**Cross-site Scripting Detection tool**

**A Project Report**

***Submitted by:***

**Vipin (A217131523068)**

**(2023-2025)**

***In partial fulfilment for the award of the degree***

***of***

**MASTER OF SCIENCE**

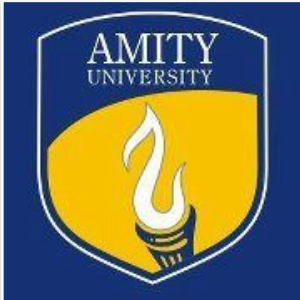
**IN**

**CYBER SECURITY**

***Submitted to :***

**Dr. Yudhveer Singh**

**Amity Institute of Information Technology**

****

**AMITY UNIVERSITY RAJASTHAN, JAIPUR**

**DECLARATION**

I hereby declare that the project entitled “Cross-stie scripting Detection Tool” submitted for the M.Sc. (Cyber Security) degree is my original work and the project has not formed the basis for the award of any other degree, diploma, fellowship, or any other similar titles.

**Signature of the Student**

**Place:**

**Date:**

**CERTIFICATE**

This is to certify that the project titled “Cross-stie scripting Detection Tool” is the bonafide work carried out Vipin, a student of M.Sc. (Cyber Security) of **DEPARTMENT OF AMITY INSTITUE OF INFORMATION TECHNOLOGY, AMITY UNIVERSITY RAJASTHAN, JAIPUR** during the academic year 2023-2025, in partial fulfilment of the requirements for the award of the degree of Master of Science (Cyber Security) and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

**Signature of the Guide**

**Place:**

**Date:**

**ACKNOWLEDGEMENT**

I would like to express my sincere appreciation and gratitude to all those who have contributed to the successful completion of our minor project. Your invaluable support and dedication have been instrumental in making this project a reality.

First and foremost, I would like to thank my project mentor Dr.Yudhveer Singh , Amity Institute of Information Technology, for their guidance, expertise, and continuous encouragement throughout the project. Their profound knowledge and insights have significantly influenced the project's direction and helped me overcome various challenges.

Furthermore, I would like to express my gratitude to the faculty members and experts who provided us with their valuable feedback, suggestions, and guidance during the project. Their inputs and constructive criticism have played a vital role in enhancing the quality and effectiveness of our work.

I would also like to acknowledge the support and understanding of our families and friends who stood by us during this project. Their encouragement, patience, and belief in our abilities have been a constant source of motivation.

Lastly, I would like to thank the University administration for providing us with the necessary resources and facilities to carry out this project. Without their support, this project would not have been possible.

In conclusion, I am deeply grateful to everyone who contributed to this minor project. Your involvement, dedication, and expertise have made this a rewarding and enriching experience. Thank you all once again for your unwavering support.

**CONTENTS**

|  |  |
| --- | --- |
| **Introduction**   * Problem Definition * Importance of Penetration Testing * Project Overview/Specifications * Hardware Specification |  |
| **Literature survey** |  |
| **System Analysis & Design**   * Requirement Specification * Flowcharts/ Methodology * Test Steps / Criteria * Algorithms and Code * Testing Process |  |
| **Results /Outputs** |  |
| **Conclusions / Recommendations** |  |
| **References** |  |
| **Appendices** |  |
|  |  |

**INTRODUCTION**

**Objective:-**

The “V “tool serves as an automated solution for detecting Cross-Site Scripting (XSS) vulnerabilities within web applications. By leveraging crafted payloads and HTTP requests, “V”efficiently scans target applications, analyzing responses for indications of reflected payloads. This automation significantly enhances testing efficiency, enabling security professionals and developers to identify potential vulnerabilities swiftly. With customizable parameters including payload, user-agent, and concurrency level, V offers tailored testing to suit diverse application contexts. Moreover, verbose output provides detailed insights into the testing process, while the option to save results facilitates in-depth analysis and vulnerability remediation efforts.

**Importance of Penetration Testing**

This tool holds significant importance in penetration testing due to its ability to within efficiently identify Cross-Site Scripting (XSS) vulnerabilities web applications. As XSS vulnerabilities remain prevalent and pose serious risks to application security, V enables penetration testers to comprehensively assess the security posture of target applications. By automating XSS detection, V saves time and resources, allowing testers to focus on other aspects of security assessment. Its customization options facilitate tailored testing to uncover vulnerabilities specific to the target application's environment. Additionally, verbose output aids in understanding testing outcomes, while saved results support detailed analysis and targeted remediation efforts, enhancing overall penetration testing effectiveness.

* **Efficiency**:- V automates the process of detecting XSS vulnerabilities, saving time and resources compared to manual testing methods.
* **Customization:**- The tool allows penetration testers to customize parameters such as payloads, user-agents, and concurrency levels, enabling tailored testing to suit the specific characteristics of each target application.
* **Comprehensive Testing:**- V conducts thorough testing to identify XSS vulnerabilities, ensuring a comprehensive assessment of the security posture of web applications.
* **Streamlined Workflow:-** By automating XSS detection, V streamlines the penetration testing workflow, allowing testers to focus on other critical aspects of security assessment.
* **Informed Decision-Making:**- The verbose output provided by V offers detailed insights into testing results, enabling penetration testers to make informed decisions during vulnerability remediation.
* **Enhanced Effectiveness:-** With its ability to uncover XSS vulnerabilities efficiently, V enhances the overall effectiveness of penetration testing efforts, leading to improved security outcomes for web applications.

**Project Overview/Specifications**

V, an advanced tool designed for penetration testing, specializes in the identification of Cross-Site Scripting (XSS) vulnerabilities within web applications. Leveraging automated techniques, V streamlines the process of vulnerability detection, significantly reducing the time and resources required for comprehensive security assessments. Its customizable parameters, including payloads, user-agents, and concurrency levels, allow testers to tailor the tool to the specific characteristics of each target application. Additionally, V provides detailed verbose output, empowering testers with comprehensive insights into testing results for informed decision-making during vulnerability remediation efforts. In summary, V serves as an indispensable asset in penetration testing, facilitating efficient and effective identification of XSS vulnerabilities within web applications.

**Hardware Specification**

The **V** tool, being a software application, does not have specific hardware requirements. However, it does require a system or environment where it can be installed and executed. Here are the general hardware requirements for running the V tool:

* **Operating System:** V is typically compatible with various operating systems such as Windows, macOS, and Linux. Ensure that your system meets the requirements of the chosen operating system.
* **Processor:** A modern processor with multiple cores, such as Intel Core i5 or AMD Ryzen, is recommended for optimal performance, especially when running multiple instances or conducting concurrent tests.
* **Memory (RAM):** At least 4 GB of RAM is recommended to handle the execution of the tool and accommodate the data processing requirements during testing.
* **Storage**: Sufficient disk space is necessary to store the tool's executable files, configuration files, and any generated output files. A few gigabytes of free disk space should be adequate.
* **Network Connectivity:** Since V interacts with web applications via HTTP requests, a stable internet connection is required to communicate with the target applications and receive responses.
* **Additional Considerations**: Depending on the scale and complexity of the testing environment, additional hardware resources such as network infrastructure (routers, switches), virtualization platforms (if using virtual machines), and external storage may be necessary.

**LITERATURE SURVEY**

In conducting a literature survey on the V tool for penetration testing, one would explore diverse sources to comprehensively understand its development, functionalities, and impact in cybersecurity. Delving into the origins and evolution of V offers insights into its design principles and capabilities. Technical documentation, user manuals, and guides provided by its developers offer essential guidance on installation, usage, and features. Research papers and publications in academic databases provide critical analyses of V's effectiveness in detecting XSS vulnerabilities, potentially comparing it with other tools. Real-world case studies and use cases highlight practical applications, while community forums and discussions offer valuable user feedback and experiences. Industry reports and training materials further enrich understanding, depicting V's role in enhancing security practices and professional skill development. Through these varied sources, a literature survey illuminates V's significance, strengths, and contributions to penetration testing and cybersecurity endeavors.

