**Web App Pen testing**

1. **Introduction**
   1. **Overview**

Web Application Penetration Testing of Cubelelo.com

* 1. **Purpose**

The purpose of this project report is to provide an overview of the web application penetration testing conducted on the website Cubelelo.com. Cubelelo.com is an e-commerce platform that specializes in selling Rubik's cubes and other related puzzles. The goal of the penetration testing was to identify potential vulnerabilities and security weaknesses within the website, with the aim of enhancing its overall security posture.

1. **Literature Survey**
   1. **Existing problem**

Cross-Site Request Forgery (CSRF): CSRF is a prevalent vulnerability in web applications that allows attackers to execute unauthorized actions on behalf of authenticated users. This occurs when a malicious website tricks a user's browser into making a request to another website where the user is authenticated, leading to unwanted actions being performed without the user's knowledge or consent. CSRF attacks can result in various detrimental consequences, such as changing passwords, making fraudulent transactions, or modifying user settings.

* 1. **Proposed Solution**

1. CSRF Tokens: Implementing CSRF tokens is a widely adopted solution to mitigate CSRF vulnerabilities. These tokens are unique values assigned to user sessions and embedded within web forms or HTTP requests. Upon submission, the server verifies the token's validity, ensuring that the request originates from the expected user session and not from a malicious source. CSRF tokens effectively protect against CSRF attacks by adding an additional layer of authentication and authorization.
2. SameSite Cookies: Configuring cookies with the "SameSite" attribute can help prevent CSRF attacks. By setting the SameSite attribute to "Strict" or "Lax," web applications can restrict cookie access to same-origin requests, significantly reducing the risk of CSRF attacks. This prevents cookies from being automatically sent by the browser in cross-origin requests, thereby protecting against unauthorized actions.
3. Input Validation and Output Encoding: Implementing strong input validation and output encoding practices is essential for web application security. Properly validating and sanitizing user inputs and encoding output data mitigate the risk of various attacks, including CSRF. By ensuring that user-supplied data is properly filtered and encoded, web applications can prevent the execution of malicious code and safeguard against CSRF vulnerabilities.
4. Security Testing with Burp Suite: Burp Suite, a popular web application security testing tool, plays a crucial role in identifying and mitigating CSRF vulnerabilities. Its features, such as the active scanner and proxy capabilities, allow security professionals to actively test and analyze web applications for CSRF vulnerabilities. By simulating real-world attack scenarios and providing detailed reports, Burp Suite helps developers and administrators identify and address CSRF vulnerabilities effectively.
5. **Theoritical Analysis**
   1. **Block Diagram**

**+------------------------------------------------+**

**| Web Application Penetration Testing |**

**+------------------------------------------------+**

**|**

**|**

**+-------------------------------+**

**| Cubelelo.com |**

**| (Target Website) |**

**+-------------------------------+**

**|**

**|**

**+---------------------------------------------------+**

**| Methodology |**

**+---------------------------------------------------+**

**| |**

**| |**

**+---------------------------------+ +-----------------------------------+**

**| Automated Tools | | Manual Testing |**

**+---------------------------------+ +-----------------------------------+**

**| |**

**| |**

**+---------------------------------+ +-----------------------------------+**

**| Vulnerability Scanning | | Exploitation |**

**+---------------------------------+ +-----------------------------------+**

**| |**

**| |**

**+---------------------------------+ +-----------------------------------+**

**| Vulnerability Identification | | Impact Assessment and Reporting |**

**+---------------------------------+ +-----------------------------------+**

**| |**

**| |**

**+---------------------------------+ +-----------------------------------+**

**| Findings and Recommendations | |**

**+---------------------------------+ +-----------------------------------+**

**|**

**|**

**+------------------------+**

**| Final Report |**

**+------------------------+**

* 1. **Hardware/ Software Desigining**

1. **Experimental Investigation**
2. **Flowchart**

**+---------------------------------------------------+**

**| CSRF Mitigation Flow |**

**+---------------------------------------------------+**

**|**

**|**

**+-----------------------------+**

**| User Authentication |**

**+-----------------------------+**

**|**

**|**

**+----------------------+**

**| Form Submission |**

**+----------------------+**

**|**

**|**

**+----------------------+**

**| Data Modification |**

**+----------------------+**

**|**

**|**

**+----------------------+**

**| Transactional Operations |**

**+----------------------+**

**|**

**|**

**+----------------------+**

**| Session Management |**

**+----------------------+**

**|**

**|**

**+----------------------+**

**| API Endpoints |**

**+----------------------+**

1. **Result**
2. **Advantages and Disadvantages**

Advantages of the Proposed Solution:

1. Mitigation of CSRF Attacks: Implementing CSRF tokens, configuring SameSite cookies, and adopting input validation and output encoding practices effectively mitigate the risk of CSRF attacks. These measures provide an additional layer of authentication and authorization, ensuring that unauthorized actions cannot be performed on behalf of authenticated users.
2. Enhanced Security Posture: By implementing the proposed solutions, web applications, including Cubelelo.com, can significantly enhance their overall security posture. The use of CSRF tokens, SameSite cookies, and robust input validation practices reduces the risk of unauthorized access, data tampering, and other malicious activities, safeguarding user data and system integrity.
3. Industry Best Practices: The proposed solutions align with industry best practices for web application security. CSRF tokens, SameSite cookies, and input validation and output encoding practices are widely recognized and recommended by security experts, ensuring that the implemented measures are in line with established security standards.
4. Effective Use of Burp Suite: Utilizing Burp Suite as a security testing tool provides several advantages. Its active scanner and proxy capabilities enable comprehensive testing and analysis, facilitating the identification and mitigation of CSRF vulnerabilities. Burp Suite's detailed reports assist in understanding and addressing the identified issues effectively.

