



Web Security – IE2062

4. Grammarly Bug Bounty

A D A Ihansa

IT22899606

Web Audit

Grammarly.com

Contents

Introduction to Bug Bounty program and audit scope	4
Information gathering phase.	6
Finding active subdomains and their states	6
Sublist3r	6
HTTPProbe	9
Netcraft.....	10
Spiderfoot.....	11
Google Dorks	13
Directory and services enumeration.....	15
Dirbuster.....	16
Nmap	18
Automated Testing	19
OWASP ZAP	19
Manual Testing	33
SQL injection.....	33
Checking for Insecure HTTP methods	34
Conclusion.....	36
References	36

Introduction to Bug Bounty program and audit scope

Grammarly is a writing aid powered by artificial intelligence that assists users in improving their writing abilities. It provides recommendations for grammar, spelling, punctuation, and style, thereby simplifying the process of communicating effectively and with confidence. The tool is versatile and can aid in a variety of writing tasks, from crafting emails to writing academic papers. It is available on a range of platforms and devices. Grammarly's goal is to enhance the clarity and quality of written communication, helping users to reach their writing objectives.

In hackerone bug bounty program, they defined these subdomains (and all inclusive) as valid subdomains for testing.

*.grammarly.com - list of *.grammarly.com subdomains.

*.grammarly.io

*.grammarlyaws.com

*.grammarly.ai

The bug bounty program specifies the eligible subdomains within its scope, stating that any subdomain falling under grammarly.com is included.

Asset name [↑]	Type [↑]	Coverage [↑]	Max. severity [↓]	Bounty [↑]	Last update [↑]
MS Office Add-In Grammarly add-on (works with MS Word and Outlook for Windows), where authorized users can check their Word documents or emails. Auto-update functionality can be tested on an older version. Download URL: https://download-office.grammarly.com/latest/GrammarlyAddInSetup.exe . Prerequisites: MS Word/Outlook, .NET Framework 4.5. Vulnerabilities are eligible for submission if they're reproducible on any version of Word/Outlook on Windows 10 with all latest security patches applied. The vulnerability should be tested on a system without additional SDKs and development kits. We cover your expenses on a Word/Outlook license if the report appears being valid.	Other	In scope	Critical	Eligible	Apr 6, 2020
com.grammarly.keyboard	iOS: App Store	In scope	Critical	Eligible	Oct 9, 2019
*.grammarly.io	Wildcard	In scope	Critical	Eligible	May 15, 2023
Grammarly for Microsoft Word Vulnerabilities are eligible for submission if they're reproducible on any version of Word on OS with all latest security patches applied. The vulnerability should be tested on a system without additional SDKs and development kits. We cover your expenses on a Word license if the report appears to be valid. You can install Grammarly for Microsoft Word at https://appssource.microsoft.com/en-us/products/office/76A200001011 English JavaScript	Executable	In scope	Critical	Eligible	Apr 6, 2020
Grammarly Desktop for macOS https://download-mac.grammarly.com/Grammarly.dmg Swift	Executable	In scope	Critical	Eligible	Nov 23, 2021
Capture the Flag The first hacker who reports the \$FLAG saved in the document { document_id: 1198436185 } of the user h1_ctf@grammarly.com (user_id: 1411519194) will be awarded a \$100K bounty.	Other	In scope	Critical	Eligible	Jul 20, 2021
Grammarly Desktop for Windows https://download-windows.grammarly.com/GrammarlyInstaller.exe C#	Executable	In scope	Critical	Eligible	Nov 23, 2021
*.grammarlyaws.com	Wildcard	In scope	Critical	Eligible	May 15, 2023
New Grammarly Assistant Grammarly's AI writing assistant is a powerful tool that leverages generative AI to assist users in composing, rewriting, ideating, and replying to texts. It's contextually aware and offers personalized suggestions that respect user authenticity. The assistant is integrated into Grammarly's existing product offerings and can be used across many popular desktop applications and websites. It provides on-demand assistance, allowing users to generate high-quality, task-appropriate writing and revisions. The assistant is also capable of incorporating organizational context for Grammarly Business customers, providing text that's tailored to the business. More information about Grammarly Assistant: https://www.grammarly.com/ai Article to get you started: https://support.grammarly.com/hc/en-us/articles/14528857014285-Introducing-generative-AI-assistance	Other	In scope	Critical	Eligible	Updated Apr 4, 2024
com.grammarly.android.keyboard Vulnerabilities in Grammarly Mobile Keyboard for Android with a working proof of concept may qualify for an additional bounty through the Google Play Security Rewards Program . To see which vulnerabilities may qualify for a bounty, please refer to the Google Play Security Rewards Program's Vulnerability Criteria .	Android: Play Store	In scope	Critical	Eligible	Oct 9, 2019
Browser Extensions The extension is available in the extension/add-on store of the respective browser: <ul style="list-style-type: none"> Chrome Firefox Edge Safari Browser Extension vulnerabilities will not be distinguished. For example, if a vulnerability exists in the Chrome and Safari extensions, we will consider it the same vulnerability and will only award one bounty.	Other	In scope	Critical	Eligible	Oct 9, 2019
AppActions With app actions, you can connect Grammarly to apps you use every day and perform common tasks directly from Grammarly. This saves time by avoiding context-switching and helps you stay in the flow of writing. URL's in scope: <ul style="list-style-type: none"> 3p-access.grammarly.com/* goldengate.grammarly.com/skills-proxy/* goldengate.grammarly.com/skills/* You can read more about "App Actions" here - https://support.grammarly.com/hc/en-us/articles/21227721882233-Introducing-App-Actions .	Other	In scope	Critical	Eligible	Jan 25, 2024
*.grammarly.com	Wildcard	In scope	Critical	Eligible	May 15, 2023
grammarly.ai This service doesn't handle, store or transfer any internal data or data of our users. Additionally, it is located in a separate VPC and isn't part of our infrastructure. We accept only critical submissions (SSRF, XXE, SQLi, RCE) with a clearly reproducible proof of concept code. <i>Reports that don't match these criteria will be closed as "N/A".</i>	Domain	In scope	Low	Eligible	Oct 9, 2019

Information gathering phase.

The initial phase of information gathering, commonly known as reconnaissance or recon, is crucial for obtaining insights into the nature and behavior of the target. This phase holds significant importance during audits or attacks as it facilitates the identification of potential vulnerabilities by gaining a deeper understanding of the target.

There are two main methods for conducting information gathering scans:

1. Active Scanning: This method involves generating substantial activity on the target system, often resulting in the retrieval of extensive information.
2. Passive Scanning: In contrast to active scanning, this approach minimizes disruption to the target system, albeit typically providing fewer comprehensive results compared to active scanning.

In bug bounty programs, where details about the underlying architecture of systems are usually not disclosed (referred to as black box pentesting), specific tools and techniques are essential for gathering insights into their services, devices, and exposed information. This enables testers to develop a better understanding of the systems they are assessing.

Finding active subdomains and their states

Sublist3r

Sublist3r, a Python-based tool, is designed to discover subdomains associated with a specified target website. Leveraging search engines and online web services, it scours the web for available subdomains linked to the designated target domain. Given the freedom to scrutinize any subdomain under reddit.com, it's prudent to identify additional subdomains for testing purposes.

To install Sublist3r, navigate to its GitHub repository at <https://github.com/aboul3la/Sublist3r.git>. This repository hosts all the necessary files required for installing the tool. Execute the following command in your shell to download it:

```
...
```

```
git clone https://github.com/aboul3la/Sublist3r.git
```

```
...
```

Please note that Sublist3r necessitates either Python 2.7 or Python 3.4 to operate smoothly.

After downloading the files, go inside the 'Sublist3r' directory and install the requirements by entering,

sudo pip install -r requirements.txt

After installing the requirements, enter

`python3 sublist3r.py -d <domain_name>`

to find subdomains under the mentioned domain.

