Project Report: Design and Implementation of Python-based Keylogger for Windows OS in a controlled virtual environment

Problem Statement :

Keystroke logging is a common tactic used by malicious actors to steal confidential information, including passwords and personal data. This project aims to ethically demonstrate how a keylogger operates in a secure, virtual environment and raises awareness about the importance of securing systems against such threats**.**

Title: Design and Implementation of Python-based Keylogger for Windows OS in a controlled virtual environment

Submitted by: Tanushree Gadekar-124A1033
Anirudh Iyer -124A1044
Revathi Gounder -124A1037
Sakshi Joshi -124A1050

Branch-Computer Engineering
Year-Second Year

Institute -Sies Graduated School of Technology

Guided by- Namrata P

Academic Year-2025-2026

Keylogger

- In the age of the internet, **cybersecurity** has become a critical concern for individuals, organizations, and governments. One of the most common and dangerous threats in the cybersecurity landscape is the use of **keyloggers**—software tools that record keystrokes made by users, often without their knowledge. While keyloggers are typically associated with malicious intent, they are also used ethically by cybersecurity professionals for penetration testing and research purposes.

- This project aims to demonstrate how a **browser-based keylogger** can be created and deployed in a controlled environment for educational and testing purposes. The system involves a **fake login page** that mimics a real website and captures keystrokes as a user types into the form fields. The captured keystrokes are then logged and sent to a back-end script (`log.php`) running on a **Kali Linux server**. These logs can be accessed from another device on the same network (or even from another network using tools like **Ngrok**).

- By building and analyzing this keylogger, students and researchers can gain practical insights into how phishing attacks and data capture work. It also emphasizes the importance of **web application security**, **secure coding practices**, and **user awareness**. This project serves both as a **technical demonstration** and a **cybersecurity awareness exercise**, showing how easy it can be to trick users into revealing sensitive information and how such methods can be detected and prevented.

Objectives

To Understand the Working of a Keylogger:

- Learn how keyloggers function by capturing user input (keystrokes) from a web interface in real time.

To Build a Simulated Phishing Environment:

- Create a fake login page that mimics a real website, designed to collect user credentials for ethical testing purposes.

To Implement Keystroke Logging Using JavaScript and PHP:

- Use JavaScript to capture input as the user types and PHP to store the data securely on the server.

To Explore Network-Based Keylogging:

- Demonstrate how keylogging data can be captured from one device (Laptop A or a phone) and received on another device (Laptop B) connected over the same or different networks.

To Host a Local Web Server on Kali Linux:

- Serve the fake webpage using Python’s HTTP server and learn how Kali Linux tools support ethical hacking and testing.

To Educate About Web-Based Threats:

- Provide a hands-on example of how attackers can use keyloggers in phishing attacks to steal sensitive data like usernames and passwords.

To Promote Ethical Hacking Awareness:

- Emphasize the need for responsible testing and awareness of cybersecurity vulnerabilities in web applications.

Tools and Technologies Used

1. HTML (HyperText Markup Language)

Usage:

HTML was used to design the **fake login web page** (i.e., the front-end interface).

Function in Project:

- Created fields like `Username` and `Password`.
- Served as the surface that captures user keystrokes through JavaScript events.

```
<input type="text" id="username" placeholder="Username">

<input type="password" id="password" placeholder="Password">
```

2. **JavaScript**

Usage:

JavaScript was used to **capture keystrokes** from the input fields and send them to the server.

Function in Project:

- Added event listeners (`onkeypress`, `oninput`, etc.).
- Used `fetch()` or `XMLHttpRequest` to send the keystrokes to a PHP file on the server.

3. PHP (Hypertext Preprocessor)

Usage:

PHP was used as the backend script to store the keystrokes sent from JavaScript.

Function in Project:

- **Captured raw data using `file_get_contents("php://input")`.**
- **Wrote that data into a file (`log.txt`) for later analysis.**

4.Apache HTTP Server

Usage:
Apache was the local web server used to host the fake website (`index.html`, `login.html`, and `log.php`).

Function in Project:

- **Served the HTML pages and handled HTTP requests.**
- **Made the keylogger accessible on a local IP (e.g., `http://192.168.0.104`).**

How to use it:

Files were placed in `/var/www/html/` on Kali Linux.

