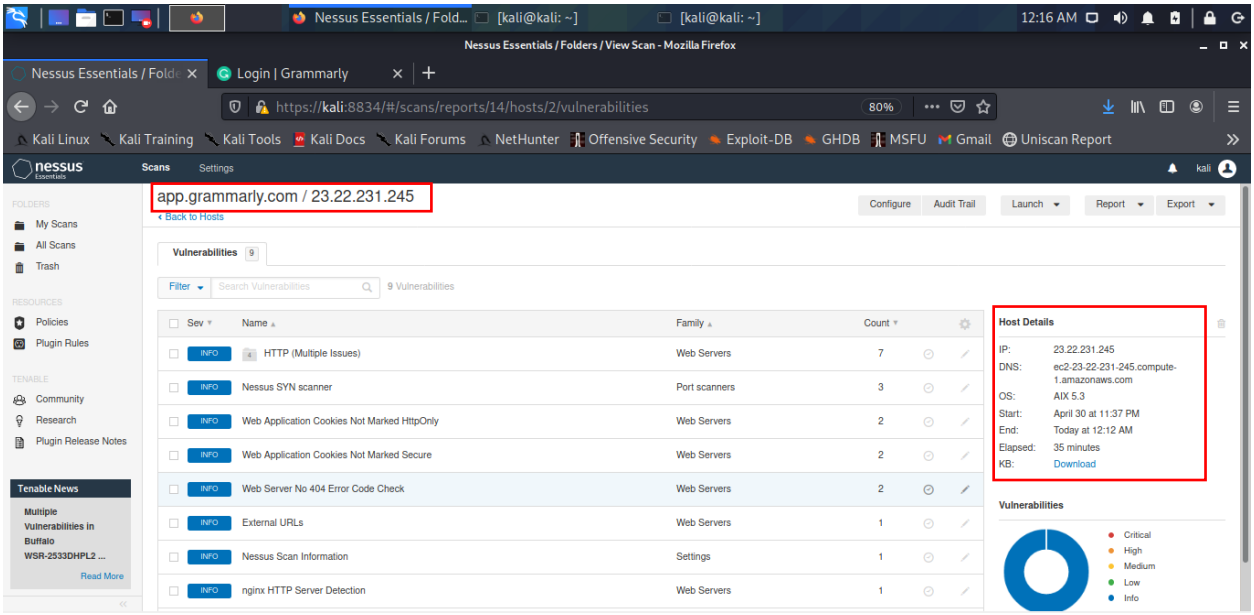

nessus Essentials
Scans
Settings
FOLDERS
My Scans
All Scans
Trash
RESOURCES
Policies
Plugin Rules
TENABLE
Tenable News
HPE Edgeline Infrastructure Manager v1.21 Authenti...
Read More
account.grammarly.com / Web Server (Multiple Issues)
Back to Vulnerabilities
Configure
Audit Trail
Hosts 1
Vulnerabilities 4
VPR Top Threats
History 1
Search Vulnerabilities
2 Vulnerabilities
Sev
Name
Family
Count
INFO
Web Server No 404 Error Code Check
Web Servers
2
INFO
Web Server Crafted Request Vendor/...
Web Servers
1





Above figures shows the result of sub-domain info vulnerabilities and the descriptions and the solutions which were already given by the Nessus tool.

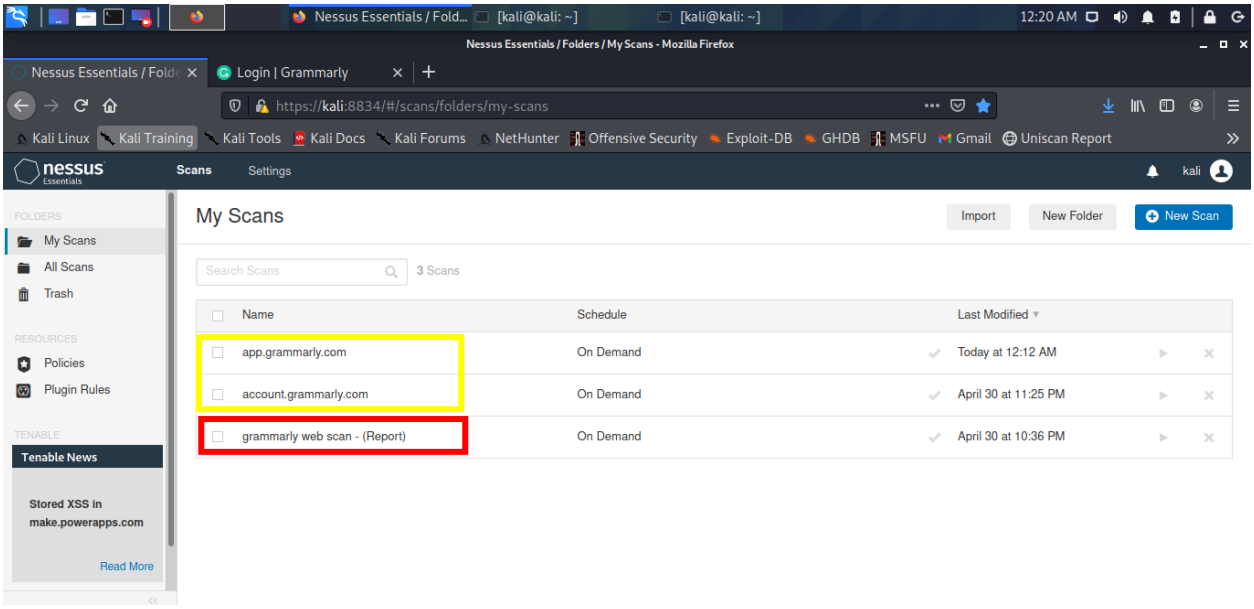
Furthermore, I have done another Nessus web application test regarding the sub-domain of app.grammarly.com as previous sub-domain scan since both of those sub-domains fallen under the scope of bug-bounty rules and regulation which were given by the Grammarly.



As previous two scans, this was also end up with several vulnerabilities which were fallen under the info category. Even though I have done several no of

separate scans regarding my target domain and its sub-domains, I could not find any low, medium, high, or critical vulnerabilities.

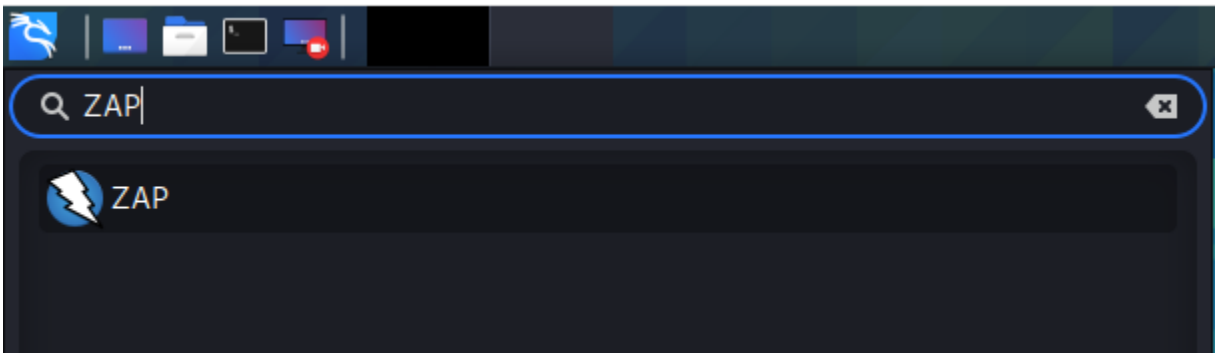
The below figure shows all the scans including main-domain and its sub-domains using Tenable's Nessus tool.



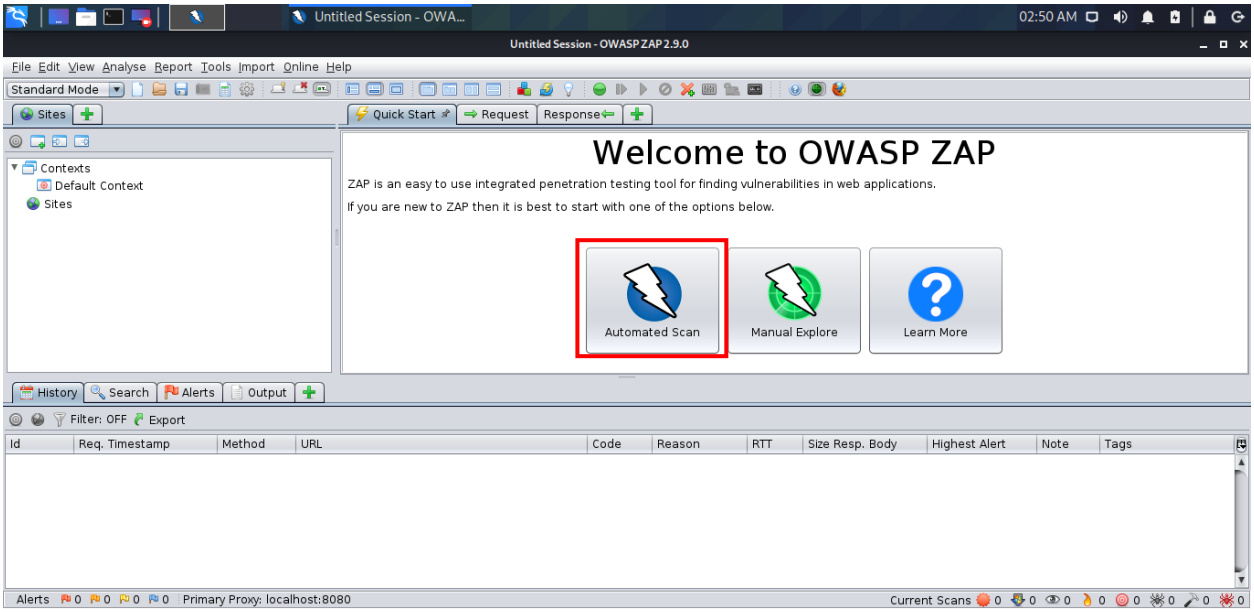
III. OWASP Zap

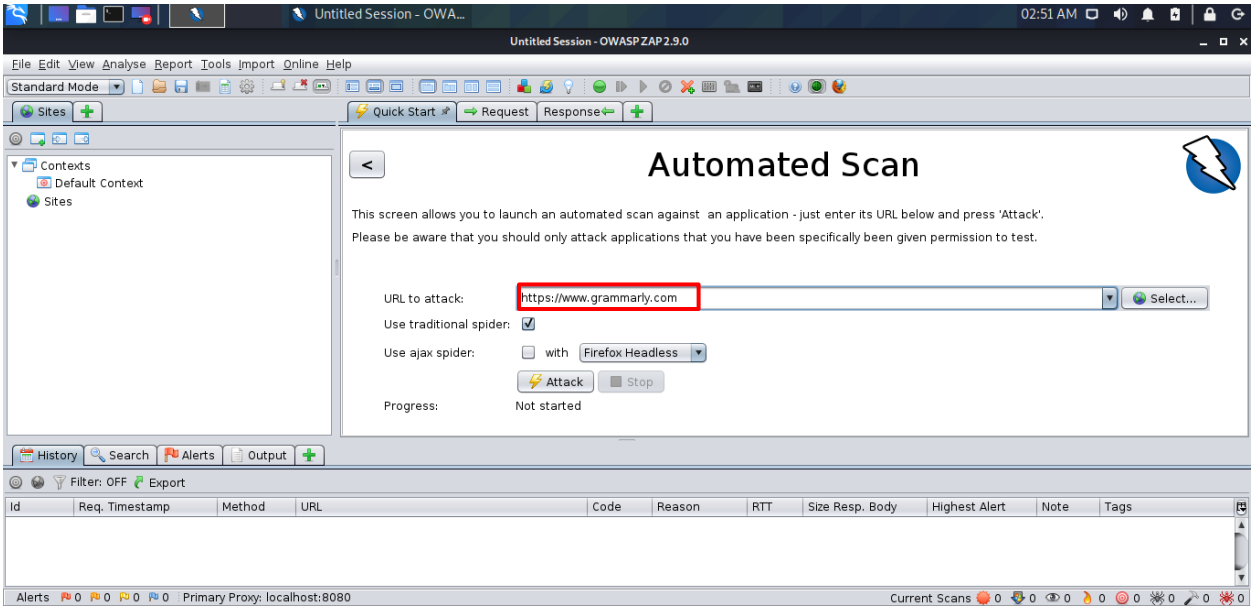
One of the most widely used web application security monitoring techniques is the OWASP Zed Attack Proxy (ZAP). OWASP contributes to and maintains it as an open source project, which is available for free. The Open Web Application Security Project (OWASP) is a vendor-neutral, non-profit organization of people devoted to improving the security of web applications. Web application security testers may use the OWASP Zed Attack Proxy, a Java-based platform with an elegant graphical GUI, to perform fuzzing, scripting, spidering, and proxying in order to attack web applications. Since it is a Java tool, it can operate on almost every operating system that supports Java. ZAP is used in the Kali Linux Penetration Testing OS by default, but it can also be downloaded and installed on any OS that has Java installed.

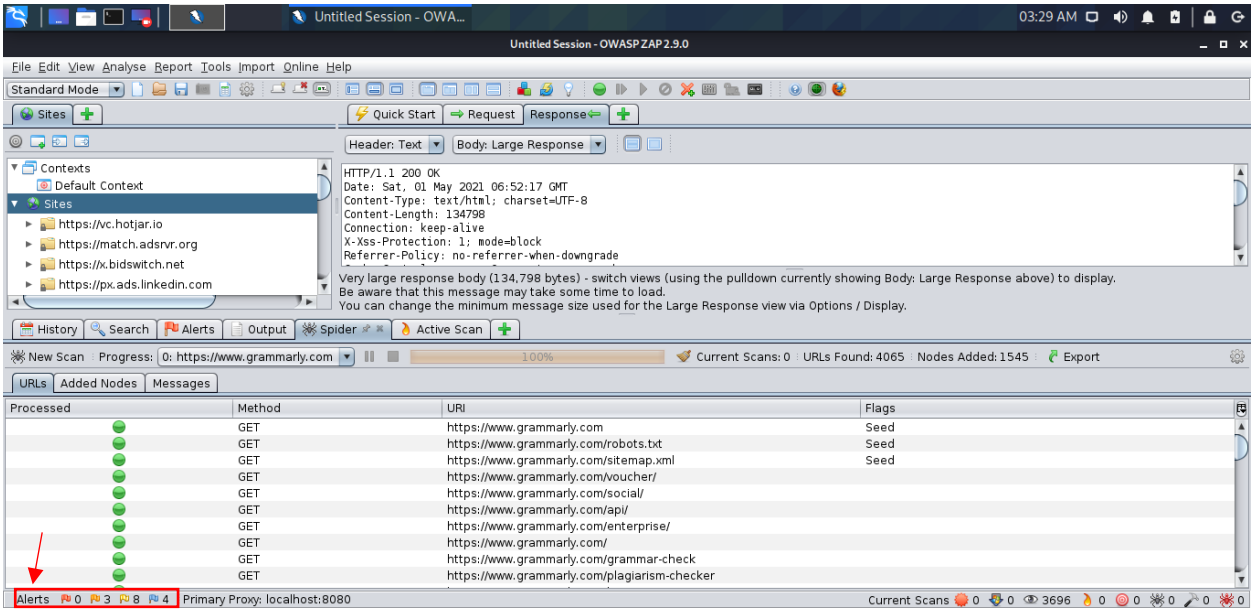
Since ZAP is used in the Kali Linux Penetration Testing OS by default, it can be accessed by simply searching and selecting the ZAP in the search bar in Kali Linux as shown below.



First, I have selected the scan type as automated and enter my target domain's URL as shown below.



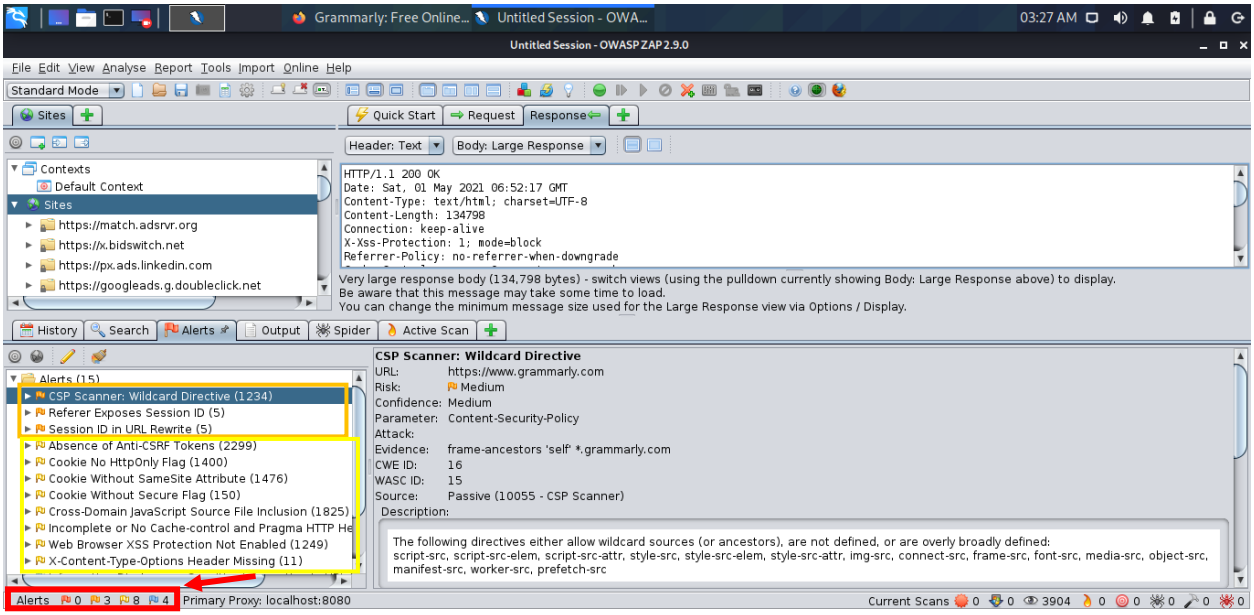




After several hours of scanning the target domain, it was end up with finding 3 medium, 8 low and 4 informational priority alerts as shown in below figures.

i. 3 medium priority vulnerabilities:

1. CSP Scanner: Wildcard Directive (1234)

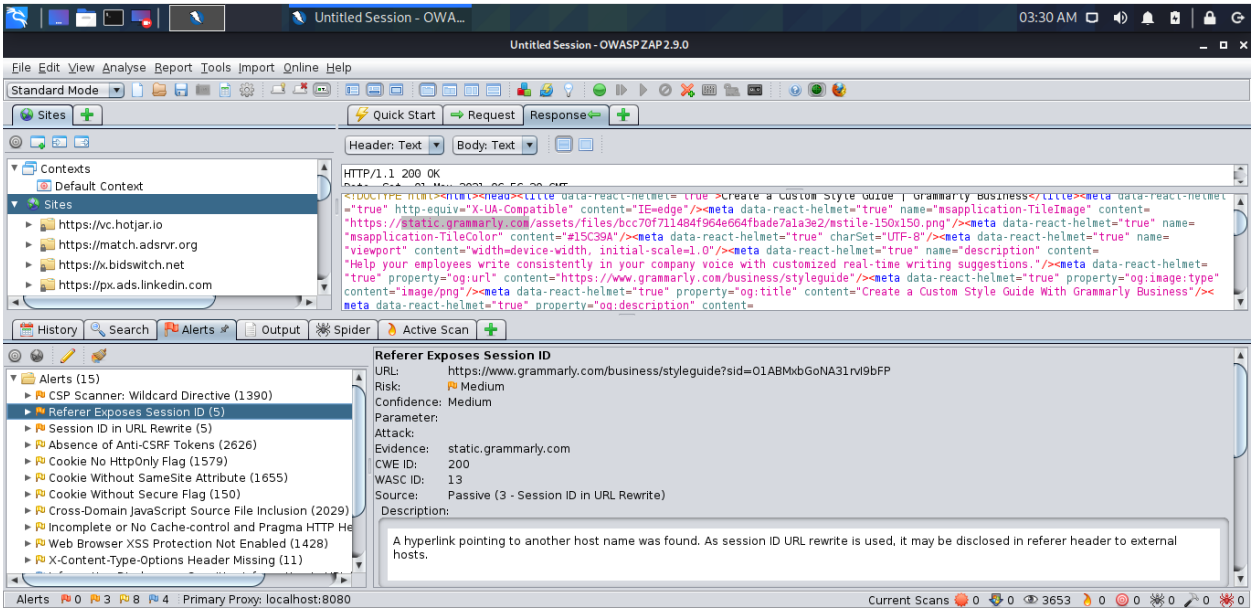


As the figure shows, this vulnerability was in under the medium priority alert with the Content-Security-Policy (CSP) parameter.

Solution:

As the solution, grammarly need to configure the CSP header properly with the web server, application server, load balancer, etc.

2. Referer Exposes Session ID (5)

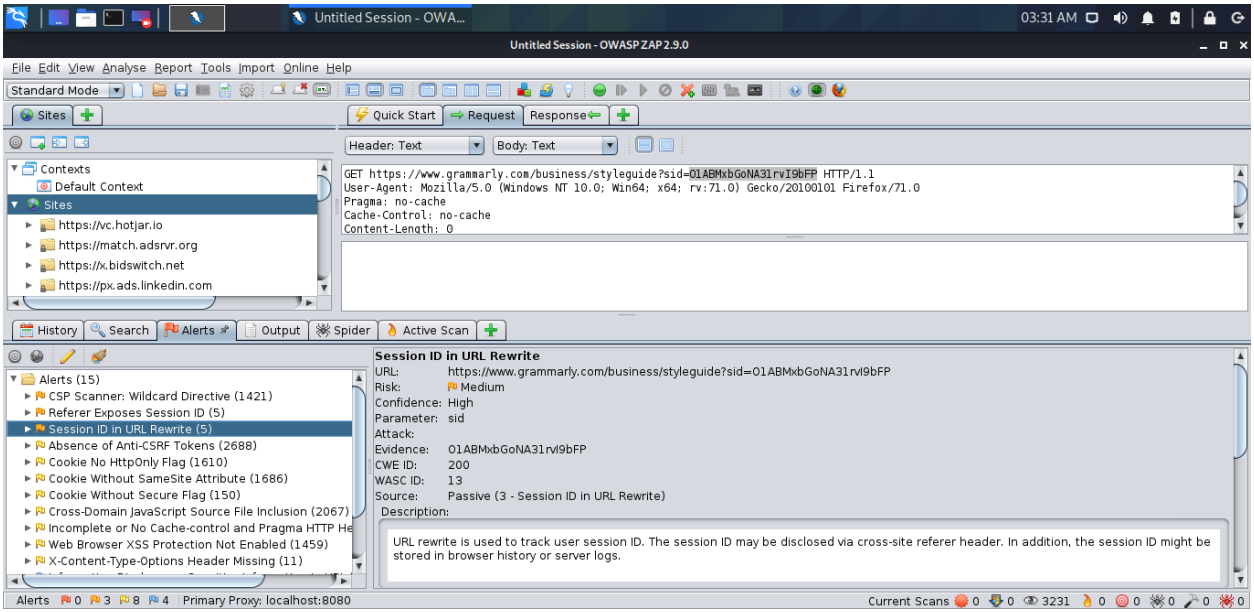


Here we can see that a hyperlink which was pointing to another host was revield and this might be disclosed in referer header to external hosts as shown above figure. This vulnerability is very sensitive when this session ID and the hyperlink refers to an external or third party host.

