ALEX MAGANA

BSCLMR183021

CSC 4244: SECURITY IN APPLICATIONS

THREAT ASSESSMENT AND ATTACK SURFACE ANALYSIS

October 16, 2023

1. **Threat Analysis :**

Using example of PayPal:

PayPal is a widely-used online payment platform that allows individuals and businesses to make transactions, send and receive money securely over the internet.(*2013-003: XSS Vulnerability in LinkedIn | Internet Security Auditors*, n.d.)

1. Cross-Site Scripting (XSS) Attack:
Description:
XSS is a type of injection attack where malicious scripts are injected into webpages viewed by users. Attackers can exploit vulnerabilities in the application to inject scripts, which are then executed in the context of the user's browser. Attackers could inject malicious scripts into PayPal's webpages, which, when viewed by users, execute within their browsers. This script could steal sensitive user information or manipulate the appearance of the PayPal site to conduct phishing attacks.
Potential Impact on PayPal:

- Data Theft: Attacker can steal PayPal login credentials, credit card information, and personal details.
- Phishing: Manipulated PayPal pages might deceive users into providing sensitive information, thinking it's a legitimate PayPal site.

Attack Vector:
Attackers might send phishing emails with links to a fake PayPal login page containing the injected script. Unsuspecting users clicking these links could have their credentials stolen.

2**. SQL Injection Attack on PayPal:**
Description:
SQL injection is a code injection technique where attackers insert malicious SQL statements into input fields of an application's database query. If the application does not properly validate or sanitize user input, attackers can manipulate the SQL query and gain unauthorized access to the database. Attackers exploit vulnerable input fields on PayPal’s website to inject malicious SQL code, potentially gaining unauthorized access to PayPal’s database.
Potential Impact on PayPal:

- Unauthorized Database Access: Attackers could access, modify, or delete user accounts, transaction records, or other sensitive data in PayPal’s database.
- Financial Loss: Attackers could manipulate transaction data, leading to unauthorized fund transfers or account manipulations.

Attack Vector:
Attackers could exploit poorly sanitized input fields, such as the search bar on PayPal's website, by injecting SQL code to manipulate the database queries.

3. Man-in-the-Middle (MitM) Attack on PayPal:
Description: In a MitM attack, an attacker intercepts communication between two parties, often without their knowledge, and can eavesdrop or modify the data exchanged between

them. This can be done through techniques like session hijacking or DNS spoofing. Attackers intercept communication between users and PayPal's servers, allowing them to eavesdrop on or modify the data being exchanged.
Potential Impact on PayPal:

- Data Interception: Attackers can intercept login credentials, credit card details, and transaction information.
- Transaction Tampering: Attackers could modify transaction data, leading to unauthorized fund transfers or changes in account balances.

Attack Vector:
Attackers could exploit unsecured Wi-Fi networks at public places or compromise DNS servers to redirect PayPal users to malicious sites, allowing them to intercept and manipulate data packets.(*Cyber Threat Analysis*, n.d.)

2. Common Attack Vectors

Cross-Site Scripting (XSS) Attack Vectors:

a. <u>Stored XSS Attack:</u>

Stored XSS occurs when an attacker injects a malicious script into a website's database, which is then served to users when they visit a specific page. For instance, if a user posts a message containing a malicious script on a forum, and other users view that message, the script executes in their browsers, potentially stealing session cookies, login credentials, or other sensitive information. Consequences include identity theft, account hijacking, or phishing attacks.

Example: In 2013, a Stored XSS vulnerability in LinkedIn social networking site allowed attackers to inject malicious scripts into user profiles. When other users viewed these profiles, their browsers executed the scripts, leading to compromised accounts and widespread phishing attempts.

2. <u>Reflected XSS Attack:</u>

Reflected XSS occurs when a malicious script is embedded in a URL and only executed when the user clicks on the manipulated link. Attackers often trick users into clicking these links through phishing emails or malicious advertisements. When the victim clicks the link, the script executes in their browser, potentially stealing sensitive data entered on the manipulated site, such as login credentials.

Example: Cybercriminals might create a phishing email with a link that appears to lead to a legitimate website, but actually contains a crafted URL with a reflected XSS payload. When the recipient clicks the link, the script executes, stealing their login credentials.

3. <u>DOM-Based XSS Attack:</u>

DOM-Based XSS occurs when the client-side script on a web page manipulates the Document Object Model (DOM) in an insecure way. Attackers modify the DOM environment using malicious scripts, which are then executed in users' browsers. This can lead to actions such as redirecting users to malicious websites or stealing sensitive data without the need for server-side vulnerabilities.

Example: Consider a banking website where the client-side script displays account balances without proper validation. An attacker could inject a script altering the displayed balance in the user's browser. While the actual account balance remains unchanged on the server, the user sees a manipulated balance, potentially causing confusion or panic.(*What Are Attack Vectors*, n.d.)

