НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

До захисту допущено
В.о. завідувача кафедри

_____ **Микола ГРАЙВОРОНСЬКИЙ**
(підпис)

«_____»_____2021 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною программою «Системи, технології та математичні методи кібербезпеки»
спеціальності: 125 «Кібербезпека»

на тему: Технологія виявлення вразливих субдоменів з метою отримання контролю над ними

Виконав (-ла): здобувач вищої освіти **IV** курсу, групи ФБ-74
(шифр групи)

Панчук Олександр Валентинович _____
(прізвище, ім'я, по батькові) (підпис)

Керівник к.т.н., доцент Гальчинський Леонід Юрійович _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Рецензент _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище, ім'я, по батькові) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Здобувач вищої освіти _____
(підпис)

Київ - 2021 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 125 «Кібербезпека»
Освітньо-професійна програма «Системи, технології та математичні методи кібербезпеки»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Микола ГРАЙВОРОНСЬКИЙ
(підпис)
«___»_____2021 р.

ЗАВДАННЯ
на дипломну роботу здобувачу вищої освіти

Панчук Олександр Валентинович _____
(прізвище, ім'я, по батькові)

1. Тема роботи Технологія виявлення вразливих субдоменів з метою отримання контролю над ними _____
_____
_____ ,
керівник роботи к.т.н., доцент Гальчинський Леонід Юрійович _____ ,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від « » _____ 2021 р. №

2. Термін подання здобувачем вищої освіти роботи 07 червня 2021 р.

3. Вихідні дані до роботи технології пошуку субдоменів за допомогою даних із загальнодоступних джерел та методів грубої сили _____
_____

4. Зміст роботи

1. Опис теоретичних відомостей та огляд літератури
2. Інструменти сканування субдоменів
3. Програма та експлуатація
4. Додаток

_____

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо) _____

6. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів дипломної роботи	Примітка
1	Формулювання теми дипломного проекту, визначення мети та постановка задач.	13.04.21-20.04.21	виконано
2	Узгодження теми з дипломним керівником.	21.04.21-23.04.21	виконано
3	Визначення структури дипломного проекту.	24.04.21-29.04.21	виконано
4	Робота над першим розділом; дослідження існуючих методів.	30.04.21-04.05.21	виконано
5	Робота над другим розділом вибір оптимального методу пошуку субдоменів	05.05.21-10.05.21	виконано
6	Робота над третім розділом написання програми	11.05.21-23.05.21	виконано
7	Робота над третім розділом пошук розвиту вразливості	24.05.21-27.05.21	виконано
8	Підготовка ілюстративного матеріалу; оформлення текстової і графічної частини.		
9	Попередній розгляд дипломного проекту на кафедрі		

Здобувач вищої освіти _____ __Олександр Панчук_____
(підпис) (Власне ім’я, ПРІЗВИЩЕ)

Керівник роботи _____ __Леонід Гальчинський__
(підпис) (Власне ім’я, ПРІЗВИЩЕ)

АНОТАЦІЯ

Дипломний проект виконаний на 50 сторінках тексту, було використано 2 таблиці та 15 рисунків. Складається з 4 розділів, вступу, висновків, переліку використаної літератури та додатків. Зміст роботи відповідає затвердженому завданню. Робота присвячена розгляду сучасних методів знаходження вразливих субдоменів з метою отримання контролю.

Метою даної роботи є збір та обробка вразливих субдоменів до перехвату контролю та автоматизація пошуку таких субдоменів для підвищення ефективності контролю потенційно вразливих субдоменів.

У кваліфікаційній роботі було досліджено спосіб перехвату контролю над субдоменами цільвого домена. Була показана критичність цієї вразливості та її наслідки, а сама реалізації інших вразливостей за допомогою Subdomain Takeover.

Результатом роботи є написана програма мета якої автоматизувати процеси пошуку вразливих доменів для швидкого отримання станів наших субдоменів, дану програму можуть використовувати як і компанії з великою розгалуженістю та і для атаки.

В результаті виконання коду, було проведено основні кроки пошуку на мові Node.js.

Ключові слова: домен, субдомен, DNS, CNAME, Subdomain takeover.

ANNOTATION

Thesis project is made on 50 pages of text, 2 tables and 15 figures were used. It consists of 4 sections, introductions, conclusions, list of references and appendices. The content of the work corresponds to the approved task. The work is devoted to the consideration of modern methods of finding vulnerable subdomains in order to gain control.

The purpose of this work is to collect and process vulnerable subdomains before interception and automate the search for such subdomains to improve the control of potentially vulnerable subdomains.

The qualification work investigated the method of interception of control over the subdomains of the target domain. The vulnerability of this vulnerability and its consequences was shown, as well as the implementation of other vulnerabilities with the help of Subdomain Takeover.

The result is a written program that aims to automate the search for vulnerable domains to quickly obtain the states of our subdomains, this program can be used as companies with a large branching and for attack.

As a result of executing the code, the basic search steps were performed in Node.js.

Keywords: domain, subdomain, DNS, CNAME, Subdomain takeover.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

DNS (Domain Name System) - Система доменних імен.

CNAME (canonical name record) використовується для перенаправлення на інше ім'я.

React - JavaScript-бібліотека з відкритим вихідним кодом для розробки призначених для користувача інтерфейсів.

Node.js - програмна платформа, заснована на движку V8 (здійснює трансляцію JavaScript в машинний код).

Vue.js - JavaScript-фреймворк з відкритим вихідним кодом для розробки призначених для користувача інтерфейсів.

REST API - архітектурний стиль взаємодії компонентів розподіленого додатка в мережі.

XSS (Cross-Site Scripting) - тип атаки на веб-системи, що полягає у впровадженні в видається веб-системою сторінку шкідливого коду.

CSRF (Cross-site request forgery) - вид атак на відвідувачів веб-сайтів, який використовує недоліки протоколу HTTP.

CORS (Cross-origin resource sharing) - технологія сучасних браузерів, яка дозволяє надати веб-сторінок доступ до ресурсів іншого домену.

IP - маршрутизації протокол мережевого рівня стека TCP / IP.

URI - уніфікований (однаковий) ідентифікатор ресурсу.

AWS (Amazon Web Services) - комерційне публічне хмара.

FTP (File Transfer Protocol) - протокол передачі файлів по мережі.

NS (Authoritative name server)- вказує на DNS-сервери, які відповідають за зберігання інших ресурсних записів домену.

MX - це запис, що відповідає за сервер, через який буде працювати пошта.

WEB (World Wide Web) - розподілена система, що надає доступ до пов'язаних між собою документів, розташованим на різних комп'ютерах, підключених до мережі Інтернет.

HTTP (HyperText Transfer Protocol)) - протокол прикладного рівня передачі даних.

HTTPS (HyperText Transfer Protocol Secure) - розширення протоколу HTTP для підтримки шифрування з метою підвищення безпеки.

RegExp - формальна мова пошуку і здійснення маніпуляцій з підрядками в тексті, заснований на використанні метасимволов.

Promise – об’єкт який використовується для відкладених і асинхронних обчислень.

JSON (JavaScript Object Notation) - текстовий формат обміну даними, заснований на JavaScript.

ЗМІСТ

ВСТУП

В наш час компанії ростуть з великою швидкістю майже всі з них використовують для полегшення бізнес процесів веб-сервіси, на яких вони зберігають інформацію про своїх користувачів. Це породжує велику складність веб-сайтів з розгалуженою структурою та використанням сучасних веб-технологій (React, Node.js, Vue.js, Angular, REST API та інші). Сайти вже давно перестали бути примітивними та не зосереджені на головній сторінці, ми не хочемо щоби наш сайт мав одну направленість, особливо якщо це сайт електронної комерції. Люди не читають сайти як книжки, від початку до кінця. Це одна з основних задач субдоменів розвантажити головну сторінку, розподілити функціонал та назначення сторінок таких як мобільна версія, форум, різноманітні категорії, регіональні розгалуження, нового функціоналу та інші, що породжує велику кількість субдоменів.

Слідкувати за такою кількістю ресурсів складно, під час розробки деякі субдомени можуть бути зарезервовані раніше ніж будуть використані або просто закинуті, все це дає зловмисникам можливість отримання контролю над субдоменом, дізнавшись DNS запису CNAME такого субдомену. Через що, зловмисник може отримати доступ до субдомену, добавляти свої файли для різноманітних маніпуляцій таких як фішінг, крадіжка cookie файлів з цільового домену, експлуатація інших вразливостей (XSS, CSRF, обхід CORS).

Метою даної роботи є збір та обробка вразливих субдоменів до перехвату контролю та автоматизація пошуку таких субдоменів для підвищення ефективності контролю потенційно вразливих субдоменів.

Об’єктом дослідження є методи знаходження вразливих субдоменів.

1 ОПИС ТЕОРЕТИЧНИХ ВІДОМОСТЕЙ ТА ОГЛЯД ЛІТЕРАТУРИ

1.1 Домен

Доменне ім’я - це ідентифікаційний рядок, який визначає сферу адміністративної автономії, повноважень чи контролю в Інтернеті. Доменні імена використовуються у різних мережевих контекстах для цільових імен та адресних цілей. Загалом, доменне ім'я ідентифікує мережевий домен або представляє ресурс Інтернет-протоколу (IP), такий як персональний комп'ютер, що використовується для доступу до Інтернету, серверний комп'ютер, що розміщує веб-сайт, або сам веб-сайт або будь-яка інша служба, про яку повідомляється через Інтернет [28].

Доменне ім’я складається з двох основних елементів. Наприклад, доменне ім’я google.com складається з імені веб-сайту (google) та розширення доменного імені (.com).

Доменні імена також використовуються як прості ідентифікаційні ярлики для позначення права власності чи контролю над ресурсом. Такими прикладами є ідентифікатори області, що використовуються в протоколі ініціювання сеансу (SIP), ключі домену, що використовуються для перевірки доменів DNS в системах електронної пошти, та в багатьох інших уніфікованих ідентифікаторах ресурсів (URI).

Важливою функцією доменних імен є надання легко впізнаваних імен для адресних Інтернет-ресурсів. Ця абстракція дозволяє переміщувати будь-який ресурс в інше фізичне розташування в адресній топології мережі, глобально чи локально.

1.2 Subdomain

Subdomain – це частина основного домена високого рівня. Наприклад, translate.google.com. У цьому випадку (translate) – це субдомен, (google) – основний домен, а (.com) – домен верхнього рівня [11].

Зазвичай використання субдомену є створення тестової або індексної версії веб-сайту. Часто розробники тестують нові плагіни та оновлення на веб-сайті піддомену, перш ніж публікувати їх в Інтернеті(developer.exapmle.com). Також поширеним використанням субдомену є створення сайту електронної комерції. Часто компанії хочуть окремий субдомен для обробки транзакцій. Також компанії використовують субдомени для своїх мобільних веб-сайтів (m.exapmle.com), сайтів, що відповідають місцезнаходженню (ua.example.com). Ми можете використовувати субдомен для обслуговування певної групи користувачів на нашому веб-сайті, наприклад, «blog.example.com», «user. example.com» тощо.
Субдомени можуть бути дуже корисними для ефективнішої організації вмісту веб-сайту. Правильне використання субдомену не впливає на SEO вашого основного веб-сайту.