**In some Linux distributions, there will be an error saying that "[!] Error: Virustotal probably now is blocking our requests". To avoid this you will need to get the API key from VirusTotal by creating an account. After the API key has been obtained, export it to an environment variable using, export VT_APIKEY=<API key>. This will work most of the time, but this is not a must.*

Since I need to check the subdomains after, I am writing the results to a file using -o switch.

```
(kali@kali)-[~/Desktop/Tools/Sublist3r]
$ python3 sublist3r.py -d grammarly.com -o /home/kali/Documents/audit/grammarly.com/grammarly.txt

File System

Sublist3r
# Coded By Ahmed Aboul-Ela - @aboul3la

[-] Enumerating subdomains now for grammarly.com
[-] Searching now in Baidu..
[-] Searching now in Yahoo..
[-] Searching now in Google..
[-] Searching now in Bing..
[-] Searching now in Ask..
[-] Searching now in Netcraft..
[-] Searching now in DNSdumpster..
[-] Searching now in Virustotal..
[-] Searching now in ThreatCrowd..
[-] Searching now in SSL Certificates..
[-] Searching now in PassiveDNS..
[!] Error: Virustotal probably now is blocking our requests
[-] Saving results to file: /home/kali/Documents/audit/grammarly.com/grammarly.txt
[-] Total Unique Subdomains Found: 177
www.grammarly.com
3p-access.grammarly.com
VDEUC-CS2.grammarly.com
VDUSW-CS1.grammarly.com
VDUSW-CS2.grammarly.com
access-manager-private-torii.grammarly.com
access-manager-torii-private.grammarly.com
account.grammarly.com
account-private.grammarly.com
admin-panel.grammarly.com
adminka.grammarly.com
qa.aicomposer.grammarly.com
answers.grammarly.com
app.grammarly.com
applet-bundles.grammarly.com
applet-bundles-static.grammarly.com
apps.grammarly.com
apps-public.grammarly.com
apps-uploads.grammarly.com
assets.grammarly.com
auth.grammarly.com
auth-private.grammarly.com
auth-secure.grammarly.com
```

Upon examining for accessible subdomains, the next step involves identifying those that are operational. This can be accomplished by employing an additional tool known as 'httpprobe'.

HTTPProbe

This tool can find domains that are up and running. To find active subdomains under this site, I am using the text file generated before by the sublist3r and writing the active subdomains to another new file.

```
(kali@kali)-[~/Desktop/Tools/Sublist3r]
$ httpprobe < /home/kali/Documents/audit/grammarly.com/grammarly.txt > /home/kali/Documents/audit/grammarly.com/grammarly_active.txt
```


Following the completion of the scan, the findings reveal that the majority of the subdomains are indeed active.

```
(kali@kali)-[~/Desktop/Tools/Sublist3r]
$ cat /home/kali/Documents/audit/grammarly.com/grammarly_active.txt
https://applet-bundles.grammarly.com
http://applet-bundles.grammarly.com
https://account.grammarly.com
https://app.grammarly.com
https://answers.grammarly.com
https://3p-access.grammarly.com
https://applet-bundles-static.grammarly.com
https://apps-public.grammarly.com
http://applet-bundles-static.grammarly.com
http://apps-public.grammarly.com
https://apps.grammarly.com
http://account.grammarly.com
https://www.grammarly.com
http://answers.grammarly.com
http://app.grammarly.com
http://3p-access.grammarly.com
http://apps.grammarly.com
http://www.grammarly.com
https://auth.grammarly.com
https://apps-uploads.grammarly.com
https://www.autoreport.grammarly.com
http://apps-uploads.grammarly.com
https://assets.grammarly.com
https://autoreport.grammarly.com
http://www.autoreport.grammarly.com
http://assets.grammarly.com
http://autoreport.grammarly.com
https://bixby-internal.grammarly.com
https://geoproximity.bixby-internal.grammarly.com
https://bixby.grammarly.com
https://geoproximity.bixby.grammarly.com
https://blog.grammarly.com
https://business-tour.grammarly.com
http://business-tour.grammarly.com
http://blog.grammarly.com
https://www.blog.grammarly.com
https://brand.grammarly.com
https://chipmunk.grammarly.com
https://capi.grammarly.com
https://capi-msdk.grammarly.com
http://www.blog.grammarly.com
https://click.grammarly.com
http://brand.grammarly.com
https://contenthub-static.grammarly.com
http://contenthub-static.grammarly.com
http://click.grammarly.com
```

Netcraft





Netcraft, an internet services firm, offers online security solutions. Their services encompass automated vulnerability scanning and application security. No downloads or intricate setups are necessary as these services are accessible online.

By utilizing Netcraft search feature on their website, users can retrieve various details including site rank, IP address, SSL/TLS versions in use, hosting country, and hosting company.

[LEARN MORE](#)[REPORT FRAUD](#)

Site report for <https://www.grammarly.com>


► [Look up another site?](#)

Share:    

Background

Site title	Not Present	Date first seen	August 2018
Site rank	Not Present	Primary language	English
Description	Not Present		

Network

Site	https://www.grammarly.com	Domain	grammarly.com
Netblock Owner	Amazon Technologies Inc.	Nameserver	NS21.DOMAINCONTROL.COM
Hosting company	Amazon	Domain registrar	godaddy.com
Hosting country	 US	Nameserver organisation	whois.wildwestdomains.com
IPv4 address	15.197.148.33 View Total	Organisation	Unknown

For full site report: [Site report for https://www.grammarly.com | Netcraft](#)

This data is publicly accessible, and some details regarding the system and its technology can be uncovered through available sources.

Since, many users utilize this website, and considering that the IP addresses match those of my specified targets, I only obtain a Netcraft report for this domain. However, reports for the other mentioned subdomains can also be retrieved from Netcraft.

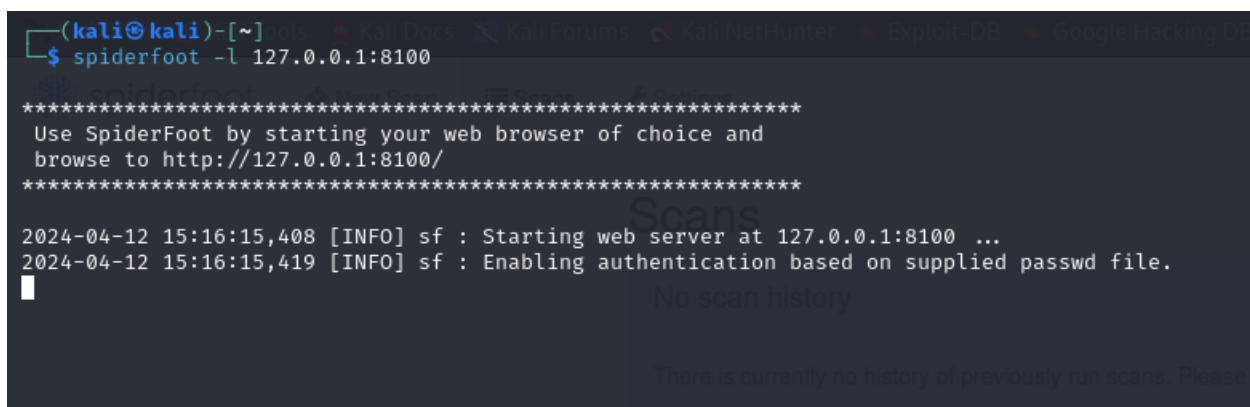
Spiderfoot

SpiderFoot is an open-source intelligence (OSINT) automation tool that is designed to simplify the process of gathering and analyzing data. It integrates with a wide range of data sources and provides an intuitive web-based interface or a command-line option. SpiderFoot is equipped with over 200 modules for various data analysis tasks, including host/sub-domain/TLD enumeration/extraction, email address, phone number and human name extraction, and much more. It also offers export options in CSV, JSON, and GEXF formats, and integrates with the TOR network for dark web searches. SpiderFoot is a powerful tool for both offensive and defensive reconnaissance, making it an asset in the field of cybersecurity.

Using spiderfoot

It must be setup, before using this tool.

Spiderfoot -l 127.0.0.1:8100

A terminal window on a Kali Linux system. The prompt is (kali@kali)-[~]. The command \$ spiderfoot -l 127.0.0.1:8100 has been entered. The output shows a series of asterisks, followed by instructions to use SpiderFoot via a web browser at http://127.0.0.1:8100/. Another series of asterisks follows. Then, two log messages are displayed: '2024-04-12 15:16:15,408 [INFO] sf : Starting web server at 127.0.0.1:8100 ...' and '2024-04-12 15:16:15,419 [INFO] sf : Enabling authentication based on supplied passwd file.' Below these, a cursor is visible. In the background, a semi-transparent window titled 'Scans' is visible, showing 'No scan history' and a message: 'There is currently no history of previously run scans. Please'.

To utilize the Spiderfoot tool, which is hosted on localhost (127.0.0.1) at port 8100, just launch a web browser and enter `http://127.0.0.1:8100` in the address bar.