Solution:

In order to ensure security, session ID need to include in a secured session cookie.

3. Session ID in URL Rewrite (5)



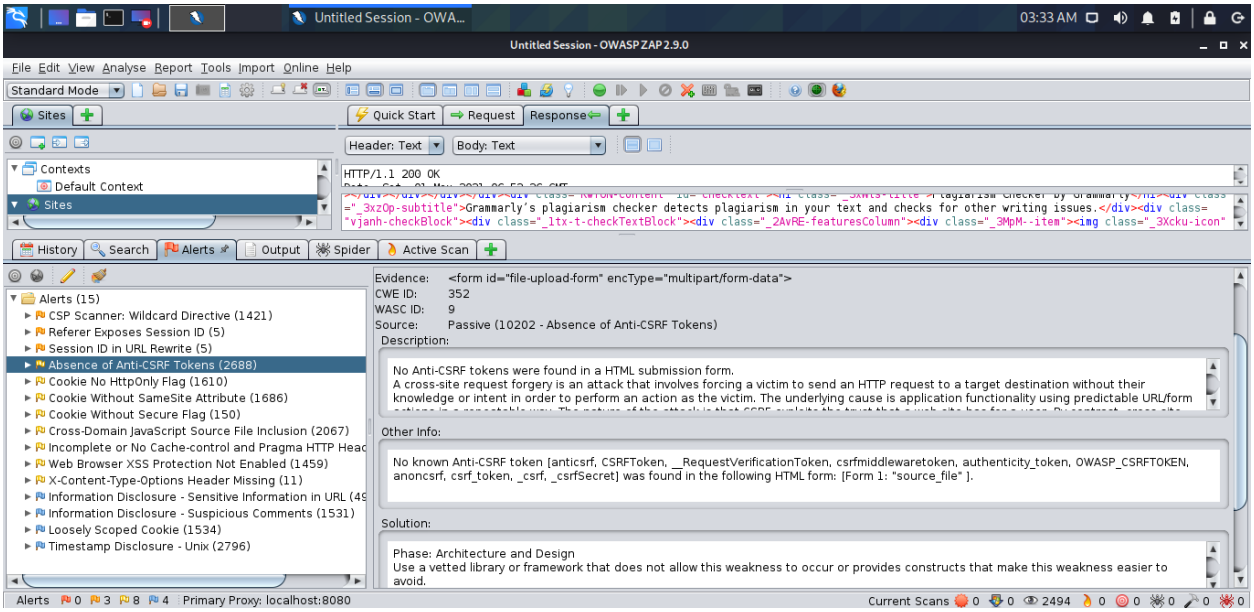
The user session ID is tracked using URL rewrite. Cross-site referer headers can reveal the session ID. The session ID can also be saved in the browser history or server logs.

Solution:

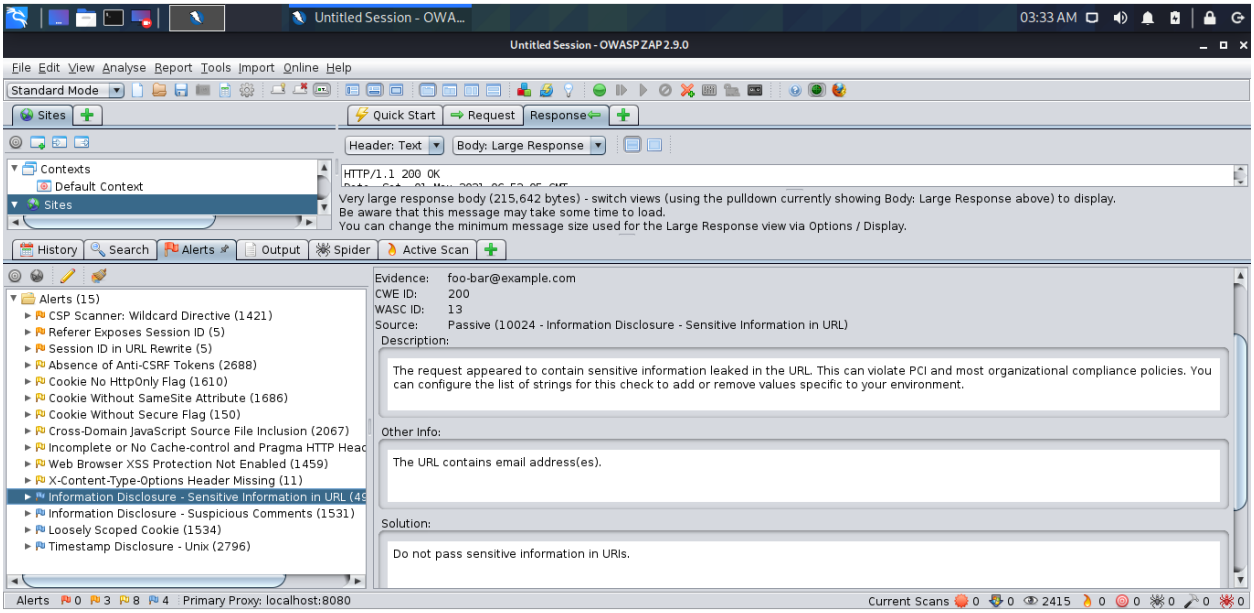
In order to ensure security, need to include the session ID in a cookie for protected material. Furthermore, site can be more safe with combination of cookie and URL rewrite.

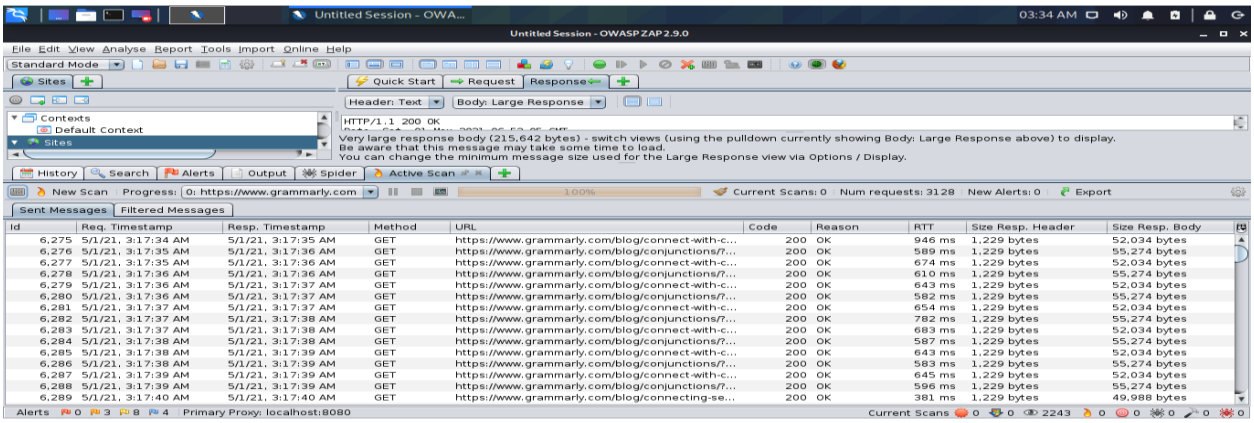
ii. Other low and informational priority vulnerabilities:

1. Absense of Anti-CRSF tokens (2688)



2. Information Disclosure - Sensitive Information in URL (491)





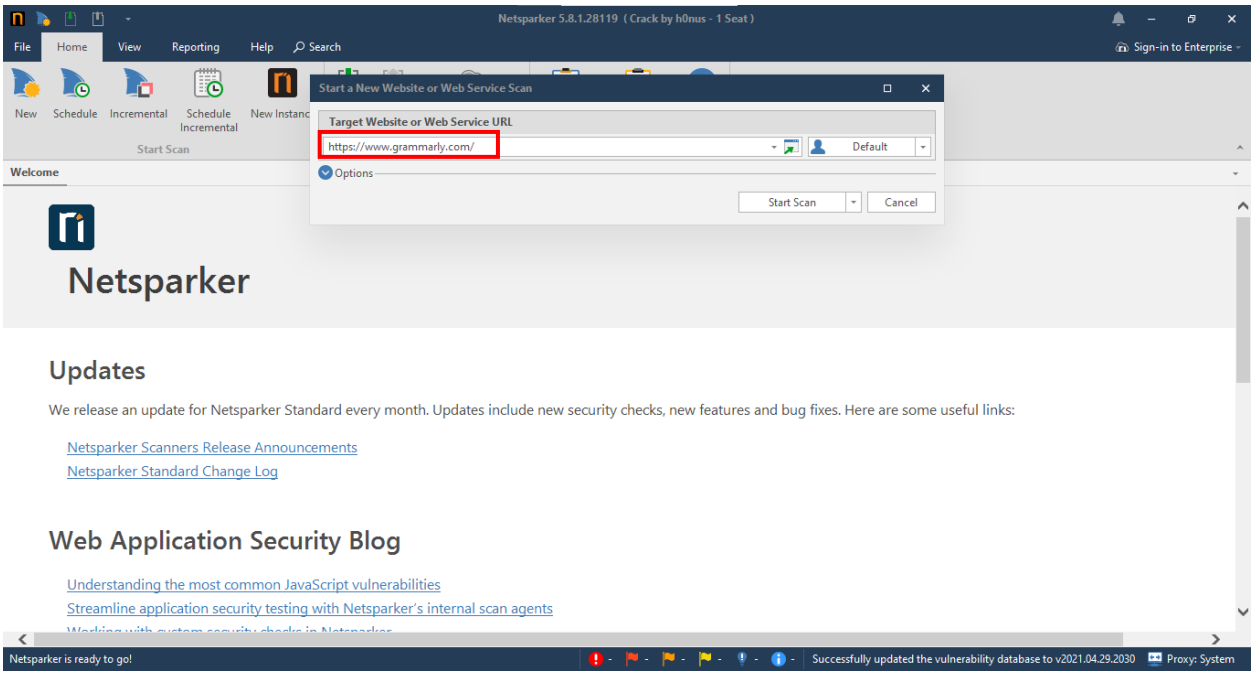
Finally, as I mentioned above, using this OWASP ZAP tool I was able to find 3 medium, 8 low and 4 informational priority vulnerabilities. In order to ensure the security, server should configure those headers and other solutions regarding those vulnerabilities securely.

V. Netsparker Professional

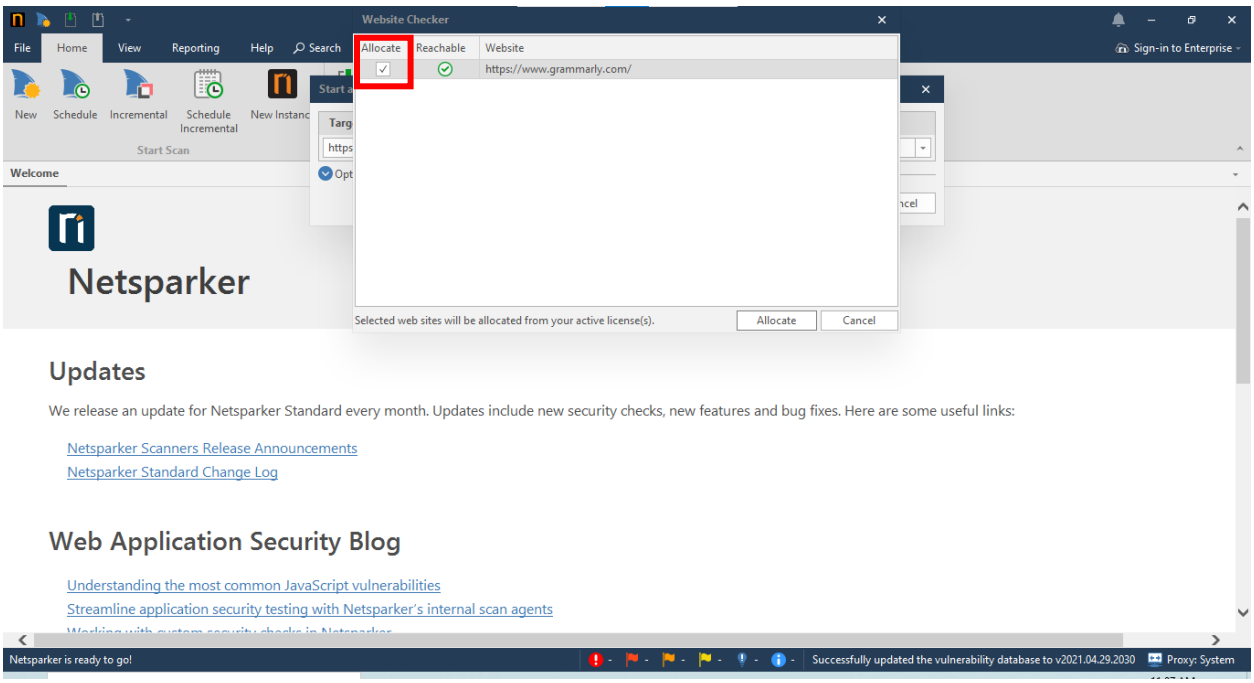
Netsparker is an Enterprise DAST (Dynamic Application Security Testing) tool that allows you to search websites, web applications, and web services for security vulnerabilities in an automatic and completely configurable manner. Netsparker is extremely extensible since it can search any kind of web app, regardless of the medium or language used to create it. Netsparker uses "Proof-Based Scanning," which instantly verifies discovered bugs and determines if they are false positives by exploiting them in a secure, read-only fashion.

I used Netsparker Professional Version to scan my target domain and I was able to get detailed report and more vulnerabilities than the other tools. I installed in on my Windows environment, and it is quite easy to work on there.

After installed it and starts it, first we need to enter our target website or web service URL in the given pop-up box as shown below.

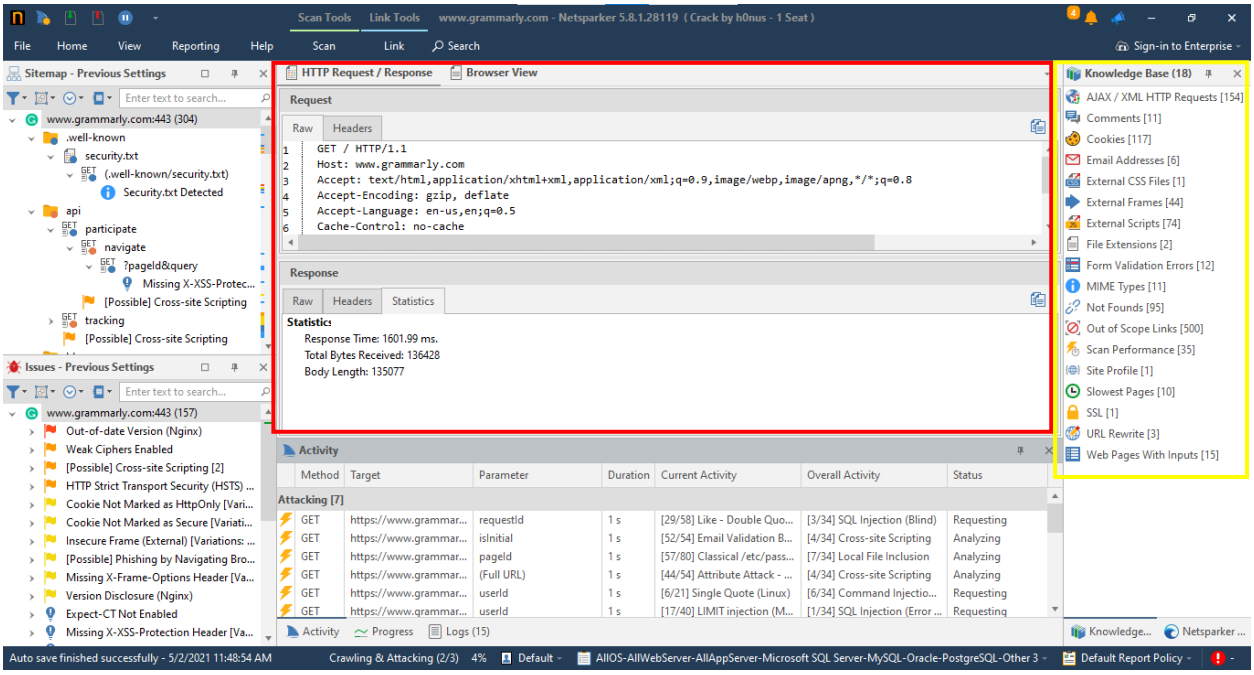


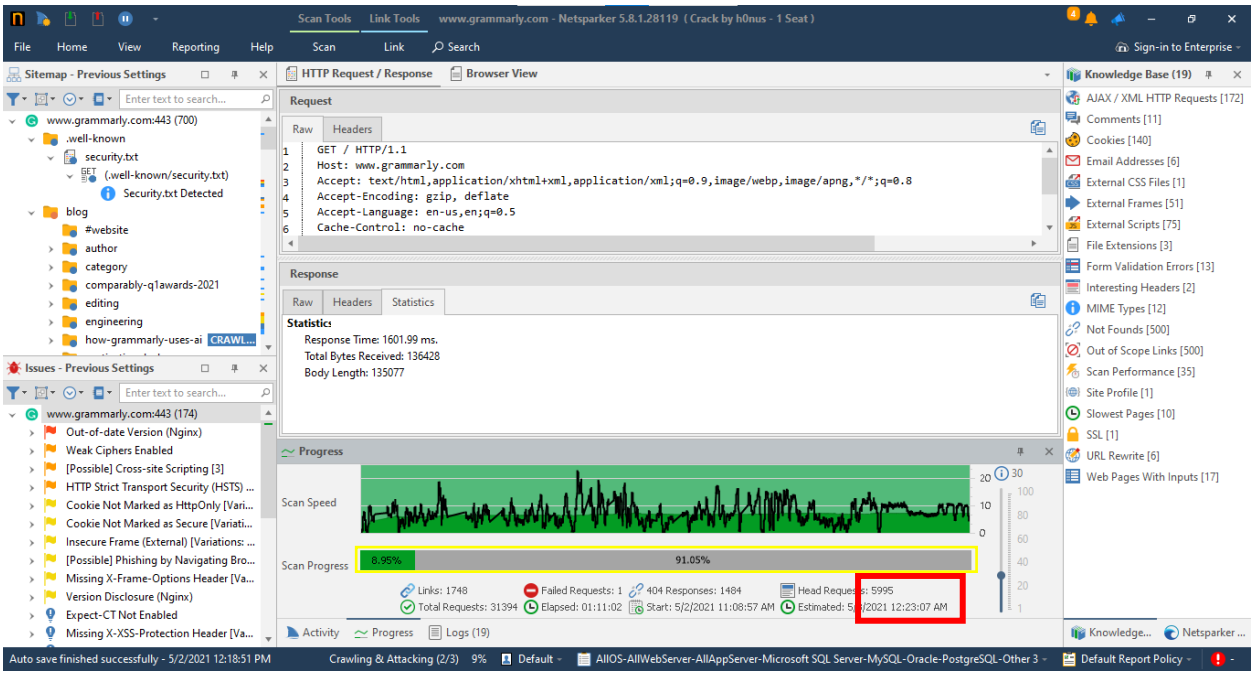
Then it will be asked to allocate that target webpage under my active license, and we need to allocate it by simply ticking the check box.



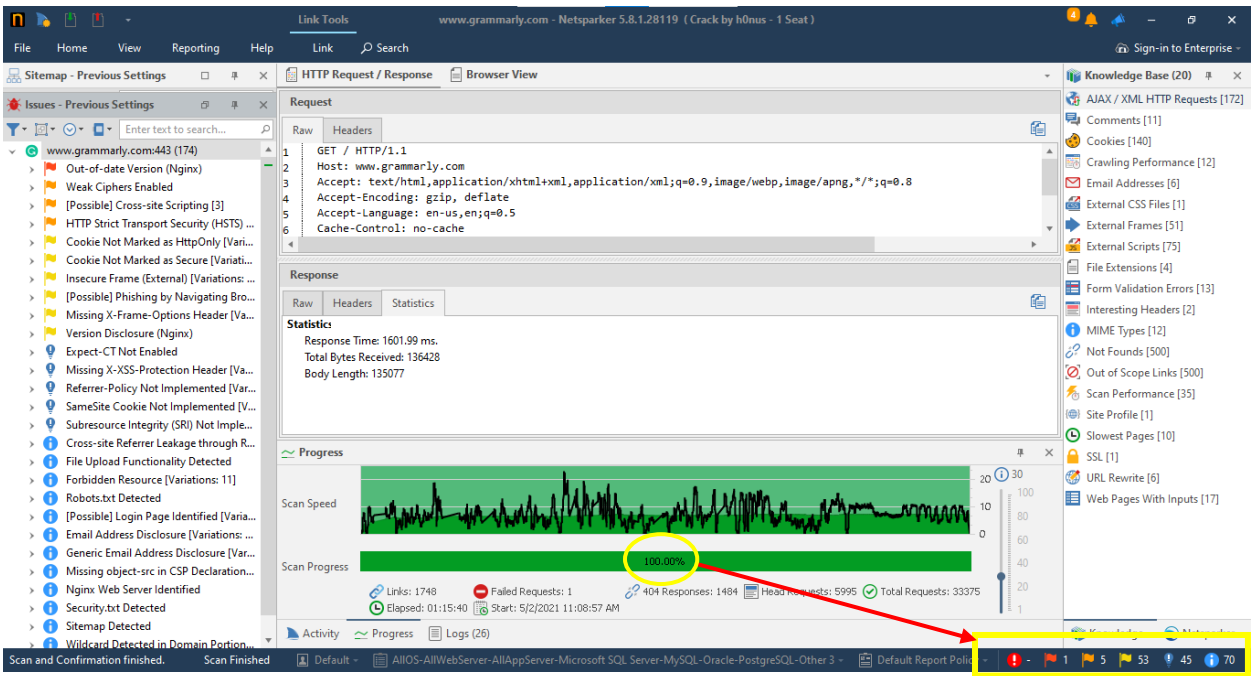
Then it will start to scan my target domain. While it was scanning, we were able to separately examine each discovered vulnerability. This tool gave us the vulnerability details, impacts, actions to take, remedy and external references for each of specific identified vulnerabilities. Basically, this tool has different types of boxes such as sitemap where browsed pages are being added to as you browse them, issues – previous settings, activity, progress logs, knowledge base and etc.