За винятком їх використовуюсь для обслуговування контенту на веб-сайту як приклад це може бути завантаження скриптів (subdomain.example.com/script.js) або якщо продукт використовує статичні данні за допомогою AWS S3 bucket (example.s3.amazonaws.com) хоча ці домени та не належать нашій цілі, але вони використовують данні з цих ресурсів це означає, що якщо ми перехопимо ці сторінки зможемо впливати та на сам домен. Якщо на веб-сайту існує (<script src=”subdomain.example.com/script.js”></script>) то при захваті subdomain посилання з скрипта ми можемо створити свій скрипт в якому буде знаходитися Cross-site scripting(XSS) який буде спрацьовувати кожен раз коли веб-сайт буде звертатися до цього файлу тобто це буде Stored XSS.

1.3 DNS

DNS (Domain Name System) - це ієрархічна та децентралізована система імен для комп’ютерів, служб чи інших ресурсів, підключених до Інтернету або приватної мережі. Він пов'язує різну інформацію з доменними іменами, призначеними кожному з суб'єктів-учасників [21].

Ми отримуємо доступ до інформації в Інтернеті через доменні імена, такі як google.com або facebook.com. Веб-браузери взаємодіють через адреси IP-протоколу. DNS перекладає доменні імена на IP-адреси, щоб браузери могли завантажувати Інтернет-ресурси. Кожен пристрій, підключений до Інтернету, має унікальну IP-адресу, за допомогою якої інші машини знаходять пристрій. DNS-сервери усувають необхідність запам'ятовувати людям IP-адреси, такі як 172.217.20.14 (в IPv4), або складніші новіші буквено-цифрові IP-адреси, такі як 2dfc:0:0:0:0217:cbff:fe8c:0 (в IPv6).

Для поліпшення продуктивності та надійності запитів даних DNS може кешуватися. Кешування DNS передбачає зберігання даних ближче до запитуючого клієнта, щоб запит DNS можна було вирішити раніше, а інші ланцюжки пошуку DNS, можна уникнути, тим самим покращуючи час завантаження та зменшуючи пропускну здатність. Дані DNS можна кешувати в різних місцях, кожен з яких буде зберігати записи DNS протягом встановленого періоду часу, визначеного часом життя (TTL). Сучасні веб-браузери вже мають можливість кешування записів DNS протягом певного періоду часу. Чим ближче кешування DNS відбувається до веб-браузера, тим менше кроків обробки повинно бути зроблено для того, щоб перевірити кеш-пам'ять і зробити правильні запити на IP-адресу. Коли робиться запит на запис DNS, кеш-пам’ять браузера є першим місцем перевірки для запитуваного запису.

1.4 CNAME

CNAME – це тип запису DNS, яка зв’язує субдомен з доменом. Тобто вказує, що одна доменна назва вказує на субдомен іншого домена. Кожна система, на якій знаходиться веб-сайт, повинен мати IP-адресу для підключення до інтернету. DNS перетворює доменну назву в свою IP-адресу, але іноді декілька доменних імен відносяться до одної IP-адреси. Цю проблему вирішує CNAME, на машині може бути необмежена кількість субдоменів, але для кожного субдомені повинен бути свій CNAME. За допомогою цього тепер на одній IP-адресі можна тримати декілька серверів з різними портами, наприклад FTP та WEB та кожен буде мати власний запис в DNS [14].

Коли ми хочемо спрямувати користувачів до тієї самої інформації, що подається за окремою URL-адресою. Наприклад, для власників веб-сайтів, які використовують домен, такий як example.com без будь-якого субдомену, канонічне ім’я може використовуватися для додавання www на початок URL-адреси. Отже, коли користувач заходить на ваш сайт через www.example.com, він побачить ту саму інформацію, так як він зайшов до вас через example.com.

CNAME має декілька обмежень:

1. Не можуть вказувати на IP-адресу.
2. Не можуть співіснувати з будь-якими іншими даними.
3. Записи NS і MX ніколи не повинні вказувати на псевдонім CNAME.

Виходячи з обмежень, згаданих вище, CNAME не можна використовувати для субдоменів кореневого домену через його неможливість співіснувати з іншими даними. Однак існують деякі рішення, які допомагають подолати цю проблему.

Наприклад, постачальники DNS, такі як DNS Simple або Cloudflare, впровадили власну версію того, що називається Alias Record. Запис псевдоніма дозволяє поєднати кореневий домен з іншою службою, зберігаючи при цьому можливість співіснувати з іншими даними. Це може виявитися корисним у

випадках, коли ми хочемо об’єднати кореневий домен із такою службою, як Heroku або GitHub (наприклад, example.com username.github.com).

1.5 GitHub

Git - це система управління версіями, яка дозволяє відстежувати зміни у файлах. Він повністю заснований на файлах, тобто немає додаткового програмного забезпечення або додатків, крім самого Git.

Використовуючи Git, можна повернути файли до попередніх версій, відновити видалені файли, видалити додані файли і навіть відстежити, куди було введено певний рядок коду.

Git створює папку .git (у поточній папці) для зберігання реквізитів файлової системи - ця папка містить усі дані, необхідні для відстеження ваших файлів і відома як сховище або репозиторій.

Git відслідковує зміни файлів користувачем, створюючи точку збереження, або в термінах Git - фіксацію. Кожна фіксація робить знімок поточної файлової системи, а не зберігає лише зміни, внесені з моменту останнього виконання. Це дозволяє витягнути фіксацію, і вся історія не потребує відновлення файлової системи.

1.6 Dig

Dig (Domain Information Groper) - це інструмент для пошуку серверів імен DNS. Він виконує пошук DNS і відображає відповіді, які повертаються з запитуваних серверів імен. З допомогою dig ми можемо виконати будь-який дійсний запит DNS, найпоширенішими з яких є: A, TXT, MX, NS, CNAME [20].

```
panchuk@aleksandr-panchuk:~/Рабочий стол/Sublist3r$ dig @8.8.8.8 translete.google.com

; <<>> DiG 9.16.1-Ubuntu <<>> @8.8.8.8 translete.google.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 32047
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;translete.google.com.          IN      A

;; AUTHORITY SECTION:
google.com.             59      IN      SOA     ns1.google.com. dns-admin.google.com. 373752905 900 900 1800 60

;; Query time: 52 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Сб мая 15 17:20:24 EEST 2021
;; MSG SIZE  rcvd: 99
```

Рисунок 1.1 - Робота Dig

В нашому випадку dig потрібен для знаходження DNS записі CNAME. Для подальшого перехопленню контролю над вказаним субдоменом.

1.7 Subdomain takeover

Subdomain takeover, це вразливість яка відбувається, коли зловмисник отримує контроль над субдоменом цільового домену. Як правило, це трапляється, коли субдомен має канонічне ім'я (CNAME) у системі доменних імен (DNS), але жоден хост не забезпечує вміст для нього, тобто код відповіді після переходу на таких субдомен буде 404. Це може статися, коли віртуальний хост ще не опублікований, або віртуальний хост видалено. Зловмисник може взяти на себе цей субдомен, надавши власний віртуальний хост, а потім розмістивши для нього власний вміст [7].

Якщо зловмисник може це зробить, він може потенційно читати файли cookie, встановлені з основного домену, виконувати XSS або обходити Content Security Policy (CSP), тим самим дозволяючи їм захоплювати захищену інформацію (включаючи логіни) або надсилати шкідливий вміст нічим не підозрюючим користувачам. Якщо процес створення або виведення з експлуатації віртуального хоста не обробляється належним чином, зловмисник може отримати можливість отримати контроль над субдоменом.

Зловмисник налаштовує віртуальний хост для імені субдомену, який ми придбали у хостинг-провайдера, перш ніж ми це зробили. Припустимо, ми контролюєте домен example.com. Ми хочемо додати свій блог за адресою blog.example.com та вирішили використовувати хостинг-провайдера.

У нас можуть бути такі кроки:

1. Реєструємо ім'я "blog.example.com" у реєстратора доменів.
2. Встановлюємо записи DNS для прямих браузерів, які хочуть отримати доступ до blog.example.com, щоб вони перейшли на віртуальний хост.
3. Створюємо віртуального хоста у хостинг-провайдера.

Якщо хостинг-провайдер не дуже обережно перевіряє, що суб’єкт, який налаштовує віртуальний хост, насправді є власником імені субдомену, зловмисник який швидше ніж ми створить віртуальний хост із тим самим провайдером хостингу, використовуючи наше ім’я субдомену, отримає до нього доступ. У такому випадку, як тільки ми налаштуємо DNS на кроці 2, зловмисник може розмістити вміст у нашому субдомені.

1.8 Статистка

Щорічна статистика публічно оприлюднених даних про поглинання субдоменів, зібраних із платформи HackerOne, проілюстрована на Рисунок 1.2. [6]

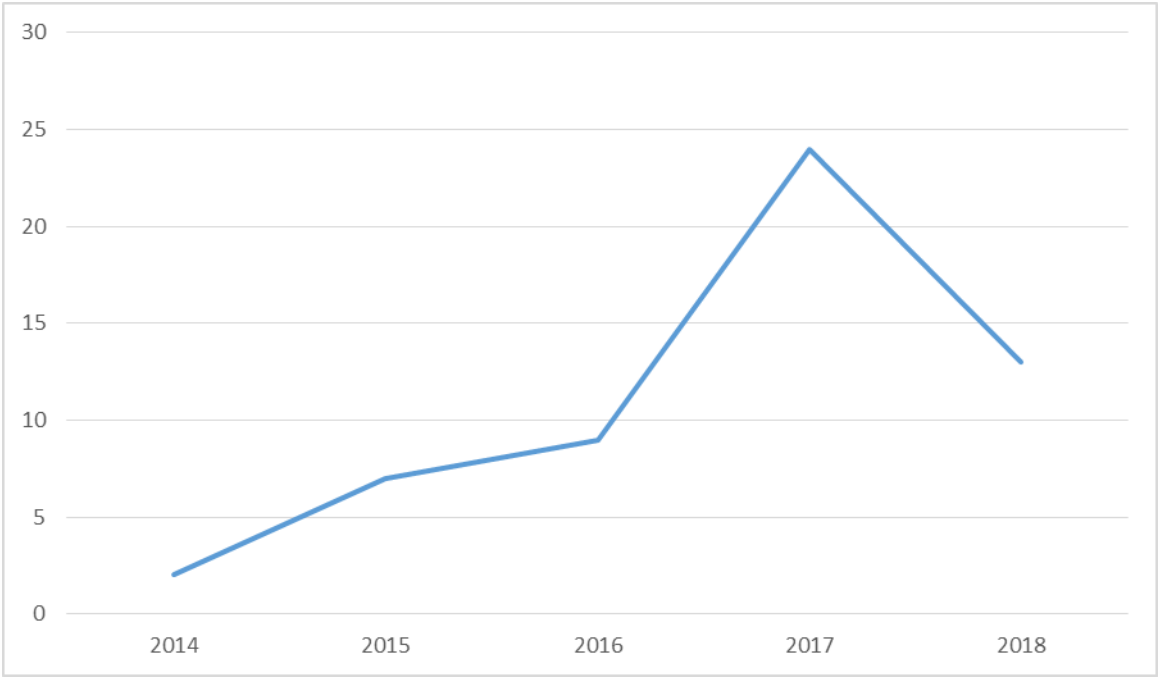

Рисунок 1.2 - Щорічна статистика публічно оприлюднених звітів про поглинання субдоменів