After the scanner loads, proceed to "New scan" and tailor your scan type according to the scope of your investigation. There are various modules at your disposal that can be activated or deactivated based on your permissions. Since you're engaging in a passive information gathering phase, opt for the 'footprint' option to crawl and collect information about the website.

Spiderfoot results

spiderfoot

New Scan

Scans

Settings

New Scan

Scan Name

grammarly

Scan Target

www.grammarly.com

Your scan target may be one of the following. SpiderFoot will automatically detect the target type based on the format of your input:

Domain Name: e.g. example.com

IPv4 Address: e.g. 1.2.3.4

IPv6 Address: e.g. 2001:4700:4700::1111

Hostname/Sub-domain: e.g. abc.example.com

Subnet: e.g. 1.2.3.0/24

Bitcoin Address: e.g. 1HesYJSP1QqyPErQ9vZBL1wjuuNGe7R

E-mail address: e.g. bob@example.com

Phone Number: e.g. +12345678901 (E.164 format)

Human Name: e.g. "John Smith" (must be in quotes)

Username: e.g. "jsmith2000" (must be in quotes)

Network ASN: e.g. 1234

By Use Case

By Required Data

By Module

All

Get anything and everything about the target.

All SpiderFoot modules will be enabled (slow) but every possible piece of information about the target will be obtained and analysed.

Footprint

Understand what information this target exposes to the Internet.

Gain an understanding about the target's network perimeter, associated identities and other information that is obtained through a lot of web crawling and search engine use.

Investigate

Best for when you suspect the target to be malicious but need more information.

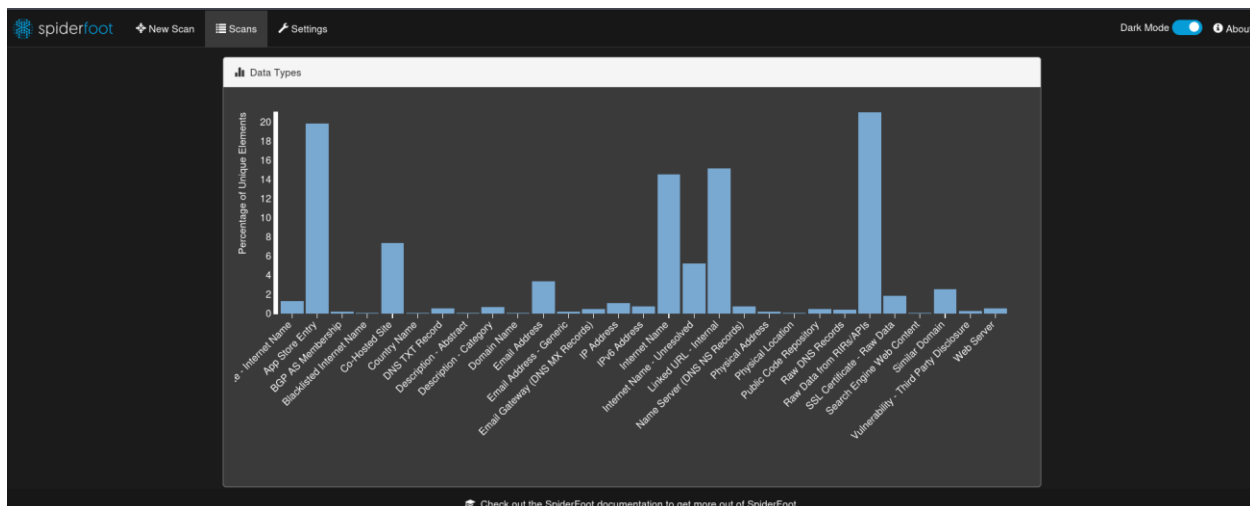
Some basic footprinting will be performed in addition to querying of blacklists and other sources that may have information about your target's maliciousness.

Passive

When you don't want the target to even suspect they are being investigated.

As much information will be gathered without touching the target or their affiliates, therefore only modules that do not touch the target will be enabled.

Run Scan Now

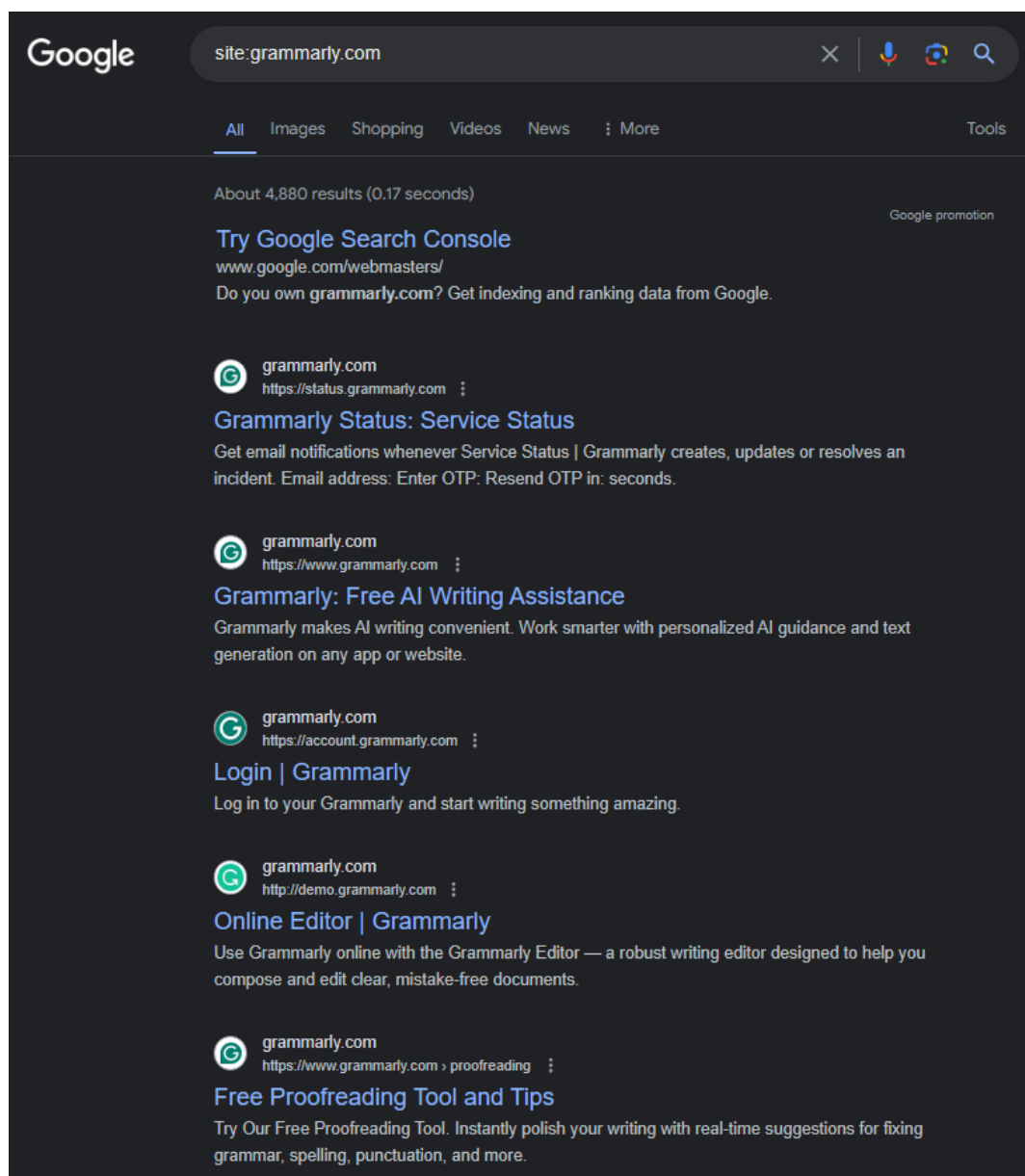


The scan has produced noteworthy findings, such as usernames, SSL certificates, and physical addresses. A significant portion of this data seems to be publicly accessible information and links leading to external websites. However, it's essential to highlight those usernames, especially when coupled with their corresponding email addresses, could potentially become avenues for social engineering or spear phishing attacks. Nevertheless, it's important to acknowledge that addressing such concerns lies beyond the boundaries of this assessment.