**System Analysis & Design**

**Requirement Specification**

**Hardware:**

* Laptop
* Power supply
* Wi-Fi Network/Internet Connection

**Software:**

* Compatible OS on host computer
* Additional libraries / python golang

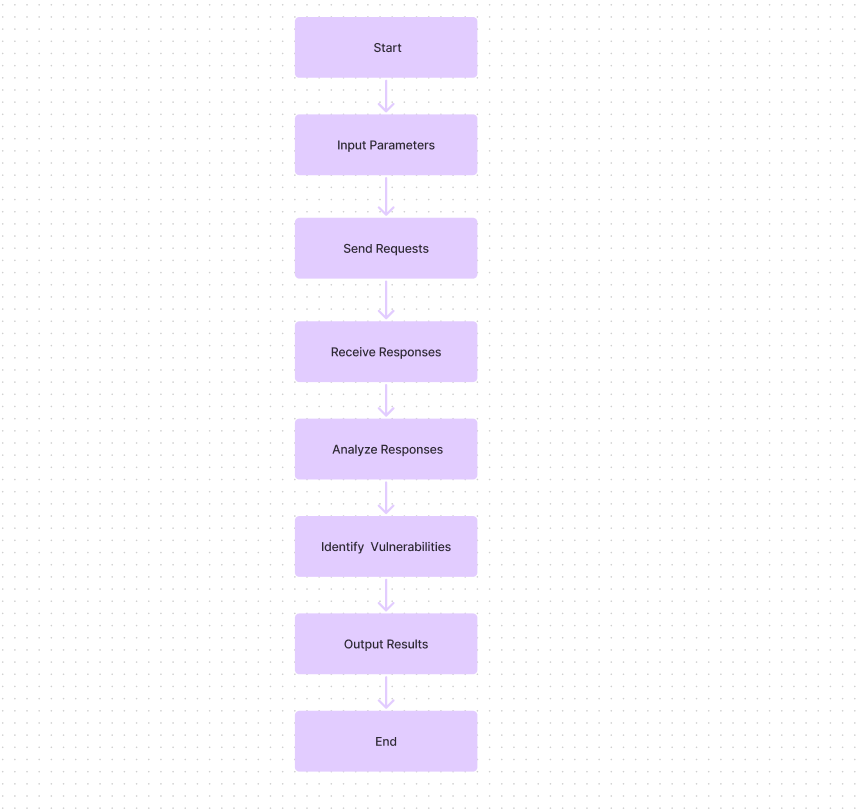
**Design Considerations:**

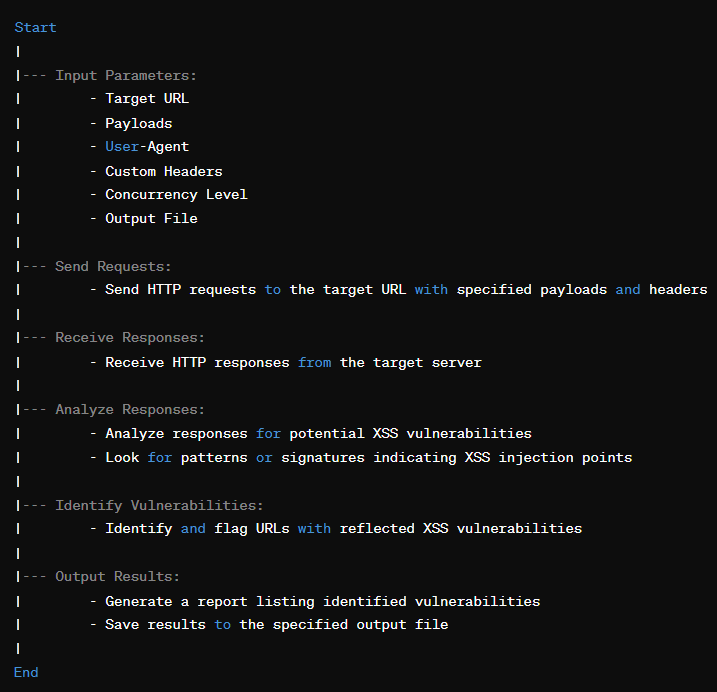
* **Ethical Usage:** Permission-based testing only.
* **Modularity:** Easy modification and expansion.
* **Documentation:** Thorough documentation for future reference.

**Additional Considerations:**

* **Power Consumption:** Optimized for low power use.
* **Legal and Regulatory:** Be aware local of restrictions.

**Flowcharts/ Methodology**

****

****

**Development Environment Setup:**

Setting up the development environment for a tool like V involves preparing the necessary software and dependencies to build, test, and deploy the application. Here's a general guide on how you can set up the development environment:

**Operating System:** Ensure you have a compatible operating system installed on your development machine. V may be developed and tested on various operating systems like Linux, macOS, or Windows.

**Programming Language:-**  For making this project I m using the python language for writing my code .

**Version Control:** Install a version control system like Git to manage the source code. Clone the project repository to your local machine to start working on it.

**Development Tools:** Install the necessary development tools and software required for building and testing the application. This may include IDEs (Integrated Development Environments) or text editors suitable for the chosen programming language. For finding the endpoint for searching I m using the Waybackurl tool .

**Dependencies:** Identify and install any dependencies or libraries required by V to compile and run successfully. Use package managers specific to your programming language to install dependencies easily.

Like:- for running the code you should installed the python .

**Build and Test:** Build the application using the provided build scripts or commands. Run unit tests and integration tests to ensure that the application behaves as expected and meets the specified requirements.

**Documentation:** Familiarize yourself with the project documentation, including README files, developer guides, and contribution guidelines. This will provide valuable information on how to develop, test, and contribute to the project.

**Collaboration Tools:** If you're working in a team, consider using collaboration tools like issue trackers, project boards, and communication channels to coordinate development efforts and track progress.

**Software Development:**

* Install a compatible operating system on your host computer (Windows, macOS, or Linux).
* Download the necessary file for this like install the Waybackurl tool should be downloaded on your system for finding the endpoints of the target.
* You should have python tool for run this tool .
* V tool work on the endpoint , find the url where it have a vulnerability of XSS.

**Testing and Evaluation:**

1. **Ethical Considerations:**

* Ensure that testing activities are conducted ethically and responsibly, respecting the privacy and security of target systems and data.
* Obtain proper authorization and consent before performing security testing on web applications, adhering to legal and regulatory requirements.
* Avoid causing harm or disruption to production systems or services during testing, and take precautions to prevent unintended consequences.

1. **Initial Testing:**

* Begin testing with a comprehensive review of the V tool's features, functionalities, and configuration options.
* Conduct initial tests to verify basic functionality, such as launching the tool, setting up scanning parameters, and initiating scans on sample web applications.
* Verify that the tool can accurately detect common XSS vulnerabilities and provide actionable results and recommendations for remediation.

1. Evaluation:

* Evaluate the effectiveness of the V tool in detecting XSS vulnerabilities across different types of web applications, frameworks, and technologies.
* Assess the tool's accuracy, precision, and reliability in identifying genuine vulnerabilities while minimizing false positives and false negatives.
* Measure the tool's performance in terms of speed, scalability, resource usage, and compatibility with various environments.
* Solicit feedback from users, security professionals, and stakeholders to gather insights on the tool's usability, user experience, and effectiveness in real-world scenarios.
* Consider conducting comparative evaluations with other XSS detection tools or manual testing methods to benchmark the V tool's performance and capabilities.

**Test Steps**

**Setup and Configuration:**

* Install and configure the V tool according to the provided documentation.
* Verify that the tool is properly configured with necessary parameters such as target URLs, scanning options, and payload lists.

**Initial Scan:**

* Initiate an initial scan using the V tool on a sample web application or URL.
* Monitor the scan progress and verify that the tool successfully analyzes the target for XSS vulnerabilities.

**Payload Injection:**

* Customize the payload list or provide specific payloads to the V tool for injection testing.
* Verify that the tool effectively injects payloads into input fields, parameters, and other parts of the web application to identify potential XSS vulnerabilities.

**Detection and Analysis:**

* Analyze the scan results generated by the V tool to identify detected XSS vulnerabilities.
* Verify that the tool accurately identifies and classifies XSS vulnerabilities based on severity, impact, and exploitability.

**False Positive/Negative Verification:**

* Validate the identified vulnerabilities to confirm true positives and distinguish them from false positives.
* Investigate any instances of false positives or false negatives to understand the root cause and refine the scanning parameters or payload lists accordingly.

**Reporting:**

* Review the generated scan reports provided by the V tool.
* Ensure that the reports contain comprehensive details about detected vulnerabilities, including affected URLs, payloads, and recommendations for remediation.