Дослідники з веб-сайту Vullnerability.com змогли захопити понад 670 субдоменів, які раніше використовувала компанія Microsoft, але згодом забула про них, серед них є такі:

- identityhelp.microsoft.com
- mybrowser.microsoft.com
- web.visualstudio.com / webeditor.visualstudio.com
- data.teams.microsoft.com
- sxt.cdn.skype.com
- download.collaborate.microsoft.com
- incidentgraph.microsoft.com
- admin.recognition.microsoft.com

Дані субдомени виглядають надійними і при фішінг-атаці користувачі були б схильні їм довіряти. Варто відмітити, що префікси даних субдоменів великих компаній, таких як microsoft.com та skype.com, є ознакою того, що вони ненадійні. Дослідники пропонують приклади, які включають пропозиції встановити розширення для шпигунства у своєму браузері, фішинг облікових даних

підприємства за допомогою підробленої сторінки входу або прохання завантажувати конфіденційні документи на data.teams.microsoft.com за допомогою програми Teams. Вони навіть можуть пошкодити субдомен, пов’язаний з більшим доменом.

Основною проблемою є слабке управління DNS, в даному випадку компанією Microsoft; це проблема, яка була посилена широким розповсюдженням субдоменів, що використовуються в хмарних послугах. По-перше, зловмисники шукають втрачені субдомени, перейшовши до одного, який, на їх думку, може бути захоплений за допомогою скануючого інструменту. Якщо вони отримують помилку 404 сторінку не знайдено, вони отримують потенційний субдомен для захоплення. Наприклад, зловмисник отримує помилку 404 для покинутого магазину на shop.example.org. Він не може редагувати записи DNS для цього сайту, оскільки він не є власником домену example.org. Натомість він перевіряє, чи є субдомен для іншого домену, над яким він міг би взяти контроль, зазначеного записом CNAME. Якщо CNAME вказує на доменне ім’я, право власності якого втратило силу, він може спробувати придбати цей домен і використовувати його для розміщення шкідливого веб-сайту.

Однак часто CNAME вказує на субдомен на хостинговій службі, такій як Azure, що дозволяє користувачам створювати веб-сайти, використовуючи субдомени .azurewebsites.net. Якщо субдомен Azure у записі CNAME більше не використовується, зловмисник може спробувати отримати його. Він може налаштувати віртуальну машину в обліковому записі Microsoft Azure, встановити веб-сервер, який видає клон цільового сайту, і додати субдомен Azure як власний домен, який вказує на нього. В результаті цього немає жодної перевірки, попередження для Microsoft про те, що один із їхніх старих субдоменів захоплено, що не є простим способом для систем безпеки підприємств виявити, що їх справжній домен став шкідливим.

Захист від цього полягає в очищенні записів DNS для субдомену, але велика кількість встановлених записів, які потім не використовуються, означає, що цей спосіб не завжди працює. Проблема поглинання субдоменів існує багато років і

може торкнутися субдоменів, що належать будь-якій компанії на будь-якій хмарній платформі, а не тільки Microsoft. Однак проблема вразливих субдоменів Microsoft є актуальною вже декулька років, адже про 280 таких вразливостей було повідомлено в період 2017-2019 рр., і компанія Microsoft виправила лише деякі з них.

Висновки до розділу 1

У цьому розділі було описано фундаментальні теоретичні основи, які є необхідними для розуміння матеріалу, який подано у наступних розділах. Також вони утворюють необхідну базу для набуття та реалізації практичних навичок.

Ознайомились що, вразливість розповсюджена та виникає в великих компаніях, що приводить до дискредитації компанії.

Для перешкоджання вразливості є необхідність швидкого сканування домену на наявність вразливих доменів, що приведе до зменшення можливості перехоплення наших субдоменів.

2 ІНСТРУМЕНТИ СКАНУВАННЯ СУБДОМЕНІВ

2.1 Google Hacking

Google Hacking , є цінним ресурсом для дослідників безпеки. Для звичайної людини Google - це лише пошукова система, яка використовується для пошуку тексту, зображень, відео та новин. Однак, це є корисним інструментом [19].

Ми не можемо зламати веб-сайти безпосередньо за допомогою Google, але оскільки він має надзвичайні можливості сканування веб-сайтів, він може проіндексувати майже все на нашому веб-сайті, включаючи конфіденційну інформацію. Це означає, що ми можемо розкривати занадто багато інформації про свої веб-технології, імена користувачів, паролі та загальні вразливості, навіть не підозрюючи про це.

Іншими словами: Google " Hacking" - це практика використання Google для пошуку вразливих веб-додатків та серверів за допомогою власних можливостей пошукової системи Google.

Якщо ми не заблокуєте певні ресурси з нашого веб-сайту за допомогою файлу robots.txt, Google індексує всю інформацію, яка присутня на будь-якому веб-сайті. Через що через якийсь час будь-яка людина може отримати доступ до цієї інформації, якщо знає, як шукати.

Хоча ця інформація є загальнодоступною в Інтернеті, і вона надається та заохочується до використання компанією Google на законних підставах, люди з неправильними намірами можуть використовувати цю інформацію, щоб завдати шкоди нашій присутності в Інтернеті.

В Google э багато операторів вони вбудовані в мову запитів їх можна використовувати, щоб знайти список файлів, інформацію про свою конкуренцію, відстежити людей, отримати інформацію про зворотні посилання SEO, скласти списки електронної пошти, виявити веб-вразливості.

Таблиця 2.1 – Оператори Google

Назва	Приклад	Опис
cache	cache:example.com	Покаже нам кешовану версію будь-якого веб-сайту.
allintext	allintext: "text"	Пошук конкретного тексту, що міститься на будь-якій веб-сторінці
allintitle	allintitle:"title"	Відображати сторінки, які містять вказані заголовки
allinurl	allinurl: example	Використовують для отримання результатів, URL-адреса яких містить усі зазначені символи
filetype	filetype: pdf	Використовується для пошуку будь-яких розширень файлів.
intitle	Intitle: "title"	Використовується для пошуку різних ключових слів всередині заголовка
intext	Intext: "text"	Корисний для пошуку сторінок, які містять певні символи або рядки всередині їх тексту
site	site: example.com	Покаже вам повний список усіх проіндексованих URL-адрес для вказаного домену та субдомену

*	Test1 * test2	Узагальнюючий знак, який використовується для пошуку сторінок, які містять "будь-що" перед нашим словом
|	Test1 | test2	Логічний оператор
+	Test1 + test2	Використовується для об’єднання слів
–	Test1 - test2	Оператор мінус використовується, щоб уникнути показу результатів, що містять певні слова

2.2 Приклади Google Hacking

Відкрийте FTP-сервери. Google не тільки індексує сервери на основі HTTP, але також індексує відкриті FTP-сервери. За допомогою наступного оператора(intitle:"index of" inurl:ftp) ми зможемо досліджувати загальнодоступні FTP-сервери [18].

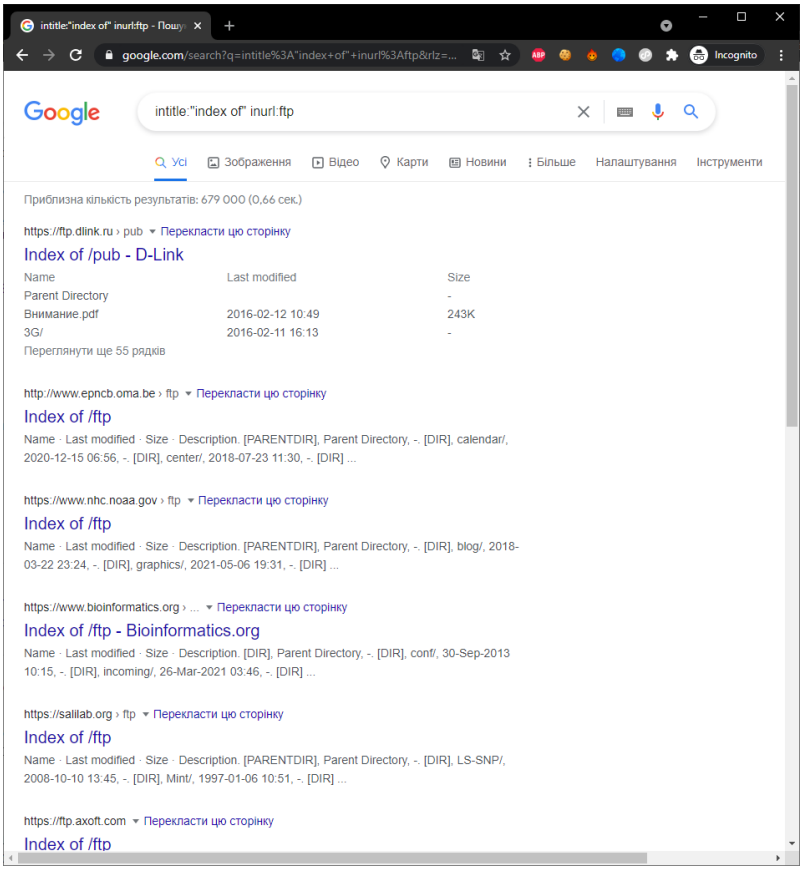

Рисунок 2.1 - Відкриті FTP-сервери

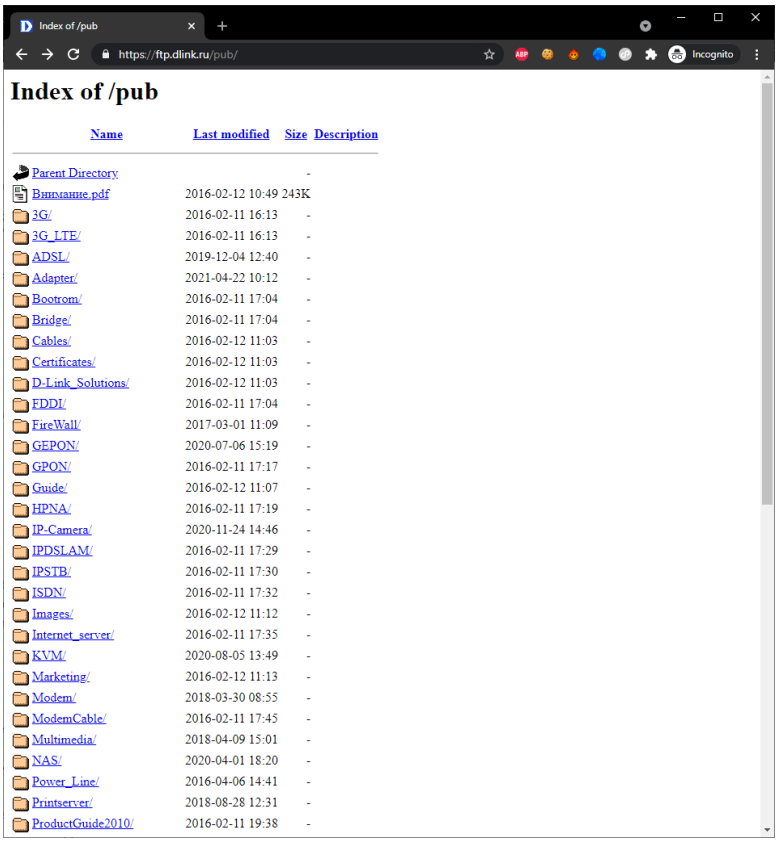

Рисунок 2.2 - Дані відкритого FTP-сервера

Списки email знайти досить просто. У наступному прикладі отримаємо файли Excel, які можуть містити адреси електронної пошти.

