Sri Lanka Institute of Information Technology

Bug Bounty - Report 09

PII Disclosure

Student Name – Wanasinghe N.K

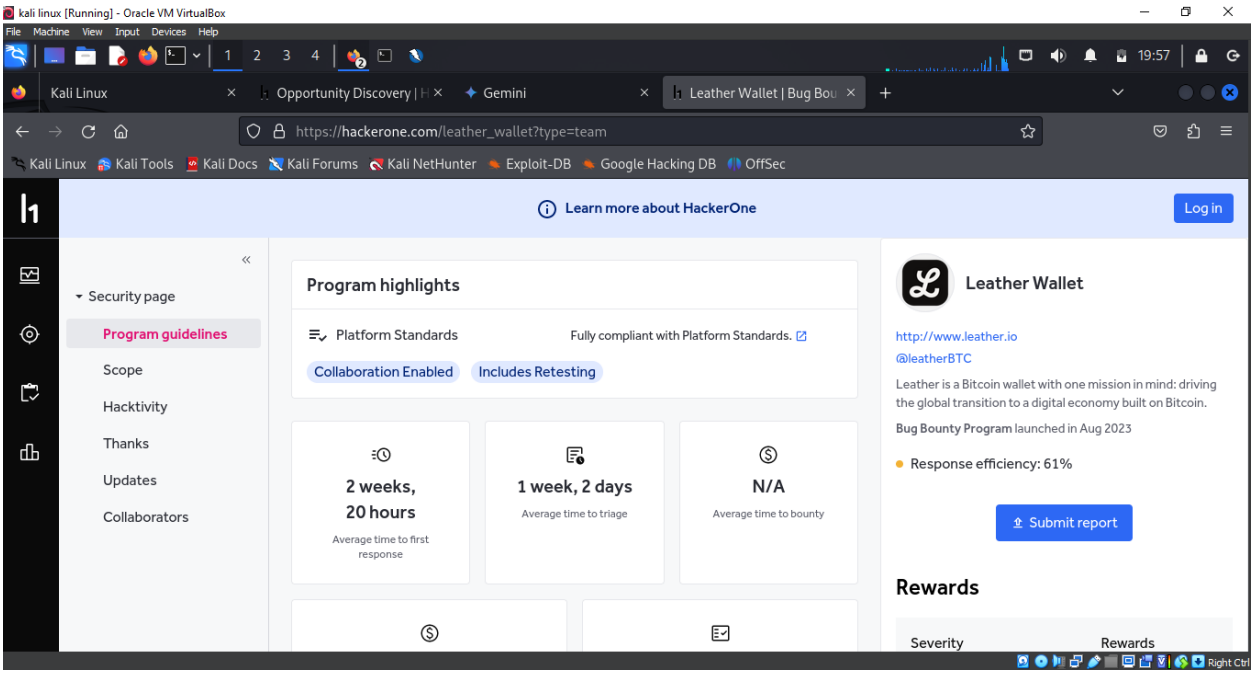
Student ID – IT23221000

