RESEARCH ARTICLE OPEN ACCESS

Security Tester & Automatic Clickjacking Tool

Harshil Patel[1], Preet Prajapati[2], Ashish S Patel[3], Kusum Lata Dhiman[4]
1(Dept. CSE - Cyber security, Parul University, India
Email: 200305126025@paruluniversity.ac.in)
2(Dept. CSE - Cyber security, Parul University, India
Email: 200305126027@paruluniversity.ac.in)
3(Dept. Computer Science & Engineering, Parul University, India
Email: ashish.patel28275@paruluniversity.ac.in)
4(Dept. Computer Science & Engineering, Parul University, India
Email: kusumlata.dhiman21133@paruluniversity.ac.in)

Abstract:

This A lot of effort has been put into researching client side to web server attacks, including vulnerabilities like cross site scripting, cross site request forgery, and more recently, clickjacking.

Similar to other client side attacks, a clickjacking vulnerability can use the browser to exploit weaknesses in cross domain isolation and the same origin policy. It does this by tricking the user to click on something that is actually not what the user perceives they are clicking on. In the most extreme cases, this vulnerability can cause an unsuspecting user to have their account compromised with a single click.

Additionally, although the possibility for an attacker to frame a page is easy to detect, it is much more difficult to demonstrate or assess the impact of a clickjacking vulnerability than more traditional client side vectors.

I. INTRODUCTION

- **Clickjacking**: - Clickjacking happens when a user intends to click on something, but through an invisible or opaque frame the user actually clicks on something else. A clickjacked page is a visible webpage that includes a reference to another, invisible framed webpage. The user clicks on a link on the invisible framed webpage that performs some unintended action while the user thinks they just clicked on the visible page. An attacker only needs some of the basic features of HTML, CSS, and possibly JavaScript to craft an attack.

- Figure is based on a figure shows how a clickjacking attack occurs. The user visits an online lottery sweepstakes website to play the lottery and try to win the grand prize. The user thinks they are clicking on the link to play the lottery sweepstakes, but in fact they are clicking on the invisible amazon framed page.

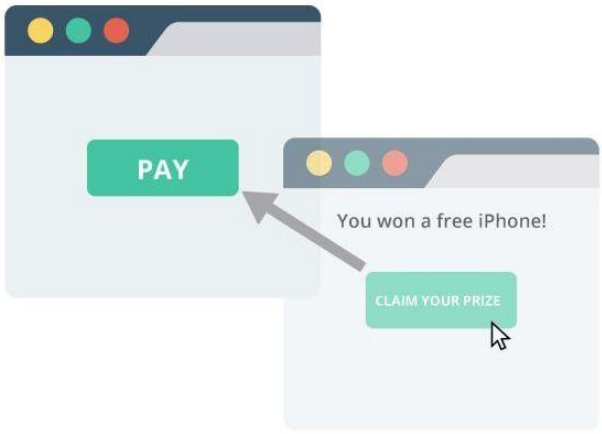

Fig.1

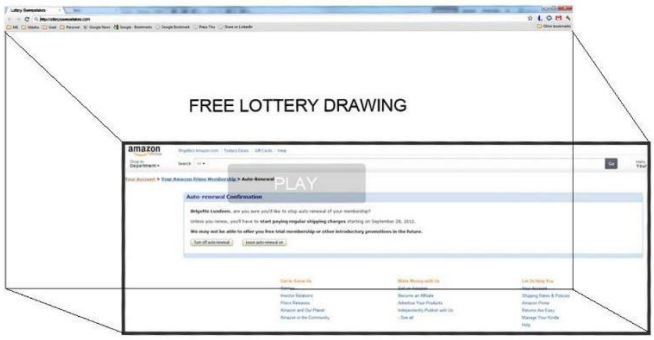

Fig.2

- Automatic Clickjacking Tester used for find a vulnerability of website, it is finding a clickjacking bug from website.
- If any type of vulnerable bug finds on website, then it is generated a alert message and create one file and save that website on selected storage path.

 # Front-end Technology:

- HTML: Markup language for creating the structure of web pages.
- CSS: Styling language used to design the appearance and layout of web pages.

 # Back-end Technology:

- Programming languages: Python
- Web frameworks: Django (Python), Flask (Python)

II. PROBLEM STATEMENT

- **Time-consuming**: Manual clickjacking tools typically require human intervention, which can be time-consuming. The process involves manually inspecting web pages, identifying potential clickjacking vulnerabilities, and devising strategies to exploit them.