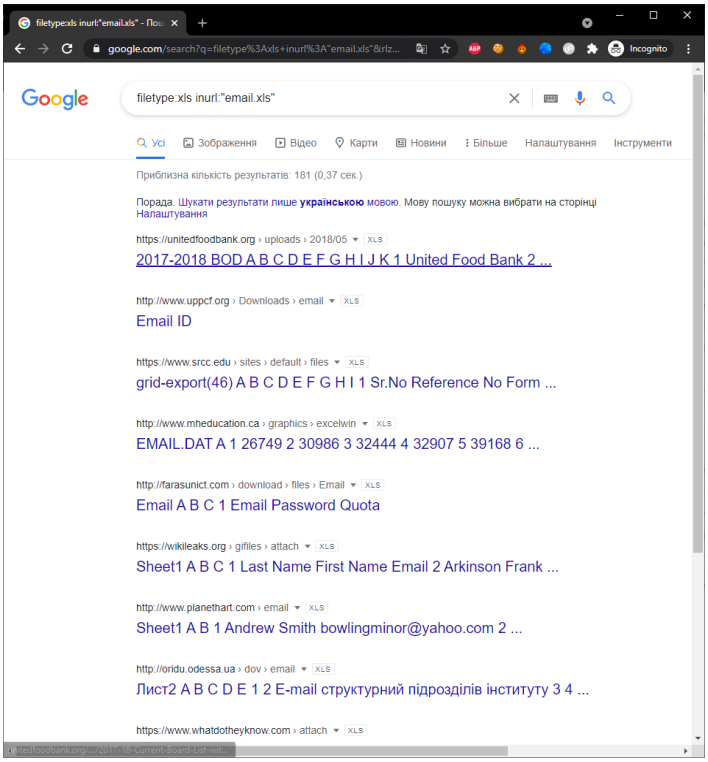

Рисунок 2.3 - Список файлів Excel з адресами електронної пошти

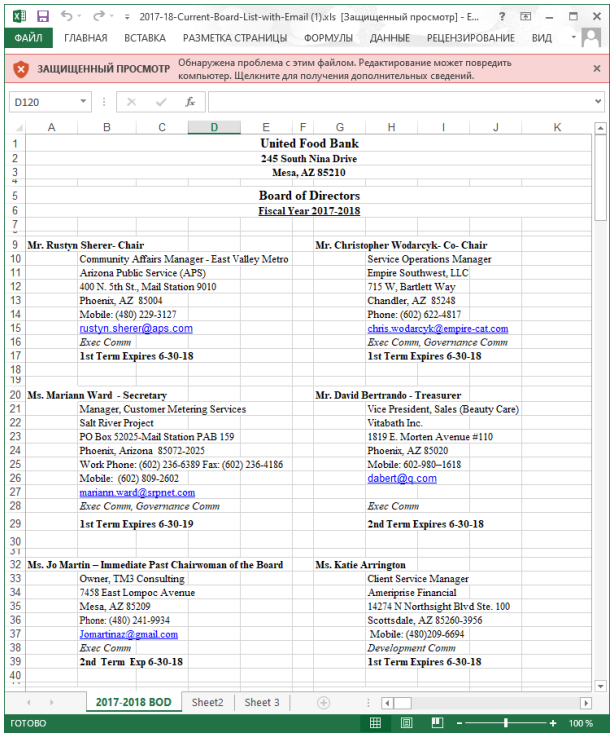

Рисунок 2.4 - Файл Excel з адресами електронної пошти

2.3 Сканери субдоменів

Запит пошукових систем. Методи Google Hacking часто використовуються для пошуку субдоменів будь-якого доменного імені. З оператора (site: example.com –www). Це може повернути повний список проіндексованих Google субдоменів. Хоча цей запит до субдомену не в реальному часі, оскільки він походить від останнього сканування GoogleBot, він часто дуже корисний для пошуку всіх субдоменів, які не захищені конфігураціями robots.txt або субдомени, що не використовують метатеги. Багато механізмів сканування субдоменів на терміналах та в Інтернеті покладаються на цей тип вбудованої мови запитів від таких пошукових систем, як Google або Bing [2].

Виконання грубої сили. Деякі інструменти виявлення використовують грубу силу та рекурсивні прийоми грубого наведення, щоб створювати списки субдоменів, більшу частину часу поєднуючи зі списками слів. Незважаючи на те, що це не найшвидший спосіб знайти субдомени, він може бути одним із найточніших. Інструменти, що використовують цей тип методів (поряд з іншими), включають Amass, Fierce та DNScan.

Передача зони DNS - це ще один спосіб повного відтворення віддаленої зони DNS. Це корисно для виявлення всіх налаштованих субдоменів на сервері DNS. Цей прийом працює лише тоді, коли зона DNS не захищена або обмежена системними адміністраторами для запитів AXFR. Незважаючи на те, що більшість DNS-серверів виправляються проти цього типу запитів DNS, варто спробувати поєднати його з дослідженням на основі грубої сили.

2.4 Sublist3r

Sublist3r - це інструмент виявлення субдоменів, який написаний на Python і призначений для перерахування субдоменів веб-сайтів із використанням даних із загальнодоступних джерел та методів грубої сили(приклад роботи на рисунку 2.1). Загальнодоступні джерела складаються з широкого кола популярних

пошукових систем, таких як Google, Yahoo, Bing, Baidu, Ask, а також Netcraft, Virustotal, ThreatCrowd, DNSdumpster та ReverseDNS для виявлення субдоменів [13].

Він широко використовується синіми та червоними командами по всьому світу для збору даних про субдомени.

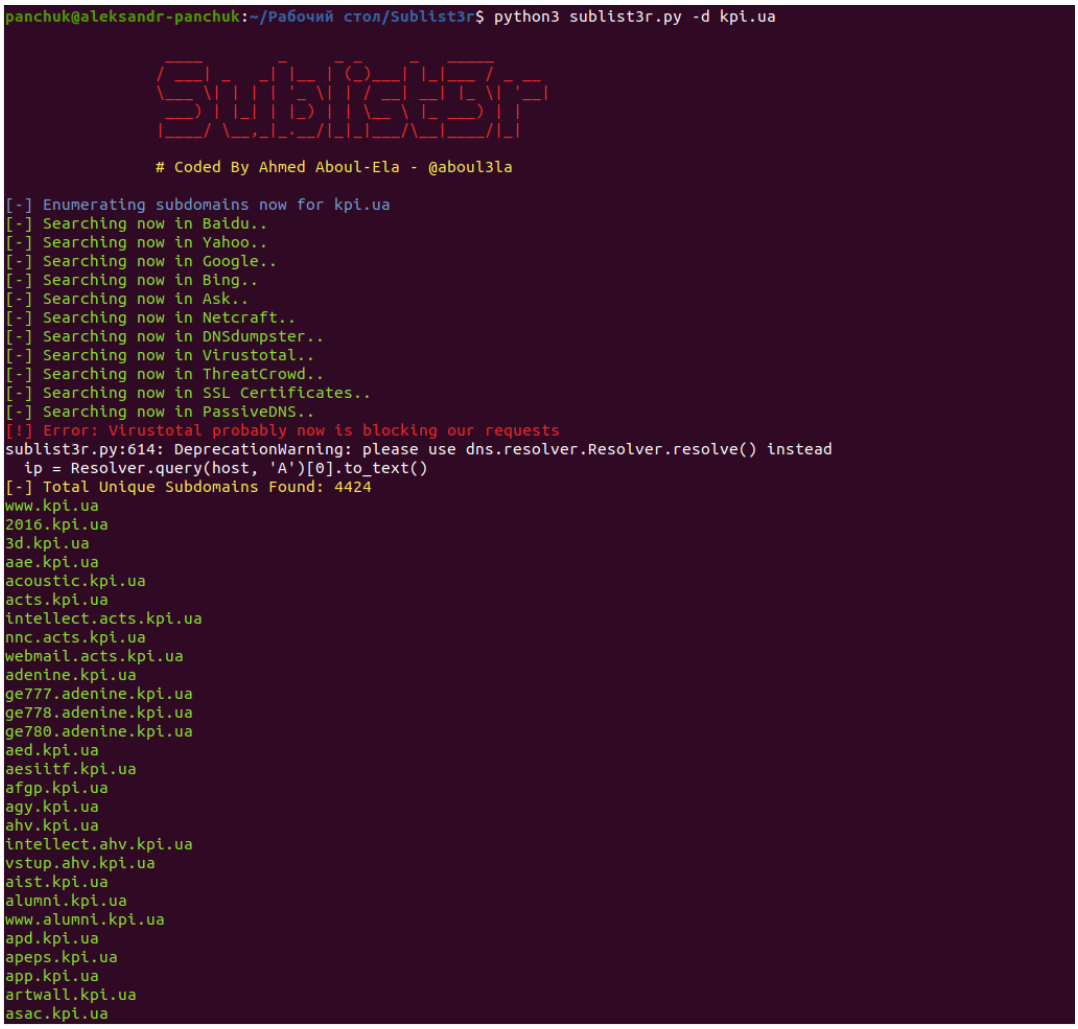

Рисунок 1.1 - Робота з Sublist3r

Програма має декілька видів ключів для більш детального пошуку субдоменів.

Таблиця 1.2 - Ключі Sublist3r

Коротка назва	Довга назва	Опис
-d	--domain	Ім'я домену для перерахування субдоменів
-b	--bruteforce	Увімкнення модуль bruteforce
-p	--ports	Сканування знайдених субдоменів для певних портів TCP
-v	--verbose	Увімкнення детального режиму та відображення результати в режимі реального часу
-t	--threads	Кількість потоків для зменшення грубої сили
-e	--engines	Список пошукових систем
-o	--output	Збереження результату у текстовий файл
-h	--help	Показ довідкових повідомлень

2.5 AMASS

Це одна з найпотужніших команд на основі терміналу для збору та накопичення великих обсягів даних субдомену.