Google Dorks

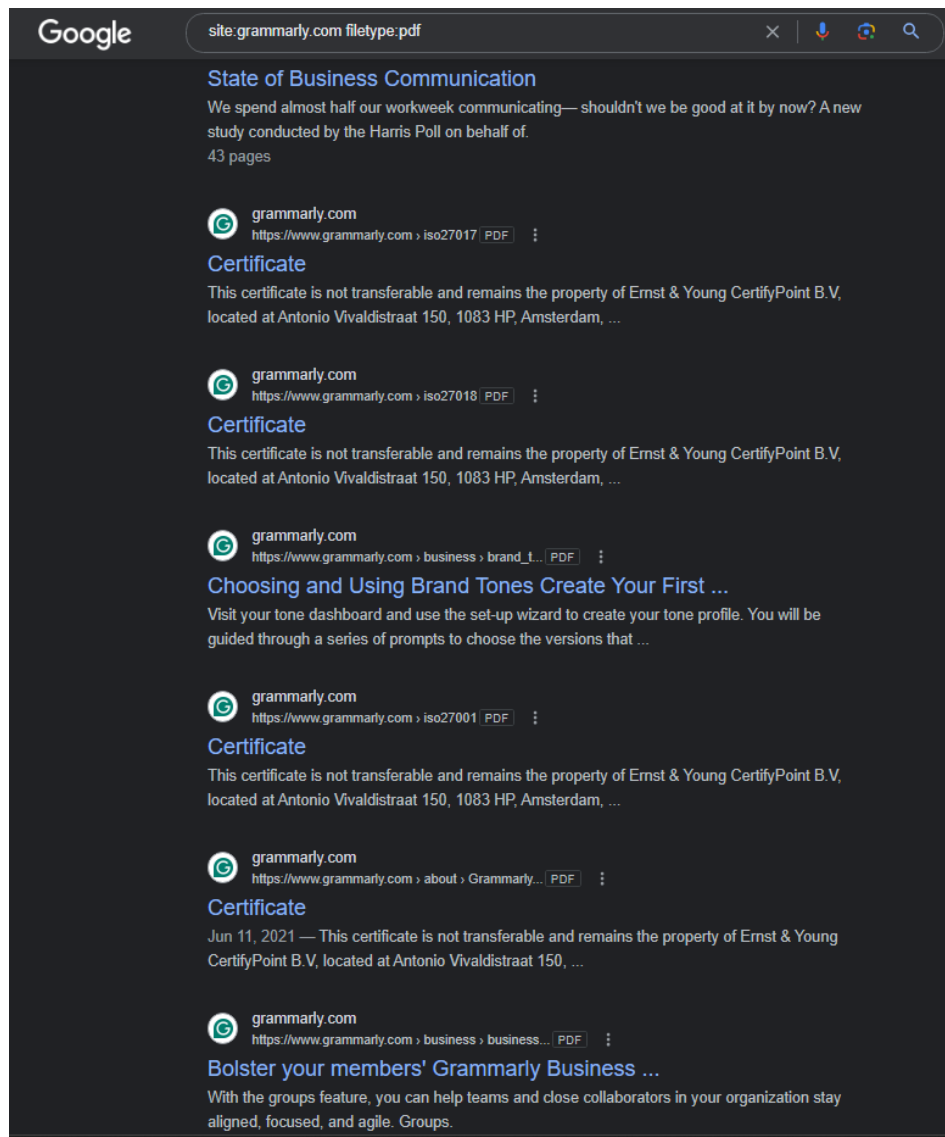
Google Dorks, also known as Google Hacking or Google Dorking, are specialized search queries that leverage Google's powerful search engine to unearth specific information and vulnerabilities that might not be accessible through standard searches. These are advanced search queries that use special operators to find specific information in Google's databases. They can be used to uncover hidden data or vulnerabilities on websites. By employing these dorks, you can focus on specific search results, unveiling hidden gems that ordinary searches might miss. They are valuable for security research but also pose risks if used maliciously. For example, they can reveal sensitive or private information about websites and the companies, organizations, and individuals that own and operate them. Google Dorks are a powerful tool for information gathering and vulnerability scanning, but they should be used responsibly to avoid unintended consequences.

site:grammarly.com operator searches for websites that has "**grammerly.com**" in their domains.



Certainly, the appearance of subdomains in search results illustrates a common utilization of Google Dorking. This method facilitates the discovery of different file types and pages that are exposed by a specific website on the internet, whether it's deliberate or unintentional. Through the refinement of search queries, users can unearth information and potentially pinpoint vulnerabilities or confidential data that might be accessible online.

During a search for various file types exposed on the internet, I came across some intriguing and possibly concerning PDFs within this subdomain.



The management of information and files within this subdomain appears to be efficient, as no additional significant files were uncovered apart from the previously mentioned PDFs.

Directory and services enumeration

Dirbuster

DirBuster, a web content scanner developed by OWASP, utilizes brute force methods to uncover different directories within a target website. By scrutinizing HTTP responses and their associated response codes, the tool detects concealed or referenced directories. Built in Java, DirBuster supports multi-threading to expedite directory scanning and produce a comprehensive file and folder structure of the target site.

Employing this tool facilitates the identification of directories or files that might be accessible yet not overtly exposed. Furthermore, it offers a glimpse into the server's file and folder arrangement, assisting in comprehending its structure and potential vulnerabilities.

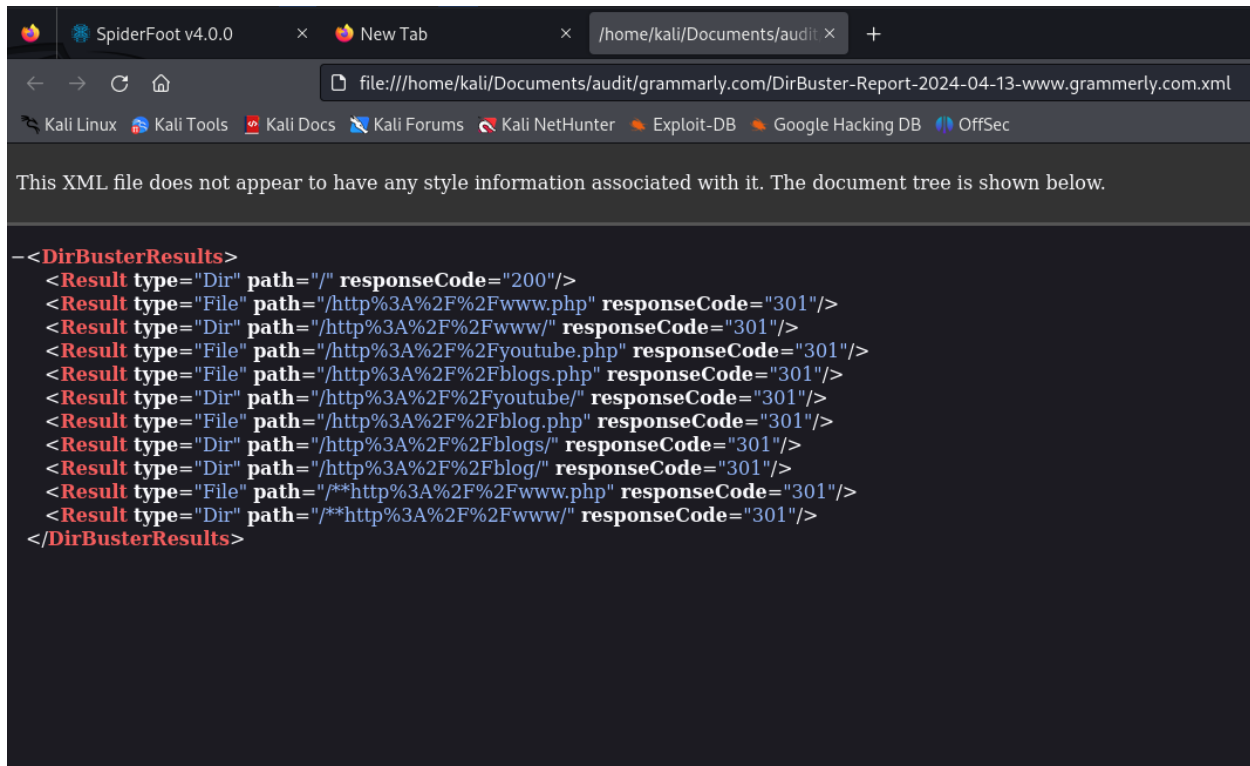
Domain: www.grammarly.com

After scanning for a while, dirbuster gave some errors and stopped working. Further looking into the issue, it seemed like the dirbuster cannot access this domain.