As shown below figures, we were able to find the request and response which were send and received. All the identified vulnerabilities were listed done on a small column in the issues field. In right side of the scanner, we could find all the number of identified disclosure details such as emails.

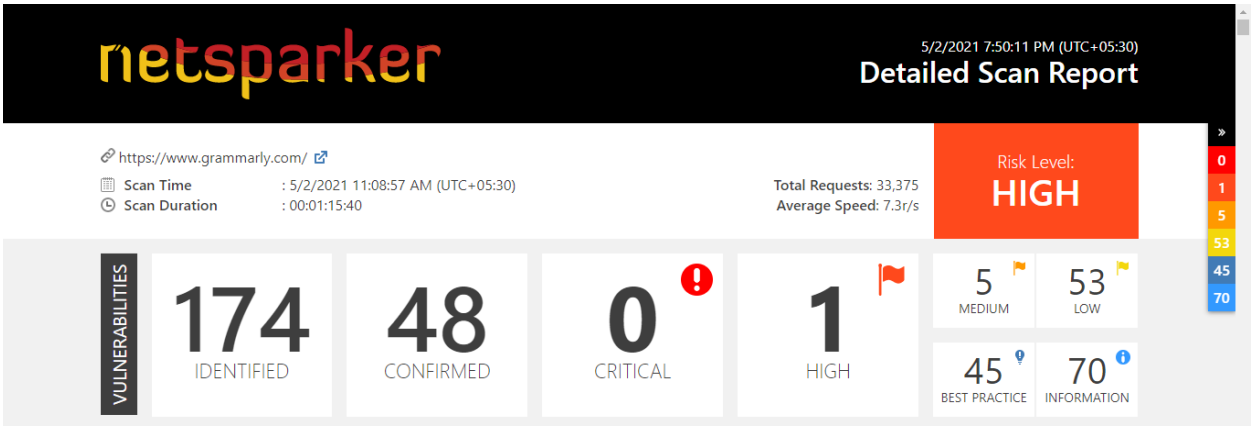


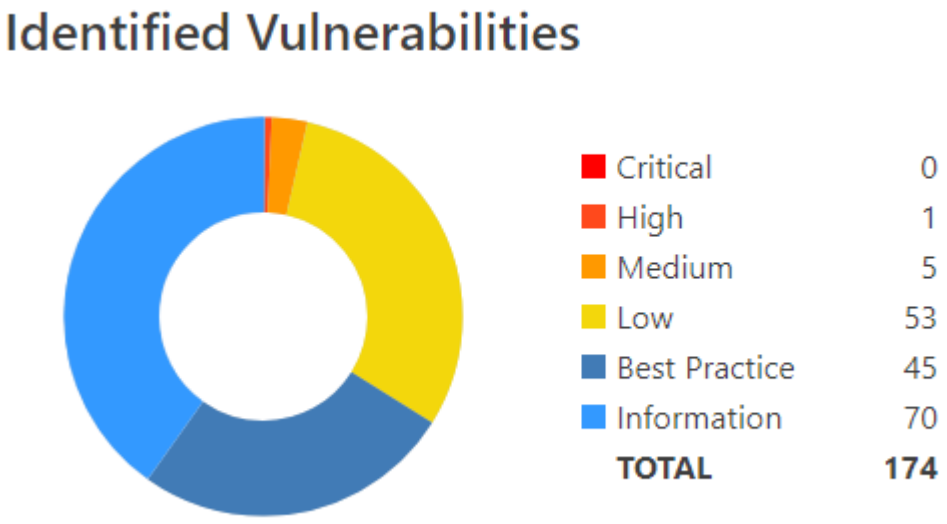


As shown in the above figure, it took so much time to finish the scanning process and at last we were able to generate a html and pdf formatted report.

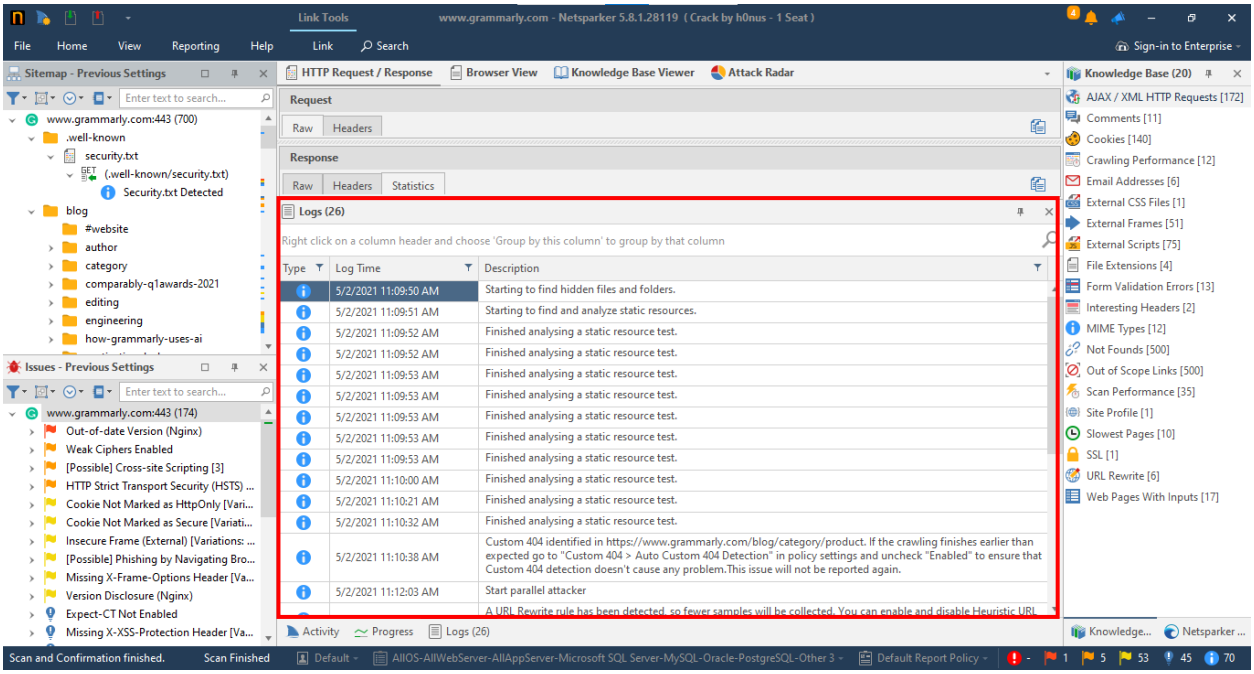


Finally, scan was end up with the identifying zero number of critical vulnerabilities, only one high vulnerability, 5 medium vulnerabilities, 53 number of low vulnerabilities, 45 number of best practices and 70 informational vulnerabilities within the total number of 33375 requests including 5995 head requests, 1485 number of 404 requests and only one failed request. And identified number of links were 1748 as shown below.

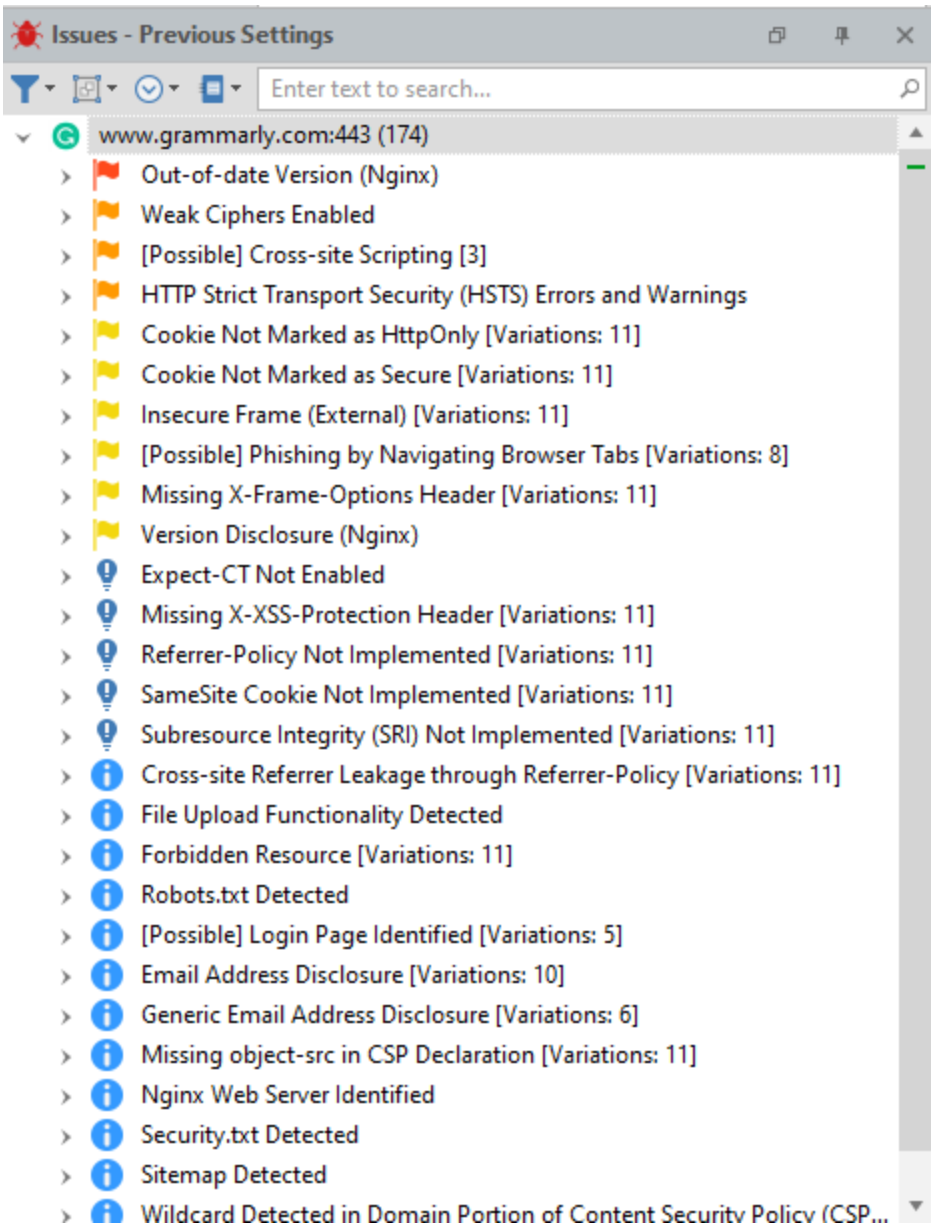




The below figure shows the log details regarding the scan.

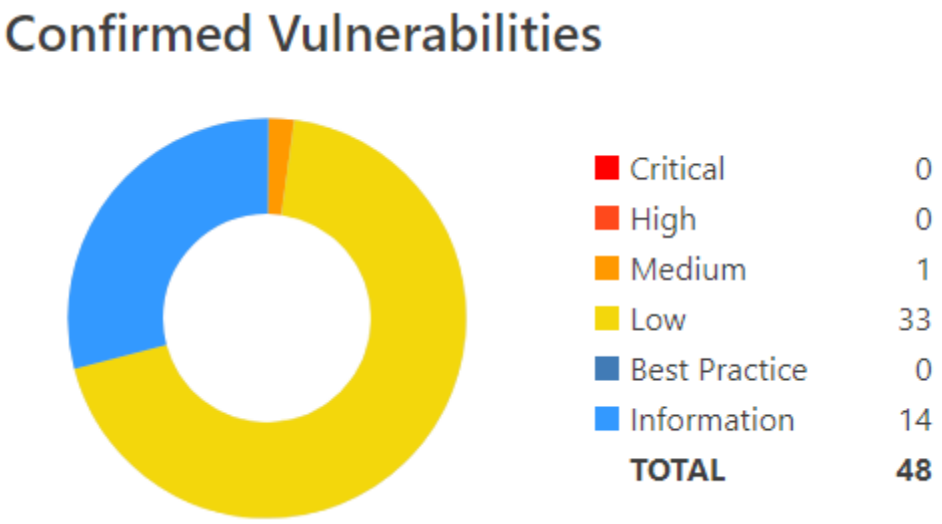


The below figure shows the name of the identified vulnerabilities with the number of counts which were identified in our scope.



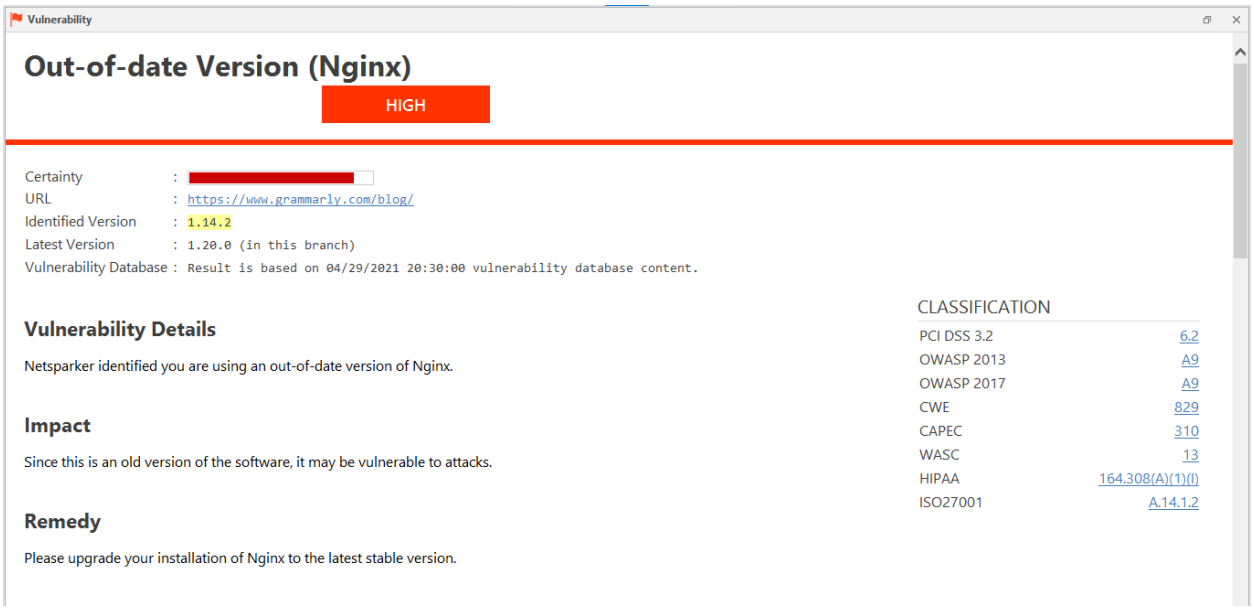
Even though total number of identified vulnerabilities were 174, confirmed number of vulnerabilities within above identified vulnerabilities were only 48. This Netsparker Professional version confirmed identified vulnerabilities by automatically exploiting them in a safe and read only manner.

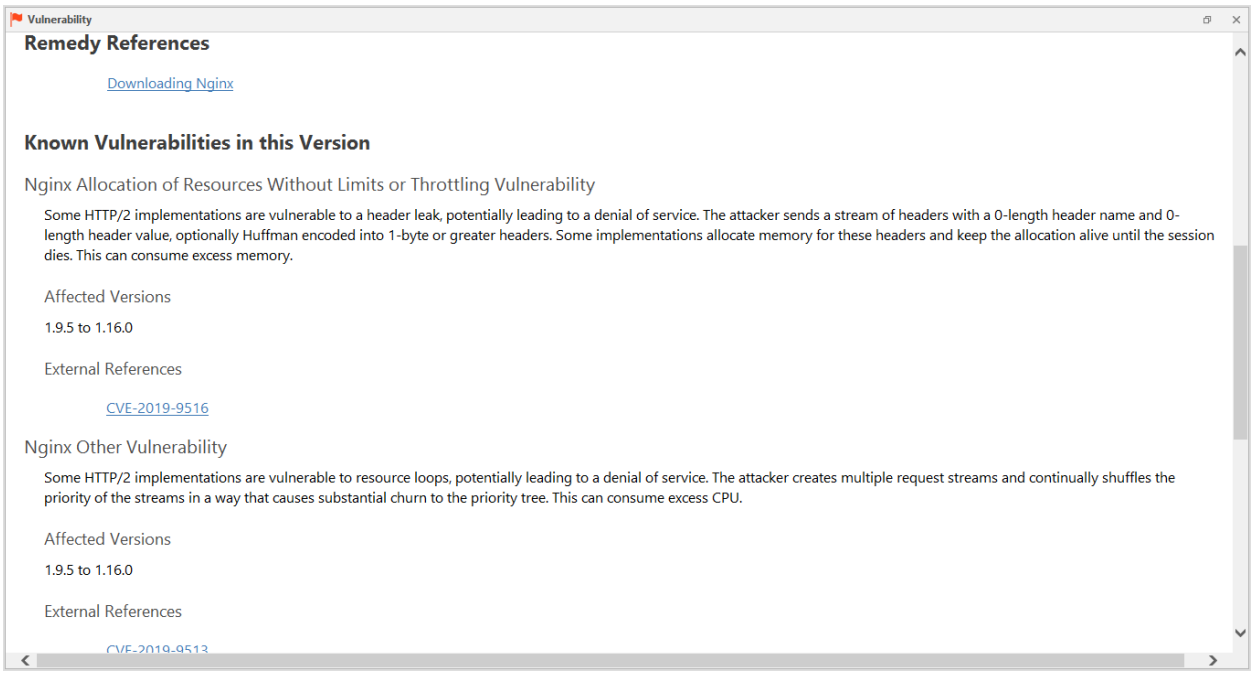
Confirmed Vulnerabilities with critical level shown below.



i. Identified Vulnerabilities

1. Vulnerability: Out-of-date Version (Nginx)
Critical Level: High

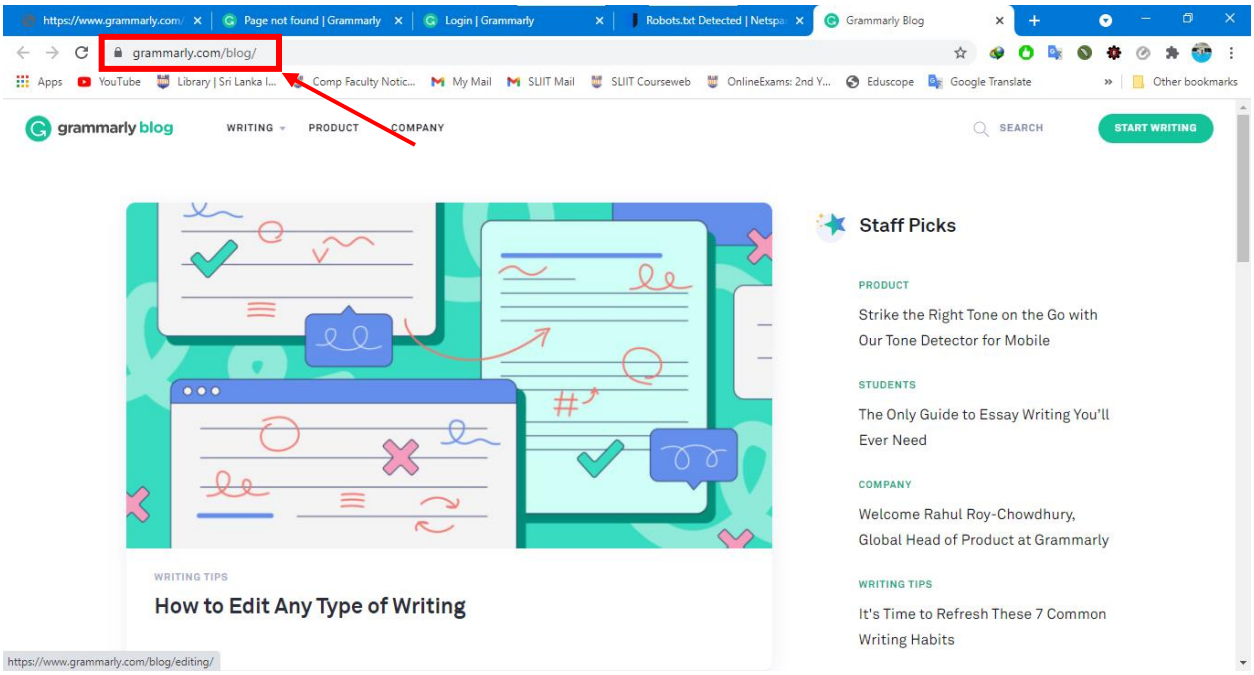




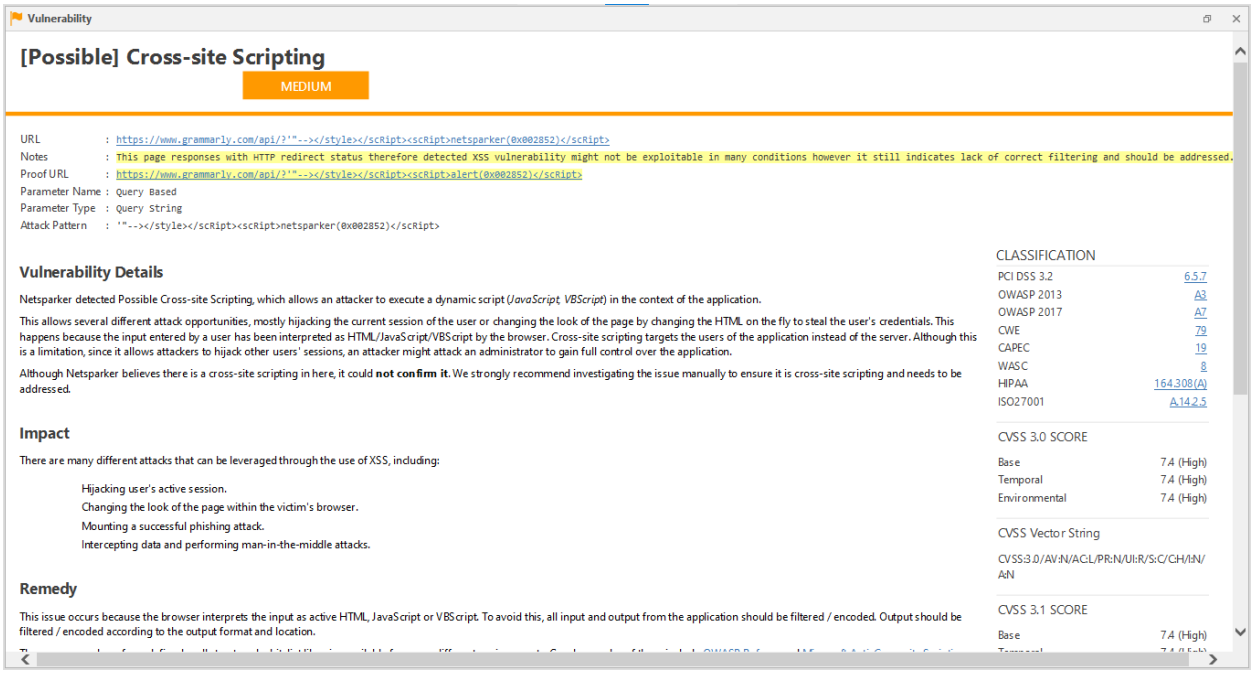
As above figures shown, using Netsparker scan I was able to identify that the webserver in blog sub directory was in outdated and this is vulnerable to attacks. As it shows, in order to prevent from those attacks, need to upgrade Nginx into latest stable version of version - 1.20.0. According to the Web Application Security Consortium (WASC) threat classification this can be consider as a high priority vulnerability and Nginx upgradable link also given through that report.