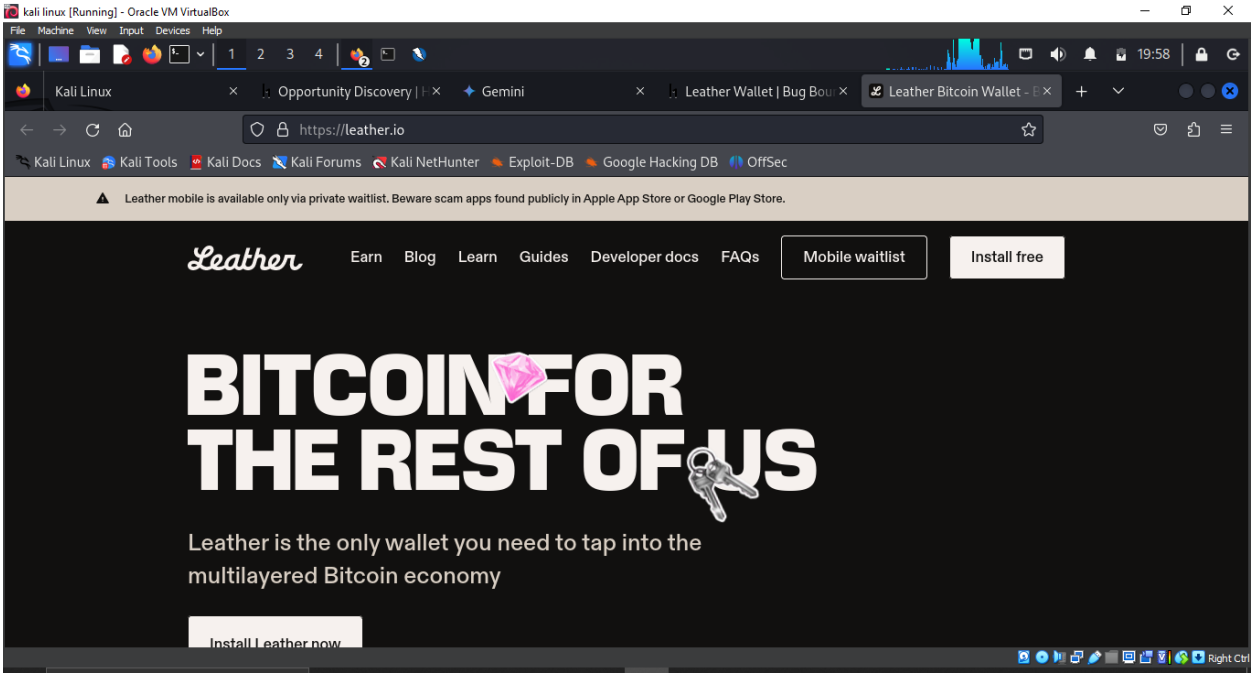
IE2062 - Web Security

B.Sc. (Hons) in information Technology Specializing in Cyber Security

Report 09 – leather.io (Hackerone)