Amass використовує різноманітні методи картографування субдоменів, включаючи злам, рекурсивну грубу силу, зворотне розгортання NDS та машинне навчання, щоб отримати повний список субдоменів. Він також включає повну інтеграцію з API SecurityTrails для швидшого пасивного розвідки субдоменів. Встановити Amass легко за допомогою попередньо скомпільованих пакетів або за допомогою оснащення Kali Linux та інших популярних дистрибутивів Linux. [22]

2.6 SubBrute

Особливість це можливість приховати походження самого сканування субдомену, використовуючи відкриті роздільники як проксі до обмежень швидкості DNS. Він також може працювати як DNS-павук, який рекурсивно сканує перелічені записи DNS, роблячи його повним набором інструментів на основі DNS-терміналів.

SubBrute підтримує фільтрацію записів DNS. Наприклад, якщо вам потрібно отримати лише записи TXT з будь-якого даного імені домену, ви можете використовувати параметр (–type: ./subbrute.py -s google.names google.com - type TXT) [23].

2.7 Knock

Knock - ще один інструмент сканування субдоменів. Він працює, виконуючи повну передачу зони DNS, і якщо це не вдається, він може запустити запит до бази даних субдомену VirusTotal. Його єдиною залежністю є пакет python-dnspython, який можна знайти у всіх основних дистрибутивах Linux. Після того, як ви це розібрали, використання цього стає справді простим завданням. Ви також можете запустити сканування за допомогою зовнішніх списків слів з параметром -w(knockpy domain.com -w wordlist.txt) [24].

Висновки до розділу 2

Google - одна з найважливіших пошукових систем у світі. Як ми всі знаємо, він має можливість індексувати все, якщо ми явно не заперечуємо це.
Ми дізналися, що Google також можна використовувати як інструмент злому, але також ми можемо використовувати його для пошуку вразливостей на власних веб-сайтах. Це може бути практичним інструментом для виконання наших обов'язків з кібербезпеки, якщо використовувати цей інструмент відповідно.

Виявили, що існує велика кількість способів отримання списку субдоменів. Для себе обрав саме Sublist3r через простоту та швидкість виконання, що нам буде потрібно далі для швидкої роботи програми.

Через велику кількість знайдених субдоменів з допомогою інструментів сканування, пошук вразливого домену вручну витрачає занадто багато часу для вирішення цієї проблеми в наступному розділі ми напишемо програму яка буде автоматизувати ці процеси для зберігання часу та зменшення шансу помилок.

3 ПРОГРАМА ТА ЕКСПЛУАТАЦІЯ

3.1 Автоматизована програма для пошуку вразливих субдоменів

Була написана програма для полегшення пошуку вразливих субдоменів яка автоматизує основні кроки на мові Node.js (код програми знаходиться в додатку).

Першим кроком було використання RegExp для перевірки вхідних даних, регулярний вираз налаштований на правильне введення назви домену, та забороняє всі інші комбінації та небезпечні команди. В нашому випадку це дуже важливо тому, що введені дані далі підставляються в термінал.

Node.js має асинхронне виконання коду, але в нашому випадку нам потрібне синхронне виконання основних кроків для коректного обміну даними які будуть передаватися далі тільки після закінчення всіх операцій в поточному кроці. Для вирішення цієї проблеми ми використовували Promise [10].

Promise являє собою обгортку для значення, невідомого на момент. Він дозволяє обробляти результати асинхронних операцій так, як якщо б вони були синхронними: замість кінцевого результату асинхронного методу повертається свого роду обіцянку отримати результат в певний момент в майбутньому. Promise має три стани:

1. Очікування (pending) початковий стан, не виконаний і не відхилений.
2. Виконано (fulfilled): операція завершена успішно.
3. Відхилено (rejected): операція завершена з помилкою.

Ми виконуємо послідовне виконання промісів які очікують дані від минулого проміса які будуть викликані з допомогою колбеку та тільки після їх отримання починають виконуватись. В залежності від результату промісу він попаде в один із методів (.then) при умові коректної роботи, або в (.catch) в випадку помилки.

Коли користувач виконує одну програму Node.js, вона працює як один процес операційної системи (ОС), який представляє екземпляр запущеної програми. В рамках цього процесу Node.js виконує програми на одному потоці. Node.js включає модуль child_process, який має функції для створення нових

процесів. Крім роботи з тривалими завданнями, цей модуль може також взаємодіяти з ОС і запускати команди оболонки. Саме цей функціонал ми використовували для створення нового потоку який буде запускати термінал та Sublist3r з допомогою Python через який ми отримаємо список субдоменів.

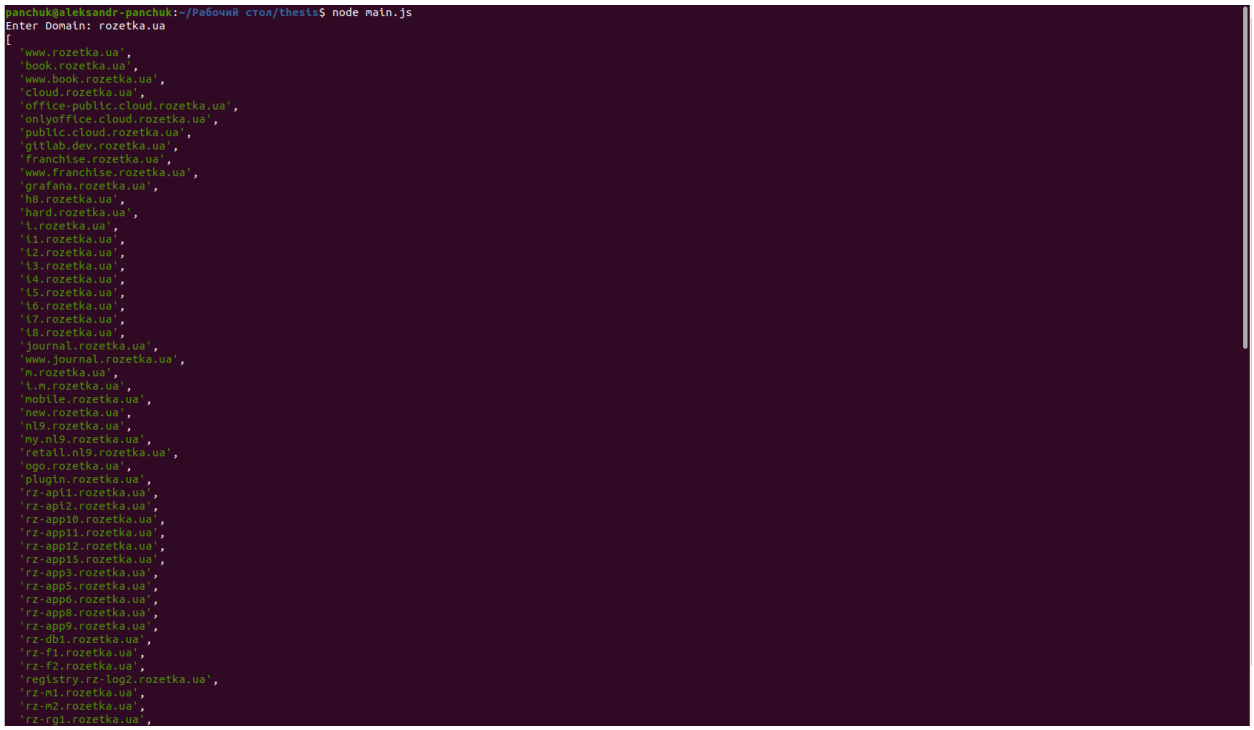

Рисунок 3.1 - Вивід всіх субдоменів в програмі

Наступним кроком буду отримання статусів по кожному субдомену. Для рішення було використано метод Promise.allSettled(), він повертає проміс, який виповнюється коли всі отримані проміси завершені незалежно від результату. Вибраний саме цей метод через те що він не зважає на результат який йому приходить в JSON, саме це нам і потрібно дня знаходження субдоменів з статусом 404. Для відправки запитів використовували axios який оснований на промісах HTTP-клієнта.

Рисунок 3.2 - Вивід статусів субдоменів в програмі

Було створено об’єкт де ключом є назва субдомену, а його значенням статус. Отримані дані потрібно відфільтрувати, нам не потрібні субдомени з статусом 200.

Рисунок 3.3 - Вивід недоступних субдоменів з програми

Отримаємо CNAME записи по цім субдоменам. Для цього знову будемо створювати новий процес для виконання команди dig.

```
mobile.rozetka.com.ua.  179     IN      CNAME   rozetka.com.ua.
rozetka.ua.             179     IN      SOA     nsa4.srv53.net. support.dnshosting.org. 1620974558 1800 600 2419200 3600
rozetka.ua.             179     IN      SOA     nsa4.srv53.net. support.dnshosting.org. 1620974558 1800 600 2419200 3600
rozetka.ua.             179     IN      SOA     nsa4.srv53.net. support.dnshosting.org. 1620974558 1800 600 2419200 3600
rozetka.ua.             179     IN      SOA     nsa4.srv53.net. support.dnshosting.org. 1620974558 1800 600 2419200 3600
rozetka.ua.             179     IN      SOA     nsa4.srv53.net. support.dnshosting.org. 1620974558 1800 600 2419200 3600
rozetka.ua.             179     IN      SOA     nsa4.srv53.net. support.dnshosting.org. 1620974558 1800 600 2419200 3600
rozetka.ua.             179     IN      SOA     nsa4.srv53.net. support.dnshosting.org. 1620974558 1800 600 2419200 3600
rozetka.ua.             179     IN      SOA     nsa4.srv53.net. support.dnshosting.org. 1620974558 1800 600 2419200 3600
rozetka.ua.             179     IN      SOA     nsa4.srv53.net. support.dnshosting.org. 1620974558 1800 600 2419200 3600
h8.rozetka.com.ua.      179     IN      CNAME   rozetka.com.ua.
rozetka.ua.             179     IN      SOA     nsa4.srv53.net. support.dnshosting.org. 1620974558 1800 600 2419200 3600
nl9.rozetka.com.ua.     179     IN      CNAME   rozetka.com.ua.
ogo.rozetka.com.ua.     179     IN      CNAME   rozetka.com.ua.
rz-s1.rozetka.com.ua.   179     IN      CNAME   rozetka.com.ua.
```

Рисунок 3.4 - Вивід кінцевого результату програми

3.2 Експлуатація з допомогою GitHub

При відомих CNAME субдомена ми можемо приступати до наступного кроку, а саме перехвату контролю.

Розглянемо випадок що CNAME буде належить GitHub, він би мав вигляд *.github.io. Після цього спробуємо заявити права на цей субдомен, вказавши нашу сторінку GitHub на субдомен. Увійдемо в GitHub і створимо дерикторію.