Start the server using:

bash
CopyEdit

```
sudo service apache2 start
```

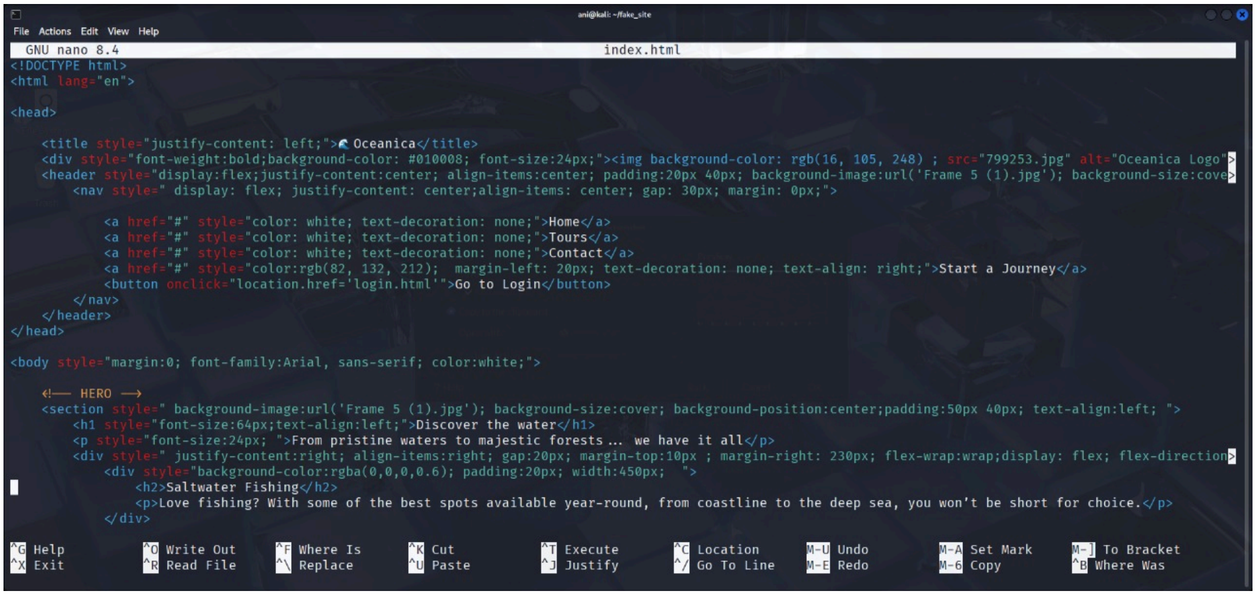
6. Kali Linux

Usage:

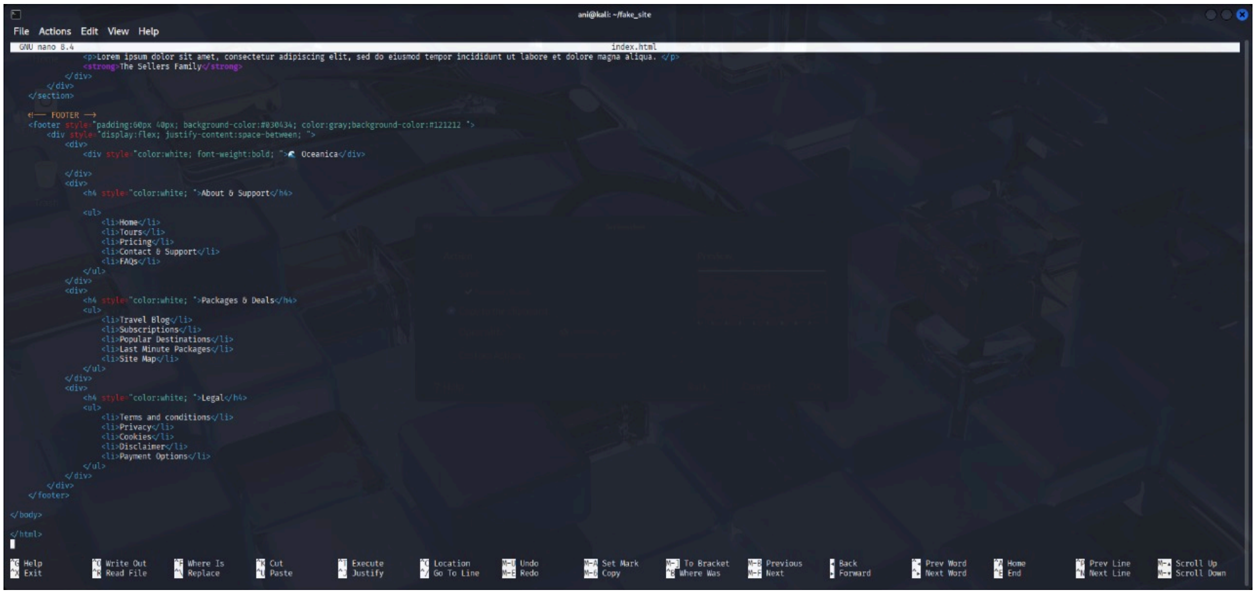
Kali Linux was the operating system used to host the server and run the keylogger.