**User Feedback and Usability:**

* Gather feedback from users and testers regarding the usability and effectiveness of the V tool.
* Assess user satisfaction, ease of use, and overall experience with the tool's interface, features, and functionality.

**Performance and Scalability:**

* Evaluate the performance and scalability of the V tool by testing its ability to handle large-scale scanning tasks and diverse web application architectures.
* Measure factors such as scan speed, resource utilization, and stability under varying load conditions.

**Security and Compliance:**

* Assess the security posture of the V tool itself, including its handling of sensitive data, encryption mechanisms, and protection against vulnerabilities.
* Ensure that the tool complies with relevant security standards, regulations, and best practices.

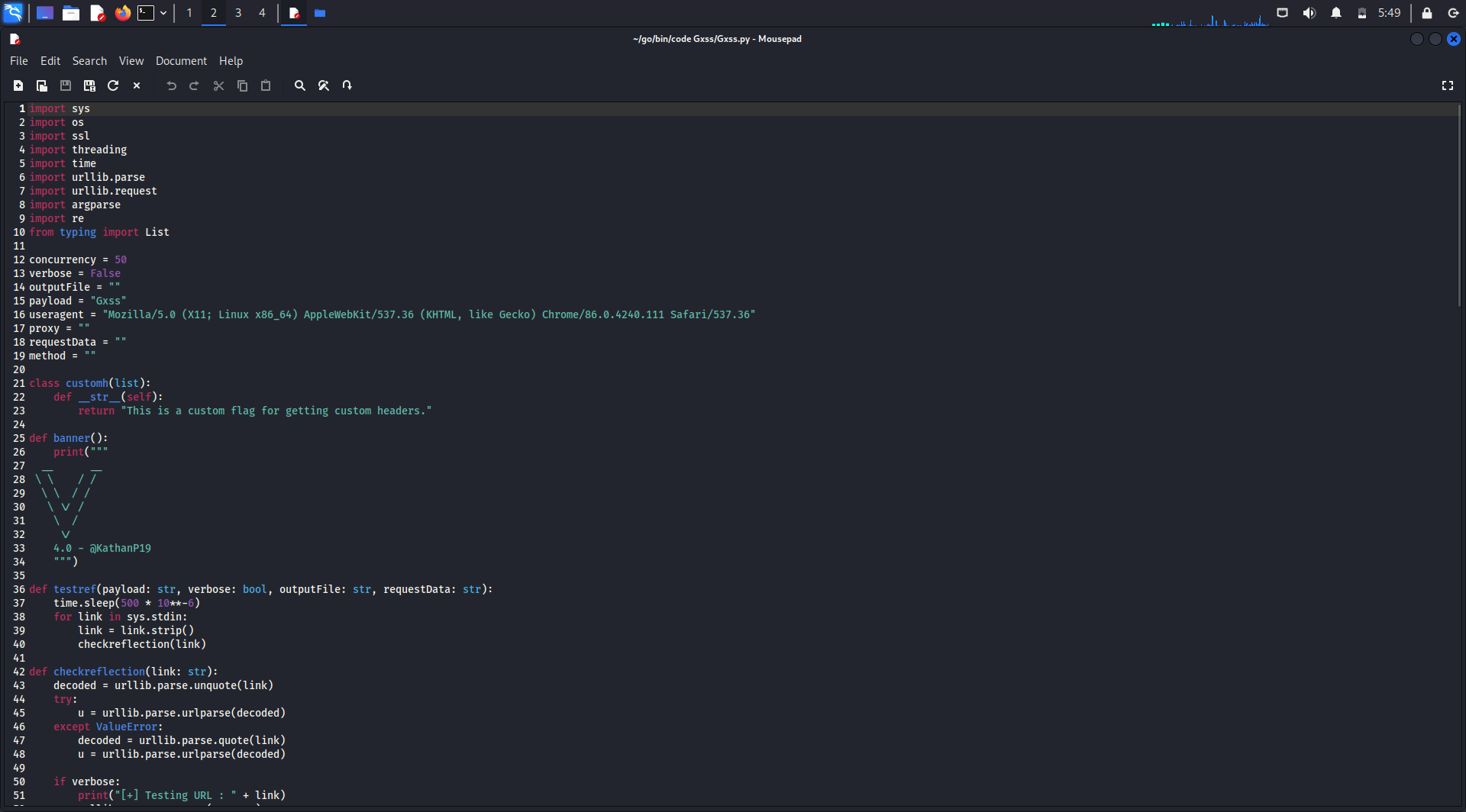
**Documentation and Training:**

* Review the documentation provided with the V tool, including installation guides, user manuals, and technical references.
* Verify that the documentation is accurate, up-to-date, and accessible to users for effective usage and troubleshooting.

**Algorithm and Methodology of V Tool**

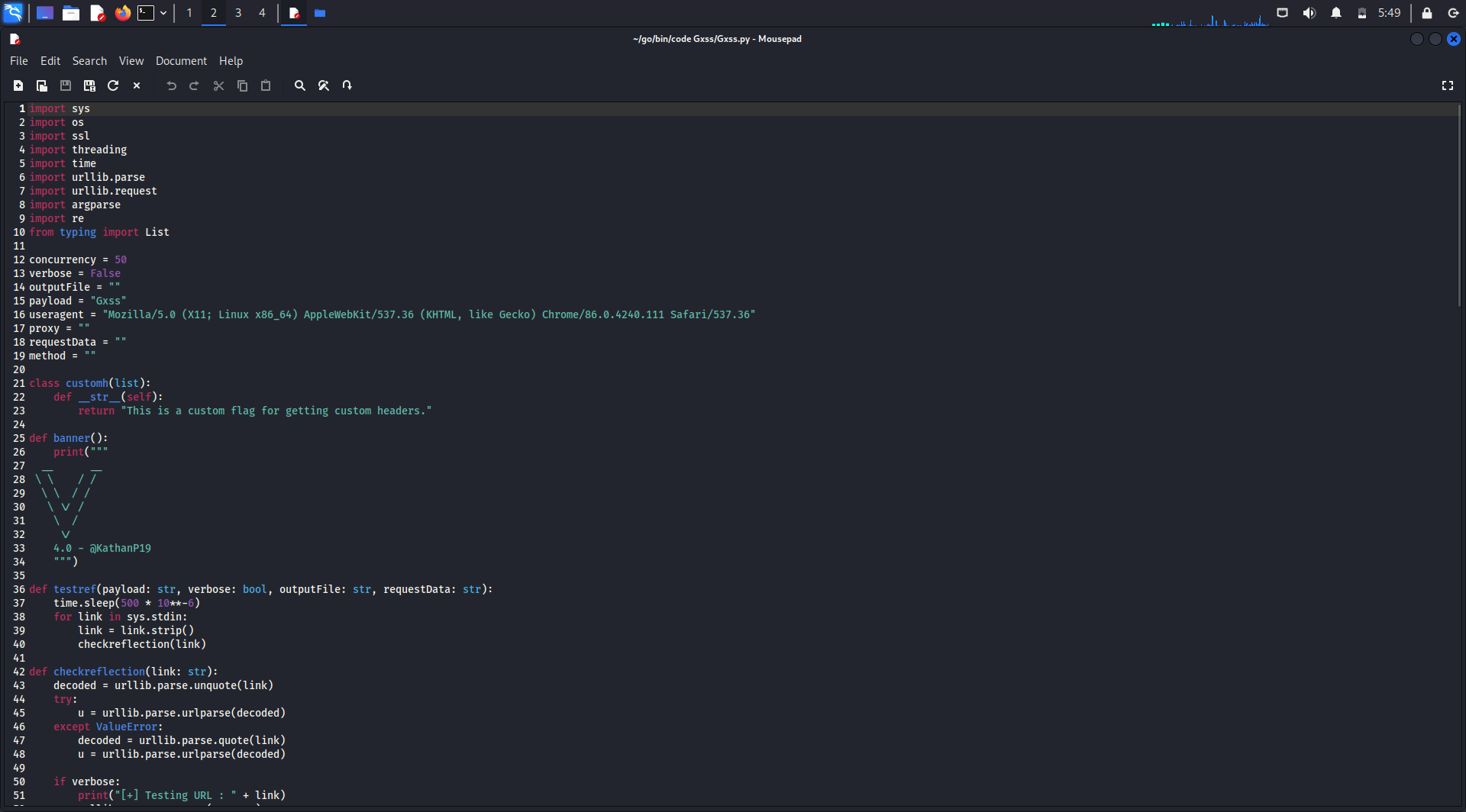
**Initialization:**

* Import required modules and libraries including sys, os, ssl, threading, time, urllib, argparse, and re.
* Initialize global variables for concurrency, verbosity, output file, payload, user agent, proxy, request data, and method.