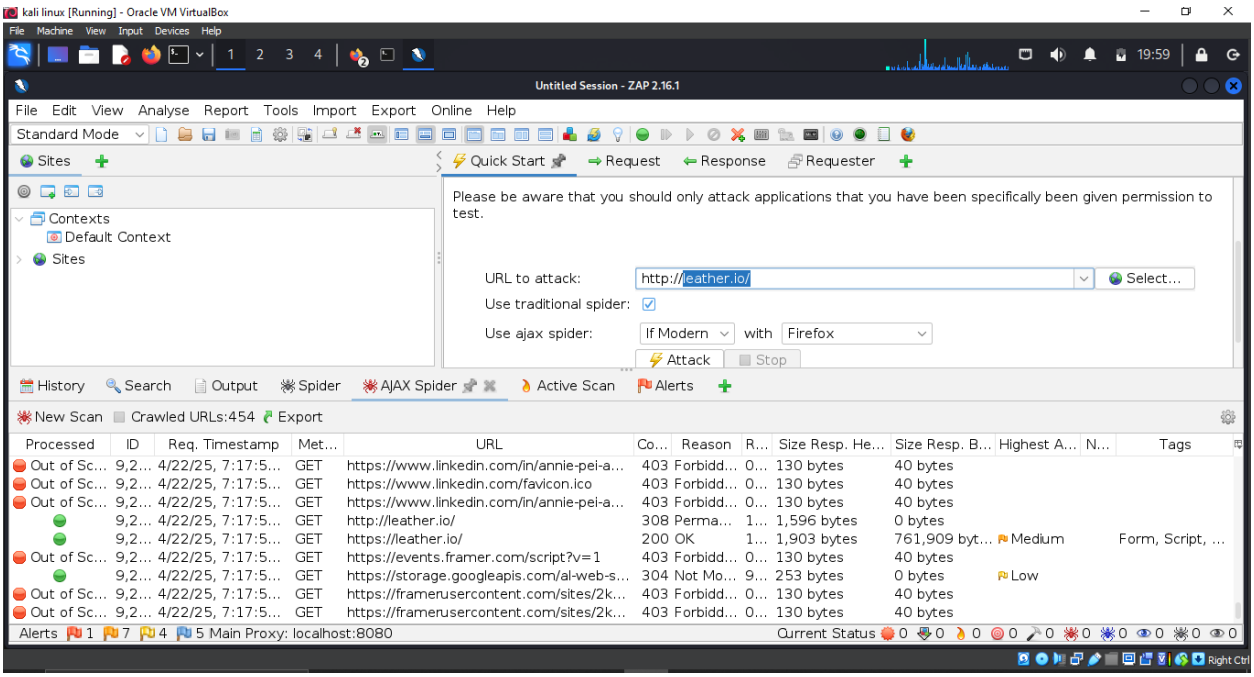
Main domain – http://www.leather.io





I used OWASP ZAP tool to scan the website

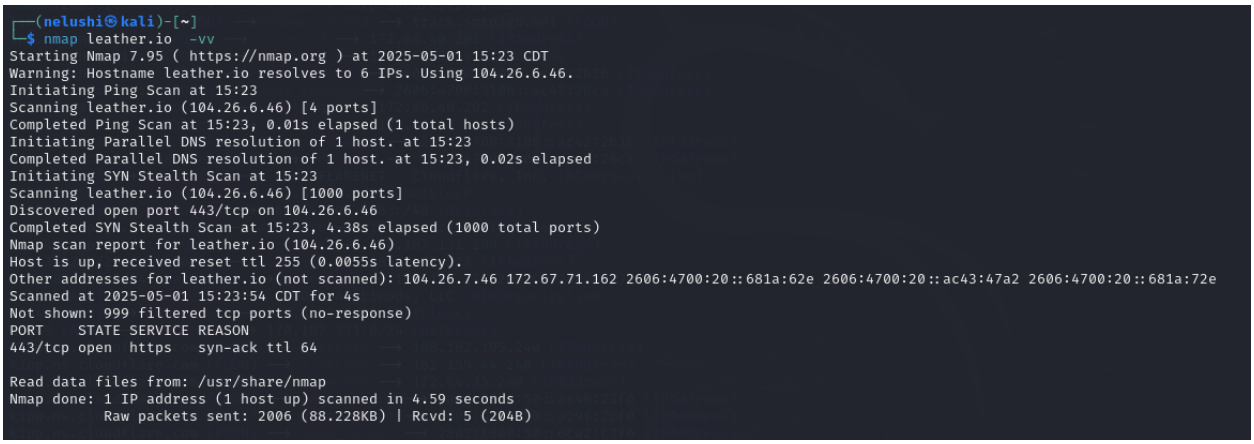
All the in scope and out of scope domains were found through running a AJAX spider attack



I found all the subdomains of the leather.io website using the **sublist3r** tool.

Nmap – Network scanning and enumeration

I found all the open ports and detected the running services on the target server using Nmap.