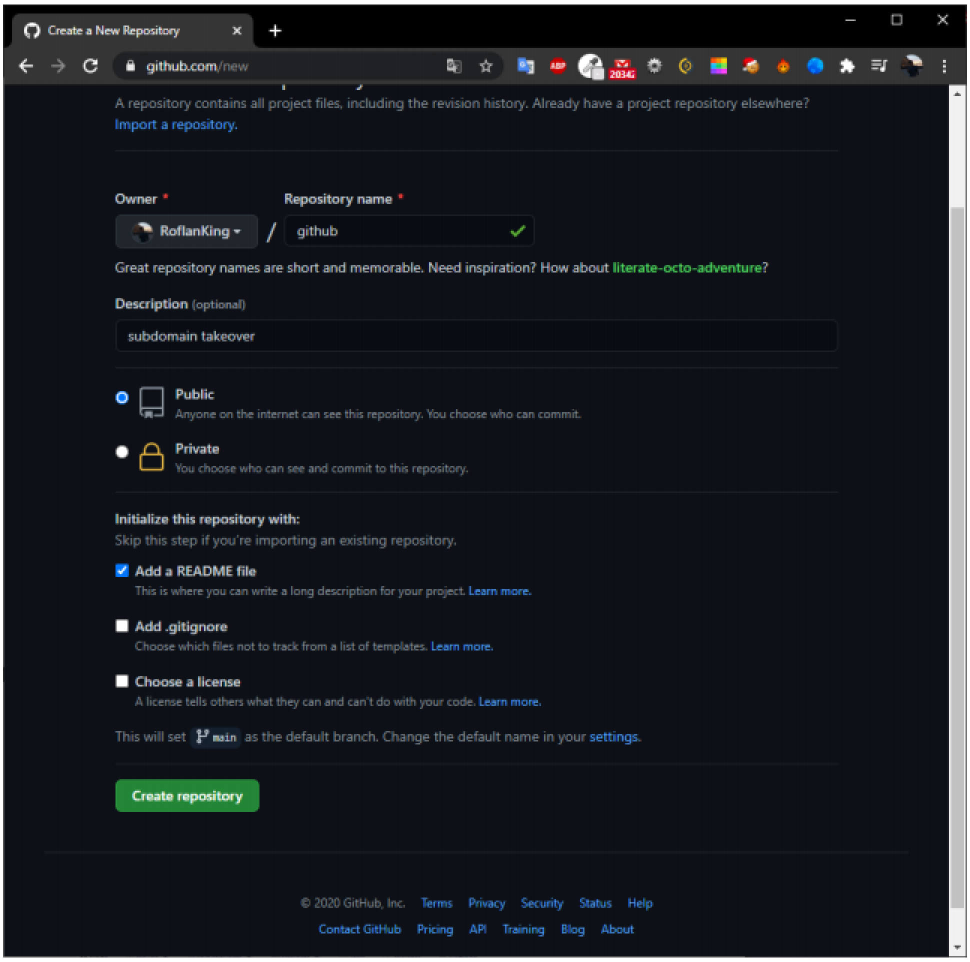

Рисунок 3.5 - Створення дерикторії GitHub

Далі переходимо в налаштування.

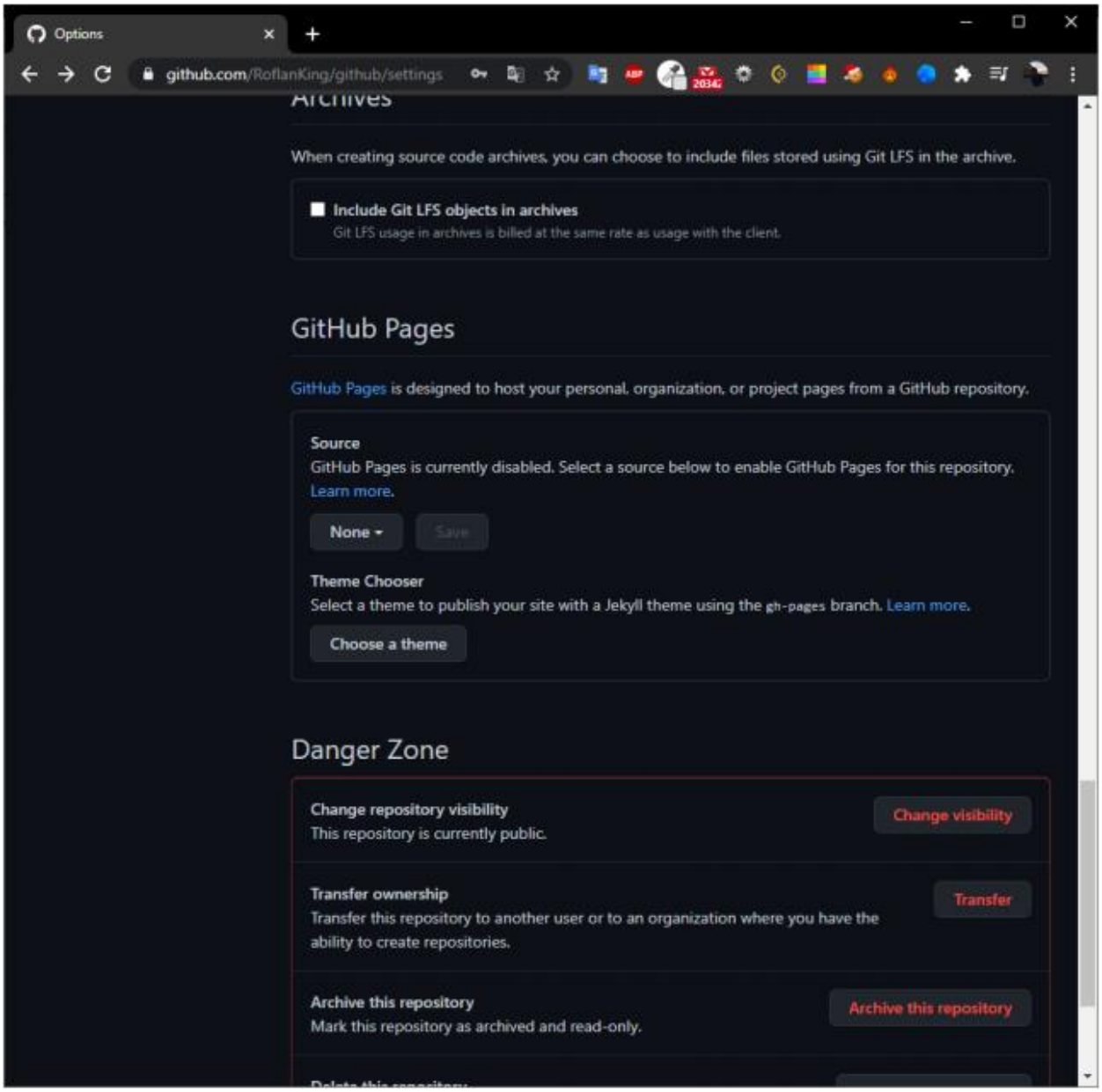

Рисунок 3.6 - Налаштування дерикторії GitHub

Нам потрібно змінити Source на головну гілку. Тепер можемо вказати субдомен над яким ми хочемо отримати контроль в нашому випадку для прикладу вкажемо 07.reddit.com.

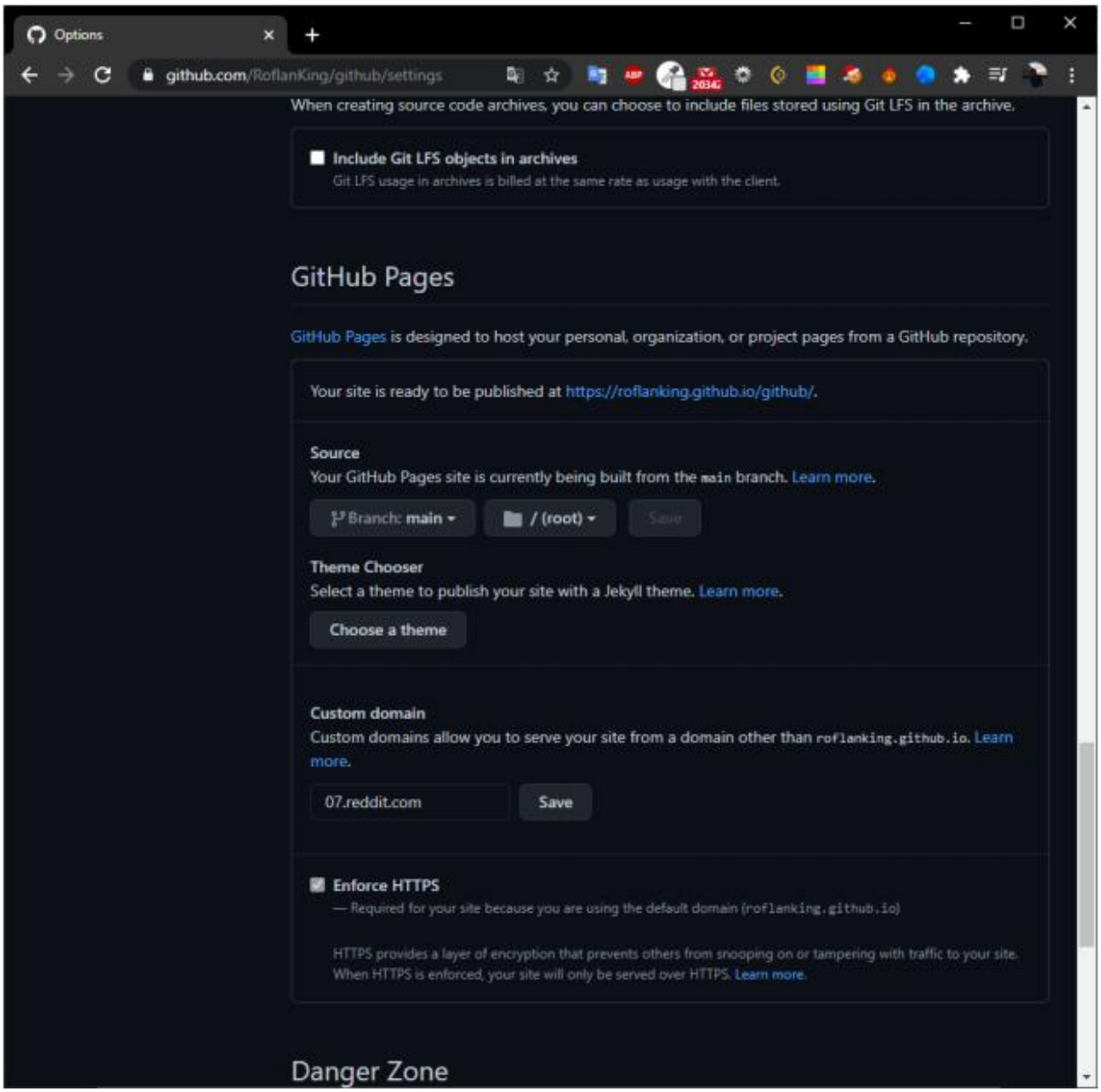

Рисунок 3.7 - Зміна гілки GitHub

Після цього якщо ми перейдемо на посилання 07.reddit.com потрапимо на наш субдомен який знаходиться тепер під нашим контролем. В даному випадком ми б побачили напис GitHub оскільки вона знаходитися в нашому файлі README.md. Тепер ми можемо створювати файли які хочемо та всі вони будуть виконуватися при переході на посилання.