The screenshot displays the DirBuster web application interface. At the top, there is a menu bar with 'File', 'Options', 'About', and 'Help'. Below the menu bar, the address bar shows 'https://www.grammarly.com:443/'. The main content area is divided into two sections: 'Scan Information' and 'Results - List View: Dirs: 5 Files: 5 Results - Tree View' with a warning icon and 'Errors: 21'. The 'Results' section contains a table with two columns: 'Request' and 'Error Message'. The 'Request' column lists various URLs, and the 'Error Message' column shows 'IOException Connection reset' for most requests. Below the table, there is a status bar with the following information: 'Current speed: 70 requests/sec', 'Average speed: (T) 66, (C) 69 requests/sec', 'Parse Queue Size: 83801', 'Total Requests: 488730/1051828', 'Current number of running threads: 10', 'Time To Finish: 02:16:00', and buttons for 'Back', 'Pause', 'Stop', and 'Report'. At the bottom, it says 'Program running again' and shows the current URL: '/http%3A%2F%2Fblogs/newsfront.php'.

Request	Error Message
https://www.grammarly.com:443/orbitz/	IOException Connection reset
https://www.grammarly.com:443/orbitz/	IOException Connection reset
https://www.grammarly.com:443/http%3A%2F%2Fwww/software_development.php	IOException Connection reset
https://www.grammarly.com:443/http%3A%2F%2Fwww/convergence-standards/	IOException Remote host terminated the handshake
https://www.grammarly.com:443/formulaire.php	IOException Remote host terminated the handshake
https://www.grammarly.com:443/msg00198/	IOException Connection reset
https://www.grammarly.com:443/101887/	IOException Remote host terminated the handshake
https://www.grammarly.com:443/4x.php	IOException Connection reset
https://www.grammarly.com:443/23341/	IOException Remote host terminated the handshake
https://www.grammarly.com:443/67999.php	IOException Remote host terminated the handshake
https://www.grammarly.com:443/http%3A%2F%2Fwww/internetprivacy.php	IOException Connection reset
https://www.grammarly.com:443/interesnoe.php	IOException Connection reset
https://www.grammarly.com:443/get_rated/	IOException Remote host terminated the handshake
https://www.grammarly.com:443/http%3A%2F%2Fwww/14880.php	IOException Remote host terminated the handshake
https://www.grammarly.com:443/47178/	IOException Connection reset
https://www.grammarly.com:443/http%3A%2F%2Fwww/builddoc.php	IOException Connection reset
https://www.grammarly.com:443/http%3A%2F%2Fblogs/946/	IOException Connection reset
https://www.grammarly.com:443/http%3A%2F%2Fblogs/vcs.php	IOException Connection reset
https://www.grammarly.com:443/http%3A%2F%2Fyoutube/Event.php	IOException Connection reset
https://www.grammarly.com:443/http%3A%2F%2Fblog/wtf.php	IOException Connection reset
https://www.grammarly.com:443/http%3A%2F%2Fblogs/4134.php	IOException Connection reset

Despite the constraints, I was able to develop a broad understanding of the folder structure within the web server through exploration with DirBuster.



The screenshot shows a web browser window with the address bar displaying a file path: `file:///home/kali/Documents/audit/grammarly.com/DirBuster-Report-2024-04-13-www.grammarly.com.xml`. The browser tabs include SpiderFoot v4.0.0, New Tab, and /home/kali/Documents/audit. The browser's bookmark bar contains links to Kali Linux, Kali Tools, Kali Docs, Kali Forums, Kali NetHunter, Exploit-DB, Google Hacking DB, and OffSec. The main content area displays a message: "This XML file does not appear to have any style information associated with it. The document tree is shown below." Below this message, the XML content is displayed in a dark-themed code editor. The XML is a DirBuster report containing several results for different paths and response codes.

```
-<DirBusterResults>
  <Result type="Dir" path="/" responseCode="200"/>
  <Result type="File" path="/http%3A%2F%2Fwww.php" responseCode="301"/>
  <Result type="Dir" path="/http%3A%2F%2Fwww/" responseCode="301"/>
  <Result type="File" path="/http%3A%2F%2Fyoutube.php" responseCode="301"/>
  <Result type="File" path="/http%3A%2F%2Fblogs.php" responseCode="301"/>
  <Result type="Dir" path="/http%3A%2F%2Fyoutube/" responseCode="301"/>
  <Result type="File" path="/http%3A%2F%2Fblog.php" responseCode="301"/>
  <Result type="Dir" path="/http%3A%2F%2Fblogs/" responseCode="301"/>
  <Result type="Dir" path="/http%3A%2F%2Fblog/" responseCode="301"/>
  <Result type="File" path="/**http%3A%2F%2Fwww.php" responseCode="301"/>
  <Result type="Dir" path="/**http%3A%2F%2Fwww/" responseCode="301"/>
</DirBusterResults>
```

I manually inspected each result and did not discover any suspicious or flawed findings.

Nmap

Nmap, also known as Network Mapper, is a flexible port scanner engineered to probe hosts and reveal their open ports, associated services, port statuses, running service versions, and the operating system in use, among other details. This open-source tool serves various purposes, offering valuable insights into network configurations and potential vulnerabilities. Additionally, Nmap supports the execution of scripts on target systems to exploit vulnerabilities or gather additional information.

Upon installation, you can access the available options by typing "nmap -h" in your command line interface. For a more detailed understanding of how the tool operates, you can consult the manual page by entering "man nmap" in your command line interface. *Note that some options may require administrator / super user privileges.

*Note that some options may require administrator / super user privileges.

I am using the following scan options for this assessment.

sudo nmap <host name> -sS -sV -O -oN <filename>

-sS: Enables SYN scan (also known as Stealth scan).

-sV: Enables version detection. It tries to detect the version of the service running in that port.

-O: Enables Operating System detection.

-oN : Outputs the scan results to text file

Scanned results for <https://www.grammarly.com/>

```
(kali@kali)-[~/Documents/audit/grammarly.com]
$ sudo nmap www.grammarly.com -sS -sV -O -oN nmap_grammarly.txt
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-13 05:47 EDT
Nmap scan report for www.grammarly.com (3.33.130.190)
Host is up (0.023s latency).
Other addresses for www.grammarly.com (not scanned): 15.197.148.33
rDNS record for 3.33.130.190: a2aa9ff50de748dbe.awsglobalaccelerator.com
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
25/tcp    open  smtp?
80/tcp    open  http    OpenResty web app server
443/tcp   open  ssl/https
2 services unrecognized despite returning data. If you know the service/version, please submit the following fingerprints at https://nmap.org/cgi-bin/submit.cgi?new-service :
=====
NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)
=====
SF-Port25-TCP:V=7.94SVNXT=SSLXI=7KD=4/13XTime=661A5486XP=x86_64-pc-linux-gnuX(H
SF:ello,2A,"552x20Invalid\x20domain\x20in\x20EHLO\x20command\,r
SF:n")Xr(GenericLines,28,"500\x20Syntax\x20error\,x20command\x20unrecogniz
SF:ed\r\n")Xr(GetRequest,28,"500\x20Syntax\x20error\,x20command\x20unrecog
SF:nized\r\n")Xr(HTTPOptions,28,"500\x20Syntax\x20error\,x20command\x20unr
SF:ecognized\r\n")Xr(RTSPRequest,28,"500\x20Syntax\x20error\,x20command\x2
SF:0unrecognized\r\n");
=====
NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)
=====
SF-Port443-TCP:V=7.94SVNXT=SSLXI=7KD=4/13XTime=661A5486XP=x86_64-pc-linux-
SF:gnuxr(GetRequest,D8,"HTTP/1.1,0x20200v200K\r\nContent-Type:\x20text/ht
SF:ml\r\nDate:\x20Sat,\x2012\x20Apr\x202024\x2009:47:47\x20GMT\r\nContent-
SF:Length:\x20114\r\n\r\n<!DOCTYPE\x20html><html><head><script>window.onl
SF:oad=function()\{window.location.href="/lander"/}</script></head></h
SF:tml>");
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
OS fingerprint not ideal because: Missing a closed TCP port so results incomplete
No OS matches for host
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 62.23 seconds
```

Scanned results for www.grammarly.io, www.grammarlyaws.com, www.grammarly.ai and they all redirect to same website – Grammarly.com.

Automated Testing

For automated testing, I've opted for OWASP ZAP, a widely used tool within the industry.