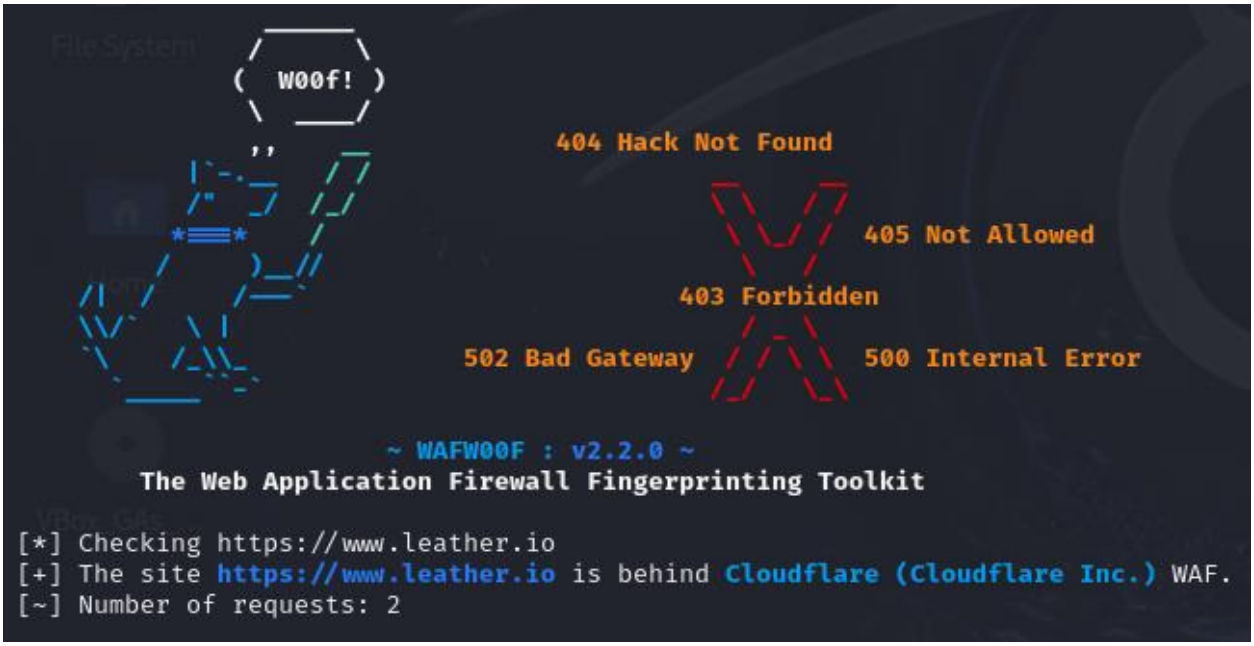


Amass – Subdomain and DNS mapping

I found all the subdomains related to the target domain using Amass.

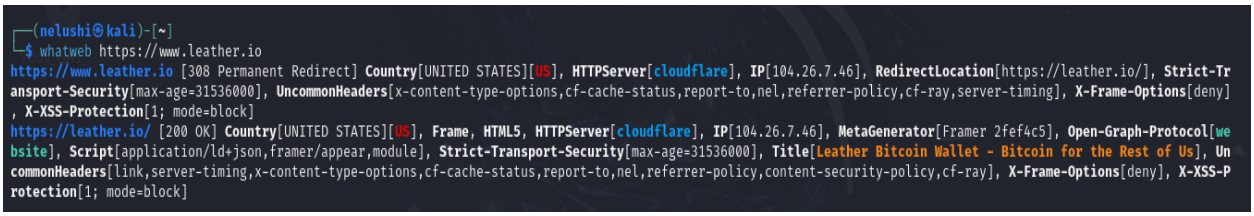
Wafw00f – Firewall Detection

Command used – wafw00f https://www.leather.io



Whatweb – to identify the technologies used by the site.