Function in Project:

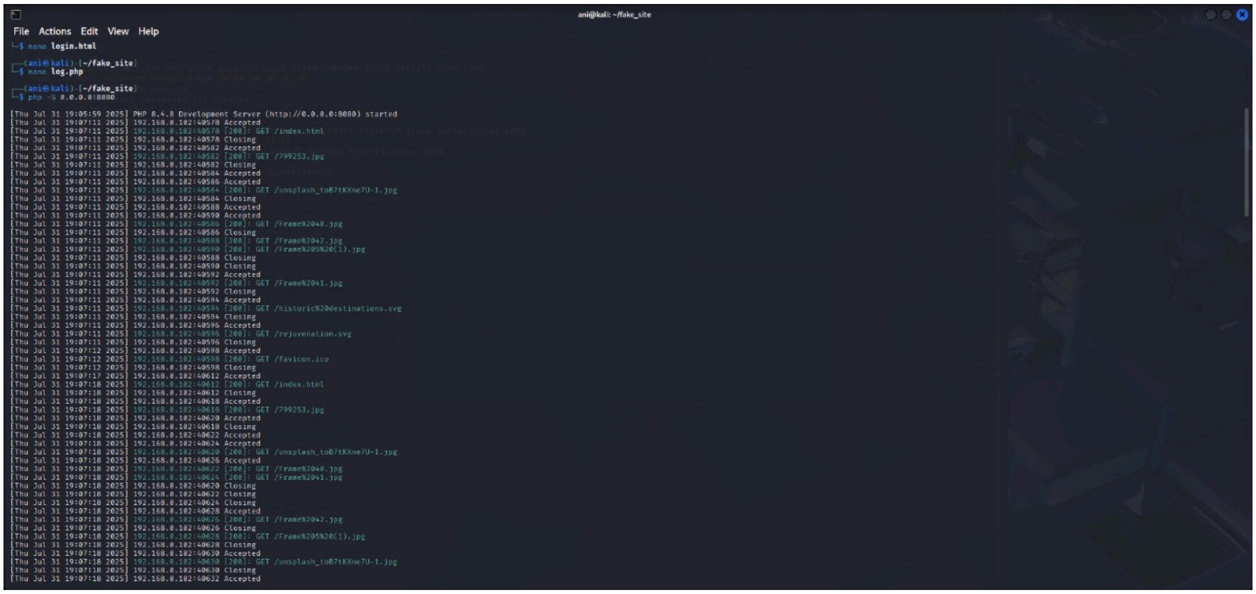
- **Hosted the fake web page files.**
- **Ran Apache server and PHP scripts.**
- **Used terminal to monitor logs and run the program.**



```
  GNU nano 8.4                                         index.html
<!DOCTYPE html>
<html lang="en">

<head>

    <title style="justify-content: left;">🌊 Oceanica</title>
    <div style="font-weight:bold;background-color: #010008; font-size:24px;"><img background-color: rgb(16, 105, 248) ; src="799253.jpg" alt="Oceanica Logo">
    <header style="display:flex;justify-content:center; align-items:center; padding:20px 40px; background-image:url('Frame 5 (1).jpg'); background-size:cove>
        <nav style=" display: flex; justify-content: center;align-items: center; gap: 30px; margin: 0px;">

            <a href="#" style="color: white; text-decoration: none;">Home</a>
            <a href="#" style="color: white; text-decoration: none;">Tours</a>
            <a href="#" style="color: white; text-decoration: none;">Contact</a>
            <a href="#" style="color:rgb(82, 132, 212);  margin-left: 20px; text-decoration: none; text-align: right;">Start a Journey</a>
            <button onclick="location.href='login.html'">Go to Login</button>
        </nav>
    </header>
</head>

<body style="margin:0; font-family:Arial, sans-serif; color:white;">

    <!-- HERO -->
    <section style=" background-image:url('Frame 5 (1).jpg'); background-size:cover; background-position:center;padding:50px 40px; text-align:left; ">
        <h1 style="font-size:64px;text-align:left;">Discover the water</h1>
        <p style="font-size:24px; ">From pristine waters to majestic forests ...  we have it all</p>
        <div style=" justify-content:right; align-items:right; gap:20px; margin-top:10px ; margin-right: 230px; flex-wrap:wrap;display: flex; flex-direction>
            <div style="background-color:rgba(0,0,0,0.6); padding:20px; width:450px;   ">
                <h2>Saltwater Fishing</h2>
                <p>Love fishing? With some of the best spots available year-round, from coastline to the deep sea, you won’t be short for choice.</p>
            </div>
```



```
 GNU nano 8.4                                                                    index.html
            <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. </p>
            <strong>The Sellers Family</strong>
        </div>
    </div>
</section>

<!-- FOOTER -->
<footer style="padding:60px 40px; background-color:#030434; color:gray;background-color:#121212 ">
    <div style="display:flex; justify-content:space-between; ">
        <div>
            <div style="color:white; font-weight:bold; ">🌊 Oceanica</div>

        </div>
        <div>
            <h4 style="color:white; ">About & Support</h4>

            <ul>
                <li>Home</li>
                <li>Tours</li>
                <li>Pricing</li>
                <li>Contact & Support</li>
                <li>FAQs</li>
            </ul>
        </div>
        <div>
            <h4 style="color:white; ">Packages & Deals</h4>
            <ul>
                <li>Travel Blog</li>
                <li>Subscriptions</li>
                <li>Popular Destinations</li>
                <li>Last Minute Packages</li>
                <li>Site Map</li>
            </ul>
        </div>
        <div>
            <h4 style="color:white; ">Legal</h4>
            <ul>
                <li>Terms and conditions</li>
                <li>Privacy</li>
                <li>Cookies</li>
                <li>Disclaimer</li>
                <li>Payment Options</li>
            </ul>
        </div>
    </div>
</footer>

</body>

</html>

^G Help       ^O Write Out  ^F Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket  M-B Previous  ◂ Back        ^◂ Prev Word  ^A Home       ^P Prev Line  M-▴ Scroll Up
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^B Where Was    M-F Next      ▸ Forward     ^▸ Next Word  ^E End        ^N Next Line  M-▾ Scroll Down
```



```
└─$ nano login.html

┌──(ani㉿kali)-[~/fake_site]
└─$ nano log.php

┌──(ani㉿kali)-[~/fake_site]
└─$ php -S 0.0.0.0:8080

[Thu Jul 31 19:05:59 2025] PHP 8.4.8 Development Server (http://0.0.0.0:8080) started
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40578 Accepted
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40578 [200]: GET /index.html
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40578 Closing
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40582 Accepted
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40582 [200]: GET /799253.jpg