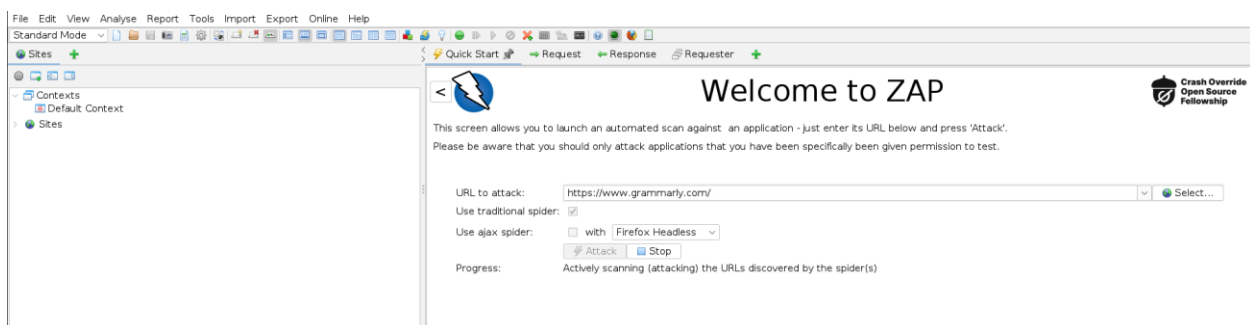
OWASP ZAP

The Open Web Application Security Project Zed Attack Proxy (OWASP ZAP) is a well-known open-source vulnerability scanner recognized for its ability to operate as a Man-in-the-Middle (MITM) proxy. It evaluates various vulnerabilities by examining responses from the web application or server. OWASP ZAP is notably user-friendly and offers customization options through the installation of modules, allowing for efficient management of results.

Within this proxy, there are primarily two types of scans available:

1. **Automated Scan:** Users input the target URL and initiate the attack. The behavior can be customized by selecting the ZAP mode, triggering all scripts against the target to detect vulnerabilities and generate reports accordingly.
2. **Manual Explore:** Users can navigate to the target web application and begin exploration. During manual exploration, ZAP HUD (Heads Up Display) captures each page, while the ZAP proxy records responses.

For this assessment, I am running ZAP on automated mode.



After entering the target URL in the designated textbox, simply click on "Attack" to begin the scanning process. Once finished, you can generate a detailed report of the findings by clicking on "Report."

Below are screenshots illustrating the results obtained after scanning several domains.

Alert counts by risk and confidence

This table shows the number of alerts for each level of risk and confidence included in the report.

(The percentages in brackets represent the count as a percentage of the total number of alerts included in the report, rounded to one decimal place.)

		Confidence				
		User Confirmed	High	Medium	Low	Total
Risk	High	0 (0.0%)	0 (0.0%)	1 (3.8%)	0 (0.0%)	1 (3.8%)
	Medium	0 (0.0%)	4 (15.4%)	2 (7.7%)	1 (3.8%)	7 (26.9%)
	Low	0 (0.0%)	2 (7.7%)	6 (23.1%)	1 (3.8%)	9 (34.6%)
	Informational	0 (0.0%)	2 (7.7%)	3 (11.5%)	4 (15.4%)	9 (34.6%)
	Total	0 (0.0%)	8 (30.8%)	12 (46.2%)	6 (23.1%)	26 (100%)

Alert counts by alert type

This table shows the number of alerts of each alert type, together with the alert type's risk level.

(The percentages in brackets represent each count as a percentage, rounded to one decimal place, of the total number of alerts included in this report.)

Alert type	Risk	Count
PII Disclosure	High	6 (23.1%)
Absence of Anti-CSRF Tokens	Medium	2248 (8,646.2%)
CSP: Wildcard Directive	Medium	2514 (9,669.2%)
CSP: script-src unsafe-inline	Medium	2514 (9,669.2%)
CSP: style-src unsafe-inline	Medium	2514 (9,669.2%)
Content Security Policy (CSP) Header Not Set	Medium	1449 (5,573.1%)
Cross-Domain Misconfiguration	Medium	9 (34.6%)
Missing Anti-clickjacking Header	Medium	2 (7.7%)
Cookie No HttpOnly Flag	Low	820 (3,153.8%)
Cookie Without Secure Flag	Low	818 (3,146.2%)
Cookie with SameSite Attribute None	Low	4 (15.4%)
Cookie without SameSite Attribute	Low	868 (3,338.5%)
Cross-Domain JavaScript Source File Inclusion	Low	29677 (114,142.3%)
Server Leaks Version Information via "Server" HTTP Response Header Field	Low	3257 (12,526.9%)
Strict-Transport-Security Header Not Set	Low	1448 (5,569.2%)

Timestamp Disclosure - Unix	Low	19 (73.1%)
X-Content-Type-Options Header Missing	Low	11 (42.3%)
Authentication Request Identified	Informational	27 (103.8%)
Content Security Policy (CSP) Report-Only Header Found	Informational	2 (7.7%)
Information Disclosure - Sensitive Information in URL	Informational	1187 (4,565.4%)
Information Disclosure - Suspicious Comments	Informational	4217 (16,219.2%)
Loosely Scoped Cookie	Informational	767 (2,950.0%)
Modern Web Application	Informational	2493 (9,588.5%)
Re-examine Cache-control Directives	Informational	2457 (9,450.0%)
Session Management Response Identified	Informational	5572 (21,430.8%)
User Controllable HTML Element Attribute (Potential XSS)	Informational	1478 (5,684.6%)
Total		26

*Please note that these vulnerabilities are rated according to the OWASP risk rating methodology, which can be found in this link. [OWASP Risk Rating Methodology](#).

Below are the vulnerabilities detected within this domain, along with their impacts. The insights provided, including screenshots and remedies, are sourced from the report generated by OWASP ZAP and the Netsparker website. [<https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities>])

PII Disclosure (1)

▼ GET <https://www.grammarly.com/blog/paraphrasing-articles/>

Alert tags

- [OWASP_2021_A04](#)
- [OWASP_2017_A03](#)

Alert description

The response contains Personally Identifiable Information, such as CC number, SSN and similar sensitive data.

Other info

Credit Card Type detected: Mastercard

Request

▼ Request line and header section (689 bytes)

```
GET https://www.grammarly.com/blog/paraphrasing-articles/ HTTP/1.1
host: www.grammarly.com
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36
pragma: no-cache
cache-control: no-cache
referer: https://www.grammarly.com/blog/
Cookie: grauth=AABNc-km_5BxXPzqdbD4WcpusFHbXsFj9JvwP5203JA1UdMWJypk_EAUV2p115xqzh2KA-hEyIx8AJj; csrf-token=AABNc1l8bMEs8Z1t5gtDx8840thw08haknKZYg; gnar_containerId=fbuk8rlm6o3e0h02; funnelType=free; redirect_location=eyJ0eXB1IjoiiIiwibG9jYXRpb24iOiJodHRwczovL3d3dy5ncmFtbWYyY29tL3VwZ3JhZGUvYnVzaW5lc3Mi fQ==; browser_info=CHROME:116:COMPUTER:SUPPORTED:FREEIUM:WINDOWS_10:WINDOWS
```

▼ Request body (0 bytes)

Response

▼ Status line and header section (390 bytes)

```
HTTP/1.1 200 OK
Date: Sun, 14 Apr 2024 07:14:51 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 147192
Connection: keep-alive
Server: nginx/1.22.1
Vary: Accept-Encoding
ETag: W/"23ef8-iTsFBw73UdJV2gIh2yS0r5B8g2c"
Content-Security-Policy: frame-ancestors 'self'
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=31536000
X-Content-Type-Options: nosniff
```

► Response body (147192 bytes)

Evidence

2590291124000974

Solution

Check the response for the potential presence of personally identifiable information (PII), ensure nothing sensitive is leaked by the application.

<https://www.grammarly.com> (4)

CSP: Wildcard Directive (1)

▼ GET <https://www.grammarly.com/>

Alert tags

- [OWASP_2021_A05](#)
- [OWASP_2017_A06](#)

Alert description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Other info

The following directives either allow wildcard sources (or ancestors), are not defined, or are overly broadly defined:

script-src, style-src, img-src, connect-src, frame-src, font-src, media-src, object-src, manifest-src, worker-src, form-action

The directive(s): form-action are among the directives that do not fallback to default-src, missing/excluding them is the same as allowing anything.

Request

▼ Request line and header section (236 bytes)

```
GET https://www.grammarly.com/ HTTP/1.1
host: www.grammarly.com
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0
Safari/537.36
pragma: no-cache
cache-control: no-cache
```

► Request body (0 bytes)

Response

► Status line and header section (1083 bytes)

► Response body (336721 bytes)

Parameter

Content-Security-Policy

Evidence

frame-ancestors 'self' *.grammarly.com

Solution

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.