Commans used – whatweb https://www.leather.io



Vulnerability 01

Domain

https://leather.io/guides/migrating-ordinals-sparrow-leather

Vulnerability title

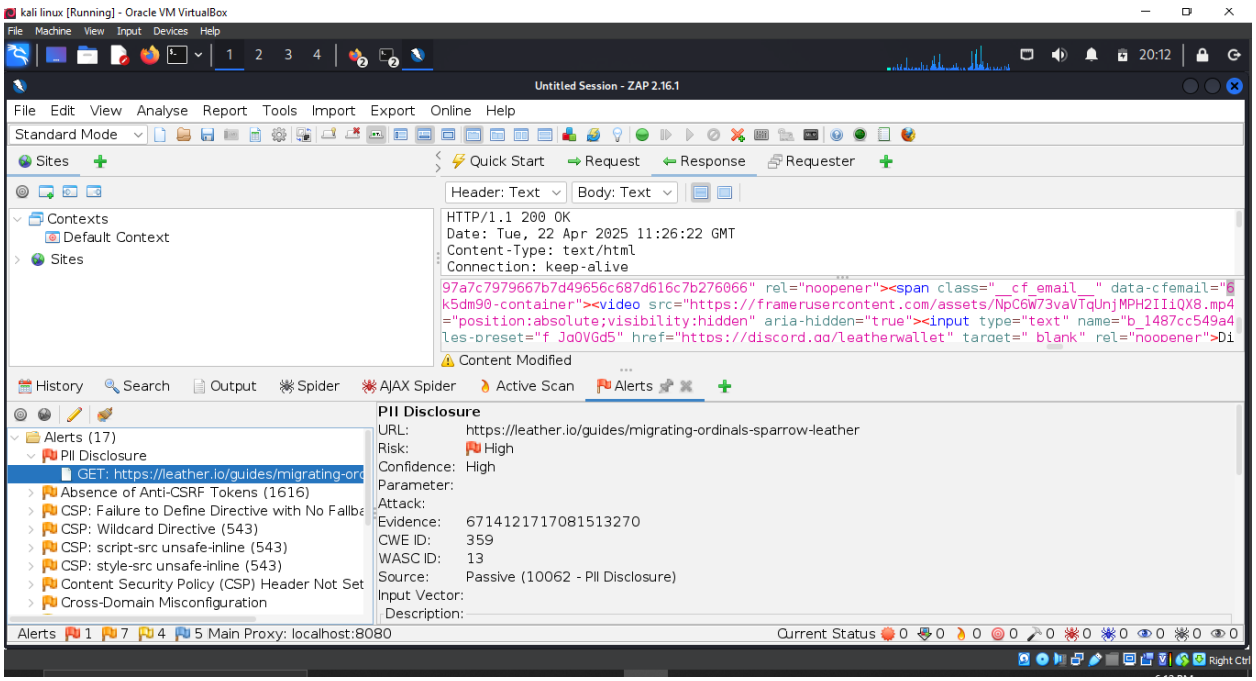
Personal Identifiable Information Disclosure

Vulnerability description

Vulnerability: PII Disclosure

PII is disclosed when the application behaves insecurely. PII is any data that can be used to identify a person in a unique way, such as names, email addresses, phone numbers, physical addresses, session IDs, or authenticating tokens.

An example of this kind of issue is when an application's implementation of access control, validation, or filtering is absent and still, it has the possibility of giving access to the user data. Alongside such a scenario, there is also an option of logging insecurely, having a poor API design, or being careless about client-side storage.



Examples of PII Disclosure

1. **Sensitive data in API responses**


```
{
  "user_id": 102,
  "name": "Jane Doe",
  "email": "jane.doe@example.com",
  "phone": "+11234567890"
}
```


2. **Session identifiers or tokens exposed in HTTP responses**


```
{
  "sessid": "ieuuq4bieu89mbdinvn4sbh655",
  "key": 10939043903816504
}
```


3. **User information exposed in URL parameter**

https://example.com/profile?email=john.doe@example.com

4. **PII stored insecurely in browser storage**

```
localStorage.setItem("user_phone", "+19876543210");
```

5. **Debug messages or error logs revealing user data**

Error: User login failed for user john.doe@example.com

Affected components

- API Endpoints

The API endpoints reveal complete user information because security measures for access control and filtering remain absent.

- Client-Side Storage

The storage of PII in localStorage, sessionStorage or cookies remains insecure when protection is missing.

- URL Parameters / Query Strings

The direct inclusion of PII in URLs enables the view of information by browser history logs as well as referrers and server logs.

- HTML/JavaScript Code (Front-end)

When user data appears in the DOM it lacks cleaning procedures and the document uses innerHTML functions.

- Error Messages / Debug Logs

After an error occurs software tools show stack information that contains email addresses and usernames alongside IDs.

- Server Logs

The access or web server maintains records of PII contained within both request and response details.

- Session Management Systems

Session tokens or keys which appear in frontend HTTP responses or non-HttpOnly cookies.

- Third-Party Integrations

The transmission of user data occurs through built-in plugins and analytics tools as well as embedded scripts.

- Database Queries in Debug Mode

While running development or debug mode the application shows clear database query results containing PII information.