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40582 Closing
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40584 Accepted
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40586 Accepted
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40584 [200]: GET /unsplash_toB7tKXne7U-1.jpg
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40584 Closing
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40588 Accepted
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40590 Accepted
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40586 [200]: GET /Frame%2040.jpg
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40586 Closing
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40588 [200]: GET /Frame%2042.jpg
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40590 [200]: GET /Frame%205%20(1).jpg
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40588 Closing
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40590 Closing
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40592 Accepted
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40592 [200]: GET /Frame%2041.jpg
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40592 Closing
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40594 Accepted
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40594 [200]: GET /historic%20destinations.svg
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40594 Closing
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40596 Accepted
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40596 [200]: GET /rejuvenation.svg
[Thu Jul 31 19:07:11 2025] 192.168.0.102:40596 Closing
[Thu Jul 31 19:07:12 2025] 192.168.0.102:40598 Accepted
[Thu Jul 31 19:07:12 2025] 192.168.0.102:40598 [200]: GET /favicon.ico
[Thu Jul 31 19:07:12 2025] 192.168.0.102:40598 Closing
[Thu Jul 31 19:07:17 2025] 192.168.0.102:40612 Accepted
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40612 [200]: GET /index.html
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40612 Closing
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40618 Accepted
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40618 [200]: GET /799253.jpg
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40620 Accepted
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40618 Closing
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40622 Accepted
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40624 Accepted
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40620 [200]: GET /unsplash_toB7tKXne7U-1.jpg
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40626 Accepted
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40622 [200]: GET /Frame%2040.jpg
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40624 [200]: GET /Frame%2041.jpg
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40620 Closing
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40622 Closing
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40624 Closing
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40628 Accepted
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40626 [200]: GET /Frame%2042.jpg
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40626 Closing
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40628 [200]: GET /Frame%205%20(1).jpg
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40628 Closing
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40630 Accepted
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40630 [200]: GET /unsplash_toB7tKXne7U-1.jpg
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40630 Closing
[Thu Jul 31 19:07:18 2025] 192.168.0.102:40632 Accepted
```

Conclusion

This project demonstrates the implementation of a browser-based keylogger using HTML, JavaScript, and PHP on a local server hosted in Kali Linux. It works by simulating a fake login page to capture keystrokes in real time and store them for analysis through a simple backend script.

Through this project, we learned about front-end scripting, client-server communication, and the role of local networking in data transmission. It also highlights how insecure input handling can lead to vulnerabilities in web applications.

This keylogger project is an educational tool to understand cyberattack methods such as phishing and keylogging. It emphasizes the importance of ethical hacking, cybersecurity awareness, and secure web development practices. Although developed in a controlled environment, it reflects techniques used by attackers and provides practical insights for students and beginners in cybersecurity.