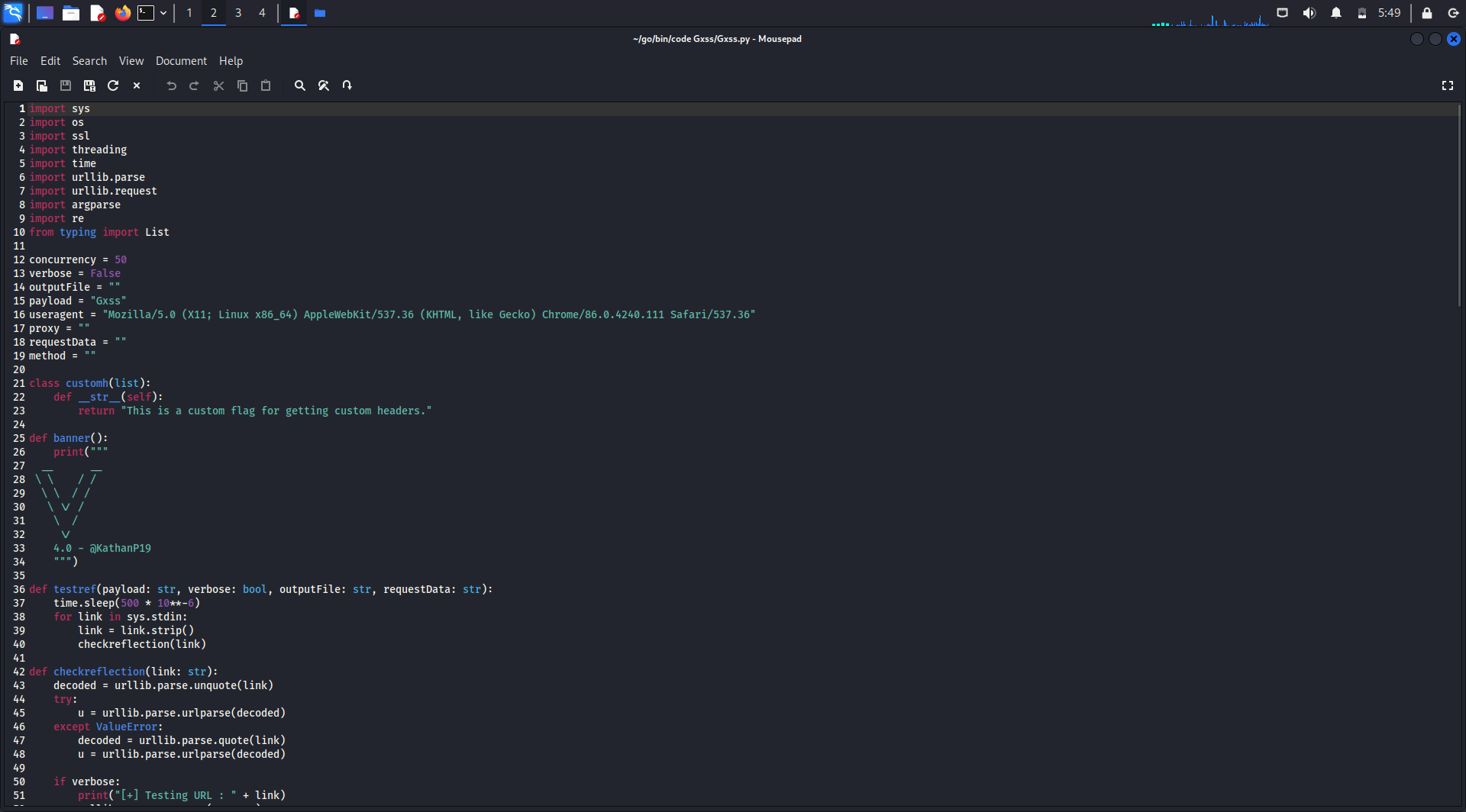
**Banner Display:**

* Define a function banner() to print the banner of the V tool displaying its version and creator.



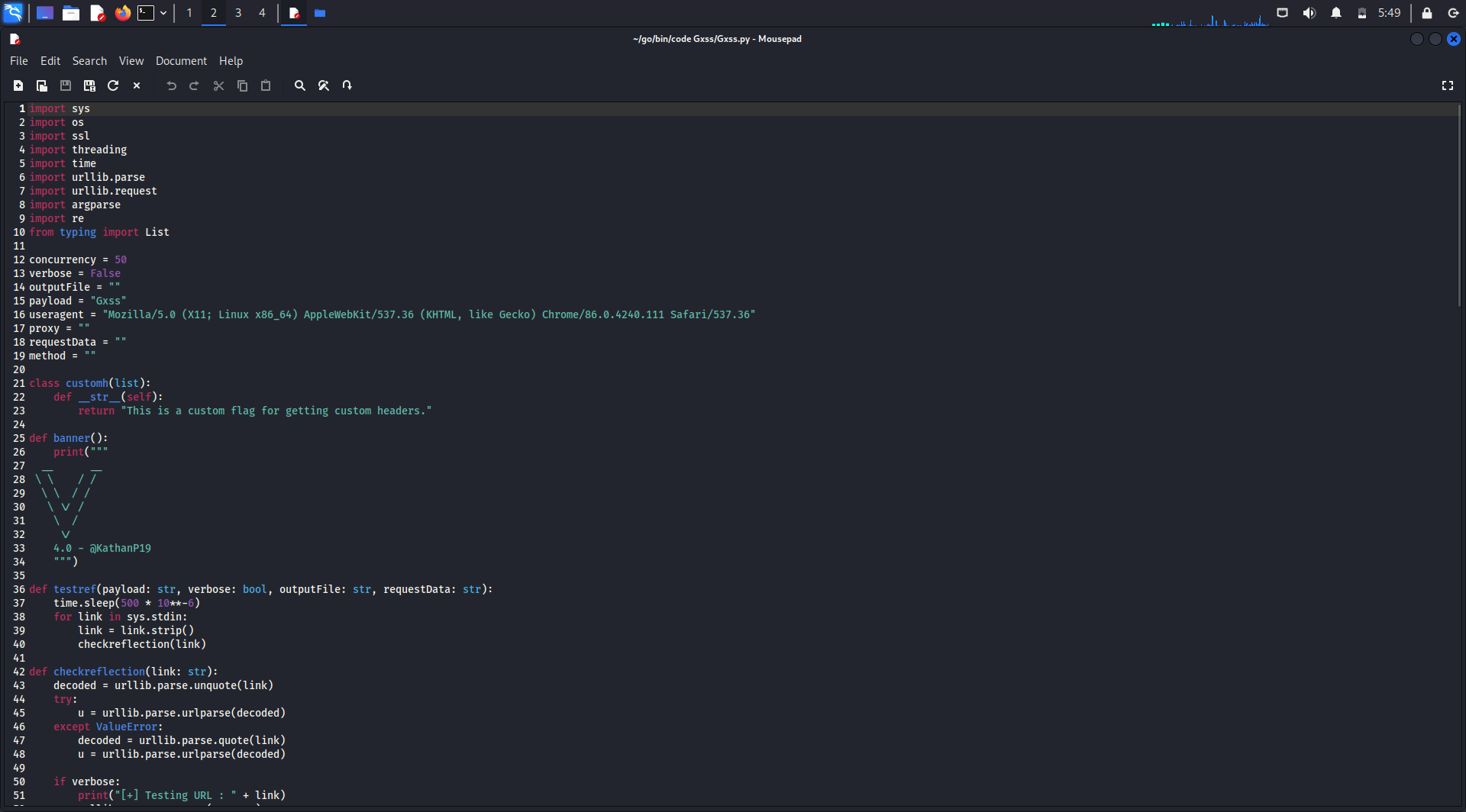
**Payload Testing Function:**

* Define a function testref() to perform payload testing on input URLs.
* Sleep for a short duration to avoid overwhelming the target server.
* Iterate through each URL from standard input and call the checkreflection() function.