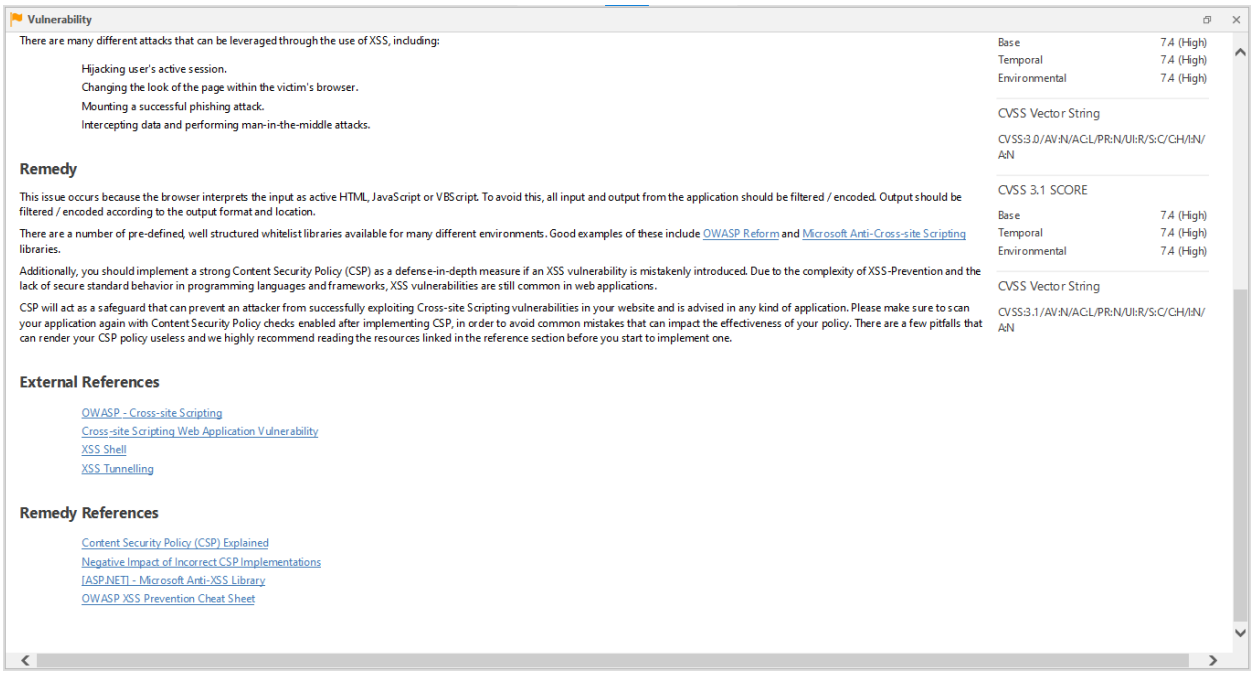
Upgradable link: **https://nginx.org/en/download.html**

Sub-directory of the identified webpage is shown below.



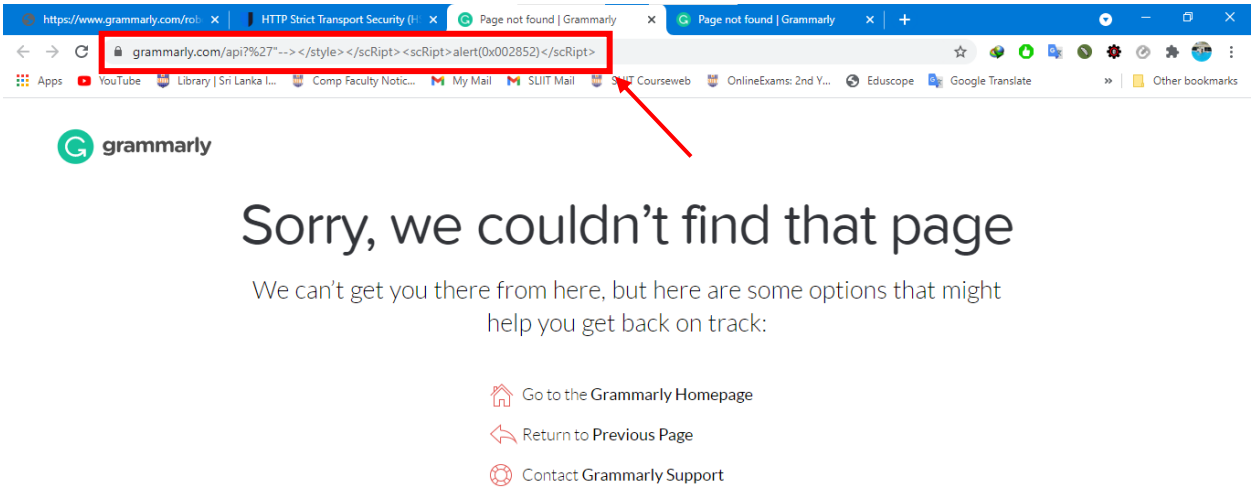
2. Vulnerability: [Possible] Cross-site Scripting
Critical Level: Medium



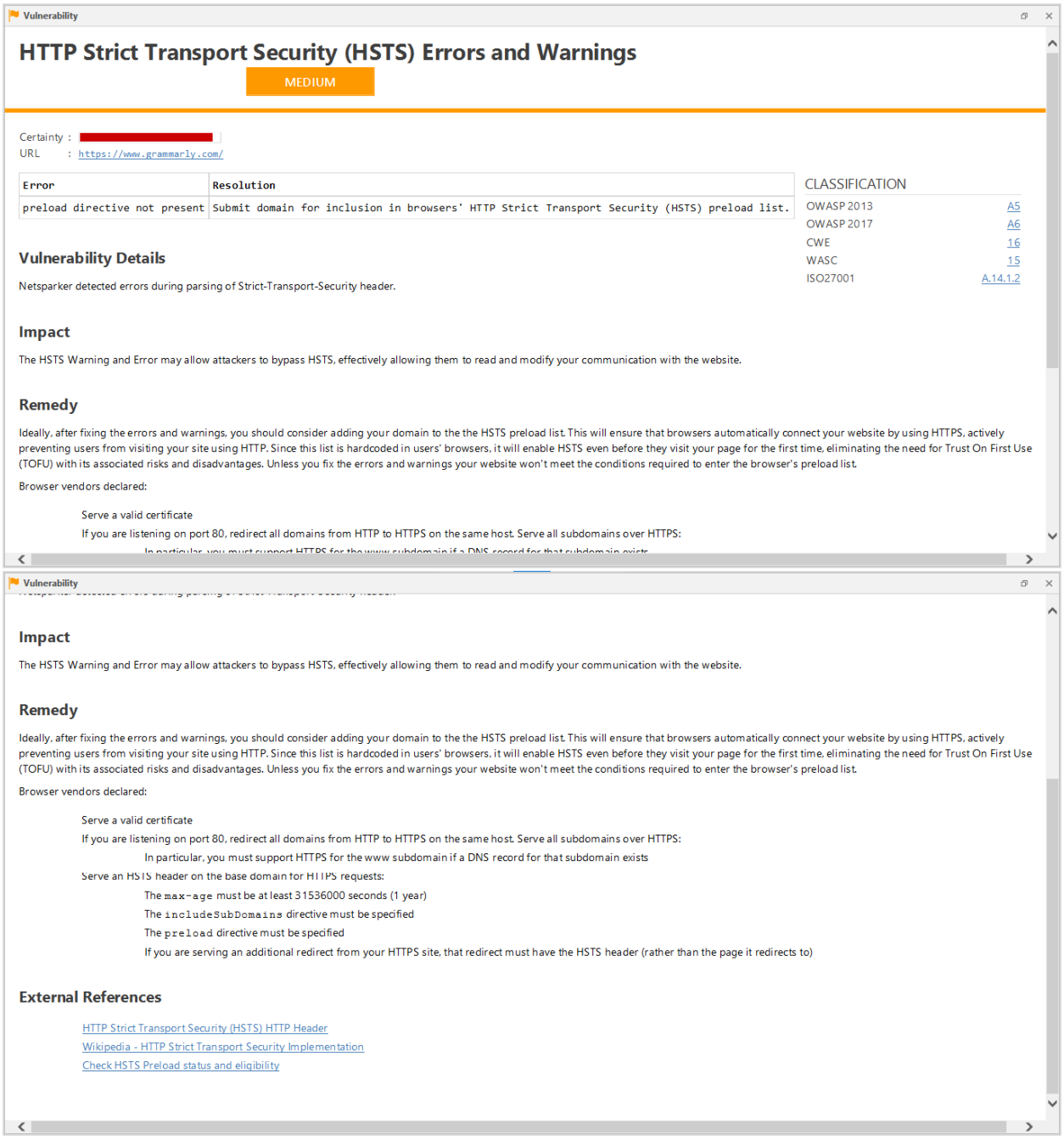


Cross-Site Scripting is a very serious vulnerability which allows the attacker to perform different types of attacks such as hijacking, phishing and man-in-the-middle (MIMT) attacks. This issue gains the 7.4 CVSS 3.0 score and implementing strong Content Security Policy (CSP) can be able to mitigate these types of attacks.

Displayed webpage of proof URL shown below.



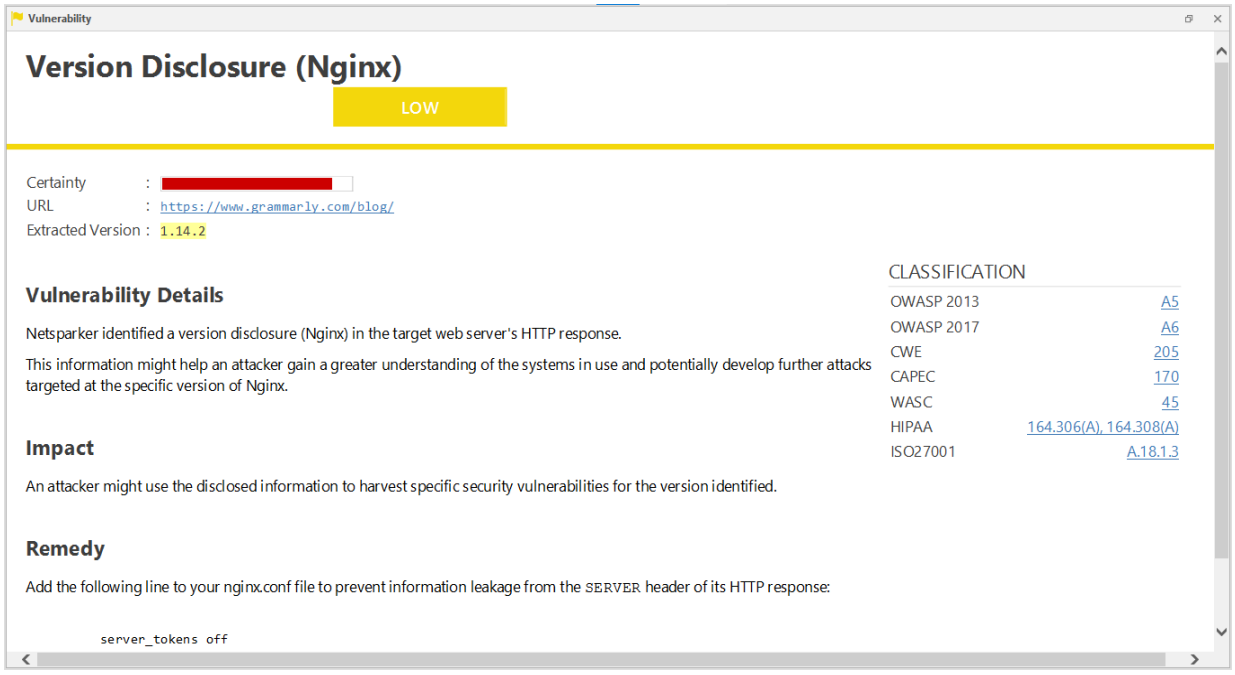
3. **Vulnerability: [HTTP Strict Transport Security (HSTS) Errors and Warnings]**
   Critical Level: Medium



As errors were detected during the parsing of Strict-Transport-Security header, attackers are able to read and change your communication with the website simply by bypassing HSTS header. In order to remediate this vulnerability, domain should

add to the HSTS preloaded list and then the user's browsers can connect to the webpage using HTTPS request automatically.

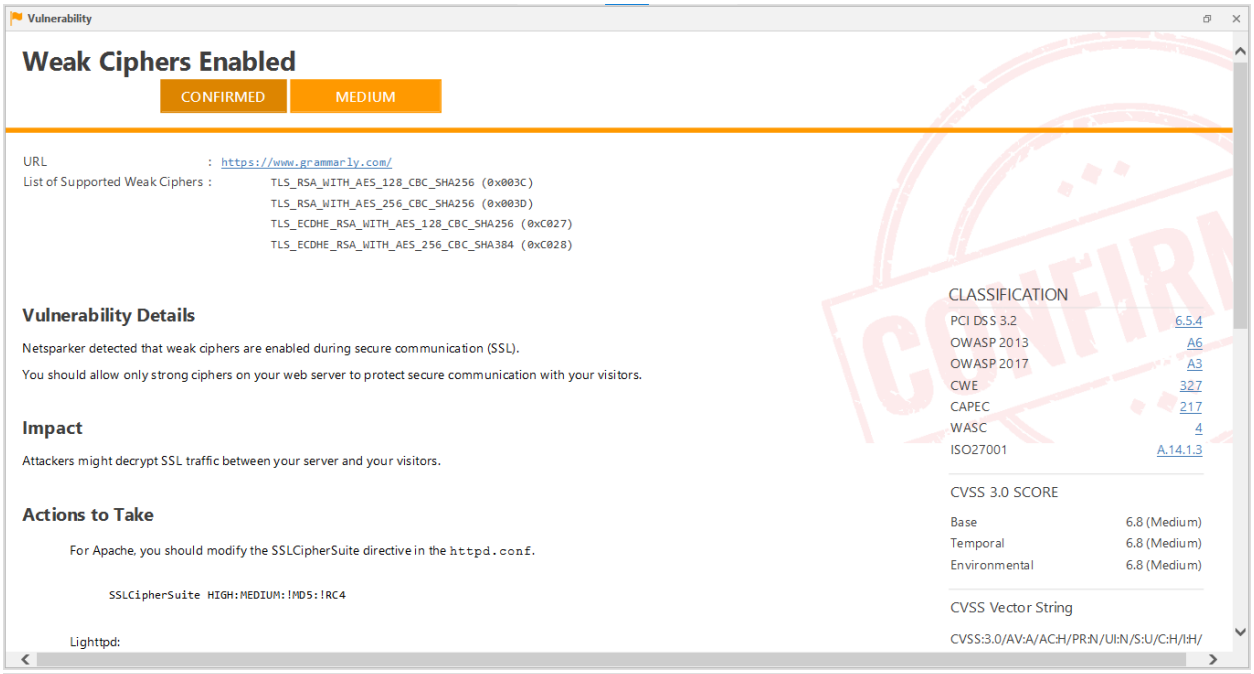
4. **Vulnerability: Version Disclosure (Nginx)**
   Critical Level: Low

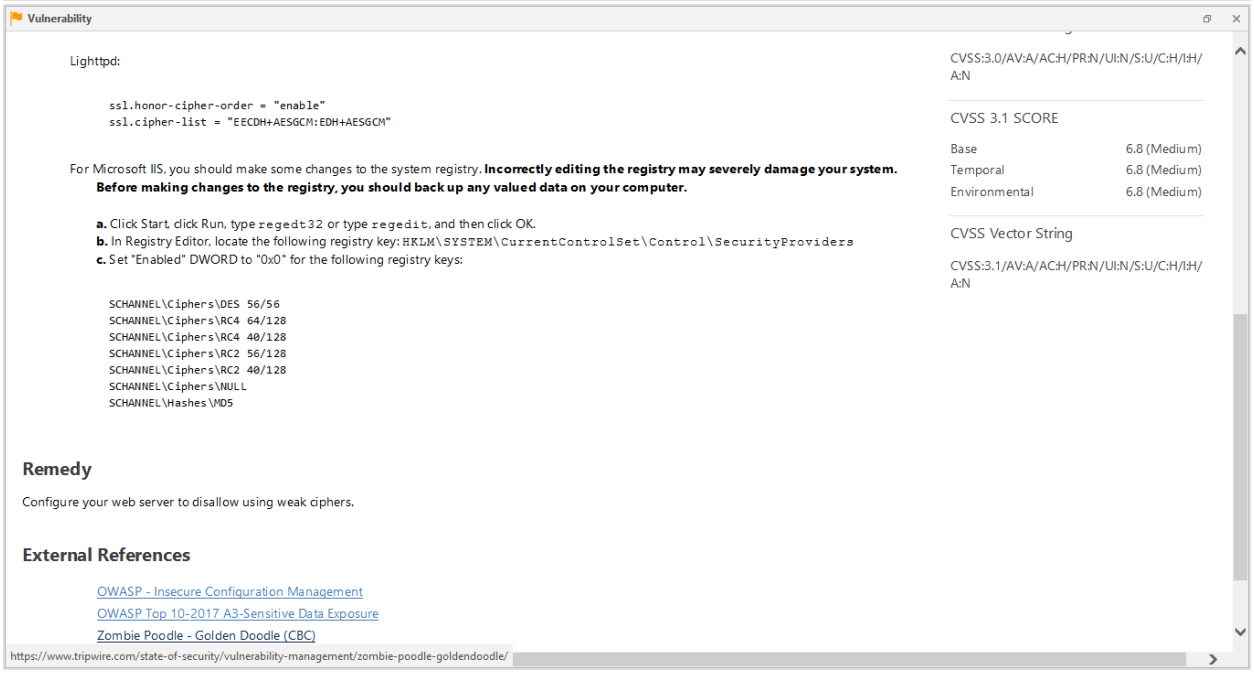


This vulnerability was regarding the previously mentioned critical vulnerability of outdated Nginx webserver. Allowing attackers to identifying the version in the target web server's HTTP response, attacker can get deep understand about the target system and able to perform specific exploits for the identified version. Simply adding **"server_tokens off"** in the webserver's configuration file, able to remediate from those type of information disclosure.

ii. Confirmed Vulnerabilities

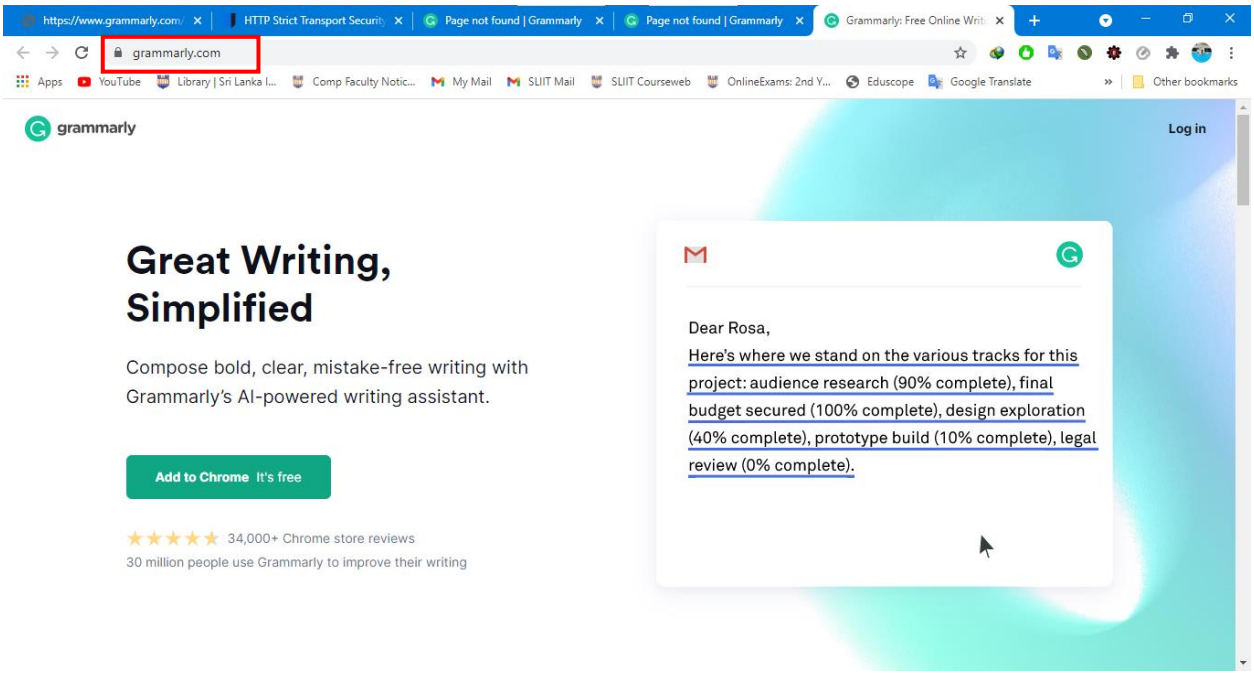
1. Vulnerability: Weak Ciphers Enabled
Critical Level: Medium



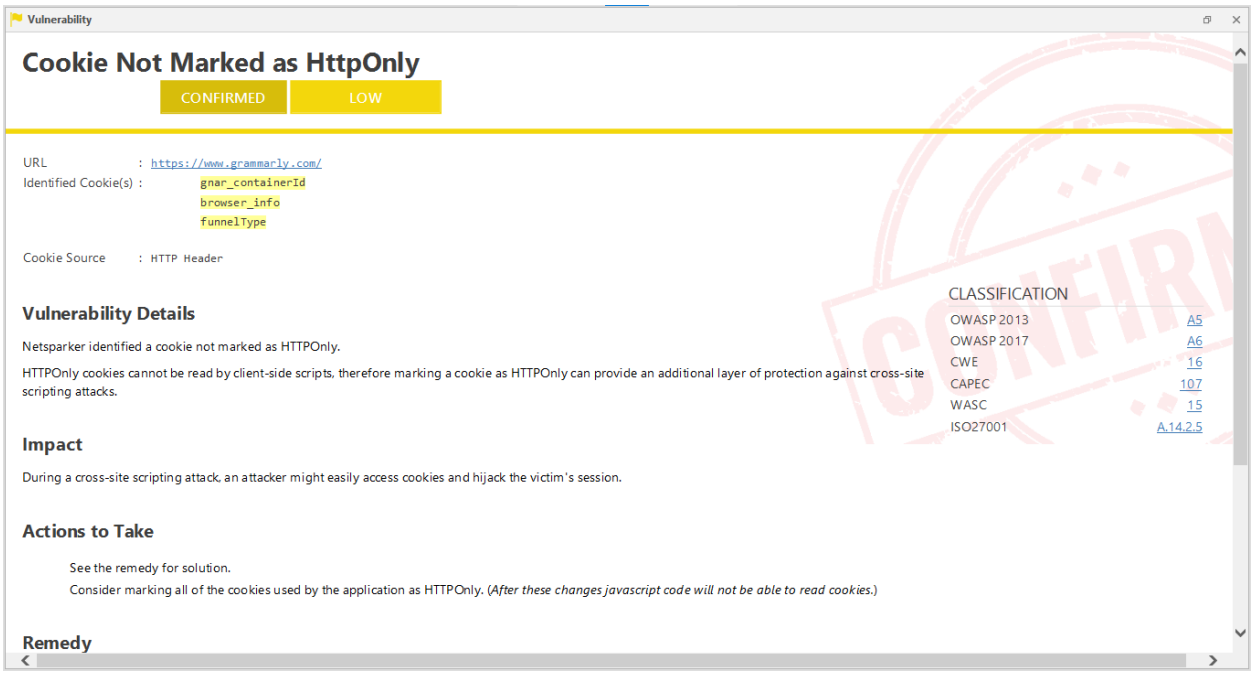


This vulnerability takes 6.8 base score from the CVSS 3.0 classification and this was a confirmed vulnerability by the Netsparker. This vulnerability allows attackers decrypt the traffic between the server and webpage visitors. In order to prevent from those attacks, need to specify only the strong ciphers in the web server.