Disadvantages of the Proposed Solution:

1. Implementation Complexity: Depending on the complexity of the web application, implementing CSRF tokens, configuring SameSite cookies, and ensuring proper input validation and output encoding can require significant development effort. Proper implementation and integration with existing systems may pose challenges, particularly in larger and more complex applications.
2. Compatibility Issues: Implementing SameSite cookies and other security measures may introduce compatibility issues with older web browsers or legacy systems. It is essential to ensure that the proposed solutions are compatible with the target web application's supported browsers and platforms to avoid any negative impact on user experience.
3. Maintenance and Updates: The proposed solutions require ongoing maintenance and updates to remain effective. Regular monitoring and review are necessary to address any changes in the threat landscape and emerging vulnerabilities. Failure to keep up with updates and patches may render the implemented measures less effective over time.
4. False Positives and Negatives: While the proposed solutions mitigate CSRF vulnerabilities, there is still a possibility of false positives and negatives during the implementation and testing phase. False positives may occur when legitimate requests are incorrectly flagged as potential CSRF attacks, causing inconvenience to users. False negatives may occur when some CSRF vulnerabilities remain undetected, leaving potential attack vectors exposed.
5. **Application**

The proposed application of these solutions for mitigating CSRF vulnerabilities can apply to various areas within a web application. Here are some key areas where these solutions can be implemented:

1. User Authentication: Implementing CSRF tokens and ensuring proper input validation and output encoding during the authentication process can prevent unauthorized actions on authenticated user accounts. It helps ensure that the authentication process remains secure and protected against CSRF attacks.

2. Form Submissions: CSRF tokens can be embedded within web forms to verify the authenticity of form submissions. This prevents malicious websites from tricking users into submitting forms that perform unintended actions on the target website.

3. Data Modification: CSRF tokens and input validation play a crucial role in securing data modification functionalities such as updating user profiles, changing passwords, or performing any action that modifies user data. By incorporating these solutions, the risk of unauthorized data modifications through CSRF attacks is significantly reduced.

4. Transactional Operations: Web applications that involve financial transactions, such as e-commerce platforms, can benefit from the proposed solutions. Implementing CSRF tokens and input validation ensures the integrity of these transactions and protects against unauthorized actions, such as making fraudulent transactions or modifying payment details.

5. Session Management: Configuring SameSite cookies and employing secure session management practices can prevent CSRF attacks that target user sessions. By enforcing strict cookie access controls and implementing session timeouts, the risk of session-based CSRF attacks can be mitigated.

6. API Endpoints: If the web application exposes API endpoints, it is crucial to apply CSRF mitigation measures to secure these endpoints as well. CSRF tokens and input validation can be implemented to prevent unauthorized API requests and protect sensitive data accessed through APIs.

It's important to note that the specific areas where these solutions apply may vary depending on the structure and functionalities of the web application. Conducting a thorough analysis of the application's architecture and identifying potential CSRF attack vectors will help determine the appropriate areas to implement the proposed solutions effectively.

1. **Conclusion**

In conclusion, the web application penetration testing of Cubelelo.com revealed several vulnerabilities and security weaknesses. The findings and recommendations provided in this project report aim to assist the website administrators and developers in improving the overall security of Cubelelo.com. By addressing the identified vulnerabilities and implementing the recommended measures, Cubelelo.com can enhance its resilience against potential attacks and safeguard the confidentiality, integrity, and availability of its users' information. We successfully performed a logout-CSRF attack on a victim browser. Patching this can be done by adding authentication CSRF Tokens in logout request. Please refer to the video for the complete procedure.

1. **Future scope**

The future scope of enhancing CSRF mitigation in web applications includes the following possibilities:

1. Multi-Factor Authentication (MFA): Implementing MFA can provide an additional layer of security during user authentication. By combining something the user knows (password) with something the user possesses (such as a one-time password from a mobile app or a hardware token), the risk of unauthorized access, including CSRF attacks, can be further reduced.
2. Content Security Policy (CSP): Integrating CSP headers into the web application's response can help mitigate the risk of various attacks, including CSRF. CSP allows web administrators to define a policy that restricts the types of content that can be loaded on a web page, thus reducing the likelihood of executing malicious scripts originating from unauthorized sources.
3. Subresource Integrity (SRI): Implementing SRI involves adding integrity checks to external resources, such as JavaScript libraries or stylesheets, referenced in a web page. This ensures that the loaded resources have not been tampered with or modified, further reducing the risk of CSRF attacks that exploit compromised or malicious external resources.
4. Security Education and Awareness: Promoting security education and awareness among developers, administrators, and end-users is essential for maintaining a robust CSRF mitigation strategy. Regular training programs and workshops can help educate stakeholders about the risks associated with CSRF attacks and provide best practices for secure development and usage of web applications.
5. **Bibliography**