Impact assessment

Severity – Medium

Privacy Violations occur when personal data exposes users to privacy breaches as well as violates multiple legal privacy requirements including GDPR and CCPA.

The attackers use user profiling techniques to build thorough profiles of individual users through exposed information.

Attackers can perform targeted attacks after information disclosure because they utilize disclosed data for phishing techniques alongside social engineering methods and identity theft practice.

Session tokens along with identifiers that get leaked may grant unauthorized access to user accounts.

Data protection regulation violations result in compliance breaches that produce legal penalties with monetary fines.

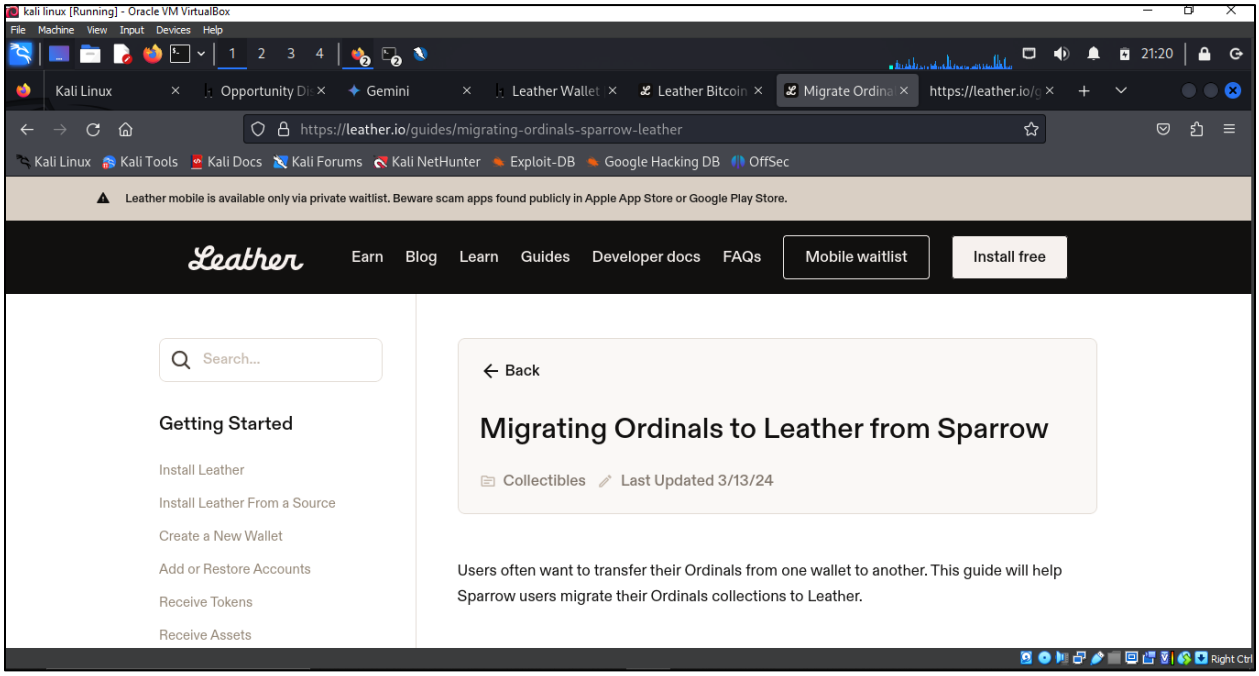
User trust may decrease because of improper handling regarding their personal data which leads to reputation damage.

The disclosure of PII enables attackers to exploit multiple system weaknesses starting from account enumeration weaknesses through brute force attacks.

The disclosure of data leads to reduced business operations through lost customers plus partner relationships as well as revenue decreases.