Below figure show the vulnerable webpage with weak cipher enabled and that was the target domain of our scan.



2. Vulnerability: Cookie Not Marked as HttpOnly
Critical Level: Low



Vulnerability

Cookie Not Marked as HttpOnly

CONFIRMED LOW

URL : https://www.grammarly.com/
Identified Cookie(s) : gnar_containerId
browser_info
funnelType

Cookie Source : HTTP Header

CLASSIFICATION

OWASP 2013	A5
OWASP 2017	A6
CWE	16
CAPEC	107
WASC	15
ISO27001	A.14.2.5

Vulnerability Details

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.

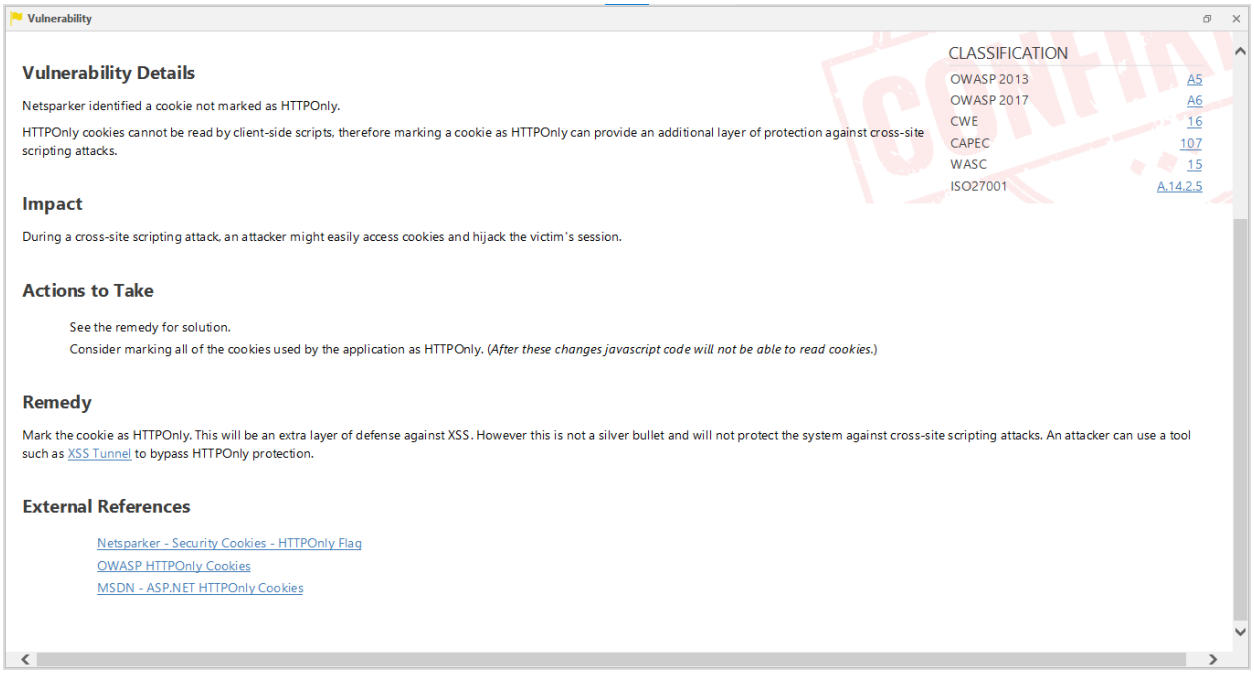
Impact

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

Actions to Take

See the remedy for solution.
Consider marking all of the cookies used by the application as HTTPOnly. (*After these changes javascript code will not be able to read cookies.*)

Remedy



Vulnerability

CLASSIFICATION

OWASP 2013	A5
OWASP 2017	A6
CWE	16
CAPEC	107
WASC	15
ISO27001	A.14.2.5

Vulnerability Details

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.

Actions to Take

See the remedy for solution.
Consider marking all of the cookies used by the application as HTTPOnly. (*After these changes javascript code will not be able to read cookies.*)

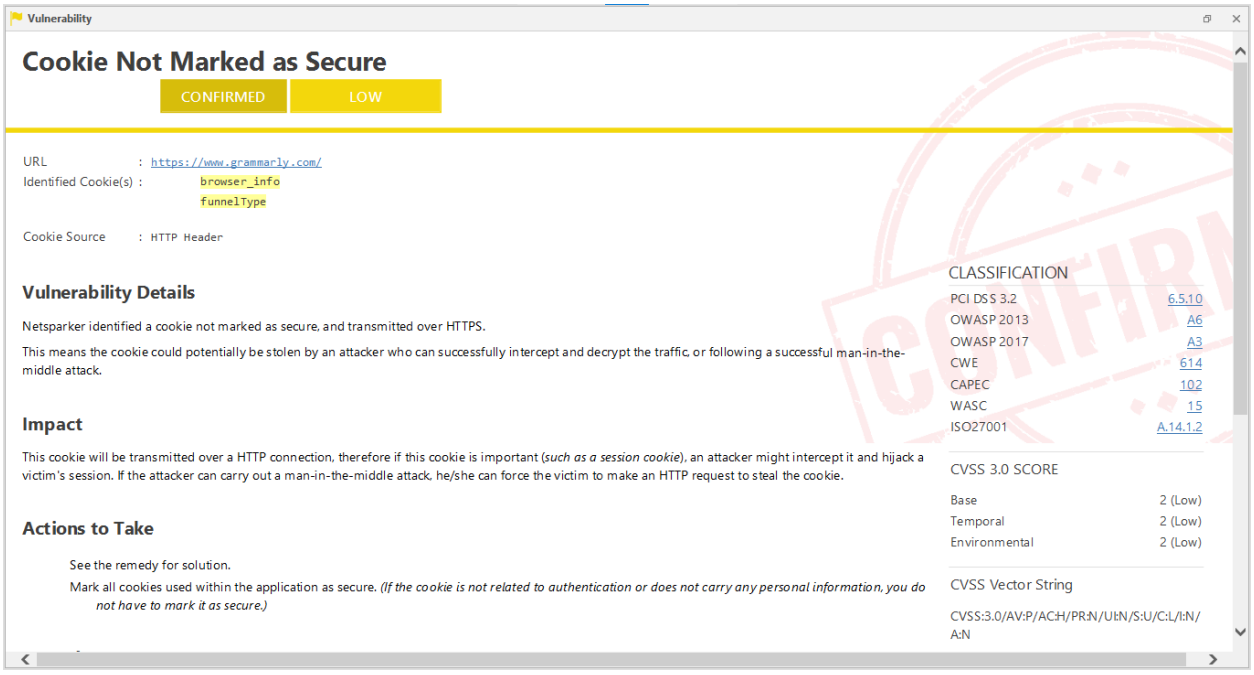
Remedy

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.

External References

Netsparker - Security Cookies - HTTPOnly Flag
OWASP HTTPOnly Cookies
MSDN - ASP.NET HTTPOnly Cookies

3. Vulnerability: Cookie Not Marked as Secure
Critical Level: Low



Vulnerability

Cookie Not Marked as Secure

CONFIRMED LOW

URL : https://www.grammarly.com/
Identified Cookie(s) : browser_info
funnelType

Cookie Source : HTTP Header

Vulnerability Details

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.

Actions to Take

See the remedy for solution.

Mark all cookies used within the application as secure. *(If the cookie is not related to authentication or does not carry any personal information, you do not have to mark it as secure.)*

CLASSIFICATION

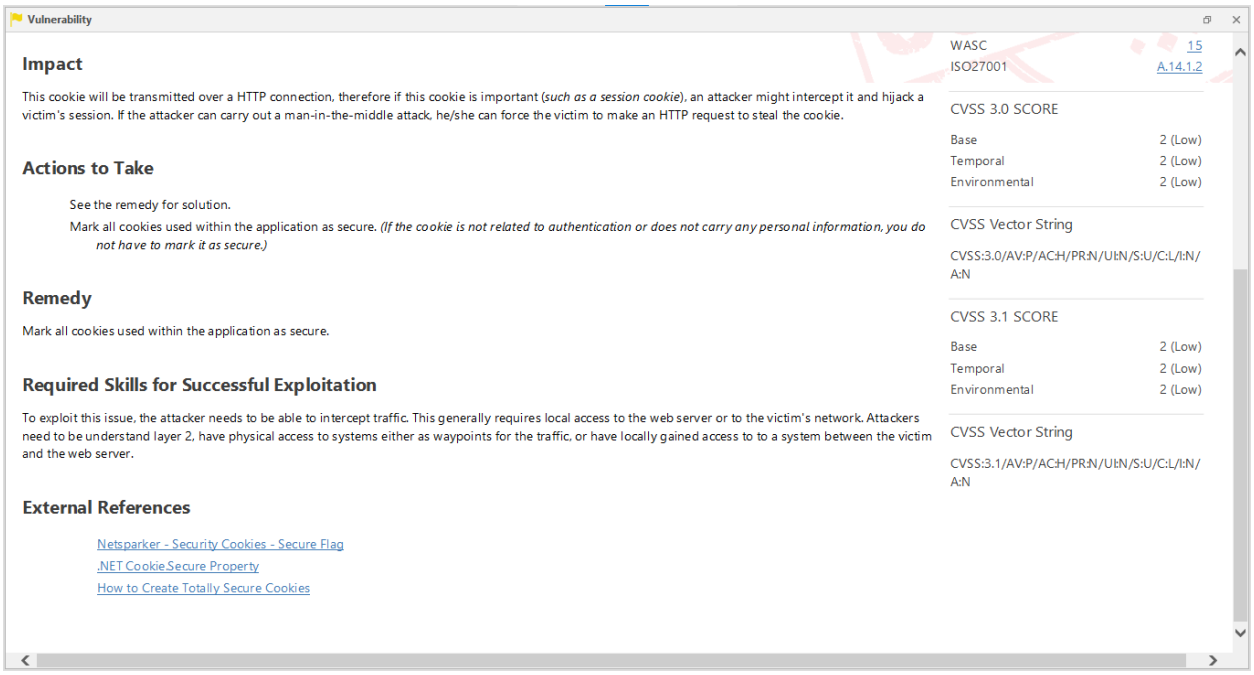
PCI DSS 3.2	6.5.10
OWASP 2013	A6
OWASP 2017	A3
CWE	614
CAPEC	102
WASC	15
ISO27001	A.14.1.2

CVSS 3.0 SCORE

Base	2 (Low)
Temporal	2 (Low)
Environmental	2 (Low)

CVSS Vector String

CVSS:3.0/AV:P/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N



Vulnerability

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.

Actions to Take

See the remedy for solution.

Mark all cookies used within the application as secure. *(If the cookie is not related to authentication or does not carry any personal information, you do not have to mark it as secure.)*

Remedy

Mark all cookies used within the application as secure.

Required Skills for Successful Exploitation

To exploit this issue, the attacker needs to be able to intercept traffic. This generally requires local access to the web server or to the victim's network. Attackers need to be understand layer 2, have physical access to systems either as waypoints for the traffic, or have locally gained access to to a system between the victim and the web server.

External References

Netsparker - Security Cookies - Secure Flag
.NET Cookie.Secure Property
How to Create Totally Secure Cookies

WASC 15
ISO27001 A.14.1.2

CVSS 3.0 SCORE

Base	2 (Low)
Temporal	2 (Low)
Environmental	2 (Low)

CVSS Vector String

CVSS:3.0/AV:P/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N

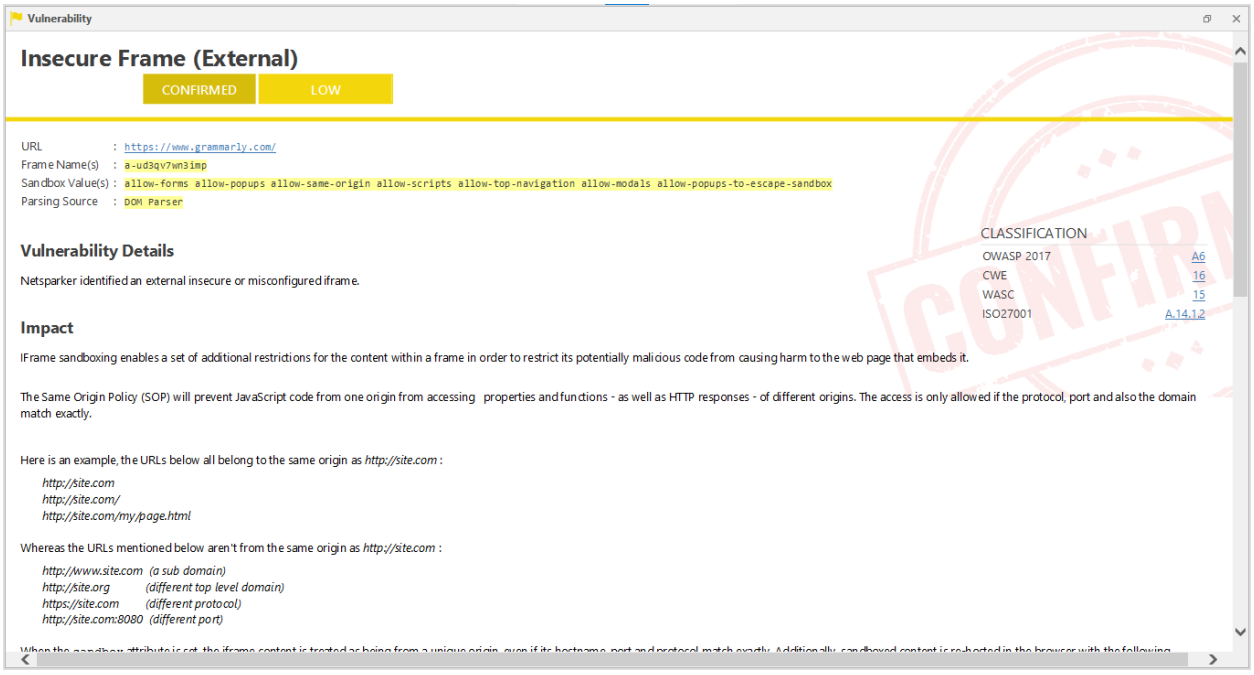
CVSS 3.1 SCORE

Base	2 (Low)
Temporal	2 (Low)
Environmental	2 (Low)

CVSS Vector String

CVSS:3.1/AV:P/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N

4. Vulnerability: Insecure Frame (External)
Critical Level: Low



Vulnerability

Insecure Frame (External)

CONFIRMED LOW

URL : https://www.grammarly.com/
Frame Name(s) : a-ud3qv7wn3imp
Sandbox Value(s) : allow-forms allow-popups allow-same-origin allow-scripts allow-top-navigation allow-modals allow-popups-to-escape-sandbox
Parsing Source : DOM Parser

CLASSIFICATION

OWASP 2017	A6
CWE	16
WASC	15
ISO27001	A.14.1.2

Vulnerability Details

Netsparker identified an external insecure or misconfigured iframe.

Impact

IFrame sandboxing enables a set of additional restrictions for the content within a frame in order to restrict its potentially malicious code from causing harm to the web page that embeds it.

The Same Origin Policy (SOP) will prevent JavaScript code from one origin from accessing properties and functions - as well as HTTP responses - of different origins. The access is only allowed if the protocol, port and also the domain match exactly.

Here is an example, the URLs below all belong to the same origin as *http://site.com* :

http://site.com
http://site.com/
http://site.com/my/page.html

Whereas the URLs mentioned below aren't from the same origin as *http://site.com* :

http://www.site.com (a sub domain)
http://site.org (different top level domain)
https://site.com (different protocol)
http://site.com:8080 (different port)

When the sandbox attribute is set, the iframe content is treated as being from a unique origin, even if its hostname, port and protocol match exactly. Additionally, sandboxed content is re-hosted in the browser with the following restrictions:

Any kind of plugin, such as ActiveX, Flash, or Silverlight will be disabled for the iframe.
Forms are disabled. The hosted content is not allowed to make forms post back to any target.
Scripts are disabled. JavaScript is disabled and will not execute.
Links to other browsing contexts are disabled. An anchor tag targeting different browser levels will not execute.
Unique origin treatment. All content is treated under a unique origin. The content is not able to traverse the DOM or read cookie information.

When the `sandbox` attribute is not set or not configured correctly, your application might be at risk.

A compromised website that is loaded in such an insecure iframe might affect the parent web application. These are just a few examples of how such an insecure frame might affect its parent:

It might trick the user into supplying a username and password to the site loaded inside the iframe.
It might navigate the parent window to a phishing page.
It might execute untrusted code.
It could show a popup, appearing to come from the parent site.

`Sandbox` containing a value of :

`allow-same-origin` will not treat it as a unique origin.
`allow-top-navigation` will allow code in the iframe to navigate the parent somewhere else, e.g. by changing parent.location.
`allow-forms` will allow form submissions from inside the iframe.
`allow-popups` will allow popups.
`allow-scripts` will allow malicious script execution however it won't allow to create popups.

Remedy

Apply sandboxing in inline frame

```
<iframe sandbox src="framed-page-url"></iframe>
```

Remedy

Apply sandboxing in inline frame

```
<iframe sandbox src="framed-page-url"></iframe>
```

For untrusted content, avoid the usage of `seamless` attribute and `allow-top-navigation`, `allow-popups` and `allow-scripts` in sandbox attribute.