- **Limited Coverage**: Manual tools may not have the capability to comprehensively scan an entire website or web application for clickjacking vulnerabilities. Human testers may miss certain vulnerabilities due to oversight or lack of expertise in identifying complex clickjacking scenarios.
- **Subject to Human Error**: Human testers are prone to errors, which can result in overlooking critical clickjacking vulnerabilities or misinterpreting the severity of identified vulnerabilities. Manual testing may lack the precision and consistency of automated tools.

- **Resource Intensive**: Manual clickjacking testing requires skilled security professionals who possess expertise in identifying and exploiting vulnerabilities. Hiring and retaining such professionals can be costly for organizations, especially for ongoing security assessments.

- **Scalability**: Manual testing may not be scalable, especially for large-scale web applications or websites with frequent updates. As the complexity and size of the target increase, manual testing becomes increasingly impractical.

- **Inconsistent Results**: Manual testing may yield inconsistent results across different testers due to variations in skill levels, experience, and testing methodologies. Automated tools, on the other hand, provide consistent and reproducible results.

III. LITERATURE REVIEW

Here is a literature review focusing on clickjacking tester tools and techniques used to identify website vulnerabilities:

- **"Practical Clickjacking Attacks in Modern Web Applications"** by J. Li, M. Mitchell, and M. Zhang:
  - This paper discusses various clickjacking techniques and presents a comprehensive analysis of clickjacking vulnerabilities in modern web applications. It highlights the importance of effective clickjacking testing tools for

detecting and mitigating such vulnerabilities.

- **"Automated Detection of Clickjacking Attacks with DOM Correlation and Classification"** by X. Yuan, X. Wang, and X. Han:
  - The authors propose an automated detection approach for clickjacking attacks using DOM (Document Object Model) correlation and classification techniques. They discuss the design and implementation of a clickjacking tester tool that leverages machine learning algorithms to identify and classify clickjacking attempts.
- **"A Survey of Web Application Security Testing Tools"** by N. Chandra, R. Amirtharajan, and P. Kuppusamy:
  - The authors conduct a survey of web application security testing tools, including both commercial and open-source solutions. They analyze the features, strengths, and limitations of existing clickjacking testing tools and provide recommendations for selecting appropriate tools based on specific testing requirements and objectives.
- **"Clickjacking Defense Mechanisms: A Survey and Taxonomy"** by A. K. Gupta, S. Singh, and M. K. Sharma:
  - This paper presents a survey and taxonomy of clickjacking defense mechanisms, including client-side and server-side techniques for mitigating clickjacking attacks. It discusses the role of clickjacking testing tools in evaluating the effectiveness of defense mechanisms and identifies areas for future research and development in clickjacking prevention and detection.

IV. PROJECT MODULE

For a project module aimed at developing a clickjacking tester tool to determine if a website is vulnerable to clickjacking attacks, several components and techniques can be involved, here's a breakdown of some key modules that might be used in such a tool:

- **HTTP Request and Response Handling**: This module deals with sending HTTP requests to the target website and analyzing the responses. It's crucial for simulating user interactions with the website and identifying potential clickjacking vulnerabilities in the web page structure.

- **HTML Parsing and DOM Manipulation**: This module parses HTML documents retrieved from the target website and manipulates the Document Object Model (DOM) to analyze the structure of the web page. It's responsible for identifying elements susceptible to clickjacking, such as iframes and buttons.

- **Clickjacking Detection Algorithms**: This module implements algorithms and heuristics to detect potential clickjacking vulnerabilities based on the analysis of the HTML structure and DOM of the target web page. Techniques such as checking for overlapping elements, iframe sandboxing attributes, and X-Frame-Options headers may be employed.

- **Rendering Engine Integration**: Some clickjacking tester tools might integrate with web rendering engines or headless browsers to render and interact with web pages in a realistic manner. This enables the tool to detect clickjacking vulnerabilities that might only manifest during the rendering process.

- **User Interface (UI)**: A user interface module allows users to interact with the tool, input URLs of target websites, view scan results, and configure scan settings. A user-friendly UI enhances the usability of the tool

and makes it accessible to security professionals and developers.

- **Reporting and Logging**: This module generates detailed reports summarizing the results of the clickjacking vulnerability scan. It logs any identified vulnerabilities, including their severity level, affected web pages, and potential impact.

- **Configuration and Customization**: Users may need the ability to configure various aspects of the scanning process, such as specifying custom HTTP headers, setting scan parameters, and defining exclusion criteria for certain URLs or page elements.