Steps to reproduce with Proof of Concept (poc)

1. First, I navigated the link and observed it.



2. In the OWASP ZAP tool, I saw that an email was found embedded in the HTML source.

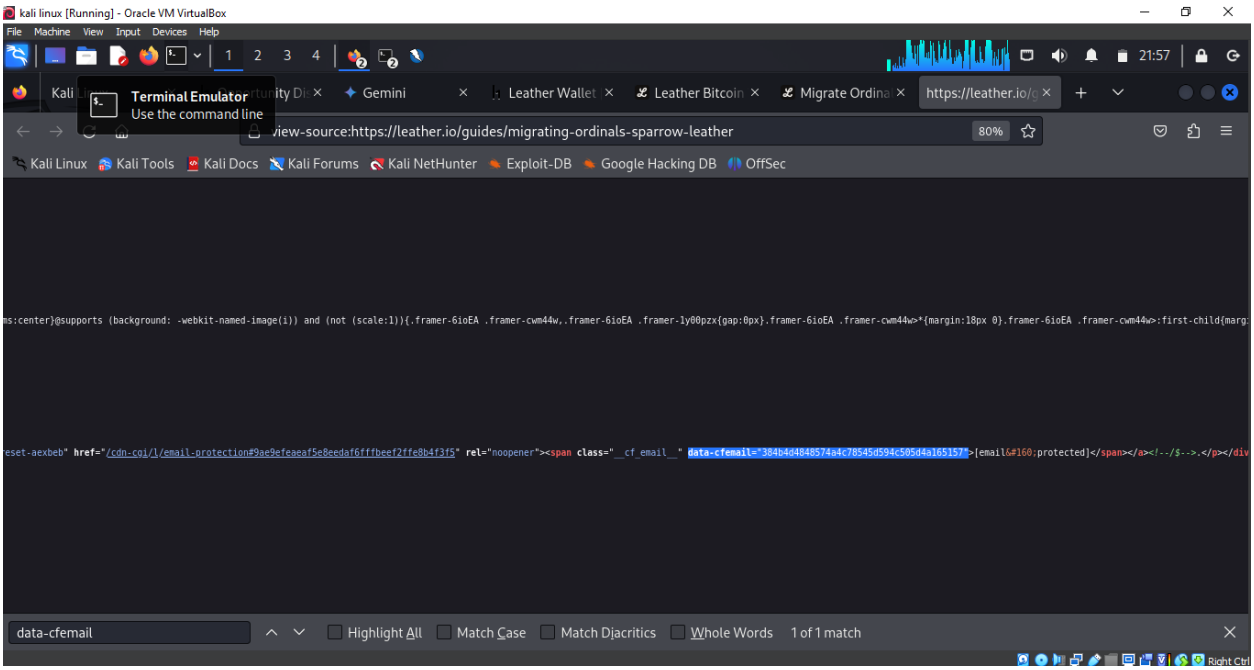
What is data-cfemail?

Certain email addresses within the site have been obfuscated by Cloudflare. This is stored in the data-cfemail attribute. This technique essentially querying through JavaScript on the client-side for decoding and rendering the email in the browser. So, no basic bots can scrape the address.