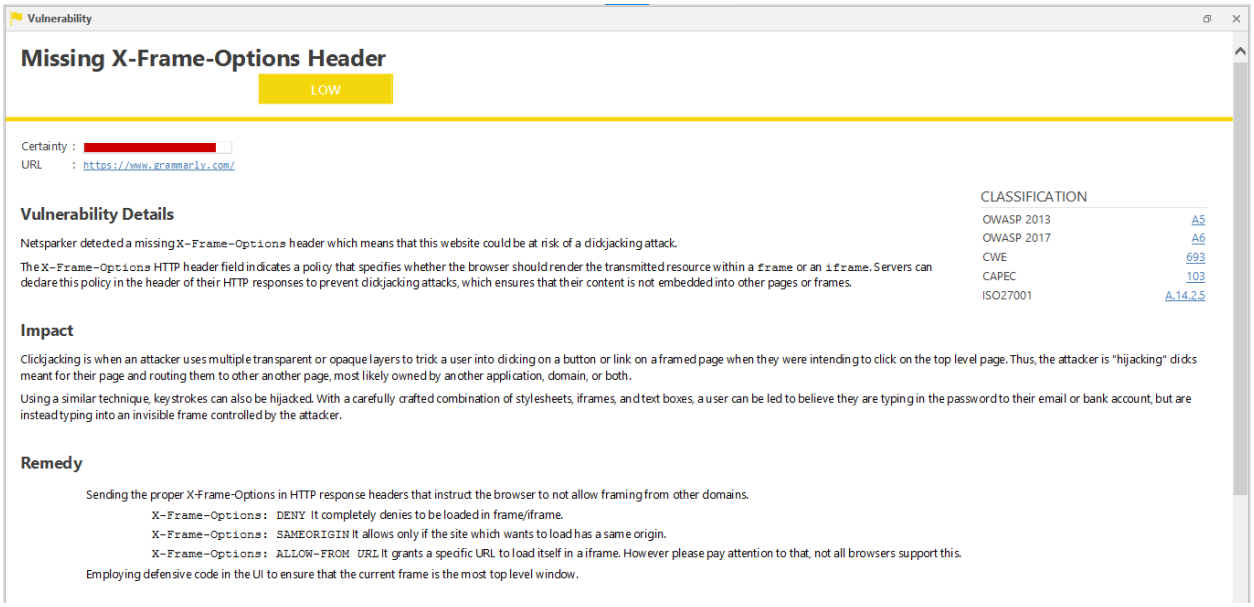
External References

HTML5 Security Cheat Sheet

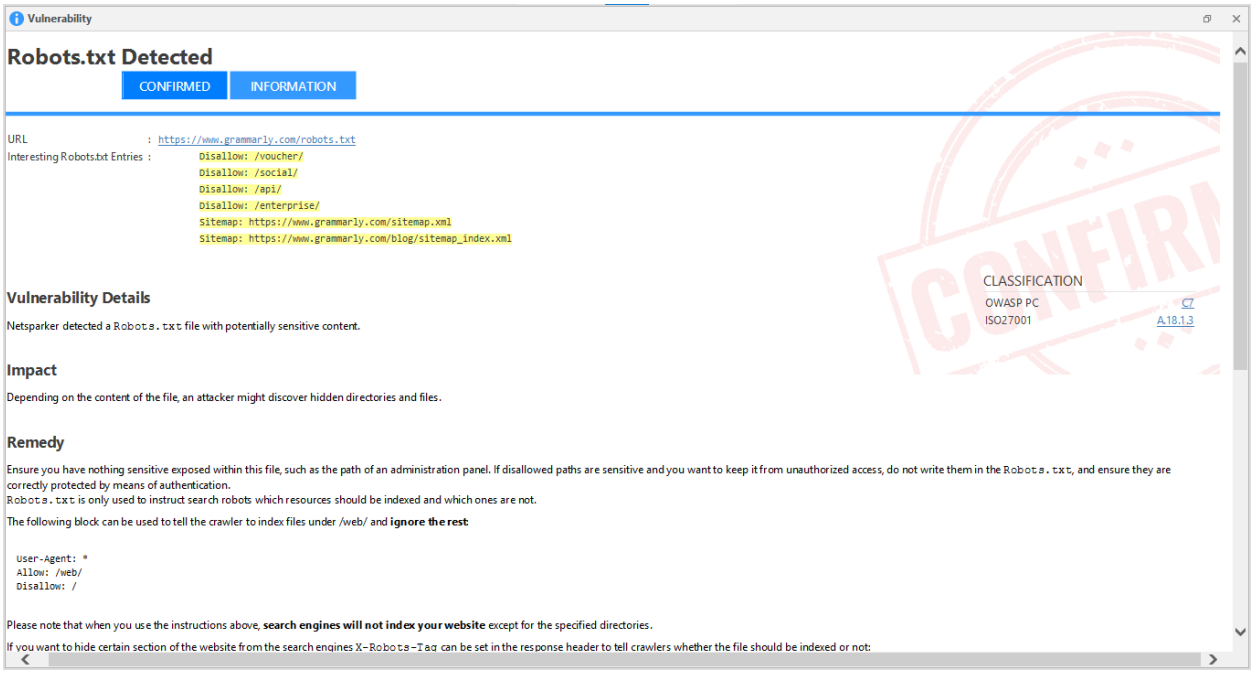
Remedy References

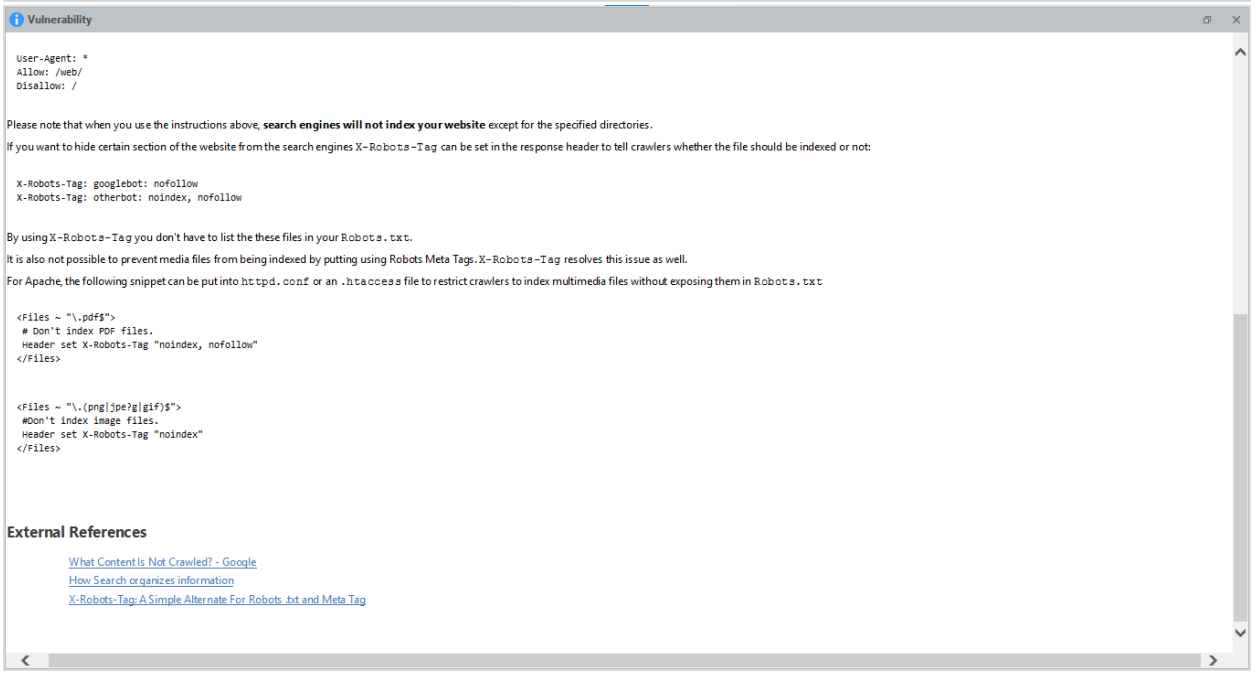
How to Safeguard your Site with HTML5 Sandbox
Play safely in sandboxed IFrames

5. Vulnerability: Missing X-Frame-Options Header
Critical Level: Low



6. Vulnerability: Robots.txt Detected
Critical Level: Information



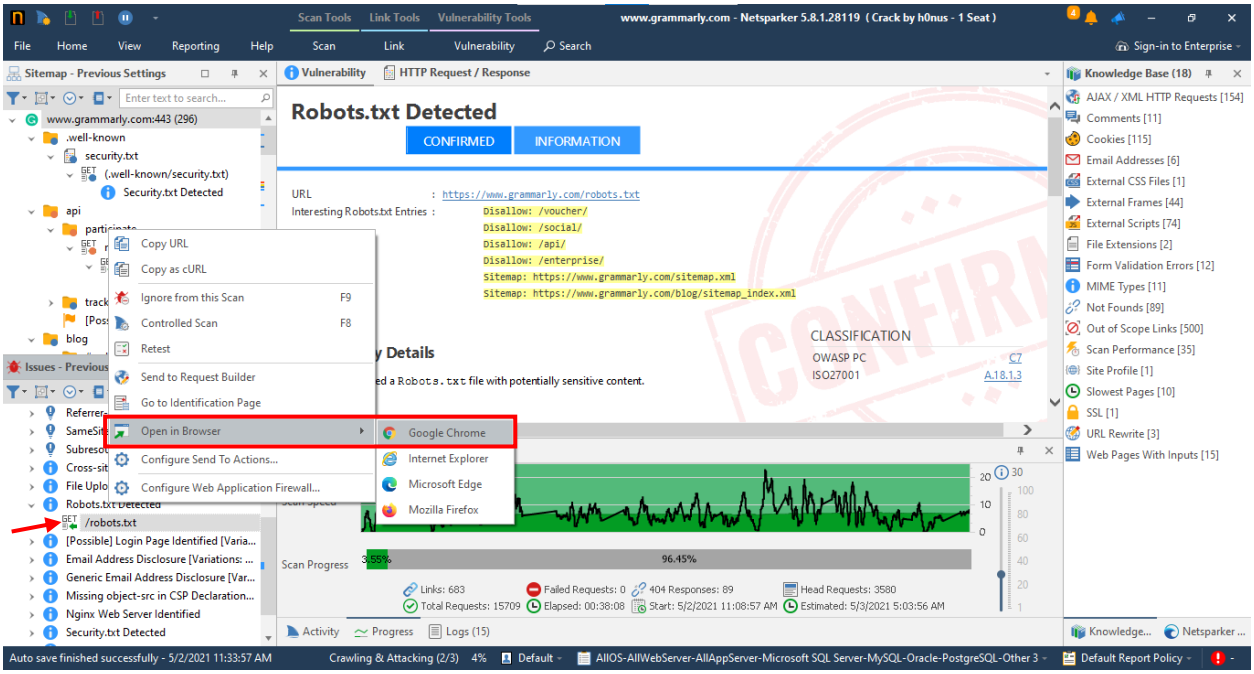


The file robots.txt is used to inform web robots, such as search engine crawlers, where they are able to crawl and index, and where they are not allowed to crawl and index. The robots.txt file is not a security risk in and of itself, and its proper use can be considered good practice for non-security purposes. It is, however,

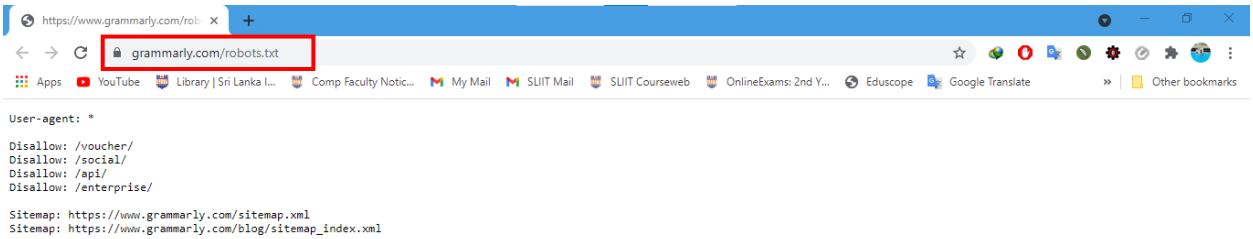
sometimes used to designate limited or private areas of a website's content. As a result, the information in the file might aid an attacker in mapping out the contents of the site, particularly if any of the locations found are not connected from somewhere else on the site. This is a major vulnerability if the program depends on robots.txt to secure access to these areas and does not apply sufficient access control. In order to prevent from those vulnerabilities, no need to include any important paths in this file, such as an administration panel's route. Those disallowed paths can be shield from unwanted entry by not including in Robots.txt file and properly securing with the authentication.

Revealed information from the Robots.txt are shown below.

Robot.txt file URL: **https://www.grammarly.com/robots.txt**



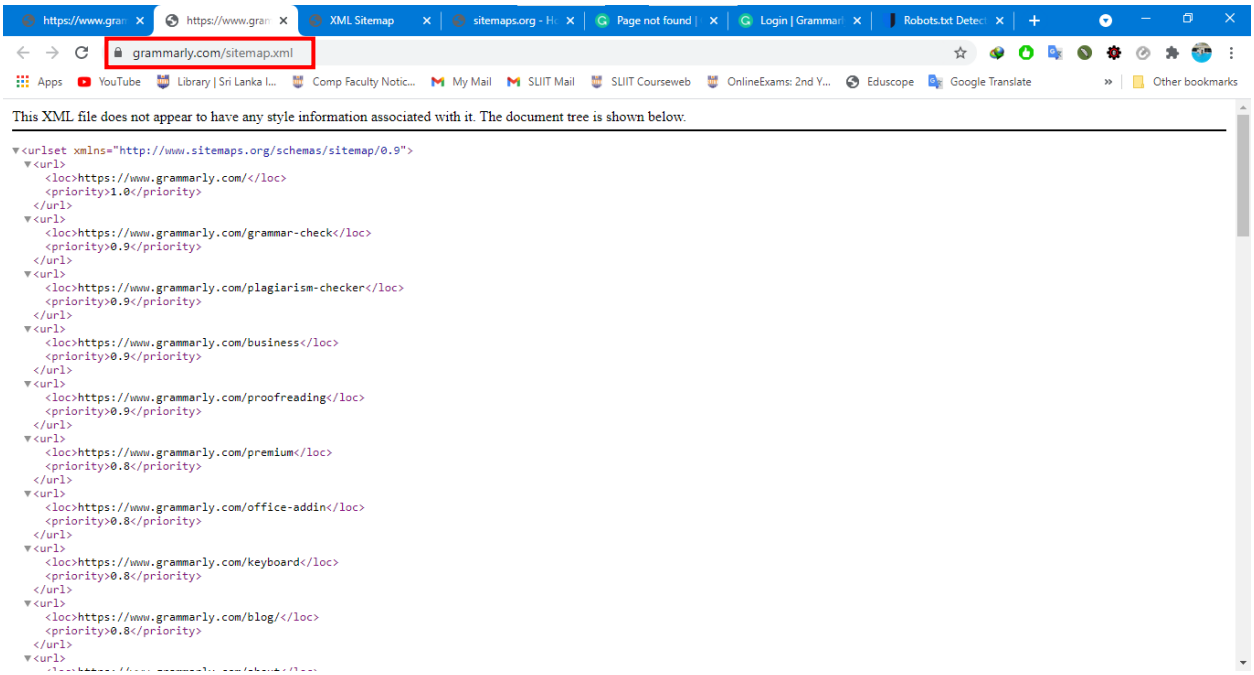
I was able to find some interesting information from that Robot.txt file.

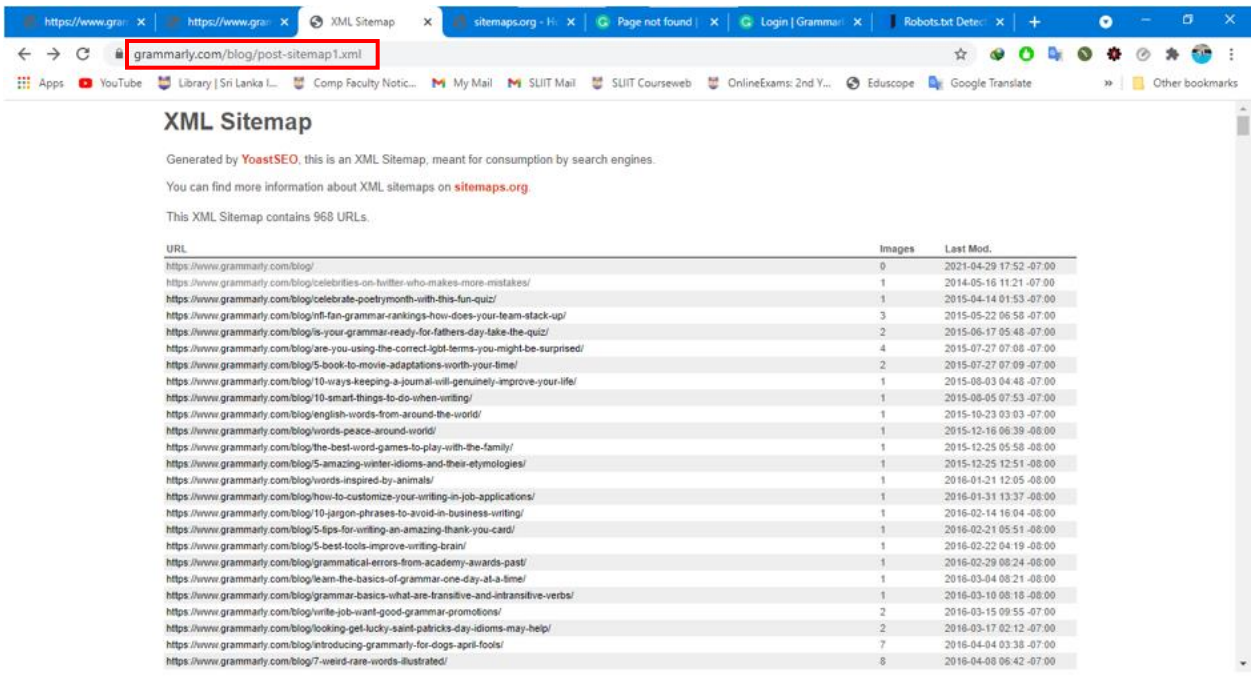


a. Sitemaps:

Sitemaps are a simple way for webmasters to let search engines know which pages on their site are crawlable. Simply a Sitemap is an XML file that lists URLs for a site along with additional information regarding each URL (when it was last changed, how much it normally varies, and how relevant it is in relation to other URLs in the site) so that search engines can crawl the site more intelligently. Crawlers find pages by following connections both inside the web and to other sites. Crawlers that help Sitemaps can pick up all URLs in the Sitemap and learn about them using the related metadata, thanks to Sitemaps. The Sitemap protocol does not ensure that web pages will be used in search engines, but it does include information for web crawlers to help them crawl your content more efficiently.

Site map information are shown below.



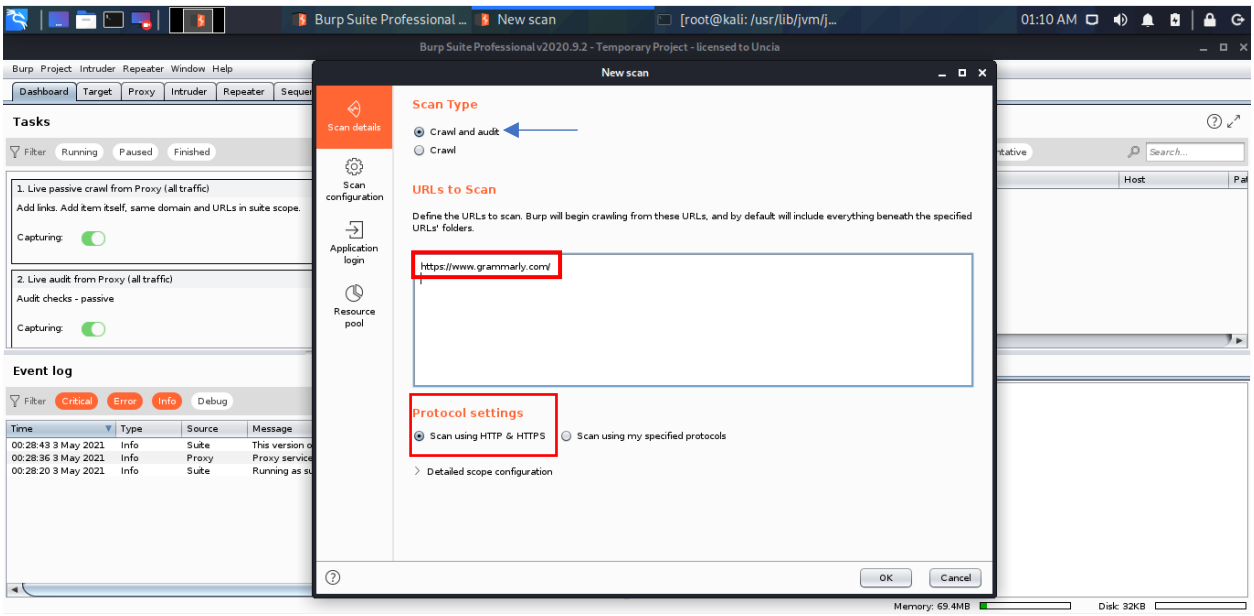


Finally, as I mentioned above, using this Netsparker tool I was able to find more vulnerabilities than the other tools and this is a very valuable tool for pen testing. In order to ensure the security of the domain, need to do those remediations and other solutions regarding those vulnerabilities securely.

VI. Burp Suite Professional

Burp Suite is an advanced framework for conducting web application security monitoring. Its different tools work in unison to help the whole testing process, from plotting and analyzing an application's attack surface to identifying and manipulating security vulnerabilities. It gives complete control to the user, allowing user to mix sophisticated manual processes with cutting-edge automation to speed up and improve the task.

I used Burp Suite Professional Version to crawl and audit my target domain and I was able to get detailed report and some vulnerabilities. I installed in on my Kali environment. After the tool opens, first we need to select the option of "new scan" and then we need to select the scan type as crawl and audit. Then we need to enter the URL to scan with the protocol of HTTP & HTTPS which is under the default settings of Burp as shown below.



i. Crawl and Audit:

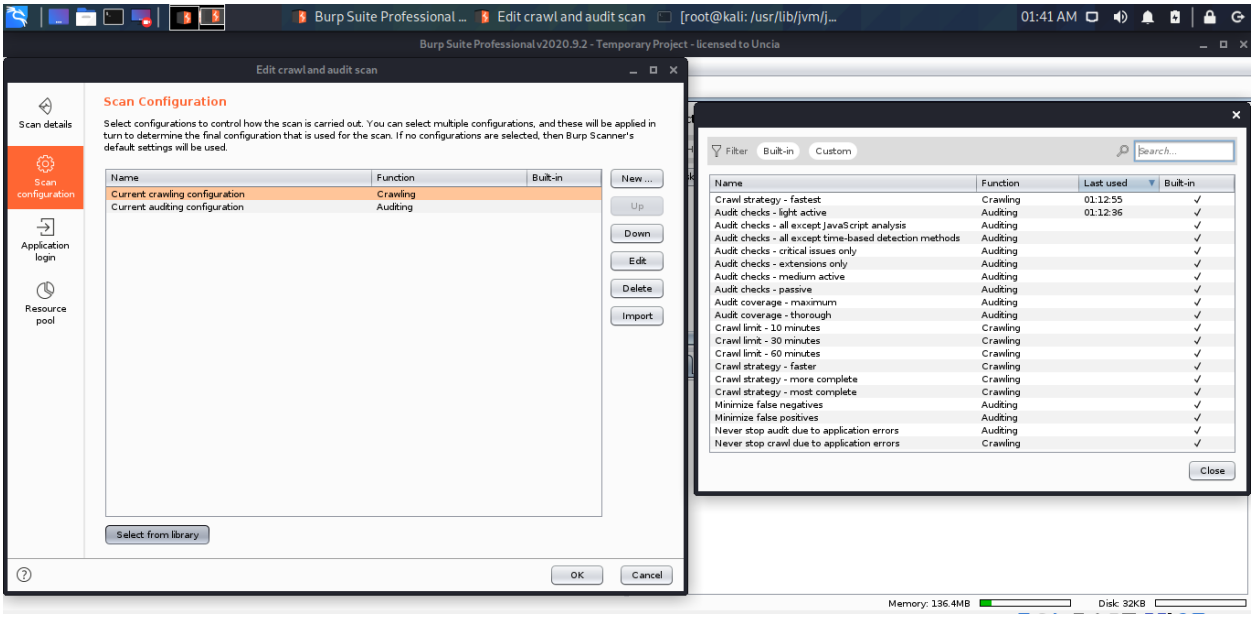
a. Crawl:

The crawl process of a scan includes traveling around the application, following connections, uploading forms, and logging in where appropriate, to index the content of the application and the navigational pathways. This seemingly easy task poses a number of obstacles that Burp's crawler is up to the task of overcoming in order to produce a detailed map of the application.