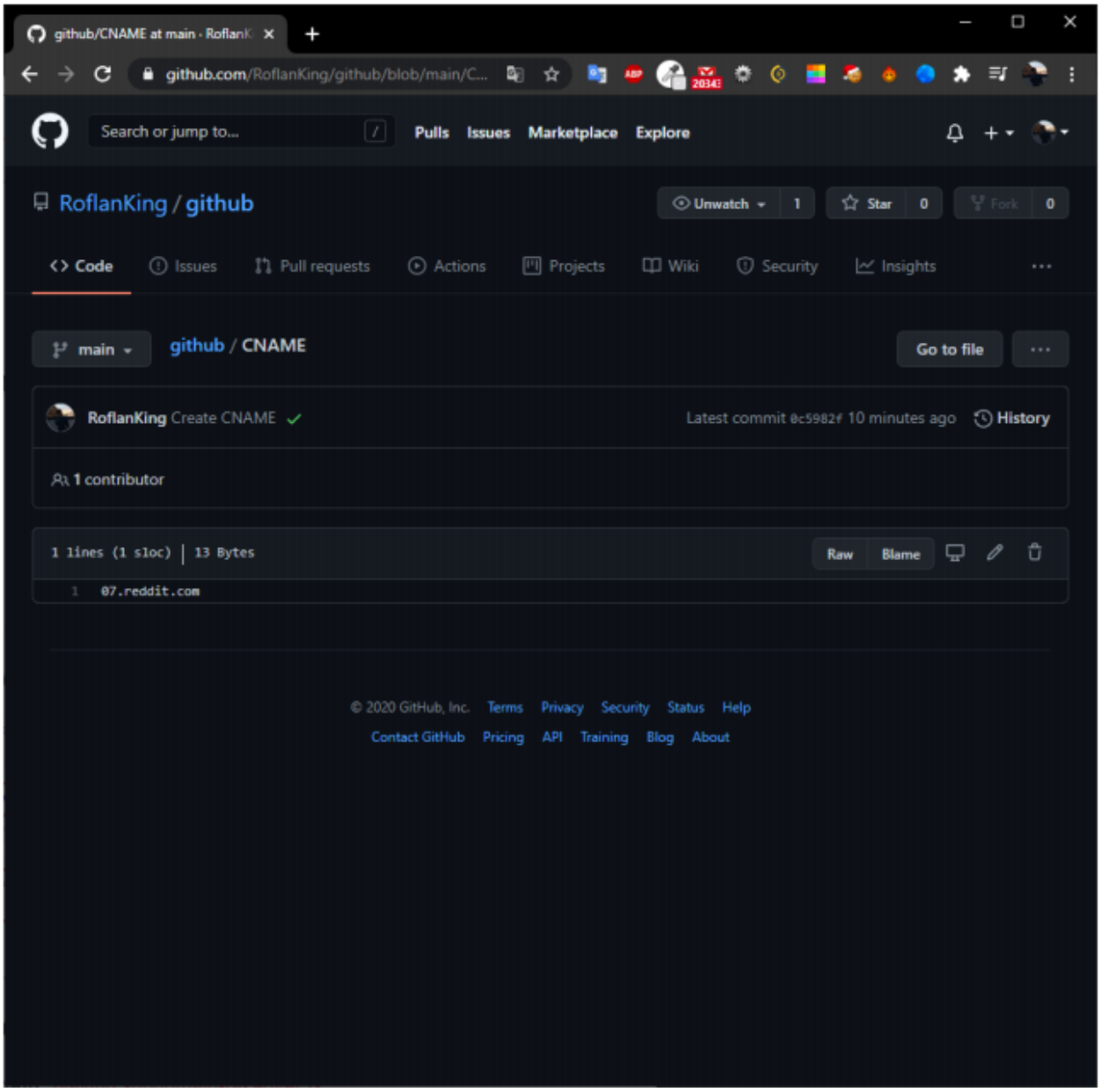

Рисунок 3.8 - Зміст дерикторії GitHub

Як ми бачимо створився файл CNAME в якому зазначена адреса нашої цілі. Але існує ще один спосіб просто додати файл CNAME в нашій дерикторії та вказати в ньому CNAME який ми отримали за допомогою команди DIG.

3.3 Фішінг

Ще одним очевидним способом використання субдомену, вразливого до поглинання, є фішинг. Більшість фішингових сайтів, що видаються за законні

(наприклад, payments.example.org), як правило, не переживають пильної перевірки адресного рядка браузера, оскільки фішери, як правило, не мають іншого вибору, крім як розмістити їх у схожому на домен домені (наприклад, payments.exarnple.com). Насправді поглинання субдомену - ідеальне місце для проведення фішинг-кампанії [1].

Завдяки захопленню субдомену фішери можуть легко використовувати репутацію законного доменного імені, щоб заманити на нього нічого не підозрюючих користувачів та отримати від них конфіденційну інформацію (наприклад, дані облікового запису, особисту інформацію та платіжні реквізити). Навіть технічно підковані відвідувачі, ймовірно, впадають у хитрощі.

У деяких випадках зловмисники можуть навіть отримати та встановити дійсний сертифікат TLS для вразливого субдомену для обслуговування їх фішингового сайту через HTTPS і, як це не сумно, багато людей, все ще розглядають використання HTTPS як надійний показник надійності сервера на іншому кінці з'єднання.

3.4 Крадіжка cookie файлів

Загальний фішинг, як правило, вимагає певної довірливості та взаємодії користувача (окрім переходу до шкідливого сайту) від жертви. Однак поглинання субдомену є більш потужним, оскільки воно може дозволити зловмисникові викрасти чутливі файли cookie просто за допомогою жертви, яка відвідує сайт зловмисника [1].

Файли cookie мають атрибут Domain, який визначає, куди браузери можуть надсилати їх у HTTP-запитах. Наприклад, якщо файл cookie встановлено з example.org із зазначеним іменем хосту як значенням його атрибута Domain, браузери прикріплять його до всіх запитів HTTP, надісланих до example.org або будь-якого його субдомену. Деякі веб-сайти, ставлячи під сумнів підхід до впровадження автентифікації єдиного входу (SSO) для всіх своїх субдоменів,

встановлюють атрибут Domain своїх файлів cookie, що ідентифікує сеанс, для свого кореневого домену (example.org).

Зловмисник, який взяв контроль над субдоменом і розміщує там власний зловмисний сервер, який налаштований для реєстрації всіх запитів HTTP. Переманивши автентифікованих користувачів на скомпрометований субдомен, зловмисник зможе збирати файли cookie з корінним доменом, просто перевіряючи журнали сервера.

Якщо файли cookie, що ідентифікують сеанси, зловмисник зможе викрасти сеанси користувачів і, можливо, навіть захопити облікові записи користувачів.

Атрибути файлів cookie безпеки не можуть захистити користувачів, які випадково відвідують порушений субдомен. Зокрема, атрибут cookie HttpOnly тут не має значення, оскільки атака не включає JavaScript на стороні клієнта припускаючи, що субдомен має дійсний сертифікат TLS, атрибут захищеного файлу cookie не може допомогти, оскільки браузери надсилатимуть файли cookie до порушеного субдомену, навіть через HTTPS.

3.5 Міжсайтовий підробка запиту

Підробка міжсайтових запитів (CSRF) - це атака, яка змушує кінцевого користувача виконувати небажані дії над веб-програмою, в якій він в даний час автентифікований .

Візьмемо для прикладу соціальну мережу. Зловмисник CSRF може виконувати делікатні дії, такі як лайкнути запис іншого користувача, оновити електронну адресу жертви, або зміна зображення профілю жертви, все від імені жертви, але всупереч його або її волі.

Атрибут cookie SameSite був введений для захисту користувачів від таких міжсайтових атак, але небагато організацій цим користуються в повній мірі. Таким чином, CSRF залишається ризиком скрізь, де використовується аутентифікація на основі файлів cookie.

Одним із можливих пом’якшувальних заходів щодо CSRF є перевірка джерела запитів про зміну держави щодо списку надійних джерел. Заголовок запиту Origin, коли він присутній, надає цю інформацію надійно, оскільки програмне змінення цього заголовка просто не дозволяється сучасними браузерами. Тому багато сайтів покладаються на перевірку заголовка джерела, щоб вирішити, відхиляти чи приймати запит на зміну стану між сайтами.

Крім того, підмножина цих веб-сайтів вирішує прийняти всі запити, що походять від їх субдоменів. Наприклад, коли api.example.com отримує запит, призначений для оновлення електронної адреси користувача, він може відхилити запит, якщо останній походить звідкись, наприклад evil.org, але прийняти запит, якщо останній походить з example.org або будь-якого іншого його субдомен.

Якщо ви захищаєте своїх користувачів від CSRF на основі перевірки заголовка джерела, але довіряєте всім субдоменам example.com, одного субдомену example.com, вразливого до поглинання, достатньо, щоб перемогти цей механізм захисту.

3.6 Зловживання довіри до CORS

Шкідливі запити на зміну стану - це не єдиний тип міжсайтових атак, яких слід бути обережними, оскільки зловмисники можуть також використовувати міжсайтові запити для проникнення конфіденційних даних. Політика того самого походження (SOP), яка, можливо, є основою безпеки браузера, зазвичай запобігає подібним атакам.

Однак спільне використання ресурсів (CORS) було запроваджено в HTML5, щоб дозволити розробникам вибірково скасувати деякі обмеження, які застосовуються SOP. Наприклад, щоб сигналізувати браузеру, що іншому джерелу дозволено читати відповіді на аутентифіковані запити, api.example.com відповість такими заголовками:

Access-Control-Allow-Origin: https://google.com

Access-Control-Allow-Credentials: true

На жаль, розробники CORS занадто часто сприймають як неприємність, які в результаті неправильно налаштовують його, не усвідомлюючи повністю наслідків. У цій конкретній ситуації, якби api.example.org було налаштовано на обробку будь-якого субдомену example.com як надійного джерела для цілей CORS, одного поглинання субдомену було б достатньо, щоб зловмисник читав та виводив вміст відповідей HTTP надходить з api.example.com.