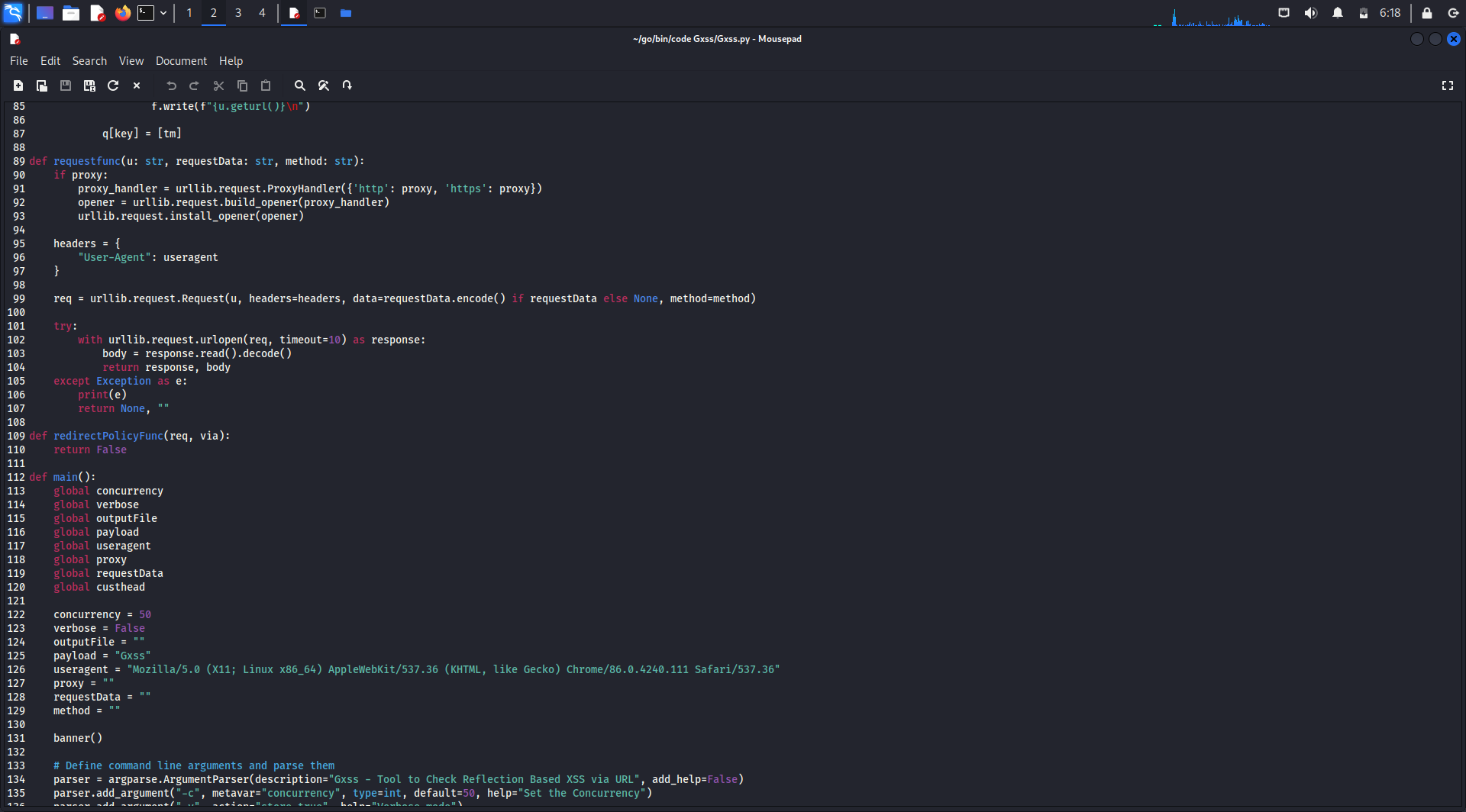
**Reflection Checking Function**

* Define checkreflection() function to analyze the given URL for reflection vulnerabilities.
* Decode the URL to handle encoded characters.
* Parse the URL and extract query parameters.
* Determine the request method (GET or POST) based on the presence of request data.
* Inject the payload into each query parameter and send the request using requestfunc() function.
* Search for the presence of the payload in the response body and print the reflected URL if found.
* Optionally, write the reflected URL to the output file.



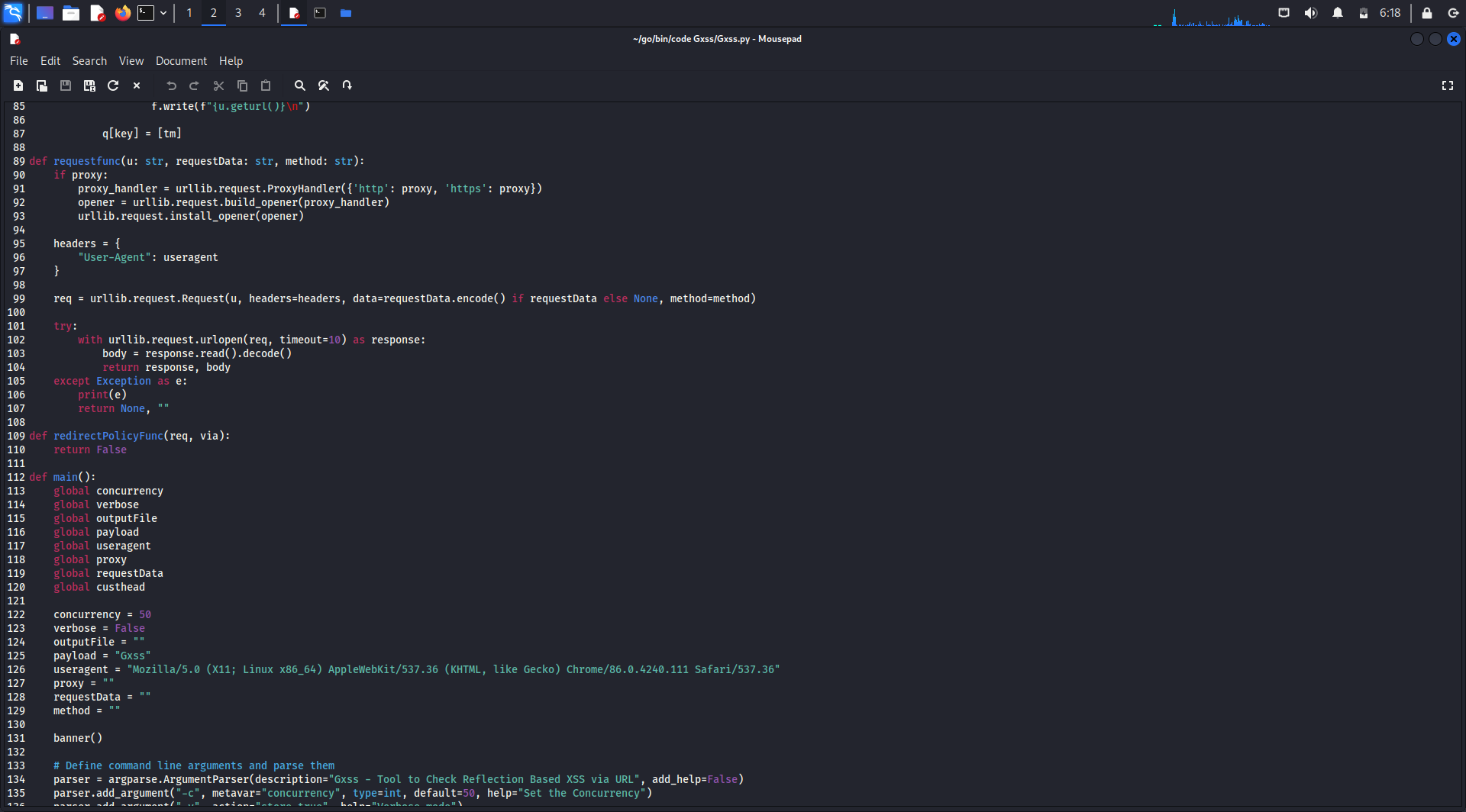
**HTTP Request Function:**

* Define requestfunc() function to send HTTP requests to the target URL.
* Set up proxy settings if provided.
* Construct the request headers with the specified user agent.
* Create an HTTP request object and send it using urllib.request.urlopen().
* Return the HTTP response object and response body.