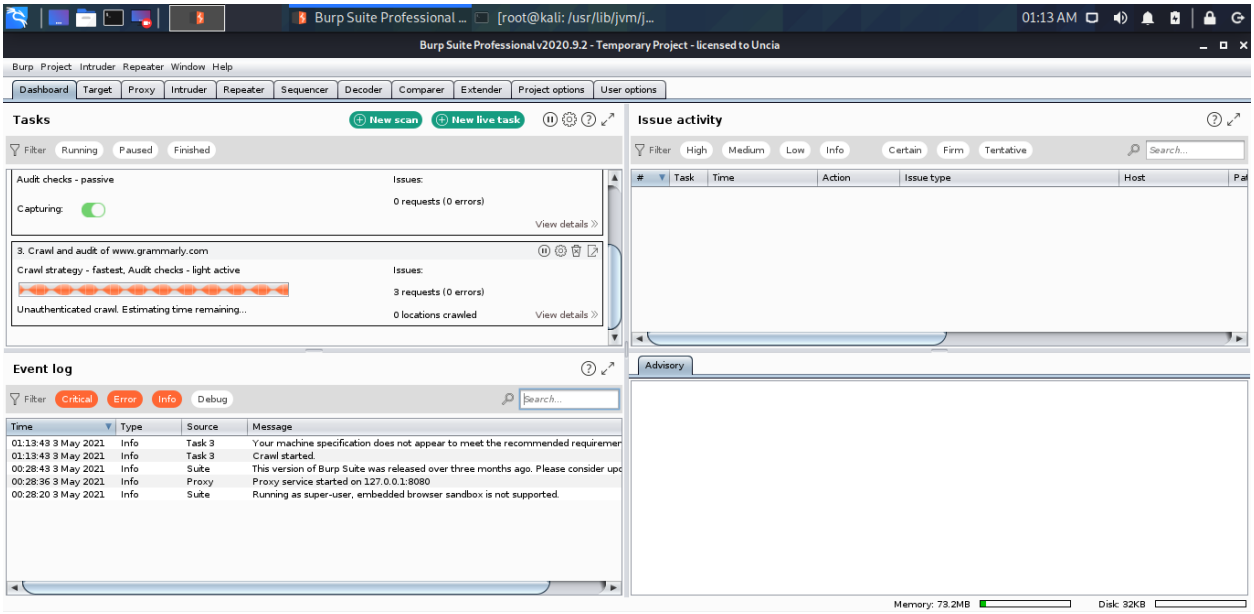
b. Audit:

During the audit process of a scan, the application's traffic and actions are analyzed for security flaws and other problems. Burp Scanner uses a variety of tools to provide a comprehensive, dead-on audit of the program being scanned.

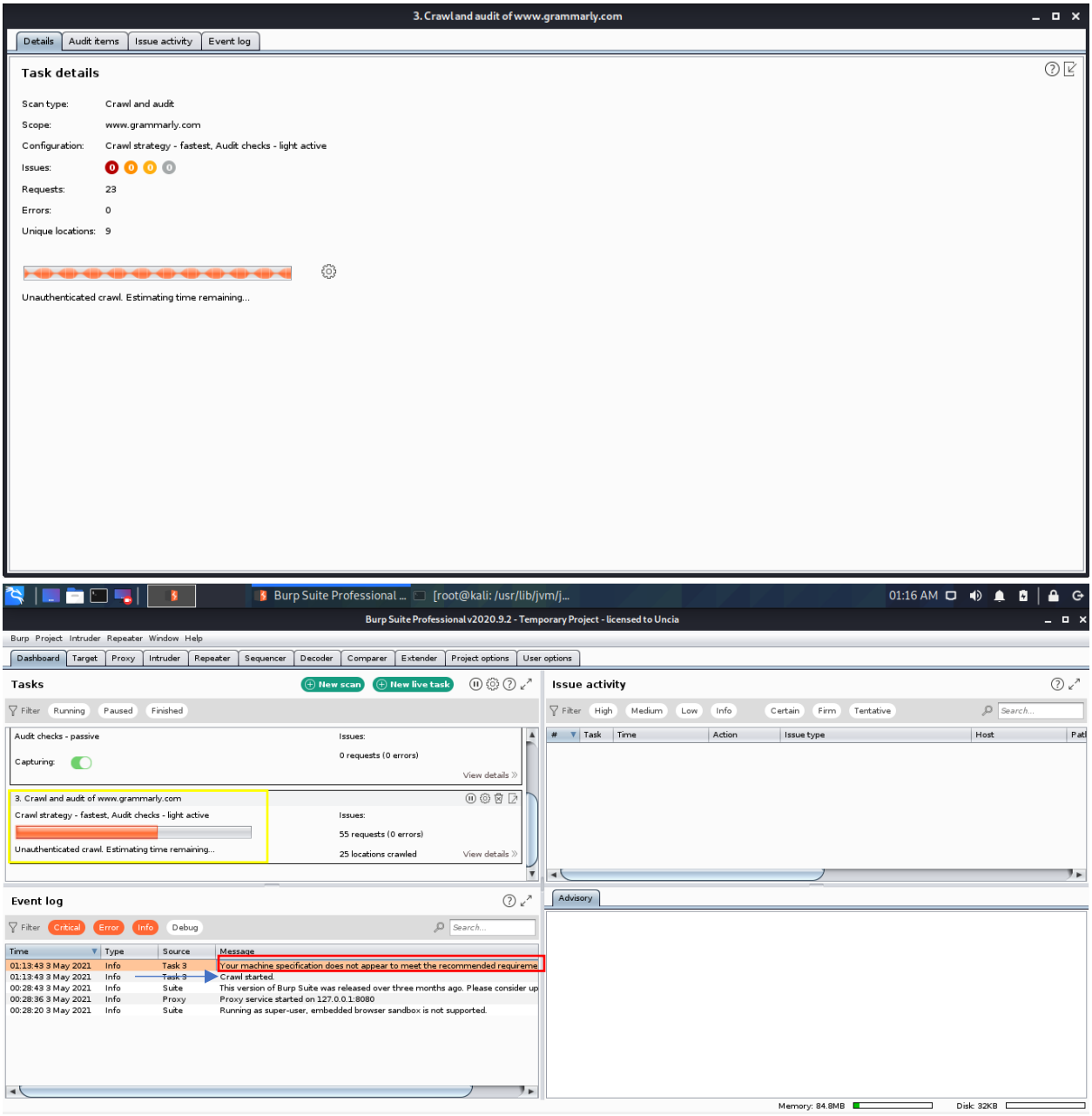
Furthermore, we can set the configurations of scan as we want, and I first scan my target using default configurations, but it takes so much time and getting my computer freeze. All the scanning configuration libraries are shown below.



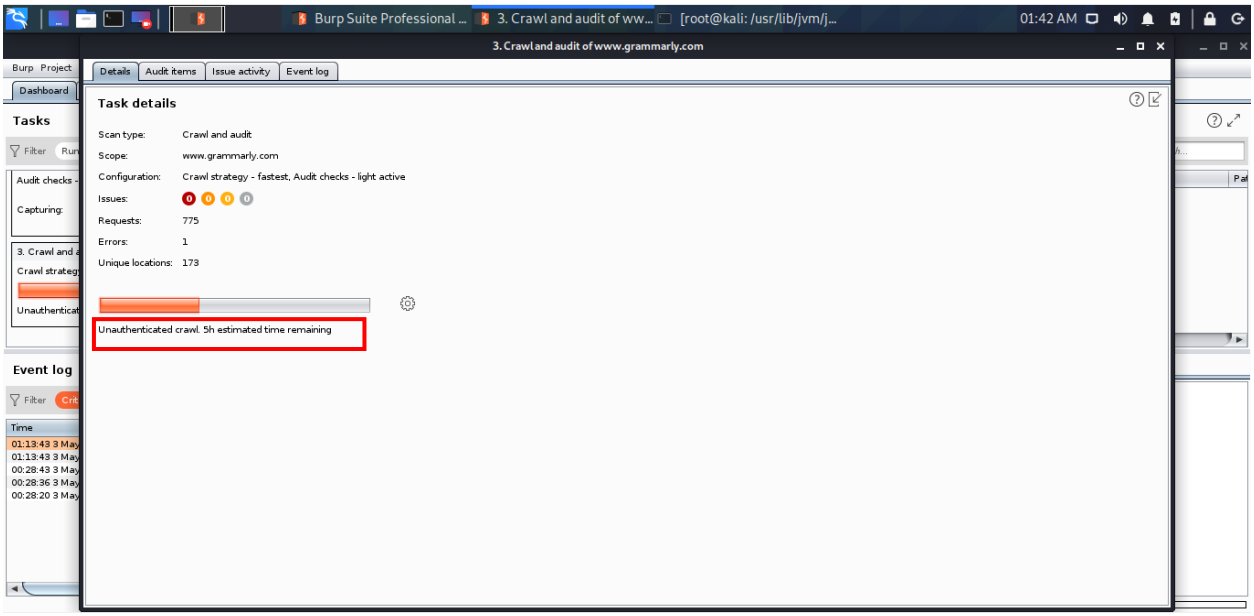
Then I change some configurations in crawl and audit feature in order to reduce the scan time and prevent my computer from freezing throughout the process. Then it will start the scan automatically.



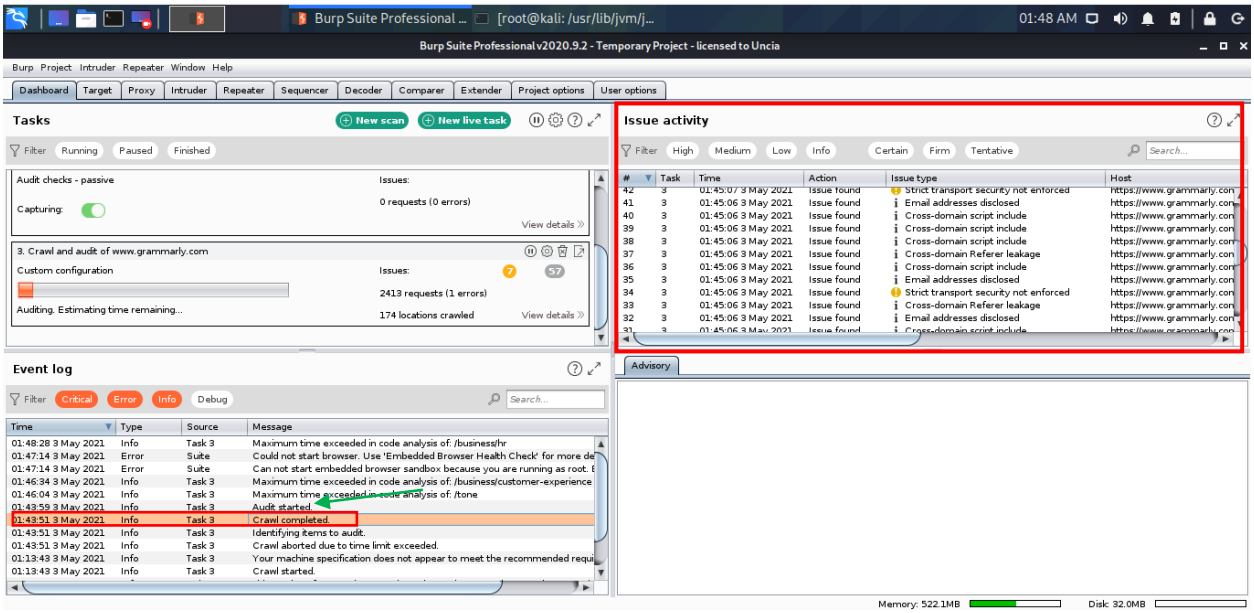
We can see the number of issues, requests, error, and some relevant information of the scan while it is continuing the web audit by maximizing the relevant tab and the following figure shows the output.

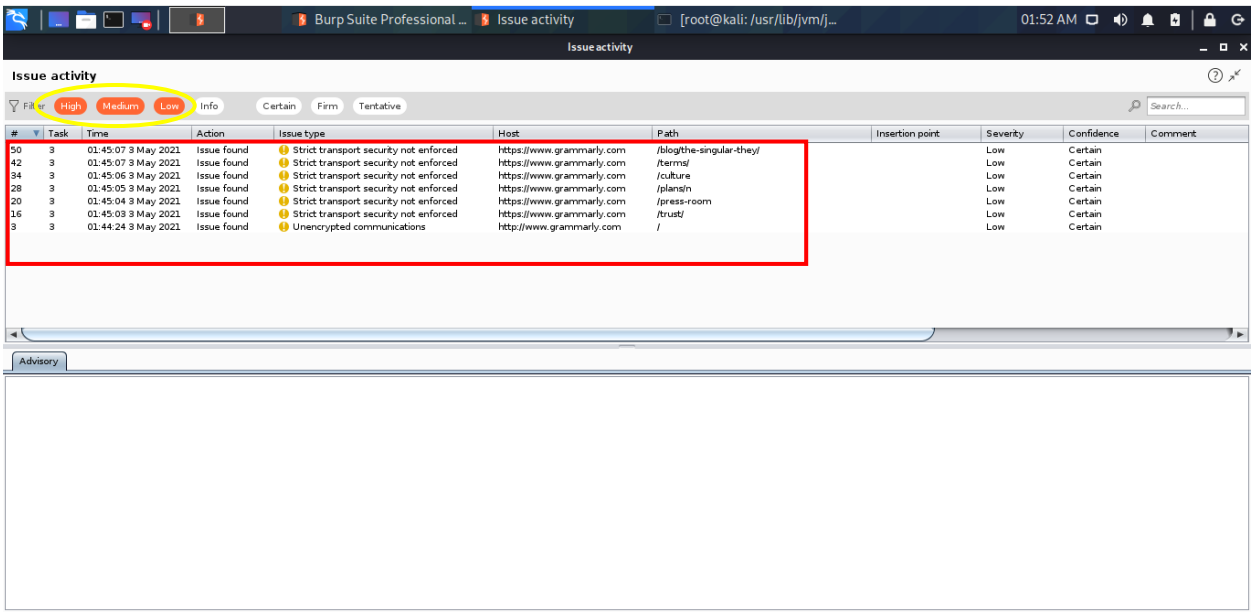


As shown in the above figure, even I change configurations to low and speedy scan, it prompts as my machine has not enough requirements and performance to continue scan and will take lot of time to finish.

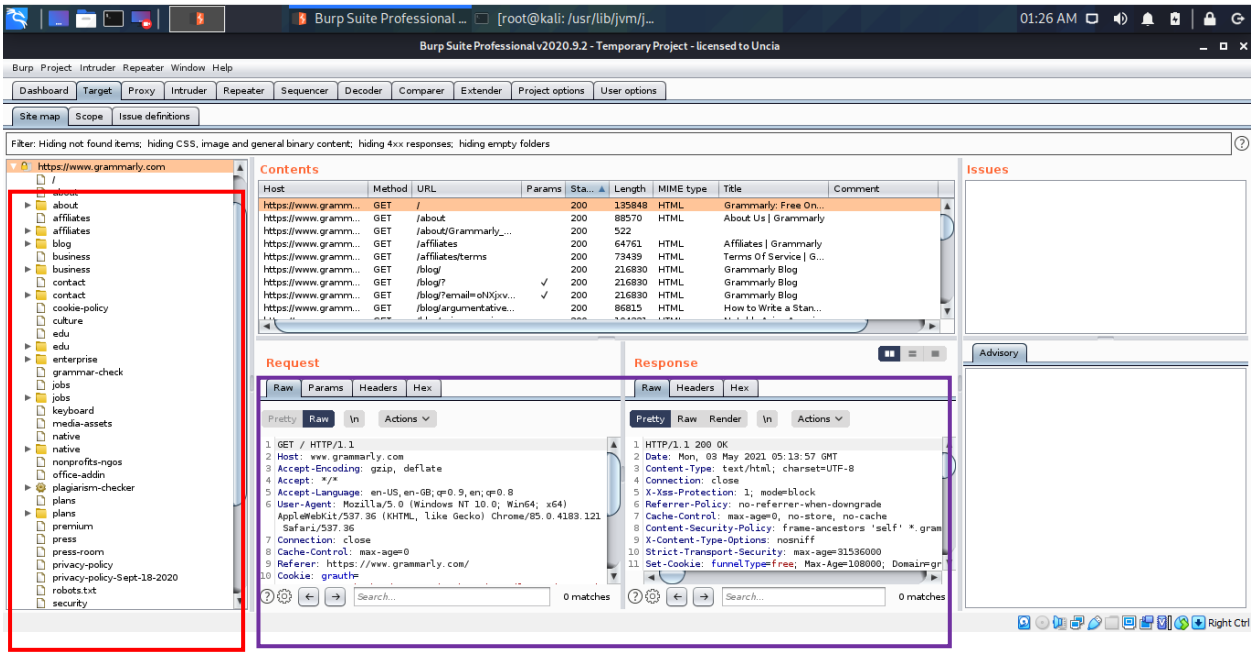


Then it finished the crawling phase and starts to audit my target domain. Then we were able to examine the identified vulnerabilities under the issue activity panel.

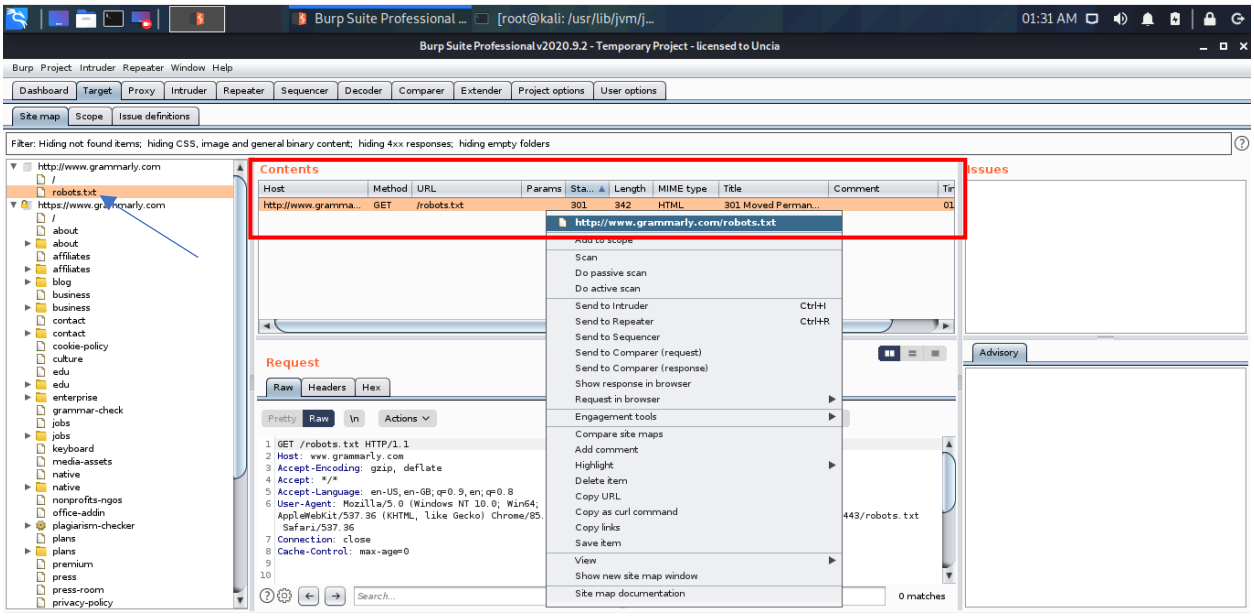




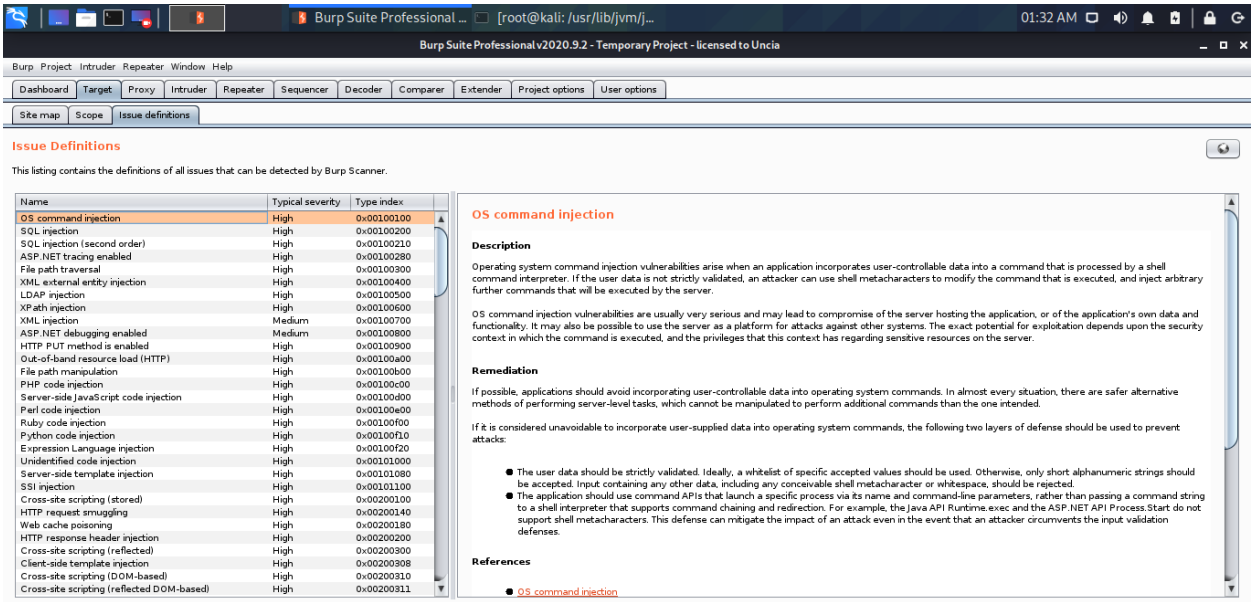
We can view the site map under the target tab and can view the request and response for specific vulnerabilities as shown in the below figure.



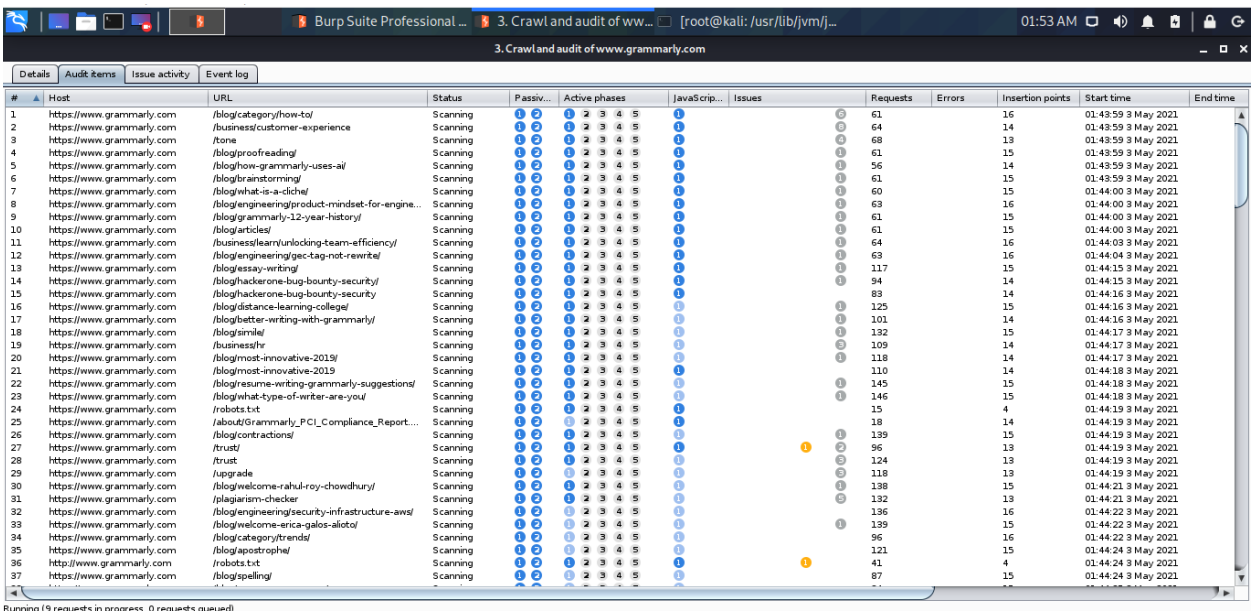
I was able to find the robot.txt file using this Burp tool as the previous tool of Netsparker.