CSP: script-src unsafe-inline (1)

▼ GET https://www.grammarly.com/

Alert tags

- [OWASP 2021 A05](#)
- [OWASP 2017 A06](#)

Alert description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Other info

script-src includes unsafe-inline.

Request

▼ Request line and header section (236 bytes)

```
GET https://www.grammarly.com/ HTTP/1.1
host: www.grammarly.com
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0
Safari/537.36
pragma: no-cache
cache-control: no-cache
```

► Request body (0 bytes)

Response

► Status line and header section (1083 bytes)

► Response body (336721 bytes)

Parameter

Content-Security-Policy

Evidence

frame-ancestors 'self' *.grammarly.com

Solution

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.

CSP: style-src unsafe-inline (1)

▼ GET https://www.grammarly.com/

Alert tags

- [OWASP 2021 A05](#)
- [OWASP 2017 A06](#)

Alert description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Other info

style-src includes unsafe-inline.

Request

▼ Request line and header section (236 bytes)

```
GET https://www.grammarly.com/ HTTP/1.1
host: www.grammarly.com
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0
Safari/537.36
pragma: no-cache
cache-control: no-cache
```

▼ Request body (0 bytes)

Response

► Status line and header section (1083 bytes)

► Response body (336721 bytes)

Parameter

Content-Security-Policy

Evidence

frame-ancestors 'self' *.grammarly.com

Solution

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.

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

▼ GET <https://www.grammarly.com/upgrade/business/try>

Alert tags

- [OWASP 2021 A05](#)
- [OWASP 2017 A06](#)

Alert description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Request

- Request line and header section (721 bytes)
- Request body (0 bytes)

Response

- Status line and header section (1539 bytes)
- Response body (279305 bytes)

Solution

Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.

Missing Anti-clickjacking Header (1)

▼ GET <https://www.grammarly.com/upgrade/business/try>

Alert tags

- [OWASP 2021 A05](#)
- [WSTG-v42-CLNT-09](#)
- [OWASP 2017 A06](#)

Alert description

The response does not include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options to protect against 'ClickJacking' attacks.

Request

▼ Request line and header section (721 bytes)

```
GET https://www.grammarly.com/upgrade/business/try HTTP/1.1
host: www.grammarly.com
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0
Safari/537.36
pragma: no-cache
cache-control: no-cache
referer: https://www.grammarly.com/business/pricing
Cookie: grauth=AABNc-
km_5BxXPzqdbD4WcpusFHbXsFj9JvwP5203JA1UdMwJypyk_EAUV2p115xq
zh2KA-hEyIx8AJj; csrf-
token=AABNc1l8bMEs8Z1t5gtDx8840thw08haknKZYg;
gnar_containerId=fbuk8rIm6o3e0h02; funnelType=free;
redirect_location=eyJ0eXB1IjoiiwibG9jYXRpb24iOiJodHRwczovL
3d3dy5ncmFtbWYyY29tL3NpZ251cD9mbG93X3NvdXJjZT1jaXRhdGlv
bl9nZW51cmF0b3IifQ==;
browser_info=CHROME:116:COMPUTER:SUPPORTED:PREMIUM:WINDOWS
_10:WINDOWS
```

▼ Request body (0 bytes)

Response

► Status line and header section (1539 bytes)

► Response body (279305 bytes)

Parameter

x-frame-options

Solution

Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app.

If you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. Alternatively consider implementing Content Security Policy's "frame-ancestors" directive.

Cookie Without Secure Flag (1)

▼ GET https://www.grammarly.com/

Alert tags

- [OWASP 2021 A05](#)
- [WSTG-v42-SESS-02](#)
- [OWASP 2017 A06](#)

Alert description

A cookie has been set without the secure flag, which means that the cookie can be accessed via unencrypted connections.

Request

▼ Request line and header section (236 bytes)

```
GET https://www.grammarly.com/ HTTP/1.1
host: www.grammarly.com
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0
Safari/537.36
pragma: no-cache
cache-control: no-cache
```

▼ Request body (0 bytes)

Response

► Status line and header section (1083 bytes)

► Response body (336721 bytes)

Parameter

gnar_containerId

Evidence

Set-Cookie: gnar_containerId

Solution

Whenever a cookie contains sensitive information or is a session token, then it should always be passed using an encrypted channel. Ensure that the secure flag is set for cookies containing such sensitive information.

Cookie with SameSite Attribute None (1)

▼ GET https://www.grammarly.com/

Alert tags

- [OWASP 2021 A01](#)
- [WSTG-v42-SESS-02](#)
- [OWASP 2017 A05](#)

Alert description

A cookie has been set with its SameSite attribute set to "none", which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.

Request

▼ Request line and header section (236 bytes)

```
GET https://www.grammarly.com/ HTTP/1.1
host: www.grammarly.com
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0
Safari/537.36
pragma: no-cache
cache-control: no-cache
```

▼ Request body (0 bytes)

Response

► Status line and header section (1083 bytes)

► Response body (336721 bytes)

Parameter

grauth

Evidence

Set-Cookie: grauth

Solution

Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.

Content Security Policy (CSP) Report-Only Header Found (1)

▼ GET <https://www.grammarly.com/upgrade/business/try>

Alert tags

- [OWASP 2021 A05](#)
- [OWASP 2017 A06](#)

Alert description

The response contained a Content-Security-Policy-Report-Only header, this may indicate a work-in-progress implementation, or an oversight in promoting pre-Prod to Prod, etc.

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Request

► Request line and header section (721 bytes)

▼ Request body (0 bytes)

Response

► Status line and header section (1539 bytes)

► Response body (279305 bytes)

Solution

Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.

Information Disclosure - Suspicious Comments (1)

▼ GET https://www.grammarly.com/

Alert tags

- [OWASP 2021 A01](#)
- [WSTG-v42-INFO-05](#)
- [OWASP 2017 A03](#)

Alert description

The response appears to contain suspicious comments which may help an attacker. Note: Matches made within script blocks or files are against the entire content not only comments.

Other info

The following pattern was used: `\bQUERY\b` and was detected in the element starting with: `"<script id="__NEXT_DATA__" type="application/json">{"props":{"pageProps":{"requestData":{"pageEntry":{"metadata":{"tags":["sys"]`, see evidence field for the suspicious comment/snippet.

Request

▼ Request line and header section (236 bytes)

```
GET https://www.grammarly.com/ HTTP/1.1
host: www.grammarly.com
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0
Safari/537.36
pragma: no-cache
cache-control: no-cache
```

► Request body (0 bytes)

Response

► Status line and header section (1083 bytes)

► Response body (336721 bytes)

Evidence

query

Solution

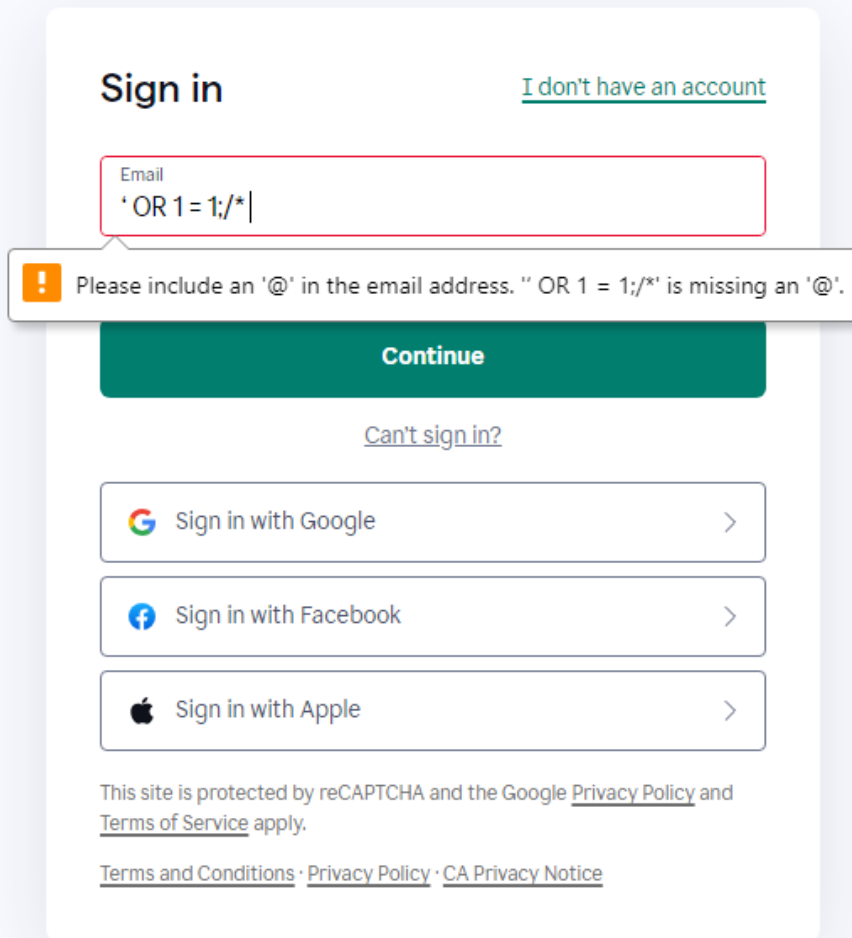
Remove all comments that return information that may help an attacker and fix any underlying problems they refer to.