**Main Function:**

* Define the main() function to parse command-line arguments, initialize variables, and execute payload testing.
* Parse command-line arguments using argparse.ArgumentParser.
* Set up concurrency, verbosity, payload, output file, request data, proxy, and user agent based on the parsed arguments.
* If payload is specified, create the output file if needed and initiate payload testing with specified concurrency.
* Print completion message if verbosity is enabled.



**Execution:**

* Execute the main() function if the script is run directly.
* Pass control to the main() function to start the V tool execution.

**Testing Process**

**Finding the Endpoint:-** To find the endpoints of a website using the WaybackURL tool.

**Install WaybackURL:**

* If you haven't already installed the WaybackURL tool, you can do so by downloading it from the official repository or by using a package manager like pip.

**Collect URLs:**

* Gather the URLs of the target website that you want to analyze. These URLs can include the root domain (e.g., example.com) or specific pages (e.g., example.com/login).

**Run WaybackURL:**

* Open your terminal or command prompt.
* Use the waybackurls command followed by the target domain to fetch archived URLs related to the website. For example:
* Copy code
* waybackurls example.com

**Analyze Endpoints:**

* Examine the list of URLs returned by WaybackURL. These URLs represent archived instances of the target website.
* Look for endpoints such as /api, /login, /admin, /wp-admin, or any other paths that may indicate functionality or sensitive areas of the website.

**Filtering:**

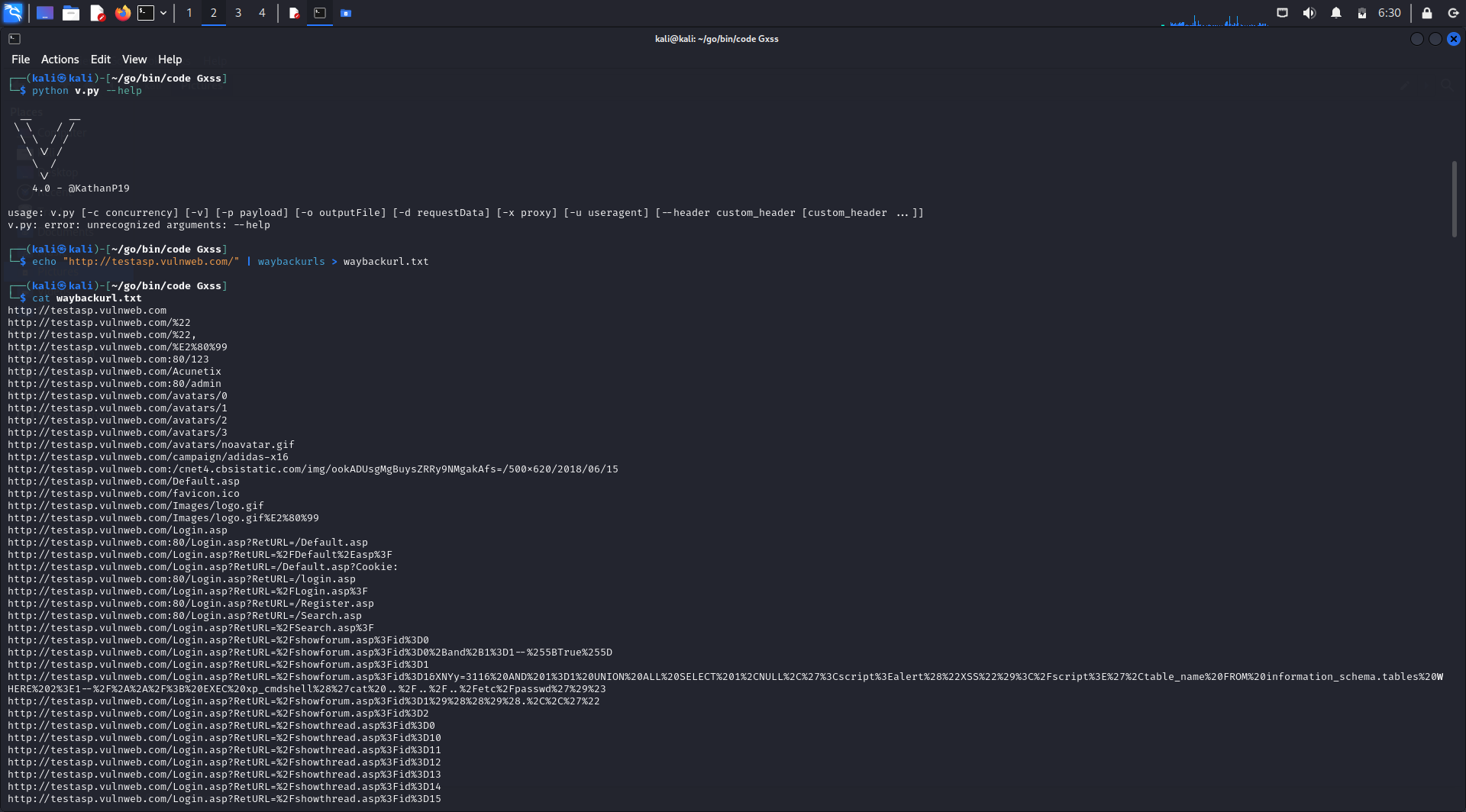
* Use filtering options provided by WaybackURL to narrow down the results if necessary. You can filter URLs based on specific patterns, keywords, or response status codes.

**Exploration:**

* Visit each endpoint URL in a web browser or use additional tools to analyze its functionality, response, and potential vulnerabilities.
* Pay attention to any endpoints that seem interesting or potentially risky, such as administrative interfaces, login pages, or API endpoints.

**Further Investigation:**

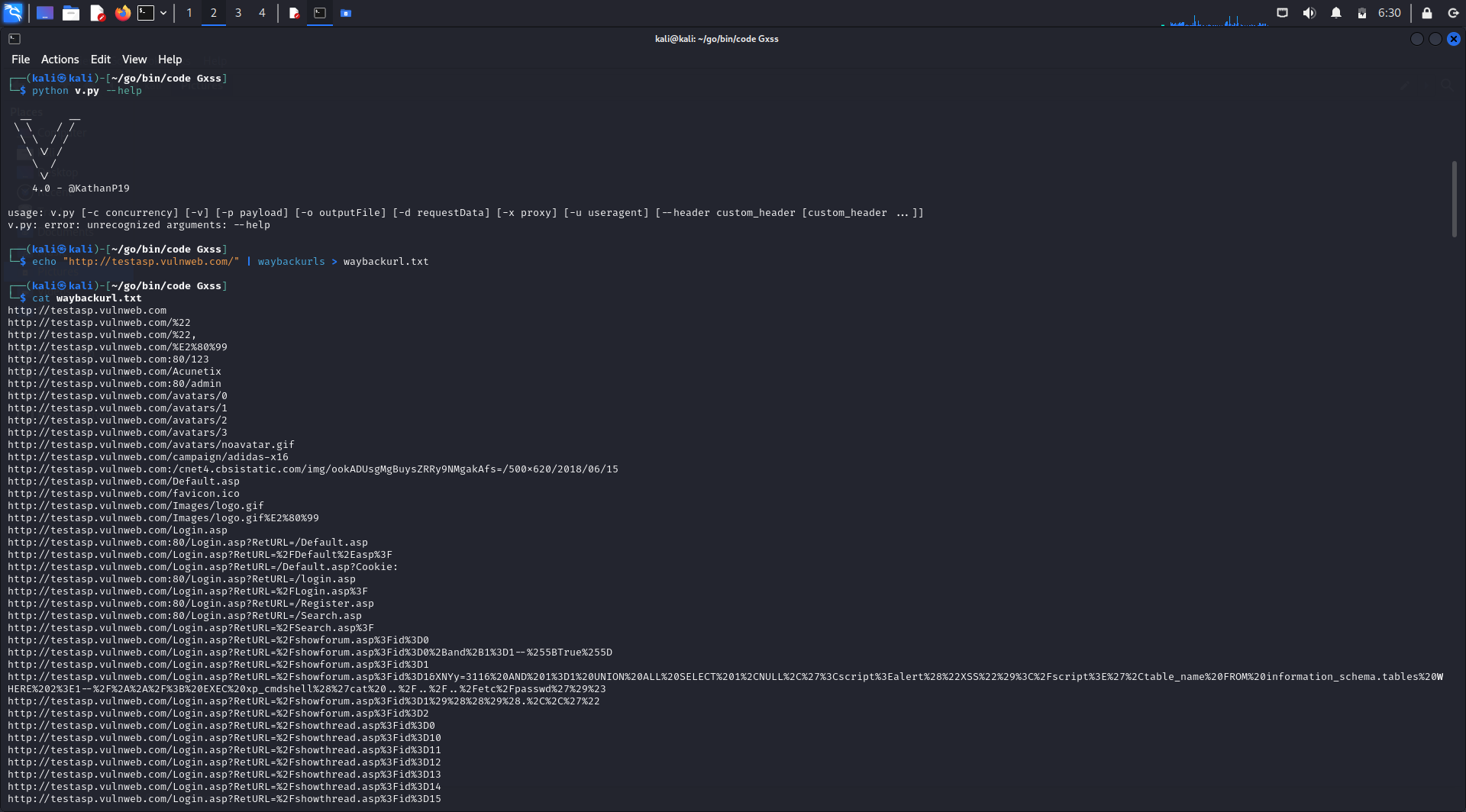
* Once you have identified endpoints of interest, you can perform further investigation using security testing tools, manual testing techniques, or by consulting additional resources.