- **Integration with Security Testing Frameworks**: For comprehensive security testing, the clickjacking tester tool may integrate with broader security testing frameworks or vulnerability scanners. This allows users to conduct holistic security assessments that cover multiple types of vulnerabilities beyond clickjacking

V. HARDWARE AND SOFTWARE REQ.

- **Hardware Requirements:**

1) OS Window XP and later/Mac OS
2) RAM 8GB
3) Processor Intel's dual-core Core i5
4) Hard drive 1GB of free space

- **Software Requirement:**

1. Windows Prompt commands
2. Python 3.12
3. Visual Studio Code

VI. EXCEPTED OUTCOME OF THE PROJECT

- **1dentifiIcation of Vulnerable Pages**: The clickjacking testing tool should identify specific web pages or components within the website that are susceptible to clickjacking attacks. This could include pages where sensitive actions can be initiated through deceptive clicks.

- **Description of Vulnerabilities**: The tool provides a description of the clickjacking vulnerabilities found on the website. This description may include information about the vulnerability could be available or not.

- **Testing Confirmation**: The tool may include features to confirm the presence of clickjacking vulnerabilities through proof-of-concept demonstrations. This helps validate the identified of vulnerabilities.

VII. LIMITATION

- Performing a clickjacking attack requires the attacker to know the exact location of what they want clicked. This can be difficult to calculate with some webpages depending on how the page is setup. However, in most cases this is easy enough to do.
- Limited Coverage: Clickjacking tester tools may only scan certain types of web elements or specific areas of a webpage for vulnerabilities. They might miss vulnerabilities in less common or dynamically generated content, such as iframes or JavaScript-based elements.

VIII. CONCLUSION

- Clickjacking is such a different type of attack that it might be best served as its own page within this tester. All tester modules follow a strict standard and have certain functionality restrictions.
- The entire attack could be setup on its own page, similar to Stone's cjtool, and other modules could be used within the Clickjacking context.

IX. FUTURE WORK

- There are protections tools available for clickjacking, the web applications implementing these mitigations are far and in between.

- Report Generation: The tool should generate a comprehensive report summarizing the findings of the clickjacking assessment. The report should include details about the vulnerabilities discovered, their severity levels, and recommended remediation steps. This report can be used by website administrators and developers to address the identified issues.

X. SECURITY TOOL

Several existing tools are available for clickjacking detection and prevention, ranging from browser extensions to specialized security scanners. Here are some examples:

- **Burp Suite**: Burp Suite is a popular web vulnerability scanner used by security professionals. It includes features for detecting clickjacking vulnerabilities among other security issues.

- **OWASP ZAP (Zed Attack Proxy)**: OWASP ZAP is an open-source web application security scanner. It can be used to identify clickjacking vulnerabilities and provides features for automated testing and reporting.

- **Netsparker**: Netsparker is a web application security scanner that helps identify vulnerabilities including clickjacking. It offers both automated and manual testing capabilities.

- **Detectify**: Detectify is a web vulnerability scanner that includes clickjacking detection among its features. It offers continuous monitoring and automatic testing for web applications.

- **SecurityHeaders.io**: SecurityHeaders.io is a free online service that checks a website's HTTP response headers for security vulnerabilities, including clickjacking protection headers like X-Frame-Options and Content-Security-Policy.

- **No Script Browser Extension**: No Script is a browser extension available for Firefox that blocks JavaScript and other active content from running on websites. It can help prevent clickjacking attacks by blocking malicious scripts.

- **HTTPCS Security**: HTTPCS Security is a web security solution that offers clickjacking detection and protection among other features. It provides automated vulnerability scanning and monitoring for web applications.

- **Content-Security-Policy (CSP) Analyzer**: Various CSP analyzers are available online to help developers and security professionals analyze and validate their Content Security Policy configurations, which can help prevent clickjacking attacks among other threats

XI. REFERENCES

- *G. Rydstedt, E. Bursztein, D. Boneh, C. Jackson, "Busting Frame Busting: a Study of Clickjacking Vulnerabilities on Popular Sites", June 17, 2010.*

- *SDL Process Guidance Version 5.1, Microsoft, April 14, 2011.*

- *W3C Working Draft, section 4.8.2, March 29, 2012, http://www.w3.org/TR/html5/the-iframeelement.html#attr-iframe-sandbox.*

- *StatCounter Global Stats: Top 12 Browser Versions on,Feb,2012,http://gs.statcounter.com/#bro*

wser_version- ww-monthly-201202-201202-bar.

- *P.Stone,ClickjackingTool,http://www.contextis.com/research/tools/clickjackingtool/ Blackhat 2010.*
- *"BeEF: The Browser Exploitation Framework Project", http://beefproject.com/.*
- *K.Freitas,"MeasureIt",https://addons.mozilla.org/en- US/firefox/addon/measureit/.*
- *https://github.com/beefproject/beef/pull/744*