Manual Testing

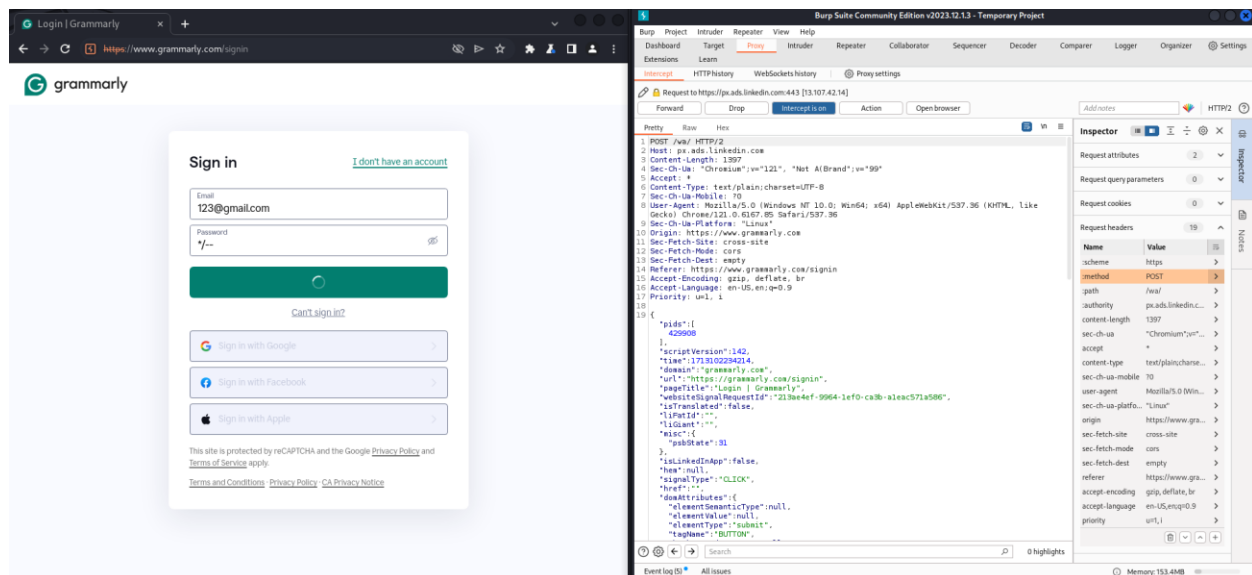
SQL injection

If this page is vulnerable to SQL injection attack the Query will be executed as:

SELECT * FROM users WHERE email address = ' OR 1 = 1;/* AND password = */--



The screenshot shows a Google sign-in interface. At the top, it says "Sign in" with a link "I don't have an account". Below this is an email input field containing the text "' OR 1 = 1;/*". A red border highlights the input field. A tooltip message appears below the field, stating: "Please include an '@' in the email address. "' OR 1 = 1;/*' is missing an '@'." Below the tooltip is a green "Continue" button. Under the button is a link "Can't sign in?". Further down are three social sign-in buttons: "Sign in with Google", "Sign in with Facebook", and "Sign in with Apple". At the bottom, there is a disclaimer: "This site is protected by reCAPTCHA and the Google Privacy Policy and Terms of Service apply." followed by links for "Terms and Conditions", "Privacy Policy", and "CA Privacy Notice".



They have implemented password policies, so that I tried to intercept the request and do the SQL injection but failed and it is secured well against SQLi injection.

Checking for Insecure HTTP methods

Specific HTTP methods, particularly DELETE and PUT, pose potential security threats to the server. The DELETE method has the capability to eliminate resources from the server, while the PUT method can upload and execute files. However, these methods need to be used with care as they could potentially jeopardize the Confidentiality, Integrity, and Availability of the server and its users.

To determine the HTTP methods that are supported, I utilized the Burp proxy to capture requests and the Repeater to alter the request method. This allowed me to dispatch the altered

requests to the server and scrutinize the responses.

The image shows a web browser window displaying the Grammarly website (https://www.grammarly.com) and a Burp Suite interface capturing an HTTP request and response. The browser window shows the Grammarly homepage with navigation links (Product, Work, Education, Pricing, Resources) and a green button labeled "Get Grammarly It's free". The Burp Suite interface shows a captured HTTP request and response. The request is a POST method to the endpoint /wa/ HTTP/2. The response is a 204 No Content status. The Burp Suite interface also shows the request and response details, including headers, cookies, and a list of request parameters.

Request Details:

- Method: POST
- URL: https://www.grammarly.com/wa/
- Headers: Host: px.ads.linkedin.com, Content-Length: 1684, Sec-CH-UA: Chrome, Sec-CH-UA-Mobile: 70, User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85 Safari/537.36, Origin: https://www.grammarly.com, Sec-Fetch-Site: cross-site, Sec-Fetch-Mode: cors, Sec-Fetch-Dest: empty, Referer: https://www.grammarly.com/, Accept-Encoding: gzip, deflate, br, Accept-Language: en-US,en;q=0.9, Priority: u=1, i

Response Details:

- Status: 204 No Content
- Headers: Vary: Origin, Set-Cookie: bcookie=...; domain=linkedin.com; Path=/; Secure; Expires=Mon, 14-Apr-2025 13:58:55 GMT; SameSite=None, Set-Cookie: lidc=...; Expires=Mon, 15-Apr-2024 13:58:55 GMT, Access-Control-Allow-Origin: https://www.grammarly.com, Access-Control-Allow-Credentials: true, X-LL-Palrsc: prod:11d, X-LL-Pop: af6-prod-11d1-x, X-LL-Proto: http/2, X-LL-Url: AAYWduFu37dYHETWduFu=, X-Cache: COMPT, NOCACHE, X-Medge-Ref: Ref A: 402D01AFD0C462992DF0503ADAA118F Ref B: 52ND0ED0203D Ref C: 2024-04-14T13:58:54Z, Date: Sun, 14 Apr 2024 13:58:54 GMT

Inspector Details:

- Request attributes: 2
- Request query parameters: 0
- Request cookies: 0
- Request headers: 19

The server only supports the POST method. Therefore, no insecure methods were utilized on the server.

Conclusion

The web application, along with its associated authorization gateway, showcases a commendable level of upkeep and control, particularly from a cybersecurity standpoint. While there have been occurrences of information exposure, these can be suitably handled, and their corresponding risk quotient is evaluated to be minimal. It's worth noting that all detected security weaknesses fell into the categories of either being Informational or of Low severity. Moreover, the vulnerabilities that were categorized as Low are not readily exploitable, as has been evidenced in the past.

It's also important to highlight that the application's security measures are regularly updated to keep up with the evolving threat landscape. Regular security audits are conducted to identify and rectify any potential vulnerabilities. The application also employs a robust encryption mechanism to protect sensitive data and ensure secure communication. Additionally, the application's user authentication process is designed to prevent unauthorized access, further enhancing its security posture. Despite the presence of some low-level vulnerabilities, the overall security of the web application and its authorization portal is well-maintained and managed. This reflects the commitment to security and the proactive approach taken towards identifying and addressing potential security issues.

.

References

[OWASP Top Ten | OWASP Foundation](#)

[WSTG - Stable | OWASP Foundation](#)

[ZAP \(zapproxy.org\)](#)

[GitHub - aboul3la/Sublist3r: Fast subdomains enumeration tool for penetration testers](#)

[GitHub - tomnomnom/httpprobe: Take a list of domains and probe for working HTTP and HTTPS servers](#)

[Kali Tools | Kali Linux Tools](#)

[Netcraft | Leader in Phishing Detection, Cybercrime Disruption and Website Takedown](#)

[Nmap: the Network Mapper - Free Security Scanner](#)