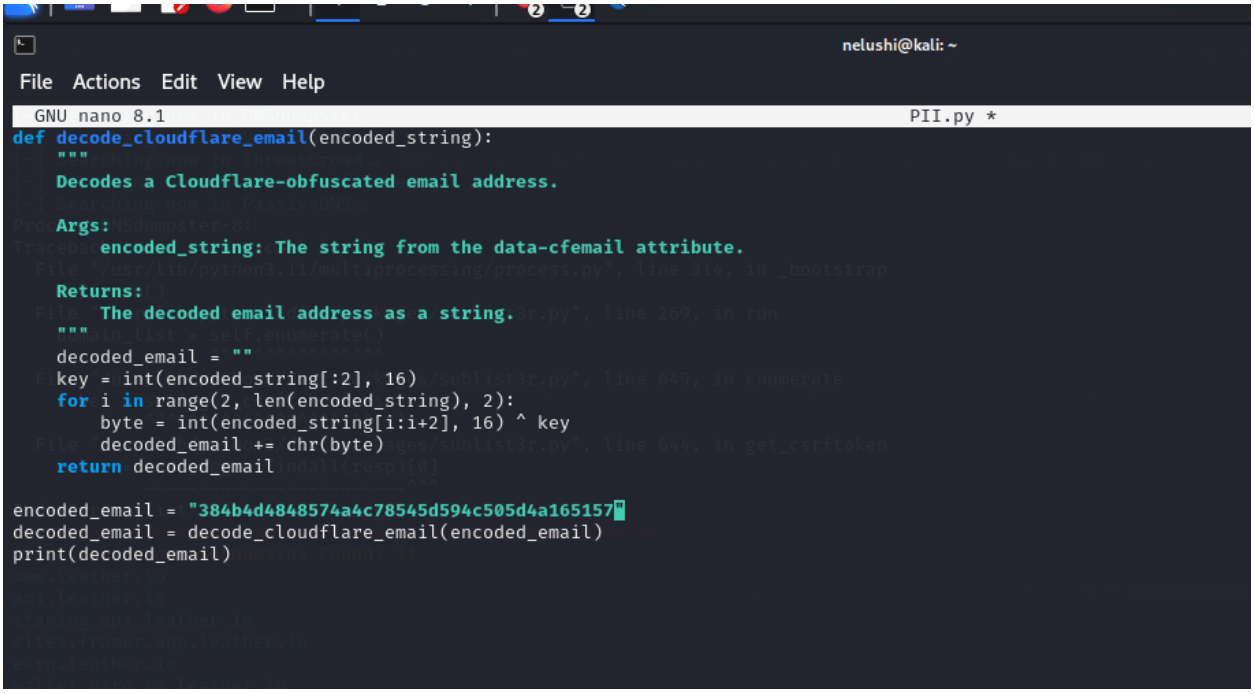
However these obfuscated email can be easily decoded by automated scripts which show outputs like PIIs such as user email or admin email. Hence the logic of decoding is public and very simple, which makes this protection insufficient against scraping and reconnaissance.

An Attacker extracts these values and decodes them from the html source can collect such information, which exposes contact information that can be then used for phishing spam or to carry out social engineering attacks.

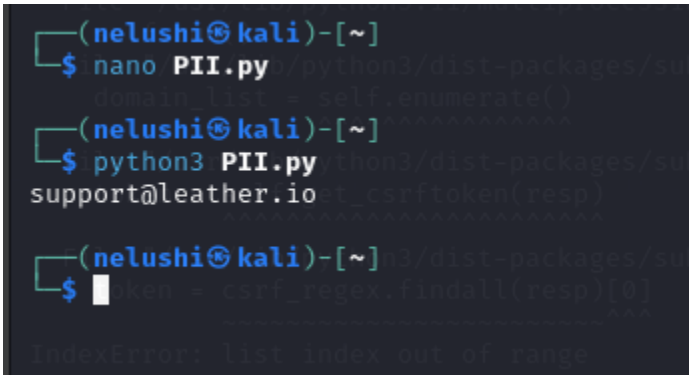
3. I checked the source code of the site to check data-cfemail attributes.



4. I created a simple python script that decodes the encoded email.



And I executed the python code.



Even though the decoded email in this case isn't sensitive, this way you can decode any email that belongs to users or administrators.

This counts as **low-severity PII disclosure**, especially if no authentication is required to access the source.

Overall Reflection

This finding suggests that even lower levels of exposure, such as data-cfemail, can create the risk of PII disclosure if they are not secured appropriately. Even though these data is obfuscated, the emails can be easily decoded, making them open to scraping and phishing. It indicates the need for improved data protection and reinforces that all PII should be treated with the same level of safety at the back end as it is visible in the front end.

Proposed Mitigation methods

Never expose emails in the source code, even though they are obfuscated and use contact forums instead of direct showing of addresses.

Implement adequate access controls to prevent unauthorized access to PII by unauthenticated users.

Use more secured obfuscation techniques if email visibility is required, and expose events only to the users whose presence is strictly necessary in that exposure.

Rates-limit, use CAPTCHA, or any bot detection tool to monitor and restrict bots from automated scraping.

Educate developers about what client-side obfuscation can and cannot do, besides ensuring that the PII is protected both while at rest and in traveling.