**Working of “V” Tool :-**

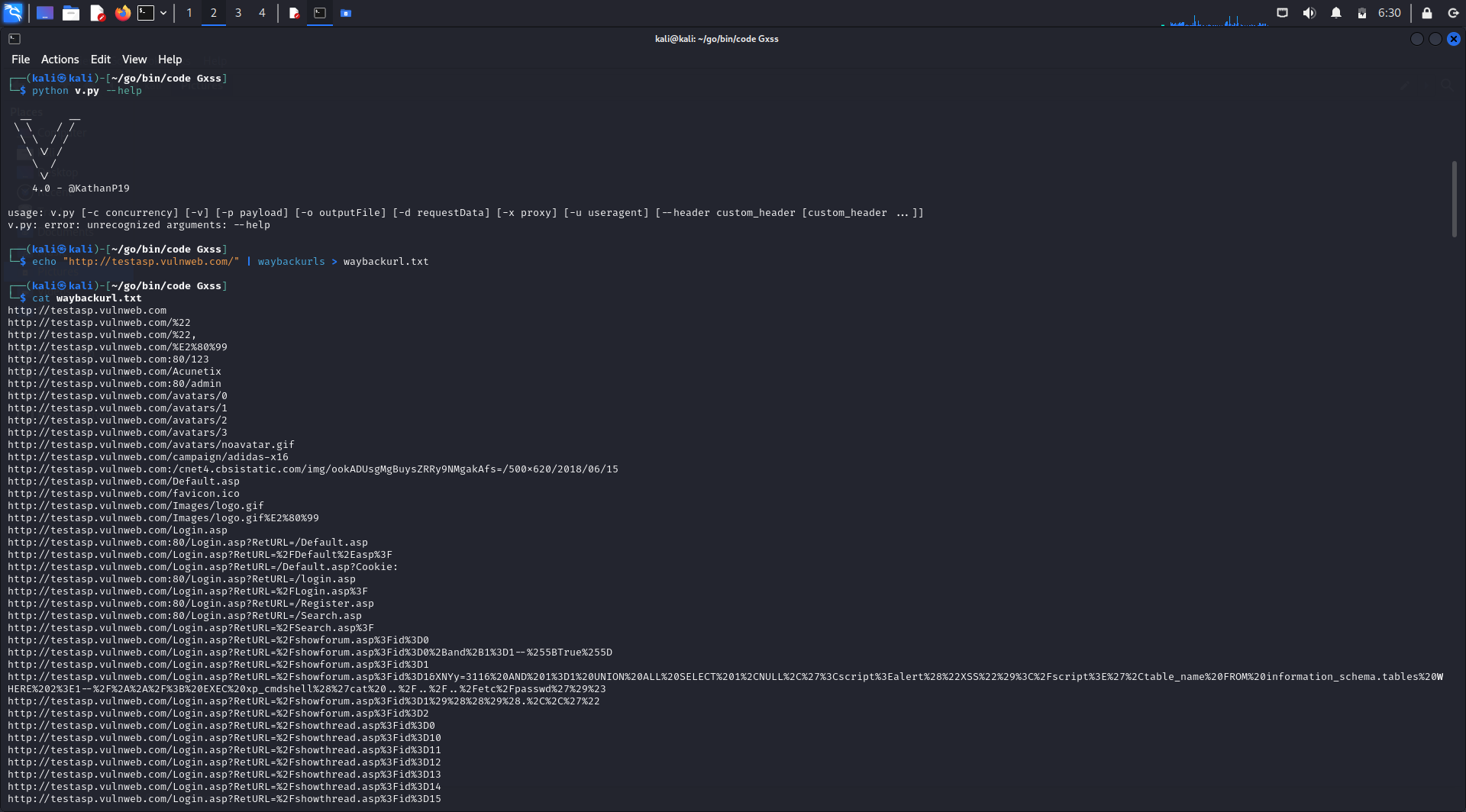
**Installation:**

* Ensure you have Python installed on your system.
* Install the V tool and its dependencies, including WaybackURL, by following the installation instructions provided in the tool's documentation.



**Collect URLs:**

* Identify the target website for which you want to find XSS vulnerabilities.
* Use the WaybackURL tool to collect archived URLs related to the target website. Run the command waybackurls example.com to fetch archived URLs for <https://testasp.vulnweb.com/>.