3.7 Способи захисту

Достатньо 5 пунктів для запобігання виконанню загрози:

- Перегляньте свої записи DNS і видаліть усі активні, але вже не використовуються записи, особливо ті, що вказують на зовнішні служби. Обов’язково видаліть застарілий запис CNAME у файлі зони DNS.
- Переконайтеся, що ваші зовнішні служби налаштовані на прослуховування вашого DNS-символу.
- Додайте "Видалення входу DNS" у свій контрольний список.
- Створюючи новий ресурс, зробіть створення запису DNS останнім кроком у процесі, щоб уникнути його вказівки на неіснуючий домен.
- Постійно відстежуйте свої записи DNS і переконайтеся, що немає звисаючих записів DNS

Висновки до розділу 3

У результаті вивченої інформації в попередніх розділах була написана програма для автоматизованого пошуку вразливих субдоменів. Дану програму можна використовувати як для аналізу свого домену для слідкуванням за станом субдоменів так і для атаки.

Було продемонстровано основний метод перехоплення контролю за допомогою GitHub при умові що даний субдомен знаходиться на ньому.

Показано основі вразливості які можна реалізувати після перехоплення субдомена, що підвищує його критичність.

ВИСНОВКИ

У кваліфікаційній роботі було досліджено спосіб перехвату контролю над субдоменами цільвого домена. Була показана критичність цієї вразливості та її наслідки, а сама реалізації інших вразливостей за допомогою Subdomain Takeover.

Результатом роботи є написана програма мета якої автоматизувати процеси пошуку вразливих доменів для швидкого отримання станів наших субдоменів, дану програму можуть використовувати як і компанії з великою розгалуженістю та і для атаки.

В результаті виконання коду, було проведено основні кроки пошуку на мові Node.js :

1. Пошук всіх субдоменів за допомогою Sublis3r.
2. Відправка запитів на кожен субдомен та отримання коду відповіді.
3. Фільтрації вихідних данних.
4. Перевірка CNAME записів через Dig.
5. Вивід кінцевого результату.

Отримані дані можна використовувати для аналізу стану субдоменів та знаходження слабких місць які потрібно виправити.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Cretel J. Subdomain Takeover: Ignore This Vulnerability at Your Peril [Електронний ресурс] / Julien Cretel // Honeybadger. – 2021. – Режим доступу до ресурсу: https://www.honeybadger.io/blog/subdomain-takeover/

2. Borges E. Top 7 Subdomain Scanner Tools: Find Subdomains in Seconds [Електронний ресурс] / Esteban Borges // Security Trails. – 2019. – Режим доступу до ресурсу: https://securitytrails.com/blog/subdomain-scanner-find-subdomains

3. Borges E. The Most Popular Types of DNS Attacks [Електронний ресурс] / Esteban Borges // Security Trails. – 2018. – Режим доступу до ресурсу: https://securitytrails.com/blog/most-popular-types-dns-attacks

4. Borges E. Domain Tools: top DNS, IP and Domain utilities to investigate any website [Електронний ресурс] / Esteban Borges // Security Trails. – 2019. – Режим доступу до ресурсу: https://securitytrails.com/blog/domain-tools

5. Dunn J. Researcher finds 670 Microsoft subdomains vulnerable to takeover [Електронний ресурс] / John E Dunn // Sophos. – 2020. – Режим доступу до ресурсу: https://nakedsecurity.sophos.com/2020/03/06/researcher-finds-670-microsoft-subdomains-vulnerable-to-takeover/

6. 670+ Subdomains of Microsoft are Vulnerable to Takeover [Електронний ресурс] // Vulnerability. – 2020. – Режим доступу до ресурсу: https://vullnerability.com/blog/microsoft-subdomain-account-takeover

7. Overflow E. A GUIDE TO SUBDOMAIN TAKEOVERS [Електронний ресурс] / E. Overflow // HackerOne. – 2018. – Режим доступу до ресурсу: https://www.hackerone.com/blog/Guide-Subdomain-Takeovers

8. Subdomain takeovers [Електронний ресурс] // MDN Web Docs. – 2020. – Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Web/Security/Subdomain_takeovers

9. Subdomain Takeover, или как найти заброшенные поддомены [Електронний ресурс] // Хабр. – 2018. – Режим доступу до ресурсу: https://habr.com/ru/sandbox/124107/

10. Метод Promise.allSettled() [Електронний ресурс] // MDN Web Docs. – 2021. – Режим доступу до ресурсу: https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/Promise/allSettled

11. Overflow E. Can I Take Over XYZ? [Електронний ресурс] / E. Overflow // GitHub. – 2021. – Режим доступу до ресурсу: https://github.com/EdOverflow/can-i-take-over-xyz

12. Subdomain takeover — Chapter one: Methodology [Електронний ресурс] // Security-aholic. – 2019. – Режим доступу до ресурсу: https://secaholic.com/subdomain-takeover-chapter-one-methodology-dd6d86b0941b

13. Sublist3r [Електронний ресурс] // GitHub. – 2020. – Режим доступу до ресурсу: https://github.com/aboul3la/Sublist3r

14. Hudak P. Subdomain Takeover: Going beyond CNAME [Електронний ресурс] / Patrik Hudak // Patrik Hudak. – 2020. – Режим доступу до ресурсу: https://0xpatrik.com/subdomain-takeover-ns/

15. Hostile Subdomain Takeover using Heroku/Github/Desk + more [Електронний ресурс] // detectify. – 2014. – Режим доступу до ресурсу: https://labs.detectify.com/2014/10/21/hostile-subdomain-takeover-using-herokugithubdesk-more/

16. Cretel J. Subdomain Takeover: Ignore This Vulnerability at Your Peril [Електронний ресурс] / Julien Cretel // Honeybadger. – 2021. – Режим доступу до ресурсу: https://www.honeybadger.io/blog/subdomain-takeover/

17. Jelen S. 5 Subdomain Takeover #ProTips [Електронний ресурс] / Sara Jelen // Security Trails. – 2019. – Режим доступу до ресурсу: https://securitytrails.com/blog/subdomain-takeover-tips

18. Google Dorking или используем Гугл на максимум [Електронний ресурс] // Хабр. – 2020. – Режим доступу до ресурсу: https://habr.com/ru/company/postuf/blog/510766/

19. Google Hacking: What is a Google Hack? [Електронний ресурс] // Acunetix. – 2020. – Режим доступу до ресурсу:

https://www.acunetix.com/websitesecurity/google-hacking/

20. Dig Command in Linux (DNS Lookup) [Електронний ресурс] // Linuxize. – 2020. – Режим доступу до ресурсу: https://linuxize.com/post/how-to-use-dig-command-to-query-dns-in-linux/

21. What is: DNS [Електронний ресурс] // wpbeginner. – 2020. – Режим доступу до ресурсу: https://www.wpbeginner.com/glossary/dns/

22. OWASP/Amass: In-depth Attack Surface Mapping and Asset Discovery [Електронний ресурс] // GitHub. – 2021. – Режим доступу до ресурсу: https://github.com/OWASP/Amass

23. A DNS meta-query spider that enumerates DNS records, and subdomains [Електронний ресурс] // GitHub. – 2017. – Режим доступу до ресурсу: https://github.com/TheRook/subbrute

24. Knock Subdomain Scan [Електронний ресурс] // GitHub. – 2021. – Режим доступу до ресурсу: https://github.com/guelfoweb/knock

25. Stephens L. How To Setup an Automated Sub-domain Takeover Scanner for All Bug Bounty Programs in 5 Minutes [Електронний ресурс] / Luke Stephens // Medium. – 2018. – Режим доступу до ресурсу: https://hakluke.medium.com/how-to-setup-an-automated-sub-domain-takeover-scanner-for-all-bug-bounty-programs-in-5-minutes-3562eb621db3

26. Adelantado D. Subdomain Takeover in Azure: making a PoC [Електронний ресурс] / Diego Bernal Adelantado // GoDiego. – 2020. – Режим доступу до ресурсу: https://godiego.tech/posts/STO/

27. Daityari S. What Are Subdomains? Definition and Examples [Електронний ресурс] / Shaumik Daityari // Themeisle. – 2020. – Режим доступу до ресурсу: https://themeisle.com/blog/what-are-subdomains/

28. Wood K. Subdomain vs Domain [Електронний ресурс] / Kevin Wood // HostGator. – 2019. – Режим доступу до ресурсу: https://www.hostgator.com/blog/subdomain-vs-domain/

29. Crabb K. What is a Subdomain? [Електронний ресурс] / Kristin Crabb // Domain.com Blog. – 2019. – Режим доступу до ресурсу:

https://www.domain.com/blog/2019/01/15/subdomain/

30. Яворски П. Основы веб-хакинга [Текст] / П. Яворски, 2016 – С.110-117.

ДОДАТОК А

main.js

```
const readline = require("readline")
const { exec } = require("child_process")
const axios = require("axios")
const rl = readline.createInterface({
    input: process.stdin,
    output: process.stdout,
})
const reDomain = /^[a-zA-Z0-9][a-zA-Z0-9-]{1,61}[a-zA-Z0-9]\.[a-zA-Z]{2,}$/
const domainStatus = new Object()

rl.question("Enter Domain: ", (answer) => {
    const domain = answer
    if (reDomain.test(domain)) {
        new Promise((resolve, reject) => {
            exec(`python3 ./Sublist3r/sublist3r.py -d ${domain}`, (error, stdout, stderr) => {
                if (error) {
                    console.log(`error: ${error.message}`)
                    return
                }
                const resultSublist3r = stdout.split('\n\u001b[92m')
                const listData = []
                for(let i = 12; i <= resultSublist3r.length -1; i++){
                    listData.push(resultSublist3r[i])
                    }
                const domainArray = []
                const domainList = []
                for(let i = 0; i <= listData.length-1; i++){
```

```
            domainArray[i] = listData[i].split('\u001b[0m')
            domainList.push(domainArray[i][0])
          }
          resolve(domainList)
        })
      })
      .then(function(domainList){
        console.log(domainList)
        const requests = domainList.map(domainName => axios.get(`https://${domainName}`))
        return Promise.allSettled(requests)
        .then(responses =>{
          for(let response of responses ){
            if(response.status === 'rejected' && response.reason.response != undefined){
              domainStatus[response.reason.response.request.socket._host] = {status: response.reason.response.status}
            }
            else if(response.status === 'fulfilled'){
              domainStatus[response.value.request.host] = {status: response.value.status}
            }
          }
          return domainStatus
        })
      })
      .then(function(domainStatus){
        console.log(domainStatus)
        const domainListError = []
        for(let key in domainStatus){
```

```
            if(domainStatus[key].status != 200){
                domainListError.push(key)
            }
        }
        for(let i = 0; i <= domainListError.length-1; i++){
            exec(`dig @8.8.8.8 ${domainListError[i]} CNAME`, (error, stdout, stderr) =>
{
                if (error) {
                    console.log(`error: ${error.message}`)
                    return
                }
                const digArray = stdout.split('\n')
                const digCNAME = digArray[14]
                console.log(digCNAME)
            })
        }
        console.log(domainListError)
    })
  }
  else {
    console.log(`Domain incorrect: ${domain}`)
  }
rl.close()
})
```