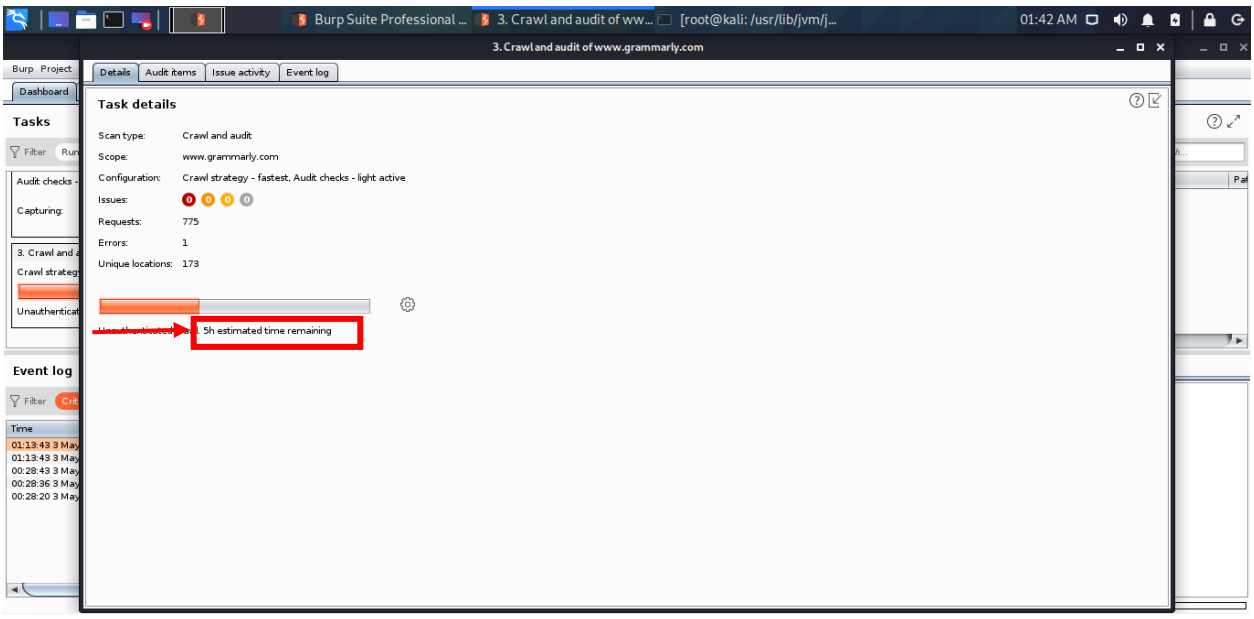
As shown in the below figure, we can see the definitions of some issues under the target tab.



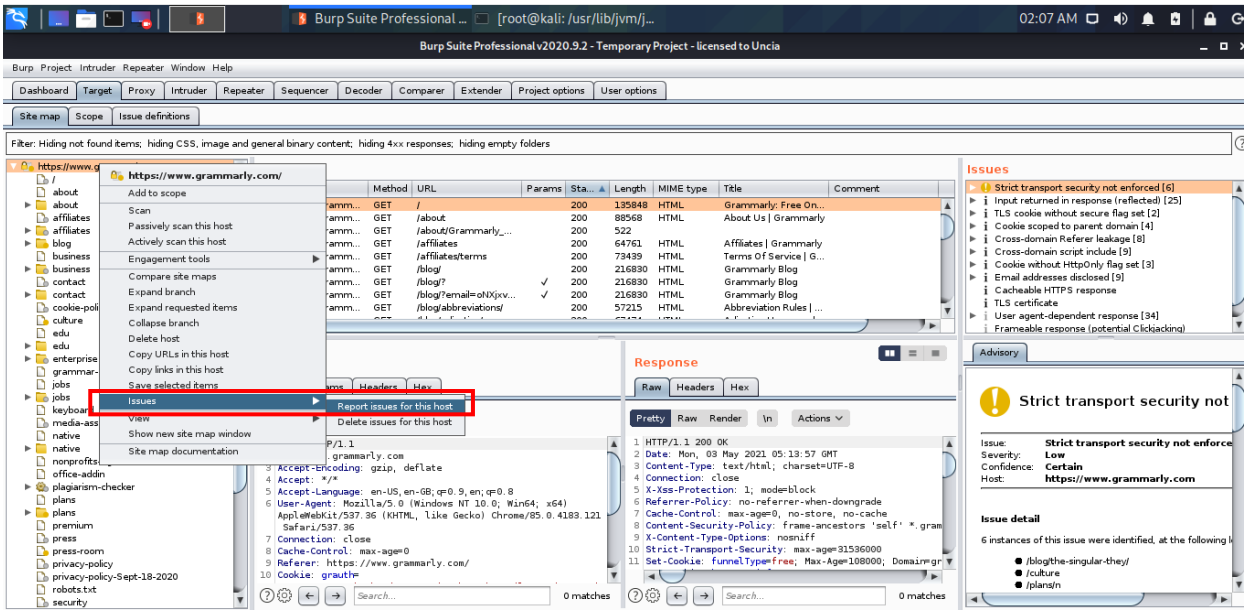
We can view the audited items with status and some of other relevant information in a diagrammatic view.



Since it took so much time, I stopped the audit phase after waiting more than three hours and generate the report based on the identified vulnerabilities. On that period, Burp Suite able to identify 6 number of low vulnerabilities and 62 number of information vulnerabilities which are under the certain confidence. Also, there were 36 number of information vulnerabilities under the firm confidence level.



We can generate the report as shown below.



The generated report based on the identified vulnerabilities is shown below. The report consists with all the details of specified vulnerability and remediation instructions as same as the previous Netsparker tool.

Burp Scanner Report

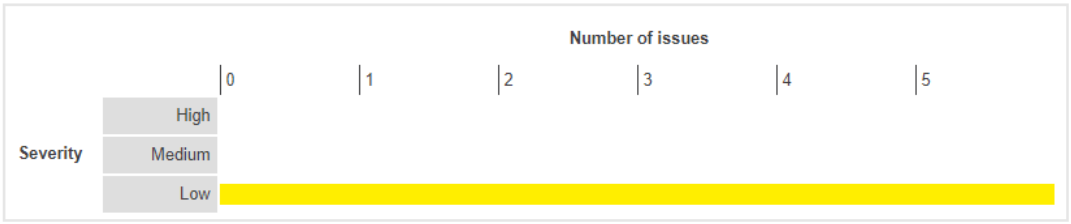
BURPSUITE PROFESSIONAL

Summary

The table below shows the numbers of issues identified in different categories. Issues are classified according to severity as High, Medium, Low or Information. This reflects the likely impact of each issue for a typical organization. Issues are also classified according to confidence as Certain, Firm or Tentative. This reflects the inherent reliability of the technique that was used to identify the issue.

		Confidence			
		Certain	Firm	Tentative	Total
Severity	High	0	0	0	0
	Medium	0	0	0	0
	Low	6	0	0	6
	Information	62	36	0	98

The chart below shows the aggregated numbers of issues identified in each category. Solid colored bars represent issues with a confidence level of Certain, and the bars fade as the confidence level falls.



Identified Vulnerabilities

1. **Vulnerability: Strict transport security not enforced**
   Severity Level: Low
   Vulnerable URLs:

https://www.grammarly.com/blog/the-singular-they/
https://www.grammarly.com/culture
https://www.grammarly.com/plans/n
https://www.grammarly.com/press-room
https://www.grammarly.com/terms/
https://www.grammarly.com/trust/

1. Strict transport security not enforced

Next

There are 6 instances of this issue:

- /blog/the-singular-they/
- /culture
- /plans/n
- /press-room
- /terms/
- /trust/

Issue description

The application fails to prevent users from connecting to it over unencrypted connections. An attacker able to modify a legitimate user's network traffic could bypass the application's use of SSL/TLS encryption, and use the application as a platform for attacks against its users. This attack is performed by rewriting HTTPS links as HTTP, so that if a targeted user follows a link to the site from an HTTP page, their browser never attempts to use an encrypted connection. The sslstrip tool automates this process.

To exploit this vulnerability, an attacker must be suitably positioned to intercept and modify the victim's network traffic.This scenario typically occurs when a client communicates with the server over an insecure connection such as public Wi-Fi, or a corporate or home network that is shared with a compromised computer. Common defenses such as switched networks are not sufficient to prevent this. An attacker situated in the user's ISP or the application's hosting infrastructure could also perform this attack. Note that an advanced adversary could potentially target any connection made over the Internet's core infrastructure.

Issue remediation

The application should instruct web browsers to only access the application using HTTPS. To do this, enable HTTP Strict Transport Security (HSTS) by adding a response header with the name 'Strict-Transport-Security' and the value 'max-age=expireTime', where expireTime is the time in seconds that browsers should remember that the site should only be accessed using HTTPS. Consider adding the 'includeSubDomains' flag if appropriate.

Note that because HSTS is a "trust on first use" (TOFU) protocol, a user who has never accessed the application will never have seen the HSTS header, and will therefore still be vulnerable to SSL stripping attacks. To mitigate this risk, you can optionally add the 'preload' flag to the HSTS header, and submit the domain for review by browser vendors.

References

- HTTP Strict Transport Security
- sslstrip
- HSTS Preload Form

Vulnerability classifications

- CWE-523: Unprotected Transport of Credentials

1.1. https://www.grammarly.com/blog/the-singular-they/

Next

Summary

Severity:	**Low**
Confidence:	**Certain**
Host:	**https://www.grammarly.com**
Path:	**/blog/the-singular-they/**

Request

```
GET /blog/use-the-singular-they/ HTTP/1.1
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en-US,en-GB;q=0.9,en;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.121 Safari/537.36
Connection: close
Host: www.grammarly.com
```

Response

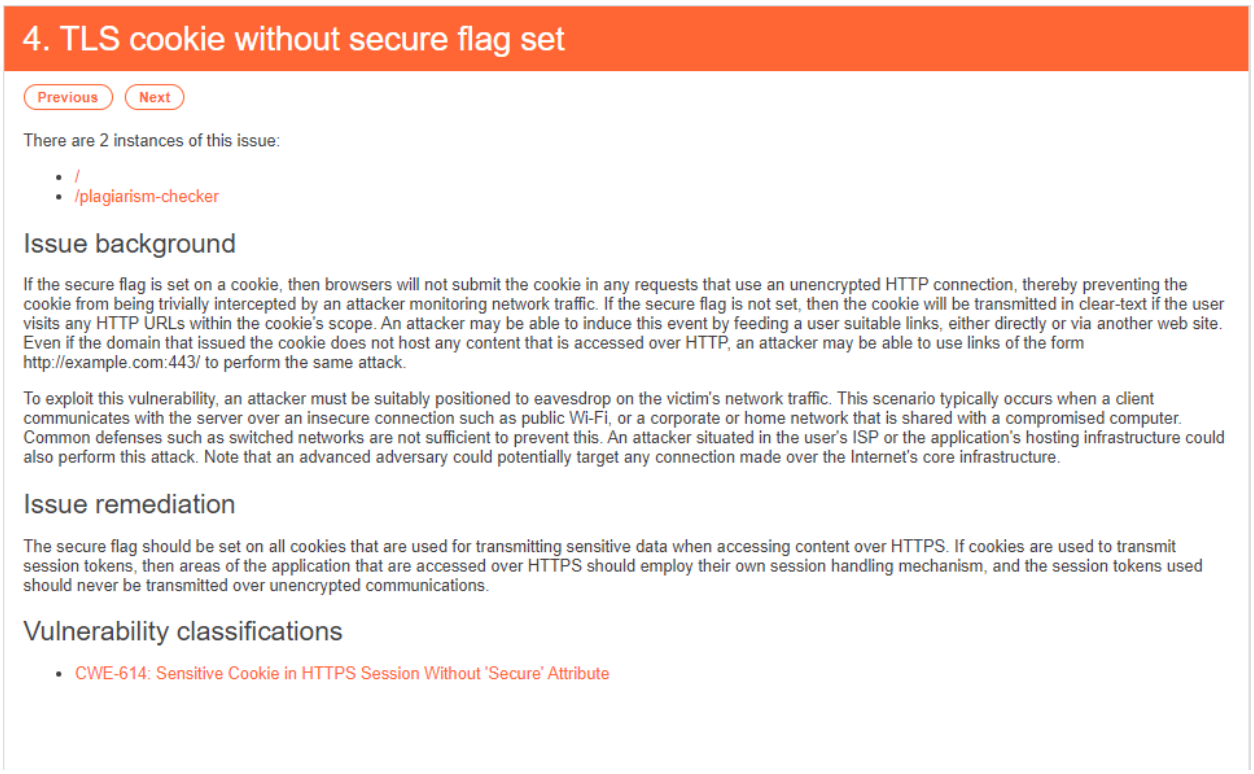
```
HTTP/1.1 301 Moved Permanently
Server: awselb/2.0
Date: Mon, 03 May 2021 05:34:19 GMT
Content-Type: text/html
Content-Length: 134
Connection: close
Location: https://www.grammarly.com:443/blog/use-the-singular-they/

<html>
<head><title>301 Moved Permanently</title></head>
<body>
<center><h1>301 Moved Permanently</h1></center>
</body>
</html>
```

This vulnerability happens due to the Strict-Transport-Security header and attackers are able to read and change your communication with the website simply by bypassing HSTS header. In order to remediate this vulnerability, domain should add to the HSTS preloaded list and then the user's browsers can connect to the webpage using HTTPS request automatically.

2. **Vulnerability: TLS cookie without secure flag set**
   Severity Level: Information
   Vulnerable URLs:

https://www.grammarly.com/
https://www.grammarly.com/plagiarism-checker



4. TLS cookie without secure flag set

Previous Next

There are 2 instances of this issue:

- /
- /plagiarism-checker

Issue background

If the secure flag is set on a cookie, then browsers will not submit the cookie in any requests that use an unencrypted HTTP connection, thereby preventing the cookie from being trivially intercepted by an attacker monitoring network traffic. If the secure flag is not set, then the cookie will be transmitted in clear-text if the user visits any HTTP URLs within the cookie's scope. An attacker may be able to induce this event by feeding a user suitable links, either directly or via another web site. Even if the domain that issued the cookie does not host any content that is accessed over HTTP, an attacker may be able to use links of the form http://example.com:443/ to perform the same attack.

To exploit this vulnerability, an attacker must be suitably positioned to eavesdrop on the victim's network traffic. This scenario typically occurs when a client communicates with the server over an insecure connection such as public Wi-Fi, or a corporate or home network that is shared with a compromised computer. Common defenses such as switched networks are not sufficient to prevent this. An attacker situated in the user's ISP or the application's hosting infrastructure could also perform this attack. Note that an advanced adversary could potentially target any connection made over the Internet's core infrastructure.

Issue remediation

The secure flag should be set on all cookies that are used for transmitting sensitive data when accessing content over HTTPS. If cookies are used to transmit session tokens, then areas of the application that are accessed over HTTPS should employ their own session handling mechanism, and the session tokens used should never be transmitted over unencrypted communications.

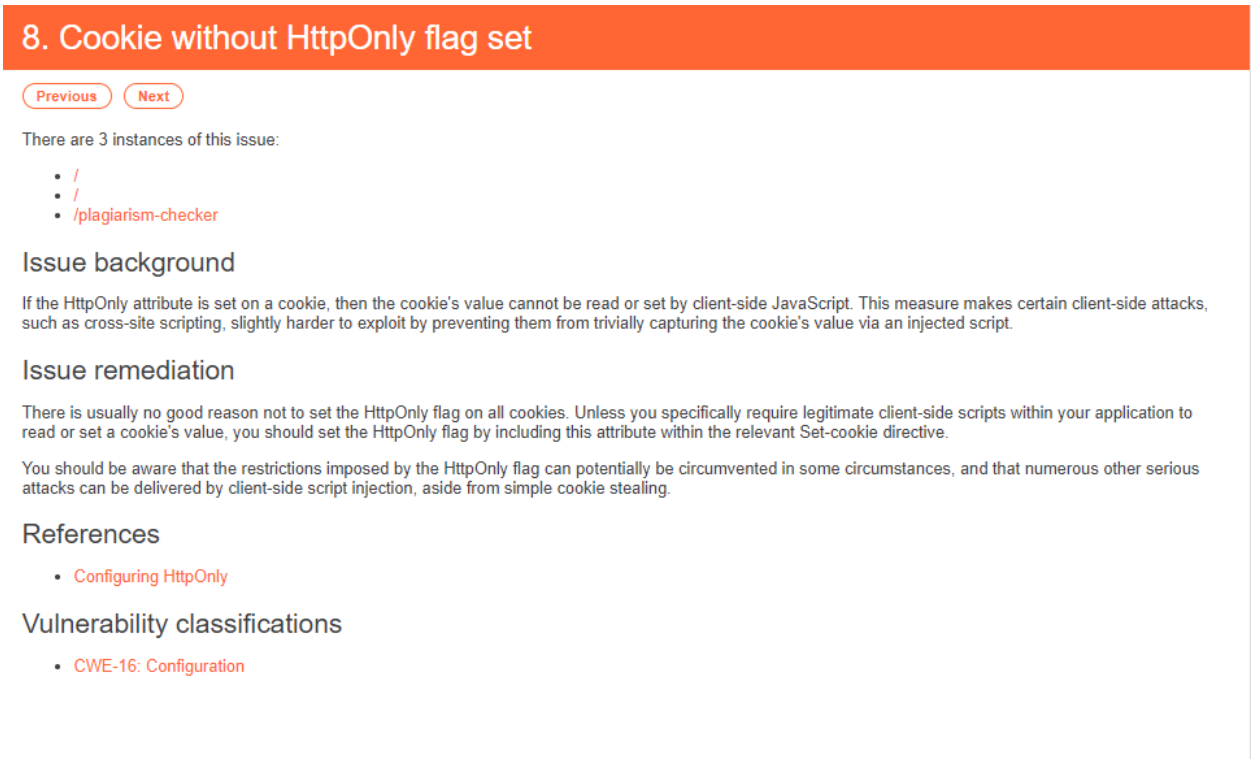
Vulnerability classifications

- CWE-614: Sensitive Cookie in HTTPS Session Without 'Secure' Attribute

Attacker can use this issue by providing some user suitable links since the secure flag is not set and then the cookie will be transmitted in plain-text format when the user visits any HTTP URLs within the cookie's scope. In order to remediate this vulnerability, secure flag should be set on all cookies that are used for transmitting sensitive data.

3. **Vulnerability: Cookie without HttpOnly flag set**
   Severity Level: Information
   Vulnerable URLs:

https://www.grammarly.com/
https://www.grammarly.com/
https://www.grammarly.com/plagiarism-checker



8. Cookie without HttpOnly flag set

Previous Next

There are 3 instances of this issue:

- /
- /
- /plagiarism-checker

Issue background

If the HttpOnly attribute is set on a cookie, then the cookie's value cannot be read or set by client-side JavaScript. This measure makes certain client-side attacks, such as cross-site scripting, slightly harder to exploit by preventing them from trivially capturing the cookie's value via an injected script.

Issue remediation

There is usually no good reason not to set the HttpOnly flag on all cookies. Unless you specifically require legitimate client-side scripts within your application to read or set a cookie's value, you should set the HttpOnly flag by including this attribute within the relevant Set-cookie directive.

You should be aware that the restrictions imposed by the HttpOnly flag can potentially be circumvented in some circumstances, and that numerous other serious attacks can be delivered by client-side script injection, aside from simple cookie stealing.

References

- Configuring HttpOnly

Vulnerability classifications

- CWE-16: Configuration

I was able to find Cookie without HTTP Only flag set vulnerability found through the scan which takes the CWE-16 classification. A Set-Cookie HTTP response header includes an additional flag called HttpOnly. When creating a cookie, use the HttpOnly flag to reduce the possibility of a client side script accessing the secure cookie. In order to remediate this issue, need to set the HTTPOnly flag by including attribute within the relevant Set-cookie directive.

Finally, as I mentioned above, using this Burp Suite tool I was able to only find few vulnerabilities since it took more time than the other tool. However, I was able to find 6 low severity vulnerabilities and 62 informational vulnerabilities on that time period. In order to ensure the security of the domain, need to do those remediations and other solutions regarding those vulnerabilities securely.

Conclusion

Overall, the web application that was audited was well-designed and had a number of security features in order to protect it from cyber-attacks. During the research, only medium to low-risk vulnerabilities were found, as well as best practices, with no high-risk vulnerabilities, which is a good sign that a website is safe and reliable. The website, on the other hand, was discovered to be vulnerable to a variety of today's most common cyber-attacks, such as Cross Site Scripting and SQL injections. Grammarly is a well-known spelling and grammar checker on the internet, and the fact that such a well-known company cannot easily protect itself from these forms of cyber-attacks shows how difficult network and cyber defense has become in recent years.

References

1. https://www.getastra.com/blog/security-audit/website-security-audit/
2. https://webscoot.io/blog/website-security-audit/#how
3. https://www.bugcrowd.com/bug-bounty-list/
4. https://hackerone.com/grammarly?type=team
5. https://www.grammarly.com/
6. https://oscp.medium.com/owasp-amass-installation-b5ba0c54f94a
7. https://hydrasky.com/network-security/kali-tools/amass-subdomain-enumeration-tool/
8. https://github.com/projectdiscovery/subfinder
9. https://www.youtube.com/watch?v=V910F5abjc0&ab_channel=MrMudhalai
10. https://www.youtube.com/watch?v=vV1wIK9CwI&ab_channel=5H4D0WR007
11. https://tools.kali.org/information-gathering/nikto
12. https://www.youtube.com/watch?v=bLLPxcE-x50&ab_channel=SemiYulianto
13. https://www.cmscritic.com/reviews/netsparker-web-application-scanner-review/#
14. https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/http-strict-transport-security-hsts-errors-and-warnings/
15. https://www.youtube.com/watch?v=VP9eQhUASYQ&ab_channel=PortSwigger

Explanation Video URL:

https://mysliit-my.sharepoint.com/:v:/g/personal/it19976686_my_sliit_lk/EewPtRFPRoBMrIYnc3WEFfABkVmYhyrCiMG0O0ytmvfttg?e=o94ZhR