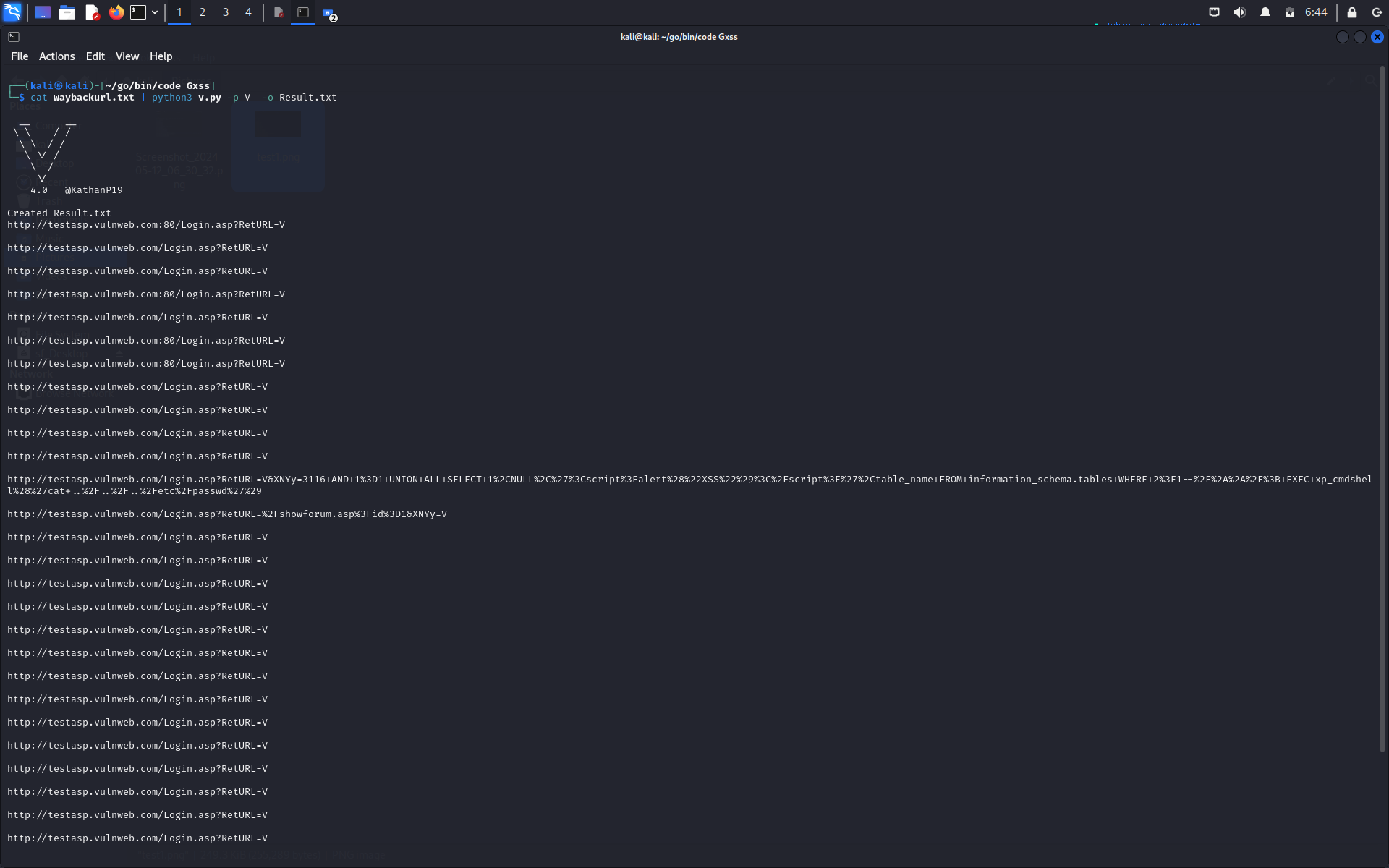


**Filter Endpoints:**

* Review the list of URLs obtained from WaybackURL.
* Filter the URLs to identify potential endpoints that could be vulnerable to XSS attacks. Look for URLs containing common endpoints like /login, /register, /admin, or any custom endpoint that handle user input

**Testing for XSS:**

* Use the V tool to perform XSS testing on the identified endpoints. For example, you can run the command v scan Waybackurl.txt to scan the login page, search page for XSS vulnerabilities.
* Alternatively, you can automate the testing process by iterating through the list of endpoints and running XSS tests on each endpoint programmatically using the V tool.

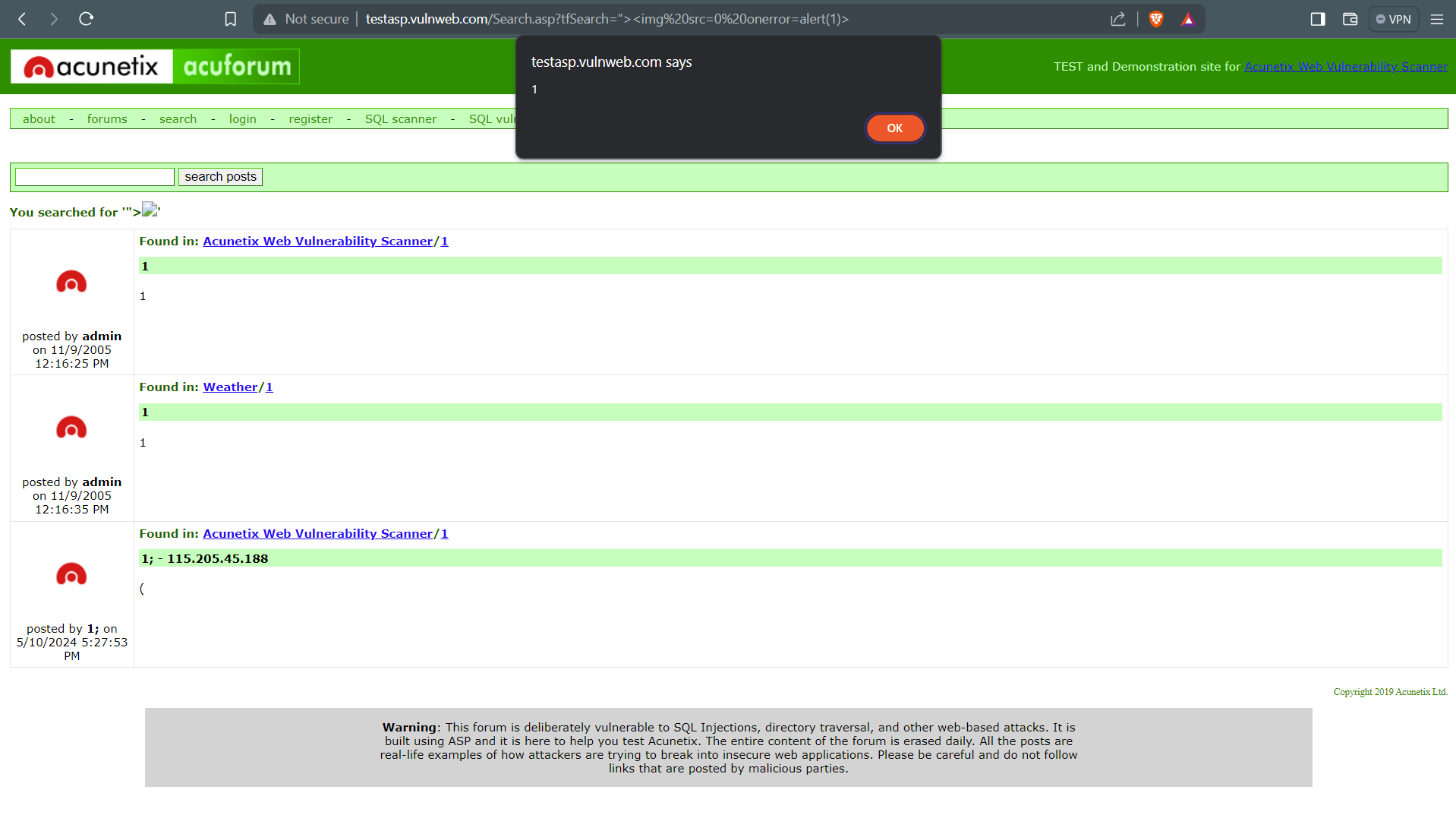


**Analyzing Results:**

* After running the XSS tests, analyze the results generated by the V tool. It will indicate whether each endpoint is vulnerable to XSS attacks or not.
* Pay attention to the endpoints flagged as vulnerable and prioritize them for further investigation and remediation

**Exploitation and Verification:**

* If XSS vulnerabilities are detected, attempt to exploit them to confirm their severity and impact.
* Check it manually , copy an endpoint where the tool detected that it is venerable and copy on the browser and make a payload for it that it prove that it is venerable endpoint and we easly inject the payload on it.
* Verify the vulnerabilities by injecting malicious payloads into input fields or URL parameters and observing the resulting behavior on the website.

****

**Reporting and Remediation:**

* Document your findings, including the vulnerable endpoints, the nature of the XSS vulnerabilities, and their potential impact.
* Report the vulnerabilities to the website owner or responsible parties, along with recommendations for remediation.
* Monitor the website for security updates and patches to address the identified XSS vulnerabilities.

**Conclusions / Recommendations**

**Conclusions:-**

The V tool provides a comprehensive solution for detecting XSS vulnerabilities in web applications, leveraging the power of automated scanning and analysis. Through its integration with tools like WaybackURL, V enables efficient identification of potential endpoints that may be susceptible to XSS attacks. By collecting archived URLs and systematically testing each endpoint, V streamlines the process of vulnerability discovery, allowing security professionals to prioritize and address critical issues effectively.

With its intuitive command-line interface and customizable options, V offers flexibility in tailoring the scanning process to suit specific requirements and target environments. Its ability to automate XSS testing across multiple endpoints simplifies the task of security assessment, reducing manual effort and accelerating the identification of vulnerabilities.

The conclusion drawn from using the V tool is that it serves as a valuable asset in the arsenal of security professionals, enabling them to enhance the resilience of web applications against XSS threats. By proactively identifying and addressing vulnerabilities, organizations can mitigate the risk of data breaches and unauthorized access, safeguarding sensitive information and preserving the trust of their users.

Overall, V emerges as a reliable and efficient tool for XSS detection, contributing to the improvement of web application security practices and the overall resilience of digital ecosystems.

**References**

1. <https://ieeexplore.ieee.org/abstract/document/7905285/figures#figures>
2. <https://academic.oup.com/comjnl/article-abstract/58/4/808/335989>
3. <https://ieeexplore.ieee.org/abstract/document/9369796>
4. <https://www.researchgate.net/profile/Shashank-Gupta-34/publication/299040084_Exploitation_of_Cross-Site_Scripting_XSS_Vulnerability_on_Real_World_Web_Applications_and_its_Defense/links/56ee6cf808ae59dd41c6a329/Exploitation-of-Cross-Site-Scripting-XSS-Vulnerability-on-Real-World-Web-Applications-and-its-Defense.pdf>