3. Attack Surface Analysis

a. User-Facing Entry Points:

- Login Page: Attackers can attempt to brute force login credentials or exploit vulnerabilities like XSS to steal user credentials.
- Payment Forms: Input fields for payments are potential targets for injection attacks like SQL Injection or XSS to manipulate transactions or gain unauthorized access.
- Account Settings: Attackers might try to exploit weaknesses in security questions or email change procedures to take control of user accounts.
- Forgotten Password: Attackers can exploit the password recovery process, possibly using social engineering techniques to gain unauthorized access.

b. Backend Components Entry Points:

- API Endpoints: Backend APIs are vulnerable to various attacks like API injection or parameter manipulation, allowing attackers to gain unauthorized access to sensitive data or perform actions on behalf of users.
- Database: Exploitable through SQL Injection attacks if input validation and parameterized queries are not properly implemented. Attackers can manipulate, extract, or delete sensitive information from the database.
- Server Components: Vulnerabilities in server software can lead to remote code execution, allowing attackers to compromise the server and gain control over the application.

c. Exploitation Scenarios:

- Phishing Attacks: Attackers can exploit user-facing entry points to create convincing phishing pages that trick users into entering sensitive information.
- Data Manipulation: By exploiting backend components, attackers can manipulate user data, transactions, or account balances, leading to financial losses or fraudulent activities.
- Identity Theft: Stolen user credentials can be used for identity theft, leading to a range of malicious activities.
- Session Hijacking: If session management is weak, attackers can hijack user sessions, allowing them to perform actions on behalf of the victim without their knowledge.
- Malware and Client-Side Attacks: Users interacting with the application through compromised devices might unknowingly transmit sensitive data to attackers.(*What Is an Attack Surface?*, n.d.)

4. Recommendations and Mitigation

To mitigate the identified Cross-Site Scripting (XSS) threats and its common attack vectors, the application's developers and administrators should implement the following security measures and best practices:

1. Input Validation and Sanitization:

Implement strict input validation by validating all user-supplied data on the client and server sides. Ensure that inputs conform to expected formats and reject any data that does not meet these criteria. Employ input sanitization techniques to neutralize malicious code, removing or encoding special characters to prevent script execution. By validating and sanitizing user inputs, developers can significantly reduce the risk of XSS vulnerabilities, as attackers won't be able to inject malicious scripts through input fields or URLs.

2. Content Security Policy (CSP):

Enforce a Content Security Policy, which specifies from where resources can be loaded. CSP mitigates XSS attacks by restricting the sources from which the application can load content, such as scripts, styles, or images. By defining trusted sources and blocking inline scripts, developers can prevent attackers from executing malicious scripts even if an XSS vulnerability exists. Properly configured CSP headers can provide an additional layer of security, reducing the attack surface and protecting against various XSS vectors.

3. Output Encoding:

Implement proper output encoding techniques when displaying user-generated content or dynamic data. Encode special characters such as <, >, &, and quotes to their respective HTML entities. Output encoding ensures that any data rendered in the browser is treated as plain text and not executable code. By encoding output, developers prevent browsers from interpreting user-generated content as scripts, thus safeguarding against XSS attacks, including both stored and reflected variants.

4. Use HttpOnly and Secure Flags for Cookies:

For session management, set the HttpOnly flag on cookies to prevent them from being accessed via JavaScript. Additionally, use the secure flag to ensure that cookies are transmitted over HTTPS connections only. This prevents attackers from stealing session cookies through XSS attacks, as these cookies become inaccessible to malicious scripts running in the browser. Proper cookie management is crucial to protecting user sessions from being hijacked.

5. Regular Security Audits and Penetration Testing:

Conduct regular security audits and penetration testing to identify and address potential vulnerabilities proactively. Security assessments can help developers discover and remediate XSS vulnerabilities, ensuring that the application remains secure over time. Regular testing also helps in staying ahead of evolving threats and attack techniques, enabling continuous improvement of security measures.(*Cross Site Scripting Prevention - OWASP Cheat Sheet Series*, n.d.)

REFERENCES.

- *2013-003: XSS vulnerability in LinkedIn | Internet Security Auditors*. (n.d.). Retrieved October 16, 2023, from https://isecauditors.com/2013-003
- *Cross Site Scripting Prevention—OWASP Cheat Sheet Series*. (n.d.). Retrieved October 16, 2023, from https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html
- *Cyber Threat Analysis: Types, Benefits, Tools, Approaches*. (n.d.). Retrieved October 16, 2023, from https://www.knowledgehut.com/blog/security/threat-analysis
- *What are Attack Vectors: Definition & Vulnerabilities - CrowdStrike*. (n.d.). Crowdstrike.Com. Retrieved October 16, 2023, from https://www.crowdstrike.com/cybersecurity-101/threat-intelligence/attack-vector/
- *What is an Attack Surface? | IBM*. (n.d.). Retrieved October 16, 2023, from https://www.ibm.com/topics/attack-surface